State estimation and localization of legged robots : a tightly-coupled approach based on a-posteriori maximization

Mederic Fourmy

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :

l'Institut National des Sciences Appliquées de Toulouse (INSA de Toulouse)

Présentée et soutenue le *21 mars 2022* **par :**

Médéric FOURMY

State estimation and localization of legged robots: a tightly-coupled approach based on a-posteriori maximization

---

JURY

Vincent PADOIS	Directeur de recherche	Rapporteur
Teresa VIDAL-CALLEJA	Associate Professor	Rapportrice
Simon LACROIX	Directeur de recherche	Examinateur
Ludovic RIGHETTI	Associate Professor	Examinateur
Diane LARLUS	Cadre scientifique	Examinatrice
Olivier STASSE	Directeur de recherche	Examinateur
Nicolas MANSARD	Directeur de recherche	Examinateur
Joan SOLÀ	Cientifico Titular CSIC	Examinateur

École doctorale et spécialité :

EDSYS : Robotique 4200046

Unité de Recherche :

LAAS - Laboratoire d'Analyse et d'Architecture des Systèmes (UPR 8001)

Directeur(s) de Thèse :

Nicolas MANSARD et *Joan SOLÀ*

Rapporteurs :

Vicent PADOIS et *Teresa VIDAL-CALLEJA*

Contents

Chapter 1

Introduction

The concept of sensory feedback is at the core of the study of cybernetics agents targeting autonomy. To realize actions, an agent needs a representation of the current state of the world, which is inferred from its past sensors measurements. Current actions may then affect this state, which is reflected in a change of its sensors output. This creates a feedback loop, that, if designed well, leads to a stable, auto-regulated system.

For most robotics applications, the complexity of building a proper feedback behavior comes both from the difficulty to capture a proper representation of the robot world and the diversity of possible control reactions. The world representation is built by an estimator that fuses several sources of information. The challenge is then to choose the set of appropriate sensors for each application and to design an efficient estimation algorithm to fuse them. Let us first discuss how this feedback loop has evolved while robotics was growing mature.

1.1 From factory automation to dancing robots

Within the field of robotics, the implementation of the feedback loop has seen dramatic changes over the years, propelled by the changing nature of the mechatronic systems, in particular in the actuation and sensor array, the applications at play, and the mathematical formulations used to model the systems. The progression of the perception side of the loop, extracting meaningful information from sensor data, can be divided into a few steps that accompany the evolution of robotics, from fixed manipulators to agile legged robots.

The first major robotic use was in the industrial space: starting in the early 60's [1], arm manipulators have been progressively integrated into many assembly lines, especially in the automotive industry. This application requires the performance of highly precise, repetitive tasks, which are predefined by specialized human operators. These highly rigid robots are controlled in position and fixed to the ground, which usually limits the perception needs to the relative angles between their different parts.

On the other end of the spectrum, researchers started to equip wheeled mobile platforms Fig. 1.1b in controlled laboratory environments with exteroceptive sensors [Nil84; CL85] to apply planning algorithms, using mainly range sensors to control the presence

[1] The Unimate manipulator Fig. 1.1a was adopted by General Motors to displace hot die casting pieces to cooling tanks or assembly lines, first tests starting in 1961.

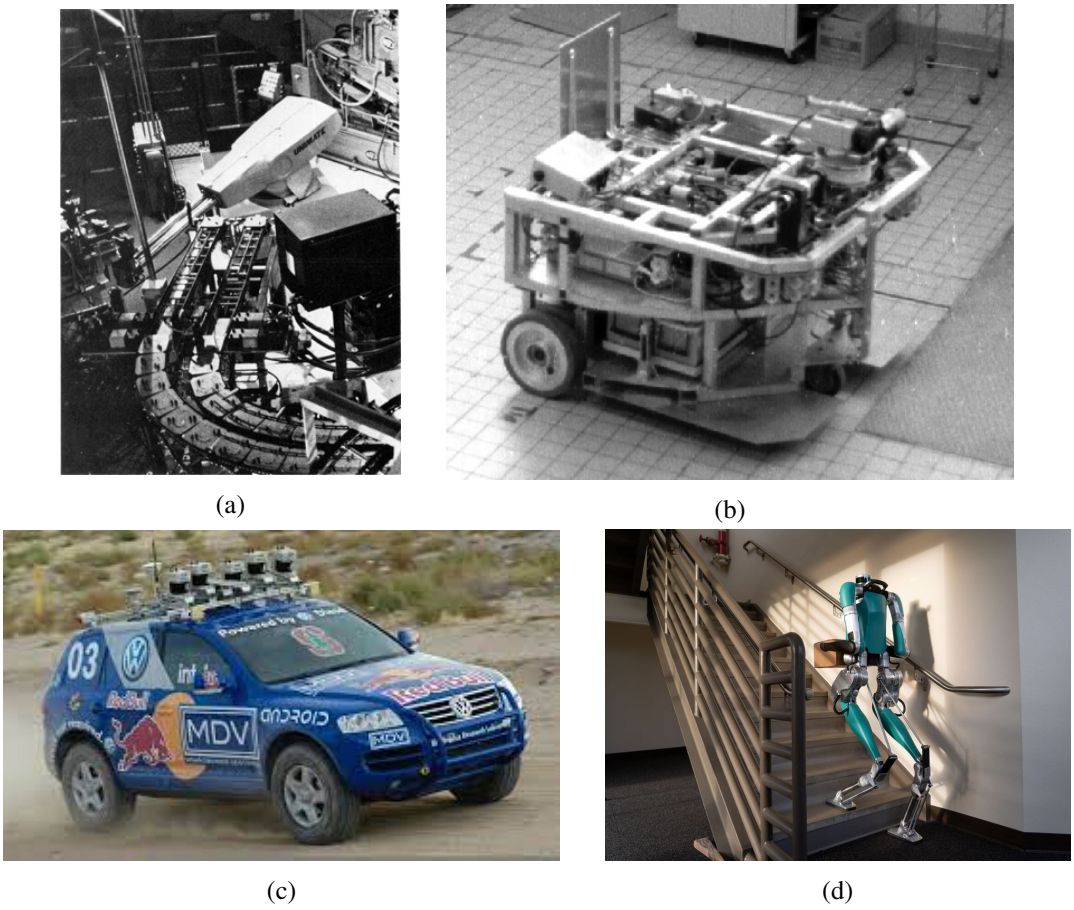


Figure 1.1: Evolution of robotics platforms from factory manipulators to humanoid robots. (a) Unimate factory manipulator, (b): Hilare mobile robotics research platform developed at LAAS-CNRS, (c): Stanford autonomous car winner of the 2005 DARPA grand challenge, (d): Digit (Agility Robotics) commercial humanoid robot.

of obstacles, along with wheel odometry [2] to detect relative displacement. Research on mobile robotics moved to outdoors applications, using motorized vehicles as research platforms. The 2005 DARPA grand challenge offered one of the first large scale proof of concept of autonomous cars, where a few teams managed to safely drive 150 miles paths in desert-environments [Thr+06] (see Fig. 1.1c), while the 2007 DARPA Urban challenge concentrated on urban road environments [Urm+08]. In both cases, cars were equipped with GPS, IMU's, cameras, and a metric map of the path. Most teams relied heavily on global positioning, exteroceptive sensors being used for minor checks and corrections [Hil+14]. Nowadays, the most successful autonomous car systems seem to tend toward exclusive use of vision for local navigation (lane following, lane changing, overtaking, etc.) [3].

In this landscape of autonomous systems, legged robots Fig. 1.1d (humanoid or quadrupeds) are singular in many regards. First and foremost, they are inherently unstable dynamical systems that require continuous active control by applying forces at chosen locations of the environment. Therefore, they require an acute sense of balance, in which Inertial Measurement Units and contact detection play important roles. Secondly, they are mobile

[2] Odometry is, in the large sense, information about the relative motion of a robot obtained from the integration of a motion sensor (wheel encoder, IMU, Doppler Velocity Logs, etc.)

[3] As Andrej Karpathy puts it, "Lidar is really a shortcut"

platforms, whose primary function is to be able to navigate environments to perform tasks such as inspection or manipulation. A perception of the environment is therefore required for any meaningful tasks to be undertaken, contrary to fixed manipulators. Thirdly, they can in theory navigate cluttered, unstructured environments, which extends their operational capacities, compared to wheeled platforms. This makes for challenging perceptual problems that require taking into account their many embarked sensor modalities.

In this thesis, we are interested in extending the perceptual capabilities of legged robots (humanoids and quadrupeds). This kind of robot requires both a high-rate (1 kHz), low latency estimates of its physical quantities to balance itself, like a precise direction of the gravity, and an accurate environment representation for safe navigation and interaction. As we will explain in the next chapter, these tasks are oftentimes handled separately in the literature. We believe that it is possible to improve existing systems by a tight integration of the many sensor modalities available on such a platform.

As a first justification of this point of view, let us consider a biological example that will motivate this approach.

1.2 Biological equivalent, an example

A digression through a biological example may provide some intuitions about complexity of the problem. For instance, the simple task of picking up a box makes use of almost all of our senses: our eyes to spot the object, our vestibular system to balance against gravity, our proprioception (kinesthetic sense) to extend our arms toward the object, our sense of touch to assess its softness, etc. However, this simplistic enumeration hides a more complex reality: in many cases, the information processed by our brain to make decisions results from the fusion of multi-modal stimuli. For instance, it is demonstrated that during communication between animals or humans, vocal stimuli, mouth movements, and facial gestures are tightly integrated [MD05; Sug+06]: watching a speaker's mouth changes our perception of its speech. The combination of these stimuli can affect the recovered information, either clarifying the underlying message [MS83] or compromising it, as for the McGurk effect [MM76]. These coupling phenomena being highlighted by group experiments [MM76], the location of the fusion can be demonstrated by recording the activity of neurons [RRM97].

This tight coupling is also present in the sense of movement. One major example is the vestibulo-ocular reflex [ML81]: our eye movements are regulated by the semicircular canals signals, that essentially provide angular-velocity measurements, stabilizing our gaze. This reflex is very fast: less than 10 ms [Aw+96] and is constantly being recalibrated throughout our life [ML81]. Notice that this performance is made possible by the colocation of the two senses and their proximity to the central nervous system. Evidence suggests that this concentration of senses and processing capability in the head has an evolutionary advantage that can be explained by several reasons [Bai+18]: it is easier to process stimuli from sensory organs whose spatial relation is fixed; shorter pathways between senses and the brain increases information bandwidth; independence of the head helps to stabilize the senses inputs while enabling active exploration strategies. Despite these facts, except for humanoid robots, few systems come equipped with a truly multisensory head nowadays. This would be justified by estimation approaches that are capable of truly coupling the different sensor modalities.

1.3 How to implement artificial sensing for legged robots

To be able to able to achieve any meaningful tasks, robots need to display a sufficient level of dynamical intelligence and perception capabilities. In particular, the perception should be adapted to real-time control at the high frequencies usually found in legged-robot controllers (in the kHz range). The estimated state should accurately reflect both the dynamics of the robot as well as its surounding environment. Balance controllers in particular require a precise estimation of the gravity vector direction, as well centroidal quantities (center of mass, angular momentum, etc.). In our opinion, this implies that the ideal perception system should encourage the multiplication of perception sources both in the number of sensors and their variety. Special care should be taken to obtain sensors of the best quality and to carry out a precise calibration, both of their inner parameters (intrinsics) as well as their location on the robot (extrinsics). In addition, to maximize the accuracy of the estimator, any available correlations between variables and prior knowledge should in principle be taken into account. We also need to enable online calibration of many sensor biases and fixed parameters. To achieve these goals, the development of *tightly-coupled* estimators, which exploit as many data correlations as possible between the sensor measurements, should be undertaken [NKT15; WCF21]. On the other end, *loosely-coupled* estimator usually neglect part of the correlations, potentially resulting in suboptimal solutions.

Two main perception problems are typically considered onboard a legged robot. On the first hand, the sense of balance is replicated through high-frequency local estimator fusing IMU and contact information to obtain the robot base velocity directed with respect to the gravity field. On the other hand, exteroceptive sensors (cameras, LIDARs) are used to localize the robot with respect to a representation of its environment, ideally built on the flight. This second layer is typically estimated at lower frequencies.

It is not exactly clear what the optimal set of sensors that needs to be integrated on legged platforms is (though Inertial Measurement Units, kinesthesis, and cameras or LIDARS are becoming more and more standard for industrial applications). This set may depend heavily on the type of application, the size of the platform (LIDARS are too bulky for smaller quadrupeds), or the acceptable price range of the robot. Thus, it is important to allow for a great *modularity* in the design of estimators. And even though modularity together with tight coupling may seem incompatible at first, we believe these two assets must be attained simultaneously. We think that this can be done by different means. First, by using a factor graph formulation together with a modular front-end/back-end architecture, that naturally leads to a tightly-coupled, optimization based, Maximum a Posteriori estimation. Second, by a flexible software architecture that allows a general formulation of estimation problems (which is the endeavor of WOLF [Sol+21][4], that we have used and contributed to during the preparation of this thesis). Third, by endeavoring to generalize the mathematical formulations of the measurement models as much as possible.

1.4 Thesis statement and organization of the manuscript

This thesis aims at contributing to the estimation of legged robots by taking into account information from many sensors, of many types, and in a modular way. Tightly-coupled estimation, whose necessity will be recurrently corroborated in this thesis, maximizes the

[4]The WOLF repository, documentation and examples can be found in www.iri.upc.edu/wolf

observability of the system, as opposed to loosely-coupled methods. In particular, tightly-coupled methods makes it possible to estimate the biases and calibration parameters on top of state variables. The Maximum a Posteriori approach, which is best described with the Factor Graph framework, lends itself comfortably to tightly-coupled approaches. I shall seek to demonstrate it by formulating several measurement models, in particular generalizing some existing approaches to other sensor modalities in the first part of this thesis. In the second part, I will then display the operational capacity of the system on several proofs of concepts, building blocks for a future whole-body estimator, providing both gravity aware estimation for balance control, and world reconstruction for navigation. Those intermediate systems will enable to experimentally qualify the performances, and feasability of the method.

This thesis is organized in two parts:

Chapter 2 presents a literature overview of state estimation legged robots, that we will use to position our objectives.

Chapter 3 serves as a general introduction to Factor Graph optimization using the Maximum a Posteriori. We emphasize the special treatment of variables belonging to manifolds and give a brief introduction to Lie theory, which is used extensively in Chapter 6.

Chapter 4 describes two different measurement models used in the object-level visual-inertial systems that we built.

Chapter 5 presents the use of robot kinematics to obtain leg-odometry measurements.

