Methods for Vision-Based State Estimation and Online Motion Model Adaptation Using Multi-Modal Measurements and Motion Constraints

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von
Haolong Li
aus Shanxi, China

Tübingen
2024

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	10.02.2025
Dekan:	Prof. Dr. Thilo Stehle
1. Berichterstatter:	Prof. Dr. Jörg-Dieter Stückler
2. Berichterstatter:	Prof. Dr. Hendrik P. A. Lensch

We want AI agents that can discover like we can, not which contain what we have discovered.

– Richard S. Sutton

Abstract

To enable robots to operate autonomously in diverse environments, it is essential for them to accurately understand their own motion state and that of surrounding objects, including both position and orientation. Additionally, they need to understand the impact of their actions within environments. The state transition model, which describes how specific actions influence system responses, is essential as it enables the robots to plan new actions effectively. These elements, state estimation and model learning, are two important aspects of robotics. In this thesis, we begin by exploring the state estimation problem, utilizing vision-based measurements along with various other data modalities. We incorporate motion prior knowledge to constrain the state estimation, aiming to enhance both accuracy and robustness. Additionally, we delve into the model adaptation issue, where we adapt the motion model by calibrating its parameters online, in conjunction with motion state estimation.

We explore the state estimation problem from two perspectives: object motion estimation and ego motion estimation. For object motion estimation, we develop a tracking method using event camera, which captures high temporal resolution events measuring intensity changes at the pixel level. These cameras have high dynamic range and are more robust against motion blur than regular cameras. We integrate high-rate event data with lower-rate image data from a traditional camera using cubic B-splines. The spline represents the motion trajectory in a continuous form, facilitating the fusion of asynchronous measurements and providing a smoothness constraint for motion estimation. For ego motion estimation, we introduce a method specifically designed for legged robots, aiming to achieve accurate, high-rate, and low-drift estimations. We loosely couple visual-inertial-based estimates, which have low drift, with high-rate leg odometry estimates, and incorporate height measurements from leg kinematics to address the height drift issue in dynamic movements.

We further propose not only using kinematic or dynamic motion models of robots as motion constraints but also calibrating their parameters jointly with state estimation. We explore a velocity-control based kinematic model that translates linear and angular velocity control inputs into 2D relative motion. Recognizing that the most recent control inputs do not always accurately reflect the robot's actual movements, we calculate a weighted average from a local window of control inputs as an effective control, using a simple yet effective parametric model. The parameters of this model are adapted online in conjunction with motion state adjustments. Furthermore, we investigate a more complex dynamic model, formulated as a system of ordinary differential equations, and integrate it to compute relative motion predictions that serve as motion constraints for state estimation. We also calibrate the parameters of the dynamic model online, enabling it to adapt to changes in the environment and enhance its prediction accuracy.

Our dual approach to state estimation and model adaptation not only enhances the

accuracy and robustness of state estimation but also enables online forward prediction that adapts to changes in the environment. We anticipate that this capability of joint state estimation and model adaptation will be particularly beneficial for downstream tasks in control and planning, where accurate knowledge of both the state and the underlying model is required by the robot.

Zusammenfassung

Um Robotern das autonome Operieren in verschiedenen Umgebungen zu ermöglichen, ist es essenziell, dass sie ihren Bewegungszustand, einschließlich Position und Orientierung, genau erfassen. Zusätzlich müssen sie den Einfluss ihrer Aktionen innerhalb der Umgebungen verstehen. Das Zustandsübergangsmodell, das beschreibt, wie spezifische Aktionen die Systemantworten beeinflussen, ist wesentlich, da es den Robotern ermöglicht, neue Aktionen effektiv zu planen. Diese Elemente, Zustandsschätzung und Modelllernen, sind zwei wichtige Aspekte der Robotik. In dieser Arbeit untersuchen wir zunächst das Problem der Zustandsschätzung, wobei wir bildbasierte Messungen zusammen mit verschiedenen anderen Datenmodalitäten nutzen. Wir integrieren Vorwissen über Bewegungen, um die Zustandsschätzung zu beschränken, mit dem Ziel, sowohl die Genauigkeit als auch die Robustheit zu erhöhen. Darüber hinaus befassen wir uns mit dem Problem der Modellanpassung, bei dem wir das Bewegungsmodell durch Online-Kalibrierung seiner Parameter anpassen, in Verbindung mit der Schätzung des Bewegungszustands.

Wir betrachten das Problem der Zustandsschätzung aus zwei Perspektiven: die Schätzung der Objektbewegung und die Ego-Bewegungsschätzung. Für die Schätzung der Objektbewegung entwickeln wir eine Tracking-Methode unter Verwendung einer Event-Kamera, die Ereignisse hoher zeitlicher Auflösung aufzeichnet, welche Intensitätsänderungen auf Pixelebene messen. Diese Kameras verfügen über einen hohen Dynamikbereich und leiden nicht unter Bewegungsunschärfe. Wir integrieren Daten hoher Ereignisrate mit Bildern niedrigerer Rate von einer traditionellen Kamera unter Verwendung eines kubischen B-Splines. Der Spline stellt die Bewegungsbahn kontinuierlich dar, erleichtert die Fusion asynchroner Messungen und bietet eine Glättungsbeschränkung für die Bewegungsschätzung. Für die Ego-Bewegungsschätzung führen wir eine speziell für Laufroboter entwickelte Methode ein, die auf genaue, hochfrequente und driftarme Schätzungen abzielt. Wir koppeln visuell-inertiale Schätzungen, die eine geringe Drift aufweisen, lose mit hochfrequenten Beinodometrieschätzungen und integrieren Höhenmessungen aus der Beinkinematik, um Höhendrift bei dynamischen Bewegungen zu reduzieren.

Darüber hinaus schlagen wir vor, kinematische oder dynamische Bewegungsmodelle von Robotern nicht nur als Bewegungsbeschränkungen zu verwenden, sondern auch ihre Parameter gemeinsam mit der Zustandsschätzung zu kalibrieren. Wir untersuchen ein geschwindigkeitsgesteuertes kinematisches Modell, das lineare und Winkelgeschwindigkeitssteuerungseingaben in eine relative 2D-Bewegung übersetzt. Da Steuerungseingaben nicht immer die tatsächlichen Bewegungen des Roboters genau widerspiegeln, berechnen wir einen gewichteten Durchschnitt aus einem lokalen Fenster von Steuerungseingaben als effektive Steuerung, unter Verwendung eines einfachen, aber effektiven parametrischen Modells. Die Parameter dieses Modells werden online mit der Bewegungschätzung angepasst. Weiterhin untersuchen wir ein komplexeres

dynamisches Modell, formuliert als System gewöhnlicher Differentialgleichungen, und integrieren es, um relative Bewegungsvorhersagen zu berechnen, die als Bewegungsbeschränkungen für die Zustandsschätzung dienen. Wir kalibrieren auch die Parameter des dynamischen Modells online, um es an Veränderungen in der Umgebung anzupassen und seine Vorhersagegenauigkeit zu erhöhen.

Unser dualer Ansatz zur Zustandsschätzung und Modellanpassung verbessert nicht nur die Genauigkeit und Robustheit der Zustandsschätzung, sondern ermöglicht auch eine Online-Vorhersage, die sich an Veränderungen in der Umgebung anpasst. Diese Fähigkeit zur gemeinsamen Zustandsschätzung und Modellanpassung könnte in zukünftigen Forschungsarbeiten vorteilhaft für nachgelagerte Aufgaben in Steuerung und Planung sein, für die genaue Kenntnisse sowohl des Zustands als auch des zugrundeliegenden Modells durch den Roboter erforderlich sind.

Acknowledgement

First and foremost, I would like to thank my supervisor Prof. Dr. Jörg Stückler for providing invaluable guidance and support throughout my PhD journey. With your help, I have been able to enter this research field and carry out this work. I also wish to express my gratitude to my thesis advisory committee Prof. Dr. Jörg Stückler, Prof. Dr. Hendrik Lensch and Prof. Dr. Martin Butz for the insightful discussions and constructive feedback during our yearly meetings. I want to thank Prof. Dr. Jörg Stückler and Prof. Dr. Hendrik Lensch for their efforts in reviewing this thesis. I also want to thank Felix Grüninger for building the amazing robots for my research projects.

I am very grateful to all my collaborators for their help with one of the projects presented in this thesis. I appreciate the help by Felix Grimminger and Dr. Majid Khadiv for their expertise and insightful advice on quadruped robots. I am also thankful to the students who have worked with me, especially Yuxuan Xue, Victor Dhédin, and Lukas Mack, who have closely collaborated with me and achieved outstanding results.

Many thanks to all members of the Embodied Vision group and my colleagues at the Max Planck Institute in Tübingen. The working atmosphere and the time spent together were truly enjoyable.

I would also like to thank my friend Dr. Huanbo Sun for sharing the times in the empty building during the Covid-19 pandemic. Thank you to Hongwei Yi, Tim Xiao, and Alex Fan for the inspiring discussions, enjoyable trips, and moments of relaxation throughout my PhD.

Finally, my deepest gratitude goes to my parents, who encouraged me to pursue a PhD and have supported me every step of the way.

As time passes, I might forget the long nights spent in front of the computer, but the thrill of exploring the unknown and coming up with new ideas will always stay with me.

Haolong Li
Tübingen, July 2024

Contents

List of Figures

List of Tables

Notation

General Notation

$\mathbb{R}$	Set of real numbers
x, X, α	Scalar
$\mathbf{x}, \boldsymbol{\theta}$	Column vector
$\boldsymbol{X}$	Matrix
$(\mathbf{x})_i$	i^{th} entry of vector $\mathbf{x}$ (1-based indexing)
$\boldsymbol{X}^\top$	Transpose of matrix
$\boldsymbol{X}^{-1}$	Inverse of matrix
$\mathcal{X}, \Omega$	Set, e.g. $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$
$\mathbf{I}$	Identity matrix
$\mathbf{0}$	All-zero vector or matrix
$\mathbf{1}$	All-one vector or matrix
$f(\cdot)$	Function
$\dfrac{\partial f(x,y)}{\partial x}$	Partial derivative of the function $f(x, y)$ with respect to the variable x
$\|\cdot\|_2$	Euclidean norm (L^2 norm)

Lie Group Notation

$\mathrm{SO}(n)$	Special orthogonal group
$\mathfrak{so}(n)$	Lie algebra associated with $\mathrm{SO}(n)$
$\mathrm{SE}(n)$	Special Euclidean group
$\mathfrak{se}(n)$	Lie algebra associated with $\mathrm{SE}(n)$
$\hat{(\cdot)}$	Hat operation associated with the Lie algebra $\mathfrak{so}(n)$ or $\mathfrak{se}(n)$
$\check{(\cdot)}$	Inverse hat operation
$\mathbf{R}_t, \mathbf{T}_t$	Orientation $\mathbf{R}_t \in \mathrm{SO}(3)$ and pose $\mathbf{T}_t \in \mathrm{SE}(3)$ at time $t \in \mathbb{R}$
${}^{\mathrm{a}}\mathbf{R}_{\mathrm{b}}$	Rotation matrix ${}^{\mathrm{a}}\mathbf{R}_{\mathrm{b}} \in \mathrm{SO}(3)$ that transforms a vector from frame b to frame a
${}^{\mathrm{a}}\mathbf{T}_{\mathrm{b}}$	Transformation matrix ${}^{\mathrm{a}}\mathbf{T}_{\mathrm{b}} \in \mathrm{SE}(3)$ that transforms a vector from frame b to frame a

Probability distributions

$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	Random variable $\mathbf{x}$ follows a Gaussian distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$
$p(\mathbf{x})$	Probability density function of $\mathbf{x}$
$p(\mathbf{x}\|\boldsymbol{\theta})$	Probability density function of $\mathbf{x}$ conditioned on parameter vector $\boldsymbol{\theta}$

Acronyms

ATE	Absolute trajectory error (see Section 3.4)
DoF	Degrees of Freedom (see Section 3.1)
EKF	Extended Kalman filter (see Section 4.1)
IMU	Inertial Measurement Unit (see Section 2.3)
MAP	Maximum a posteriori (see Section 2.2)
MLE	Maximum likelihood estimation (see Section 2.2)
MPC	Model-predictive control (see Section 4.3
ODE	Ordinary differential equation (see Subsection 1.1.3)
RBF	Radial basis function (see Subsection 5.3.3)
RMSE	Root mean square error (see Section 3.4)
RPE	Relative pose error (see Section 3.4)
SLAM	Simultaneous localization and mapping (see Section 1.1)
VIO	Visual-inertial odometry (see Section 1.1)

Overview 1

1.1 Introduction

The development of autonomous robots, capable of undertaking repetitive and time-consuming tasks such as household chores, warehouse logistics, and autonomous driving, has been a long-standing and extensively researched objective. Accurate state estimation is crucial for these robots to interact effectively with their environment. The state comprises the information available to an agent about its environment and serves as the input signal for the decision-making process (Sutton and Barto, 2018). For example, autonomous navigation requires that a robot continuously updates its position and orientation relative to a 3D map. Similarly, effective grasping depends on the robot's understanding of the relative pose between objects and its endeffector. In this thesis, we focus on estimating motion states, such as position, orientation, and velocity, from sensor measurements of the environment.

Among these sensors, cameras serve as vital exteroceptive devices, capturing rich visual observations of the surroundings. They are favored for their lightweight and cost-effectiveness. The problems of motion estimation and 3D map reconstruction from image data have been long studied in the field of computer vision. Notable techniques include visual odometry, which tracks incremental camera motion from an image stream, and simultaneous localization and mapping (SLAM), which integrates incremental camera motion tracking with 3D map construction.

However, relying solely on visual cues for state estimation is problematic. Firstly, the task can be inherently ill-posed. For example, in techniques such as visual odometry and SLAM, the global pose and scale cannot be determined from monocular images alone. Additionally, hardware limitations can impair performance as images may become blurry in high-speed, low-light conditions.

To address these limitations, integrating complementary modalities is both desirable and necessary (Barfoot, 2017). For instance, the visual-inertial odometry (VIO) method combines cameras with an interoceptive sensor, namely the Inertial Measurement Unit (IMU), to track camera motion. The IMU captures angular velocity and linear acceleration at high frequencies, making scale and roll-pitch orientation observable (Hesch et al., 2014). While high-rate IMU measurements offer robust short-term motion data, they are prone to drift over time. Cameras, however, provide rich environmental details that can assist in correcting this drift.

Another promising approach involves the use of event cameras, which provide a distinct modality by detecting changes in pixel intensity and delivering data streams at microsecond rates (Gallego et al., 2020). These cameras offer a high dynamic range and are not susceptible to blur, distinguishing them from traditional frame-based cameras.

However, because they capture only changes in visual signals, event cameras have limitations in fully encoding the scene context, unlike standard cameras that measure absolute intensity frames.

While multi-modal data integration is beneficial for state estimation, it is not always sufficient. For instance, visual-inertial odometry may degrade under certain motions (Wu et al., 2017). An orthogonal approach to using multi-modal measurements involves utilizing prior knowledge of the motion state to constrain the state estimation problem. For example, with ground wheeled robots, it is understood that they move on a plane, allowing us to apply plane motion constraints. More broadly, we can incorporate the robot's physical properties, the kinematics and dynamics, as fundamental elements of the motion constraints. The first goal of this thesis is to investigate various motion constraints for vision-based state estimation using multi-modal data, aiming to enhance tracking accuracy. In addition to estimating their motion, robots also need to understand the impact of their control actions for navigation and planning. Therefore, the second focus of this thesis is to calibrate models of robots' kinematics and dynamics online. This calibration enables real-time adjustments to the models as the robot interacts with varying environments, facilitating their use for accurate forward motion prediction.

In summary, this thesis investigates and addresses the following two primary challenges:

1. Enhancing motion state estimation by incorporating motion constraints and utilizing multi-modal data, investigated from two different aspects: event-based object motion estimation and camera-IMU-based ego-motion estimation for both legged and wheeled robots.

2. Adapting kinematic and dynamic motion models online for wheeled robots to improve their motion prediction capabilities.

In the following sections, we review related works addressing these two challenges.

1.1.1 Event-Based Object Motion Estimation

Object motion estimation is a critical task in computer vision and plays a vital role in many robotic applications, such as manipulation and obstacle avoidance. Traditional methods (Choi and Christensen, 2012a,b) involve extracting features or patterns from images and estimating object motion by matching these with features or patterns from the 3D model of the object. Meanwhile, learning-based methods like Deng et al. (2021) and Y. Li et al. (2018) have demonstrated great performance by utilizing neural networks to extract more expressive image features for matching. More recently, model-free methods have gained increasing attention. These methods estimate object pose based on neural field representations without relying on 3D object models (Wen et al., 2023, 2024). Motion priors, such as constant velocity with first-order state dynamics (Choi and Christensen, 2012b; Deng et al., 2021; Krull et al., 2014) or zero velocity with specific

noise models (Azad et al., 2011; Klein and Murray, 2006; Teulière et al., 2010), are often used to propagate the estimated pose.

While these methods based on conventional image or depth data achieve great results, their performance can be limited by the hardware constraints of the cameras, such as low dynamic range, high latency, and motion blur. Event cameras, on the other hand, measure relative intensity changes at very high frequencies and offer complementary properties. Using an event camera to track object pose has the potential to address the limitations of conventional cameras.

Research on motion estimation with event cameras majorly focuses on ego motion estimation. H. Kim et al. (2016) developed a method utilizing three separate probabilistic filters to estimate camera motion, log intensity gradient, and inverse depth. Their method leverages an event generative model that predicts intensity changes based on estimated motion and depth. By minimizing the discrepancy between these predictions and actual measurements, the method effectively optimizes the state variables. A similar approach by Bryner et al. (2019) tracks camera motion relative to a static background, with a variant of the generative event measurement model. Mueggler et al. (2018) integrate image, IMU, and event data for camera motion estimation, assuming a known scene map. They employ a continuous trajectory representation based on splines to efficiently fuse these asynchronous multi-modal measurements.

While few studies (Mitrokhin et al., 2019a; Vasco et al., 2017) have focused on image-level object tracking, there is only a limited number that addresses the challenge of tracking objects in 3D space using event cameras. Dubeau et al. (2020) propose a method that tracks object pose in 3D using a textured 3D model, combining image, depth, and event data in a cascaded fashion. The method augments a learning-based object tracker for image and depth data with events. A neural network uses accumulated events to predict a coarse pose, which is then refined by the original object tracker. Z. Li et al. (2023) also augment a 3D object pose tracker for image and depth data with events. Their approach employs a dual Kalman filter strategy: an inner loop estimates object velocity using event observations based on a dampened motion model, while the outer loop fuses this velocity estimate with pose estimates derived from a deep neural network. In Chapter 3, we address the challenge of event-based object motion estimation by utilizing image and event data, assuming a textureless 3D model of the object is provided. We employ splines to represent the object trajectory, which implicitly provides smooth motion constraints for the estimation process.

1.1.2 Camera-IMU-Based Ego Motion Estimation

Ego motion estimation from camera and IMU measurements is a well-established field in computer vision and robotics. There are two categories depending on how the IMU measurements are integrated with images. The first category, known as the loosely coupled method, utilizes IMU data to propagate the state, while image data is used to update the state through a Kalman filter (Geneva et al., 2020; Mourikis and Roumeliotis,

2007). The second category, referred to as the tightly coupled method, estimates motion using a factor graph optimization framework, with vision and IMU measurements serving as factors (Leutenegger et al., 2015; Qin et al., 2018; Usenko et al., 2020). The tightly coupled approach is better constrained and results in more accurate and robust estimates, albeit at the cost of relatively high computational demands. For methods from both categories, the scale and the roll-pitch orientation with respect to gravity are observable, while the global position and yaw orientation remain unobservable (Hesch et al., 2014). To address this unobservability, we can define the local initial position and yaw orientation as references, thereby facilitating the estimation of ego motion.

Wu et al. (2017) further demonstrate that under specific motions, both the scale and global orientation can become unobservable. The scale turns unobservable if the local acceleration remains constant, and global orientation becomes unobservable in the absence of rotational motion. To address the issue of scale unobservability, they propose utilizing the wheel encoders and computing relative motion with forward kinematics based on wheel encoders measurements to constrain the motion estimation for a differential drive robot. F. Ma et al. (2019) extend this idea with an Ackerman drive robot. Similarly, Lee et al. (2020) also incorporate wheel odometry and they calibrate the wheel odometry parameters online to further improve the estimation accuracy. Many studies utilize wheel odometry as motion constraints for wheeled robots. However, this approach relies on wheel encoders. We instead employ velocity-control-based kinematics to compute relative motion serving as a constraint for the motion estimation problem, as detailed in Chapter 5.

For legged robots such as quadruped and humanoid robots, leg odometry is commonly used as a motion constraint. This method estimates the incremental motion of the robot's base using the forward kinematics of the legs that are in stable contact with the ground, with the assumption that the feet in contact are static. Early work (Blösch et al., 2012; Hartley et al., 2020, 2018c; J.-H. Kim et al., 2021) fuse leg odometry with IMU data and estimate robot base velocity and roll-pitch orientation. Chilian et al. (2011) fuse IMU measurements with leg odometry and visual odometry to enhance robustness. Similarly, Teng et al. (2021) integrate inertial and velocity measurements from a tracking camera with leg kinematic constraints and additionally perform observability analysis for the state parameters. Hartley et al. (2018b) extend an optimization-based visual-inertial state estimation framework by incorporating a forward kinematic factor. This addition is based on hybrid contact preintegration, which utilizes leg joint encoder and contact measurements. Y. Kim et al. (2022) propose an optimization-based approach that tightly integrates vision, IMU measurements, and preintegrated leg velocity factors without relying on the static contact assumption. In Chapter 4, we will also explore the use of forward kinematics as a motion constraint, aiming to develop a low-latency, high-update-rate motion estimator suitable for agile motion control.

1.1.3 Motion Model Adaptation

For model-based control and planning, understanding the impact of control actions on the robot through the control-based motion model is essential. Research into formulating and estimating motion models for ground-wheeled robots has been a long-standing area of interest. Typically, these models are derived from the physical properties of the robot and expressed as ordinary differential equation (ODE) systems. In a kinematic model, the ODE system directly relates the robot's velocity or acceleration to control inputs. In contrast, a dynamic model incorporates external forces and moments to model the acceleration, providing a more comprehensive and accurate representation. The parameters of the ODE system determine the model's behavior and can be optimized from data.

Many previous studies focus on identifying and adapting parameters online to achieve better control assuming motion states can be directly or partially measured. S. H. You et al. (2009) estimate sideslip and tire stiffness for a single-track dynamic model from steering angle, lateral acceleration, and yaw rate measurements using an extended Kalman filter. Similarly, Reina et al. (2017) employ an Extended Kalman filter to estimate slip angle and mass based on steering angle, throttle inputs, and measurements from gyroscopes and linear accelerometers. Wielitzka et al. (2015) use an Unscented Kalman filter to adapt parameters of a double-track dynamic model, incorporating vehicle states like side-slip angle, with data from a GPS-gyroscope measurement system. A similar approach from C. You and Tsiotras (2017) estimates parameters of both a single-track model and an extended double-track model that separately models sprung and unsprung mass, using known motion states.

Recently, learning-based approaches have been proposed to regress the non-linear vehicle dynamics from input states and actions. These data-driven methods train the model offline and rely on the assumption that the training data adequately cover the state and parameter distributions. Spielberg et al. (2019) develop a two-layer feed-forward neural network to learn one-step transitions of yaw rate and lateral velocity using state and steering control input history. Xu et al. (2019) propose a learning-based approach, training multi-layer perceptrons and recurrent neural networks to model vehicle dynamics with large-scale data. Similarly, Hermansdorfer et al. (2020) utilize recurrent neural networks to model vehicle dynamics, and demonstrate that their learned model can effectively replace traditional single-track model for vehicle dynamics simulation. However, they note that the model's accuracy depends on proper data selection and the availability of the training data. Additionally, T. Kim et al. (2022) propose a hybrid model that uses a single-track model to represent the dynamics, while training a neural network to predict parameters for an analytical tire model.

In Chapter 5, we integrate a kinematic motion model into the visual-inertial state estimator. Then, in Chapter 6, we explore a more complex and expressive dynamic motion model and incorporate it into the visual-inertial state estimator. These models serve not only as motion constraints to enhance motion tracking accuracy but also have

their parameters calibrated online. This joint calibration with state estimation enables accurate forward motion prediction.

1.2 Contributions

The contributions of this thesis, in connection to the research challenges outlined earlier, are summarized below.

1. In Chapter 3, we address the first challenge by focusing on object pose estimation using event cameras, which offer high temporal resolution and minimal motion blur. However, they only capture changes in intensity, and each individual event contains limited information. To overcome this, our method effectively combines measurements from both event- and frame-based cameras. We utilize a probabilistic generative event measurement model to track object motion from high-rate events. In a second layer, we refine the object trajectory using slower rate image frames through direct image alignment. To represent the motion trajectory, we employ splines, which not only facilitate the effective combination of asynchronous, high-rate event data with low-rate image data but also provide a smooth motion constraint for the noisy, low-information event data.

2. Apart from object pose estimation, Chapter 4 explores the ego motion estimation for dynamic locomotion of legged robots, focusing on reliable and high-rate estimation of robot posture, especially the base height. We develop a method that loosely couples VIO with leg odometry, utilizing visual and inertial measurements along with robot kinematics. The VIO module, which includes a stereo camera and an IMU, provides low-drift 3D position and yaw orientation, as well as drift-free pitch and roll orientations of the robot's base link within the inertial frame. However, the considerable latency due to image processing and optimization, combined with a relatively low update rate, makes these measurements less suited for low-level control. To mitigate this issue, we enhance the frequency of the VIO state estimates to match the high-rate IMU measurements within the VIO sensor. We then fuse the enhanced base pose and linear velocity estimates from the VIO with a second high-rate IMU and leg odometry measurements, resulting in robot state estimates with high frequency and minimal latency, suitable for dynamic control. Additionally, because VIO does not provide an absolute reference to the ground plane and can drift in height over time, we utilize contact detection and leg kinematics to obtain accurate height measurements, compensating for the height drift.

3. From Chapter 5, we address the second challenge by utilizing the robot's physical properties not only as motion constraints to enhance motion state estimation but also to identify and calibrate the motion model, enabling accurate motion prediction. In Chapter 5, we present an approach for VIO with a stereo camera that integrates and calibrates the velocity-control-based kinematic motion model

of wheeled mobile robots. We use a radial basis function kernel to compensate for time delays and deviations between control commands and actual robot motion. The motion model is calibrated online by the VIO system and can potentially be used as a forward model for motion control and planning.

4. Chapter 6 extends the concept of state estimation and motion model calibration under motion constraints by replacing the kinematic motion model with a dynamic motion model. The dynamic model accounts for robot-environment interactions, unlike its kinematic counterpart. We tightly fuse a single-track dynamic model for wheeled ground robots with VIO. Our approach continuously calibrates and adapts the dynamic model online, thereby enhancing the accuracy of forward predictions based on anticipated control inputs. Formulated through ordinary differential equations, the single-track dynamic model approximates the motion of wheeled vehicles under specific control inputs on flat ground. We employ a singularity-free and differentiable variant of the single-track model, which facilitates seamless integration as a dynamic factor within the VIO framework and allows for the online optimization of model parameters alongside VIO state variables.

1.3 Publications

1.3.1 Main Publications

Large parts of this thesis were published in peer-reviewed conference proceedings and journals. Each of the chapters mentioned above (Chapters 3 to 6) is drawn from one of the publications listed below:

(i) **H. Li** and J. Stückler (2021). 'Tracking 6-DoF Object Motion from Events and Frames'. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA48506.2021.9561760

(ii) V. Dhédin, **H. Li**, S. Khorshidi, L. Mack, A. K. C. Ravi, A. Meduri, P. Shah, F. Grimminger, L. Righetti, M. Khadiv, and J. Stückler (2023). 'Visual-Inertial and Leg Odometry Fusion for Dynamic Locomotion'. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA48891.2023.10160898

(iii) **H. Li** and J. Stückler (2022). 'Visual-Inertial Odometry with Online Calibration of Velocity-Control Based Kinematic Motion Models'. In: *IEEE Robotics and Automation Letters (RA-L)*. DOI: 10.1109/LRA.2022.3169837

(iv) **H. Li** and J. Stückler (2024). 'Online Calibration of a Single-Track Ground Vehicle Dynamics Model by Tight Fusion with Visual-Inertial Odometry'. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. To appear, preprint available at https://doi.org/10.48550/arXiv.2309.11148

1.3.2 Side Publications

During my PhD studies, I have contributed to the following works, which are not included in this thesis:

- Y. Xue, **H. Li**, S. Leutenegger, and J. Stückler (2022). 'Event-Based Non-Rigid Reconstruction from Contours'. In: *Proceedings of British Machine Vision Conference (BMVC)*. Available at https://bmvc2022.mpi-inf.mpg.de/0078.pdf
- S. Guttikonda, J. Achterhold, **H. Li**, J. Boedecker, and J. Stückler (2023). 'Context-Conditional Navigation with a Learning-Based Terrain- and Robot-Aware Dynamics Model'. In: *Proceedings of European Conference on Mobile Robots (ECMR)*. DOI: 10.1109/ECMR59166.2023.10256414
- Y. Xue, **H. Li**, S. Leutenegger, and J. Stückler (2024). 'Event-Based Non-Rigid Reconstruction of Low-Rank Parametrized Deformations from Contours'. In: *International Journal of Computer Vision (IJCV)*. DOI: 10.1007/s11263-024-02011-z

1.4 Open-Source Software Release

The source code of the approach ST-VIO presented in Chapter 6 is published open-source at https://github.com/EmbodiedVision/st-vio.

Background 2

In this chapter, we provide the theoretical background and mathematical tools necessary for this thesis. We begin by introducing the concept of rigid body motion and the pose state representation. Following this, we describe the inference framework and optimization methods employed in this thesis. Lastly, we provide an overview of the visual-inertial odometry method.

2.1 Motion Representations

In this thesis, state estimation pertains to rigid body motions, which describe the orientation and position of a rigid body in three-dimensional physical space (Lynch and Park, 2017).

2.1.1 Rotational Motions

The orientation of a frame o relative to another frame w can be expressed through a rotation matrix ${}^{w}\mathbf{R}_{o} \in \mathbb{R}^{3\times3}$. The set of such rotation matrices constitutes the *special orthogonal group* SO(3) under the matrix multiplication operation. By definition, a group is a set equipped with an operation with the following properties:

- closure: $\mathbf{R}_1\mathbf{R}_2 \in \mathrm{SO}(3), \forall \mathbf{R}_1, \mathbf{R}_2 \in \mathrm{SO}(3)$,
- associativity: $(\mathbf{R}_1\mathbf{R}_2)\mathbf{R}_3 = \mathbf{R}_1(\mathbf{R}_2\mathbf{R}_3), \forall \mathbf{R}_1, \mathbf{R}_2, \mathbf{R}_3 \in \mathrm{SO}(3)$,
- identity element $\mathbf{I}$: $\mathbf{I}\mathbf{R} = \mathbf{R}\mathbf{I} = \mathbf{R}, \forall \mathbf{R} \in \mathrm{SO}(3)$,
- inverse element $\mathbf{R}^{-1}$: $\mathbf{R}^{-1}\mathbf{R} = \mathbf{I}, \forall \mathbf{R} \in \mathrm{SO}(3)$.

Additionally, matrices in SO(3) satisfy the property $\mathbf{R}^{\top} = \mathbf{R}^{-1}$, indicating that each matrix is orthogonal. Based on the SO(3) group properties, we can compute the orientation of frame w in frame o as:

$$ {}^{o}\mathbf{R}_{w} = {}^{w}\mathbf{R}_{o}^{\top}. \tag{2.1} $$

The orientation of one frame relative to another can also be determined through composition. For instance, given a rotation matrix ${}^{w}\mathbf{R}_{o}$ describing the orientation of frame o in frame w and a rotation matrix ${}^{o}\mathbf{R}_{i}$ describing the orientation of frame i in frame o, the orientation of frame i in frame w can be formulated as:

$$ {}^{w}\mathbf{R}_{i} = {}^{w}\mathbf{R}_{o}\,{}^{o}\mathbf{R}_{i}. \tag{2.2} $$

We can also use the rotation matrix to rotate or change the reference frame of a vector. Specifically, for a vector ${}^{o}\mathbf{t} \in \mathbb{R}^3$ in reference frame o, its transformation to the reference frame w is given by:

$$ {}^{w}\mathbf{t} = {}^{w}\mathbf{R}_{o}\,{}^{o}\mathbf{t}. \tag{2.3} $$

The length of vector $\mathbf{t}$ remains unchanged after the transformation. Let us now define frame w as the fixed world frame and frame o as the body frame. Consequently, the orientation of the body frame relative to the world frame at time t can be denoted as ${}^{w}\mathbf{R}_{o,t}$. For brevity, we will use $\mathbf{R}$ to represent this term throughout this chapter. The matrix $\dot{\mathbf{R}}$ is the time rate of its change and satisfies:

$$ \dot{\mathbf{R}} = \mathbf{R}\,{}^{o}\hat{\boldsymbol{\omega}}, \tag{2.4} $$

where ${}^{o}\hat{\boldsymbol{\omega}}$ is the *skew-symmetric* matrix representation of the vector ${}^{o}\boldsymbol{\omega}$. This vector ${}^{o}\boldsymbol{\omega}$ is the angular velocity expressed in the body frame o and can be transformed to another frame i by:

$$ {}^{i}\boldsymbol{\omega} = {}^{i}\mathbf{R}_{o}\,{}^{o}\boldsymbol{\omega}. \tag{2.5} $$

For the sake of simplicity, the superscript of the angular velocity term will also be omitted within this chapter. Equation (2.4) can be viewed as a differential equation. By assuming constant angular velocity and $\mathbf{R}_{t=0} = \mathbf{I}$, we can compute the solution as:

$$ \mathbf{R} = \exp(\hat{\boldsymbol{\omega}} t). \tag{2.6} $$

As further shown in Lynch and Park (2017), for any rotation matrix $\mathbf{R} \in \mathrm{SO}(3)$, one can always find a vector $\boldsymbol{\omega} \in \mathbb{R}^3$ such that $\mathbf{R} = \exp(\hat{\boldsymbol{\omega}})$. Throughout thesis, we use $\mathrm{Exp}(\boldsymbol{\omega})$ as a shorthand notation for $\exp(\hat{\boldsymbol{\omega}})$. The skew-symmetric matrix $\hat{\boldsymbol{\omega}}$ represents the so-called exponential coordinates of the rotation matrix $\mathbf{R}$ and the set of the 3×3 skew-symmetric matrices is called $\mathfrak{so}(3)$, the *Lie algebra* of SO(3). We can also view $\hat{(\cdot)}$ as a mapping from $\mathbb{R}^3$ to $\mathfrak{so}(3)$:

$$ \hat{\boldsymbol{\omega}} = \widehat{\begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{pmatrix}} = \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix}, \tag{2.7} $$

and it is commonly called the *hat* operation. The exponential mapping can be used to map elements from $\mathfrak{so}(3)$ to SO(3), and its inverse, the logarithmic mapping, can be used to map from SO(3) to $\mathfrak{so}(3)$:

$$ \begin{aligned} \exp &: \hat{\boldsymbol{\omega}} \in \mathfrak{so}(3) \to \mathbf{R} \in \mathrm{SO}(3) \\ \log &: \mathbf{R} \in \mathrm{SO}(3) \to \hat{\boldsymbol{\omega}} \in \mathfrak{so}(3). \end{aligned} $$

The exponential mapping, also known as *Rodrigues' formula* is

$$ \exp(\hat{\boldsymbol{\omega}}) = \mathbf{I} + \sin(\|\boldsymbol{\omega}\|)\frac{\hat{\boldsymbol{\omega}}}{\|\boldsymbol{\omega}\|} + (1 - \cos(\|\boldsymbol{\omega}\|))\frac{\hat{\boldsymbol{\omega}}^2}{\|\boldsymbol{\omega}\|^2}, \tag{2.8} $$

and the logarithmic mapping is

$$\log(\mathbf{R}) = \frac{\theta(\mathbf{R} - \mathbf{R}^\top)}{2\sin\theta} \tag{2.9}$$

$$\theta = \arccos\frac{\operatorname{trace}(\mathbf{R}) - 1}{2}. \tag{2.10}$$

The logarithmic function returns an element of $\mathfrak{so}(3)$, namely a skew-symmetric matrix, we can further map it to $\mathbb{R}^3$ with the inverse hat operation $\widecheck{(\cdot)}$:

$$\widecheck{\begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix}} = \begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{pmatrix}. \tag{2.11}$$

We use $\operatorname{Log}(\mathbf{R})$ as a shorthand notation for $\widecheck{\log(\mathbf{R})}$.