Chapter 6 introduces a generalization of the IMU pre-integration theory. Examples of the classic on-manifold pre-integration are recalled and then extended to a compact Lie group formulation.

Chapter 7 shows that the generalized pre-integration theory can be use to pre-integrate force sensors present at end effectors. Along with the centroidal kinematics, this enables to obtain unbiased estimates of the centroidal states of legged robots

The chapters of the second part present three different applications where we fuse several of the sensing modalities described so far. Regarding their practical implementation, we contributed to different parts of the WOLF state estimation framework [Sol+21] which was submitted to RAL journal this year. We present these applications in the chronological order of their development, reflecting the opinions that we had at those respective times.

Chapter 8 presents the first application, which is a visual-inertial object-level SLAM system based on fiducial markers. This chapter describes results presented in Humanoids 2019 conference paper [Fou+19].

In **Chapter 9** we propose a whole-body (base and centroidal states) estimator based on the fusion of IMU, kinematics, and force-torque sensors. This work was presented at ICRA 2021 conference [Fou+21].

In **Chapter 10**, a visual-inertial SLAM system using deep-learning-based object pose estimation is presented. This work will be submitted to IROS 2022 conference [Deb+21].

Chapter 11 describes a multi sensor dataset taken at LAAS on the Solo quadruped on which we plan to benchmark our future estimators.

In **Chapter 12**, we will present the conclusions and perspectives of this work.

Chapter 2

State estimation for legged robots: a literature review

The field of state estimation for autonomous systems is at the frontier between many fields including control theory, probability theory, nonlinear optimization, etc. The general aim is to design and implement computationally tractable algorithms using noisy causal sensor measurements that can be embedded in feedback-controlled systems. With the advent of digital computers and the theoretical breakthroughs of Kalman [Kal60] among others, a vast family of observers has been developed [SSM62; BD67; WVH01; Thr+04]. Applications range the full scope of autonomous systems from spacecraft guidance [McG+85] to autonomous underwater vehicle navigation [LB16]. Each application has its own specific sets of requirements, depending on the physical nature of the system and its environment, available sensors, and embedded computation power. Legged robots in particular require high frequency (kHz) and low latency (<1 ms) estimates owing to the inherent instability of their dynamics. Moreover, they interact with the environment through intermittent contacts to move around, which requires to plan in advance the motion using Model Predictive Control (MPC). These tight requirements have given birth to a wealth of research works that are now commonly used in commercial products [Hut+16].

We will focus our review on works applying state estimation theory on legged robots, which include mainly quadrupeds and humanoid robots. The first part will be centered around proprioceptive estimation of the robot's base. We will then examine the estimation of centroidal quantities. The use of exteroceptive sensors to obtain information from the robot environment will then be described. Finally, we will see that a new class of optimization-based estimator is a promising alternative to filtering-based methods.

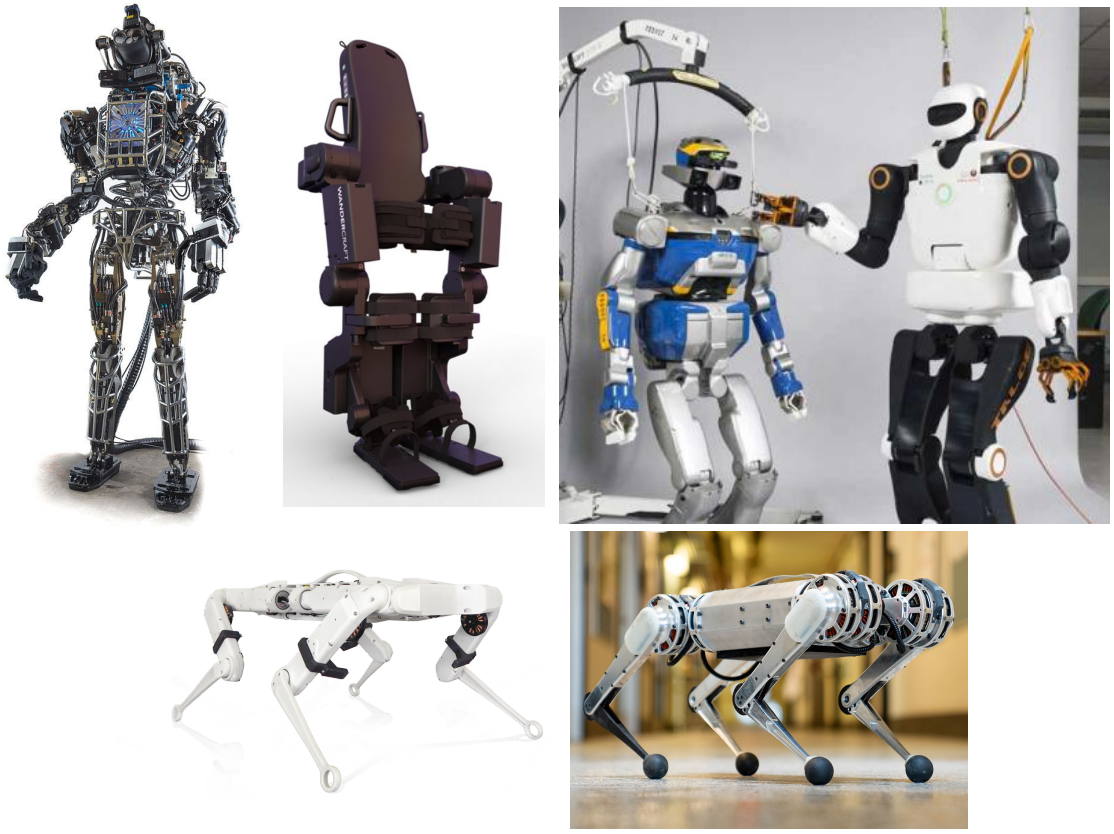


Figure 2.1: A few legged robots examples cited in the literature review. Top (left to right): Boston Dynamics' **Atlas** (2013 version), Wandercraft exoskeleton **Atalante** [Har+18a] , Kawada Industries' **HRP-2**, PAL Robotic's **Talos** [Sta+17]. Bottom: **Solo** from the Open Dynamic Robot Initiative [Gri+20], MIT **Mini Cheetah**.

2.1 Definitions

Let us first introduce terms and key concepts that are commonly used in the legged robot estimation literature.

Legged robots are poly-articulated systems that use a set of end-effectors (commonly referred to as feet whatever their nature) to move their main body by pushing on its environment. The kinematic chain of the robot is modeled as a graph composed of fixed shape segments (aka. links) linked together by articulations (aka. joints). Joints come in various forms (revolute, prismatic, ball...) and can be actuated or not. The *base* refers to a reference frame rigidly attached to the main body of the robot (often the trunk for quadrupeds and the pelvis for humanoids). We are most often interested in the estimation of the position, velocity, and orientation of this frame with regard to an inertial *world* frame, which is called the *base state*. An example of these frames is given for the Solo robot in Fig. 2.2. The base state is the main focus in most of legged robot estimation literature. In fact, knowing the base and other quantities directly measurable (such as joint angles), and assuming a perfect robot kinematic model, the state of any part of the poly-articulated system can be recovered using *forward kinematics*. This algorithm permits to compute *poses* (position and orientation) and *spatial velocities* (linear and angular velocities) of reference frames attached to the robot segments relative to the base or the world frames. Rigid body algorithms [Fea14] provide methods to efficiently compute the center of mass, linear and angular momentum of the poly-articulated systems. These computa-

tions are again based on the robot kinematic model as well as the segments inertia, which we together call the $kinodynamic$ robot model. This model is often found in URDF files (Unified Robot Description Format) which are generated from CAD models (Computer Aided Design).

The *centroidal state* refers to the center of mass, angular momentum, and their derivatives. The center of mass (barycenter of the robot segment masses) is a virtual point that is the main control variable for locomotion.

Proprioceptive sensors measure values about the internal state of the robot. For legged robots, they include joint encoders, strain gauges measuring either joint torques or end-effector torques, dedicated feet contact sensors, and Inertial Measurement Units (IMUs). These sensors do not directly provide information about the external robot environment and can therefore only be used to compute a drifting pose of the robot, the odometry [1].

Exteroceptive sensors (cameras, depth cameras, LIDARs...) provide information about the environment of the robot. They are needed to obtain non-drifting estimates of the robot pose. They are used either to localize with respect to a map or to build a representation of the environment online, that the robot can use to plan contacts for instance.

These measurement sources convey tremendously different types of information and are captured at various frequencies (*e.g.* tens of Hz for cameras, kHz for encoders). In this work, we assume that they are reasonably well time-synchronized, that is to say with delays inferior to the highest sampling period of all sensors. Measurements are generally *noisy* and sometimes *biased*. High levels of noise can come from the lower quality of the sensor (a low-cost MEMS IMU gyroscope exhibits much more noise than a high-end fiber-optic gyroscope) or from the process needed to obtain the data (joint velocities measurements are obtained by numerical differentiation of the encoder measurements). Noise can either be filtered using classical finite impulse response filters, though this may introduce delays, or by fusing measurements using probabilistic filters. Biases are constant or slowly varying quantities affecting measures and are either inherent to the sensor nature (IMU accelerometers and gyroscopes exhibit slowly varying biases) or to uncertainties of the used model (inaccuracies in the robot calibration produce biased kinematic estimates whatever the precision of the encoders). Biases may be compensated by explicitly introducing them in the estimator. It is for instance common to estimate IMU biases in visual-inertial filters.

With these few key terms explicated, we can now explore the legged estimation literature.

[1] The status of an IMU is debatable since an accelerometer measures the external gravity vectors which can be used to infer an approximate but non-drifting orientation. It is however often considered a proprioceptive sensor in the legged robot literature [RSR18; Sco+17; Yan+19; Lin+21]

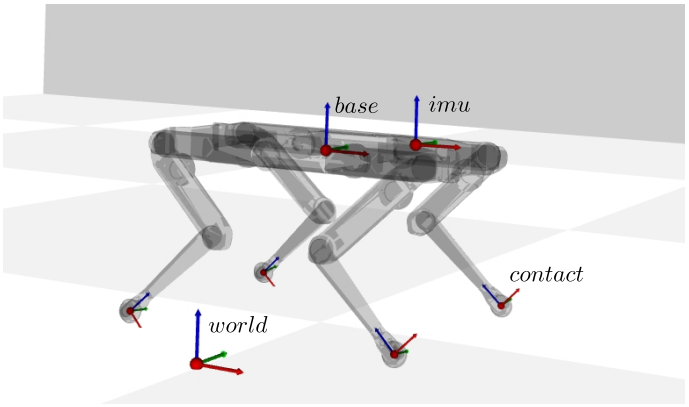


Figure 2.2: Solo 12 [Gri+20] important reference frames

2.2 Proprioceptive base estimation

We refer to proprioceptive base estimation for any state observer that estimates the base frame states by purely relying on proprioceptive sensors. The goal is then to design observers fusing IMU, kinematics, and possibly strain gauges measurements to obtain an odometry and orientation of the robot with respect to the gravity vector. The robot is akin to a blindfolded animal that has to balance and perform locomotion using only its inner ear, kinesthesis and sense of touch. Bloesch [Blo+13b; Rot+14] showed through an observability analysis that the absolute velocity, pitch, and roll angles as well as IMU biases are observable using only IMU and kinematic measurements when at least one contact is kept with the ground.

The roboticist has to make many design choices when building such a system. Those choices mainly include: the type of kinematic information used, the nature of the observer (tightly-coupled vs loosely-coupled), stable contacts detection, and whether extra sensors or modeling need to be used to mitigate model errors.

2.2.1 Kinematic information

Legged robots move by interacting with their environment through intermittent contacts. Once a stable contact (no slipping/tipping) is detected, the relative pose and velocity of the base with respect to the end-effector in contact can be computed through forward kinematics. Integrated over time, the relative displacement of the base of the robot can be inferred, a computation often referred to as *leg odometry* (by analogy with wheel odometry). This computation takes about 1 microsecond thanks to libraries such as [Car+19; HA17]. It uses readings from the joint encoders as well as the robot kinematic model. In systems with actuator reduction steps, encoders are usually placed before the reduction step to increase accuracy (precision of 0.002 degrees for Solo equipped with optical encoders [Gri+20]!). Joint velocities are often less precise since they are generally computed from numerical differentiation [Rot+]. The main sources of uncertainty usually come from inaccuracies in the parameters of the kinematic models such as approximate segment lengths, flexibilities [Vig+18; Vil+22], or joint backlash [Fal+14; Koo+16]. Bloesch [BH18] classifies kinematic measurement models in three categories that define the structure of this section. Those measurement models are summarized in Fig. 2.3.

Feet matching is the earliest example of leg-odometry to be used in the leg robotics

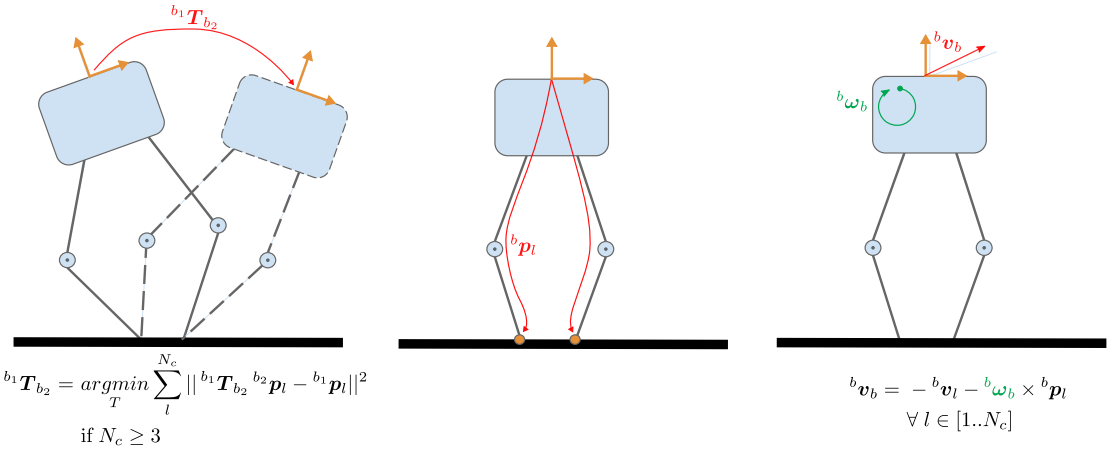
literature. Pioneered by [RK91] [2], *multiple feet matching* provides a relative 6D pose between moments during which at least three feet are in stable contact with the ground. For point-feet robots (such as most quadrupeds, if their feet are sufficiently small to avoid rolling on the contact surface), the problem is an instance of the orthogonal Procrustes problem [ELF97]. Follow up works adapted the method to smaller hexapods [LKK05] and began to fuse it with other sensors such as GPS [Gas+05; CED08] and most importantly IMUs [LKK06; RH11]. The inherent limitation of this method is that for point-feet robots it requires that at least three feet stay contact with the ground between given timesteps, limiting applications to hexapods or very slow quadruped gaits. *Single foot matching* is also possible for humanoid robots as each flat foot contact defines a 6D constraint [BH18]. After fixing the position of the contact by computing forward kinematics at the beginning of the stance phase using the current base estimate, this constraint directly produces 6D relative displacements [Fla+17; Xin+14; Joh+15]. This approach is less investigated for point-feet robots and was first demonstrated (to the best of our knowledge) in my work [Fou+21]. Another work published later this year [Kim+21] also includes this factor formulation along with a null velocity factor.

Instantaneous relative pose between the base and the foot can also be directly used as a residual in the estimator. This formulation was introduced in [Blo+13b; Blo+13a] for a point-feet quadruped using only relative positions. It was subsequently adapted for a humanoid robot [Rot+14], whose 6D foot constraints permitted to add orientation information. In this formulation, states variables corresponding to the robot feet pose have to be added to the estimator. This approach was adopted by several other groups [Har+18c; Har+18b; Har+20; Ble+18a]. Potential undetected slips and rolls of the foot are modeled in this context as a random walk on the stance foot position [Blo+13b; Rot+14]. When the estimator is formulated as a Kalman Filter [Kal60], this phenomenon is represented by a process noise on the feet position dynamics, which is a crucial parameter that the user needs to tune.

When a single point foot is in contact with the ground, the leg can move around the three remaining rotational degrees of freedom without changing encoder measurements. The base *relative velocity* can however be computed by using joint velocities and the angular velocity of the robot body. Joint velocities are usually obtained through numerical differentiation of the joint encoder outputs, which may result in noisy measurements [Rot+]. Gyro measurements are also subject to noise and affected by a bias that should be compensated for. These velocity measurements can then directly be used as residuals for the base velocity [Blo+13a; Ble+18a] or integrated over time as relative displacements [Ma+12; WCF20]. Some authors such as [Blo+13a; Ble+18a] use these types of measurements in conjunction with *instantaneous relative pose* which seems to reduce position drift. On the other end, one may argue [Fal+14] that in the case of an erroneous kinematic model (backlash, flexibilities), using exclusively direct velocity measurements prevents the estimator to become inconsistent when another source of position measurement is present (such as LIDAR localization).

These measurement models are the base of all legged robot estimators. We will see in the next section how they can be used with other data sources, in particular IMUs, to derive proprioceptive filters.

[2] An earlier example might exist in [WM86] even though the technical report is unclear about the method they used: "Leg-position feedback is used from legs in support phase for the purpose of correcting for gyro and integration drift in the inertial reference system."



(a) point-feet matching (b) point-feet direct (c) point-feet linear velocity

(d) Flat feet matching (e) Flat feet direct (f) Flat feet twist

Figure 2.3: Legged robots kinematic measurement models classification. Red arrows correspond to the residuals derived for each model using encoder measurements and contact information. Orange elements correspond to the state variables affected by these residuals. Figures (a),(b), and (c) are models for point feet robot, commonly used for quadrupeds. Figures (d),(e), and (f) are flat feet based models, commonly used for humanoid robots. The green rotational arrow represents gyroscope measurements used in addition to encoders for the point-feet linear velocity model (c). N_c: number of feet in contact, $\mathbf{T} \in SE(3)$: spatial transformation, $\mathbf{p} \in \mathbb{R}^3$: translation, $\nu \in \mathbb{R}^6$: spatial velocity [Fea14, Section 2.2].

2.2.2 Filter based data fusion

Many types of observer designs have been put to test on legged robots proprioceptive estimation. The problem being Markovian in nature, most are Bayes filters such as variants of the Kalman Filter [Kal60] and complementary filters [Hig75]. While a complete history of the evolution of proprioceptive filters is instructive [Gas+05; LKK05; LKK06; CED08; APL08; LAP11; CHG11; RH11; GS12; Ma+12; GS13], this has been treated exhaustively by other authors [Blo17; Cam17]. It is however interesting to delve into a particular dichotomy between different estimators implemented in recent works. Those can be roughly divided between *loosely-coupled* (LC) approaches, which divide the estimation in several steps whose results are being successively taken as fixed priors to the following steps and *tightly-coupled* (TC) approaches, which try to capture all statistical cross-correlations between the estimated states.

In the context of proprioceptive estimation, a common loosely-coupled filter is to precompute the base orientation independently by a first filter and using it as a fixed value in the next estimation steps. The 2015 Darpa robotic challenge, which mimicked a disaster intervention scenario, was an interesting testing field for humanoid robotics team implementing control and estimation strategies. For instance, the CMU [Fen+15] and IHMC [Joh+15] teams, that were both using the Atlas humanoid robot (picture in Fig. 2.1), explain that they implemented tightly-coupled nonlinear estimators for the early trials, based on ad hoc implementations of a proprioceptive filter, but ended up using the orientation estimation provided by the Atlas IMU without modification later in the competition preparation. This was helped by the fact that Atlas IMU was a tactical-grade fiber-optic model. An experimental comparison of two decoupled proprioceptive filters for HRP-2 was proposed [Fla+17]. For both, a complementary filter estimates the orientation of the base from IMU measurements only. The second step is either a straightforward Kalman Filter on the position and velocity or an ad hoc two-stage weighting algorithm: first orientation weighting between the IMU and both feet, then position and velocity measurements weighting between both feet. Similar decoupled approaches have been implemented on quadruped robots such as a Kalman Filter for [Ble+18a] and a two-stage complementary filter for [Léz+21].

The *tightly-coupled* approach to proprioceptive base estimation was first introduced in works like [CHG11], but [Blo+13b] was the most decisive step. In this work, states of interest (position, velocity, orientation, IMU biases) are jointly estimated in an error state Kalman Filter (ErKF). Strapdown integration of an IMU and a direct kinematic model were used. In particular, translation-orientation coupling was showed to play a central role in making the IMU biases observable. An observability analysis and experiments showed that the orientation degrees of freedom need to be slightly excited to decipher accelerometer bias from the projection of the gravity vector. The same coupled approach was applied to humanoid robots [Rot+14; Fal+14]. [Har+20; Lin+21] take another step toward coupling the state variables by expressing the aforementioned variables as a single matrix Lie group. This new kind of estimator called Invariant Kalman Filter [BB18] exhibits a larger pool of convergence compared to the EKF and does not become inconsistent because of linearization issues.

All in all, one of the major factors affecting the choice in favor of either a tightly or loosely-coupled estimator is the quality of the platform sensors. Other factors may include personal experience of the designer, software/hardware architecture, need for modularity, etc. A very high-end IMU such as the Atlas fiber optics IMU will be able to compute a very consistent orientation and even biases inside its internal filter using an approxi-

mate motion model. On the other end, a lower quality IMU may require data fusion from sources such as leg-odometry, provided that this source of information is not too biased. Progress in miniaturization and production of MEMS-based IMUs, which equip most legged robots nowadays, seem to guide to the use of staged approaches for quadruped robots [Ble+18a; Léz+21]. However, these IMU classes still represent a significant portion of the robot's price. Furthermore, we believe that, in the long-range, tightly-coupled approaches present several advantages. Firstly, as legged robot gains in industrial maturity, tight coupling carries the promise of augmented performances by being able to more easily integrate online debiasing and calibration of lesser cost sensors. Secondly, as a software tool, we will show in this thesis that libraries based around tightly-coupled estimators may enable more versatility in the observer design. However, as we will see in Section 2.5 tightly-coupled estimators implementations based on Bayes filters do not scale up well to use exteroceptive sensors, which motivate us to use in our work of a new class of estimators, based on factor graph optimization.

2.2.3 Contact detection

As we saw, a critical part of legged robot estimation is the integration of kinematic information when feet are in contact. A major assumption of those methods is that the stable contacts are known a priori. The definition of a stable contact depends on the system: for quadrupeds that are usually equipped with point-feet, three positional degrees of freedom are blocked, the leg only being able to rotate around the contact point; while for a humanoid robot with planar feet, the six degrees of freedom are constrained. Slipping appears when the contact forces go outside of the Coulomb friction cone, that is when the ratio $f_{\parallel}/f_{\perp}$ (where $f_{\parallel}$ and $f_{\perp}$ correspond to the tangential and normal components of the contact force) exceeds a certain threshold, called friction coefficient. This coefficient is generally unknown as it depends on the nature of the foot and ground surface properties.

The most generally available information is the planned sequence of contacts [Ble+18b], which may be assumed to be approximately respected in nominal operation. While relying only on a prior plan lacks robustness due to unexpected events such as changes in terrain heights and feet slips, it is straightforward to implement and does not require extra sensors. Some robustness can be gained by reducing the confidence in the contact soon after impact or just before takeoff [Léz+21; Ble+18b]. Most high-end humanoid robots however [Sta+17; Eng+14] are equipped with strain gauges at their feet that can fairly accurately measure the ground reaction forces (GRFs). Though often biased depending on temperature, force measurements can be used as a proxy to stable contact by setting a reasonably large threshold [Fal+14]. This method is an approximation to checking if the contact force lies in Coulomb cone of the foot. However, for quadruped slip detection, Focchi [Foc+15] argues that force sensing is not enough, because the friction coefficient, as well as the contact normal, is generally hard to infer. The authors propose a simple algorithm checking relative feet velocities values in the base frame and discarding those far from the median. A similar choice is made by [Blo+13a] in which Unscented Kalman Filter velocity updates are rejected as outliers if their innovation exceeds a threshold. More recent papers try to fuse these different sources of information in Bayes filters. Hwangbo [Hwa+16] and subsequently Jenelten [Jen+19] fuse the kinodynamic models of the robot, IMU measurements, joint encoders values (and their first and second-order differentiation) as well as joint torques in a discrete state Hidden Markov Model (HMM) where contact states are binary values. A HMM is used instead of a continuous Bayes filter

because the feet state are modeled as taking a finite set of values: either in contact or not. They build separate filters for contact and slip detections and demonstrate walking on ice with an ANYmal-C using a special controller. A similar approach formulated as a Kalman Filter directly estimates contact probabilities [Ble+18b]. It questions previous works that assume the availability [Hwa+16] or negligibility [Cam+17] of joint angular accelerations in the context of whole-body dynamics end-effector force estimation. It proposes to instead rely on a new formulation of the *generalized momentum* estimator to estimate forces based on first-order derivative only by exploiting the structure of the whole-body dynamics Coriolis matrix. The gait cycle is taken into account as a process model and is fused with kinematics and contact forces updates.

This problem is not trivial and practical solutions seem to hesitate between simple heuristics and complicated Bayes filters. The problem with simple heuristics is that they neglect the temporal aspect of the data stream. On top of that, their parameters (though few in numbers) may be hard to tune. Bayes filters, on the other hand, provide more fine-tuned control of the modeled aspects of the problem, at the expense of complex parameter tuning. One thread of research tries to alleviate the need for tuning by relying on data-driven approaches. Supervised learning can be used [Cam+17] to introduce a probabilistic contact detector using only joint torques and robot kinematics, expressed as a logistic regression. The detectors provide a probability distribution on feet contacts that is used to ponder kinematic measurements of a tightly-coupled proprioceptive filter. The method outperforms a baseline based on threshold selection. However, different thresholds are fitted for each type of gait which may hardly generalize to different terrains and robot loads. Rotella [RSR18] proposes to cluster fuzzy contact states by training an unsupervised model on humanoid robot simulated data. The datasets are augmented with IMU measurements at the feet that are removed at evaluation time. The resulting estimator odometry system is shown to outperform a contact force classifier but was not validated on real hardware. More recently, Lin [Lin+21] proposed to train a deep neural network to infer contact states using a buffer of 150 ms of raw IMU, encoder, and kinematic measurements. The emphasis is put on training the estimator in various ground types in outdoors extended environment using an MIT Mini Cheetah. Ground truth is generated by finding a region around the local minimum of low passed feet height trajectories in the hip frames. The estimator provides a reliable source of contact information and generalizes better to different environments. It does not provide covariances about the contacts, contrary to [Cam+17].

Contact estimation is a still burgeoning field of research, especially for small quadrupeds which often do not embed contact sensors. The main reason is the fact that adding extra weight at the very end of the legs can increase significantly the leg's inertia, limiting the explosivity of possible movements. The extra electronics and wiring required may also be a limiting factor. As part of our work is to include leg-odometry in an factor graph estimator, for which contact detection is important, we scrutinized closely the state of the art on the topic. However, the tests we conducted were on flat ground for which the planned contacts are very similar to the actual contacts, so that slightly reducing the size of the planned contact phase was a sufficient heuristic for us.

2.2.4 Mitigation of kinematic model inaccuracies

We have seen in Section 2.2.2 that most legged robot proprioceptive filters rely on the robot kinematics in order to bound the drift in the integration of the inertial odometry and

debias IMU raw measurements. However, to obtain consistent filters, special care has to be taken to ensure that the kinematic measurements are not themselves biased. Mainly two main issues have been identified and addressed so far in the literature: inaccuracies in the joint velocities derived from encoder measurements and the undesired flexibilities of robot segments and joints.

Encoders measuring joint angles are usually placed at the actuator level [Xin+14]. They produce a very precise angle estimation when subsequent reduction steps are present. It is then acceptable to numerically differentiate and slightly filter those values to use them for joint impedance control for instance. However, hydraulic actuators do not include such reduction steps and fall victim to important joint velocity noise, which can degrade feedback control. Xinjilefu [Xin+14] proposes to use the dynamic model of the robot and joint torques to obtain filters on the joint angles and velocities of the Atlas robot. The same author [XFA16] also implemented a network of low-cost gyroscopes to estimate the joint velocities on the same platform. A Kalman Filter is used to fuse desired joint accelerations from the control as process input and angular velocity measurements coming from the MEMS with a numerical differentiation of the encoders. It requires a calibration procedure of the gyroscopes orientations and a good quality attitude estimation of the base, from a high-grade IMU for instance. This method was extended in [Rot+] by also including accelerometers measurements, explicitly compensating IMU biases, and alleviating the need for a global attitude estimation.