2.1.2 Rigid Body Motions

The full rigid body motions in three-dimensional physical space can be represented by the combination of rotational and translational motions with the matrix form ${}^{\mathrm{w}}\mathbf{T}_{\mathrm{o}} \in \mathbb{R}^{4\times4}$:

$${}^{\mathrm{w}}\mathbf{T}_{\mathrm{o}} = \begin{pmatrix} {}^{\mathrm{w}}\mathbf{R}_{\mathrm{o}} & {}^{\mathrm{w}}\mathbf{t}_{\mathrm{o}} \\ \mathbf{0} & 1 \end{pmatrix}. \tag{2.12}$$

The rotation matrix ${}^{\mathrm{w}}\mathbf{R}_{\mathrm{o}} \in \mathrm{SO}(3)$ represents the orientation of frame o in frame w and the vector ${}^{\mathrm{w}}\mathbf{t}_{\mathrm{o}} \in \mathbb{R}^3$ represents the origin of frame o in frame w. Thus, we can use this matrix to describe the pose of frame o in frame w. Analogous to the rotation matrix, the matrices for the rigid body motion under the matrix multiplication operation also form a group called *special Euclidean group* SE(3). The inverse of ${}^{\mathrm{w}}\mathbf{T}_{\mathrm{o}}$ is:

$${}^{\mathrm{w}}\mathbf{T}_{\mathrm{o}}^{-1} = {}^{\mathrm{o}}\mathbf{T}_{\mathrm{w}} = \begin{pmatrix} {}^{\mathrm{w}}\mathbf{R}_{\mathrm{o}}^\top & -{}^{\mathrm{w}}\mathbf{R}_{\mathrm{o}}^\top {}^{\mathrm{w}}\mathbf{t}_{\mathrm{o}} \\ \mathbf{0} & 1 \end{pmatrix} \tag{2.13}$$

representing the pose of frame w in frame o. A vector ${}^{\mathrm{o}}\mathbf{t}$ can be transformed with ${}^{\mathrm{w}}\mathbf{T}_{\mathrm{o}}$ to frame w. By slightly abusing the notation and treating ${}^{\mathrm{w}}\mathbf{T}_{\mathrm{o}}$ as a mapping instead of a matrix, this transformation can be written as:

$${}^{\mathrm{w}}\mathbf{T}_{\mathrm{o}}({}^{\mathrm{o}}\mathbf{t}) = {}^{\mathrm{w}}\mathbf{t} = {}^{\mathrm{w}}\mathbf{R}_{\mathrm{o}}\,{}^{\mathrm{o}}\mathbf{t} + {}^{\mathrm{w}}\mathbf{t}_{\mathrm{o}}. \tag{2.14}$$

We can use ${}^{\mathrm{w}}\mathbf{T}_{\mathrm{o},t}$ to describe the pose of a moving body frame o to a fixed world frame w at time t. To reduce clutter in this discussion, we will denote this term as $\mathbf{T}$ for the remainder of this chapter. The time derivative $\dot{\mathbf{T}}$ can be computed with the following

equations:

$$\dot{\mathbf{T}} = \mathbf{T}\,{}^{\mathrm{o}}\hat{\boldsymbol{\xi}} \tag{2.15}$$

$${}^{\mathrm{o}}\hat{\boldsymbol{\xi}} = \begin{pmatrix} {}^{\mathrm{o}}\hat{\boldsymbol{\omega}} & {}^{\mathrm{o}}\mathbf{v} \\ \mathbf{0} & 0 \end{pmatrix}. \tag{2.16}$$

with ${}^{\mathrm{o}}\hat{\boldsymbol{\omega}} \in \mathfrak{so}(3)$ and ${}^{\mathrm{o}}\mathbf{v} \in \mathbb{R}^3$. The corresponding vector ${}^{\mathrm{o}}\boldsymbol{\xi} \in \mathbb{R}^6$ is called *body twist*:

$${}^{\mathrm{o}}\boldsymbol{\xi} = \begin{pmatrix} {}^{\mathrm{o}}\mathbf{v} \\ {}^{\mathrm{o}}\boldsymbol{\omega} \end{pmatrix} \tag{2.17}$$

representing the instantaneous linear and angular body velocity viewed in the body frame. The set of the 4×4 matrices $\hat{\boldsymbol{\xi}}$ is referred to as $\mathfrak{se}(3)$, the *Lie algebra* of SE(3). We can also use exponential mapping to map elements from $\mathfrak{se}(3)$ to SE(3) with the following equation:

$$\mathbf{T} = \exp(\hat{\boldsymbol{\xi}}) = \begin{pmatrix} \exp(\hat{\boldsymbol{\omega}}) & \mathbf{V}\mathbf{v} \\ 0 & 1 \end{pmatrix}, \tag{2.18}$$

where

$$\mathbf{V} = \mathbf{I} + \frac{1 - \cos(\|\boldsymbol{\omega}\|)}{\|\boldsymbol{\omega}\|^2}\hat{\boldsymbol{\omega}} + \frac{\|\boldsymbol{\omega}\| - \sin(\|\boldsymbol{\omega}\|)}{\|\boldsymbol{\omega}\|^3}\hat{\boldsymbol{\omega}}^2. \tag{2.19}$$

Its inverse mapping, the logarithmic mapping is:

$$\hat{\boldsymbol{\xi}} = \begin{pmatrix} \mathbf{V}^{-1}\mathbf{t} \\ \log(\mathbf{R}) \end{pmatrix}. \tag{2.20}$$

Again, we can use the inverse hat operation $\check{(\cdot)}$ to map the result of the logarithmic function to $\mathbb{R}^6$.

2.1.3 Derivative on SO(3) and SE(3)

The derivative of a function operating on group, either SO(3) or SE(3) in our case, can be defined in various ways. In this thesis, we use the definition from Deray and Solà (2020), which is sufficient and effective for the problem settings discussed. Formally, for a function f whose domain and codomain are either SO(3) or SE(3), its derivative is defined as:

$$\frac{\partial f(\mathbf{X})}{\partial \mathbf{X}} = \lim_{\boldsymbol{\tau} \to \mathbf{0}} \frac{f(\mathbf{X} \oplus \boldsymbol{\tau}) \ominus f(\mathbf{X})}{\boldsymbol{\tau}}, \tag{2.21}$$

with the special plus $\oplus$ and minus $\ominus$ defined as

$$\oplus : \mathbf{Y} = \mathbf{X} \oplus \boldsymbol{\tau} = \mathbf{X}\exp(\hat{\boldsymbol{\tau}}) \tag{2.22}$$

$$\ominus : \boldsymbol{\tau} = \mathbf{Y} \ominus \mathbf{X} = \widehat{\log(\mathbf{X}^{-1}\mathbf{Y})}. \tag{2.23}$$

The $\oplus$ operation enables the computation of the infinitesimal increment for the group elements, while the $\ominus$ operation allows us to compare the change in the function value induced by the infinitesimal increment. Based on this definition, we can compute closed-form Jacobians for the operations on SO(3) and SE(3) groups. For detailed derivations and formulae, we refer to Deray and Solà (2020).

2.2 Probabilistic Inference and Optimization

In this section, we introduce the probabilistic inference framework and the associated optimization methods used for the state estimation problem discussed in this thesis.

2.2.1 Maximum Likelihood Estimation

In robotics or computer vision, the challenge often lies in recovering unknown agent states from a series of noisy measurements, denoted as $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$. For instance, given a series of images, the objective might be to determine the camera poses at which these images were captured. It is generally assumed that these measurements are independent and each follows an unknown distribution $p_{\text{data}}(\mathbf{x})$.

Our goal is to approximate this unknown distribution with a probabilistic model $p(\mathbf{x}|\boldsymbol{\theta})$ parameterized by $\boldsymbol{\theta}$ to infer these unknown states. In order to estimate $\boldsymbol{\theta}$, we employ the maximum likelihood estimation (MLE) framework, which seeks to minimize the dissimilarity between the model distribution and the empirical data distribution defined by the samples (Goodfellow et al., 2016). The likelihood function of $\boldsymbol{\theta}$ is the probability density

$$L(\boldsymbol{\theta}) = p(\mathcal{X}|\boldsymbol{\theta}) = \prod_{i=1}^{n} p(\mathbf{x}_i|\boldsymbol{\theta}). \tag{2.24}$$

The maximum likelihood estimator is defined as:

$$\boldsymbol{\theta}^* = \arg\max L(\boldsymbol{\theta}). \tag{2.25}$$

In practice, it is more convenient to maximize the logarithm of the likelihood, as it turns the product into a sum:

$$\boldsymbol{\theta}^* = \arg\max \log L(\boldsymbol{\theta}) = \arg\max \sum_{i=1}^{n} \log p(\mathbf{x}_i|\boldsymbol{\theta}). \tag{2.26}$$

Under certain conditions, the result of the maximum likelihood estimator $\boldsymbol{\theta}^*$ approaches the true value asymptotically as the number of samples goes to infinity.

Assuming that the noisy measurement follows a Gaussian distribution $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, a link can be established between maximum log-likelihood estimation and the least

squares method. To demonstrate this, consider the probability density function of the Gaussian measurement model:

$$p(\mathbf{x}|\boldsymbol{\mu}(\boldsymbol{\theta}), \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^k \det(\boldsymbol{\Sigma})}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}(\boldsymbol{\theta}))^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu}(\boldsymbol{\theta}))\right), \tag{2.27}$$

where k represents the dimension of the measurements, $\boldsymbol{\mu}$ denotes the mean parameterized by $\boldsymbol{\theta}$, and $\boldsymbol{\Sigma}$ is the covariance matrix. In the context of state estimation addressed in this thesis, the goal is to estimate the parameter vector $\boldsymbol{\theta}$ that defines the mean, while treating the covariance matrix as a diagonal matrix with known scalar elements σ_i. Notably, minimizing the negative log-likelihood is equivalent to maximizing the log-likelihood itself. The optimal parameter vector $\boldsymbol{\theta}$ for Gaussian measurement noise is thus given by:

$$\boldsymbol{\theta}^* = \arg\min \sum_{i=1}^{n} \frac{1}{\sigma_i} \|\mathbf{x}_i - \boldsymbol{\mu}_i(\boldsymbol{\theta})\|_2^2. \tag{2.28}$$

This expression can also be viewed as a weighted least squares estimator. Consequently, we can utilize the well-established solving tools from least squares problems for our maximum log-likelihood estimation.

We can further determine the posterior distribution over the parameter vector $\boldsymbol{\theta}$ using Bayes' rule:

$$p(\boldsymbol{\theta}|\mathcal{X}) \propto p(\mathcal{X}|\boldsymbol{\theta}) p(\boldsymbol{\theta}), \tag{2.29}$$

where $p(\boldsymbol{\theta})$ is the prior distribution of the parameter. The method of maximum a posteriori (MAP) estimation then estimates the parameter $\boldsymbol{\theta}$ as:

$$\boldsymbol{\theta}^* = \arg\max p(\boldsymbol{\theta}|\mathcal{X}) = \arg\max \sum_{i=1}^{n} \log p(\mathbf{x}_i|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}). \tag{2.30}$$

Compared to MLE, MAP estimation leverages information from the prior that is not present in the training data (Goodfellow et al., 2016). If we assume the prior distribution $p(\boldsymbol{\theta})$ is uniform, then MAP estimation becomes equivalent to MLE. If we assume that the noise of the measurement is Gaussian and the prior distribution of the parameter follows a zero-mean Gaussian distribution, we obtain the regularized parameter estimate:

$$\boldsymbol{\theta}^* = \arg\min \sum_{i=1}^{n} \frac{1}{\sigma_i} \|\mathbf{x}_i - \boldsymbol{\mu}_i(\boldsymbol{\theta})\|_2^2 + \|\boldsymbol{\theta}\|_2^2, \tag{2.31}$$

which can also be solved using tools for least-squares problems.

2.2.2 Gauss-Newton Method

The least squares estimator is a widely utilized approach for estimating unknown parameters $\boldsymbol{\theta}$ based on noisy measurements (Nocedal and Wright, 2006). The objective

function E is defined as:

$$E(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i}^{n}\frac{1}{\sigma_i}r_i(\boldsymbol{\theta})^2 = \frac{1}{2}\mathbf{r}(\boldsymbol{\theta})^\top\boldsymbol{\Sigma}^{-1}\mathbf{r}(\boldsymbol{\theta}). \tag{2.32}$$

Here, $r(\boldsymbol{\theta}) = (x - \mu(\boldsymbol{\theta}))$ is a function mapping from $\mathbb{R}^p$ to $\mathbb{R}$, where p represents the dimension of the parameter vector $\boldsymbol{\theta}$. The weight factor $\frac{1}{\sigma}$ scales the function $\mathbf{r}$ up or down and constitutes the diagonal elements of the weight matrix $\boldsymbol{\Sigma}^{-1}$. This function quantifies the discrepancy between the measurement and the parameterized estimation, and is commonly referred to as the residual or error term. We assume that the number of measurements n exceeds the dimension of the parameter vector p, ensuring that the least squares problem is consistently overdetermined. The optimal parameters $\boldsymbol{\theta}^*$ are determined by minimizing the squared sum of the residuals:

$$\boldsymbol{\theta}^* = \arg\min E(\boldsymbol{\theta}). \tag{2.33}$$

This gives us the same estimator for the parameters as derived from a different perspective in the previous subsection.

For a linear model, the minimum of the objective function can be analytically determined by setting the derivative of the objective function with respect to $\boldsymbol{\theta}$ to zero. However, the inherent limitations of linearity restrict the model's ability to fully express complex systems. In this thesis, the state estimation tasks require non-linear models to effectively capture such complexities. For the non-linear least squares problem, numerical method can be employed to compute optimal solutions.

Gauss-Newton method is an iterative approach for solving non-linear least squares problems based on their linear approximation. In each iteration, the residual function r is approximated by the first-order Taylor expansion:

$$\mathbf{r}(\boldsymbol{\theta} + \delta\boldsymbol{\theta}) = \mathbf{r}(\boldsymbol{\theta}) + \mathbf{J}(\boldsymbol{\theta})\delta\boldsymbol{\theta}, \tag{2.34}$$

where $\delta\boldsymbol{\theta}$ is the update for the parameter vector in the current iteration and $\mathbf{J}$ is the Jacobian of r, defined as:

$$J_{ij} = \frac{\partial r_i(\boldsymbol{\theta})}{\partial\theta_j}. \tag{2.35}$$

By substituting this linear approximation into the objective function, the original non-linear least squares problem is transformed to a linear least squares problem with respect to $\delta\boldsymbol{\theta}$:

$$\begin{aligned} E(\delta\boldsymbol{\theta}) =& \frac{1}{2}\mathbf{r}(\boldsymbol{\theta} + \delta\boldsymbol{\theta})^\top\mathbf{r}(\boldsymbol{\theta} + \delta\boldsymbol{\theta}) \\ =& \frac{1}{2}\mathbf{r}(\boldsymbol{\theta})^\top\mathbf{r}(\boldsymbol{\theta}) + \delta\boldsymbol{\theta}^T\mathbf{J}(\theta)^T\mathbf{r}(\theta) + \frac{1}{2}\delta\boldsymbol{\theta}^T\mathbf{J}(\theta)^T\mathbf{J}(\theta)\delta\boldsymbol{\theta}. \end{aligned} \tag{2.36}$$

The minimum of $E(\delta\boldsymbol{\theta})$ can be calculated by setting its derivative to zero:

$$\mathbf{J}(\theta)^T\mathbf{J}(\theta)\delta\boldsymbol{\theta} = -\mathbf{J}(\theta)^T\mathbf{r}(\theta). \tag{2.37}$$

The optimal parameter update $\delta\boldsymbol{\theta}$ is derived by solving the above linear system:

$$\delta\boldsymbol{\theta}^* = -(\mathbf{J}(\theta)^T\mathbf{J}(\theta))^{-1}\mathbf{J}(\theta)^T\mathbf{r}(\theta). \tag{2.38}$$

The product of the Jacobians is the Hessian matrix of the linearized objective function $\mathbf{H} = \mathbf{J}(\theta)^T\mathbf{J}(\theta)$. The parameter vector is then updated based on the optimal $\delta\boldsymbol{\theta}$ for the current iteration $k+1$:

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k + \delta\boldsymbol{\theta}^*. \tag{2.39}$$

This process is repeated until a specified iteration number is reached or the parameter update $\delta\boldsymbol{\theta}$ becomes negligible.

If the parameter vector $\boldsymbol{\theta}$ belongs to the SO(3) or SE(3) group, the increment $\delta\boldsymbol{\theta}$ can be computed with Lie algebra, based on the special Jacobians for these groups. The parameter vector is then updated using the special $\oplus$ operator introduced in Subsection 2.1.3 as:

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k \oplus \delta\boldsymbol{\theta}^*. \tag{2.40}$$

The Gauss-Newton method convergence rate can be close to quadratic if $\mathbf{r}$ is small (Nocedal and Wright, 2006), emphasizing the importance of a good initial guess for the parameter vector. In subsequent chapters, we will demonstrate how the Gauss-Newton method is applied to various state estimation tasks.

2.3 Visual-Inertial Odometry

Visual-inertial odometry (VIO) has been an active research topic over the past few years. This section provides an overview of the optimization-based VIO method, which estimates pose and velocity states using image measurements from cameras combined with angular velocity and linear acceleration data from the Inertial Measurement Unit (IMU) for autonomous agents (Leutenegger et al., 2015; Qin et al., 2018). We begin by introducing the commonly used notation and definitions for VIO, followed by a description of the methods used for processing visual information and IMU measurements. Finally, we discuss the optimization framework that tightly couples these two different modalities.

2.3.1 Notation and Definition

By convention, VIO estimates the pose and linear velocity of the IMU coordinates frame (denoted with i). The pose state, which is the transformation from the IMU frame i to the fixed world frame w, is represented as ${}^{\mathrm{w}}\mathbf{T}_{\mathrm{i}} \in \mathrm{SE}(3)$. This transformation comprises a

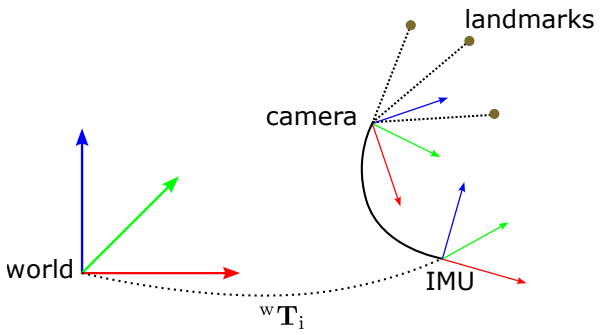


Figure 2.1: The frames involved in VIO are the fixed world frame, the IMU frame for which the pose and linear velocity are estimated, and the camera frame.

rotation component ${}^{w}\mathbf{R}_{i} \in \mathrm{SO}(3)$ and a translational component ${}^{w}\mathbf{t}_{i} \in \mathbb{R}^3$ (see Figure 2.1). The velocity state of frame i is the linear velocity ${}^{w}\mathbf{v} = {}^{w}\dot{\mathbf{t}}_{i}$, observed in the world frame w. To fuse information from the camera and the IMU, the relative pose ${}^{c}\mathbf{T}_{i}$ between them is also required. This relative pose is typically provided by the device manufacturer or can be measured separately. In this thesis, it is assumed to be known and constant. Additionally, VIO estimates the 3D landmarks $\mathcal{L} = \{\mathbf{l}_1 \dots \mathbf{l}_L\}$ from the surrounding environment based on the visual information extracted from images. Moreover, since the velocity and acceleration measurements from IMU can drift over time, two bias terms $\mathbf{b}_{g}$ and $\mathbf{b}_{a}$ are included to model this drift. The comprehensive state of the VIO at any given time t can thus be summarized as $\mathbf{s}_t = \left({}^{w}\mathbf{T}_{i,t}, {}^{w}\mathbf{v}_t, \mathcal{L}, \mathbf{b}_{g,t}, \mathbf{b}_{a,t}\right)$, and this state is estimated at the image frame rate.

2.3.2 Visual Measurements and Reprojection Error

The camera captures images over time, providing core information for localizing the robot in a 3D environment. To understand how image data can be used to estimate motion, we first need to introduce the basic concept of image formation.

Following the introduction in Y. Ma et al. (2003), we start with a point ${}^{c}\mathbf{p} = ({}^{c}x, {}^{c}y, {}^{c}z)^{\top}$ in 3D observed in the camera frame attached at the projection center. By convention, the camera frame's z-axis points forward and aligns with the optical axis. The x- and y-axes of the camera frame point to the right and downward, respectively. The z value of a point with respect to these coordinates is referred to as the depth. The projection π of such 3D point through an ideal pinhole camera onto the image plane is given by:

$$\bar{\mathbf{x}} = \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \pi({}^{c}\mathbf{p}) = \begin{pmatrix} f_{\mathbf{x}} & 0 & c_{\mathbf{x}} \\ 0 & f_{\mathbf{y}} & c_{\mathbf{y}} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} {}^{c}x/{}^{c}z \\ {}^{c}y/{}^{c}z \\ 1 \end{pmatrix}. \tag{2.41}$$

The horizontal and vertical focal lengths $f_{\mathbf{x}}$ and $f_{\mathbf{y}}$ scale the normalized 3D point. The horizontal and vertical shift terms $c_{\mathbf{x}}$ and $c_{\mathbf{y}}$ are used to translate the projection so that its position (u, v) is specified relative to the top-left corner of the image. These four

elements, known as the camera's intrinsic parameters, are assumed to be known in this thesis.

After introducing the image formation process, we now describe how we associate data across multiple images to infer the positions of 3D environment points and the camera poses when the images are captured. The data association process involves two key steps: feature detection and feature tracking.

In the feature detection step, feature points are extracted from images using a corner detection algorithm such as FAST (Rosten et al., 2010), which identifies feature points based on the values of the image. These feature points typically correspond to corner points that contain high-texture information within the scene. Once the feature points are detected, we assign a depth parameter to each of them to construct the landmarks $\mathcal{L} = \{\mathbf{l}_1 \dots \mathbf{l}_L | \mathbf{l} = (u, v, d)^\top\}$, where (u, v) are the 2D coordinates of the feature point in image space and d is the unknown depth parameter. We can back-project $(u, v, d)^\top$ to 3D using the inverse projection process π^{-1}:

$$^{\mathrm{c}}\widetilde{\mathbf{p}} = \pi^{-1}(u, v, d) = \begin{pmatrix} f_{\mathbf{x}} & 0 & c_{\mathbf{x}} \\ 0 & f_{\mathbf{y}} & c_{\mathbf{y}} \\ 0 & 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} ud \\ vd \\ d \end{pmatrix}. \tag{2.42}$$

The second step is to track the feature points in consecutive images. A commonly used approach is the KLT tracker proposed by Lucas and Kanade (1981), which computes the position of the correspondences of the original feature points in subsequent images. Given an image at time t_j that hosts the landmarks, and a target image at time t_k, we compare the feature point position (u_l, v_l) with the tracked 2D position $\mathbf{z}_{jkl}$ in the target frame. The reprojection error can then be formulated as

$$\mathbf{r}_{\mathrm{img},jkl}(d_l, {}^{\mathrm{w}}\mathbf{T}_{\mathrm{i},t_j}, {}^{\mathrm{w}}\mathbf{T}_{\mathrm{i},t_k}) = \mathbf{z}_{jkl} - \pi\left(({}^{\mathrm{w}}\mathbf{T}_{\mathrm{i},t_k}\,{}^{\mathrm{i}}\mathbf{T}_{\mathrm{c}})^{-1\,\mathrm{w}}\mathbf{T}_{\mathrm{i},t_j}\,{}^{\mathrm{i}}\mathbf{T}_{\mathrm{c}}\left(\pi^{-1}(u_l, v_l, d_l)\right)\right). \tag{2.43}$$

2.3.3 IMU Preintegration and Relative Motion Error

While image measurements provide the core geometry information about the scene, the IMU measures short-term motion and offers complementary properties that enhance the robustness of the vision-based motion state estimation (Forster et al., 2015; Leutenegger et al., 2015). An IMU typically includes a gyroscope and an accelerometer, measuring angular velocity and linear acceleration at frequencies ranging from a few hundred to thousands of Hz. In contrast, camera measurements typically occur at frequencies under a hundred Hz. While IMUs can capture very high dynamic motions, they also exhibit significant noise.

To tightly combine these two modalities, an efficient preintegration method has been proposed by Forster et al. (2015). The core idea of IMU preintegration is to aggregate the high-frequency IMU measurements between two consecutive images at times t_j and t_k into a single relative motion constraint. We assume that both the angular velocity

measurement ${}^{\mathrm{i}}\widetilde{\boldsymbol{\omega}}$ and the linear acceleration measurement ${}^{\mathrm{i}}\widetilde{\mathbf{a}}$ are affected by additive white noise $\boldsymbol{\eta}$ and slowly drifting sensor biases $\mathbf{b}$:

$$ {}^{\mathrm{i}}\widetilde{\boldsymbol{\omega}} = {}^{\mathrm{i}}\boldsymbol{\omega} + \mathbf{b}_{\mathrm{g}} + \boldsymbol{\eta}_{\mathrm{g}} \tag{2.44} $$

$$ {}^{\mathrm{i}}\widetilde{\mathbf{a}} = {}^{\mathrm{i}}\mathbf{R}_{\mathrm{w}} \left({}^{\mathrm{w}}\mathbf{a} - \mathbf{g} \right) + \mathbf{b}_{\mathrm{a}} + \boldsymbol{\eta}_{\mathrm{a}}, \tag{2.45} $$

where $\mathbf{g}$ is the gravity vector, ${}^{\mathrm{i}}\boldsymbol{\omega}$ is the true body angular velocity and ${}^{\mathrm{w}}\mathbf{a}$ is the true linear acceleration in world frame. The relative motion, namely the pseudo-measurements between t_j and t_k are calculated as follows:

$$ \Delta\mathbf{R}_{jk} = \prod_{n=j}^{k-1} \mathrm{Exp}\left(({}^{\mathrm{i}}\widetilde{\boldsymbol{\omega}}_n - \mathbf{b}_{\mathrm{g}}) \Delta t \right) \tag{2.46} $$

$$ \Delta\mathbf{v}_{jk} = \sum_{n=j}^{k-1} \Delta\mathbf{R}_{jn} ({}^{\mathrm{i}}\widetilde{\mathbf{a}}_n - \mathbf{b}_{\mathrm{a}}) \Delta t \tag{2.47} $$

$$ \Delta\mathbf{t}_{jk} = \sum_{n=j}^{k-1} \left(\Delta\mathbf{v}_{jn} \Delta t + \frac{1}{2} \Delta\mathbf{R}_{jn} ({}^{\mathrm{i}}\widetilde{\mathbf{a}}_n - \mathbf{b}_{\mathrm{a}}) \Delta t^2 \right), \tag{2.48} $$

with $\Delta t = t_k - t_j$. The relative motion residual from the IMU measurements is expressed as:

$$ \begin{aligned} \mathbf{r}_{\mathrm{rel},jk}({}^{\mathrm{w}}\mathbf{T}_{\mathrm{i},t_j}, {}^{\mathrm{w}}\mathbf{v}_{\mathrm{i},t_j}, \mathbf{b}_{\mathrm{g},t_j}, \mathbf{b}_{\mathrm{a},t_j}, {}^{\mathrm{w}}\mathbf{T}_{\mathrm{i},t_k}, {}^{\mathrm{w}}\mathbf{v}_{\mathrm{i},t_k}) = \\ \begin{pmatrix} \mathrm{Log}\left(\Delta\mathbf{R}_{jk} {}^{\mathrm{w}}\mathbf{R}_{\mathrm{i},t_k}^{\top} {}^{\mathrm{w}}\mathbf{R}_{\mathrm{i},t_j} \right) \\ {}^{\mathrm{w}}\mathbf{R}_{\mathrm{i},t_j}^{\top} ({}^{\mathrm{w}}\mathbf{v}_{\mathrm{i},t_k} - {}^{\mathrm{w}}\mathbf{v}_{\mathrm{i},t_j} - \mathbf{g}\Delta t) - \Delta\mathbf{v}_{jk} \\ {}^{\mathrm{w}}\mathbf{R}_{\mathrm{i},t_j}^{\top} ({}^{\mathrm{w}}\mathbf{t}_{\mathrm{i},t_k} - {}^{\mathrm{w}}\mathbf{t}_{\mathrm{i},t_j} - {}^{\mathrm{w}}\mathbf{v}_{\mathrm{i},t_j} \Delta t - \frac{1}{2}\mathbf{g}\Delta t^2) - \Delta\mathbf{t}_{jk} \end{pmatrix}. \end{aligned} \tag{2.49} $$

Additionally, the slowly varying sensor biases are modeled with Brownian motion and discretized as follows:

$$ \mathbf{b}_{\mathrm{g},t_k} = \mathbf{b}_{\mathrm{g},t_j} + \boldsymbol{\eta}_{\mathrm{bg}} \tag{2.50} $$

$$ \mathbf{b}_{\mathrm{a},t_k} = \mathbf{b}_{\mathrm{a},t_j} + \boldsymbol{\eta}_{\mathrm{ba}}. \tag{2.51} $$

The bias residual can thus be computed as:

$$ \mathbf{r}_{\mathrm{bias},jk}(\mathbf{b}_{\mathrm{g},t_k}, \mathbf{b}_{\mathrm{g},t_j}, \mathbf{b}_{\mathrm{a},t_k}, \mathbf{b}_{\mathrm{a},t_j}) = \begin{pmatrix} \mathbf{b}_{\mathrm{g},t_k} - \mathbf{b}_{\mathrm{g},t_j} \\ \mathbf{b}_{\mathrm{a},t_k} - \mathbf{b}_{\mathrm{a},t_j} \end{pmatrix}. \tag{2.52} $$

The overall IMU measurements related residual is defined as: $\mathbf{r}_{\mathrm{IMU},jk} = (\mathbf{r}_{\mathrm{rel},jk}, \mathbf{r}_{\mathrm{bias},jk})^{\top}$.

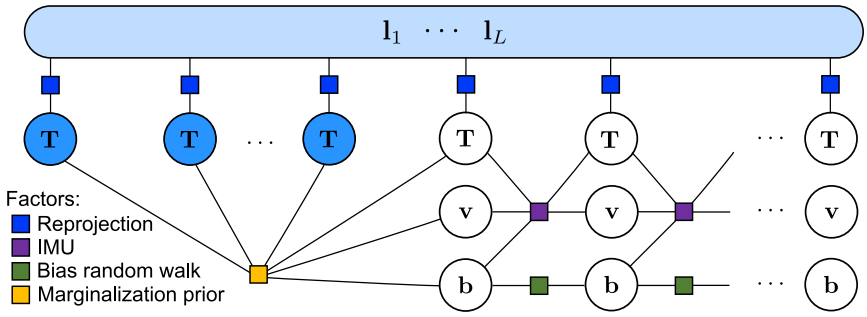


Figure 2.2: Factor graph of the VIO method. The blue circles represent the keyframes and the white circles are the regular frames.

2.3.4 Optimization and Marginalization

We tightly couple the image measurements and IMU measurements using the factor graph optimization method. A factor graph is a type of graph that consists of factors representing the residuals we derived from the image and IMU measurements, and are connected to the state variables we aim to estimate. Specifically, the reprojection factor is linked to the pose and landmark variables, while the IMU relative motion factor connects to the poses and velocities of consecutive images as well as the sensor biases. The objective function based on these measurements can be summarized as:

$$E_{\text{img, IMU}} = \sum_{j,k \in \mathcal{K}} \sum_{l \in \mathcal{L}} \mathbf{r}_{jkl}^{\top} \boldsymbol{\Sigma}_{jkl}^{-1} \mathbf{r}_{jkl} + \sum_{j,k \in \mathcal{F}} \mathbf{r}_{jk}^{\top} \boldsymbol{\Sigma}_{jk}^{-1} \mathbf{r}_{jk}, \tag{2.53}$$

where $\mathcal{K}$ and $\mathcal{L}$ denote the sets of images and landmarks, respectively, and $\mathcal{F}$ represents the set of indices of subsequent image pairs. The diagonal matrix $\boldsymbol{\Sigma}^{-1}$ contains the corresponding weight factors for the residuals.

The objective function $E_{\text{img, IMU}}$ is non-linear with respect to the state variables and is solved using the Gauss-Newton method by iteratively linearizing the objective function and updating it according to Equation (2.38). However, this equation requires inverting the Hessian $\mathbf{H}$ of the linearized objective function. When dealing with multiple pose states and several hundred landmarks, the size of this matrix can become very large, making the inversion operation computationally expensive. Fortunately, this matrix exhibits a special structure because the landmarks are independent conditioned on the pose states, thus the diagonal block of the Hessian pertaining to the landmarks is sparse. This structure allows for efficient inversion using the Schur complement technique (Leutenegger et al., 2015). First, we reformulate Equation (2.37) as:

$$\begin{pmatrix} \mathbf{B} & \mathbf{E} \\ \mathbf{E}^{\top} & \mathbf{C} \end{pmatrix} \begin{pmatrix} \delta \boldsymbol{\xi} \\ \delta \mathbf{d} \end{pmatrix} = \begin{pmatrix} \mathbf{a} \\ \mathbf{b} \end{pmatrix}, \tag{2.54}$$

where the Hessian matrix $\mathbf{H}$ is represented in block form with a dense diagonal block $\mathbf{B}$ for poses and biases, a sparse diagonal block $\mathbf{C}$ for landmarks, and a off-diagonal block $\mathbf{E}$. The incremental updates for the poses and biases $\delta\boldsymbol{\xi}$ and for the landmarks $\delta\mathbf{d}$ are also separated, as are the components of the right-hand side of the Equation (2.37) $\mathbf{a}$

and $\mathbf{b}$. Using the Schur complement, we can compute the solution as follows:

$$\begin{pmatrix} \delta\boldsymbol{\xi} \\ \delta\mathbf{d} \end{pmatrix} = \begin{pmatrix} (\mathbf{B} - \mathbf{E}\mathbf{C}^{-1}\mathbf{E}^{\top})^{-1}(\mathbf{a} - \mathbf{E}\mathbf{C}^{-1}\mathbf{b}) \\ \mathbf{C}^{-1}(\mathbf{b} - \mathbf{E}^{\top}\delta\boldsymbol{\xi}) \end{pmatrix}, \tag{2.55}$$

where $\mathbf{C}$ is diagonal and can be efficiently inverted. Furthermore, the product $(\mathbf{B} - \mathbf{E}\mathbf{C}^{-1}\mathbf{E}^{\top})$ has a much smaller size compared to the original Hessian and can be inverted efficiently using Cholesky decomposition.

To prevent the factor graph from continuously growing as new measurements are captured, we employ a sliding window approach to maintain a fixed number of variables. The factor graph of the sliding window-based VIO is illustrated in Figure 2.2. Each window contains regular image frames and keyframes, the latter being the frames where landmarks are extracted and hosted. When a new image is measured, new state variables are added to the graph, and the oldest states are dropped. If the state belongs to a keyframe, we marginalize out the velocity and biases with Schur complement while preserving the pose state. If it belongs to a regular frame, we marginalize all its variables. For more detailed information on keyframe selection and marginalization strategy, we refer the readers to Leutenegger et al. (2015) and Usenko et al. (2020). Once the old states are marginalized out, the remaining information forms the so-called marginalization prior $\mathbf{r}_{\text{marg}}^{\top}\boldsymbol{\Sigma}_{\text{marg}}^{-1}\mathbf{r}_{\text{marg}}$, which provides essential prior knowledge for the remaining state variables in the window. The final objective function for the sliding window-based VIO is:

$$E_{\text{VIO}} = E_{\text{img, IMU}} + E_{\text{marg}} = E_{\text{img, IMU}} + \mathbf{r}_{\text{marg}}^{\top}\boldsymbol{\Sigma}_{\text{marg}}^{-1}\mathbf{r}_{\text{marg}}. \tag{2.56}$$

We can estimate the states by minimizing this objective function under the MAP estimation framework using the Gauss-Newton method, as introduced in Section 2.2.

Spline-Based 6-DoF Object Motion Estimation with Event Cameras

3

Declaration of Contributions

The contents of this chapter are based on the peer-reviewed conference publication

(H. Li and Stückler, 2021)

with the following co-author contributions:

	Scientific Ideas	Data Generation	Analysis & Interpretation	Paper Writing
Haolong Li	50%	100%	70%	60%
Jörg Stückler	50%	0%	30%	40%

Jörg Stückler conceived the idea of using event camera and shape prior to estimate object motion. Jörg Stückler and Haolong Li conceived the idea of the two-layer optimization from events and frames. Haolong Li implemented the algorithm and performed the experiments. Haolong Li and Jörg Stückler wrote the paper.

Compared to the conference publication, this chapter contains unified notation and some reformulated and reorganized paragraphs.

3.1 Introduction

Event cameras measure intensity changes in pixels and provide an asynchronous data stream at high speed (in microseconds). This brings several potential advantages over traditional frame-based cameras such as high-dynamic range, no motion blur, and low latencies. They have the potential for novel robotics applications such as autonomous driving or flying robots with fast image motion or low-light settings. Significant progress has been made in developing approaches for camera motion estimation, depth reconstruction, and high dynamic image reconstruction with event-based cameras (Gallego et al., 2020). The measurement principle can also provide novel opportunities for 6-Degrees of Freedom (DoF) object tracking which is still largely unexplored by the research community.

In this chapter, we propose a novel approach for 6-DoF object tracking with event cameras. Object pose tracking is an important perception capability in many robotics

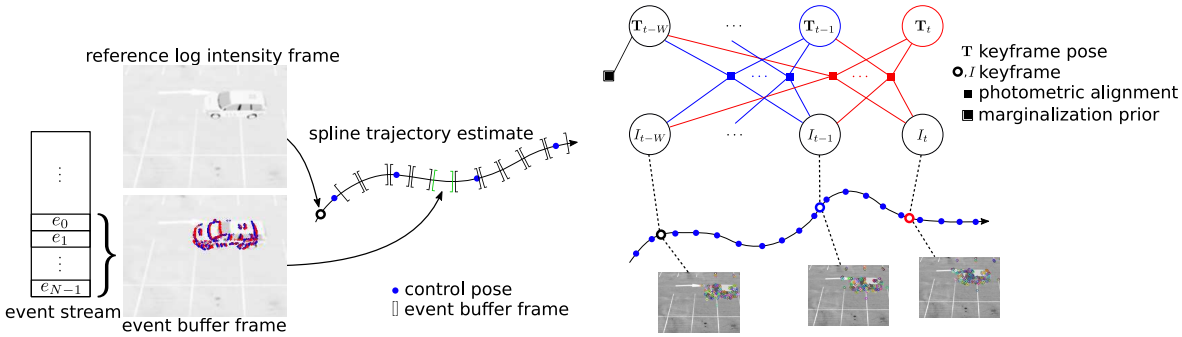


Figure 3.1: Left: We track the motion of the object from the stream of events. We parametrize the motion using cubic B-splines. The control poses are optimized based on a probabilistic generative event measurement model. The model predicts intensity changes using the object velocity and the intensity gradient in a reference frame. For computational efficiency, we accumulate N consecutive events e_i in event buffer frames. Right: We refine the object pose estimates of keyframes extracted from the images of a frame-based camera. We align the images photometrically at keypoints based on the known object shape and the estimated object poses. The latest keyframe and its estimated pose serve as reference for event-based tracking.

applications in dynamic scenes, for instance, for dynamic obstacle avoidance or robotic manipulation. In our tracking approach, we leverage a combination of event and frame-based camera measurements, which can be obtained from devices like the iniVation DAVIS camera that also provides intensity frame measurements, or by pairing an event camera with a traditional frame-based camera. Our approach tracks objects with known shapes which can be given by a mesh. We derive a probabilistic measurement model for the object's shape under rigid-body motion and track the motion of the object from the event stream using probabilistic inference. On a second optimization layer, we refine the tracked object poses using probabilistic direct image alignment in a window of frames captured by the camera at lower rates. We use cubic B-splines to parameterize the object's motion state, enabling the integration of high-speed event stream with lower-speed image frames and providing continuous and smooth motion constraints. We evaluate our approach on synthetic scenes with moving objects in indoor and outdoor scenarios and analyze the accuracy of our approach for 6-DoF pose tracking. We also demonstrate our approach in experiments with real data.