Another source of kinematic error comes from the presence of flexibilities in the structure of the robot, which is a common problem in many human-sized legged robots. For the HRP-2 humanoid robot (see Fig. 2.1), a rubber joint is placed at the ankle to mechanically absorb feet impacts. It also acts as a rotational spring that, given the length of the ankle-base lever, leads to important and unmeasured base accelerations. HRP-2 being equipped with 6-axis force sensors at the end-effectors, Flayols [Fla+17] proposes to map these measurements to relative orientations that can directly be included in the kinematic chain as an intermediate ball joint. Calibration of the rotational stiffness matrix is done by comparing kinematics to motion capture measurements. Similarly, Villa [Vil+22] proposes to account for the flexibility of the hip joint of Talos robot [Sta+17] (see Fig. 2.1) by modeling it as a passive joint with a damped-spring model. Benallegue [BL15] derives a procedure to alleviate the need for force-torque sensors by designing a centroidal filter using an inverted pendulum dynamical model. In Wandercraft' exoskeleton Atalante [Har+18a], flexibilities are shown to be spread along the successive segments of the kinematic chain, and can be modeled as punctual, 3D rotations with a linear spring behavior [Vig+18]. This work implements an IMU network similar to the ideas of [XFA16; Rot+] but it uses independent complementary filters for each IMU to recover their tilt (pitch and roll). Each filter uses the "zero-on-average" assumption, considering linear accelerations of the IMU frame negligeable on average before gravity and filtering them as noise on the accelerometer measurements. These orientations are then used as virtual joints and fused with the robot whole-body dynamics and the linear spring models to recover segments relative orientation and angular velocities. In a follow up work [Vig+22], the same authors revisit the linear acceleration negligibility assumption and drop the use of the robot dynamical model. They design an observer jointly estimating linear velocity and tilt of each IMU by fusing IMU data and linear velocities from the kinematics, which are propagated forward starting from the contacts by applying the filter recursively. It is shown that the new model converges more quickly than the zero-on-average estimators and has a reduced phase shift. The approach is validated with a controller which exhibits a better

behavior than the rigid one (that trips and falls) when a dummy of human size and weight (80 kg) is put in the exoskeleton, exciting the flexibilities.

Other kinematic uncertainties are however often present and are harder to model or sense. For instance, the foot of the quadruped used in [Blo+13b] is spherical but is only modeled as a point. Backlash prevented Fallon [Fal+14] to use a direct kinematic measurement model and was one of the limiting factors of Vigne first flexibilities estimator [Vig+18]. These works advocate for the addition of many different sensor sources in the estimator, in particular to account for the limited observability capabilities of legged systems and to benefit from cheap, distributed, low consumption sensors. Ideally, we could see the evolution of legged robot estimation going toward whole-body perception. In our opinion, the main limiting factor to that goal is the mathematical formulation of efficient tightly-coupled estimators, to which this thesis would like to contribute.

2.3 Centroidal dynamics estimation

2.3.1 Why is centroidal dynamic estimation important?

Legged robots are highly nonlinear systems that use contact forces between their end-effector and the environment to implement stable locomotion. The effect of these forces is described by the Newton/Euler equations, also known as the underactuated dynamics, which state that the variation of the system momentum is equal to the external wrench applied to the robot. Though these equations may appear simple, they introduce a non-linear coupling between the CoM position and the angular momentum. Many reduced-order models are based directly on various levels of approximation of these equations [Kaj+01; Wie06b; Car+16b] to derive predictive control algorithm, for locomotion for instance. These works assume knowledge of centroidal quantities of the system at control frequency, namely the position of the center of mass, angular momentum, and their derivatives. It is therefore crucial to implement accurate and efficient estimators for these quantities.

2.3.2 Information sources

Assuming that the base state and joint configurations are perfectly known (up to linear and angular accelerations), it is theoretically possible to compute all centroidal quantities using the kinematic model of the robot and the mass distribution in the different segments. However, this model is most often obtained from CAD data that may be inaccurate. This may call for a calibration of the platform [AVN08; AVN14; Bon+18; Bon+19]. This problem was closely scrutinized in the biomechanics literature where mass distributions often come from standardized anthropomorphic tables [De 96]. A second kind of information comes from measurements of the external wrench. Forces provide the CoM accelerations while moments relate to the CoM position through a line called the central axis. This line passes only approximately by the CoM because of gesticulation [3] induced angular momentum variations [Wie00; Wie06a]. Moreover, the wrench measurements are usually noisy and require the presence of expensive deformation gauge sensors at each contact

[3] We call gesticulation the part of the angular momentum due to the movement of the limbs. The total angular momentum is the sum of the centroidal angular momentum and the gesticulation. If no torque is applied to it, a polyarticulated system angular momentum is conserved but its angular velocity may be non-zero due to compensate for the gesticulation. Walking in space makes you turn!

point. A complete analysis of the particularities of these different information sources along with an observability analysis can be found in [Car+16a]. This fosters the use of more complex algorithms by fusing kinematic and external wrench measurements.

2.3.3 Reduced dynamical models

The contact wrench is not always directly accessible to the estimator. For instance, the Atlas robot gauge sensors measure only the contact normal force and horizontal moments. Thus, the exact underactuated dynamics cannot be directly used. Some works propose to use simplified models of the dynamics that require less information. For instance, the popular Linear Inverse Pendulum Model (LIPM) [Kaj+01] treats the body as a lumped mass, centered at the center of mass (COM), that only moves horizontally and therefore neglects the robot angular momentum. It only requires the position of the Center of Pressure (CoP) to obtain the CoM dynamics. The CoP is defined as the point where the moment component of the resulting wrench is aligned with the normal axis of the plane. The computation of this point only requires the contact normal force and horizontal moments, which is coincidingly the kind of sensor present on the Atlas robot.

Stephens [Ste11] proposes to use the LIPM for the process model to an EKF that also includes kinematic measurements. Model errors in the form of a CoM position measurement offset and external forces are added to the formulation and it is shown that either of these quantities is observable. Xinjilefu [XA12] ponders the use of a more complete planar sagital dynamical model by comparing it to the LIPM model without introducing model biases as was done in Stephens [Ste11]. It instead relies on tuning filter covariances to alleviate their effects. Simulations shows that the LIPM seems more able to cope with model errors but performances on the real robot show similar results between the two estimators. One of the benefits the LIPM model is that force measurements can be summarized by the Center of Pressure (CoP) (aka. Zero Moment Point (ZMP) [SB04]). In the context of the DARPA robotic challenge, the CMU team implemented a centroidal estimator [XFA15] similar to that of Stephen [Ste11] that was able to prevent a fall during the challenge finals. Similar EKF based estimators using respectively a LIPM [PT16] or a fly-wheel process model [PKT18], which models a non-zero angular momentum contrary to the LIPM, is later implemented on a NAO robot. A LIPM process can also be used with a fixed contact point assumption [BL15]. By a tight fusion with IMU measurements and kinematic information, centroidal quantities can jointly be estimated with joint flexibilities at the ankles under external force disturbances. The particularity of this approach is to avoid the need for force sensors but it is restricted to motions with fixed rigid contacts with the ground such as during manipulation. Xinjilefu [XFA14] took a different approach by estimating unmeasured components of the feet contact wrench (only normal force and pitch+roll torque are measured). This was done by leveraging the whole-body dynamics of the humanoid robot, IMU, and kinematic measurements. A generalized torque slack variable was introduced to represent the dynamical model errors. The estimator is implemented using Quadratic Programming which allows to enforce inequality constraints on state variables. Constraints are used to model joint limits and the positivity of normal ground reaction forces (the robot can only push on the ground).

2.3.4 Estimators based on underactuaded dynamics

While previously mentioned works rely on approximate dynamical models, we must acknowledge that an inaccurate process model must lead to a biased estimation. Rotella [Rot+15] proposes instead to use the underactuated dynamics, which is possible since the Sarcos robot used in their experiments is equipped with 6-axis force-torque sensors at its feet. The authors build several estimators by fusing force-torque and kinematic measurements in an EKF. These estimators introduce offsets on CoM position and linear momentum as well as an external 6D wrench disturbance. A nonlinear observability analysis is conducted and shows that either the biases or the external wrench are observable. Carpentier [Car+16a] chooses a different viewpoint by proposing a frequency analysis of the information sources used for centroidal estimation. The model assumes the presence of wrench sensors at the contacts and does not requires explicit modelling of the kinematic bias and external disturbances. Instead, it relies on a complementary filter to filter out problematic frequency bands in each signal. For instance, kinematic measurements are high passed filtered to remove its slowly varying bias. Bailly [Bai+19] extends this methodology to include CoM acceleration and angular momentum derivative in the estimation variables. Both works outperform a simple EKF by offering an unbiased CoM position estimate. The same author [BCS21] propose to use Differential Dynamic Programming (DDP), an algorithm traditionally used as a Optimal Control Problem solver [Mas+20b]. This algorithm estimates the same quantities as the previous work of the author [Bai+19] by solving a Maximum a Posteriori (MAP) problem on a sliding window of states sampled at IMU frequency, which permits to back-propagate information from the future to the past. The approach compares favorably to both previous work [Bai+19] and a simple EKF.

2.4 Environment awareness

Any useful task undertaken by an autonomous robot involves some level of awareness of its environment, even for partially teleoperated robots [Koo+16]. For ground robots, this problem is most often tackled with exteroceptive sensors such as cameras, depth cameras, and LIDARs. The environment might be known through a previous mapping procedure like Structure-from-Motion (SfM) [Tri+99] or mapped on the fly. Applications can be such as localizing with respect to a known map [Del+99], following a previously traversed path without a metric map [FB10], Simultaneous Localization And Mapping [Aul+08; Cad+16], object detection and pose retrieval [Du+21], autonomous exploration [Rou+19; Kul+21]... An intermediate case is the one of visual/LIDAR odometry in which a local representation of the environment is built to provide a precise odometry source [SF11] and is later discarded.

The particularity of legged platforms is that they use intermittent contacts to move, usually by using predefined cyclic gaits. While some controllers are robust enough so that a purely proprioceptive estimation provides enough feedback even in complex environments [Tan+18; Lee+20], having an estimate of the terrain shape may help planning steps. Moreover, multicontact approaches [Car+17; Hen+17] in which arms of a humanoid may also be used for locomotion are a promising way to increase the range of possible movements and reduce energy consumption.

The literature related to these questions is immensely vast and goes largely beyond the scope of this thesis, although the formal underlying methods are closely related, as

we will see in Section 2.5. Moreover, we will see that our ambition and latest results tend to merge legged state estimation with exteroceptive localization and mapping at large. We will quickly review the most relevant works, first focusing on localization *e.g.* for navigation, then on dense mapping for contact planning.

2.4.1 Localization and mapping

IMU/kinematics fusion inherently drifts in position and yaw orientation. For teleoperated tasks, this drift may not be problematic as the balance control loop mainly requires instantaneous base velocity and orientation with regard to the gravity while navigation can be handled by the operator. However, for autonomous navigation, it becomes critical to have some kind of precise odometry or localization strategy, or, even better, being able to run onboard Simultaneous Localization And Mapping. Davison [Dav+07] proposed the first example of a monocular vision-based SLAM system implemented for the navigation of a humanoid robot HRP-2. The gyro of the robot is also incorporated in the filter at the camera rate to minimize the growth of uncertainty before loop closing. [Sta+06] expends on this concept by also fusing kinematic velocity and altitude (known from the planning). Other systems based on sparse features were later developed [Ahn+12; Ori+12; Ori+16; Kwa+09]. Visual Teach and Repeat [FB10] has also been recently brought to quadruped robots [Mat+21; MCF22], which opens up practical long-term operations in industrial environments. Some works also propose to use mature visual odometry libraries as black-boxes sources of odometry by integrating them as relative pose measurements [Har+18c; Har+18b]. While it may lead to sub-optimal estimators, this method has the benefit of greatly simplifying the development process.

While a camera system benefits from low-cost and hardware integration difficulty, outdoors environments may cause some difficulties to such systems, such as shadows being mistaken for solid edges (Figure 8. of [Fal+14]). During the 2015 DARPA robotic challenge, where robots had to semi-autonomously traverse challenging environments while performing manipulation tasks at known given checkpoints, LIDAR information was crucial for teleoperated manipulation tasks [Koo+16] which required a precise metric representation of the scene. Localization with respect to a fixed map was also proposed [Fal+14] but not used in the finals [Fal16]. In fact, though supposed to represent a closed static industrial environment, the final trials happened in a semi-opened outside place with an important moving crowd on one side, making LIDAR localization very noisy.

Hornung [Hor+14] proposes to apply Monte Carlo Localization using depth measurements (LIDAR) thanks to a highly efficient data structure called Octomap [Hor+13]. The algorithm is validated on a NAO humanoid robot and used a learned leg-odometry motion model, pitch and roll information from the IMU as well as an edge-based vision measurement model. The MIT team [Fal+14] uses the same method but replaces the motion model with an inertial kinematics proprioceptive filter. Other occupancy mapping algorithms such as Bayesian generalized kernel OctoMap (BGKOctoMap) [DWE17] try to infer unobserved parts of the of map by considering correlations between grid cells. This inference model has the interesting property to smoothly transition to sanely defined prior (like a uniform probability distribution) while keeping similar computation time as the Octomap. BGKOctoMap is generalized to jointly infer the probabilistic occupancy grid and semantic information using stereo camera and LIDAR information [Gan+20]. Potential for legged robot navigation was demonstrated with a dataset taken outdoor on a Cassie robot. Semantic information prior come from a Convolutional neural network fine-tuned

to manually labeled pictures ($\approx$ 1000) of similar scenes.

LIDAR can also be used as an odometry source by matching point clouds taken at successive timestamps with ICP. However, this procedure is time-consuming and leads to time delays with other measurement sources, which is generally a problem for Bayes filters. Nobili [Nob+17] solves this issue by keeping a buffer of past belief state of the EKF, a method on which is based the Pronto framework [Cam+20].

2.4.2 Reconstruction for footstep planning

Even though some legged robots designs and controllers [RMA19; Ble+18a] are robust, to some extent, to terrain model uncertainty, having information about the local shape of the environment enables a more reactive contact planning. For instance, the guide path implemented in the contact planner [Ton+18] uses the complete 3D structure of the environment provided as an STL file while a mixed integer contact planner like [Ton+20] uses a list of vertex tuples representing convex planar surfaces. Most reconstruction implementations rely on a depth sensor to obtain these representations.

An interesting representation is to build a grid-based *elevation maps*, that associates a probabilistic distribution to the height of each 2D cell. This is a simplified representation of the 3D environment, that does not take terrain slopes into account for instance, but provides advantageous computational features. For rover applications, one can argue that building a globally consistent elevation map is too costly and it is instead more efficient to switch the representation to the local robot frame [KD07]. The map is constantly updated using two sources of information: range measurements that refine the visible parts of the map; wheel odometry that increases uncertainty on the map using heuristics based on the traveled distance. Fankhauser [Fan+14; FBH18] adapted these ideas to legged robots. The map grid probabilistic representation was also improved by including horizontal uncertainty due to the robot's relative motion. A costly map fusion process that computes lower and upper bound on the elevations is decoupled from the sensor updates and can be computed intermittently at the discretion of the user. This system was recently integrated in a reactive planning system [HG21] able to navigate outdoor uneven terrain with a biped using LIDAR data. Taking another route, Kim [Kim+20] proposed a much simpler algorithm, relying on integrated RealSense sensors, in order to alleviate on-board computations. A T265 integrated SLAM algorithm provides localization and a D435 provides noisy depth information. A 2.5D world-centered height map is produced from scratch at each point cloud acquisition using opening morphological transformation filtering. Classification of the terrain, based on local gradient thresholds, is used to implement dynamical manoeuvers in a cluttered environment with an MIT Mini-Cheetah.

An extension of the elevation grid map is to represent occupied 3D space as an 3D occupancy voxel map, which can be efficiently stored using an Octomap. While [Hor+14; Fal+14] only used this representation as prior for online localization, it was used for online foot planning in [Win+15; Mas+15].

Kolter [KKN09] adopts a mesh representation built from point clouds aligned twice a second off-board using an ICP procedure initialized by proprioceptive odometry and filled with a texture synthesis step. This representation directly provides richer information such as the slope of the terrain which is used in [Mas+20a] to tightly couple motion/step planning and terrain reconstruction.

Fallon [Fal+15] used the Kintinuous framework [Whe+12] to obtain a local Truncated Signed Distance Function based dense representation. This map was shown to be of

equivalent quality to that of a LIDAR and used as an input to a feasible contact surface segmentation algorithm selecting planar convex regions of uneven cinder blocks. In order to handle local loopy trajectories better than Kintinuous, Elastic Fusion [Whe+16], which is a dense depth-based vision system using surfels [4], was shown to be usable on humanoid platforms [Sco+17]. The system augmented the Elastic Fusion algorithm by adding proprioceptive odometry term to the solved nonlinear least-squares (NLLS) problem, using a heuristic to increase the weight of proprioception in visually degenerate situations. The dense reconstruction is accurate up to 2 cm, which is enough for motion planning. However, in this case, the map was only used for localization.

Online terrain reconstruction extends the capabilities of legged robots to uneven surfaces. It is however not central to the problem statement of this thesis and is considered as out of scope. Our main goal is to investigate a new class of estimators based on Factor Graph optimization, which we will now introduce.

2.5 Factor Graph optimization

As we have previously discussed, the largest part of the legged robotics state estimation literature consists in fusing sensor modalities using various implementations of the Bayesian and complementary filters. The core of those systems consists of a tightly or loosely-coupled IMU/kinematics proprioceptive filter. Then, either this proprioception is used as a prior odometry source for a SLAM system or the exteroceptive sensors are used to localize with respect to a known map, making position and yaw observable. A different kind of estimator has been investigated over the last decades in the visual/LIDAR SLAM community and has become predominant in applications such as drone navigation: *factor graph optimization* (FGO), also known as simply *graph optimization*.

2.5.1 Visual SLAM transition to graph optimization

While the Bayes filter summarizes the accumulated information about the current estimated state and its cross-correlations as a single probabilistic distribution, Factor Graph optimization keeps a trajectory of arbitrarily spaced past states that are regularly re-optimized in the least-squares sense, by finding the so-called Maximum a Posteriori over the trajectory joint distribution. The terms smoothing and mapping or sliding window estimator are also used when a limited window of past states is kept in the estimator, the old ones being marginalized. The earliest example [LM97] proposed to obtain globally consistent 2D LIDAR range scan alignments by minimizing a cost function over a trajectory of 2D poses constrained by LIDAR scans and wheel odometry. Many ad hoc efficient solvers such as [Gri+11; Kae+12; Ila+17; AM+] have been developed over the years to leverage the specific sparsity of the SLAM problem.

While the first successful online monocular visual SLAM algorithm was implemented using an EKF [Dav+07], a breakthrough happened with the Parallel Tracking And Mapping (PTAM) system [KM09] that proposed to split the visual tracking of features and camera motion from the mapping in separate processes. This new modularity enabled the use of a specific algorithm for the mapping part: *Bundle Adjustment* (BA) [Tri+99; SF16] of sparse features, a method that was previously reserved for offline SfM pipelines. PTAM

[4] " A surfel is a zero-dimensional n-tuple with shape and shade attributes that locally approximate an object surface [Pfi+00]

enabled Augmented Reality applications in limited spaces. BA is actually a special case of a more general estimation algorithm called factor graph optimimization which has several conceptual and computational advantages over filtering. A great wealth of papers following either Gaussian filtering (to which EKF belongs) or graph optimization were subsequently published. A thorough comparison of the two approaches [SMD12] concluded that BA was computationally superior in that it had the greatest precision to computational cost ratio, for most benchmarks. Most of the state-of-the-art visual SLAM framework now follow a factor graph approach [For+17; MMT15; QLS18; Leu+15; Fer+21].

In the context of real-time visual SLAM, the prime conclusion of Strasdat experiments [SMD12] is that to increase accuracy, it is more profitable to increase the number of features than the number of frames. In this regard, the superiority of the graph optimization approach is based on a cost argument: the complexity of BA scales linearly with the number of features while filtering exhibits cubic scaling. A second important argument is that Gaussian filters linearize the nonlinear measurement models only once, around the current estimate. On the contrary, for FGO re-linearization is done at each solver step, the resulting Jacobians getting better as the estimate converges to the optimal solution. BA is therefore more accurate than its filtering counterpart, especially for nonlinear motion (due to the state rotation matrix). The greater cost of Gaussian filtering is due to the explicit computation of the state covariance function, which becomes rapidly dense, while FGO keeps the variables' correlations in a sparse information matrix. An advantage of the Gaussian form is direct access to the covariance matrix. This was leveraged in early SLAM systems such as the work of Davison [Dav+07] to perform active tracking: search for image feature correspondences in a limited uncertainty-aware region. However, modern visual feature front-ends do not tend to use active tracking anymore, relying on robust matching (RANSAC) [MMT15] or KLT tracking [BM04; Fer+21]. Another advantage of FGO optimization for visual systems is that the front-end operations (in particular camera pose tracking and feature tracking) can be decoupled from the back-end, which implements BA optimization at a lower rate (and usually in a separate thread). This results in a more modular software architecture which can be leveraged to include other "front-ends" treating raw measurements from other sensor modalities.

A very popular extension to these works is to inject IMU measurements into the SLAM system.

2.5.2 Visual-Inertial odometry

To robustify visual SLAM/odometry systems in cases of adverse situations, IMUs provide complementary high rate odometry information that can be debiased by tightly coupling mapping and odometry. Another important feature of visual-inertial estimation is that the orientation absolute pitch and roll can be observed thanks to the gravity (absolute yaw is unobservable, unless a compass is used). Estimation of the pose and velocity of the system can be estimated at the IMU frequency (which is usually an order of magnitude higher than the camera's).

A careful pre-integration [LS09; For+17] of these high rate measurements had to be developed in the context of smoothing for the optimization to remain tractable. The core idea of this algorithm has since been applied to other information sources such as drone thrust commands [Nis+19] to make external force estimation possible; while we propose in this thesis to preintegrate force-torque measurements present in legged robots. A detailed description of the theory of pre-integration and discussion of related works is found

in Chapter 6.

2.5.3 Factor Graph estimation for legged robots

Very recently, two teams started to apply factor graph optimization principles as a general framework for legged robot estimation. At Michigan University, Hartley [Har+18c] proposes to fuse IMU pre-integration with a novel kinematic factor by adding the contact foot pose in the optimized states, similarly to [Blo+13b; Rot+14]. Semi-direct Visual Odometry [FPS14] is also used to integrate camera measurements as relative pose. The estimation backend was based on GTSAM framework [Del12] and tests were conducted on a Cassie bipedal robot. In a later work [Har+18b], the same authors extended the kinematic factor formulation, taking into account uncertainty induced by contact switches. At the Dynamic Robot Systems Group at Oxford, Wisth proposed in [WCF19] to replace a modeled kinematic factor by integrating odometry from the onboard proprioceptive filter of the ANYmal robot. Stereo vision was also used as a source of odometry by using a sparse 3D landmark map. In a following work [WCF20], the proprioceptive odometry factor was changed to include a slowly variable bias on odometry measurements by adapting the pre-integration theory from [For+17]. The same authors extended this work by including LIDAR measurements in [WCF21]. [Kim+21] also implemented an inertial kinematic proprioceptive smoothing estimator on the MIT Mini Cheetah (see Fig. 2.1), mitigating slipping of the feet by removing kinematic measurements whose residual errors exceed a certain threshold.

The maturity of the aforementioned factor graph optimization solver libraries certainly played an important role in these recent developments. It seems that tools originally developed in the SLAM community are making their way in the legged robot community. Software libraries are becoming more generalist, more precise and, based on Lie theory [SDA18], they handle more nicely the specificities of the manifold structure of the problem variables. Other projects propose to deal with the inherent complexity of multi-sensory systems, proposing principled ways to handle multiple data sources, potentially asynchronous and at different frequencies. Those frameworks also provide a mathematical formulation for most common sensor models and higher level interfaces with NNLS solvers in an effort to bring these techniques to a broader audience [Sol+21; Bla19; Col+20].

From here, the reader is invited to revisit the main scientific statement of the thesis in Section 1.4, as we hope the presentation of the state of the art cast a new light on this section. The remaining of the thesis will then first go through the theoretical formulation (starting with a tutorial on MAP), before reporting the application of the theory to 3 estimators for legged robotics

Part I

Theory

Chapter 3

Tutorial on factor graph state estimation

Contents

The *state* of the robot is a reduced set of variables of particular interest to the roboticist, be it for control, parameter identification, etc. For example, on a quadruped robot, the state may typically be composed of the base position, velocity, and orientation, the center of mass (CoM) position and velocity, joint angles, etc. Those quantities may not directly be measurable, due to their physical nature (the center of mass is a virtual point) or because sensor data is too noisy, biased, or impractical to obtain (*e.g.* GPS for localization is bad close to flat surfaces because of beam reflections). Those latent variables can however be estimated by fusing multiple sensors data using a state estimator (aka. observer in automation). The task of *estimation* can then simply be stated as finding the robot state given these measurements. This intuition can be formalized as finding the mode of the posterior distribution on the states conditioned by the measurements.

Probabilistic theory applied to signal processing and information theory has been the bedrock of the state estimation theory development. For probabilistic estimators, states are random variables, that, in the robotics context, are mostly continuous. The goal is to find the best estimate using sensor measurements. In the Bayesian perspective, this means to find a distribution over a collection of random variables $\mathcal{X}$ given a set of measurements $\mathcal{Z}$, $p(\mathcal{X}|\mathcal{Z})$, which is known as the *posterior* distribution. The Bayes law represents this inference:

$$p(\mathcal{X}|\mathcal{Z}) = \frac{p(\mathcal{Z}|\mathcal{X})p(\mathcal{X})}{p(\mathcal{Z})}. \tag{3.1}$$

$p(\mathcal{Z}|\mathcal{X})$ is the measurement model, also called the *likelihood* of the observation. $p(\mathcal{X})$ is a *prior* that we have on the state variable distribution. This may include for instance knowledge about the initial state of the robot or an approximate value of parameters that we seek to estimate. The *marginalized likelihood* $p(\mathcal{Z}) = \int p(\mathcal{Z}|\mathcal{X})p(\mathcal{X})d\mathcal{X}$ can be thought of as a global normalization constant ([KF09, Chapter 20]) in case the $\mathcal{Z}$ random variable is observed, which is our case. In general, this term is computationally intractable to compute, since it requires marginalizing the likelihood distribution over all possible states. Exact inference is therefore rarely possible, the Bayesian practitioner instead relying on approximate inference.

For example, in the context of robotics, recursive Monte Carlo sampling ([KF09, Chapter 12]) has been leveraged in the popular recursive particle filter for tasks such as localization [Del+99] and SLAM [Mon+02]. A very interesting property of this approximation is the fact that multimodal distributions can be modeled, which can be useful for multi hypothesis problems such as the kidnapped robot problem [Del+99] or target tracking [Gus+02].

Another type of approximate inference, with scarcer applications in robotics until now, is variational inference ([KF09, Chapter 11]). The idea here is to fit the parameters of a candidate distribution so that the Kullback-Leibler divergence between the posterior and the candidate distribution is minimized. Gaussian distributions are then often used because closed forms of their divergence are easy to evaluate. Very recent works start to find applications in robotics as an alternative to MAP-based graph optimization [BFY20; Won+20] or to Bayes filtering [LBB22].

A more popular approach to the estimation problem is to concentrate on finding the mode of the posterior distribution, *a.k.a.* the *Maximum a Posteriori* (MAP).

3.1 Maximum a Posteriori estimation

An efficient way to characterize the posterior distribution is to first find its mode, that is the states that result in the highest posterior probability. The estimation problem is, in this case, an unconstrained optimization problem:

$$\mathcal{X}^{MAP} \triangleq \arg\max_{\mathcal{X}} p(\mathcal{X}|\mathcal{Z}) = \arg\max_{\mathcal{X}} p(\mathcal{Z}|\mathcal{X})p(\mathcal{X}). \tag{3.2}$$

Notice that the $p(\mathcal{Z})$ term does not appear anymore on the right side since it is constant relative to $\mathcal{X}$. States variables $\mathcal{X}$ in our case are a collection of n random variables $\{\mathcal{X}_i\}_{i\in[1..n]}$ that each relate to a physical quantity of interest (*e.g.* the initial robot position, constant camera parameters, orientation of an object in the scene, IMU biases, etc.). Measurements $\mathcal{Z}$ are similarly a collection of m individual sensor measurements $\{\mathbf{z}_i\}_{i\in[1..m]}$.