3.2 Related Work

Event-based cameras have several potential advantages over frame-based cameras for tracking: They provide high temporal resolution in the microseconds, high dynamic range, and low power consumption (Gallego et al., 2020). Hence, methods for processing and interpreting event stream have become a popular subject in computer vision and robotics research.

While significant research has been devoted to localizing event cameras in a given 3D map or simultaneous localization and mapping (e.g. H. Kim et al. (2016) and Rebecq et al. (2017)), only a few works consider the problem of object tracking. Vasco et al.

(2017) detect independent moving objects by tracking corners detected in event images integrated over short time windows. Mitrokhin et al. (2019a) estimate a warp field for the events in a time window with a 4-parameter global motion model. Using a time image representation, they segment events on moving objects that do not comply with the motion in the estimated background motion model. Mitrokhin et al. (2019b) train a neural network that predicts depth images, motion estimates, and motion segmentation from integrated event images and their time image in a time window. Different from our tracking approach, these methods estimate the 2D motion of the object.

Very recently, Dubeau et al. (2020) extend a deep learning-based object tracker for RGB-D cameras by also incorporating events from an event camera for high-speed object tracking. In contrast to this learning-based approach, we model the measurement process in a probabilistic generative way and derive an optimization-based method that combines events and frames. This way, our approach is not limited by the variation of objects and scenes that are seen by a deep neural network during training.

Some approaches for localization and mapping with event cameras have several similarities with our method. H. Kim et al. (2016) estimate 6-DoF camera motion, log intensity gradient, and inverse depth using three decoupled probabilistic filters in real-time. Using the log intensity gradient, high-dynamic range log intensity frames can be recovered through convex optimization. The method uses a probabilistic measurement model similar to ours which is derived from the event-generation process. For camera motion tracking, the method assumes the depth and log intensity of the scene given from the other filtering steps and uses the estimates to raycast intensity changes in the camera motion estimate. These expected measurements are compared with the intensity changes detected by the events to filter the camera pose. Bryner et al. (2019) propose a method for tracking camera motion with regard to a static background that has been captured using an RGB-D camera. They also apply a variant of the generative event measurement model used in H. Kim et al. (2016) and our approach. Different from their method, we track and segment objects in the event stream. Moreover, we combine event- and frame-based tracking in a two-layer optimization approach.

3.3 Method

Our method tracks the 3D motion of objects in a combined way from measurements of event- and frame-based cameras. We formulate tracking as optimization using a probabilistic measurement model of the event generation process. In a second optimization thread, we refine the object poses in the image frames using direct image alignment.

Figure 3.1 illustrates the key steps of our method. The camera provides an asynchronous stream of event measurements together with a stream of intensity frames at a regular slower rate. We use both types of data for tracking the 6-DoF object motion. On a fast optimization layer, we fit SE(3) trajectory splines to the events measured on the object. We formulate event-based tracking as a probabilistic inference problem based on a

generative model of the event measurement process. Inference is achieved by non-linear least squares to optimize the spline control poses.

A second slower optimization layer refines the object poses in the intensity frames measured by the camera. To this end, we formulate the optimization again as probabilistic inference, whereas now we measure the photometric alignment of the images using the object pose and shape in the frames. This optimization layer provides the reference object poses and intensity images for the event-based tracking layer. The event-based tracker on the other hand provides the frame-based optimization layer with initial poses of the objects in the frames and achieves higher frame-rate high-speed tracking.

3.3.1 Generative Event Model

Let $\mathbf{x}(t)$ be the image projection of 3D points in the scene at time t. The image projection is changing over time due to the underlying motion of the camera or the object. Assuming the brightness of a 3D point observation in the image remains constant, the measured intensity at a corresponding pixel location at different time steps t, t' stays equal, i.e.

$$L(\mathbf{x}(t), t) = L(\mathbf{x}(t'), t'), \tag{3.1}$$

where L here denotes the log intensity image. Linearizing the log intensity image L for the time t' using a first-order Taylor approximation yields:

$$L(\mathbf{x}(t+\delta t), t+\delta t) \approx L(\mathbf{x}(t), t) + \frac{\partial L}{\partial \mathbf{x}}\frac{\partial \mathbf{x}}{\partial t}\delta t + \frac{\partial L}{\partial t}\delta t. \tag{3.2}$$

Hence, we obtain the optical flow constraint:

$$\frac{\partial L}{\partial t}((\mathbf{x}(t), t)) + \nabla L(\mathbf{x}(t), t)\dot{\mathbf{x}}(t) = 0, \tag{3.3}$$

which relates the intensity change over time with the spatial intensity change $\nabla L(\mathbf{x}(t), t)$ and the optical flow $\dot{\mathbf{x}}(t)$ in the image. To avoid clutter, we will denote $\mathbf{x}(t)$ simply as $\mathbf{x}$ and $\dot{\mathbf{x}}(t)$ as $\dot{\mathbf{x}}$, unless specifically noted otherwise. Events measure intensity changes at pixels $\mathbf{x}$. By approximating the intensity change at a pixel as:

$$\Delta L(\mathbf{x}, t) = L(\mathbf{x}, t) - L(\mathbf{x}, t-\Delta t) \approx \frac{\partial L}{\partial t}(\mathbf{x}, t)\Delta t, \tag{3.4}$$

we can rewrite the change using the optical flow constraint as

$$\Delta L(\mathbf{x}, t) \approx -\nabla L(\mathbf{x})\dot{\mathbf{x}}\Delta t. \tag{3.5}$$

This relation explains the intensity changes generating events by the component of the flow $\dot{\mathbf{x}}$ along the image gradient $\nabla L(\mathbf{x})$ in the time interval Δt (Bryner et al., 2019; H. Kim

et al., 2016). The intensity change measurement model is

$$C = -\nabla L(\mathbf{x})\dot{\mathbf{x}}\Delta t + \delta, \tag{3.6}$$

where $\delta \sim \mathcal{N}(0, \mathbf{Q})$ models Gaussian measurement noise, and C is the log intensity change for the event. The optical flow $\dot{\mathbf{x}}$ is determined by the camera velocity $(\mathbf{v}, \boldsymbol{\omega})^\top$ and the depth $d := d(\mathbf{x})$ at the pixel $\mathbf{x} := (u, v)^\top$ (Corke, 2013):

$$\dot{\mathbf{x}} = \mathbf{B}(\mathbf{x}) \begin{pmatrix} \mathbf{v} \\ \boldsymbol{\omega} \end{pmatrix} \tag{3.7}$$

with

$$\mathbf{B}(\mathbf{x}) = \begin{pmatrix} -1/d & 0 & u/d & uv & -1-u^2 & v \\ 0 & -1/d & v/d & 1+v^2 & -uv & u \end{pmatrix}. \tag{3.8}$$

Since the event stream does not provide the image gradient directly, we determine the image gradient in a keyframe. The image gradient can be obtained from an intensity frame as measured for instance by a frame-based camera. In our experiments, we use the intensity frames measured concurrently with the events by a (simulated) DAVIS240C camera. The image gradient is determined by reprojecting the pixel with its raycasted depth on the object in the keyframe.

3.3.2 Event-Based Object Tracking

We track the motion of the object using the asynchronous event stream. We choose to increase the computational efficiency and average over noisy individual event readings by buffering multiple events at the pixels over short time windows. We use continuous SE(3) splines to represent the object trajectory. The spline provides pose estimates at arbitrary continuous times with a small set of control poses which we leverage for the asynchronous event buffer frames. The spline also inherently regularizes the estimated trajectory to be smooth, thereby introducing an implicit motion constraint to the motion tracking process. We formulate a probabilistic optimization objective to fit the spline trajectory to the event stream.

Spline Trajectory Representation

We use the cubic B-splines (Patron-Perez et al., 2015) to represent the time-continuous object pose in camera frame ${}^{\mathrm{c}}\mathbf{T}_{\mathrm{obj},t} \in \mathrm{SE}(3)$ at time t. For simplicity, we will refer to the object pose as $\mathbf{T}_t$ in the following discussion. The cubic B-spline for SE(3) can be formulated as:

$$\mathbf{T}_t = \mathrm{Exp}\left(\mathrm{Log}\left(\mathbf{T}_{t_0}\right)\right) \prod_{k=1}^{K-1} \mathrm{Exp}\left(\mathrm{Log}\left(\mathbf{T}_{t_{k-1}}^{-1}\mathbf{T}_{t_k}\right)\mathbf{B}_{t_k}\right), \tag{3.9}$$

where $K = 4$, $\mathbf{T}_k$ are control poses at knot times t_k and $\mathbf{B}_{t_k}$ are the cumulative basis functions.

Event-Based Trajectory Optimization

We optimize the spline segment of the four most recent control poses with all events in the segment. To increase computational efficiency and cancel noise due to image discretization, we accumulate event buffer frames from the events in short time windows.

Event Buffer Frames We accumulate the intensity changes of N consecutive events $e_0, \ldots, e_{N-1}$ in event buffer frames F. Events $e_i \coloneqq (u_i, v_i, \rho_i) \in \mathcal{E}$ are generated at pixel (u_i, v_i) and provide the sign ρ_i of the log intensity change with magnitude C. Since we assume consecutive events, it holds $t_0 \leqslant \ldots \leqslant t_{N-1}$. Pixels $F(u, v)$ of the event buffer frame accumulate intensity changes

$$F(u, v) = \sum_{\{e_i | e_i \in \mathcal{E}, (u_i, v_i) = (u, v)\}} \rho_i C. \tag{3.10}$$

We refer to the time t_0 of the first event in the event frame buffer as its start time $t_F \coloneqq t_0$ and $\Delta t_F \coloneqq t_{N-1} - t_0$ is the time span of the events in the buffer. If N events are accumulated, a new buffer is started.

Trajectory Optimization We optimize the control poses of the most recent spline segment when the segment is filled with event buffer frames, i.e. when the start time of the newest event buffer frame becomes later than the knot time t_{K-2} of the second to last control pose $\mathbf{T}_{t_{K-2}}$. After optimization, we include a new control pose at a fixed time interval. The new segment shifted by a control pose is again optimized as soon as it is filled.

We optimize the objective function

$$E(\mathbf{T}_{t_{K-4}}, \ldots, \mathbf{T}_{t_{K-1}}) = E_{\text{data}} + \lambda_{\text{reg}} E_{\text{reg}}, \tag{3.11}$$

which consists of a data term E_{data} and regularization term E_{reg} with weighting factor λ_{reg}, which we derive from the maximum a posterior estimation of the control poses given the event buffer frame measurements and a regularizing prior on the object acceleration.

The data log-likelihood of the event buffer frames within the spline segment is determined

by the probabilistic intensity change measurement in Equation (3.6),

$$E_{\text{data}} = \sum_{F \in \mathcal{F}} \sum_{\mathbf{x} \in \Omega} \frac{w_c^2}{w_c^2 + \|\nabla L(\mathbf{x})\|_2^2} \left\| \frac{F(\mathbf{x})}{\sqrt{\sum_{\mathbf{x}' \in \Omega} \left(F(\mathbf{x}')\right)^2}} + \frac{\nabla L(\mathbf{x}) \dot{\mathbf{x}}(t_F) \Delta t_F}{\sqrt{\sum_{\mathbf{x}' \in \Omega} \left(\nabla L(\mathbf{x}') \dot{\mathbf{x}}'(t_F) \Delta t_F\right)^2}} \right\|_2^2, \quad (3.12)$$

where $\mathcal{F}$ is the set of event buffer frames in the spline segment, Ω is the set of pixel coordinates in the image. We follow the approach of Bryner et al. (2019) and normalize the intensity changes due to the unknown log intensity threshold C of the camera in practice. Similar to the approach described in Engel et al. (2018), we down-weight pixels with high gradient, where the term w_c determines the strength of this factor. For evaluating the expected intensity changes $\nabla L(\mathbf{x})$, we use the last keyframe I_{KF} and its pose $\mathbf{T}_{t_{\text{KF}}}$ from the frame-based photometric optimization layer:

$$\nabla_{\mathbf{x}} L(\mathbf{x}) \approx \nabla_{\mathbf{x}} L_{\text{KF}}(\tau(\mathbf{x}, \mathbf{T}_{t_F})), \quad (3.13)$$

where $L_{\text{KF}} = \log(I_{\text{KF}})$ and $\mathbf{T}_{t_F}$ is the pose of the event buffer frame at time t_F. The event pixel location is projected to the keyframe as

$$\tau(\mathbf{x}, \mathbf{T}_{t_F}) = \pi \left(\mathbf{T}_{t_{\text{KF}}} \mathbf{T}_{t_F}^{-1} \left(\pi^{-1} \left(\mathbf{x}, d(\mathbf{x}, \mathbf{T}_{t_F}, \mathbf{\Phi}) \right) \right) \right), \quad (3.14)$$

where π and π^{-1} project 3D coordinates to image pixels and vice versa using the known camera calibration, the latter requiring the depth at the pixel. We raycast the depth $d(\mathbf{x}, \mathbf{T}_{t_F}, \mathbf{\Phi})$ on the object shape in the given pose, and this depth function can be differentiated for the pose as in Wang et al. (2017). The shape is represented by the signed distance function $\mathbf{\Phi}$. The 3D points in camera coordinates are transformed between the frames using the SE(3) relative pose $\mathbf{T}_{t_{\text{KF}}} \mathbf{T}_{t_F}^{-1}$ determined from the object pose in the keyframe $\mathbf{T}_{t_{\text{KF}}}$ and the spline.

The acceleration prior

$$E_{\text{reg}} = \sum_{F \in \mathcal{F}} \left\| \ddot{\mathbf{T}}_{t_F} \right\|_2^2 \quad (3.15)$$

favors constant velocity trajectories and is evaluated at the start times of the event buffer frames.

We assume an initial guess of the object pose is known, allowing us to initialize the first four control poses. To this end, we use ground truth for this purpose in our implementation. Note that accurate frame-based object detection and pose estimation might also be used.

3.3.3 Keyframe-Based Photometric Trajectory Optimization

A second layer optimizes for the poses of the keyframes which are used as reference for the event-based tracking. Based on the known object shape, we estimate object poses in the keyframes, which best align the keyframe images photometrically. The tracked object pose from the event-based tracking layer is used to initialize the object pose in new keyframes. New keyframes are selected from the intensity frames based on thresholds on rotation and translation. We optimize the object trajectory in a window of recent keyframes and marginalize old keyframes that shift outside the window.

Photometric Alignment

The object pose is determined in the keyframes by photometric alignment between pairs of keyframes in the optimization window. From each keyframe, we extract ORB feature points as keypoints (Rublee et al., 2011) and limit computation to only those points which project within a soft silhouette mask of the object given its current pose estimate. Photometric residuals are computed at the remaining keypoints on the object to optimize for the object pose.

Silhouette Projection We adapt the projection function proposed by Dame et al. (2013) and Prisacariu et al. (2013)

$$\sigma(\mathbf{\Phi}, \mathbf{x}) = 1 - \prod_{\mathbf{p} \sim \mathcal{R}(\mathbf{x})} \frac{\exp\left(\mathbf{\Phi}(\mathbf{p})\zeta\right)}{\exp\left(\mathbf{\Phi}(\mathbf{p})\zeta\right) + 1} \tag{3.16}$$

which projects the object shape, represented by the signed distance function $\mathbf{\Phi}$, into the image to find a smooth silhouette. The projection samples points $\mathbf{p}$ on the ray $\mathcal{R}(\mathbf{x})$ through the pixel $\mathbf{x}$ and evaluates a smooth indicator function for each point being inside or outside the object based on the signed distance function. We threshold at a specific value (0.95 in our experiments) and discard all ORB keypoints that project outside the object.

Photometric Residuals We formulate a photometric measurement model at the keypoints in each keyframe I_i towards other keyframes I_j

$$I_i(\mathbf{x}) = I_j(\omega(\mathbf{x}, \mathbf{T}_{t_i}, \mathbf{T}_{t_j})) + \epsilon, \tag{3.17}$$

where $\epsilon \sim \mathcal{N}(0, \sigma_I^2)$ is Gaussian noise, and ω reprojects pixels $\mathbf{x}$ from image I_i to image I_j based on the object shape and the corresponding poses $\mathbf{T}_{t_i}, \mathbf{T}_{t_j}$ at the keyframes. We reproject pixels through the warping function

$$\omega(\mathbf{x}, \mathbf{T}_{t_i}, \mathbf{T}_{t_j}) = \pi\left({}^{c,t_j}\mathbf{T}_{c,t_i}\left(\pi^{-1}(\mathbf{x}, d(\mathbf{x}, \mathbf{T}_i, \mathbf{\Phi}))\right)\right).$$

Here, we use the SE(3) transform ${}^{c,t_j}\mathbf{T}_{c,t_i} = \mathbf{T}_{t_j}\mathbf{T}_{t_i}^{-1}$ determined from the object poses in the frames to transform pixels. Keypoints are discarded that do not hit the object shape during raycasting for depth estimation.

Windowed Optimization

We formulate the optimization for the object poses in the keyframes as maximum a posteriori estimation using the probabilistic photometric measurement model

$$E\left(\{\mathbf{T}_{t_i}\}_{i=0}^{W-1}\right) = \sum_{i=0}^{W-1}\sum_{j<i}\rho\left(I_i(\mathbf{x}) - I_j(\omega(\mathbf{x}, \mathbf{T}_{t_i}, \mathbf{T}_{t_j}))\right), \tag{3.18}$$

where we determine photometric residuals from a keyframe to older keyframes, and ρ is the robust Cauchy norm.

For bounding the run-time complexity, the keyframe poses are optimized in a sliding window of a fixed number $W = 7$ of keyframes similar to Leutenegger et al. (2015). Information about old keyframes that drop out of the sliding window is marginalized in the probabilistic formulation. The optimization is triggered only if a new keyframe is inserted. Once the optimization converges, the last keyframe is passed to the event-based tracking layer as the new reference frame.

Keyframe Selection

We select the latest current frame as a new keyframe, if the translation or rotational distance traveled by the object according to the event-based tracking layer is larger than some thresholds. The pose of a new keyframe is initialized with the current tracked pose of the object by the event-based tracking.

3.4 Experiments

We evaluate our approach for tracking accuracy on synthetic datasets where ground-truth poses are available. We use the root mean square error (RMSE) of the absolute trajectory error (ATE) and the relative pose error (RPE) measures (Sturm et al., 2012), that evaluate the global alignment of the trajectory and the drift, respectively. We evaluate RPE for $10, 20, \ldots, 50\%$ sequence lengths of the full trajectory length in m. We obtain a single RPE measure by averaging the results over all sequence lengths.

Our synthetic sequences are generated with an event camera simulator (Rebecq et al., 2018) based on Blender. We generate sequences for 2 objects from the YCB dataset (Calli et al., 2015) and 2 cars from the ShapeNet dataset (Chang et al., 2015). For the YCB objects, we have 3 sequences each with falling, sliding, and dice motion. The cars move

Table 3.1: Accuracy of trajectory estimates in RPE and ATE on YCB sequences. Our tracking approach recovers the object motion at good accuracy.

	transl. RMSE RPE in m			rot. RMSE RPE in deg			transl. RMSE ATE in m		
object	falling	sliding	dice	falling	sliding	dice	falling	sliding	dice
box	0.086	0.056	0.024	5.196	5.686	3.840	0.064	0.028	0.012
can	0.038	0.107	0.012	5.640	6.110	1.423	0.029	0.119	0.031

either straight, turn left, or turn right, whereas we use 3 different speed settings per sequence type.

3.4.1 Setup

The time span between the knots in the spline is set to 15 ms, and we set $\lambda_{\text{reg}} = 0.1$. The frame rate of the camera is 30 Hz. For the YCB objects, we accumulate $N = 1500$ events in each event buffer frame. We generate new keyframes after the object has traveled a translational distance of 0.1 m and a rotational distance of 5 deg. The contrast weight factor w_c is 0.005. The pose is initialized without noises. For the car objects, we create the motions with 3 different speed levels. In the first level, the average linear speed is about 1.25 m/s. The second speed level is about 2.5 m/s, and the third speed level is about 5 m/s. The pose is initialized without noises. The translational distance of keyframes is 0.5 m and the rotational distance is 10 deg. We set the number of events per event buffer frame to $N = 4000$. For the textureless car, the contrast weight factor w_c is 0.3, and for the textured one, it is set to 0.1.

3.4.2 Results

Synthetic YCB Object Sequences

Table 3.1 shows RPE and ATE results for the various sequences with the box and can YCB objects. The maximum sizes of the object are 0.3 meters and 0.45 meters. The camera is positioned above the objects. The average distance of the objects from the camera is 1.6 m for the sequences. We observe that the position and rotation of the objects are tracked by our approach at good accuracy on these sequences. Figure 3.2 depicts trajectory overlays of the estimates by our approach alongside the ground-truth. It can be seen that our approach well recovers the object motion.

Synthetic Car Sequences

In Table 3.2 we present our RPE and ATE results for the car sequences. We use a mean SDF shape over a set of car shapes to track both cars. The maximum length of the

Table 3.2: Accuracy of trajectory estimates in RPE and ATE on car sequences.

		transl. RMSE RPE in m			rot. RMSE RPE in deg			transl. RMSE ATE in m		
object	speed	left turn	right turn	straight	left turn	right turn	straight	left turn	right turn	straight
textured	1x	0.189	0.283	0.250	2.764	3.306	2.958	0.097	0.171	0.149
textured	2x	0.409	0.355	0.206	5.612	2.991	2.876	0.114	0.217	0.109
textured	4x	0.181	0.406	0.453	2.957	3.033	3.478	0.080	0.175	0.138
textureless	1x	0.811	0.450	0.528	10.816	6.168	6.245	0.471	0.224	0.287
textureless	2x	0.365	0.648	0.436	3.717	4.509	4.069	0.183	0.235	0.261
textureless	4x	0.356	0.551	0.299	4.195	6.548	3.077	0.146	0.231	0.173

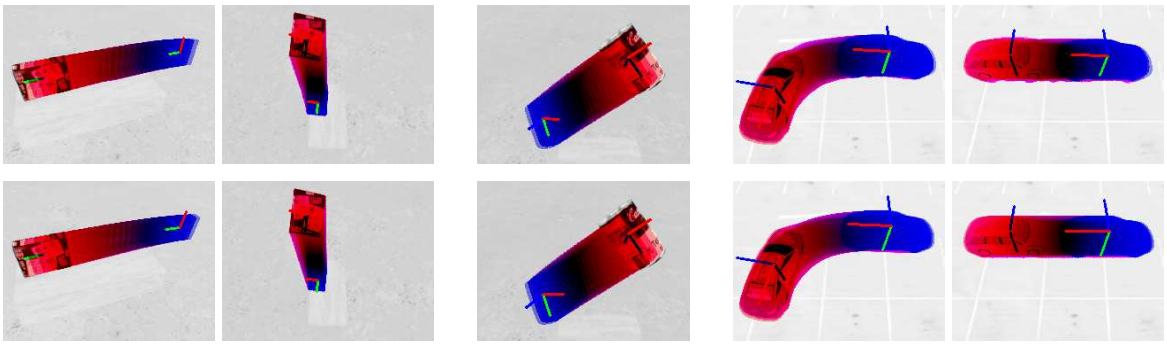


Figure 3.2: Estimated (top) vs. ground-truth trajectory (bottom) as image overlays for the YCB box object (left two columns: sliding, falling), the YCB can object (middle column: dice), and the two car sequences (right two columns). Time is visualized from red to blue for start to end. Our approach well recovers the ground-truth motion of the objects.

shape is about 4.2 m. The camera has an average distance of 8.3 m. Again, our approach determines the position and rotation of the objects with good accuracy. The textured object provides more events on the objects and hence can be tracked more accurately. For the textureless object, fewer events are available inside the objects (see Figure 3.1), which makes the object more difficult to track. We see a slight degradation in accuracy in both translational and rotational motion. Our approach handles varying speeds of the cars well with similar accuracy. Interestingly, it is less accurate for slow cars since fewer events are generated and fewer event buffer frames are available for optimizing the spline segments. We also show trajectory overlays in Figure 3.2 and qualitative comparisons with ground truth.

3.4.3 Ablation Study

We compare our approach with purely event- or frame-based variants (see Table 3.3). For purely event-based tracking, we use the latest frames within the spline as keyframe. For purely frame-based alignment, we use all frames as keyframes and initialize the pose of new keyframes based on the previous frame. In the car and YCB sequences, there are about 900 and 2500 event buffer frames per second on average, respectively. While frame-based alignment can outperform our method on the car sequences, it diverges for the can object early in the sequence. For the latter, the frame differences are too large for frame-based tracking, which is alleviated by the higher rate event buffer frames.

Table 3.3: Accuracy of trajectory estimates in RPE and ATE for purely event- or frame-based tracking.

		transl. RMSE RPE in m			rot. RMSE RPE in deg			transl. RMSE ATE in m		
method	dataset	falling	sliding	dice	falling	sliding	dice	falling	sliding	dice
event-based tracking	box	0.162	0.046	0.067	8.182	2.628	11.172	0.151	0.024	0.035
event-based tracking	can	0.045	0.091	0.039	5.196	2.745	3.069	0.021	0.068	0.027
frame-based alignment	box	0.072	∞	∞	7.792	∞	∞	0.042	∞	∞
frame-based alignment	can	∞	∞	∞	∞	∞	∞	∞	∞	∞
method	dataset	left turn	right turn	straight	left turn	right turn	straight	left turn	right turn	straight
event-based tracking	textured 4x	0.958	0.8607	0.833	15.613	13.590	4.133	0.701	0.503	0.785
event-based tracking	textureless 4x	1.186	∞	0.978	11.299	∞	10.084	0.807	∞	0.549
frame-based alignment	textured 4x	0.077	0.048	0.063	0.851	1.041	0.555	0.051	0.024	0.033
frame-based alignment	textureless 4x	0.149	0.139	0.062	2.198	2.609	0.659	0.066	0.077	0.036

Moreover, our event-based tracking layer provides updates on the continuous spline estimate at twice the time resolution. Our results demonstrate that the combination of both layers is advantageous for tracking.

3.4.4 Computation Time

We measure the run-time of our approach with the textureless car sequences on an Intel Xeon Silver 4112 CPU@2.60GHz with 8 cores. On average our implementation requires 109.5 ms per spline segment optimization on the event-based tracking layer. For each windowed keyframe optimization, it uses 363 ms on average. Our implementation processes trajectories that are, on average, 2.4 s long, in 24.5 s. Note that it has not been tuned yet e.g. through parallel processing.

3.4.5 Real Data

We also test our method on real data from a DAVIS240C sensor with challenging low-resolution gray scale images and noisy event measurements. In the first frame, the tracked pose is initialized with ground-truth obtained by a motion capture system. Results on three sequences with translational (0.77 s), rotational (0.54 s), and circular motion (1.56 s) can be seen in Figure 3.3. We observe that our approach requires careful tuning and accurate initialization for this noisy data. The incremental tracking on both layers leads to drift which can make tracking fail, especially on longer sequences. The photometric alignment error in Equation (3.17) is determined in patches of size 3×3 around each keypoint and a keyframe window size of $W = 10$ is used. Additionally, a velocity regularization term $E_{\text{reg},2} = \lambda_{\text{reg},2} \sum_{F\in\mathcal{F}} \Delta t_F \left\|\dot{\mathbf{T}}(t_F)\right\|_2^2$ is added with $\lambda_{\text{reg},2} = 1.6$. The term favors small velocities, if the events are sparse and an event buffer frame spans larger time intervals. We set $w_c = 0.005$ for the circular motion and $w_c = 0.001$ otherwise.

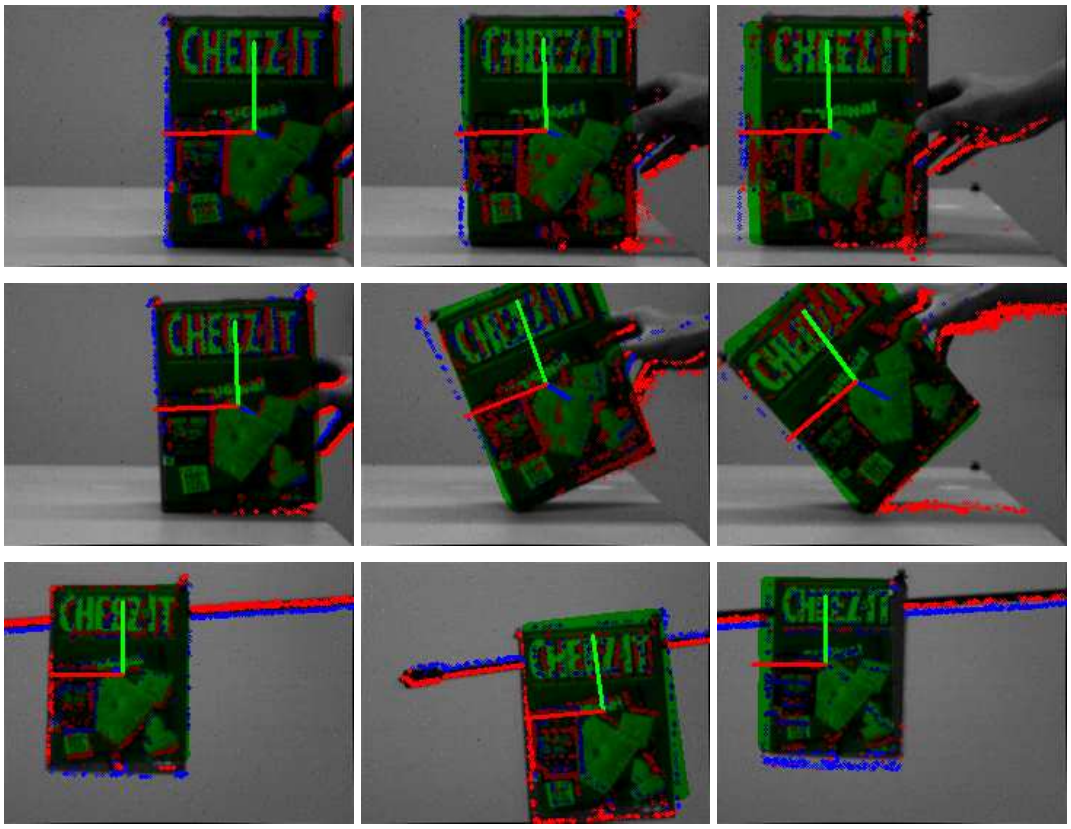


Figure 3.3: Results on real data sequences with translational (top), rotational (middle), and circular motion (bottom). The motion progresses from the start frame on the left to the end frame on the right. The green shaded areas and axes represent the estimated object pose, while the red and blue points indicate positive and negative events, respectively.

3.5 Conclusion

In this chapter, we present a novel model- and optimization-based approach for object tracking from events and frames. We propose a two-stage processing pipeline, where the faster layer tracks the object motion from the asynchronous event stream with regard to a reference image frame. The tracking is formulated as probabilistic inference for the object trajectory based on a generative event measurement model. We represent the trajectory continuously using cubic B-splines, which allows us to determine object pose and velocity estimates at arbitrary times on the spline for the event measurement model. The poses of the reference frames are estimated in a second optimization layer, which optimizes the object poses in keyframes using direct image alignment on keypoints. We evaluate our approach on synthetic sequences of typical household objects and car objects, demonstrating good tracking accuracy. Additionally, we analyze the combination of events and frames in an ablation study. We also report on experiments with real camera data.

Current limitations of our approach include the requirement of accurate initialization, trajectory smoothness assumption by the spline, and the incremental tracking which leads to drift. If accumulated drift or motion changes become too high, our tracking approach fails.

In future work, we plan to scale our approach further for improved tracking on longer sequences and real event measurements. The robustness of our method could be improved, for instance, by combining it with object pose detection or by tight coupling of the event- and frame-based tracking.

Visual-Inertial and Leg Odometry Fusion for Dynamic Locomotion

4

Declaration of Contributions

The contents of this chapter are based on the peer-reviewed conference publication

with the following co-author contributions:

	Scientific Ideas	Data Generation	Analysis & Interpretation	Paper Writing
Victor Dhédin	15%	35%	20%	40%
Haolong Li	10%	45%	20%	35%
Shahram Khorshidi	10%	0%	5%	5%
Lukas Mack	0%	10%	5%	0%
Adithya Kumar Chinnakkonda Ravi	5%	0%	5%	0%
Avadesh Meduri	5%	0%	0%	0%
Paarth Shah	5%	0%	0%	0%
Felix Grimminger	5%	0%	5%	0%
Ludovic Righetti	5%	0%	0%	0%
Majid Khadiv	15%	0%	20%	10%
Jörg Stückler	25%	10%	20%	10%

Jörg Stückler and Majid Khadiv proposed using visual-inertial odometry (VIO) as the state estimator for a control framework BiConMP (Meduri et al., 2023) of the quadruped robot Solo12. Jörg Stückler conceived the idea of combining VIO with leg odometry in an extended Kalman filter (EKF) framework. Victor Dhédin, Haolong Li and Jörg Stückler proposed predicting the VIO state estimate at the rate of the IMU measurements. Jörg Stückler and Haolong Li proposed ground height bias estimation. Victor Dhédin implemented the VIO state prediction, ground height bias model and combined the VIO with the leg odometry EKF implemented by Shahram Khorshidi. Haolong Li helped with code debugging and implemented the evaluation tool. Victor Dhédin, Haolong Li, Lukas Mack and Jörg Stückler collected the data. For the final publication, Haolong Li implemented, executed, and analyzed all presented experiments. Victor Dhédin, Haolong Li, Shahram Khorshidi, Lukas Mack, Adithya Kumar Chinnakkonda Ravi, Felix

Grimminger, Majid Khadiv and Jörg Stückler regularly discussed the approach. Victor Dhédin, Haolong Li, Shahram Khorshidi, Majid Khadiv and Jörg Stückler wrote the paper.

Compared to the conference publication, this chapter contains unified notation and some reformulated and reorganized paragraphs.

4.1 Introduction

Legged robots are potentially capable of traversing uneven and unstructured terrains through making and breaking contacts with their environments using their feet. However, this capability introduces new challenges for estimation and control algorithms. For instance, a state estimation algorithm should constantly fuse the exteroceptive and proprioceptive measurements with the kinematics of the limbs currently in contact with the environment to estimate the robot floating base pose and velocity for motion control.

Early works for base state estimation of legged robots focused on fusing an on-board IMU with the leg odometry through an extended Kalman filter (EKF) framework to provide estimates of base states for the low-level controller (Blösch et al., 2013, 2012; Rotella et al., 2014). While this approach can provide drift-free base velocity and roll-pitch orientation, the base position and yaw orientation are unobservable which poses limitations especially for locomotion on uneven surfaces or motions with considerable vertical motion of the base such as jumping.

Recent works couple these proprioceptive measurements with exteroceptive modalities, e.g., camera or LiDAR, through loosely (Camurri et al., 2020) or tightly (Hartley et al., 2018a; Wisth et al., 2023) coupled methods. While the tightly coupled approach has the benefit of fusing all the modalities with direct consideration of their measurement uncertainty, it can be computationally demanding, especially for robots with limited computational resources. In our approach, we aim at a loosely coupled approach to integrate visual-inertial state estimation with leg odometry in a high-rate EKF state estimator to provide low-drift states that are sufficiently accurate and smooth for control. This way, the EKF and controller computation can run on a different device than the visual-inertial odometry (VIO). Furthermore, we can predict the VIO measurements and use them to reduce the delay, while the EKF can access the low-drift pose estimates from VIO. The main contributions of this work are

- We propose a novel approach to combine the benefits of VIO and leg odometry in a loosely coupled EKF approach to estimate low-latency and low-drift base states for agile locomotion. We compensate for height drift of the VIO using leg kinematics measurements when the legs are in contact with the ground.
- We perform an extensive set of experiments including outdoors on the open-source quadruped Solo12 (Grimminger et al., 2020). This is the first work that integrates

visual and proprioceptive measurements with nonlinear model predictive control for dynamic locomotion on this hardware.

4.2 Related Work

State estimation from only leg odometry and IMU such as in Blösch et al. (2012), Hartley et al. (2020, 2018c), and J.-H. Kim et al. (2021) has limitations in observability of state variables such as yaw rotation or absolute position in a world reference frame. To this end, several approaches combine proprioceptive and IMU measurements with exteroceptive sensors such as vision (Chilian et al., 2011; Dudzik et al., 2020; Hartley et al., 2018b; Y. Kim et al., 2022; Teng et al., 2021), LiDAR (Nobili et al., 2017), or both (Camurri et al., 2020; Wisth et al., 2019).