Additional assumptions have to be made to obtain a numerical implementation of this problem. First, the measurements are supposed to be conditionally independent of each other, so that the likelihood function can be factorized

$$p(\mathcal{Z}|\mathcal{X})p(\mathcal{X}) = p(\mathcal{X}_{S_0}) \prod_{i=1}^{n} p(\mathbf{z}_i|\mathcal{X}_{S_i}). \tag{3.3}$$

Each factor represents the measurement model associated to the observation $\mathbf{z}_i$ and depends only on a subset S_i of the state variables $\mathcal{X}_{S_i}$. S_0 denotes the subset of random variables with nonuniform priors. Secondly, the measurements are assumed to be corrupted by multivariate Gaussian noise:

$$p(\mathbf{z}_i|\mathcal{X}_{S_i}) = \frac{1}{\sqrt{2\pi\boldsymbol{\Sigma}_i}} \exp(-\frac{1}{2}(||\mathbf{e}_i(\mathcal{X}_{S_i})||^2_{\boldsymbol{\Sigma}_i}) \triangleq K_i\, \phi_i(\mathcal{X}_{S_i}) \tag{3.4}$$

where the *residuals* $\mathbf{e}_i(\mathcal{X}_{S_i}) \in \mathbb{R}^{M_i}$ are (potentially) nonlinear functions of the state variables, $K_i \in \mathbb{R}$ are constants, $\boldsymbol{\Sigma}_i \in \mathbb{R}^{M_i \times M_i}$ are the covariances of the measurements' noise, $\phi_i(\mathcal{X}_{S_i})$ are the un-normalized measurement likelihoods called *factors*, and

$$||\mathbf{e}_i(\mathcal{X}_{S_i})||_{\boldsymbol{\Sigma}_i} \triangleq \sqrt{\mathbf{e}_i(\mathcal{X}_{S_i})\boldsymbol{\Sigma}_i^{-1}\mathbf{e}_i(X_{S_i})}$$

is known as the squared Mahalanobis distance. Residuals $\mathbf{e}_i$ can generally be formulated as a difference between an *expectation function* $\mathbf{h}$ and the actual measurements

$$\mathbf{e}_i(\mathcal{X}_{S_i}) = \mathbf{h}(\mathcal{X}_{S_i}) - \mathbf{z}_i, \tag{3.5}$$

although some exceptions may exist (see for instance the IMU pre-integration residual (6.21) in Section 6.2.4).

Thus, the posterior probability is proportional to a product of individual factors:

$$p(\mathcal{X}|\mathcal{Z}) \propto \phi_0(\mathcal{X}_{S_0}) \prod_{i=1}^{n} \phi_i(\mathcal{X}_{S_i}) \tag{3.6}$$

Recognizing that maximizing the likelihood in (3.2) is equivalent to minimizing the negative log-likelihood, we can apply the assumptions successively to write:

$$\begin{aligned}
\mathcal{X}^{MAP} &= \arg\max_{\mathcal{X}} p(\mathcal{X}|\mathcal{Z}) && \text{MAP problem definition} && (3.7)\\
&= \arg\min_{\mathcal{X}} -\log p(\mathcal{X}|\mathcal{Z}) && \text{Negative log likelihood} && (3.8)\\
&= \arg\min_{\mathcal{X}} -\log p(\mathcal{Z}|\mathcal{X})p(\mathcal{X}) && \text{Unaffected by constant denominator} && (3.9)\\
&= \arg\min_{\mathcal{X}} -\log p(\mathcal{X}_0) \prod_{i=1}^{n} p(\mathbf{z}_i|\mathcal{X}_{S_i}) && \text{Conditional independences} && (3.10)\\
&= \arg\min_{\mathcal{X}} -\log \phi_0(\mathcal{X}_{S_0}) \prod_{i=1}^{n} \phi_i(X_{S_i}) && \text{Factorized likelihood} && (3.11)\\
&= \arg\min_{\mathcal{X}} \sum_{i=0}^{n} ||\mathbf{e}_i(\mathcal{X}_{S_i})||^2_{\boldsymbol{\Sigma}_i} && \text{Gaussian measurement models} && (3.12)
\end{aligned}$$

Thus, solving the MAP problem with the aforementioned hypotheses boils down to solving a nonlinear weighted least-squares (NLLS) problem. Notice that we included the

prior in the sum of the residuals, as it is mathematically equivalent to a measurement model. The weights are the inverse of the measurement covariances: the more uncertain a measurement is, the higher its covariance and, therefore, the lower its influence on the weighted squared residuals sum.

A vast part of the literature on MAP estimation has been dedicated to the implementation of efficient ad hoc NLLS solvers. Most of them are gradient-based algorithm, typically some variation of the Gauss-Newton algorithm, such as the Levenberg-Marquardt algorithm [BBV04]. We chose to use the Ceres solver [AM+] for its maturity and wide range of features and because it was already integrated with the team state estimation framework WOLF [Sol+21].

3.2 The Gauss-Newton algorithm and some variants

In this section, we will give a brief introduction to a classical numerical algorithm used in robotics to solve an NNLS problem efficiently: the Gauss-Newton algorithm. We will use this solver to address the MAP problem. For many practical robotics applications, some state variables (such as rotation matrices) live on manifolds. For now, we will assume that state variables and measurements all live in vector spaces to simplify the derivations and we will extend the method to manifold states in Section 3.3.2.

We purposely keep this tutorial to a minimal version; the reader is referred to [DK+17; Sol17] for an extended version.

3.2.1 The algorithm

The Gauss-Newton algorithm is an iterative algorithm to find the minimum of NLLS cost functions that can be decomposed in a series of steps.

1. Initialize the state estimate at an initial value $\check{\mathcal{X}} := \mathcal{X}^0$
2. Approximate the NLLS cost function around the current estimate as a quadratic function (see (3.17) below).
3. Find the optimal step $\Delta\mathbf{x}^*$ as the root of $\mathcal{F}(\Delta\mathbf{x})$, corresponding to a linear set of equations (see (3.18) below)
4. Update the current state estimate $\check{\mathcal{X}} := \check{\mathcal{X}} + \Delta\mathbf{x}^*$
5. Loop over steps 2-4 until convergence

3.2.2 Derivation of the Gauss-Newton step

We will now detail the critical parts of the algorithm, namely the linearization of the residuals and the computation of an optimal step. First, we will simplify notations by dropping dependencies on state variables X_{S_i} where evident. Secondly, we will make a change of variables. The weighted squared residuals can be expressed as:

$$||\mathbf{e}_i||^2_{\mathbf{\Sigma}_i} = \mathbf{e}_i \mathbf{\Sigma}_i^{-1} \mathbf{e}_i = (\mathbf{\Sigma}_i^{-\frac{1}{2}} \mathbf{e}_i)^\top \mathbf{\Sigma}_i^{-\frac{1}{2}} \mathbf{e}_i = ||\mathbf{\Sigma}_i^{-\frac{1}{2}} \mathbf{e}_i||^2 = ||\mathbf{r}_i||^2 \tag{3.13}$$

where $\mathbf{r}_i(X_{S_i}) \triangleq \mathbf{\Sigma}_i^{-\frac{1}{2}}\mathbf{e}_i(X_{S_i})$ can be interpreted as a whitened residual. $\mathbf{\Sigma}_i^{-\frac{1}{2}}$ can be obtained from the Cholesky factorization of $\mathbf{\Sigma}_i^{-1}$. Therefore, the NLLS MAP problem can simply be written:

$$\mathcal{X}^{MAP} = \arg\min_{\mathcal{X}} ||\mathbf{r}(\mathcal{X})||^2 \tag{3.14}$$

where $\mathbf{r}$ is a vector of vertically stacked residuals (column vectors) and the cost function is the squared norm of the residual vector. Let us assume we have a current estimate $\check{\mathcal{X}}$ of the state variables $\mathcal{X}$. Each residual can be linearized with respect to the states it depends on $\mathcal{X}_{S_i}$ at their current estimate $\check{\mathcal{X}}_{S_i}$:

$$\mathbf{r}_i(\mathcal{X}_{S_i}) = \mathbf{r}_i(\check{\mathcal{X}}_{S_i} + \Delta\mathbf{x}_i) \approx \check{\mathbf{r}}_i + \mathbf{J}_i\Delta\mathbf{x}_i \tag{3.15}$$

where J_i is the Jacobian of the residual at $\check{\mathcal{X}}_{S_i}$:

$$\mathbf{J}_i \triangleq \left.\frac{\partial\mathbf{r}_i}{\partial\mathcal{X}_{S_i}}\right|_{\check{\mathcal{X}}_{S_i}} \tag{3.16}$$

We can note $N = \sum_{i=1}^{n} dim(\mathbf{x}_i)$ and $M = \sum_{i=1}^{m} dim(\mathbf{e}_i)$. We stack up the $\Delta\mathbf{x}_i$ column vectors as $\Delta\mathbf{x} \in \mathbb{R}^N$ and the residual Jacobians at the linearization point $\mathbf{J} \in \mathbb{R}^{M\times N}$. We call this new function $\mathcal{F}(\Delta\mathbf{x}) \approx ||\mathbf{r}(\mathcal{X} + \Delta\mathbf{x})||^2$, so that the linearized cost function writes:

$$\mathcal{F}(\Delta\mathbf{x}) = ||\check{\mathbf{r}} + \mathbf{J}\Delta\mathbf{x}||^2 = \check{\mathbf{r}}^\top\check{\mathbf{r}} + 2\check{\mathbf{r}}^\top\mathbf{J}\Delta\mathbf{x} + \Delta\mathbf{x}^\top\mathbf{J}^\top\mathbf{J}\Delta\mathbf{x}. \tag{3.17}$$

$\mathcal{F}(\Delta\mathbf{x})$ is therefore a local parabolic approximation of the squared residual around the current estimate and $\mathbf{H} \triangleq \mathbf{J}^\top\mathbf{J}$ is the approximate Hessian of the squared residual with respect to the state variables [1]. Note that by construction, $\mathbf{H}$ is always semi-definite positive.

The optimal step (Gauss-Newton step) $\Delta\mathbf{x}^*$ is by definition the step that minimizes $\mathcal{F}$. This minimum is found by differentiating $\mathcal{F}$ with respect to $\Delta\mathbf{x}$ and equaling to 0, giving the linear system of equation:

$$\mathbf{H}\Delta\mathbf{x}^* = -\mathbf{J}^\top\mathbf{r}. \tag{3.18}$$

Substituting $\mathbf{H}$ by its expression, the Gauss-Newton step is found by solving the linear system, that is to say, inverting $\mathbf{H}$:

$$\Delta\mathbf{x}^*_{GN} = -\mathbf{H}^{-1}\mathbf{J}^\top\mathbf{r} = -(\mathbf{J}^\top\mathbf{J})^{-1}\mathbf{J}^\top\mathbf{r} = -\mathbf{J}^+\mathbf{r}$$

where $\mathbf{J}^+$ is the (right) pseudo-inverse of the residual gradient.

Taking a Gauss-Newton step then refers to applying the optimal step to get a new estimate:

$$\check{\mathcal{X}} := \check{\mathcal{X}} + \Delta\mathbf{x}^*. \tag{3.19}$$

In practice, the individual Jacobians (3.16) are usually obtained by automatic differentiation, *e.g.* using dual number [AM+] and $\mathbf{J}^+$ is not constructed explicitly. Efficient linear system solvers based on the Cholesky factorization of $\mathbf{H}$ and the QR factorization of $\mathbf{J}$ are commonly used in SLAM since they can exploit the sparsity of these matrices.

[1] See [Sol17, Section 4.2.1] for more details on the nature of the approximation.

3.2.3 The Levenberg-Marquardt variants

The quality of the Gauss-Newton step depends on the validity of the quadratic approximation of its cost function. When the cost is truly quadratic, this approximation is quite good and leads to a near quadratic (super linear) convergence of the optimization procedure. Otherwise, for example in the typical pathological situation where the local shape of the cost function is flat, that is if the hessian has small eigenvalues, the resulting step can largely differ from the optimal descent direction or even lead to a divergent behavior. The Levenberg-Marquardt is an extension of the Gauss-Newton algorithm in which the linear system (3.18) is modified to alleviate this phenomenon.

Levenberg Levenberg [Lev44] contribution was to propose to dampen the Hessian by the identity matrix:

$$\Delta\mathbf{x}_L^* = -\alpha(\mathbf{H} + \lambda\mathbf{I})^{-1}\mathbf{J}^\top\mathbf{r} \tag{3.20}$$

where α and λ are scalar coefficients that can be tuned depending on the evolution of the cost function. In particular, λ controls the amount of damping: if a step computed with a given λ step is wrong (the cost function goes up), λ is increased so that the Hessian influence is regularized. For large values, the steps are close to a gradient descent step. α provides a way to tune the size of the descent steps.

Marquardt Marquardt [Mar63] improves on Levenberg by proposing to dampen by the Hessian diagonal $\text{diag}(\mathbf{H})$ instead of the identity matrix:

$$\Delta\mathbf{x}_L^* = -\alpha(\mathbf{H} + \lambda\text{diag}(\mathbf{H}))^{-1}\mathbf{J}^\top\mathbf{r}. \tag{3.21}$$

Thus, the damping affects each direction of the state differently, depending on the local shape of the cost function.

The values of α and λ are continuously adapted to accommodate for the local shape of the cost function. This can be understood as an implementation of the Trust Region paradigm [BBV04].

3.2.4 Other algorithms

The Levenberg-Marquardt algorithm is only one example of a vast family of algorithmic strategies to mitigate the shortcomings of the "pure" Gauss-Newton algorithm. Other examples include the Trust region algorithm Dogleg. For very large structure-from-motion problems, Conjugate Gradient Descent is also used as an alternative to Gauss-Newton derivates, though extra care has to be taken toward the conditioning of the Hessian [JBD12].

Those algorithms are trade-offs between accuracy, speed, memory budget, and implementation complexity, that suit better in different applications. It seems that, for robotics, Levenberg-Marquardt has proven to be a good trade-off between the real-time and accuracy requirements.

3.3 State estimation on manifolds

Some of the state variables that we manipulate in robotics are challenging since they do not belong to vector spaces.

These variables live on smooth manifolds, also known as Riemannian manifolds. Some of the equations involved in the Gauss-Newton algorithm cannot be directly applied without extra care when it is the case. Many examples of Riemannian geometry arise in data science and robotics [Mio+20]. In our case, taking care of the manifold structure is necessary and enough to derive optimization algorithms.

Besides, many of these robotics state variables also exhibit a group structure. Especially useful are the existence of an identity element (that can be thought of as the origin of the group), the existence of a unique inverse for each element, the possibility to interpolate between elements, and the consequent existence of the adjoint linear map.

If a variable has these two properties, they can be described as belonging to a so-called Lie group, whose properties we will review. We will introduce operations such as the exponential map that are necessary to describe the measurement models proposed in the following chapters.

This section is organized by first describing the geometrical properties of manifold elements and how these properties can be exploited to derive optimization algorithms on manifolds. Of particular interest is the special definition that are given to covariances and Jacobians on manifold elements. Finally, we will see how these properties can be brought together by the Lie theory, of which we give a very brief introduction, illustrating properties with the important example of rotation matrices.

3.3.1 Smooth manifold structure

A *smooth manifold* $\mathcal{M}$, or differentiable manifold, is a topological space that can be pictured as a smooth surface embedded in a higher dimensional vector space. The smoothness property (there are no spikes or edges on the manifold) means that to each manifold point $\mathbf{x}$ corresponds a unique tangent (hyper-)plane $\mathcal{T}_{\mathbf{x}}\mathcal{M}$, called the tangent-space. The tangent space is a vector space on which traditional calculus operations are applicable.

The dimension of this vector space is equal to the dimension of the manifold as well as the degrees of freedom of its elements. We will denote by p the dimension of the space in which the manifold is embedded and by n the dimension of the manifold. Note that by definition $n \leq p$.

An element of tangent space at $\mathbf{x}$, noted $\boldsymbol{\tau}^\wedge \in \mathcal{T}_{\mathbf{x}}\mathcal{M}$ can naturally be decomposed as a linear combination of the tangent space basis vectors E_i. We can therefore define an element by its coefficient, which is the vector $\boldsymbol{\tau} \in \mathbb{R}^n$, called the Cartesian tangent vector. The *hat* and *vee* are mutually inverse linear maps permit to pass back and forth from $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ to $\mathbb{R}^m$, which are therefore isomorphic:

$$\text{Hat} : \quad \mathbb{R}^n \to \mathcal{T}_{\mathbf{x}}\mathcal{M}; \qquad \boldsymbol{\tau} \to \boldsymbol{\tau}^\wedge = \sum_{i=1}^{n} \tau_i E_i \tag{3.22}$$

$$\text{Vee} : \quad \mathcal{T}_{\mathbf{x}}\mathcal{M} \to \mathbb{R}^n; \qquad \boldsymbol{\tau}^\wedge \to (\boldsymbol{\tau}^\wedge)^\vee = \boldsymbol{\tau} = \sum_{i=1}^{n} \tau_i \mathbf{e}_i \tag{3.23}$$

where $\mathbf{e}_i$ are basis vectors of $\mathbb{R}^n$.

Vector spaces operations such as the addition of a vector to a point or subtraction of two vectors do not apply in Riemannian geometry. For instance, if we take a point on a sphere (the 2-sphere embedded in the 3-dimensional Euclidean space for instance) and add to it an arbitrary vector, we do not get in general another point on the sphere. These operations have equivalent in the *retraction* and the *lift*, its inverse. Retraction pulls an

element from the local tangent space back to the manifold as represented in Fig. 3.1. We denote the retraction and lift operators $\oplus$ and $\ominus$:

$$\text{Retraction} : \quad \mathcal{M} \times \mathbb{R}^n \to \mathcal{M}; \qquad (\mathbf{x}_1, \boldsymbol{\tau}) \to \mathbf{x}_2 = \mathbf{x}_1 \oplus \boldsymbol{\tau} \tag{3.24}$$

$$\text{Lift} : \quad \mathcal{M} \times \mathcal{M} \to \mathbb{R}^n; \qquad (\mathbf{x}_1, \mathbf{x}_2) \to \boldsymbol{\tau} = \mathbf{x}_2 \ominus \mathbf{x}_1 \tag{3.25}$$

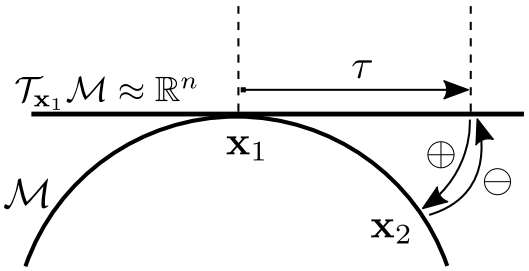


Figure 3.1: Manifold, tangent space, retraction and lift operations

As in [SDA18], we use the concept of composite manifold, which allows us to deal with a collection of elements $\mathbf{x}^i$ living in their respective manifold $\mathcal{M}^i$, which very often occur in robotics. We define the composite manifold as the concatenation $\mathcal{M}_c \triangleq \left\langle \mathcal{M}^1, \ldots, \mathcal{M}^M \right\rangle$. An element of this composite manifold is denoted by $\mathcal{X}$. We will also write a composite Lie group equivalent of the retraction $\oplus$ and lift $\ominus$ operations. It consists in applying operations to individual elements of the manifold and its tangent space:

$$\mathcal{X} = \begin{bmatrix} \mathbf{x}^1 \\ \vdots \\ \mathbf{x}^M \end{bmatrix}, \quad \mathcal{X} \oplus \boldsymbol{\tau} = \begin{bmatrix} \mathbf{x}^1 \oplus \boldsymbol{\tau}^1 \\ \vdots \\ \mathbf{x}^M \oplus \boldsymbol{\tau}^M \end{bmatrix} \quad \mathcal{X}_2 \ominus \mathcal{X}_1 = \begin{bmatrix} \mathbf{x}_2^1 \ominus \mathbf{x}_1^1 \\ \vdots \\ \mathbf{x}_2^M \ominus \mathbf{x}_1^M \end{bmatrix} \tag{3.26}$$

3.3.2 Back to the MAP optimization problem

Elements of the tangent space can be interpreted as steps that we can make to go from one point of the manifold to another. This ties closely to the notion of Gaussian steps that we defined in Section 3.2.2. We will now consider that the estimated state $\mathcal{X}$ is an element of a composite manifold. In this case, the Gaussian steps of (3.19) are in the composite manifold tangent space and the step updates write:

$$\check{\mathcal{X}} := \check{\mathcal{X}} \oplus \Delta\mathbf{x}^* \tag{3.27}$$

obtained by solving

$$\Delta\mathbf{x}^* = \underset{\Delta\mathbf{x}}{\arg\min} \, ||\mathbf{r}(\check{\mathcal{X}} \oplus \Delta\mathbf{x})||^2. \tag{3.28}$$

Concerning residual formulation in (3.5), if the measurement $\mathbf{z}_i$ is itself an element of a manifold, then the vector subtraction needs to be replaced:

$$\mathbf{e}_i(\mathcal{X}_{S_i}) = \mathbf{h}(\mathcal{X}_{S_i}) \ominus \mathbf{z}_i, \tag{3.29}$$

Note that many state variables belong to vector spaces (such as the robot position), which are trivial manifolds. For those, the $\oplus$ and $\ominus$ operators simply reduce to the addition and subtraction of Cartesian vectors.

3.3.3 Uncertainty on manifolds, Jacobians

Our state variables are random variables approximated by multivariate Gaussian distributions, that can be summarized by their mean vector and covariance matrix. These concepts need to be slightly modified to account for the manifold nature of some of these by defining Gaussian distributions on manifolds. Let $\boldsymbol{\tau} \in \mathbb{R}^n$ be a small perturbation around a point $\bar{\mathbf{x}} \in \mathcal{M}$, so that we can write:

$$\mathbf{x} = \bar{\mathbf{x}} \oplus \boldsymbol{\tau}, \quad \boldsymbol{\tau} = \mathbf{x} \ominus \bar{\mathbf{x}} \tag{3.30}$$

where $\bar{\mathbf{x}}$ designates the mean value of the distribution on $\mathbf{x}$. We define a zero mean "Cartesian" Gaussian distribution on the tangent space, the covariance being defined as:

$$\boldsymbol{\Sigma}_{\mathbf{x}} \triangleq \mathbb{E}[\boldsymbol{\tau}\boldsymbol{\tau}^\top] = \mathbb{E}[(\mathbf{x} \ominus \bar{\mathbf{x}})(\mathbf{x} \ominus \bar{\mathbf{x}})^\top] \in \mathbb{R}^{n \times n} \tag{3.31}$$

where $\mathbb{E}[\cdot]$ denotes the expectation operator. By a slight abuse of notation, we note that the manifold distribution is $\mathbf{x} \sim \mathcal{N}(\bar{\mathbf{x}}, \boldsymbol{\Sigma}_{\mathbf{x}})$.

As an illustration, the group of unit quaternions $\mathbb{H} = \{\mathbf{q} \in \mathbb{R}^4 \mid \mathbf{q} \odot \mathbf{q}^* = 1\}$ under quaternion multiplication $\odot$ is a manifold (the S^3 sphere, p=4 and n=3) [Sol12]. The tangent space is the set of antisymmetric matrices, which is isomorphic to the angle axis rotation vectors $\boldsymbol{\theta} \in \mathbb{R}^3$. If we naively define the covariance matrix of a quaternion as $\mathbb{E}[(\mathbf{q} - \bar{\mathbf{q}})(\mathbf{q} - \bar{\mathbf{q}})^\top] \in \mathbb{R}^{4\times4}$, this covariance matrix is ill-defined. The covariance must be more properly defined as a 3×3 real matrix on the angle axis space.

In general, covariances can be propagated through nonlinear function of random variables using the chain rule. For instance, if $\mathbf{x} \in \mathbb{R}^{n_x}$ and $\mathbf{y} \in \mathbb{R}^{n_y}$ are Cartesian multivariate Gaussian distribution related by the nonlinear function f, propagating the $\mathbf{x}$ covariance $\boldsymbol{\Sigma}_{\mathbf{x}} \in \mathbb{R}^{n_x \times n_x}$ through f writes:

$$f : \mathbf{x} \in \mathbb{R}^{n_x} \to \mathbf{y} \in \mathbb{R}^{n_y}, \qquad \boldsymbol{\Sigma}_{\mathbf{y}} \approx \left.\frac{Df}{D\mathbf{x}}\right|_{\mathbf{x}} \boldsymbol{\Sigma}_{\mathbf{x}} \left.\frac{Df}{D\mathbf{x}}\right|_{\mathbf{x}}^\top \tag{3.32}$$

The same equation applies for random variables $\mathbf{x} \in \mathcal{M}_{\mathbf{x}}$ and $\mathbf{y} \in \mathcal{M}_{\mathbf{y}}$ living on manifolds. The definition of Jacobians has however to be redefined as:

$$\frac{Df}{D\mathbf{x}} \triangleq \lim_{\boldsymbol{\tau}\to 0} \frac{f(\mathbf{x} \oplus \boldsymbol{\tau}) \ominus f(\mathbf{x})}{\boldsymbol{\tau}} \tag{3.33}$$

These Jacobians also play a central role in the derivation of the residual Jacobians (3.16). Examples of manifold Jacobian derivations in the context of Lie groups can be found in [SDA18].

3.3.4 Lie groups for robotic state estimation

Riemannian geometry is enough to deal with optimization problems with variables living on smooth manifolds. However, many of these state variables, in the context of robotics, also exhibit a group structure, which gives them extra interesting properties.

A *group* is an ordered pair $(\mathcal{G}, \circ)$ comprised of a set $\mathcal{G}$ and a composition law $\circ$ that together satisfy the group axioms:

$$\text{Closure under } \circ \ : \mathbf{x} \circ \mathbf{y} \tag{3.34}$$
$$\text{Associativity } \circ \ : (\mathbf{x} \circ \mathbf{y}) \circ \mathbf{z} = \mathbf{x} \circ (\mathbf{y} \circ \mathbf{z}) \tag{3.35}$$
$$\text{Identity element } \mathcal{E} \ : \mathbf{x} \circ \mathcal{E} = \mathcal{E} \circ \mathbf{x} = \mathbf{x} \tag{3.36}$$
$$\text{Inverse existence } \mathbf{x}^{-1} \ : \mathbf{x}^{-1} \circ \mathbf{x} = \mathbf{x} \circ \mathbf{x}^{-1} = \mathcal{E} \tag{3.37}$$

$\forall\, \mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathcal{G}$.

Simply stated, a Lie group is a group whose elements live in a smooth manifold $\mathcal{M}$.

A vector can represent a displacement from one point to another along a straight line in the euclidean space, or directly a point (displacement from the origin). Analogously, an element of a Lie group represents either a path from one element to another along a geodesic of the manifold or directly an element of the group (displacement from the identity element $\mathcal{E}$).

In particular, the identity element $\mathcal{E}$ relates to a notion of a global reference from which each element of the manifold can be compared. The tangent space at this element is called the Lie algebra $\mathfrak{m} \triangleq \mathcal{T}_{\mathcal{E}}\mathcal{M}$.

Let us compute the retraction operations in the case of Lie groups. We will follow the example of 3D rotations as they are of great use in robotics and provide good intuitions regarding the general behavior of Lie groups. The group of 3D rotations is defined as the Special Orthogonal group in 3D:

$$SO(3) = \left\{ \mathbf{R} \in \mathrm{GL}(n, \mathbb{R}) |\, \mathbf{R}\,\mathbf{R}^\top = \mathbf{R}^\top\,\mathbf{R} = \mathbf{I},\ det(\mathbf{R}) = 1 \right\} \tag{3.38}$$

under matrix multiplication.

Firstly, properties that make $SO(3)$ a group are immediate from the properties of matrix multiplication:

- Closure under the matrix product
- Associativity through the matrix product
- Identity element $\mathbf{I}_3$
- Inverse element $\mathbf{R}^{-1} = \mathbf{R}^\top$ (from the group definition)

Let us explicit the manifold structure of $SO(3)$. It can be shown [SDA18] that taking the time difference of a rotation matrices results in an element of its local tangent space:

$$\dot{\mathbf{R}} = \mathbf{R}[\boldsymbol{\omega}]^\wedge \in \mathcal{T}_{\mathbf{R}} SO(3) \tag{3.39}$$

where $[\cdot]^\wedge$ is the skew-symmetric operator associated with the cross product. $[\boldsymbol{\omega}]^\wedge = \boldsymbol{\omega} \times \cdot$. The Lie algebra $\mathfrak{so}(3)$ is therefore the 3 dimensional vector space of antisymmetric matrices. For a constant $\boldsymbol{\omega}$, this defines an ordinary differential equation (ODE) whose solution is $\mathbf{R}(t) = \mathbf{R}_0 \exp([\boldsymbol{\omega}]^\wedge t)$ where exp is the matrix exponential:

$$\exp(\mathbf{A}) = \sum_{k=0}^{\infty} \frac{\mathbf{A}^k}{k!}. \tag{3.40}$$

For $SO(3)$ and most other Lie groups, properties of the Lie algebra simplify the infinite sum of the matrix exponential to a simpler closed-form formula. In the $SO(3)$ case, this formula is known as the Rodrigues' formula:

$$\mathrm{Exp}(\boldsymbol{\theta}) = \exp([\boldsymbol{\theta}]^\wedge) = \mathbf{I} + [\mathbf{u}]^\wedge \sin(\theta) + [\mathbf{u}]^{\wedge,2}(1 - \cos(\theta)) \tag{3.41}$$

where $\boldsymbol{\theta} \triangleq \theta\mathbf{u} \in \mathbb{R}^3$ and θ is the norm of the rotation vector (angle in radians) and $\mathbf{u}$ its unitary direction. Finally, we have denoted by Exp the composition of the *hat* operator ($\boldsymbol{\theta}^\wedge \triangleq [\boldsymbol{\theta}]^\wedge \in \mathfrak{so}(3)$) and the matrix exponential: $\mathrm{Exp}(\boldsymbol{\tau}) \triangleq \exp(\boldsymbol{\tau}^\wedge)$. The inverse operation, that maps elements from the group to the Lie algebra, is defined as the Log map. A general representation of Lie group operations can be found in Fig. 3.2

With all of that in mind, the link with the Riemannian manifold terminology becomes clearer. We can compose a rotation with an increment ${}^1\boldsymbol{\theta}$ taken in the tangent space at $\mathbf{x}_1$ to obtain another rotation: $\mathbf{R}_2 = \mathbf{R}_1 \mathrm{Exp}({}^1\boldsymbol{\theta})$. This corresponds to the $\oplus$ retraction operator. Conversely, the lift $\ominus$ corresponds to the application of The logarithm on the relative rotation as described here:

$$\mathbf{R}_2 = \mathbf{R}_1 \oplus {}^1\boldsymbol{\theta} = \mathbf{R}_1 \mathrm{Exp}({}^1\boldsymbol{\theta}) \tag{3.42}$$

$${}^1\boldsymbol{\theta} = \mathbf{R}_2 \ominus \mathbf{R}_1 = \mathrm{Log}(\mathbf{R}_1^\top \mathbf{R}_2) \tag{3.43}$$

In robotics terms, ${}^1\boldsymbol{\theta}$ is the axis angle representation of the relative rotation ${}^1\delta\mathbf{R}_{12} = \mathbf{R}_1^\top \mathbf{R}_2$ and *local* rotations represent the orientation of reference frames $\{1, 2\}$ in a *global* world reference frame. The identity element, therefore, corresponds to the world frame.