Vision sensors are particularly lightweight compared to LiDARs. They typically impose only little constraints on the payload of the quadruped which is particularly important for dynamic quadrupeds. Chilian et al. (2011) propose an early multi-sensor fusion approach which integrates IMU pose measurements with relative pose measurements from visual and leg odometry. The pose information is combined in a weighted manner. Teng et al. (2021) extend an EKF approach which fuses IMU and leg odometry to also integrate velocity measurements from a visual-inertial odometry method. In Hartley et al. (2018b) a fixed-lag smoothing approach based on factor graph optimization has been proposed. The approach uses visual odometry estimates as relative pose factors. Y. Kim et al. (2022) tightly integrate visual keypoint depth estimation with inertial measurement and preintegrated leg velocity factors. Our approach integrates absolute yaw and position measurements by the VIO, while height drift of the VIO with respect to the ground height is compensated by estimating the height bias in the EKF. In our approach, we aim at a lightweight system which decouples visual-inertial state estimation from the high-rate EKF state estimator used for control. This allows the EKF and controller to run on a different computer than the VIO. Moreover, by predicting the VIO measurements, delay is reduced and computational load for reintegration of measurements in the EKF can be avoided.

4.3 Method

In our approach, we fuse visual and inertial measurements with leg odometry for estimating the position, orientation, and velocity of the robot with respect to the ground plane. Figure 4.1 provides an overview of our system. Base state estimation is performed at high frequency with low latency using an EKF to be used in a real-time model-predictive control (MPC) approach for trotting and jumping motions (Meduri et al., 2023). The EKF fuses information from different sensory sources (see Figure 4.2b), it takes as input measurements of an IMU mounted on the robot, leg odometry data from the joints of the legs (angular position, angular velocity and torque), and pose

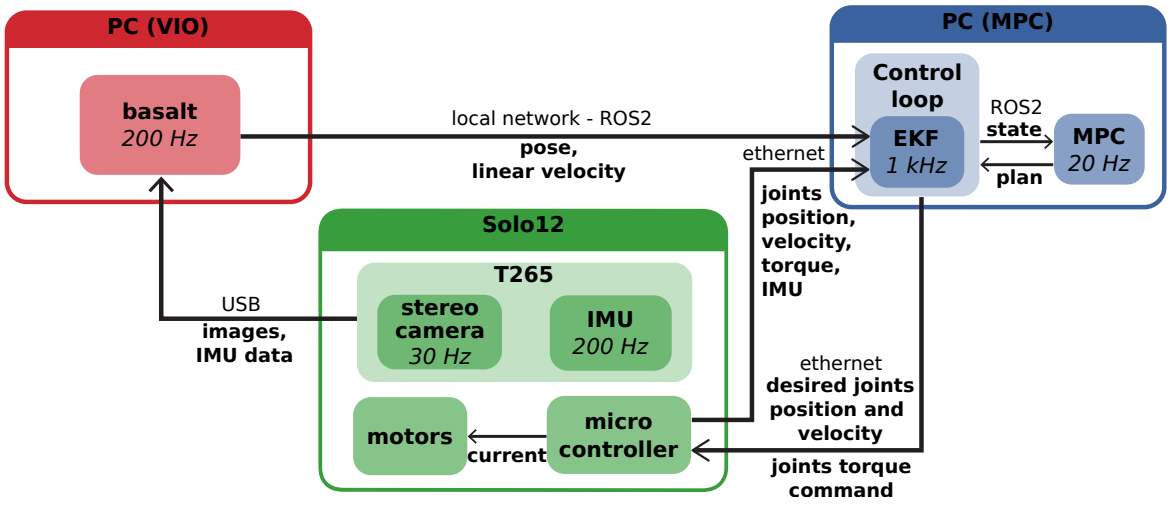


Figure 4.1: System overview and communication diagram.

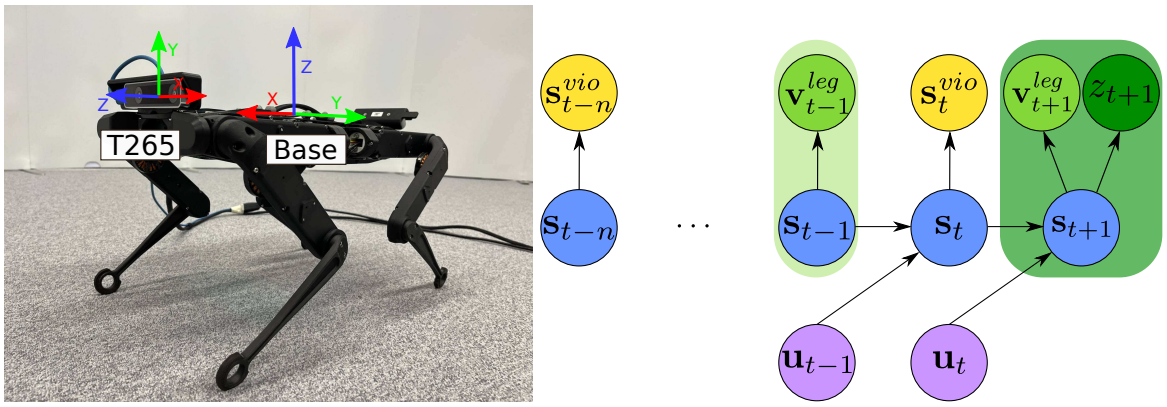
measurements estimated by a visual-inertial odometry (VIO) algorithm (using a second IMU in a visual-inertial camera). For dynamic locomotion, accurately estimating the height of the robot above the ground plane is important. While the VIO does not provide an absolute reference to the ground plane directly and will drift in height over time, we use contact detection and leg kinematics to obtain height measurements. In fact, VIO and leg odometry provide complementary strengths. VIO can measure the absolute roll and pitch in the environment, and build a map of the environment for estimating the base position and yaw orientation (rotation around gravity direction) with respect to this local map. While the local map estimate still drifts, this estimation error is typically significantly smaller than obtained by leg odometry which is prone to foot slippage and inaccuracies of contact detection.

4.3.1 Visual-Inertial Odometry

VIO algorithms estimate the motion of a camera over time by tracking landmarks detected in successive camera images from one or several cameras and integrating inertial measurement from an IMU using kinematics. As introduced in Section 2.3, this problem is usually formulated as finding a state that minimizes both a reprojection objective $E_{\text{img}}(\mathbf{s})$ computed on landmarks, an objective $E_{\text{IMU}}(\mathbf{s})$ associated with the motion determined from the IMU measurements with the marginalization prior $E_{\text{marg}}(\mathbf{s})$:

$$\mathbf{s}^*_{\text{VIO}} = \arg\min E_{\text{img}}(\mathbf{s}) + E_{\text{IMU}}(\mathbf{s}) + E_{\text{marg}}(\mathbf{s}). \tag{4.1}$$

We base our VIO estimator on basalt (Usenko et al., 2020). Assuming the extrinsic pose ${}^{\text{b}}\mathbf{T}_{\text{i}}$ between the robot base link frame b and the visual-inertial device frame i can be computed, we utilize the VIO method to estimate the robot state $\mathbf{s}_{\text{VIO}} = \left({}^{\text{w}}\mathbf{T}_{\text{b}}, {}^{\text{w}}\mathbf{v}, \mathcal{L}, \mathbf{b}_{\text{g}}, \mathbf{b}_{\text{a}}\right)$. Here, ${}^{\text{w}}\mathbf{T}_{\text{b}} \in \text{SE}(3)$ is the pose of the robot base frame b expressed in the VIO's world frame w, and ${}^{\text{w}}\mathbf{v} \in \mathbb{R}^3$ denotes the linear velocity of the robot base with respect to the world, expressed in the world frame. These quantities are directly transformed from the visual-



(a) Robot base and camera frames. **(b)** Graphical model.

Figure 4.2: Left: Robot base and camera frames. Right: Graphical model. The x-axis of robot base points forward and z-axis points upward. The state of the EKF is represented as $\mathbf{s}_t$ in the blue circle. We use the measurement of IMU (circle in magenta) mounted on the robot to predict the next state at 1000 Hz. The yellow circle represents VIO measurement at 200 Hz, the shallow green circle is the leg velocity measurement at contact. The height measurement in the dark green circle is added if all four legs are in contact with ground.

inertial device frame i based on the extrinsic pose ${}^{b}\mathbf{T}_{i}$. Additionally, we also consider the set of landmarks $\mathcal{L}$ and the IMU biases $\mathbf{b}_g$, $\mathbf{b}_a$, as described in Section 2.3.

Windowed Optimization The reprojection error is computed over a set of keypoints that are observed in different frames. To prevent the size of the optimization problem from growing, basalt uses a bounded window of recent frames and keyframes, and marginalizes information of old frames that drop out of the optimization window. In our case, the window corresponds to the 3 most recent frames and 7 most recent keyframes.

Low-Latency VIO Prediction In practice, the VIO has a moderate latency due to computation (approximately 5.8 ms optimization time on average with a standard deviation of 3.1 ms in our setup) and additional communication delays. The output rate is limited by the image frame rate. We propose to use IMU predictions to update the last VIO state estimate at a higher rate and to fuse these output states with leg odometry and a high precision IMU on the robot which helps reduce the latency and increases the output rate. By this, the EKF does not require memorization of old states and measurements, and reestimation after each image-rate update on the EKF/MPC computing device (as e.g. in Camurri et al. (2020)). The VIO outputs the prediction of the robot pose and velocity at the rate of the IMU in the VIO sensor (200 Hz for our camera) estimated using the IMU preintegration model (see Section 2.3) from the latest camera frame with optimization result available. Once the optimization result for the current frame is available, we reintegrate the IMU measurements and continue predicting the VIO state from this updated pose estimate (on the VIO computing device).

4.3.2 Sensor Fusion for Legged Robot State Estimation

We adapt the approach in Camurri et al. (2020) to fuse measurements of the pose and velocity of the robot's base link using an EKF. Different from Camurri et al. (2020), we integrate high-rate, low latency state observations from VIO and estimate the difference between VIO height estimate and ground height by leg kinematics. The EKF allows for integrating measurements with various rates and asynchronous timing. The state estimated by the EKF is $\mathbf{s}_{\text{EKF}} = \left({}^{\text{w}}\mathbf{T}_{\text{b}}, {}^{\text{b}}\mathbf{v}, \mathbf{b}_{\text{g}}, \mathbf{b}_{\text{a}}, \mathbf{b}_{\delta z}\right)$, where ${}^{\text{w}}\mathbf{T}_{\text{b}}$ is the robot base pose in world frame, ${}^{\text{b}}\mathbf{v}$ is the body linear velocity, and $\mathbf{b}_{\text{g}}, \mathbf{b}_{\text{a}}$ are the biases of the gyroscope and accelerometer measurements from an IMU mounted on the robot base (different IMU than used for VIO). The height bias $\mathbf{b}_{\delta z}$ compensates for the vertical drift of the VIO. We use the IMU prediction model in Camurri et al. (2020) to propagate the state with the measurements and estimate the gyroscope and accelerometer biases.

Leg Odometry Measurements

By determining the set of feet in contact with the ground, we can measure the linear velocity of the robot's base link from the leg kinematics (Camurri et al., 2020). Assuming that the foot k remains stationary while it is in contact with the ground, the linear velocity of the floating base can be measured as

$$ {}^{\text{b}}\mathbf{v} = -{}^{\text{b}}\dot{\mathbf{t}}_k - {}^{\text{b}}\boldsymbol{\omega} \times {}^{\text{b}}\mathbf{t}_k. \tag{4.2} $$

Here, ${}^{\text{b}}\mathbf{v}$ and ${}^{\text{b}}\boldsymbol{\omega}$ are the body linear and angular velocity of the robot base. ${}^{\text{b}}\mathbf{t}_k$ is the foot position in the robot base frame and ${}^{\text{b}}\dot{\mathbf{t}}_k$ is the corresponding time derivative. This method enables good accuracy on velocities and low latency. However, since only the velocity is observable, this method hardly handles drift in position, especially in height, which is detrimental for control of dynamic motions with significant changes in base height. The angular velocity in this observation model is measured directly by the IMU compensated with the estimated gyroscope bias.

We choose a simpler contact classification model than Camurri et al. (2020) in order to estimate the set of feet in contact. By assuming that the robot base remains flat during contact transitions, we can consider an equal distribution of the robot's total weight over the feet in contact with the ground. We use a Schmitt trigger to implement a robust hysteresis on the contact decision. If the norm of the force at each endeffector is higher than an upper threshold, we consider the foot as in contact with the ground, and if the norm is below a lower threshold, the endeffector is no longer in contact. The hysteresis in the contact detection helps to reject outliers due to high joint acceleration when the endeffector leaves the ground. We compute the endeffector force norm $F_k = ||\mathbf{F}_k||$ using the joint torque $\boldsymbol{\tau}$ by

$$ \mathbf{F}_k = (\mathbf{S}_k \mathbf{J}_k^\top)^{-1} \mathbf{S}_k \boldsymbol{\tau}, \tag{4.3} $$

where $\mathbf{S}_k$ is the selection matrix for the joints and $\mathbf{J}_k$ is the Jacobian of leg k. To exclude outliers, the leg odometry measurement is updated only if the leg is in contact with the ground for N_{contact} consecutive steps.

By having the joint positions and velocities sensed from the encoders one can use forward kinematics to compute the velocity and position of each endeffector in the base frame. By collecting all the effects of noise into one additive noise term, the measurement model can be rewritten as (Camurri et al., 2020):

$$^{\text{b}}\widetilde{\mathbf{v}} = -\boldsymbol{J}(\widetilde{\mathbf{q}}_k)\dot{\widetilde{\mathbf{q}}}_k - {}^{\text{b}}\widetilde{\boldsymbol{\omega}} \times \text{fk}(\widetilde{\mathbf{q}}_k) = {}^{\text{b}}\mathbf{v}^{\text{EKF}} + \boldsymbol{\eta}^{\mathbf{v}}, \tag{4.4}$$

where $\widetilde{\mathbf{q}}_k$ are the measured joint angles of leg k, $^{\text{b}}\widetilde{\boldsymbol{\omega}}$ is the gyroscope measurement compensated by its bias, and $\text{fk}(\widetilde{\mathbf{q}}_k) = {}^{\text{b}}\mathbf{t}_k$ is the forward kinematics for the foot contact point.

VIO Pose and Velocity Measurements

The VIO provides additional pose and velocity estimates of the robot base link in the inertial frame (world frame). Roll and pitch are estimated drift-free by the VIO, while 3D position and yaw orientation are estimated with respect to the estimated keypoint map and can drift. However, the drift in position and yaw orientation is significantly smaller than the drift by fusing leg odometry and IMU alone. The measurement model of the VIO pose and velocity is

$$^{\text{w}}\mathbf{t}_{\text{b}}^{\text{VIO}} = {}^{\text{w}}\mathbf{t}_{\text{b}}^{\text{EKF}} + (0, 0, \mathbf{b}_{\delta z})^{\top} + \boldsymbol{\eta}^{\mathbf{t}} \tag{4.5}$$

$$^{\text{w}}\boldsymbol{\theta}_{\text{b}}^{\text{VIO}} = {}^{\text{w}}\boldsymbol{\theta}_{\text{b}}^{\text{EKF}} + \boldsymbol{\eta}^{\boldsymbol{\theta}} \tag{4.6}$$

$$^{\text{w}}\mathbf{v}^{\text{VIO}} = {}^{\text{w}}\mathbf{R}_{\text{b}}\,{}^{\text{b}}\mathbf{v}^{\text{EKF}} + \boldsymbol{\eta}^{\mathbf{v}}, \tag{4.7}$$

where $^{\text{w}}\boldsymbol{\theta}_{\text{b}}$ is the orientation of the base in world frame expressed in $\mathfrak{so}(3)$. To tackle drift of the VIO in the height estimate, we estimate a height bias $\mathbf{b}_{\delta z}$ which is the difference between the measured height of the base link above the ground and the estimated height by the VIO.

Ground Height Measurements

The ground height is only measured when all the legs are considered as being in contact with the ground. The ground height is measured as the average of the height measurements by the different legs which is computed by forward kinematics, i.e.

$$^{\text{w}}z(\widetilde{\mathbf{q}}) := \left(\frac{1}{N_{\text{legs}}} \sum_{i=1}^{N_{\text{legs}}} - \text{fk}(\widetilde{\mathbf{q}}_i) \right)_3, \tag{4.8}$$

where N_{legs} is the number of legs in contact ($N_{\text{legs}} = 4$ in our case), and the operator $(\cdot)_k$ selects the k-th entry of a vector. Additionally, to exclude outliers and inaccurate measurements, the ground height is measured only after all the legs are considered as in contact with the ground for N_{standing} consecutive steps. The measurement model for the EKF is

$$\Delta z = \mathbf{b}_{\delta z} + \boldsymbol{\eta}^{\mathbf{b}_{\delta z}}, \tag{4.9}$$

with $\Delta z = \left({}^{\text{w}}\mathbf{t}_{\text{b}}^{\text{VIO}}\right)_3 - {}^{\text{w}}z(\widetilde{\mathbf{q}})$ and additive Gaussian noise $\boldsymbol{\eta}^{\mathbf{b}_{\delta z}}$.

4.3.3 Control Architecture

We use the non-linear MPC developed by Meduri et al. (2023) to control the robot. The MPC requires a contact plan as input and determines whole-body trajectories for the robot. Here, we only consider cyclic gaits, e.g., trotting and jumping, where the contact plan is automatically generated based on a command linear velocity (sidewards and forwards/backwards motion at a constant yaw angle). In this case, the Raibert heuristics is used to adapt the contact locations based on the feedback of the base linear velocity (Meduri et al., 2023). The framework generates centroidal trajectories using alternating direction method of multipliers approach and then a differential dynamic programming based kinematic optimizer is used to generate desired joint trajectories. Using an unconstrained inverse dynamics, the desired joint torques are computed and fed to the robot joint controller at 1 kHz.

4.4 Experiments

We evaluate our approach with the torque-controlled quadruped platform Solo12 by the Open Dynamic Robot Initiative (Grimminger et al., 2020) which we augment with an Intel Realsense T265 stereo-inertial sensor (see Figure 4.2a). The Solo12 robot weighs 2.5 kg and can carry an approximate payload of 1 kg. Stable trotting and jumping motions are generated by the MPC (Meduri et al., 2023) which uses our state estimate and calculates joint commands. The communication diagram is illustrated in Figure 4.1. The robot communicates joint measurements and targets via Ethernet with the robot control PC (Intel Xeon CPU E5-1680@3.40GHz, 8 cores) which runs Linux with a real-time kernel. A second vision PC (Intel Xeon CPU E5-1630@3.70GHz, 8 cores) computes visual-inertial odometry. The visual-inertial odometry result is communicated to the robot control PC via Ethernet. The Intel T265 camera provides 3-axis accelerometer and gyroscope data at 62.5 Hz and 200 Hz, respectively. The accelerometer data is upsampled to match the gyroscope measurement rate. The sensor also provides fisheye stereo images with a wide field of view (ca. 173 degrees) at a frame rate of 30 Hz. We use the calibration tools of Usenko et al. (2020) to calibrate the camera intrinsics and the extrinsic pose of the camera with respect to the IMU mounted on the robot, and the relative location of the IMU with respect to the robot base link. The orientation of the IMU with respect to the

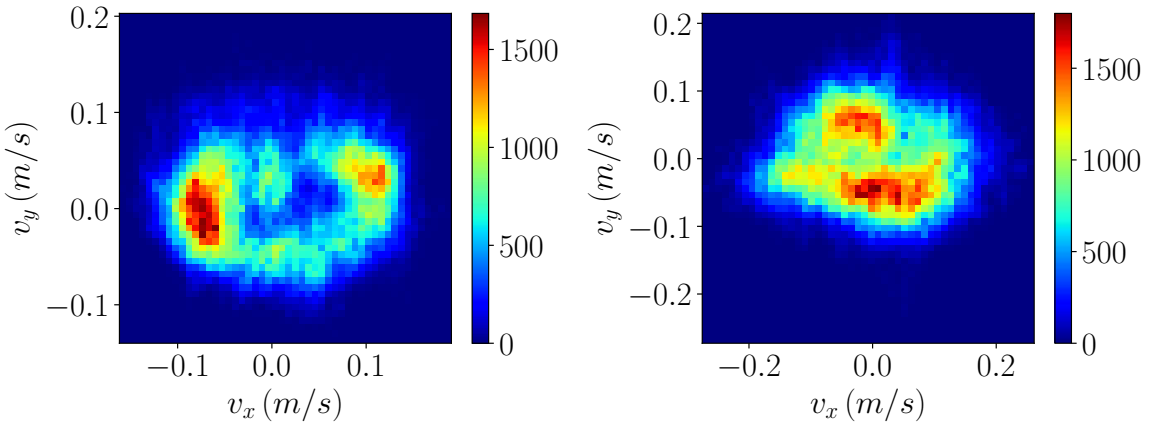


Figure 4.3: Distribution of horizontal linear velocity (m/s) of the base in experiment runs (left: trot, right: jump). The velocities are determined by fusing Vicon and IMU measurements in the EKF to obtain smoothed estimates. Min./max. are at the histogram boundaries. The Pearson correlation coefficients between estimated and control velocities are: Trot: 0.96 in x, 0.79 in y; Jump: 0.86 in x, 0.85 in y. According to the estimate, the robot follows the command partially due to competing MPC objectives, constraints, and Raibert heuristics for the contact plan (Trot: factor 0.49 in x, 0.32 in y. Jump: 1.23 in x, 0.72 in y).

robot base link is taken from the CAD model. For wheeled robots, it has been shown that the accelerometer bias is unobservable if the robot does not move sufficiently in yaw (Wu et al., 2017). Since the robot maintains a fixed yaw rotation, we fix the bias after a short initialization phase in which the robot is moved with 6-Degrees of Freedom before each run. We validate our approach in both indoor and outdoor environments. For indoor environments, we collect ground-truth data with a Vicon motion capture system at the rate of 800 Hz and upsample it to 1 kHz with the latest measurement. VIO at 30 Hz is denoted as vio in the following tables and figures, while VIO with IMU prediction at 200 Hz is denoted as vio+. For evaluation, both VIO versions are upsampled to the EKF rate of 1 kHz using the latest available estimates to demonstrate the performance of using these estimates as input for the controller. Note that our approach is not directly comparable to previous approaches such as Pronto (Camurri et al., 2020), since we propose a lightweight fusion method tailored to our control system. Our system uses VIO predictions to avoid computations for rolling back the EKF and to leave as much computation for the controller as possible. We use $N_{\text{contact}} = 1$ and $N_{\text{standing}} = 3$ in our experiments.

4.4.1 Evaluation Metrics

Since the control performance relies on the accuracy of state estimation, we evaluate the robot trajectory quantitatively using the relative pose error (RPE) metric (Z. Zhang and Scaramuzza, 2018) with various subtrajectories of time intervals (0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50) in seconds. We record 5 runs for each gait type (approx. 2 min per run for trotting and jumping) at varying target horizontal linear velocity using the EKF with augmented VIO measurements for state estimation. Figure 4.3 shows the distribution of the horizontal velocity as estimated by a ground-truth variant of the EKF which uses

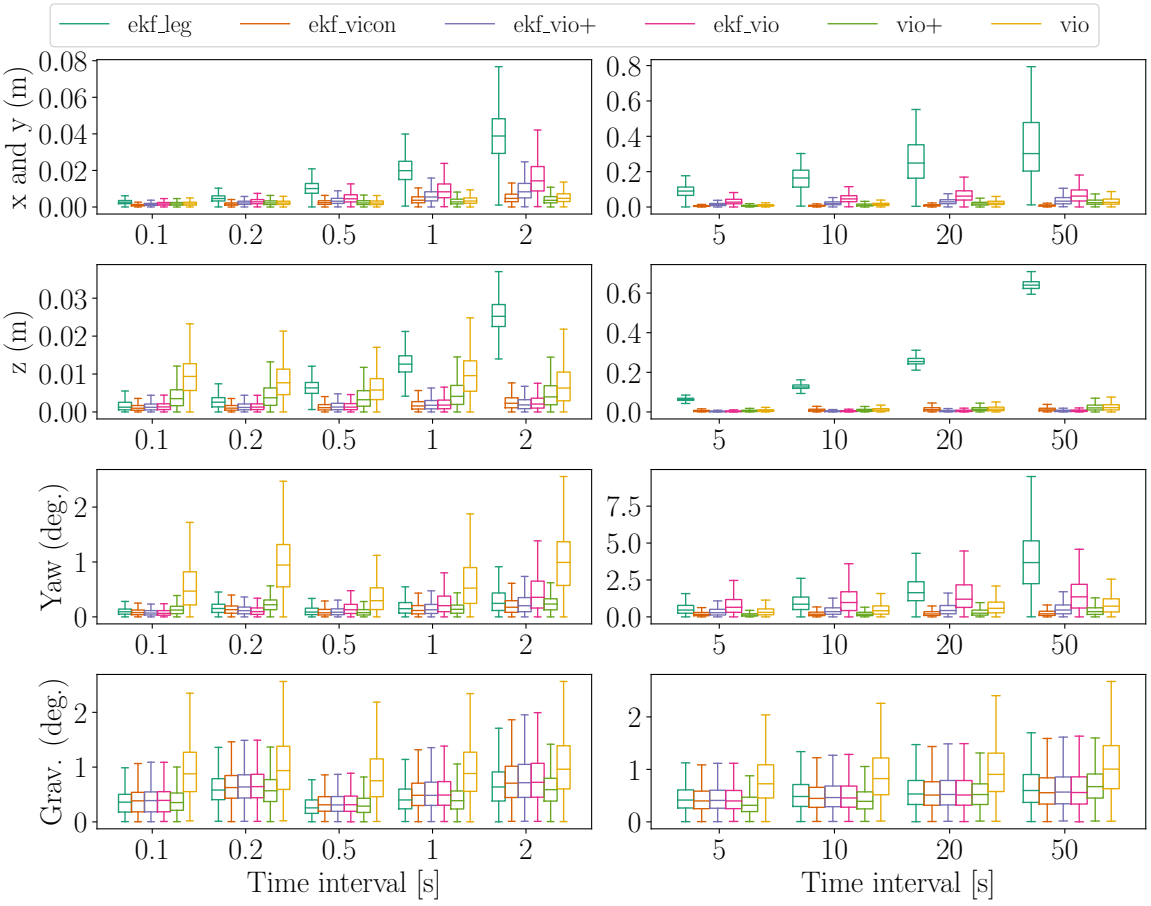


Figure 4.4: Trotting RPE for all time intervals.

IMU and Vicon measurements only. Besides the output of the EKF, additionally the estimates of VIO with and without predictions, all other input data to the EKF, and the Vicon ground-truth are recorded at 1 kHz to be able to assess the state estimate of other EKF variants on the runs. We compare variants and ablations of our approach including EKF with leg velocity measurements only ekf_leg, EKF with Vicon ekf_vicon, EKF with augmented VIO ekf_vio+, EKF with original VIO ekf_vio, augmented VIO and original VIO. For ekf_leg and ekf_vio+ we tuned separate covariance parameters for the EKF empirically. For the variants ekf_vicon and ekf_vio, we use the same parameters as ekf_vio+. We compute position errors (x, y, z) in meter and rotation errors (roll, pitch, yaw) in degree separately.

4.4.2 Trajectory Accuracy Evaluation for Indoor Experiments

Trotting Gait

In the trotting gait, at least two feet of diagonal legs are always in contact with the ground. The base link oscillates vertically with an amplitude of ca. 2 cm. The RPE evaluation is summarized in Table 4.1 and Figure 4.4. The EKF with only leg velocity

Table 4.1: Trotting trajectory accuracy in RPE.

		ekf_leg	ekf_vicon	ekf_vio+	ekf_vio	vio+	vio
x and y (m)	mean	0.151	0.011	0.022	0.041	0.016	0.018
	max	0.794	0.264	0.155	0.237	0.138	0.143
z (m)	mean	0.198	0.009	0.005	0.005	0.011	0.014
	max	0.746	0.098	0.057	0.058	0.076	0.107
yaw (deg)	mean	1.440	0.466	0.400	1.044	0.269	0.816
	max	9.532	6.707	2.563	7.588	1.670	3.086
roll and pitch (deg)	mean	0.593	0.615	0.632	0.637	0.545	1.117
	max	2.256	2.593	2.783	2.802	2.109	4.878

Table 4.2: Jumping trajectory accuracy in RPE.

		ekf_leg	ekf_vicon	ekf_vio+	ekf_vio	vio+	vio
x and y (m)	mean	0.229	0.013	0.035	0.090	0.021	0.022
	max	1.286	0.054	0.175	0.358	0.124	0.123
z (m)	mean	0.187	0.013	0.015	0.017	0.084	0.100
	max	0.806	0.086	0.107	0.131	0.610	0.646
yaw (deg)	mean	4.923	0.277	0.632	1.728	0.365	0.540
	max	32.928	2.848	3.754	9.765	2.262	3.236
roll and pitch (deg)	mean	0.686	0.900	0.923	0.860	0.711	1.646
	max	3.357	3.360	3.368	3.212	3.769	7.845

measurements (ekf_leg) shows significant drift in position and yaw orientation (avg. 0.333 m x-y-pos., 3.902 deg yaw at 50 s). Integrating predicted VIO measurements (ekf_-vio+) reduces this drift strongly, reducing the horizontal position and the yaw error to avg. 0.039 m and 0.552 deg at 50 s. We also observe that upsampling the VIO with IMU predictions improves the accuracy of pure VIO. Note that the data is further upsampled with the latest estimate to 1 kHz for reference to show its performance as potential input to the controller. For shorter time intervals below the gait cycle time (0.5 s), fusing leg odometry in the EKF variants improves the accuracy of the pure VIO variants. Fusing vio+ or ground truth with the leg odometry increases the roll pitch drift slightly towards ekf_leg, even though vio+ shows lower drift. At larger time intervals, the EKF finds a trade-off with high accuracy in horizontal position and orientation. VIO shows a small drift in height for trotting, which is also reflected by the RPE. Importantly, filtering leg kinematics and VIO allows for estimating the absolute height of the base with respect to the ground with high accuracy. Finally, we observe that the ekv_vicon does not have the lowest RPE, we attribute this to the noisier Vicon velocity measurement compared to the IMU.

Jumping Gait

In the jumping gait all four legs contact the ground at the same time during the landing and rebound phases. Each jump takes about 0.4 s with a height of 12 cm (robot base

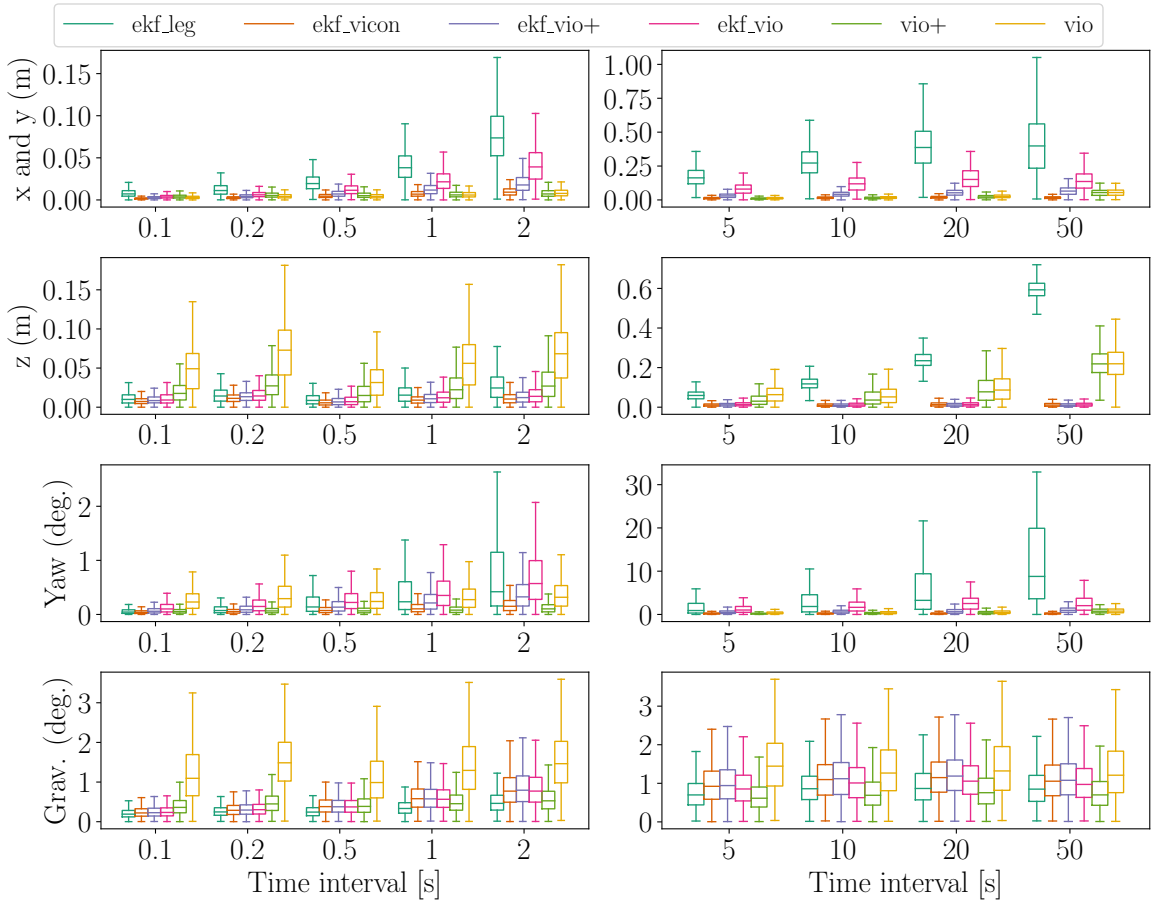


Figure 4.5: Jumping RPE for all time intervals.

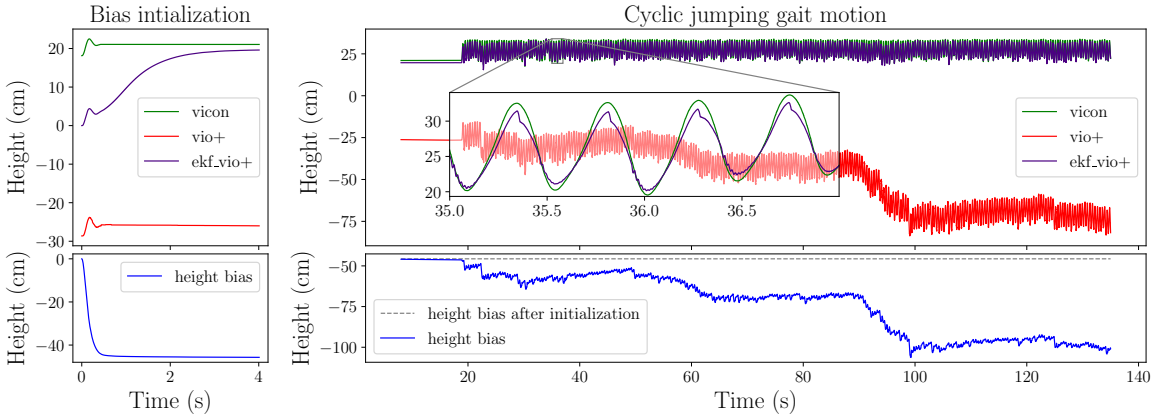


Figure 4.6: Height estimate of VIO with IMU predictions (vio+) and our approach (ekf_vio+) compared with ground-truth for jumping. Left: Initialization (standing), right: Jumping. The fast decline in the flight phase is due to false contact detection.

height change). In Table 4.2 and Figure 4.5 we provide RPE results. It can be seen that despite the agile motion, our approach ekf_vio+ can track the robot position and orientation. The pure VIO shows significant drift in height due to the difficulty of tracking and reconstructing keypoints in the close vicinity of the robot and the larger noisy IMU accelerations. This can be well compensated for by our EKF fusion approach

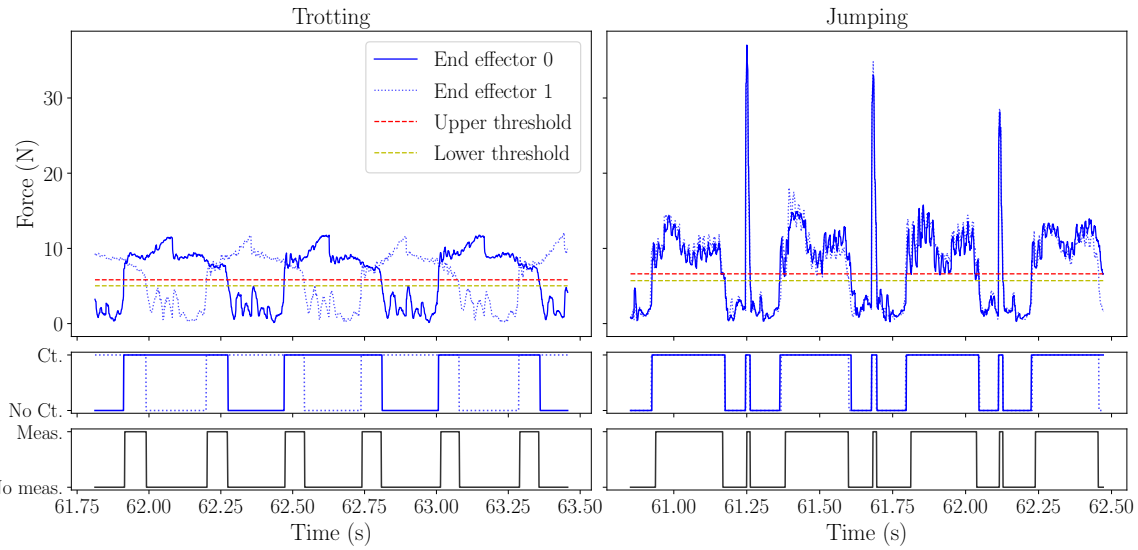


Figure 4.7: Contact detection for trotting and jumping gait for two endeffectors. The force estimate for jumping contains outliers that lead to false contact detection ($N_{\text{standing}} = 3$).

(ekf_vio+, see also Figure 4.6). The height bias estimate compensates for the differences and enables control for the jumping gait. The bias takes about 1 s to converge during the initialization phase in which the robot is standing before the jumping gait is started. The yaw and horizontal position drift of ekf_vio+ is slightly higher than in the trotting experiments. It clearly improves over the drift of ekf_leg.

4.4.3 Contact Detection and Height Measurement

We also provide a qualitative assessment of the contact detection in Figure 4.7. For the jumping gait, high acceleration of the legs while pulling them leads to high force estimates. Our experiments demonstrate that the system can be sufficiently robust against these spurious false measurements for trotting and jumping at moderate speeds. It is an interesting direction for future work to investigate more sophisticated ways of classifying contacts for dynamic gaits. This could potentially improve the accuracy and reliability of the system.

By setting higher contact duration thresholds ($N_{\text{contact}} = N_{\text{standing}} = 20$) for leg odometry and ground height measurements, the false contact detection can be avoided. However, this also decreases the accuracy of the filter. We observe that mean RMSE increases from 0.015 m to 0.038 m for z and from 0.632 deg to 0.869 deg for yaw, while ekf_leg fails.

4.4.4 Outdoor Experiment

We also test our system outdoors in challenging environments, specifically on asphalt and grass with slight slopes (see Figure 4.8). The tests include both trotting and jumping gaits, as well as transitions between them. Moreover, we also experimented with varying

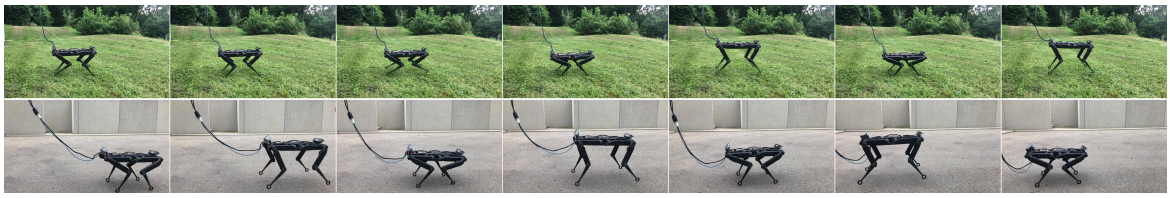


Figure 4.8: Solo12 in outdoor experiments. Top: Trotting and jumping on grass. Bottom: Jumping from right to left on asphalt.

control speeds. A video showcasing the experiments is provided as supplemental material to the corresponding conference publication (Dhédin et al., 2023). Across all scenarios, the control pipeline demonstrated reliable performance, supported by our state estimation method, for both gaits on the different terrains.

4.5 Conclusion

In this chapter, we present a lightweight EKF-based framework that fuses VIO estimates with leg odometry to calculate the pose and velocity of the robot at high frequency. To compensate for the delay and low rate from VIO, we propose to use IMU predictions to update the VIO state estimate such that the output of VIO is streamed at IMU rate with a significantly smaller delay and higher rate. Additionally, we compensate for the drift of the height estimate by measuring height based on leg kinematics and contact detection. We validate our approach with real-world experiments in both indoor and outdoor environments. The quantitative results of our experiments indicate that the low latency VIO with IMU prediction improves the accuracy of the EKF state estimate and the height measurement can prevent drift of the height estimate despite the existence of outliers in contact detection for the jumping gait. We also provide qualitative results for our system in challenging outdoor experiments. In these examples, our approach can estimate the robot state and perform trotting and jumping gaits including gait switching on different terrains. A limitation of our approach is the assumption of a hard and flat terrain for contact detection and height estimation. In future work, we aim to increase the robustness of our method and integrate terrain measurements to enable trajectory planning and control on complex terrain.

Online Adaptation of Kinematic Model with Visual-Inertial Odometry

5

Declaration of Contributions

The contents of this chapter are based on the peer-reviewed journal publication

 H. Li and J. Stückler (2022). 'Visual-Inertial Odometry with Online Calibration of Velocity-Control Based Kinematic Motion Models'. In: *IEEE Robotics and Automation Letters (RA-L)*. DOI: 10.1109/LRA.2022.3169837 (H. Li and Stückler, 2022)

with the following co-author contributions:

	Scientific Ideas	Data Generation	Analysis & Interpretation	Paper Writing
Haolong Li	60%	100%	80%	80%
Jörg Stückler	40%	0%	20%	20%

Jörg Stückler conceived the idea of integrating a velocity-based kinematics factor into visual-inertial odometry. Haolong Li proposed to use the velocity-control-based inverse motion model and Jörg Stückler proposed to use kernel function to aggregate the control inputs. Haolong Li implemented the method, collected the data, and performed the experiments. Haolong Li and Jörg Stückler wrote the paper.

Compared to the conference publication, this chapter contains unified notation and some reformulated and reorganized paragraphs. The section on Observability Analysis was originally part of the appendix and supplementary material of the journal publication. Figure 5.2 has been redrawn to ensure consistent notation.

5.1 Introduction

In recent years, visual-inertial odometry (VIO) has seen tremendous progress (e.g. Campos et al. (2021), Leutenegger et al. (2015), Mourikis and Roumeliotis (2007), and Usenko et al. (2020)), driven by the many potential applications of such technology for augmented/virtual reality and autonomous robots, in particular flying robots. Surprisingly, for wheeled robots, VIO is not trivial to employ due to observability limitations for planar linear motions (Wu et al., 2017). This can be alleviated by integrating motion model constraints into the state estimate. A popular approach in the literature is to use wheel odometer measurements to this end (see e.g. F. Ma et al. (2019), Wu et al. (2017), and Yang and Huang (2019)).

In this chapter, we take a conceptually different approach by integrating a velocity-control-based kinematic motion model which does not rely on wheel encoders. By integrating the model into the state estimation, the model parameters such as the relative position of the sensor on the robot or the offset between model and real robot can be calibrated online. Differently to wheel odometry based models, velocity-control-based models can be directly used for downstream tasks such as model-predictive motion control and planning (Kavraki et al., 1996; Kuwata et al., 2008; LaValle and Kuffner, 1999). We base our method on a non-linear optimization-based approach (Usenko et al., 2020) to visual-inertial odometry which optimizes state variables such as sensor pose and velocity, IMU biases, and keypoint map in a window of recent frames. Old frames are marginalized in a proper probabilistic way to maintain the prior observations as prior knowledge. The IMU measurements are preintegrated into relative motion measurements between frames. In this framework, we include a velocity-based motion model which models the motion of a wheeled robot in a plane based on linear forward and rotational velocity controls. For accurate integration of the measurements and controls, an accurate calibration of the sensor placement with respect to the drive, the time synchronization of the controls relative to the visual and inertial measurements, and an identification of the effect of control commands for the underlying low-level robot motion controller on actual executed motion are required. To model the unknown properties of the controller, we aggregate the raw control commands with a kernel function. We add parameters for the kernels and the placement (extrinsic pose) of the sensor on the robot to the estimation problem. The parameters are calibrated online in the non-linear optimization framework.

We evaluate our approach on data obtained with a mobile robot in several sizes of environments. We demonstrate that incorporating the velocity-control-based kinematic motion model improves the accuracy and robustness of the VIO estimate. Moreover, we provide results on the prediction accuracy of our online calibrated model for reference for model-based control and planning approaches.

In summary, our contributions are:

- We propose a novel visual-inertial odometry approach for wheeled robots which includes a velocity-control-based kinematic motion model into the state estimate.
- The parameters of the motion model are calibrated online with the VIO estimate.
- We demonstrate that inclusion of the motion can improve the VIO estimate in various indoor environments of different sizes. We also provide evaluation of the prediction accuracy of the calibrated model.

Our model can be an alternative to wheel-odometry based methods when a velocity-control-based model should be directly calibrated for use in model-predictive control and motion planning methods.

5.2 Related Work

Motion estimation by fusing odometry, IMU and camera data has recently spurred significant interest in the robotics community due to its applications for inertial navigation systems in service robotics and autonomous driving.

5.2.1 Inertial and Inertial-Wheel Odometry

Various recent approaches combine IMU and wheel odometry. Brossard et al. (2020) suggest an extended Kalman filter (EKF) based approach which uses deep learning to predict the noise properties in an EKF framework which fuses IMU measurements to predict motion. In Brossard and Bonnabel (2019), deep kernel Gaussian Process models are learned for the motion and observation models which are used to fuse IMU and wheel odometry measurements in an EKF framework. Brossard et al. (2019) propose to estimate the motion from IMU and odometry measurements using an recurrent neural network which detects different motion profiles in an EKF framework. These approaches, however, do not use the complementary strengths of visual measurements and do not provide a forward model.

5.2.2 Visual-Inertial-Wheel Odometry

Cameras provide complementary information to inertial measurement units for motion estimation. For general motions, 3-DoF linear acceleration and rotational velocity measurements make roll and pitch orientation observable relative to the gravity direction. However, double integration of the linear acceleration requires accurate estimation of biases (offsets) in the measurements and makes linear position estimation prone to drift. Similarly, the yaw orientation around the gravity direction is not observable and prone to drift due to noisy and biased gyroscope measurements. Visual measurements provide a reference for pose estimation to a local 3D map of the environment which is concurrently built with the pose estimates in VIO approaches. By this, all DoFs become observable, while the IMU provides high frame-rate measurements which improve the accuracy between images.

The Multi-State-Constrained Kalman filter (Mourikis and Roumeliotis, 2007) for VIO has been recently extended to incorporate wheel odometry and overcome observability issues of monocular visual-inertial odometry on wheeled robots in VINS on wheels (Wu et al., 2017). The authors analyze the observability of monocular visual-inertial navigation systems on a mobile robot platform and show that for specific motions, scale, and 3-DoF rotation become unobservable. They also show that adding wheel encoder measurements makes scale observable. The approach does not calibrate the motion model parameters online like our method. In our setting, scale is already observable through the fixed calibrated baseline of our stereo camera.

F. Ma et al. (2019) adopt the VINS on wheels approach and extend it with an Ackerman drive model. Jung et al. (2020) add GPS measurements directly to VINS on wheels to make position observable. Yang and Huang (2019) analyze observability for VINS on wheels with line and plane observations. Another approach concurrently estimates the wheel slippage with VIO (Dang et al., 2018). More closely related to our method is the approach by Lee et al. (2020) which investigates online calibration of the wheel odometry parameters and analyzes observability of the calibration parameters for different constraint motion scenarios. In contrast, we employ a non-linear optimization based approach for visual-inertial odometry and incorporate an inverse motion model for constraints.

Some approaches integrate wheel odometry into non-linear optimization based approaches. J. Liu et al. (2019) develop online calibration of the extrinsic pose between camera, IMU and odometer. J. Liu et al. (2021) propose a novel initialization approach which corrects the initial state estimates after the first turning motion to handle the unobservability of the calibration parameters for straight motions. Y. Chen et al. (2019) calibrate visual-inertial-wheel odometry offline. The approach in Z. Zhang et al. (2018) integrates smooth motion manifold constraints. Zheng and Y.-H. Liu (2019) incorporate a planar motion constraint for the mobile base but allow small deviations from this motion in 6-DoF for the camera-IMU system. Also, different from ours, these approaches use wheel encoders to measure odometry. We propose a new approach which incorporates an inverse motion model into optimization-based visual-inertial odometry and calibrates the model online including extrinsic pose and time synchronization.

5.2.3 Learning Motion Models for Control

Optimal control approaches typically rely on action-conditional motion models which use them as forward models to plan towards goals. One recent example is the approach of Williams et al. (2016) which learns a dynamic model offline from GPS. In Kabzan et al. (2019) a sensor-based localization method using LiDAR, optical speed sensors and INS is used to provide feedback for learning deviations from an analytic dynamic model. In our approach, we tightly integrate a kinematic motion model in a visual-inertial odometry which allows for calibrating the model online.

5.3 Method

We integrate a velocity-control-based mobile robot motion model into visual-inertial odometry and optimize the parameters of the motion model concurrently with the camera trajectory and bias parameters of the IMU. The motion model further constrains the camera motion estimate. The calibrated motion model could be useful for model-predictive motion control and path planning.

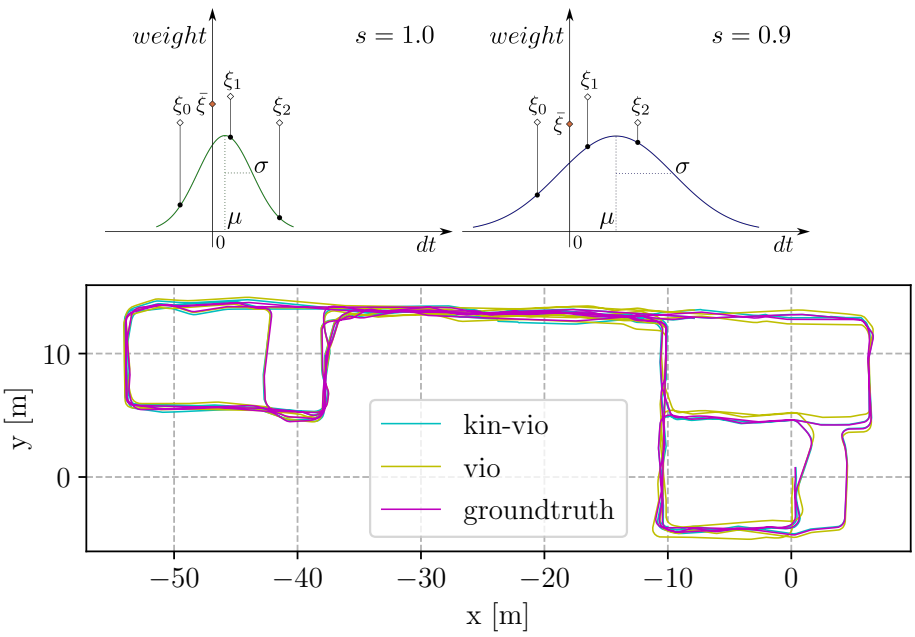


Figure 5.1: Weighted aggregation of effective controls and an exemplary result of motion-constrained VIO (large-01). Top: Time delays of controls and hardware restrictions (such as acceleration limits) can be handled implicitly by weighting and averaging the commands in a window with an RBF kernel. We optimize for the mean value μ, the variance σ and the scale s to shift the kernel and change its shape. Bottom: Our motion-constrained VIO approach achieves smaller deviation with respect to the ground-truth. The pure VIO result is shown in yellow, our kinematics-constraint VIO estimate in cyan, ground-truth in purple.

5.3.1 Visual-Inertial Odometry

We extend the non-linear optimization-based visual-inertial odometry approach in Usenko et al. (2020). The approach uses a KLT-based keypoint tracking frontend to track the camera motion from frame to frame. Keyframes are extracted along the camera trajectory and the keypoint tracks generate landmarks in the keyframes with corresponding point measurements. Only a set of recent keyframes and regular frames is optimized (3 regular frames and 7 keyframes in our experiments). Older regular frames and keyframes are marginalized from the optimization window and their information serves as a prior.

Formally, we estimate the state $\mathbf{s}_{\text{VIO}} = \left({}^{\text{w}}\mathbf{T}_{\text{i}}, {}^{\text{w}}\mathbf{v}, \mathcal{L}, \mathbf{b}_{\text{g}}, \mathbf{b}_{\text{a}}\right)$, where ${}^{\text{w}}\mathbf{T}_{\text{i}} \in \text{SE}(3)$ represents the pose of the sensor frame i expressed in the VIO's world frame w, and ${}^{\text{w}}\mathbf{v} \in \mathbb{R}^3$ denotes the linear velocity of the sensor with respect to the world, expressed in the world frame. Additionally, we also optimize the set of landmarks $\mathcal{L}$ and the IMU biases $\mathbf{b}_{\text{g}}, \mathbf{b}_{\text{a}}$. This state can be estimated by minimizing the objective function:

$$\mathbf{s}^*_{\text{VIO}} = \arg\min E_{\text{img}}(\mathbf{s}) + E_{\text{IMU}}(\mathbf{s}) + E_{\text{marg}}(\mathbf{s}), \tag{5.1}$$

where $E_{\text{img}}(\mathbf{s})$ is the squared sum of the reprojection residuals from the image data and $E_{\text{IMU}}(\mathbf{s})$ contains the relative pose information from the IMU sensor, and $E_{\text{marg}}(\mathbf{s})$ contains the marginalization prior information. For further details on the visual-inertial odometry method, we refer the reader to Section 2.3 and Usenko et al. (2020).

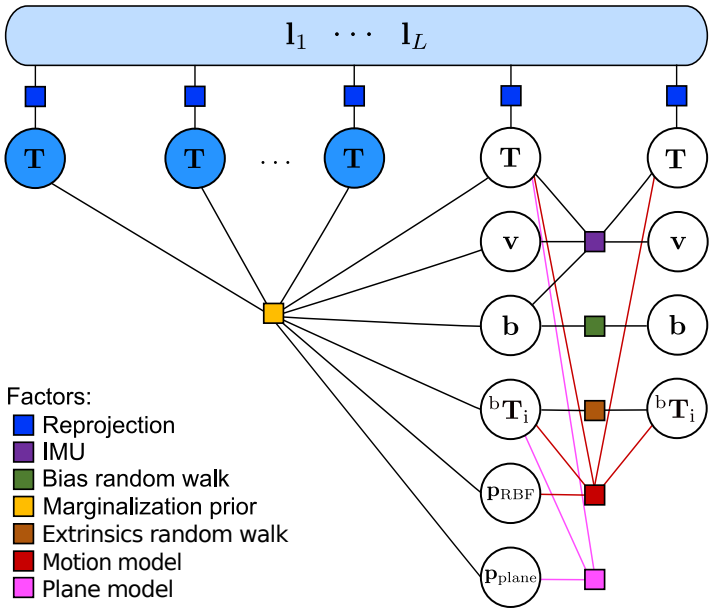


Figure 5.2: Factor graph of the proposed method, where $^{b}\mathbf{T}_i$ is the extrinsic pose. The motion factor consists of frame poses, extrinsic poses and RBF parameters $\mathbf{p}_{RBF}$, and the plane factor includes the frame poses, extrinsic poses and plane parameters $\mathbf{p}_{plane}$.

5.3.2 Velocity-Control-Based Motion Model

We use a velocity-control-based motion model (Thrun et al., 2005) and assume that the robot can be controlled by a control command $\mathbf{u} = (v, \omega)^\top$ through a linear velocity $v \in \mathbb{R}$ in forward direction and a rotational velocity $\omega \in \mathbb{R}$. The motion model propagates the robot pose $\mathbf{P}_t \in \mathrm{SE}(2)$ at time t on the ground plane with the control command to the pose at time $t' = t + \Delta t$ as: $\mathbf{P}_{t'} = \mathbf{P}_t \exp\left(\Delta t\, \widehat{\boldsymbol{\xi}}\right)$, where

$$\widehat{\boldsymbol{\xi}} = \begin{pmatrix} 0 & -\omega & v \\ \omega & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{5.2}$$

is the corresponding *Lie algebra* $\mathfrak{se}(2)$ with $\boldsymbol{\xi} = (v, 0, \omega)^\top \in \mathbb{R}^3$. The exponential mapping for SE(2) group is:

$$\exp\left(\Delta t\, \widehat{\boldsymbol{\xi}}\right) = \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta\theta \end{pmatrix} = \begin{pmatrix} \cos(\omega\Delta t) & -\sin(\omega\Delta t) & \frac{v}{w}\sin(\omega\Delta t) \\ \sin(\omega\Delta t) & \cos(\omega\Delta t) & \frac{v}{w} - \frac{v}{w}\cos(\omega\Delta t) \\ 0 & 0 & 1 \end{pmatrix}. \tag{5.3}$$

This exponential mapping finds the relative SE(2) motion for constant velocities $\boldsymbol{\xi}$ over time duration Δt. The logarithm mapping $\widehat{\boldsymbol{\xi}} = \log(\mathbf{P})$ of SE(2) is the inverse of the exponential mapping and maps relative poses $\mathbf{P} \in \mathrm{SE}(2)$ to Lie algebra elements in $\widehat{\boldsymbol{\xi}} \in \mathfrak{se}(2)$. The *hat* operator $\widehat{(\cdot)}$ maps 3D twist coordinate vectors to twists in $\mathfrak{se}(2)$.

5.3.3 Motion Model Residuals

We incorporate the velocity-control motion model into the visual-inertial odometry framework in order to calibrate the parameters of the model and improve the robustness of the VIO. There are basically two choices to form residuals with a motion model: relative pose constraint (forward) or velocity constraint (inverse). Mathematically the forward and inverse model residuals are equivalent to each other. Both kinds of residuals achieve similar results, while the Jacobian matrix of the forward model is computationally more expensive. For our motion constraint, we treat the velocity controls as measurements and assume the measurement noise comes from the velocity controls. In this case, using the inverse model residuals is simpler while using the forward model residuals requires error propagation with linear approximation of the exponential mapping.

Velocity commands are executed by the robot in the robot base frame whose pose relative to the world frame is denoted by ${}^{\mathrm{w}}\mathbf{T}_{\mathrm{b}} \in SE(3)$ (transforming coordinates from base b to world frame w). In the base frame, the x-axis points in forward driving direction, while the z-axis points upward and is the axis of rotational robot motion. The VIO provides pose estimates of the body frame of the IMU-camera sensor in the world frame ${}^{\mathrm{w}}\mathbf{T}_{\mathrm{i}}$. The sensor is placed rigidly on the robot at a relative pose ${}^{\mathrm{b}}\mathbf{T}_{\mathrm{i}} \in SE(3)$ to the robot base frame. To quantify the relative motion ${}^{\mathrm{b},t}\mathbf{T}_{\mathrm{b},t'}$ of the robot base frame from times t to t' of subsequent image frames, we can hence determine ${}^{\mathrm{b},t}\mathbf{T}_{\mathrm{b},t'} = {}^{\mathrm{b},t}\mathbf{T}_{\mathrm{i},t}\,{}^{\mathrm{w}}\mathbf{T}_{\mathrm{i},t}^{-1}\,{}^{\mathrm{w}}\mathbf{T}_{\mathrm{i},t'}\,{}^{\mathrm{b},t'}\mathbf{T}_{\mathrm{i},t'}^{-1}$. The rotation $\Delta\theta$ around the z-axis of the base frame is calculated from the relative rotation ${}^{\mathrm{b},t}\mathbf{R}_{\mathrm{b},t'}$ in ${}^{\mathrm{b},t}\mathbf{T}_{\mathrm{b},t'}$ as the z-component of $\mathrm{Log}\left({}^{\mathrm{b},t}\mathbf{R}_{\mathrm{b},t'}\right)$. The translational motion $(\Delta x, \Delta y)^\top$ in the x-y-plane is determined from the corresponding entries of ${}^{\mathrm{b},t}\mathbf{T}_{\mathrm{b},t'}$. The estimated twist is

$$\boldsymbol{\zeta} = \frac{1}{\Delta t}\mathrm{Log}\begin{pmatrix} \cos(\Delta\theta) & -\sin(\Delta\theta) & \Delta x \\ \sin(\Delta\theta) & \cos(\Delta\theta) & \Delta y \\ 0 & 0 & 1 \end{pmatrix}, \tag{5.4}$$

where $\Delta t = t' - t$. We compute the motion model residual as

$$\mathbf{r}_{\xi} = \boldsymbol{\zeta} - \bar{\boldsymbol{\xi}}, \tag{5.5}$$

where $\bar{\boldsymbol{\xi}}$ represents the effective control, which will be described in the next section. This residual implicitly measures the difference between the state estimates and the motion model prediction.

5.3.4 Effective Control Command

In practice, the real action of the robots differs from the received control commands due to effects such as time offsets and properties of low-level controllers.

Typically, control inputs and image frames are not synchronized but run asynchronously and often also at different rates. In our experiments, the control rate is 15 Hz and is

lower than the 30 Hz image frame rate which is also used to update the VIO estimate. Moreover in the real world, a delay exists between the control command sent by the controller and the control command executed by the robot.

The robot physical hardware acceleration limits and internal controllers also prevent the robot from directly executing the control command even if the delay is known. To mitigate this difference and build a meaningful residual, we estimate an effective control $\bar{\boldsymbol{\xi}}_t$ at arbitrary time t, e.g. at the time of an image frame, from a window of most recent commands. We average a window of recent control commands with weights determined by a radial basis function (RBF) kernel (see Figure 5.1) for the translational and rotational parts separately:

$$\bar{\boldsymbol{\xi}}_t = \begin{pmatrix} s_{\text{lin}} \dfrac{\sum_{\tau \in \mathcal{W}_t} \exp\left(-\frac{\|d\tau - \mu_{\text{lin}}\|^2}{2\sigma_{\text{lin}}^2}\right) v_\tau}{\sum_{\tau \in \mathcal{W}_t} \exp\left(-\frac{\|d\tau - \mu_{\text{lin}}\|^2}{2\sigma_{\text{lin}}^2}\right)} \\ 0 \\ s_{\text{ang}} \dfrac{\sum_{\tau \in \mathcal{W}_t} \exp\left(-\frac{\|d\tau - \mu_{\text{ang}}\|^2}{2\sigma_{\text{ang}}^2}\right) w_\tau}{\sum_{\tau \in \mathcal{W}_t} \exp\left(-\frac{\|d\tau - \mu_{\text{ang}}\|^2}{2\sigma_{\text{ang}}^2}\right)} \end{pmatrix}. \tag{5.6}$$

Here $\mathcal{W}_t$ is a window of N control commands indexed by their times τ at or before time t and $d\tau := t - \tau$. For an image frame at time t, the window typically spans the N control commands that have occurred before the frame. We optimize for μ and σ and scale factor s of both linear and angular parts as global parameters together with the VIO states. The RBF parameters are summarized in the state variables $\mathbf{p}_{\text{RBF}}$. In the experiments, we demonstrate that the optimized RBF kernel can be used for motion prediction.

5.3.5 Motion-Model-Based Error Function

The robot body can vibrate during operation, the extrinsic pose ${}^{\text{b}}\mathbf{T}_{\text{i}}$ is thus modeled as a time-variant state. For K factors within the optimization window this kinematics-based objective can be summarized as

$$E_{\text{kin}} = \sum_k^K \mathbf{r}_{\xi,k}^\top \boldsymbol{\Sigma}_{\xi,k}^{-1} \mathbf{r}_{\xi,k} + \sum_k^K \mathbf{r}_{\text{extr},k}^\top \boldsymbol{\Sigma}_{\text{extr},k}^{-1} \mathbf{r}_{\text{extr},k}, \tag{5.7}$$

where $\boldsymbol{\Sigma}_{\xi,k}^{-1}$ is the diagonal weight matrix for the motion model residual, $\mathbf{r}_{\text{extr},k}$ is the difference between two adjacent extrinsic pose estimates and $\boldsymbol{\Sigma}_{\text{extr},k}^{-1}$ is the diagonal weight matrix.

The VIO system takes the image frame at time t' and the raw controls up to the time t of the previous frame as input for the optimization which also calibrates the RBF parameters for the motion model. The controls can be generated by manual control or an automatic high-level controller such as model-predictive control for path tracking.

In our experiments, we use manual control commands as input and calibrate the RBF parameters online. A high-level controller would potentially require extrapolation of the last state estimate at the image rate to the current control time using the previous controls and the motion model.

5.3.6 Plane Motion Constraint

We exploit prior knowledge that our robot moves on flat ground in indoor environments and add a stochastic plane constraint (Wu et al., 2017) for the robot pose. The plane parameter $\mathbf{p}_{\text{plane}}$ includes a 2-Degrees of Freedom quaternion ${}^{\text{g}}\mathbf{q}_{\text{w}}$ representing the plane orientation and a scalar ${}^{\text{g}}d_{\text{w}}$ which represents the distance between the ground plane to the world frame origin. The residual is

$$\mathbf{r}_{\text{plane}} = \begin{pmatrix} \left(\mathbf{R}({}^{\text{g}}\mathbf{q}_{\text{w}})\,{}^{\text{w}}\mathbf{R}_{\text{i}}\,{}^{\text{b}}\mathbf{R}_{\text{i}}^{\top}\mathbf{e}_3\right)_{1,2} \\ {}^{\text{g}}d_{\text{w}} + \mathbf{e}_3^{\top}\mathbf{R}({}^{\text{g}}\mathbf{q}_{\text{w}})({}^{\text{w}}\mathbf{t}_{\text{i}} - {}^{\text{w}}\mathbf{R}_{\text{i}}\,{}^{\text{b}}\mathbf{R}_{\text{i}}^{\top}\,{}^{\text{b}}\mathbf{t}_{\text{i}}) \end{pmatrix}, \tag{5.8}$$

with $\mathbf{e}_3 = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^{\top}$. The plane motion error term becomes

$$E_{\text{plane}} = \sum_{l} \mathbf{r}_{\text{plane},l}^{\top} \boldsymbol{\Sigma}_{\text{plane},l}^{-1} \mathbf{r}_{\text{plane},l} \tag{5.9}$$

with the diagonal covariance matrix $\boldsymbol{\Sigma}_{\text{plane}}$. The stochasticity of the constraint allows for handling vibrations of the robot.

5.3.7 Visual-Inertial Odometry with Motion Model Constraints

We integrate the above introduced calibration parameters of the constraints as additional variables into the visual-inertial odometry. As shown in Figure 5.2, the state of each frame in our optimization framework becomes $\mathbf{s}_{\text{kin-vio}} = (\mathbf{s}_{\text{VIO}}, {}^{\text{b}}\mathbf{T}_{\text{i}}, \mathbf{p}_{\text{RBF}}, \mathbf{p}_{\text{plane}})$, which comprises the VIO state, the base frame to sensor frame extrinsic pose ${}^{\text{b}}\mathbf{T}_{\text{i}}$, and the global variables including the RBF parameters and the plane parameters. The full objective function can be summarized as

$$E_{\text{kin-vio}} = E_{\text{VIO}} + E_{\text{kin}} + E_{\text{plane}}. \tag{5.10}$$

During optimization the extrinsic poses will be marginalized like linear velocity and IMU biases while the global variables are kept and their linearization point is fixed once the first connected state is marginalized.

5.3.8 Observability Analysis

In this subsection, we discuss the observability properties of the state variables in our kinematic motion-constrained VIO method. By definition, observability of a system determines if the initial state can be uniquely inferred from a finite amount of measurements (Barfoot, 2017). Observability can be analyzed based on the underlying state-space model, independent of the estimator's implementation (Wu et al., 2017). Following the derivations from Hesch et al. (2014) and Wu et al. (2017), we analyze the observability of the linearized state-space model for our method.

Before delving into the details, we first define several key concepts to ensure that the analysis in this part is self-contained. Differing from the factor graph framework introduced in Section 2.3, we now approach the VIO problem from another perspective, namely the discrete-time state-space model. For a linear system, we define the following transition and observation models:

$$\mathbf{s}_{k+1} = \boldsymbol{\phi}_{k+1,k}\mathbf{s}_k + \mathbf{m}_k \tag{5.11}$$

$$\mathbf{x}_k = \mathbf{J}_k\mathbf{s}_k + \mathbf{n}_k, \tag{5.12}$$

where $\mathbf{s}$ represents the system state, $\boldsymbol{\phi}_{k+1,k}$ the transition matrix, $\mathbf{J}_k$ the observation matrix and $\mathbf{m}_k$, $\mathbf{n}_k$ are transition and measurement noises, respectively. In the context of VIO, IMU measurements typically are used to propagate the state and to compute the transition matrix (Wu et al., 2017), while camera measurements serve as observations and are used to compute the observation matrix. For our linearized VIO system, $\mathbf{J}_k$ corresponds to the Jacobians of vision-based factors computed with the state estimate at time k. The observability matrix $\mathbf{M}$ is defined as:

$$\mathbf{M} = \begin{pmatrix} \mathbf{J}_1 \\ \mathbf{J}_2\boldsymbol{\phi}_{2,1} \\ \vdots \\ \mathbf{J}_k\boldsymbol{\phi}_{k,1} \end{pmatrix} \tag{5.13}$$

with $\boldsymbol{\phi}_{k,1} = \boldsymbol{\phi}_{k,k-1}\ldots\boldsymbol{\phi}_{2,1}$. If the observability matrix is full column rank, the linearized VIO system is considered observable.

Hesch et al. (2014) demonstrate that VIO has four unobservable directions: the three global translations and one global yaw direction in the general case. Wu et al. (2017) further show that the scale is unobservable if the camera has constant acceleration, and the roll and pitch become also unobservable if there is no rotation. Since we use a stereo camera, the scale of our VIO system is observable. In addition to the original VIO state, we also have extrinsic pose state, RBF parameters, and plane parameters in our model. As we employ the same plane constraint as detailed in Wu et al. (2017), our focus here is on analyzing the observability of the extrinsic pose state and RBF parameters. Furthermore, we demonstrate that the plane's orientation can make the global orientation observable.

We treat the kinematic motion constraint and the planar constraint as observations. The VIO state transition is given by the IMU propagation model. We use the constant propagation model for the other state variables. The transition matrix of our method becomes:

$$\bar{\mathbf{\Phi}}_{k,1} = \begin{pmatrix} \mathbf{\Phi}_{k,1} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}, \tag{5.14}$$

where $\mathbf{\Phi}_{k,1}$ is the linearized transition matrix for VIO state from time 1 to k from the IMU propagation, detailed in Hesch et al. (2014).

Observability of Extrinsic Pose

The observability of VIO state and extrinsic pose has been studied in previous work (Lee et al., 2020), where a velocity-based forward kinematic motion model is integrated into the VIO. In our approach, we use an inverse kinematic motion model. Here, we show that the observability of integrating the inverse kinematic model is the same as using the forward kinematic model.

Given the relative pose between two frames ${}^{\mathrm{b},t}\mathbf{T}_{\mathrm{b},t'}$, we can compute the corresponding SE(2) pose ${}^{\mathrm{b},t}\mathbf{P}_{\mathrm{b},t'}$ in the horizontal plane consisting of the rotational part ${}^{\mathbf{b},t}\theta_{\mathbf{b},t'} \in \mathbb{R}$ and the translational part ${}^{\mathrm{b},t}\mathbf{p}_{\mathrm{b},t'} \in \mathbb{R}^2$ as discussed in Subsection 5.3.3. The effective control input $\boldsymbol{\xi} = (v, 0, w)^\top$ consists of linear and angular velocity. With ${}^{\mathrm{b},t}\bar{\mathbf{P}}_{\mathrm{b},t'} = \mathrm{Exp}(\Delta t \boldsymbol{\xi})$, the forward kinematic model, the residual and its derivatives can be written as

$$\mathbf{r}_{\mathrm{fwd}} = f({}^{\mathrm{b},t}\mathbf{p}_{\mathrm{b},t'}, {}^{\mathrm{b},t}\bar{\mathbf{P}}_{\mathrm{b},t'}) = \begin{pmatrix} {}^{\mathbf{b},t}\mathbf{p}_{\mathbf{b},t'} - {}^{\mathbf{b},t}\bar{\mathbf{p}}_{\mathbf{b},t'} \\ {}^{\mathbf{b},t}\theta_{\mathbf{b},t'} - w\Delta t \end{pmatrix} \tag{5.15}$$

$$\frac{\partial \mathbf{r}_{\mathrm{fwd}}}{\partial {}^{\mathrm{b},t}\mathbf{P}_{\mathrm{b},t'}} = \frac{\partial f(.)}{\partial {}^{\mathrm{b},t}\mathbf{P}_{\mathrm{b},t'}}, \tag{5.16}$$

while the residual of the inverse model and its derivatives are

$$\mathbf{r}_{\mathrm{inv}} = \mathrm{Log}({}^{\mathrm{b},t}\mathbf{P}_{\mathrm{b},t'}) - \Delta t \boldsymbol{\xi} \tag{5.17}$$

$$\frac{\partial \mathbf{r}_{\mathrm{inv}}}{\partial {}^{\mathrm{b},t}\mathbf{P}_{\mathrm{b},t'}} = \frac{\partial \mathrm{Log}(.)}{\partial {}^{\mathrm{b},t}\mathbf{P}_{\mathrm{b},t'}}. \tag{5.18}$$

While in the forward kinematic model, the derivative with respect to ${}^{\mathrm{b},t}\mathbf{P}_{\mathrm{b},t'}$ is an identity matrix, the derivative in the inverse model is the Jacobian of the logarithmic mapping. Because this Jacobian is an invertible matrix by definition (Deray and Solà, 2020), when we compute the observability matrix by multiplying this Jacobian matrix with the transition matrix, the rank of the observability matrix remains the same. As shown in Lee et al. (2020) the extrinsic pose will be unobservable under certain motions. We use the extrinsic pose derived from the CAD model as an initial Gaussian prior to counteract this problem.

Observability of Global Orientation

We further demonstrate that the global orientation becomes observable through the application of the plane constraint, combined with prior knowledge of the plane orientation. To simplify the analysis, we treat the extrinsic pose as fixed in this part. We use ${}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}$ as a shorthand notation for $\mathbf{R}({}^{\mathrm{g}}\mathbf{q}_{\mathrm{w}})$ representing the plane orientation, and ${}^{\mathrm{i},k}\mathbf{R}_{\mathrm{w}}$ to represent the inverse of the sensor orientation at step k. For the plane constraint, the Jacobians for step k are:

$$
\begin{aligned}
\frac{\partial \mathbf{r}^1_{\text{plane}}}{\partial {}^{\mathrm{i},k}\mathbf{R}_{\mathrm{w}}} &= \mathbf{J}^1_{{}^{\mathrm{i},k}\mathbf{R}_{\mathrm{w}}} = \widehat{({}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}\,{}^{\mathrm{w}}\mathbf{R}_{\mathrm{i}}\,{}^{\mathrm{i}}\mathbf{R}_{\mathrm{b}}\mathbf{e}_3)}\,{}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}\,{}^{\mathrm{w}}\mathbf{R}_{\mathrm{i}})_{(:2,:)} \\
\frac{\partial \mathbf{r}^1_{\text{plane}}}{\partial {}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}} &= \mathbf{J}^1_{{}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}} = -(\widehat{{}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}\,{}^{\mathrm{w}}\mathbf{R}_{\mathrm{i}}\,{}^{\mathrm{i}}\mathbf{R}_{\mathrm{b}}\mathbf{e}_3})_{(:2,:2)} \\
\frac{\partial \mathbf{r}^2_{\text{plane}}}{\partial {}^{\mathrm{i},k}\mathbf{R}_{\mathrm{w}}} &= \mathbf{J}^2_{{}^{\mathrm{i},k}\mathbf{R}_{\mathrm{w}}} = -\mathbf{e}_3^\top\,{}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}\,{}^{\mathrm{w}}\mathbf{R}_{\mathrm{i},k}\widehat{({}^{\mathrm{i},k}\mathbf{R}_{\mathrm{b}}\,{}^{\mathrm{b}}\mathbf{t}_{\mathrm{i}})} \\
\frac{\partial \mathbf{r}^2_{\text{plane}}}{\partial {}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}} &= \mathbf{J}^2_{{}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}} = -\mathbf{e}_3^\top \widehat{\left({}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}({}^{\mathrm{w}}\mathbf{t}_{\mathrm{i},k} - {}^{\mathrm{w}}\mathbf{R}_{\mathrm{i},k}\,{}^{\mathrm{i},k}\mathbf{R}_{\mathrm{b}}\,{}^{\mathrm{b}}\mathbf{t}_{\mathrm{i}})\right)}_{(:,:2)} \\
\frac{\partial \mathbf{r}^2_{\text{plane}}}{\partial {}^{\mathrm{w}}\mathbf{t}_{\mathrm{i},k}} &= \mathbf{J}^2_{{}^{\mathrm{w}}\mathbf{t}_{\mathrm{i},k}} = \mathbf{e}_3^\top\,{}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}} \\
\frac{\partial \mathbf{r}^2_{\text{plane}}}{\partial {}^{\mathrm{g}}d_{\mathrm{w}}} &= \mathbf{J}^2_{{}^{\mathrm{g}}d_{\mathrm{w}}} = 1,
\end{aligned}
\tag{5.19}
$$

where $\mathbf{r}^1_{\text{plane}}$ and $\mathbf{r}^2_{\text{plane}}$ are the first and second row of the plane constraint residual, $\mathbf{J}^1$ and $\mathbf{J}^2$ are the first and second row in the Jacobian matrix. Operator $(.)_{(:2,:)}$ denotes the first two rows and operator $(.)_{(:,:2)}$ denotes the first two columns. The observability matrix can be written as:

$$
\begin{aligned}
\mathbf{M}^{\text{plane}}_k &= \mathbf{J}^{\text{plane}}_k \bar{\mathbf{\Phi}}_{k,1} \\
&= \begin{pmatrix} \mathbf{J}^1_{{}^{\mathrm{i},k}\mathbf{R}_{\mathrm{w}}} & \dots & \mathbf{0} & \mathbf{J}^1_{{}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}} & 0 & \dots \\ \mathbf{J}^2_{{}^{\mathrm{i},k}\mathbf{R}_{\mathrm{w}}} & \dots & \mathbf{J}^2_{{}^{\mathrm{w}}\mathbf{t}_{\mathrm{i},k}} & \mathbf{J}^2_{{}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}} & 1 & \dots \end{pmatrix} \bar{\mathbf{\Phi}}_{k,1} \\
&= \begin{pmatrix} \mathbf{J}^1_{{}^{\mathrm{i},k}\mathbf{R}_{\mathrm{w}}}\mathbf{\Phi}^{1,1}_{k,1} & & \mathbf{J}^1_{{}^{\mathrm{i},k}\mathbf{R}_{\mathrm{w}}}\mathbf{\Phi}^{1,2}_{k,1} & & \\ \mathbf{J}^2_{{}^{\mathrm{i},k}\mathbf{R}_{\mathrm{w}}}\mathbf{\Phi}^{1,1}_{k,1} + \mathbf{J}^2_{{}^{\mathrm{w}}\mathbf{t}_{\mathrm{i},k}}\mathbf{\Phi}^{5,1}_{k,1} & \mathbf{J}^2_{{}^{\mathrm{i},k}\mathbf{R}_{\mathrm{w}}}\mathbf{\Phi}^{1,2}_{k,1} + \mathbf{J}^2_{{}^{\mathrm{w}}\mathbf{t}_{\mathrm{i},k}}\mathbf{\Phi}^{5,2}_{k,1} & & & \\ & & \dots & & \\ & & \mathbf{J}^2_{{}^{\mathrm{w}}\mathbf{t}_{\mathrm{i},k}}\delta t_k & \mathbf{J}^2_{{}^{\mathrm{w}}\mathbf{t}_{\mathrm{i},k}}\mathbf{\Phi}^{5,4}_{k,1} & \mathbf{J}^2_{{}^{\mathrm{w}}\mathbf{t}_{\mathrm{i},k}} & \mathbf{0} \end{pmatrix} \\
&\qquad \begin{matrix} \mathbf{0} & \mathbf{J}^1_{{}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}} & 0 & \dots \\ \mathbf{J}^2_{{}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}} & 1 & \dots \end{matrix} \end{pmatrix}.
\end{aligned}
\tag{5.20}
$$

The derivation of the null space $\mathbf{N}_o$ for the global orientation of VIO is given in Wu and Roumeliotis (2016). Since the change of the global orientation also affects the plane angle ${}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}$, following Wu and Roumeliotis (2016), we compute the null space of the global

orientation for the plane constraint in our method as follows:

$$\bar{\mathbf{N}}_o = \begin{pmatrix} \mathbf{N}_o^\top & {}^{\mathrm{w}}\mathbf{R}_{\mathrm{g}(:,:2)} & \mathbf{0} & \dots & \mathbf{0} \end{pmatrix}^\top. \tag{5.21}$$

To demonstrate that this indeed constitutes a null space, we compute the product of the observability matrix $\mathbf{M}_k^{\text{plane}}$ and the null space $\bar{\mathbf{N}}_o$:

$$\begin{aligned} &\mathbf{M}_k^{\text{plane}}\bar{\mathbf{N}}_o = \\ &\begin{pmatrix} \widehat{{}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}{}^{\mathrm{w}}\mathbf{R}_{\mathrm{i},k}{}^{\mathrm{i}}\mathbf{R}_{\mathrm{b}}\mathbf{e}_3}{}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}})_{(:2,:)} - (\widehat{{}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}{}^{\mathrm{w}}\mathbf{R}_{\mathrm{i},k}{}^{\mathrm{i}}\mathbf{R}_{\mathrm{b}}\mathbf{e}_3})_{(:2,:2)}{}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}(:2,:)} \\ -\mathbf{e}_3^\top{}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}{}^{\mathrm{w}}\mathbf{R}_{\mathrm{i},k}\widehat{({}^{\mathrm{i},k}\mathbf{R}_{\mathrm{b}}{}^{\mathrm{b}}\mathbf{t}_{\mathrm{i}})}{}^{\mathrm{i},k}\mathbf{R}_{\mathrm{w}} + \mathbf{e}_3^\top{}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}\widehat{{}^{\mathrm{w}}\mathbf{t}_{\mathrm{i},k}} - \mathbf{e}_3^\top\widehat{\left({}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}({}^{\mathrm{w}}\mathbf{t}_{\mathrm{i},k} - {}^{\mathrm{w}}\mathbf{R}_{\mathrm{i},k}{}^{\mathrm{i},k}\mathbf{R}_{\mathrm{b}}{}^{\mathrm{b}}\mathbf{t}_{\mathrm{i}})\right)}_{(:,:2)}{}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}(:2,:)} \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \end{pmatrix} \end{aligned} \tag{5.22}$$

with property of the rotation matrix: $\widehat{\mathbf{R}\mathbf{t}} = \mathbf{R}\hat{\mathbf{t}}\mathbf{R}^\top$. This result implies that prior knowledge of the plane orientation is necessary to make the global orientation observable. Since our robot starts from still state, we use the initial accelerator measurement to initialize the plane angle. By adding this information as a Gaussian initial prior to the plane angle ${}^{\mathrm{g}}\mathbf{R}_{\mathrm{w}}$, the global orientation becomes observable.

Observability of RBF Parameters

Finally, we show the observability of the RBF parameters $(s_{\text{lin}}, \mu_{\text{lin}}, \sigma_{\text{lin}})$ for the translational velocity. The parameters for the angular velocity have similar observability. For simplicity, we drop the subscript here. We denote $\exp\left(-\frac{\|d\tau_i-\mu\|^2}{2\sigma^2}\right)$ as $\exp(.)$, where $d\tau_i$ is the time difference between control command at t_i and image frame at t. The Jacobian matrix of the motion residual $\mathbf{r}_\xi$ is:

$$\begin{aligned} \mathbf{J}_k^\xi &= \begin{pmatrix} \dots & \frac{\partial \mathbf{r}_\xi}{\partial s} & \frac{\partial \mathbf{r}_\xi}{\partial \mu} & \frac{\partial \mathbf{r}_\xi}{\partial \sigma} \end{pmatrix} \\ &= \begin{pmatrix} \dots & -\frac{\bar{v}}{s} & \frac{\bar{v}\sum_{i=1}^N \exp(.)\frac{d\tau_i-\mu}{\sigma^2} - s\sum_{i=1}^N v_i \exp(.)\frac{d\tau_i-\mu}{\sigma^2}}{\sum_{i=1}^N \exp(.)} & \frac{\bar{v}\sum_{i=1}^N \exp(.)\frac{(d\tau_i-\mu)^2}{\sigma^3} - s\sum_{i=1}^N v_i \exp(.)\frac{(d\tau_i-\mu)^2}{\sigma^3}}{\sum_{i=1}^N \exp(.)} \end{pmatrix}, \end{aligned} \tag{5.23}$$

where

$$\bar{v} = s\frac{\sum_{i=1}^N \exp(.) v_{t_i}}{\sum_{i=1}^N \exp(.)}. \tag{5.24}$$

Table 5.1: Trajectory accuracy in RPE and ATE of our proposed approach (kin-vio) and a pure VIO method (vio).

	transl. RMSE RPE in m		rot. RMSE RPE in deg		transl. RMSE ATE in m		rot. RMSE ATE in deg	
dataset	vio	kin-vio (ours)	vio	kin-vio (ours)	vio	kin-vio (ours)	vio	kin-vio (ours)
small-01	0.035	**0.021**	0.659	**0.597**	0.037	**0.014**	0.713	**0.463**
small-02	0.120	**0.106**	**0.664**	0.756	0.097	**0.077**	0.656	**0.622**
small-03	0.042	**0.027**	0.832	**0.693**	0.037	**0.019**	1.060	**0.561**
mid-01	0.232	**0.197**	1.439	**1.242**	0.190	**0.153**	0.978	**0.957**
mid-02	0.195	**0.158**	1.171	**1.100**	0.150	**0.108**	0.807	**0.739**
mid-03	0.342	**0.271**	1.490	**1.257**	0.150	**0.088**	1.674	**1.224**
large-01	0.828	**0.402**	2.278	**1.253**	0.512	**0.179**	2.360	**0.907**
large-02	0.467	**0.381**	1.495	**1.001**	0.237	**0.216**	1.032	**0.749**
large-03	1.275	**0.972**	3.501	**2.480**	0.953	**0.735**	2.861	**2.101**

Since we use a constant propagation model for the RBF parameters, the corresponding 3×3 bottom right block of the observability matrix based on our motion model is:

$$\mathbf{M}_k^{\xi,\mathrm{RBF}} = \begin{pmatrix} -\frac{\bar{v}}{s} & 0 & 0 \\ 0 & \frac{\bar{v}\sum_{i=1}^{N}\exp(.)\frac{d\tau_i-\mu}{\sigma^2} - s\sum_{i=1}^{N} v_i \exp(.)\frac{d\tau_i-\mu}{\sigma^2}}{\sum_{i=1}^{N}\exp(.)} & 0 \\ 0 & 0 & \frac{\bar{v}\sum_{i=1}^{N}\exp(.)\frac{(d\tau_i-\mu)^2}{\sigma^3} - s\sum_{i=1}^{N} v_i \exp(.)\frac{(d\tau_i-\mu)^2}{\sigma^3}}{\sum_{i=1}^{N}\exp(.)} \end{pmatrix}. \tag{5.25}$$

The rank of this 3×3 matrix $\mathbf{M}_k^{\xi,\mathrm{RBF}}$ determines the observability of the linear or angular RBF parameters. The RBF parameters are unobservable if the velocities v_{t_i} in the window are constant, which can happen especially at the beginning of the datasets when the robot stands still. We place a weak Gaussian prior on the initial values of the RBF parameter estimates. Due to the marginalization prior, the RBF parameters will remain observable even if they become temporarily unobservable in the window.

5.4 Experiments

We evaluate the proposed kinematics-constraint VIO on a differential drive robot with a fisheye-stereo camera and IMU (see Figure 5.3) in indoor environments. Similar as in Lee et al. (2020) and Wu et al. (2017), global offline optimization results are used as ground-truth. To make sure enough loops can be found and the global optimization is accurate, the robot travels to the same location for several times in each recorded sequence. We evaluate the accuracy of the estimate in terms of absolute trajectory error (ATE), relative pose error (RPE) (Z. Zhang and Scaramuzza, 2018), and the error of the effective control velocity with the ground-truth velocity. The RPE is computed by averaging the errors over $10, 20, \ldots, 50\%$ sequence lengths of the full trajectory. We also

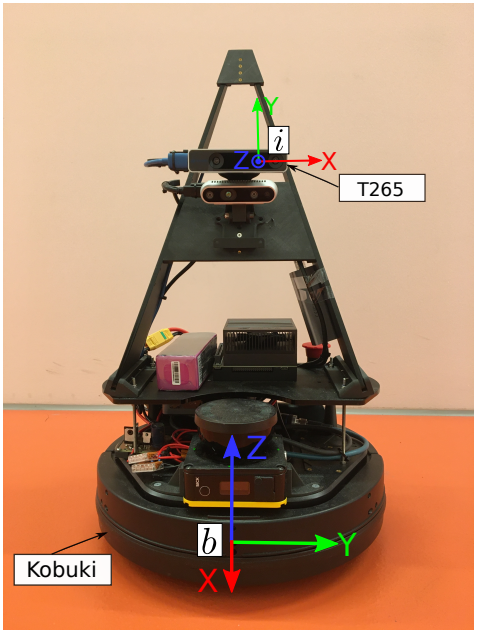


Figure 5.3: Robot platform used in our experiments. The robot is built on a Kobuki mobile base with differential drive and is equipped with a Realsense T265 fisheye-stereo camera with IMU. The other sensor elements are not used in the experiment.

validate the prediction accuracy of the calibrated RBF kernel for different time horizons.

Three groups of data with different environment scales are collected and each group consists of three different sequences. In the sequences, the robot starts from a static pose and then approximately drives at its maximum speed of 0.5 m/s. The robot travels over wooden floor, concrete and tiles, which also cause vibrations on the robot. The average lengths are 57.2 m (small scale), 222.3 m (middle scale), and 413.3 m (large scale). The ground-truth for trajectory evaluation is computed using the global bundle adjustment layer of Usenko et al. (2020). The method uses non-linear factor recovery to bound the computational and memory complexity of bundle adjustment using keyframes and to transfer information accumulated from intermediate frames and IMU measurements during VIO. We use the dense global mapping result as ground-truth where we set every frame as a keyframe and optimize them globally. The image rate is 30 Hz, the IMU rate is 200 Hz and the linear and angular commands are sent at the rate of 15 Hz. The RBF parameters μ, σ and s are initialized to 0, 0.5 and 1 for both linear and angular velocity commands. A small command window can not collect enough information, while a large command window includes commands that are far away from the current frame. We empirically choose a command window size of 3. The extrinsic pose between the base and sensor frame is initialized with values from the robot CAD model.

5.4.1 Tracking Accuracy Evaluation

Table 5.1 summarizes the RPE and ATE evaluation results. By integrating the kinematic motion constraint and the plane constraint (kin-vio) both ATE and RPE are reduced over

Table 5.2: Average translational accuracy in RPE and ATE and linear velocity error for different constraints over all sequences.

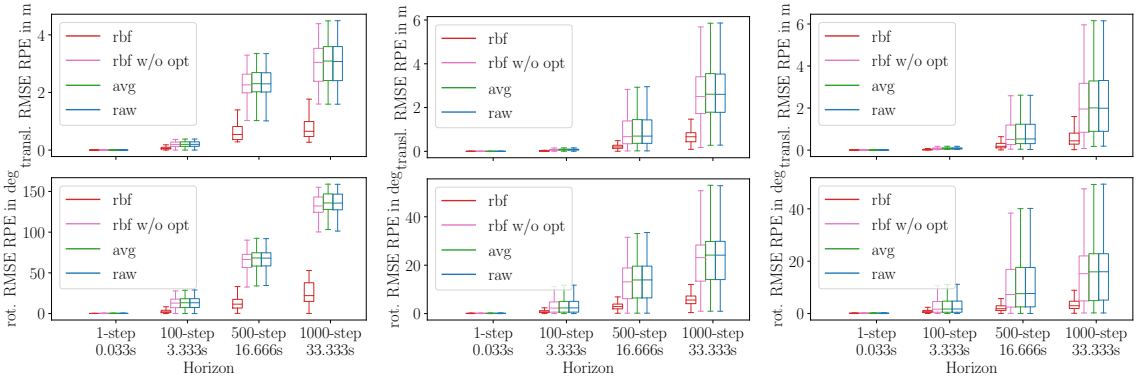
	avg. transl. RMSE in m				
	kin-vio ours: rbf (w/o plane)	kin-vio rbf w/o opt (w/o plane)	kin-vio avg (w/o plane)	kin-vio raw (w/o plane)	kin-vio only plane
RPE	**0.282** (**0.393**)	0.324 (0.441)	0.336 (0.452)	0.337 (0.452)	0.296
ATE	**0.177** (**0.270**)	0.210 (0.303)	0.219 (0.311)	0.220 (0.311)	0.190
	avg. RMSE of linear velocity in m/s				
vel. error	**0.025**	0.034	0.036	0.036	

Table 5.3: Average rotational trajectory accuracy in RPE and ATE and angular velocity error for different constraints over all sequences.

	avg. rot. RMSE in deg				
	kin-vio ours: rbf (w/o plane)	kin-vio rbf w/o opt (w/o plane)	kin-vio avg (w/o plane)	kin-vio raw (w/o plane)	kin-vio only plane
RPE	**1.153** (**1.507**)	1.180 (1.592)	1.183 (1.600)	1.184 (1.601)	1.164
ATE	**0.925** (**1.356**)	0.947 (1.413)	0.948 (1.418)	0.950 (1.418)	0.935
	avg. RMSE of angular velocity in deg/s				
vel. error	**0.031**	0.062	0.064	0.067	

pure VIO (vio). Figure 5.1 illustrates and compares the results of data sequence large-01 estimated with pure VIO and our kinematics-constraint VIO. As can be seen, with the proposed method the deviation is decreased, especially in those parts of the trajectory with larger rotational motion.

In an ablation study, we compare the estimation accuracy of using RBF kernel weighting (kin-vio rbf) with other different weighting methods including RBF kernel with fixed initial parameters (kin-vio rbf w/o opt), non-weighted averaging of the command window (kin-vio avg), and using the last command that comes before the first frame of each frame pair (kin-vio raw). In addition, we also evaluate the estimate with only the plane motion constraint in addition to the VIO constraints. The error values are averaged over all data sequences and summarized in Table 5.2 and Table 5.3, in which our combined method (kin-vio) consistently outperforms the others. The motion model can improve the tracking accuracy while the parameters are calibrated together with VIO. This can be attributed to the regularization by the parametric motion model whose parameters are adapted for a range of positive and negative velocity commands.



(a) Prediction on dataset small-01 **(b)** Prediction on dataset mid-01 **(c)** Prediction on dataset large-01

Figure 5.4: Prediction error on small-01, mid-01 and large-01. Our approach consistently has the smallest prediction error.

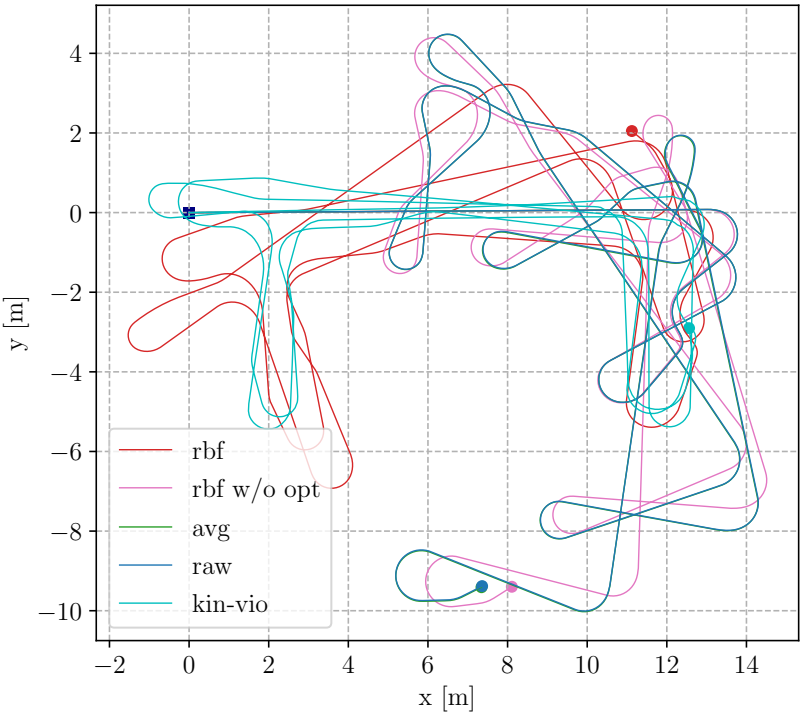


Figure 5.5: Predicted trajectories from start (square) to end (circle) on small-02. Our approach (rbf) follows the kinematics-constrained VIO result (kin-vio) closer.

5.4.2 Prediction Accuracy Evaluation

We also evaluate the performance of our method for forward motion prediction. Results for online prediction on data sequence small-01, mid-01 and large-01 are shown in Figure 5.4. For this evaluation, we compute the prediction from each frame in the trajectory with different horizon lengths starting from the current estimate of the parameters. The command window is shifted along the future trajectory with the prediction step. We also compare the prediction accuracy with three alternative calibration methods. The first method uses the RBF model with constant initial parameters (rbf w/o opt). The

second method uses the unweighted average value of the commands in the command window (avg). The third method calculates the prediction with the latest command (raw). Our proposed approach with calibrated RBF kernel parameters is denoted as rbf.

It can be observed that the RBF kernel with optimized parameters has the smallest prediction error especially for longer prediction horizons. During the experiment we noticed that the improvement of the prediction accuracy is relatively smaller on the longer data sequences. This is because the rotations introduce larger errors and in the longer data sequences the robot mostly performs translational motion along the corridors with constant speed. Figure 5.5 illustrates a prediction result on data sequence small-02 from the beginning to the end of the trajectory with the final optimized RBF parameters. Our approach shows the smallest deviation compared with other methods.

5.4.3 Computation Time

We compare the run-time of our approach with the base VIO on our dataset using one Intel Xeon Silver 4112 CPU@2.60GHz with 8 cores. On average the computation time increases by 34.14% from 14.79 ms to 19.84 ms for processing one frame, the maximum time stays similar with 60.05 ms at rare peaks, the minimum time is 3.90 ms. The approach can still process faster than real-time.

5.4.4 Discussion and Limitations

A potential limitation of our approach can be estimation bias in the VIO in settings such as texture-less scenes or biased camera intrinsic parameters. In our work, we assume that the VIO result is sufficiently accurate with negligible systematic offsets so it can be used to calibrate the effective control. In future work, we could additionally integrate other types of sensors like GPS. Outlier measurements can be handled with robust norms in the VIO. One could also only activate the motion model when state variables like IMU biases converge and indicate an accurate VIO.

Integrating wheel encoder information can also improve VIO accuracy, as it measures the actual rotation of the wheels at high frequency. To convert the wheel encoder information to body velocity or relative pose, one needs to consider the type of the vehicle. Our approach is simpler to integrate for robots whose motion can be modeled with our model (such as differential or Ackerman drives (Thrun et al., 2005)). Moreover, our calibrated model could be used for downstream tasks such as model-predictive control-based path tracking. In future work, we are also interested in combining wheel encoder measurements with our method.

5.5 Conclusion

In this chapter, we present a VIO approach based on non-linear windowed optimization that includes velocity-control-based kinematic motion model constraints. The motion model is integrated as a new factor between each image pair. We compute the 2D robot velocity between two consecutive images from the state estimate and compare it against the control command sent to the robot. To compensate for the difference between the control command and the real robot action, we use RBF kernels along the time domain to determine an effective control command from the raw commands and calibrate the RBF parameters online in the VIO system. Our experiments demonstrate that by using this motion constraint in addition to a planar motion constraint not only the accuracy of the VIO is improved but the learned motion model can also predict the robot motion more accurately. In future work, more complex motion models or including other sensors like GPS or wheel odometry could be investigated.

Online Adaptation of Dynamic Model with Visual-Inertial Odometry

6

Declaration of Contributions

The contents of this chapter are based on the peer-reviewed conference publication

 'Online Calibration of a Single-Track Ground Vehicle Dynamics Model by Tight Fusion with Visual-Inertial Odometry'. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. To appear, preprint available at https://doi.org/10.48550/arXiv.2309.11148 (H. Li and Stückler, 2024)

with the following co-author contributions:

	Scientific Ideas	Data Generation	Analysis & Interpretation	Paper Writing
Haolong Li	65%	100%	80%	80%
Jörg Stückler	35%	0%	20%	20%

Jörg Stückler conceived the idea of using an ordinary differential equation-based dynamic model as the motion constraint. Haolong Li proposed to use the singularity-free single-track model, implemented the tight integration, collected the data, and performed the experiments. Haolong Li and Jörg Stückler wrote the paper.

Compared to the conference publication, this chapter contains unified notation and some reformulated and reorganized paragraphs.

6.1 Introduction

Autonomous mobile robot navigation requires the ability to perceive the extrinsic environment for localization and knowledge of an accurate robot dynamic model for path planning and control. Most previous works for ground robots solve the problems of state estimation and calibration of the dynamic model of the robot drive separately. For the perception and localization part, visual-inertial odometry (VIO) methods (e.g. Leutenegger et al. (2015), Qin et al. (2018), and Usenko et al. (2020)) have become popular in the computer vision community due to their low-cost and outstanding tracking accuracy. On the other hand, many works from the robotics community try to estimate vehicle slip angle and vehicle parameters such as mass or tire coefficients where vehicle pose, velocity, and acceleration can be measured by GPS or other odometry methods (e.g. Reina et al. (2017), Wielitzka et al. (2015), and C. You and Tsiotras (2017)).

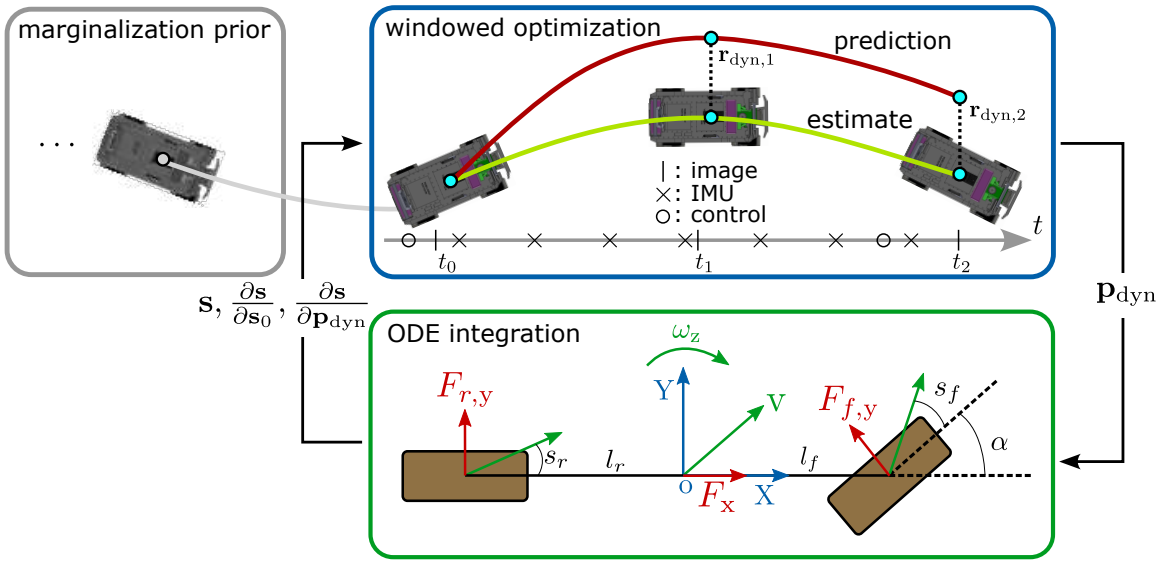


Figure 6.1: ST-VIO performs windowed optimization (blue box) with marginalization of old states (gray box) to estimate vehicle motion and parameters of a single-track dynamic model. The dynamic model is used as a factor in the optimization through ODE integration (green box, the wheels are represented by brown rectangles, velocity is indicated in green, force is shown in red, the x-axis is longitudinal, and the y-axis is the lateral axis.).

In this chapter, we also focus on wheeled robots and propose a VIO method that can estimate robot pose states and calibrate the dynamic model of the drive jointly.

The integration of motion models into VIO has already been extensively researched. Some studies aim to use motion models with sensors such as wheel odometers to constrain the VIO and improve the tracking accuracy (e.g. Lee et al. (2020) and Wu et al. (2017)). In contrast, studies like Cioffi et al. (2023) and Nisar et al. (2019) integrate the dynamic models of a multicopter not just for improving tracking accuracy but also for external force prediction. Here, we focus on ground wheeled robots and incorporate a dynamic model with friction and tire-ground interactions into VIO. This model serves not only as a motion constraint for tracking but is also calibrated online, improving the accuracy of forward prediction based on control inputs. We employ the single-track model, also referred to as the bicycle model, where the left and right wheels are lumped as one (green box in Figure 6.1). It provides a good trade-off between accuracy and computational efficiency (Kabzan et al., 2019). The integration of this dynamic model into VIO presents significant challenges. Firstly, the model's behavior can vary due to changes in terrain properties or tire conditions. Secondly, this model encounters a singularity when vehicle speed nears zero.

To overcome these hurdles, we introduce our method ST-VIO in this chapter. We modify the dynamic model to eliminate the singularity and implement real-time online parameter calibration together with VIO state variable optimization. Our method continuously adapts the model, enabling more accurate predictions of vehicle pose and velocity based on the latest state estimates and new control inputs. This adaptive prediction could enable potential applications in downstream tasks such as model-predictive control and navigation planning.

We evaluate our method in robot experiments in indoor and outdoor environments. Our experiments demonstrate that integrating the robot drive dynamic model can improve tracking accuracy. Moreover, the online calibration is capable of adapting the parameters so that the accuracy of prediction with the model is improved. In summary, the main contributions of our work are:

- We tightly integrate a singularity-free single-track vehicle model which is formulated as an ordinary differential equation (ODE) as a multistep motion constraint for ground wheeled robots into VIO. This enables online real-time estimation and calibration of model parameters alongside VIO state variables.
- We demonstrate that our method not only enhances VIO tracking accuracy but also allows the model to adapt to variations in terrain and vehicle properties.

6.2 Related Work

Several prior works exist in the literature that model wheeled vehicle dynamics and identify model parameters by matching state estimates from the model with ground-truth recordings from real vehicles. Wielitzka et al. (2015) filter parameters of a double-track dynamic model with vehicle states including side-slip angle using an unscented Kalman filter based on a GPS-gyro measurement system. A similar approach is taken in C. You and Tsiotras (2017) which estimates parameters of both a single-track model and an extended double-track model that models air resistance and sprung and unsprung mass separately. Aghli and Heckman (2018) identify the parameters of the robot dynamic model online using ground-truth from a motion capture system. In our approach, we calibrate a single-track dynamic model online jointly with visual-inertial odometry state estimation. Xu et al. (2019) develop a learning-based approach that trains multilayer perceptrons or LSTMs (Hochreiter and Schmidhuber, 1997) to model the dynamics. Kabzan et al. (2019) and Jiang et al. (2021) train a Gaussian process or neural residual models to improve predictions of a base dynamic model. Such data-driven methods, however, require that training and test distribution are sufficiently similar to generalize well to cases unseen during training, while our method adapts online.

Using visual-inertial sensors to estimate robot states is appealing due to their lower cost and greater flexibility compared to satellite measurements or motion capture systems. Since the VIO system for ground robot motion is ill-posed (Wu et al., 2017), numerous previous studies have sought to integrate and calibrate either kinematic or dynamic motion models with VIO to enhance tracking accuracy (e.g., F. Ma et al. (2019), Xiong et al. (2022), and P. Zhang et al. (2020)). These methods, however, typically do not pursue calibrating the parameters of the motion model online for motion prediction as in our approach. Weydert (2012) estimate the vehicle ego motion and model parameters with a dual ensemble EKF using stereo cameras. The method does not learn a forward model mapping between control commands and vehicle state like our method. H. Li and Stückler (2022) calibrate parameters of a velocity-control based kinematic motion

model online by tight integration with stereo visual-inertial odometry (Usenko et al., 2020). However, the kinematic motion model does not take tire-ground interaction and dynamics into consideration. Notably, above mentioned methods such as Weydert (2012) and Xiong et al. (2022) do not address low-speed scenarios due to singularities of the dynamic model at zero speed, potentially compromising consistent model calibration at low speeds. V. Zhang et al. (2018) propose a non-smooth model which caps velocities at low speeds. An alternative approach mentioned in V. Zhang et al. (2018) switches to a kinematic model which would add complexity for integration as motion factor and prediction. We adjust the dynamic model to eliminate singularities and ensure it remains differentiable.

6.3 Method

In our approach, we tightly fuse a single-track vehicle dynamic model with VIO to improve state estimation and facilitate online calibration. We begin by introducing the single-track dynamic model and our modifications. We then demonstrate how we integrate the dynamics factor from the single-track model with multistep prediction. Finally, we cover implementation details and parameter initialization procedures.

6.3.1 Single-Track Dynamic Model

The single-track vehicle dynamic model is commonly used for navigation of ground wheeled robots due to its balance of simplicity and accuracy (Kabzan et al., 2019). As depicted in the green box of Figure 6.1, the local body frame o is located at the center of mass of the vehicle and is assumed to be fixed. The x-axis of the body frame points forward and the z-axis (yaw axis) points upward. We define the state variables of the dynamics system in the body frame o as:

$$\mathbf{s} = \begin{pmatrix} x, & y, & \theta, & v_{\mathrm{x}}, & v_{\mathrm{y}}, & \omega_{\mathrm{z}} \end{pmatrix}^{\top}, \tag{6.1}$$

where x and y are the 2D position, and θ is the yaw rotation along the z-axis. The velocities $v_{\mathrm{x}}, v_{\mathrm{y}}$ are the corresponding linear velocities and ω_{z} is the yaw velocity. The control inputs of the dynamics system are the throttle control $u_{\mathrm{thr}} \in [0, 1]$ and steering control $u_{\mathrm{str}} \in [-1, 1]$. The dynamics system itself is an ordinary differential

equation (ODE) system expressed as:

$$\dot{\mathbf{s}} = \begin{pmatrix} v_{\mathrm{x}} - \omega_{\mathrm{z}} y \\ v_{\mathrm{y}} + \omega_{\mathrm{z}} x \\ \omega_{\mathrm{z}} \\ \frac{1}{m}(F_{\mathrm{x}} - F_{f,\mathrm{y}} \sin(\alpha)) + v_{\mathrm{y}} \omega_{\mathrm{z}} \\ \frac{1}{m}(F_{f,\mathrm{y}} \cos(\alpha) + F_{r,\mathrm{y}}) - v_{\mathrm{x}} \omega_{\mathrm{z}} \\ \frac{1}{I_{\mathrm{z}}}(l_f F_{f,\mathrm{y}} \cos(\alpha) - l_r F_{r,\mathrm{y}}) \end{pmatrix}, \tag{6.2}$$

where m and I_{z} are the vehicle mass and yaw momentum of inertia. As illustrated in the green box of Figure 6.1, $l_{f/r}$ represent the distances from the front and rear wheels to the center of mass, respectively, α denotes the front wheel angle, F_{x} is the longitudinal force at the center of mass in the body frame depending on the throttle input, while $F_{f/r,\mathrm{y}}$ refer to the lateral tire forces at the front and rear wheels.

We write the parameters that we aim to calibrate online as a vector

$$\mathbf{p}_{\mathrm{dyn}} = \begin{pmatrix} \gamma, & C_{\mathrm{thr},1} & C_{\mathrm{thr},2} & C_{\mathrm{res}}, & C_{\mathrm{tire}} \end{pmatrix}^{\top}. \tag{6.3}$$

The first term γ represents the steering ratio between the front wheel angle and the steering input as $\alpha = \gamma u_{\mathrm{str}}$. $C_{\mathrm{thr},1/2}$ and C_{res} are longitudinal force related parameters:

$$F_{\mathrm{x}} = f(C_{\mathrm{thr},1} u_{\mathrm{thr}} - C_{\mathrm{thr},2} v_{\mathrm{x}}) - \tanh(\sigma v_{\mathrm{x}}) C_{\mathrm{res}}. \tag{6.4}$$

The first term approximates the power-train force, which is a non-linear function of throttle input, motor speed, and other effects (Lynch and Park, 2017). We empirically approximate the non-linearity for our motor with function

$$f(x) = \psi x + \tau \log(1 + \exp(x)) - \log(2) \tag{6.5}$$

that scales up the acceleration force and scales down the deceleration force of the motor. The hyper-parameters ψ and τ control the scaling ratio. We ignore the air drag force and model the resistance as a scalar C_{res}, which is multiplied with a hyperbolic tangent function $\tanh(\sigma v_{\mathrm{x}})$ such that no false longitudinal force will be applied when the vehicle stands still and no throttle input is given. Here, σ is a hyper-parameter that controls the steepness of the hyperbolic tangent around zero. The hyper-parameters related to the longitudinal force, $\{\psi, \tau, \sigma\}$, are optimized offline as explained in a later section. Lastly, C_{tire} is the tire coefficient of a linear tire model for the lateral tire force $F_{f/r,\mathrm{y}}$ as explained next.

Similar to Kabzan et al. (2019), we estimate the lateral tire force from the front and rear slip angle $s_{f/r}$, as depicted in Figure 6.1. They are the difference between the wheel

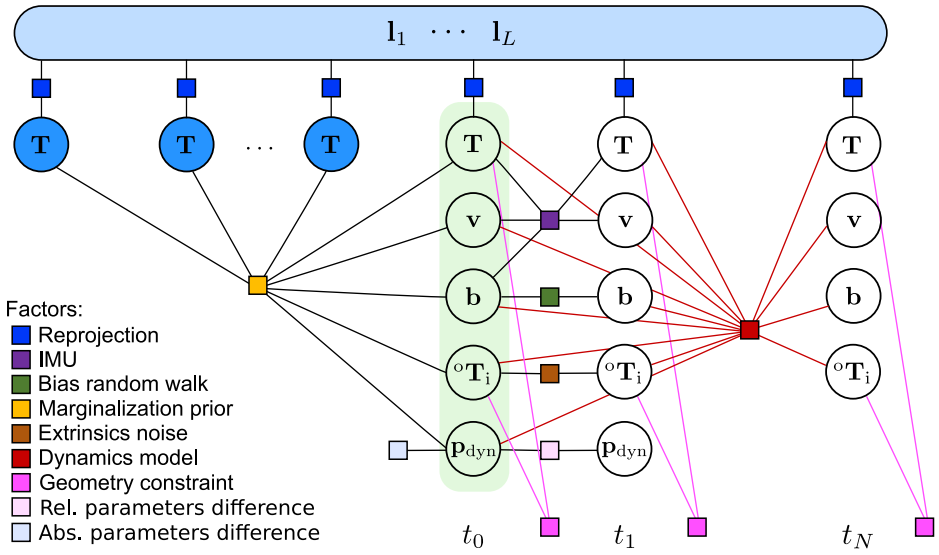


Figure 6.2: Factor graph of ST-VIO. The green shaded circles represent the active frames at t_0, while the unshaded circles represent the remaining active frames. Their information has not yet been marginalized. The blue shaded circles represent the inactive keyframes, whose pose state is kept, and the rest information is marginalized out. The dynamics factor (red) connects poses, velocities, gyroscope biases, extrinsic poses of all active recent frames and the dynamic model parameters at t_0.

angle and wheel velocity angle and can be computed as

$$s_f = \arctan \frac{v_x \sin(\alpha) - (v_y + l_f \omega_z)\cos(\alpha)}{g(v_x \cos(\alpha) + (v_y + l_f \omega_z)\sin(\alpha))} \tag{6.6}$$

$$s_r = \arctan \frac{l_r \omega_z - v_y}{g(v_x)}, \tag{6.7}$$

where $g(x) = x$ in the original model. They are undefined at zero longitudinal velocity, $v_x = 0$. In V. Zhang et al. (2018), this singularity is handled by setting a constant lower bound for v_x using a threshold function. However, this non-smoothness complicates its application in factor graph optimization. Instead, we use a soft thresholding function

$$g(x) = \log(\exp(2x) + 1) - x \tag{6.8}$$

to maintain the differentiability of the model. This function acts as a smooth approximation of the absolute value function and is always greater than zero, thus naturally providing a lower bound in a smooth way regardless of whether the velocity is positive or negative. We use a linear tire model for computational efficiency and compute the lateral tire forces by:

$$F_{f,y} = C_{\text{tire}} s_f \tag{6.9}$$

$$F_{r,y} = C_{\text{tire}} s_r. \tag{6.10}$$

6.3.2 Integration of Single-Track Dynamics with VIO

We utilize the smooth single-track model as the motion constraint of the VIO and calibrate the model parameters together with VIO state variables. The backbone of our approach is a sliding-window optimization-based VIO system from Usenko et al. (2020). The factor graph of our ST-VIO is depicted in Figure 6.2. Within each window, there's a collection of landmarks $\mathcal{L}$, inactive keyframes, and active recent frames whose state variables are not marginalized yet. In the base VIO (Usenko et al., 2020), the inactive keyframe's state is the pose of the IMU in the world frame ${}^{\mathrm{w}}\mathbf{T}_{\mathrm{i},t} \in \mathrm{SE}(3)$, the recent active frames' states consist of frame pose ${}^{\mathrm{w}}\mathbf{T}_{\mathrm{i},t} \in \mathrm{SE}(3)$, linear velocity in world frame ${}^{\mathrm{w}}\mathbf{v} \in \mathbb{R}^3$ and IMU biases $\mathbf{b}_{\mathrm{g}}, \mathbf{b}_{\mathrm{a}} \in \mathbb{R}^3$. The VIO method optimizes these state variables by minimizing the reprojection residual between landmarks and detected keypoints in the image frames, the relative pose residual between consecutive recent frames using IMU measurements, as well as the changes of the IMU biases assuming random walk noise. As the window shifts to the subsequent timestamp, all state variables from the oldest recent frame are marginalized unless chosen as a keyframe. The marginalized data is retained as the marginalization prior.

In our ST-VIO, we expand the recent frames' state by incorporating the extrinsic pose between the vehicle body frame to the IMU frame ${}^{\mathrm{o}}\mathbf{T}_{\mathrm{i}} \in \mathrm{SE}(3)$, as well as the dynamic model parameters $\mathbf{p}_{\mathrm{dyn}}$. As a result, we model both elements as time-varying to accommodate changes in suspension and environmental conditions, respectively. In previous works (e.g., F. Ma et al. (2019), Xiong et al. (2022), and P. Zhang et al. (2020)), the motion constraints are computed between every two consecutive frames using the state estimates and the parameters stored in the first one of them. Since we are interested in multistep predictions into the future, we compute the multistep motion constraint with the state estimates and the parameters $\mathbf{p}_{\mathrm{dyn},t_0}$ stored at the first active recent frame (denoted as at t_0) in the current optimization window. As shown in Figure 6.2, the dynamics factor thus connects all recent active frames in the current window.

For a time interval between two frames $[t_n, t_{n+1}]$, we solve the ODE via the Runge–Kutta method based on the most recent control input, and set the initial state as

$$\mathbf{s}(t_n) = \left(0, 0, 0, v_{\mathrm{x},t_n}, v_{\mathrm{y},t_n}, \omega_{\mathrm{z},t_n}\right)^\top . \tag{6.11}$$

If $n = 0$, the initial velocity in body frame is computed from the VIO linear velocity estimate ${}^{\mathrm{w}}\mathbf{v}$, gyroscope measurement ${}^{\mathrm{i}}\boldsymbol{\omega}$ and rotation ${}^{\mathrm{o}}\mathbf{R}_{\mathrm{i}} \in \mathrm{SO}(3)$ of the extrinsics transformation ${}^{\mathrm{o}}\mathbf{T}_{\mathrm{i}}$:

$${}^{\mathrm{o}}\boldsymbol{\omega}_{\mathrm{z}} = \left({}^{\mathrm{o}}\mathbf{R}_{\mathrm{i}}({}^{\mathrm{i}}\boldsymbol{\omega} - \mathbf{b}_{\mathrm{g}})\right)_3 \tag{6.12}$$

$${}^{\mathrm{o}}\mathbf{v}_{\mathrm{x,y}} = ({}^{\mathrm{o}}\mathbf{R}_{\mathrm{w}}\,{}^{\mathrm{w}}\mathbf{v} + {}^{\mathrm{o}}\mathbf{t}_{\mathrm{i}} \times {}^{\mathrm{o}}\boldsymbol{\omega})_{1,2} . \tag{6.13}$$

If the frame at t_n is not the first recent frame in the current window, $v_{\mathrm{x},t_n}, v_{\mathrm{y},t_n}, \omega_{\mathrm{z},t_n}$ are set to the velocity solution of the previous time interval. The control input can also appear in between $[t_n, t_{n+1}]$. In this special case, we first solve the ODE based on the

control before t_n until this new control input. From the intermediate solution, we solve the ODE again based on the new control input until t_{n+1}. The numerical solution of the ODE yields

$$\mathbf{s}_{t_{n+1}} = \left(x_{t_{n+1}},\ y_{t_{n+1}},\ \theta_{t_{n+1}},\ v_{\mathrm{x},t_{n+1}},\ v_{\mathrm{y},t_{n+1}},\ \omega_{\mathrm{z},t_{n+1}}\right)^\top, \tag{6.14}$$

where the first three entries represent the relative 2D pose between t_n and t_{n+1}, and the last three entries represent the the 2D velocity in the local vehicle body frame o at the time t_{n+1}. The relative pose between t_0 and t_{n+1}, namely the multistep prediction of the dynamic model, can be computed as

$$\begin{pmatrix} {}^{t_0}x_{t_{n+1}} \\ {}^{t_0}y_{t_{n+1}} \\ {}^{t_0}\theta_{t_{n+1}} \end{pmatrix} = \begin{pmatrix} \cos\left({}^{t_0}\theta_{t_n}\right)x_{n+1} - \sin\left({}^{t_0}\theta_{t_n}\right)y_{n+1} + {}^{t_0}x_{t_n} \\ \sin\left({}^{t_0}\theta_{t_n}\right)x_{n+1} + \cos\left({}^{t_0}\theta_{t_n}\right)y_{n+1} + {}^{t_0}y_{t_n} \\ {}^{t_0}\theta_{t_n} + \theta_{n+1} \end{pmatrix}. \tag{6.15}$$

To compare the 6-DoF camera motion estimate of the VIO with the 3-DoF ground motion prediction by the motion model, we need to transform and project the VIO estimate into the vehicle body frame. The first step is to compute the 6-DoF relative pose between two timestamps e.g. t_0 and t_{n+1} in body frame o from VIO estimates and extrinsic poses:

$${}^{\mathrm{o},t_0}\mathbf{T}_{\mathrm{o},t_{n+1}} = {}^{\mathrm{o},t_0}\mathbf{T}_{\mathrm{i},t_0}\left({}^{\mathrm{w}}\mathbf{T}_{\mathrm{i},t_0}\right)^{-1}{}^{\mathrm{w}}\mathbf{T}_{\mathrm{i},t_{n+1}}\left({}^{\mathrm{o},t_{n+1}}\mathbf{T}_{\mathrm{i},t_{n+1}}\right)^{-1}. \tag{6.16}$$

Then we map the 6-DoF relative pose to 3-DoF by taking only the x and y component of the translation ${}^{\mathrm{o},t_0}\mathbf{p}_{\mathrm{o},t_{n+1}}$ and z component of the rotation ${}^{\mathrm{o},t_0}\mathbf{R}_{\mathrm{o},t_{n+1}}$

$$\begin{pmatrix} {}^{t_0}\widetilde{x}_{t_{n+1}} \\ {}^{t_0}\widetilde{y}_{t_{n+1}} \\ {}^{t_0}\widetilde{\theta}_{t_{n+1}} \end{pmatrix} = \begin{pmatrix} \left({}^{\mathrm{o},t_0}\mathbf{p}_{\mathrm{o},t_{n+1}}\right)_1 \\ \left({}^{\mathrm{o},t_0}\mathbf{p}_{\mathrm{o},t_{n+1}}\right)_2 \\ \mathrm{Log}\left({}^{\mathrm{o},t_0}\mathbf{R}_{\mathrm{o},t_{n+1}}\right)_3 \end{pmatrix}. \tag{6.17}$$

We penalize the difference between the predicted 2D pose by the dynamic model (Equation (6.15)) and the estimated 2D pose by the VIO (Equation (6.17)) in our dynamics residual. Additionally, we compare the predicted local body velocity $v_\mathrm{x}, v_\mathrm{y}, \omega_\mathrm{z}$ and the velocity derived from VIO estimates $\widetilde{v}_\mathrm{x}, \widetilde{v}_\mathrm{y}$ and gyroscope measurement $\widetilde{\omega}_\mathrm{z}$. The dynamics objective function is

$$E_\mathrm{dyn} = \sum_{n\in\mathcal{N}'} \mathbf{r}_{\mathrm{dyn},n}^\top \boldsymbol{\Sigma}_{\mathrm{dyn},n}^{-1}\mathbf{r}_{\mathrm{dyn},n}, \tag{6.18}$$

where $\mathcal{N}'$ is the set of the recent frames except for the last one in the window, $\boldsymbol{\Sigma}_{\mathrm{dyn},n}^{-1}$ is a diagonal weight matrix, and $\mathbf{r}_{\mathrm{dyn},n}$ is the residual vector stacked from position, orientation, and velocity differences.

6.3.3 Geometry Constraints

Similar as in Wu et al. (2017) we add a stochastic plane constraint into the VIO system as the single-track model depicts only planar motion. We assume that the trajectory of the vehicle body frame always lies on a plane in the world frame and its z-axis is always perpendicular to this plane. The plane residual is

$$\mathbf{r}_{\text{plane}} = \left(({}^{\text{w}}\mathbf{R}_{\text{i}} {}^{\text{o}}\mathbf{R}_{\text{i}}^{\top} \mathbf{e}_3)_{1,2}^{\top},\ d + \mathbf{e}_3^{\top} ({}^{\text{w}}\mathbf{p}_{\text{i}} - {}^{\text{w}}\mathbf{R}_{\text{i}} {}^{\text{o}}\mathbf{R}_{\text{i}}^{\top} {}^{\text{o}}\mathbf{p}_{\text{i}}) \right)^{\top}, \tag{6.19}$$

where $\mathbf{e}_3 = (0, 0, 1)^{\top}$ and d is the distance between world origin and initial vehicle body position, and ${}^{\text{o}}\mathbf{p}_{\text{i}}$ is the translation of the extrinsic transformation ${}^{\text{o}}\mathbf{T}_{\text{i}}$. Moreover, we also incorporate prior knowledge of the vehicle's geometry information. Figure 6.3 shows the mobile robot we use. We assume that the vehicle body frame is located close to the longitudinal axis of the vehicle because the vehicle is roughly symmetric along this axis. Since the suspension does not affect the yaw rotation of the camera with respect to the vehicle body frame, we also penalize the yaw component of ${}^{\text{o}}\mathbf{R}_{\text{i}}$ with $({}^{\text{o}}\mathbf{R}_{\text{i}}\mathbf{e}_3)_2$, where $\mathbf{e}_3$ is the forward axis of the IMU frame. We assume that the lateral distance between the body and IMU frame $({}^{\text{o}}\mathbf{p}_{\text{i}})_2$ is close to the lateral distance between the vehicle center and the IMU frame $l_{\text{cam},1}$. The longitudinal distance between body and IMU frame $({}^{\text{o}}\mathbf{p}_{\text{i}})_1$ is close to the sum of l_f and the distance between camera and front wheel $l_{\text{cam},2}$. The geometric residual is

$$\mathbf{r}_{\text{geom},n} = \left(\mathbf{r}_{\text{plane}}^{\top},\ ({}^{\text{o}}\mathbf{R}_{\text{i}}\mathbf{e}_3)_2,\ ({}^{\text{o}}\mathbf{p}_{\text{i}})_2 - l_{\text{cam},1},\ ({}^{\text{o}}\mathbf{p}_{\text{i}})_1 - l_f - l_{\text{cam},2} \right)^{\top}, \tag{6.20}$$

where $l_{\text{cam},1}$ and $l_{\text{cam},2}$ can be measured in the CAD model. The corresponding objective function is

$$E_{\text{geom}} = \sum_{n \in \mathcal{N}} \mathbf{r}_{\text{geom},n}^{\top} \boldsymbol{\Sigma}_{\text{geom},n}^{-1} \mathbf{r}_{\text{geom},n}, \tag{6.21}$$

where $\mathcal{N}$ is the set of all active recent frames and $\boldsymbol{\Sigma}_{\text{geom},n}^{-1}$ is a diagonal weight matrix.

6.3.4 Optimization

The above introduced dynamics factor and geometry constraints are integrated into the VIO system. As illustrated in Figure 6.2, the dynamics factor connects all recent active frames in the current window, and the geometry constraint factor is added to each recent active frame. To guarantee a smooth change of the extrinsics over time, we minimize the term

$$E_{\text{extr}} = \sum_{n \in \mathcal{N}'} \mathbf{r}_{\text{extr},n}^{\top} \boldsymbol{\Sigma}_{\text{extr},n}^{-1} \mathbf{r}_{\text{extr},n}, \tag{6.22}$$

where $\mathbf{r}_{\text{extr}}$ is the translation and rotation difference between two adjacent extrinsic pose estimates and $\boldsymbol{\Sigma}_{\text{extr},n}^{-1}$ is the diagonal weight matrix. The dynamic model parameters $\mathbf{p}_{\text{dyn},t_0}$ stored in the first recent frame at t_0 are used to perform multistep prediction until the end of the current window. We additionally include the model parameters

$\mathbf{p}_{\text{dyn},t_1}$ at the second recent frame at t_1 in the current window into the factor graph and minimize the difference $\mathbf{r}_{\mathbf{p}_{\text{dyn,rel}}}$ between $\mathbf{p}_{\text{dyn},t_0}$ and $\mathbf{p}_{\text{dyn},t_1}$. When the first recent frame of the old window is marginalized out, the marginalization prior information can thus be propagated to the parameters at the first recent frame in the new window and prevent rapid change of the model parameters. Besides this relative difference term, a weak absolute prior is added for $\mathbf{p}_{\text{dyn},t_0}$ by minimizing its difference $\mathbf{r}_{\mathbf{p}_{\text{dyn,abs}}}$ to the most recent marginalized parameters to avoid drift when the parameters become unobservable. Note that the relative difference term is not sufficient to alleviate drift in this case. This can be seen from the full probabilistic model without marginalization in which the parameters could drift consistently across all frames in the unobservable dimensions. The corresponding objective function is summarized as

$$E_{\text{param}} = \mathbf{r}_{\mathbf{p}_{\text{dyn,rel}}}^{\top} \boldsymbol{\Sigma}_{\mathbf{p}_{\text{dyn,rel}}}^{-1} \mathbf{r}_{\mathbf{p}_{\text{dyn,rel}}} + \mathbf{r}_{\mathbf{p}_{\text{dyn,abs}}}^{\top} \boldsymbol{\Sigma}_{\mathbf{p}_{\text{dyn,abs}}}^{-1} \mathbf{r}_{\mathbf{p}_{\text{dyn,abs}}}, \tag{6.23}$$

where $\boldsymbol{\Sigma}_{\mathbf{p}_{\text{dyn,rel}}}^{-1}$ and $\boldsymbol{\Sigma}_{\mathbf{p}_{\text{dyn,abs}}}^{-1}$ are the diagonal weight matrices. In summary, the overall objective function of our dynamics augmented VIO is

$$E_{\text{ST-VIO}} = E_{\text{VIO}} + E_{\text{dyn}} + E_{\text{geom}} + E_{\text{extr}} + E_{\text{param}}, \tag{6.24}$$

where E_{VIO} contains terms for the visual-inertial odometry and the marginalization prior (see Section 2.3 and Usenko et al. (2020)). The VIO system for planar motion is ill-posed and the accelerometer bias is unobservable if there is no rotation (Wu et al., 2017). Therefore, the dynamics factor should only be used when the accelerometer bias converges. We approximate the variance of accelerometer bias $\mathbf{b}_{\text{a}}$ by inverting the related Hessian matrix part. The dynamics factor is integrated when the variance of the accelerometer bias $\mathbf{b}_{\text{a},t_0}$ at the first active recent frame is smaller than a threshold.

6.3.5 Offline Initial Guess Estimation

The dynamics augmented VIO requires a reasonable initialization of the parameters and extrinsics to guarantee that the numerical solver of the ODE can output a plausible solution. We determine the center of mass position of our mobile robot and initialize the extrinsics ${}^{\text{o}}\mathbf{T}_{\text{i}}$ from the CAD model. The initial guess of steering ratio γ is approximated by the ratio between the max. value of steering control and the max. front wheel steering angle. The remaining parameters like the throttle mapping and tire coefficient are agnostic and thus an initial guess is found through offline optimization. We first manually select reasonable hyper-parameters $\{\psi, \tau, \sigma\}$ in the longitudinal force function and only optimize for $C_{\text{thr},1/2}$, C_{res} with purely forward motion data. Once the longitudinal force related parameters are identified, we optimize for all hyper-parameters, dynamic model parameters and extrinsics together using data collected with various steering inputs mixed with stop-and-go motion based on the dynamics factor and geometry constraints introduced in the previous section. During online optimization the hyper-parameters $\{\psi, \tau, \sigma\}$ are fixed. Since the extrinsic pose is time-varying, we still use the CAD model

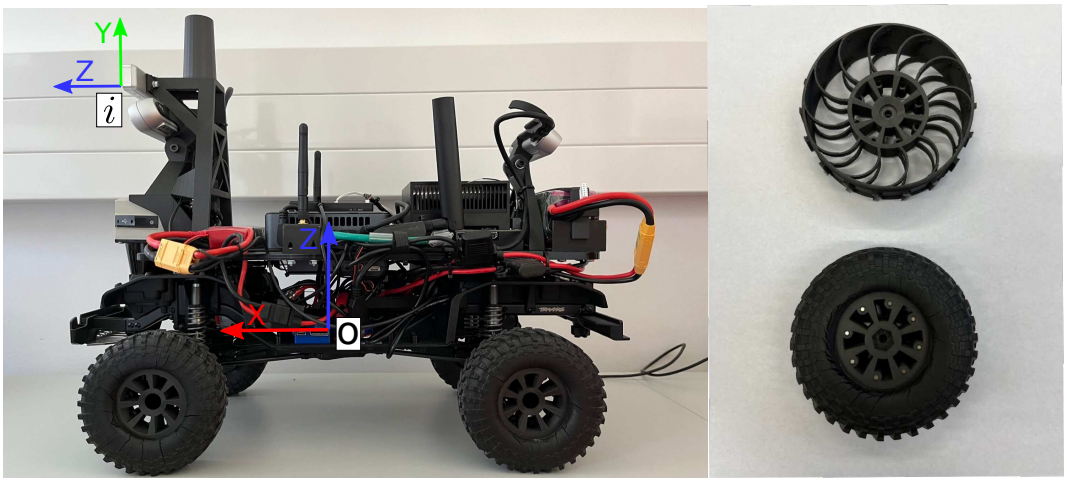


Figure 6.3: Left: Our mobile robot is a modified 1/10 electric RC car equipped with an Intel Realsense T265 stereo camera. Right: We primarily use the bottom wheel in our experiments and also evaluate with wheels without rubber tire (top).

estimate as an initial guess of the neutral position of the suspension for the online optimization phase.

6.4 Experiments

We evaluate our proposed method with real-world data collected by our robot (Figure 6.3). It is a 1/10 scale electric car equipped with a Realsense T265 stereo-fisheye camera with built-in IMU. We control the robot manually and record the images at 30 Hz, IMU measurements at 200 Hz and control inputs at 20 Hz. We use the ACADO toolbox (Houska et al., 2011) to efficiently integrate the dynamic model and compute the derivatives of the solution with respect to parameters and initial values using automatic differentiation. We evaluate tracking and prediction accuracy in various scenarios and compare with the original VIO. We use the same settings for pure VIO and our method with 3 active recent and 7 keyframes in the window. Similar like Lee et al. (2020) and Wu et al. (2017), we use the results of global mapping as ground-truth, where we set every frame as keyframe and perform dense bundle adjustment for accuracy and consistency. For all experiments, the weight values are set to 10^3, 10^4, 1, 30 $\mathrm{d}t \times 10^4$, 20, and 30 $\mathrm{d}t \times 10^8$ for the dynamics, geometry constraint, extrinsics initial prior, extrinsics random walk, dynamics parameters absolute prior, and dynamics parameters random walk factor, respectively, where $\mathrm{d}t$ is the time interval between two frames. Parameters ψ, τ and σ are found as 0.202, 2.335 and 10 by offline optimization. The variance threshold for the accelerometer bias is 4.5×10^{-4}.

6.4.1 Experiment Setup

To evaluate tracking and prediction accuracy and calibration capability, we collect data in both indoor and outdoor scenes. Each scene contains two different groups of terrain and each group has three data sequences. The indoor scenes include a lobby with tile floor and a corridor with concrete floor. In contrast to the indoor scene, the outdoor scene is not perfectly flat with small bumpiness and contains places with concrete surface and paved brick road. For each scene, we control our robot with various steering inputs and either full or varying throttle inputs. In all data recordings, the robot starts from a static pose. The mass m and wheel distance $l_f + l_r$ for our robot are measured directly. The momentum of inertia I_z and the distance between center of mass and front wheel l_f are approximated from CAD software. We then perform the offline initialization strategy described in Subsection 6.3.5 to initialize the dynamic model parameters $\mathbf{p}_{\text{dyn}}$ using two short trajectories of 10 s in the corridor scene.

6.4.2 Tracking Accuracy Evaluation

We evaluate the tracking accuracy of our ST-VIO by comparing the relative pose error (RPE) (Z. Zhang and Scaramuzza, 2018) with the original VIO, to show the relative improvement. Note that since no previous method is available that optimizes a single-track dynamic model with VIO, we can only compare our approach with the baseline VIO in our experiments. The comparison between our base VIO and other popular VIO methods can be found in Campos et al. (2021) and Usenko et al. (2020). The RPE value is generated by computing the errors over $10, 20, \ldots, 50\%$ sequence lengths of the full trajectory. We exclude the standing still segment at the end of the trajectories to avoid biasing the tracking error to low values in this trivial case. Table 6.1 provides average results over all indoor and outdoor sequences. For the indoor data, our approach ST-VIO overall improves trajectory accuracy.

The outdoor data are challenging for our method, as the bumpy terrain could violate the single-track dynamic model. In most outdoor sequences, our method can still improve the accuracy. In the concrete data group with varying throttle, the rotational accuracy drops slightly. Besides the less even terrain, another reason could be that the vehicle speed in the varying throttle case is relatively small compared to the full throttle case, and integrating the dynamic model cannot improve the accuracy further.

We also perform an ablation study for tracking accuracy evaluation where only geometry constraints are applied, and the dynamics factor is deactivated. VIO with only geometry constraints shows similar accuracy of 0.133 m and 1.126 deg like the original VIO while our approach achieves 0.124 m and 1.093 deg RMSE of transl. and rot. RPE on average for all data sequences. The algorithm with dynamics factor diverges on some sequences without geometry constraints.

Table 6.1: Average trajectory RPE on indoor and outdoor sequences (ST-VIO: ours, VIO: pure VIO, -full: full throttle maneuver, -varying: varying throttle maneuver).

	transl. RMSE RPE [m]		rot. RMSE RPE [deg]	
dataset	VIO	ST-VIO	VIO	ST-VIO
lobby-full	0.118	**0.108**	1.633	**1.573**
lobby-varying	0.076	**0.069**	1.038	**1.006**
corridor-full	0.183	**0.174**	0.993	**0.941**
corridor-varying	0.120	**0.114**	0.675	**0.659**
concrete-full	0.176	**0.162**	1.500	**1.423**
concrete-varying	0.168	**0.164**	**1.192**	1.195
brick-full	0.108	**0.107**	1.220	**1.200**
brick-varying	0.116	**0.108**	0.764	**0.748**

Table 6.2: Average prediction RPE on indoor and outdoor sequences (init: offline-calibrated, calib: online-calibrated, -full: full throttle, -varying: varying throttle maneuver).

	transl. RMSE RPE [m]		rot. RMSE RPE [deg]	
dataset	init	calib	init	calib
lobby-full	1.120	**0.453**	22.383	**7.558**
lobby-varying	0.764	**0.477**	9.588	**5.790**
corridor-full	0.550	**0.524**	7.367	**6.128**
corridor-varying	0.647	**0.552**	6.092	**4.883**
concrete-full	1.962	**0.508**	25.884	**6.283**
concrete-varying	1.007	**0.518**	12.191	**4.202**
brick-full	0.701	**0.310**	12.077	**3.781**
brick-varying	0.625	**0.496**	8.613	**5.493**

6.4.3 Prediction Accuracy Evaluation

We also evaluate prediction accuracy for different time horizons (0.33 s, 0.66 s, 1.66 s, 3.33 s and 10 s) to validate the online calibration of the parameters. The prediction is computed with the current dynamic model parameters from the start state estimated by ST-VIO at each frame. The prediction is a relative 2D pose in the vehicle body frame. To compare with ground-truth camera poses, we project the relative camera pose to the vehicle body frame using the optimized extrinsics. The standing still part at the end of the trajectories is excluded again to avoid including the perfect but trivial predictions (zero relative pose and velocity) into the evaluation. RPE results are summarized in Table 6.2. The prediction accuracy is denoted as calib and init using online calibrated and initial parameters, respectively. We observed improved prediction accuracy using the online calibrated parameters across all sequences. For the corridor sequences, the improvement is relatively modest. This is attributed to the fact that offline optimization is performed on sequences captured within the corridor scene.

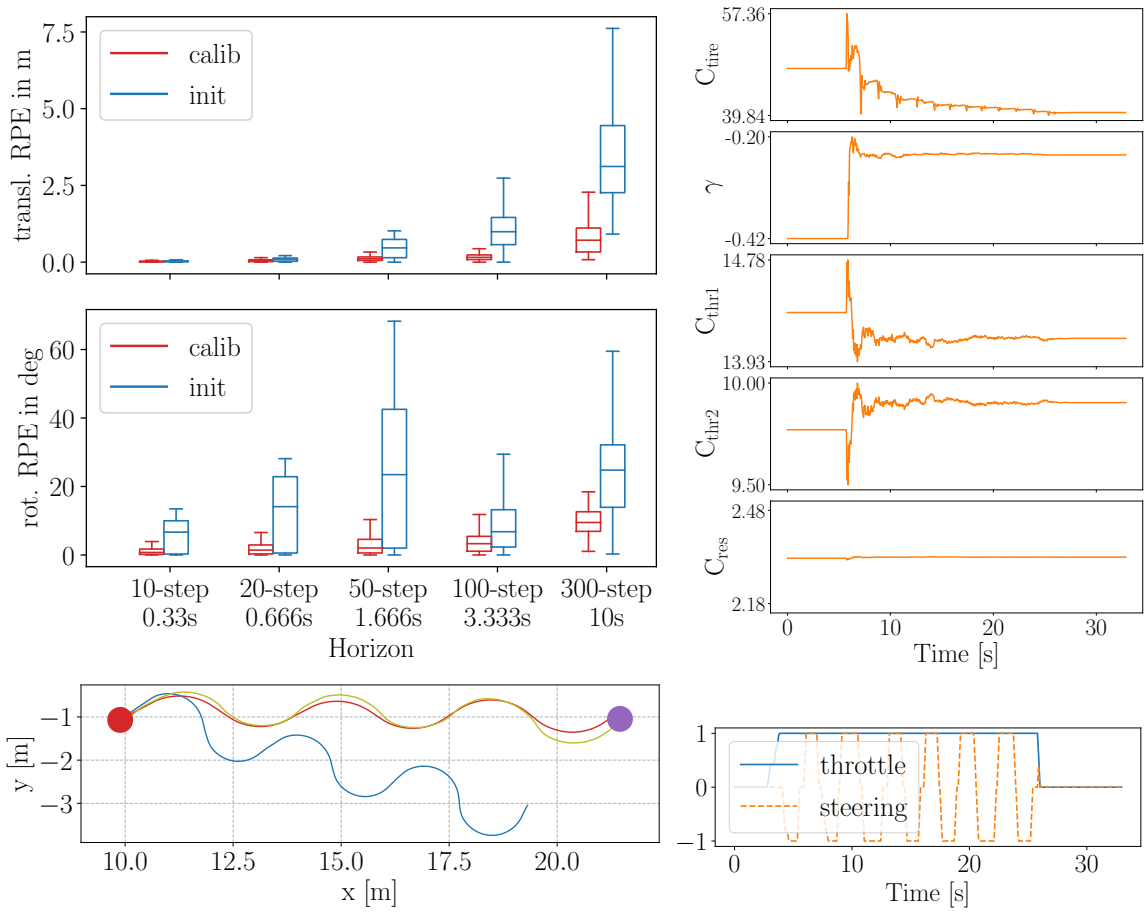


Figure 6.4: Top left: online calibration (calib) by ST-VIO for the new wheels clearly improves prediction over offline calibration (init) for the old wheels. Top right: evolution of online calibrated parameters (calib). Bottom left: 10 s prediction results (red: calib, blue: init, yellow trajectory: ground-truth, red/purple circle: start/end, rotated by 30 deg for visualization). Bottom right: control inputs.

Change of Robot Properties We additionally collect a data sequence with different wheels (top one in Figure 6.3). These special wheels are printed by a 3D printer without rubber tire outside and have significantly lower traction. We run our method using the initial dynamic model parameters for the old set of wheels as in the experiments above. The bottom image in Figure 6.4 illustrates the prediction error qualitatively. The top left image in Figure 6.4 demonstrates that the online calibrated parameters show significantly less error than the parameters calibrated offline for the old wheel for various time horizons. The top right figure in Figure 6.4 depicts the evolution of the online calibrated parameters during the optimization. The biggest adaptation is in the tire coefficient C_{tire} and steering ratio γ parameters, while the throttle mapping parameters are only adapted in a relatively small range.

Stop-and-Go Motion We also collect three sequences in each indoor environment for repeated stop-and-go motion with varying steering to demonstrate that our singularity-free formulation enables calibration and prediction in this case. In the lobby scene, the average transl. RMSE RPE improves from 0.405 m to 0.340 m, the average rot. RMSE RPE improves from 11.294 deg to 7.874 deg. For the corridor sequences, the average transl.

RMSE RPE changes from 0.332 m to 0.387 m, the average rot. RMSE RPE improves slightly from 8.812 deg to 8.645 deg. For the corridor sequences, the online calibration does not further enhance prediction accuracy since offline calibration was already conducted on similar sequences. In the lobby scene, our method demonstrates adaptability and improves prediction even under stop-and-go movements.

6.4.4 Computation Time

We evaluate the run-time of our method compared to the pure VIO on an Intel i9-10900X CPU@3.70GHz with 20 threads. For pure VIO processing, one frame takes 6.15 ms on average, while our dynamics augmented VIO needs 16.60 ms. Our method needs about three times more run-time than the original VIO yet is still real-time capable since the average run-time is below the frame interval (33.3 ms).

6.5 Conclusion

In this chapter, we propose ST-VIO for wheeled robots which integrates a singularity-free single-track vehicle dynamic model and optimizes the vehicle parameters online together with the VIO states in a sliding window fashion. The vehicle model is tightly integrated by introducing a dynamics factor which minimizes the difference between the pose and velocity prediction based on the model and the state estimate. A multistep objective function is constructed by predicting the pose and velocity from the first frame until the end frame in the window. In experiments, we demonstrate that our method is real-time capable and can improve the tracking accuracy on flat ground, especially for motions with full throttle. We also demonstrate that online calibration can improve motion prediction and adapt the parameters to changes in the environment and wheel properties. In future work, we aim to integrate vehicle models which can handle more complex terrain properties.

7 Conclusion

In this thesis, we have explored two key areas in robotics: state estimation and model adaptation. First, we introduced several innovative approaches for vision-based state estimation under specific motion constraints. These constraints act as prior knowledge, enhancing both the accuracy and robustness of state estimation. We investigated the application of such constraints across various scenarios, including event-based object motion estimation and visual-inertial ego motion estimation.

Unlike much of the previous research, our work also explores the integration of state estimation with the motion model adaptation problem. This integration addresses these two key areas simultaneously. We utilize motion models as prior knowledge to constrain state estimation and jointly calibrate the model parameters. Consequently, this approach not only leverages the motion model to enhance the precision of state estimation but also refines and adapts the motion model online in response to environmental changes, thereby enabling accurate forward predictions.

7.1 Summary

Event-Based Object Motion Estimation In Chapter 3, we present an optimization-based 6-DoF pose tracking method that utilizes both event and image measurements. The method integrates high-frequency events, which provide limited data, with low-frequency images that offer more detailed information about the scene. Our approach consists of a two-layer processing pipeline. In the first layer, we estimate the object trajectory from the event stream in relation to a reference image frame using a generative event measurement model. In the second layer, we refine the trajectory using a direct image alignment model based on the image frames. The trajectory is parameterized using cubic B-splines, which allows for the integration of these two different data modalities at different rates. This spline approach helps in managing the sparse data from high-rate events effectively, ensuring smoother motion tracking.

One limitation of our approach is its dependency on ground-truth pose data for initialization. A potential workaround is to initially employ a conventional image-based object tracking method to estimate the object pose under slow motion conditions, using this estimate to bootstrap our method. Another limitation arises from the smoothness assumption inherent in the spline modeling, which may not adequately represent motion involving rapid directional changes. Employing a higher order spline could mitigate this issue. Moreover, in our current method, we project events onto the image frame for the generative event measurement model, which omits information from pixels where no events are detected. A more effective strategy could be to project keypoints from the image to the event frame. Additionally, our method performs incremental tracking, and the accumulated errors can lead to tracking failures. Integrating event and image

measurements tightly, rather than processing them in separate layers, could enhance the robustness and accuracy of the tracking.

VIO and Leg Odometry Fusion In Chapter 4, we delve into ego motion estimation for a quadruped robot using visual-inertial measurements combined with forward kinematics. We introduce a lightweight EKF-based framework that integrates VIO estimates as measurements with leg odometry to determine the robot's pose and velocity. These state variables are estimated at a high frequency, which is crucial for dynamic local motion control. Due to the low rate and delays associated with VIO estimates, we enhance these estimates with IMU predictions. This augmentation allows the VIO output to match the higher IMU rate, significantly reducing delays. Furthermore, during agile jumping movements, camera measurements often suffer from motion blur, which leads to inaccuracies in height estimation. To address this, we employ leg kinematics and contact detection to create a pseudo-height measurement that corrects the drift from the VIO-based height estimate. Our approach is validated through real-world experiments conducted in both indoor and outdoor environments. Indoor testing shows that the augmented VIO estimate, enhanced with IMU predictions, substantially improves the accuracy of our EKF state estimation. Additionally, our kinematics-based height measurement helps to mitigate drift in height estimates over time during dynamic jumping gaits. In outdoor settings, we qualitatively assess our method by executing jumping and trotting motions, as well as their combinations, on various flat terrains using our EKF state estimates.

The primary limitation of our method is the assumption that the feet in contact with the ground remain static. This assumption may not hold if there is slippage, particularly on uneven or low-friction terrain. To address this, we could employ a more sophisticated contact model that incorporates direct contact and terrain measurements.

Online Calibration of Kinematic Model with VIO In Chapter 5 we merge VIO with a velocity-control-based kinematic motion model, which acts as a motion constraint. This model serves as a relative motion factor between each pair of images. We employ inverse kinematics to calculate the 2D velocity of the robot between two consecutive images based on the VIO state estimate. We then minimize the difference between this calculated velocity and the velocity control input for the robot. Given the inherent discrepancies between the control input and the actual robot action, we utilize an RBF kernel along the time domain to compute a weighted average from a window of recent control inputs, which serves as an effective control measure. The parameters of the RBF kernel are calibrated online jointly with the VIO state estimation. Our real-world experiments demonstrate that integrating this kinematic motion model and a plane motion constraint with VIO enhances the accuracy of the motion estimation. Furthermore, the online-calibrated kinematic model substantially improves the prediction of robot motion.

One limitation of our approach arises from the assumption that the VIO results are sufficiently accurate, which is not always the case. For example, when dealing with uncalibrated camera intrinsics, the accuracy of VIO can be compromised and the estimate can be biased. To mitigate this, introducing another modality like GPS, which provides absolute motion measurements, could be beneficial.

Online Calibration of Dynamic Model with VIO In Chapter 6, we incorporate a more complex motion model, specifically a single-track dynamic model, into VIO as a motion constraint for wheeled robots. We have modified this single-track model to be singularity-free, ensuring it is differentiable and can be seamlessly integrated into the VIO optimization framework. Utilizing this dynamic model, we calculate the relative motion, including multi-step predictions of pose and velocity, and compare these with the VIO estimates to minimize their differences. We optimize the model's parameters online jointly with the VIO states. In our experiments, we demonstrate that our approach not only improves tracking accuracy but also adapts the motion model to changes in the terrain and wheel properties, enabling more accurate forward prediction.

While our method is adaptable to variations in terrain, it currently assumes that the motion occurs on flat ground, which restricts its applicability. Extending the model to accommodate 3D movements and complex interactions between the robot and terrain, while maintaining a lightweight framework that can be calibrated online, presents a significant challenge. An alternative strategy could involve employing a more expressive model, such as a deep neural network, combined with a large dataset collected offline, to learn the dynamics conditioned on image and depth measurements.

7.2 Future Work

Uncertainty Estimation We employ the motion model as a probabilistic factor in the factor graph optimization framework for the state estimation problem. However, the motion model cannot fully capture the underlying mechanisms of the real world, making the incorporation of uncertainty into the model predictions beneficial. Currently, we determine this uncertainty empirically. Future work could explore methods for learning this uncertainty from data, such as estimating the uncertainty of the motion model on pre-collected training data using statistical methods. There are existing studies that employ Gaussian processes to integrate a probabilistic description of system dynamics in control problems (Kuss and Rasmussen, 2003; Rottmann and Burgard, 2009). Incorporating a Gaussian process as the motion model prior to constrain the VIO problem could be a viable area for further exploration. This approach could potentially enhance the robustness and accuracy of the VIO system by more effectively managing the inherent uncertainties of the model.

The state estimation problem addressed in this thesis operates within the frameworks of MLE and MAP, both aimed at deriving the best single estimate for the state. Often,

the state estimate is utilized by downstream tasks such as control and planning, where understanding the uncertainty of the state's posterior distribution could be advantageous for robust and safe operations. However, accurately capturing this uncertainty is challenging. Even if we assume a Gaussian prior for the state, the VIO system's high non-linearity and complexity make finding an analytical solution impractical. One potential approach is to estimate the uncertainty of the linearized system as an approximation, where the posterior would still be Gaussian. Such an approximation essentially fits a Gaussian distribution at the mode of the true posterior distribution. However, due to non-linearity, the true posterior may be non-Gaussian. Consequently, the effectiveness of this approximation requires further investigation.

Combination with Control Since we perform online model calibration with VIO to enable accurate forward prediction, it naturally lends itself to integration with model-predictive control (MPC) and adaptive control. Both methodologies are well-established and active research areas. MPC computes optimal actions through the minimization of a cost function by predicting states over a specified horizon. Meanwhile, adaptive control adjusts control parameters dynamically to adapt to changes and uncertainties within the system dynamics (Nguyen-Tuong and Peters, 2011). This synergy could enhance the system's responsiveness and efficiency in real-world applications. Furthermore, successful identification of model parameters requires collecting adequate data by sufficiently exciting the system. At present, we manually gather data through diverse maneuvers. In the future, exploring methods to generate optimal robot excitation trajectories could be beneficial.

Motion Model Learning In this thesis, the motion models used to constrain the state estimation are primarily analytical models based on physical laws. While these analytical models offer transparency and have a compact parameter space, they struggle to cope with the complexities of robot systems and their operation in unstructured environments due to unmodelled effects. Conversely, learning-based methods using more expressive models, such as deep neural networks, can directly capture complex nonlinearities from data. However, these models require that the distribution of the training data adequately covers the model's state space to ensure effective generalization. A promising approach is the physics-augmented neural ODE (R. T. Q. Chen et al., 2018; Yin et al., 2021), which synergizes the strengths of both analytical and learning-based methods. For instance, we could employ a simple feedforward neural network to model acceleration while retaining the kinematic parts used in Equation (6.2). The kinematic model could potentially describe general rigid body motion without introducing bias and aid the neural network in learning the acceleration. Furthermore, such hybrid dynamic model could be applied to any point on the rigid body, not just the center of mass, which could often be challenging to measure accurately in practical settings.

Robust state estimation and model learning continue to pose significant challenges in the field of robotics. Looking ahead, it is essential for robots to possess the capability

for continual learning, enabling them to adapt to dynamic environments. The method of online model calibration with an analytical model, as discussed in this thesis, adjusts parameters in response to environmental changes and represents a foundational approach to this adaptation. However, this adaptability is constrained by the model's structure and does not support the accumulation of knowledge over time. It is therefore crucial to explore methods that allow robots to acquire and retain knowledge throughout their operational life. Such lifelong learning could enhance the robot's internal model and improve its state estimation capabilities. Additionally, advancing robots' abilities to develop high-level behavior models and make multimodal predictions is vital. These advancements could greatly broaden the possibilities for robots.

Bibliography

Aghli, S. and C. Heckman (2018). 'Online System Identification and Calibration of Dynamic Models for Autonomous Ground Vehicles'. In: *Proceedings of International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2018.8460691 (cited on page 73).

Azad, P., D. Münch, T. Asfour, and R. Dillmann (2011). '6-DoF Model-Based Tracking of Arbitrarily Shaped 3D Objects'. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2011.5979950 (cited on page 3).

Barfoot, T. D. (2017). *State Estimation for Robotics*. 1st. USA: Cambridge University Press (cited on pages 1, 60).

Blösch, M., C. Gehring, P. Fankhauser, M. Hutter, M. A. Hoepflinger, and R. Siegwart (2013). 'State Estimation for Legged Robots on Unstable and Slippery Terrain'. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2013.6697236 (cited on page 38).

Blösch, M., M. Hutter, M. A. Hoepflinger, S. Leutenegger, C. Gehring, C. D. Remy, and R. Siegwart (2012). 'State Estimation for Legged Robots - Consistent Fusion of Leg Kinematics and IMU'. In: *Proceedings of Robotics: Science and Systems (RSS)*. DOI: 10.7551/mitpress/9816.003.0008 (cited on pages 4, 38, 39).

Brossard, M., A. Barrau, and S. Bonnabel (2019). 'RINS-W: Robust Inertial Navigation System on Wheels'. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS40897.2019.8968593 (cited on page 53).

– (2020). 'AI-IMU Dead-Reckoning'. In: *IEEE Transactions on Intelligent Vehicles (T-IV)*. DOI: 10.1109/TIV.2020.2980758 (cited on page 53).

Brossard, M. and S. Bonnabel (2019). 'Learning Wheel Odometry and IMU Errors for Localization'. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2019.8794237 (cited on page 53).

Bryner, S., G. Gallego, H. Rebecq, and D. Scaramuzza (2019). 'Event-Based, Direct Camera Tracking from a Photometric 3D Map using Nonlinear Optimization'. In: *Proceedings of International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2019.8794255 (cited on pages 3, 25, 26, 29).

Calli, B., A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar (2015). 'The YCB Object and Model Set: Towards Common Benchmarks for Manipulation Research'. In: *Proceedings of International Conference on Advanced Robotics (ICAR)*. DOI: 10.1109/ICAR.2015.7251504 (cited on page 31).

Campos, C., R. Elvira, J. J. Gomez, J. M. M. Montiel, and J. D. Tardós (2021). 'ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM'.

In: *IEEE Transactions on Robotics (T-RO)*. DOI: 10.1109/TRO.2021.3075644 (cited on pages 51, 82).

Camurri, M., M. Ramezani, S. Nobili, and M. Fallon (2020). 'Pronto: A Multi-Sensor State Estimator for Legged Robots in Real-World Scenarios'. In: *Frontiers in Robotics and AI*. DOI: 10.3389/frobt.2020.00068 (cited on pages 38, 39, 41–43, 45).

Chang, A. X., T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu (2015). *ShapeNet: An Information-Rich 3D Model Repository* (cited on page 31).

Chen, R. T. Q., Y. Rubanova, J. Bettencourt, and D. K. Duvenaud (2018). 'Neural Ordinary Differential Equations'. In: *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)* (cited on page 90).

Chen, Y., M. Zhang, D. Hong, C. Deng, and M. Li (2019). 'Perception System Design for Low-Cost Commercial Ground Robots: Sensor Configurations, Calibration, Localization and Mapping'. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS40897.2019.8968078 (cited on page 54).

Chilian, A., H. Hirschmüller, and M. Görner (2011). 'Multisensor Data Fusion for Robust Pose Estimation of a Six-Legged Walking Robot'. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2011.6094484 (cited on pages 4, 39).

Choi, C. and H. I. Christensen (2012a). 'Robust 3D Visual Tracking Using Particle Filtering on the Special Euclidean Group: A Combined Approach of Keypoint and Edge Features'. In: *International Journal of Robotics Research (IJRR)*. DOI: 10.1177/0278364912437213 (cited on page 2).

– (2012b). '3D Textureless Object Detection and Tracking: An Edge-Based Approach'. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2012.6386065 (cited on page 2).

Cioffi, G., L. Bauersfeld, and D. Scaramuzza (2023). 'HDVIO: Improving Localization and Disturbance Estimation with Hybrid Dynamics VIO'. In: *Proceedings of Robotics: Science and Systems (RSS)*. DOI: 10.15607/RSS.2023.XIX.071 (cited on page 72).

Corke, P. (2013). *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*. 1st. Springer (cited on page 27).

Dame, A., V. A. Prisacariu, C. Y. Ren, and I. Reid (2013). 'Dense Reconstruction Using 3D Object Shape Priors'. In: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR.2013.170 (cited on page 30).

Dang, Z., T. Wang, and F. Pang (2018). 'Tightly-coupled Data Fusion of VINS and Odometer Based on Wheel Slip Estimation'. In: *Proceedings of IEEE International Conference on Robotics and Biomimetics (ROBIO)*. DOI: 10.1109/ROBIO.2018.8665337 (cited on page 54).

Deng, X., A. Mousavian, Y. Xiang, F. Xia, T. Bretl, and D. Fox (2021). 'PoseRBPF: A Rao–Blackwellized Particle Filter for 6-D Object Pose Tracking'. In: *IEEE Transactions on Robotics (T-RO)*. DOI: 10.1109/TRO.2021.3056043 (cited on page 2).

Deray, J. and J. Solà (2020). 'Manif: A Micro Lie Theory Library for State Estimation in Robotics Applications'. In: *Journal of Open Source Software*. DOI: 10.21105/joss.01371 (cited on pages 12, 13, 61).

Dhédin, V., H. Li, S. Khorshidi, L. Mack, A. K. C. Ravi, A. Meduri, P. Shah, F. Grimminger, L. Righetti, M. Khadiv, and J. Stückler (2023). 'Visual-Inertial and Leg Odometry Fusion for Dynamic Locomotion'. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA48891.2023.10160898 (cited on pages 7, 37, 50).

Dubeau, E., M. Garon, B. Debaque, R. Charette, and J.-F. Lalonde (2020). 'RGB-D-E: Event Camera Calibration for Fast 6-DOF Object Tracking'. In: *Proceedings of IEEE International Symposium on Mixed and Augmented Reality (ISMAR)* (cited on pages 3, 25).

Dudzik, T., M. Chignoli, G. Bledt, B. Lim, A. Miller, D. Kim, and S. Kim (2020). 'Robust Autonomous Navigation of a Small-Scale Quadruped Robot in Real-World Environments'. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS45743.2020.9340701 (cited on page 39).

Engel, J., V. Koltun, and D. Cremers (2018). 'Direct Sparse Odometry'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*. DOI: 10.1109/TPAMI.2017.2658577 (cited on page 29).

Forster, C., L. Carlone, F. Dellaert, and D. Scaramuzza (2015). 'IMU Preintegration on Manifold for Efficient Visual-Inertial Maximum-a-Posteriori Estimation'. In: *Proceedings of Robotics: Science and Systems (RSS)*. DOI: 10.15607/RSS.2015.XI.006 (cited on page 18).

Gallego, G., T. Delbruck, G. M. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davison, J. Conradt, K. Daniilidis, and D. Scaramuzza (2020). 'Event-Based Vision: A Survey'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*. DOI: 10.1109/TPAMI.2020.3008413 (cited on pages 1, 23, 24).

Geneva, P., K. Eckenhoff, W. Lee, Y. Yang, and G. Huang (2020). 'OpenVINS: A Research Platform for Visual-Inertial Estimation'. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA40945.2020.9196524 (cited on page 3).

Goodfellow, I., Y. Bengio, and A. Courville (2016). *Deep Learning*. MIT Press (cited on pages 13, 14).

Grimminger, F., A. Meduri, M. Khadiv, J. Viereck, M. Wüthrich, M. Naveau, V. Berenz, S. Heim, F. Widmaier, T. Flayols, et al. (2020). 'An Open Torque-Controlled Modular Robot Architecture for Legged Locomotion Research'. In: *IEEE Robotics and Automation Letters (RA-L)*. DOI: 10.1109/LRA.2020.2976639 (cited on pages 38, 44).

Guttikonda, S., J. Achterhold, H. Li, J. Boedecker, and J. Stückler (2023). 'Context-Conditional Navigation with a Learning-Based Terrain- and Robot-Aware Dynamics Model'. In: *Proceedings of European Conference on Mobile Robots (ECMR)*. DOI: 10.1109/ECMR59166.2023.10256414 (cited on page 8).

Hartley, R., M. Ghaffari, R. M. Eustice, and J. W. Grizzle (2020). 'Contact-Aided Invariant Extended Kalman Filtering for Robot State Estimation'. In: *International Journal of Robotics Research (IJRR)*. DOI: 10.1177/0278364919894385 (cited on pages 4, 39).

Hartley, R., M. G. Jadidi, L. Gan, J.-K. Huang, J. W. Grizzle, and R. M. Eustice (2018a). 'Hybrid Contact Preintegration for Visual-Inertial-Contact State Estimation Using Factor Graphs'. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2018.8593801 (cited on page 38).

– (2018b). 'Hybrid Contact Preintegration for Visual-Inertial-Contact State Estimation Using Factor Graphs'. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2018.8593801 (cited on pages 4, 39).

Hartley, R., J. Mangelson, L. Gan, M. Ghaffari Jadidi, J. M. Walls, R. M. Eustice, and J. W. Grizzle (2018c). 'Legged Robot State-Estimation Through Combined Forward Kinematic and Preintegrated Contact Factors'. In: *Proceedings of International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2018.8460748 (cited on pages 4, 39).

Hermansdorfer, L., R. Trauth, J. Betz, and M. Lienkamp (2020). 'End-to-End Neural Network for Vehicle Dynamics Modeling'. In: *Proceedings of IEEE Congress on Information Science and Technology (CiSt)*. DOI: 10.1109/CiSt49399.2021.9357196 (cited on page 5).

Hesch, J. A., D. G. Kottas, S. L. Bowman, and S. I. Roumeliotis (2014). 'Consistency Analysis and Improvement of Vision-aided Inertial Navigation'. In: *IEEE Transactions on Robotics (T-RO)*. DOI: 10.1109/TRO.2013.2277549 (cited on pages 1, 4, 60, 61).

Hochreiter, S. and J. Schmidhuber (1997). 'Long Short-Term Memory'. In: *Neural Computation*. DOI: 10.1162/neco.1997.9.8.1735 (cited on page 73).

Houska, B., H. Ferreau, and M. Diehl (2011). 'ACADO Toolkit – An Open Source Framework for Automatic Control and Dynamic Optimization'. In: *Optimal Control Applications and Methods*. DOI: 10.1002/oca.939 (cited on page 81).

Jiang, S., W. Lin, Y. Cao, Y. Wang, J. Miao, and Q. Luo (2021). 'Learning-Based Vehicle Dynamics Residual Correction Model for Autonomous Driving Simulation'. In: *Proceedings of IEEE Conference on Intelligent Transportation Systems (ITSC)*. DOI: 10.1109/ITSC48978.2021.9564486 (cited on page 73).

Jung, J. H., J. Cha, J. Y. Chung, T. I. Kim, M. H. Seo, S. Y. Park, J. Y. Yeo, and C. G. Park (2020). 'Monocular Visual-Inertial-Wheel Odometry Using Low-Grade IMU in Urban Areas'. In: *IEEE Transactions on Intelligent Transportation Systems (T-ITS)*. DOI: 10.1109/TITS.2020.3018167 (cited on page 54).

Kabzan, J., L. Hewing, A. Liniger, and M. N. Zeilinger (2019). 'Learning-Based Model Predictive Control for Autonomous Racing'. In: *IEEE Robotics and Automation Letters (RA-L)*. DOI: 10.1109/LRA.2019.2926677 (cited on pages 54, 72–75).

Kavraki, L., P. Svestka, J.-C. Latombe, and M. Overmars (1996). 'Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces'. In: *IEEE Transactions on Robotics and Automation*. DOI: 10.1109/70.508439 (cited on page 52).

Kim, H., S. Leutenegger, and A. Davison (2016). 'Real-Time 3D Reconstruction and 6-DoF Tracking with an Event Camera'. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. DOI: 10.1007/978-3-319-46466-4_21 (cited on pages 3, 24–26).

Kim, J.-H., S. Hong, G. Ji, S. Jeon, J. Hwangbo, J.-H. Oh, and H.-W. Park (2021). 'Legged Robot State Estimation With Dynamic Contact Event Information'. In: *IEEE Robotics and Automation Letters (RA-L)*. DOI: 10.1109/LRA.2021.3093876 (cited on pages 4, 39).

Kim, T., H. Lee, and W. Lee (2022). 'Physics Embedded Neural Network Vehicle Model and Applications in Risk-Aware Autonomous Driving Using Latent Features'. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS47612.2022.9981303 (cited on page 5).

Kim, Y., B. Yu, E. M. Lee, J. Kim, H. Park, and H. Myung (2022). 'STEP: State Estimator for Legged Robots Using a Preintegrated Foot Velocity Factor'. In: *IEEE Robotics and Automation Letters (RA-L)*. DOI: 10.1109/LRA.2022.3150844 (cited on pages 4, 39).

Klein, G. and D. W. Murray (2006). 'Full-3D Edge Tracking with a Particle Filter'. In: *Proceedings of British Machine Vision Conference (BMVC)* (cited on page 3).

Krull, A., F. Michel, E. Brachmann, S. Gumhold, S. Ihrke, and C. Rother (2014). '6-DOF Model Based Tracking via Object Coordinate Regression'. In: *Proceedings of Asian Conference on Computer Vision (ACCV)*. DOI: 10.1007/978-3-319-16817-3_25 (cited on page 2).

Kuss, M. and C. Rasmussen (2003). 'Gaussian Processes in Reinforcement Learning'. In: *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)* (cited on page 89).

Kuwata, Y., J. Teo, S. Karaman, G. Fiore, E. Frazzoli, and J. P. How (2008). 'Motion Planning in Complex Environments Using Closed-Loop Prediction'. In: *Proceedings of AIAA Guidance, Navigation, and Control Conference and Exhibit (GNC)*. DOI: 10.2514/6.2008-7166 (cited on page 52).

LaValle, S. and J. Kuffner (1999). 'Randomized Kinodynamic Planning'. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ROBOT.1999.770022 (cited on page 52).

Lee, W., K. Eckenhoff, Y. Yang, P. Geneva, and G. Huang (2020). 'Visual-Inertial-Wheel Odometry with Online Calibration'. In: *Proceedings of IEEE/RSJ International Conference*

on Intelligent Robots and Systems (IROS). DOI: 10.1109/IROS45743.2020.9341161 (cited on pages 4, 54, 61, 64, 72, 81).

Leutenegger, S., S. Lynen, M. Bosse, R. Siegwart, and P. Furgale (Mar. 2015). 'Keyframe-Based Visual–Inertial Odometry Using Nonlinear Optimization'. In: *International Journal of Robotics Research (IJRR)*. DOI: 10.1177/0278364914554813 (cited on pages 4, 16, 18, 20, 21, 31, 51, 71).

Li, H. and J. Stückler (2021). 'Tracking 6-DoF Object Motion from Events and Frames'. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA48506.2021.9561760 (cited on pages 7, 23).

– (2022). 'Visual-Inertial Odometry with Online Calibration of Velocity-Control Based Kinematic Motion Models'. In: *IEEE Robotics and Automation Letters (RA-L)*. DOI: 10.1109/LRA.2022.3169837 (cited on pages 7, 51, 73).

– (2024). 'Online Calibration of a Single-Track Ground Vehicle Dynamics Model by Tight Fusion with Visual-Inertial Odometry'. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. To appear, preprint available at https://doi.org/10.48550/arXiv.2309.11148 (cited on pages 7, 71).

Li, Y., G. Wang, X. Ji, Y. Xiang, and D. Fox (2018). 'DeepIM: Deep Iterative Matching for 6D Pose Estimation'. In: *Proceedings of European Conference on Computer Vision (ECCV)*. DOI: 10.1007/978-3-030-01231-1_42 (cited on page 2).

Li, Z., N. A. Piga, F. Di Pietro, M. Iacono, A. Glover, L. Natale, and C. Bartolozzi (2023). 'Hybrid Object Tracking with Events and Frames'. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS55552.2023.10342300 (cited on page 3).

Liu, J., W. Gao, and Z. Hu (2019). 'Visual-Inertial Odometry Tightly Coupled with Wheel Encoder Adopting Robust Initialization and Online Extrinsic Calibration'. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS40897.2019.8967607 (cited on page 54).

– (2021). 'Bidirectional Trajectory Computation for Odometer-Aided Visual-Inertial SLAM'. In: *IEEE Robotics and Automation Letters (RA-L)*. DOI: 10.1109/LRA.2021.3059564 (cited on page 54).

Lucas, B. D. and T. Kanade (1981). 'An Iterative Image Registration Technique with an Application to Stereo Vision'. In: *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI))* (cited on page 18).

Lynch, K. M. and F. C. Park (2017). *Modern Robotics: Mechanics, Planning, and Control*. 1st. USA: Cambridge University Press (cited on pages 9, 10, 75).

Ma, F., J. Shi, Y. Yang, J. Li, and K. Dai (2019). 'ACK-MSCKF: Tightly-Coupled ackermann multi-state constraint kalman filter for autonomous vehicle localization'. In: *Sensors*. DOI: 10.3390/s19214816 (cited on pages 4, 51, 54, 73, 77).

Ma, Y., S. Soatto, J. Kosecka, and S. S. Sastry (2003). *An Invitation to 3-D Vision: From Images to Geometric Models*. Springer (cited on page 17).

Meduri, A., P. Shah, J. Viereck, M. Khadiv, I. Havoutis, and L. Righetti (2023). 'BiConMP: A Nonlinear Model Predictive Control Framework for Whole Body Motion Planning'. In: *IEEE Transactions on Robotics (T-RO)*. DOI: 10.1109/TRO.2022.3228390 (cited on pages 37, 39, 44).

Mitrokhin, A., C. Fermüller, C. Parameshwara, and Y. Aloimonos (2019a). 'Event-Based Moving Object Detection and Tracking'. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2018.8593805 (cited on pages 3, 25).

Mitrokhin, A., C. Ye, C. Fermüller, Y. Aloimonos, and T. Delbruck (2019b). 'EV-IMO: Motion Segmentation Dataset and Learning Pipeline for Event Cameras'. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS40897.2019.8968520 (cited on page 25).

Mourikis, A. I. and S. I. Roumeliotis (2007). 'A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation'. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ROBOT.2007.364024 (cited on pages 3, 51, 53).

Mueggler, E., G. Gallego, H. Rebecq, and D. Scaramuzza (2018). 'Continuous-Time Visual-Inertial Odometry for Event Cameras'. In: *IEEE Transactions on Robotics (T-RO)*. DOI: /10.1109/TRO.2018.2858287 (cited on page 3).

Nguyen-Tuong, D. and J. Peters (2011). 'Model Learning for Robot Control: A Survey'. In: *Cognitive processing*. DOI: 10.1007/s10339-011-0404-1 (cited on page 90).

Nisar, B., P. Foehn, D. Falanga, and D. Scaramuzza (2019). 'VIMO: Simultaneous Visual Inertial Model-Based Odometry and Force Estimation'. In: *IEEE Robotics and Automation Letters (RA-L)*. DOI: 10.1109/LRA.2019.2918689 (cited on page 72).

Nobili, S., M. Camurri, V. Barasuol, M. Focchi, D. G. Caldwell, C. Semini, and M. F. Fallon (2017). 'Heterogeneous Sensor Fusion for Accurate State Estimation of Dynamic Legged Robots'. In: *Proceedings of Robotics: Science and Systems (RSS)*. DOI: 10.15607/RSS.2017.XIII.007 (cited on page 39).

Nocedal, J. and S. J. Wright (2006). *Numerical Optimization*. 2e. New York, NY, USA: Springer (cited on pages 14, 16).

Patron-Perez, A., S. Lovegrove, and G. Sibley (July 2015). 'A Spline-Based Trajectory Representation for Sensor Fusion and Rolling Shutter Cameras'. In: *International Journal of Computer Vision (IJCV)*. DOI: 10.1007/s11263-015-0811-3 (cited on page 27).

Prisacariu, V. A., A. V. Segal, and I. Reid (2013). 'Simultaneous Monocular 2D Segmentation, 3D Pose Recovery and 3D Reconstruction'. In: *Proceedings of the Asian Conference on Computer Vision (ACCV)*. DOI: 10.1007/978-3-642-37331-2_45 (cited on page 30).

Qin, T., P. Li, and S. Shen (2018). 'VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator'. In: *IEEE Transactions on Robotics (T-RO)*. DOI: 10.1109/TRO.2018.2853729 (cited on pages 4, 16, 71).

Rebecq, H., T. Horstschaefer, G. Gallego, and D. Scaramuzza (2017). 'EVO: A Geometric Approach to Event-Based 6-DOF Parallel Tracking and Mapping in Real Time'. In: *IEEE Robotics and Automation Letters (RA-L)*. DOI: 10.1109/LRA.2016.2645143 (cited on page 24).

Rebecq, H., D. Gehrig, and D. Scaramuzza (2018). 'ESIM: an Open Event Camera Simulator'. In: *Proceedings of the 2nd Conference on Robotics Learning (CoRL)* (cited on page 31).

Reina, G., M. Paiano, and J. L. Blanco-Claraco (2017). 'Vehicle Parameter Estimation Using a Model-Based Estimator'. In: *Mechanical Systems and Signal Processing (MSSP)*. DOI: 10.1016/j.ymssp.2016.06.038 (cited on pages 5, 71).

Rosten, E., R. Porter, and T. Drummond (2010). 'Faster and Better: A Machine Learning Approach to Corner Detection'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*. DOI: 10.1109/TPAMI.2008.275 (cited on page 18).

Rotella, N., M. Bloesch, L. Righetti, and S. Schaal (2014). 'State Estimation for a Humanoid Robot'. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2014.6942674 (cited on page 38).

Rottmann, A. and W. Burgard (2009). 'Adaptive Autonomous Control Using Online Value Iteration with Gaussian Processes'. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ROBOT.2009.5152660 (cited on page 89).

Rublee, E., V. Rabaud, K. Konolige, and G. Bradski (2011). 'ORB: An Efficient Alternative to SIFT or SURF'. In: *Proceedings of International Conference on Computer Vision (ICCV)*. DOI: 10.1109/ICCV.2011.6126544 (cited on page 30).

Spielberg, N. A., M. Brown, N. R. Kapania, J. C. Kegelman, and J. C. Gerdes (2019). 'Neural Network Vehicle Models for High-Performance Automated Driving'. In: *Science Robotics*. DOI: 10.1126/scirobotics.aaw1975 (cited on page 5).

Sturm, J., N. Engelhard, F. Endres, W. Burgard, and D. Cremers (2012). 'A Benchmark for the Evaluation of RGB-D SLAM Systems'. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2012.6385773 (cited on page 31).

Sutton, R. S. and A. G. Barto (2018). *Reinforcement Learning: An Introduction*. Second. The MIT Press (cited on page 1).

Teng, S., M. W. Mueller, and K. Sreenath (2021). 'Legged Robot State Estimation in Slippery Environments Using Invariant Extended Kalman Filter with Velocity Update'. In: *Proceedings of International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA48506.2021.9561313 (cited on pages 4, 39).

Teulière, C., E. Marchand, and L. Eck (2010). 'Using Multiple Hypothesis in Model-Based Tracking'. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ROBOT.2010.5509284 (cited on page 3).

Thrun, S., W. Burgard, and D. Fox (2005). *Probabilistic Robotics*. MIT Press (cited on pages 56, 68).

Usenko, V., N. Demmel, D. Schubert, J. Stückler, and D. Cremers (2020). 'Visual-Inertial Mapping With Non-Linear Factor Recovery'. In: *IEEE Robotics and Automation Letters (RA-L)*. DOI: 10.1109/LRA.2019.2961227 (cited on pages 4, 21, 40, 44, 51, 52, 55, 65, 71, 74, 77, 80, 82).

Vasco, V., A. Glover, E. Mueggler, D. Scaramuzza, L. Natale, and C. Bartolozzi (2017). 'Independent motion detection with event-driven cameras'. In: *Proceedings of the International Conference on Advanced Robotics (ICAR)*. DOI: 10.1109/ICAR.2017.8023661 (cited on pages 3, 24).

Wang, R., N. Yang, J. Stückler, and D. Cremers (2017). 'DirectShape: Photometric Alignment of Shape Priors for Visual Vehicle Pose and Shape Estimation'. In: *Proceedings of International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA40945.2020.9197095 (cited on page 29).

Wen, B., J. Tremblay, V. Blukis, S. Tyree, T. Müller, A. Evans, D. Fox, J. Kautz, and S. Birchfield (2023). 'BundleSDF: Neural 6-DoF Tracking and 3D Reconstruction of Unknown Objects'. In: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR52729.2023.00066 (cited on page 2).

Wen, B., W. Yang, J. Kautz, and S. Birchfield (2024). 'FoundationPose: Unified 6D Pose Estimation and Tracking of Novel Objects'. In: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)* (cited on page 2).

Weydert, M. (2012). 'Model-Based Ego-motion and Vehicle Parameter Estimation Using Visual Odometry'. In: DOI: 10.1109/MELCON.2012.6196577 (cited on pages 73, 74).

Wielitzka, M., M. Dagen, and T. Ortmaier (2015). 'Joint Unscented Kalman Filter for State and Parameter Estimation in Vehicle Dynamics'. In: *Proceedings of IEEE Conference on Control and Applications (CCA)*. DOI: 10.1109/CCA.2015.7320894 (cited on pages 5, 71, 73).

Williams, G., P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou (2016). 'Aggressive Driving with Model Predictive Path Integral Control'. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2016.7487277 (cited on page 54).

Wisth, D., M. Camurri, and M. Fallon (2023). 'VILENS: Visual, Inertial, Lidar, and Leg Odometry for All-Terrain Legged Robots'. In: *IEEE Transactions on Robotics (T-RO)*. DOI: 10.1109/TRO.2022.3193788 (cited on page 38).

Wisth, D., M. Camurri, and M. F. Fallon (2019). 'Robust Legged Robot State Estimation Using Factor Graph Optimization'. In: *IEEE Robotics and Automation Letters (RA-L)*. DOI: 10.1109/LRA.2019.2933768 (cited on page 39).

Wu, K. J. and S. I. Roumeliotis (Sept. 2016). *Unobservable Directions of VINS Under Special Motions*. Tech. rep. Department of Computer Science, University of Minnesota (cited on page 62).

Wu, K. J., C. X. Guo, G. Georgiou, and S. I. Roumeliotis (2017). 'VINS on Wheels'. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2017.7989603 (cited on pages 2, 4, 45, 51, 53, 59, 60, 64, 72, 73, 79–81).

Xiong, L., R. Kang, J. Zhao, P. Zhang, M. Xu, R. Ju, C. Ye, and T. Feng (2022). 'G-VIDO: A Vehicle Dynamics and Intermittent GNSS-Aided Visual-Inertial State Estimator for Autonomous Driving'. In: *IEEE Transactions on Intelligent Transportation Systems (T-ITS)*. DOI: 10.1109/TITS.2021.3107873 (cited on pages 73, 74, 77).

Xu, J., Q. Luo, K. Xu, X. Xiao, S. Yu, J. Hu, J. Miao, and J. Wang (2019). 'An Automated Learning-Based Procedure for Large-scale Vehicle Dynamics Modeling on Baidu Apollo Platform'. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS40897.2019.8968102 (cited on pages 5, 73).

Xue, Y., H. Li, S. Leutenegger, and J. Stückler (2022). 'Event-Based Non-Rigid Reconstruction from Contours'. In: *Proceedings of British Machine Vision Conference (BMVC)*. Available at https://bmvc2022.mpi-inf.mpg.de/0078.pdf (cited on page 8).

– (2024). 'Event-Based Non-Rigid Reconstruction of Low-Rank Parametrized Deformations from Contours'. In: *International Journal of Computer Vision (IJCV)*. DOI: 10.1007/s11263-024-02011-z (cited on page 8).

Yang, Y. and G. Huang (2019). 'Observability Analysis of Aided INS With Heterogeneous Features of Points, Lines, and Planes'. In: *IEEE Transactions on Robotics (T-RO)*. DOI: 10.1109/TRO.2019.2927835 (cited on pages 51, 54).

Yin, Y., V. L. Guen, J. Dona, E. de Bézenac, I. Ayed, N. Thome, and P. Gallinari (2021). 'Augmenting Physical Models with Deep Networks for Complex Dynamics Forecasting'. In: *Journal of Statistical Mechanics: Theory and Experiment*. DOI: 10.1088/1742-5468/ac3ae5 (cited on page 90).

You, C. and P. Tsiotras (2017). 'Vehicle Modeling and Parameter Estimation Using Adaptive Limited Memory Joint-State UKF'. In: *Proceedings of the American Control Conference (ACC)*. DOI: 10.23919/ACC.2017.7962973 (cited on pages 5, 71, 73).

You, S. H., J. O. Hahn, and H. Lee (2009). 'New Adaptive Approaches to Real-Time Estimation of Vehicle Sideslip Angle'. In: *Control Engineering Practice*. DOI: 10.1016/j.conengprac.2009.07.002 (cited on page 5).

Zhang, P., L. Xiong, Z. Yu, R. Kang, M. Xu, and D. Zeng (2020). 'VINS-PL-Vehicle: Points and Lines-Based Monocular VINS Combined with Vehicle Kinematics for Indoor Garage'. In: *IEEE Intelligent Vehicles Symposium (IV)*. DOI: 10.1109/IV47402.2020.9304639 (cited on pages 73, 77).

Zhang, V., S. M. Thornton, and J. C. Gerdes (2018). 'Tire Modeling to Enable Model Predictive Control of Automated Vehicles From Standstill to the Limits of Handling'. In: *International Symposium on Advanced Vehicle Control (AVEC)* (cited on pages 74, 76).

Zhang, Z., G. Gallego, and D. Scaramuzza (2018). 'On the Comparison of Gauge Freedom Handling in Optimization-Based Visual-Inertial State Estimation'. en. In: *IEEE Robotics and Automation Letters (RA-L)*. DOI: 10.1109/LRA.2018.2833152 (cited on page 54).

Zhang, Z. and D. Scaramuzza (2018). 'A Tutorial on Quantitative Trajectory Evaluation for Visual(-Inertial) Odometry'. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2018.8593941 (cited on pages 45, 64, 82).

Zheng, F. and Y.-H. Liu (2019). 'Visual-Odometric Localization and Mapping for Ground Vehicles Using SE(2)-XYZ Constraints'. In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2019.8793928 (cited on page 54).