Due to the general non-commutativity of the group operation, an additional $\oplus$ operator exists that retracts and composes vectors at the Lie algebra instead of the local tangent space, the left-$\oplus$. The local operator, that corresponds to the manifold retraction, is called the right-$\oplus$. The Adjoint $\mathrm{Ad}_{\mathbf{x}}\mathcal{M}$ is an operator that maps elements from the local tangent space to the Lie algebra.

$${}^{\mathcal{E}}\boldsymbol{\tau} = \mathrm{Ad}_{\mathbf{x}}\mathcal{M}\, {}^{\mathbf{x}}\boldsymbol{\tau} \tag{3.44}$$

For rotation matrices, the adjoint matrix is $\mathrm{Ad}_{\mathbf{R}}SO(3) = \mathbf{R}$: to change the reference frame of the angle axis from local to global reference, it needs to be rotated from one frame to another. Those operations are illustrated in Fig. 3.3.

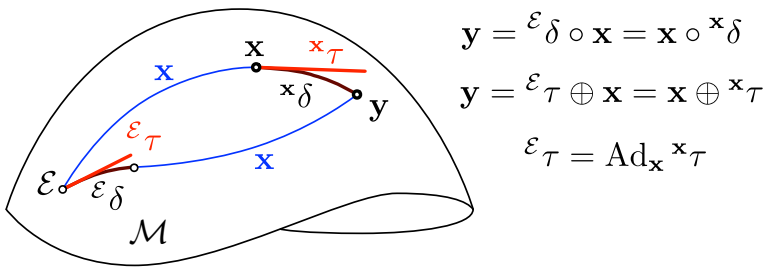
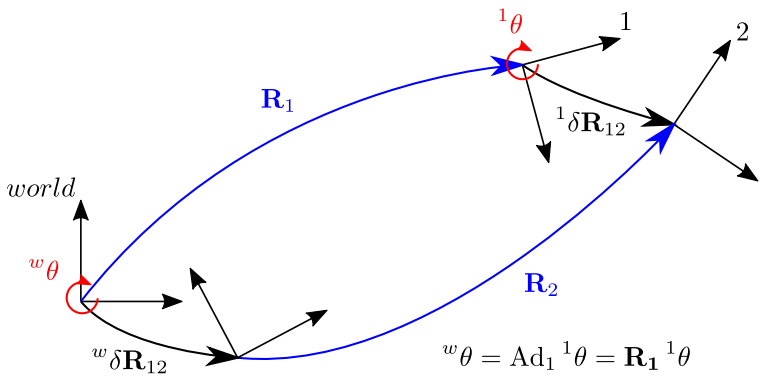


Figure 3.2: Representations of a Lie group including the identity element $\mathcal{E}$, operations of composition between increments and relations between a tangent space increment at the identity ${}^{\mathcal{E}}\boldsymbol{\tau}$ and at the local element ${}^{\mathbf{x}}\boldsymbol{\tau}$ through the adjoint matrix $\mathrm{Ad}_{\mathbf{x}}$. Figure adapted from [SDA18].



$$\text{Right } \oplus \ : \ \mathbf{R_2} = \mathbf{R_1} \oplus {}^1\theta = \mathbf{R_1}\mathrm{Exp}({}^1\theta) = \mathbf{R_1}\,{}^1\delta\mathbf{R}_{12}$$
$$\text{Left } \oplus \ : \ \mathbf{R_2} = {}^w\theta \oplus \mathbf{R_1} = \mathrm{Exp}({}^w\theta)\mathbf{R_1} = {}^w\delta\mathbf{R}_{12}\mathbf{R_1}$$

Figure 3.3: Representation of $SO(3)$ Lie groups operations. Note that we represented frames with different origins for better readability while the group elements only represent pure rotations (no translation). $world$ is the "global" reference frame of the problem to which we assign the identity element of $SO(3)$: $\mathbf{I}_3$. 1 and 2 are two frames of reference defined by their orientations with respect to $world$, $\mathbf{R}_1$ and $\mathbf{R}_2$. $\delta\,\mathbf{R}_{12}$ is the relative rotation between elements $\mathbf{R}_1$ and $\mathbf{R}_2$ and can be computed using the exponential map on the angle axis $\boldsymbol{\theta}$ expressed in $world$ or local frame, depending on whether we use the right or left $\oplus$ operator.

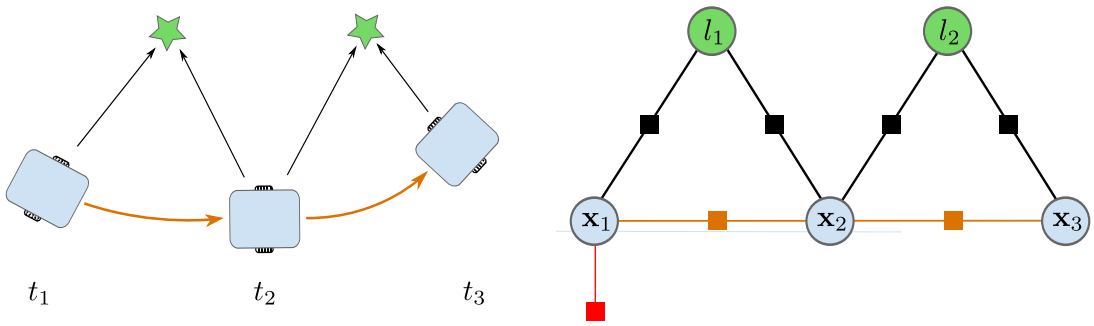
3.4 Factor Graphs: a visual language for robotics estimation

A crucial aspect of solving the MAP problem is the fact that the likelihood function is factorizable. This represents the fact that the problem exhibits a particular structure that has important computational implications. We will first explain how this factorization can be described visually using a graphical model known as the *Factor Graph* and then link this representation to the sparsity of the matrices involved in solving the NLLS problem.

3.4.1 Factor Graph representation

Let us consider the toy example represented in Fig. 3.4a. We wish to estimate the trajectory of a differential robot, that is its states at chosen timestamps called *Keyframes*, and remarkable elements of the environment, called *landmarks*. We suppose that this robot is equipped with an odometer, whose measurements integrated over time provide relative transformations between Keyframes, and an exteroceptive sensor that provides relative measurements between Keyframes and landmarks.

In this case, the state variables are $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, l_1, l_2\}$ and measurements $\mathcal{Z} = \{z_{o,1}, z_{o,2}, z_{e,1}, z_{e,2}\}$. We also apply a prior on the pose of the first Keyframe of the trajectory, which can be understood as fixing the frame origin of the reference frame in which the estimation is done.



(a) (b)

Figure 3.4: (a): toy estimation problem, a differential drive robot equipped with an odometer moves in an scene with landmarks represented by stars. (b): Factor Graph representation the problem, the estimated variables are represented by circles (blue for robot Keyframes, green for landmarks) and factors by squares (orange for odometry, black for exteroceptive sensor, red for the prior).

The factorization of the likelihood of (3.6) in this example writes:

$$p(\mathcal{X}|\mathcal{Z}) \propto \phi_0(\mathbf{x}_0) \tag{3.45}$$
$$\phi_1(\mathbf{x}_1, \mathbf{x}_2)\phi_2(\mathbf{x}_2, \mathbf{x}_3) \tag{3.46}$$
$$\phi_3(\mathbf{x}_1, l_1)\phi_4(\mathbf{x}_2, l_1, l_2)\phi_5(\mathbf{x}_3, l_2) \tag{3.47}$$

This factorization can be represented as a *factor graph* as seen in Fig. 3.4b. Factor graphs are popular probabilistic graphical models [KF09] that can describe a vast family of statistical models [Loe04]. In the general sense, a factor graph is a bipartite graph that represents the factorization of a function of several variables. In estimation , we use it to represent the factorized likelihood (3.11), or equivalently the NLLS problem (3.12).

We adopt the visual notation commonly found in robotics: round nodes for variables, square nodes for factors, edges represent the dependency of each factor on a subset of variables. In the following parts, we will use the following terminology to designate factors depending on which variables they depend on:

- **Unary factor** ϕ: depends on a single Keyframes, obtained from a an absolute measurement (*e.g.* a prior, GPS etc.)
- **Motion factor** ϕ: depends on successive Keyframes, obtained from a motion sensor (*e.g.* wheel odometry, IMU etc.)
- **Relative** or **exteroceptive factor** ϕ: depends on a Keyframe and a part of the map, a few landmarks in this case, obtained from exteroceptive sensors

Dellaert and Kaess [DK06] were the first to recognize the link between NLLS problems and factor graphs [DL19]. Over the last two decades, they have grown in popularity among roboticists as a visual language to describe estimation problems [DK+17] and planning problems [Don+16]. As Frank Dellaert puts it [2], factor graphs are "an amazing thing to write on blackboards". Aside from providing insights into the computational structure

[2] Citation from a recent talk, see this video, around 15:30.

of the problems, they can be used as a common language between engineering teams to convey insights about the nature of the problems to solve.

The problems typically represented by factor graphs and found in estimation are very sparse. Many specialized solvers [Gri+11; Kae+12; Ila+17] have been implemented to exploit this sparsity. Let us discuss the basis of sparse NLLS to conclude this chapter.

3.4.2 Sparsity of the NLLS problem

Each residual of the NLLS problem is a function of a small subset of the state variables. Therefore, each factor node in the Factor Graph representation of the problem is connected to but a few of the variable nodes. This makes for a graph with low connectivity. We will show that this low connectivity is linked to an important sparsity of the Jacobian matrix of the residuals.

The Jacobian matrix of the residuals $\mathbf{J} \in \mathbb{R}^{M \times N}$ is a big sparse matrix of dimensions

- M rows = $\sum_{i=1}^{m} dim(\mathbf{e}_i) \rightarrow$ the sum of residual dimensions
- N columns = $\sum_{i=1}^{n} dim(\mathcal{T}_{\mathbf{x}_i}\mathcal{M}_i) \rightarrow$ the sum of variables tangent-spaces dimensions

Each block column corresponds to one of the state variables and each row corresponds to a residual. For each row, only the blocks corresponding to the state variables that the residual depends on are non-zero. For instance, the Jacobian corresponding to the toy problem described in Fig. 3.4a is:

$$\mathbf{J} = \begin{pmatrix} J^{0}_{\mathbf{x}_1} & & & & \\ J^{o1}_{\mathbf{x}_1} & J^{o1}_{\mathbf{x}_2} & & & \\ & J^{o2}_{\mathbf{x}_2} & J^{o2}_{\mathbf{x}_3} & & \\ J^{e1}_{\mathbf{x}_1} & & & J^{o2}_{\mathbf{l}_1} & \\ & J^{e1}_{\mathbf{x}_2} & & J^{e1}_{\mathbf{l}_1} & \\ & J^{e2}_{\mathbf{x}_2} & & & J^{e2}_{\mathbf{l}_2} \\ & & J^{e3}_{\mathbf{x}_3} & & J^{e3}_{\mathbf{l}_2} \end{pmatrix} \tag{3.48}$$

The approximate Hessian matrix $\mathbf{H} \in \mathbb{R}^{N \times N}$ displays a similarly remarkable sparsity that makes the resolution of the linear system at each step very efficient. Algorithms such as the Schur complement [BBV04, Section A.5.5], Cholesky factorization [BBV04, Section C.3], and QR factorization [BBV04, Section C.5] were applied by different authors to the problem of SLAM and estimation.

3.4.3 The algorithm (revisited)

To tie things together, we now recapitulate the Gauss-Newton algorithm in the context of variables living on manifolds. We highlight the parts that differ from the regular Gauss-Newton algorithm with vector spaces.

1. Initialize the state estimate at an initial value $\check{\mathcal{X}} := \mathcal{X}^0$
   $\mathcal{X}$ lives on a composite manifold. Its error and increments are expressed in its local tangent space.

2. Approximate the NLLS cost function around the current estimate as a quadratic function

The linear approximation is done using the notation of Jacobian on manifolds described in Section 3.3.3. $\Delta\mathbf{x}$ is defined as the Cartesian representation of a composite tangent space element.

3. Find the optimal step $\Delta\mathbf{x}^*$ by solving a linear set of equations (3.18)
   This part does not change. As mentioned in Section 3.2.3, globalization of Gauss-Newton such as the Levenberg-Marquardt algorithm may alter the linear system to be solved for increased performance.

4. Update the current state estimate $\check{\mathcal{X}} := \check{\mathcal{X}} \oplus \Delta\mathbf{x}^*$
   The update of the current state is done using the retraction operator $\oplus$ of the composite manifold (3.26).

5. Loop over steps 2-4 until convergence

When the estimation algorithm has converged we can also obtain the covariance on the estimates (details and illustration are available in the appendix Chapter A). This is however a costly computation that is usually reserved for offline investigation of the estimation quality.

3.5 Conclusion

We have described a general algorithm to solve NLLS problems with variables belonging to manifolds and weighted by covariance matrices. This problem comes from framing the state estimation as a MAP problem and using multiple assumptions on the nature of the measurement models. These measurement models are described as residuals weighted by their respective covariance matrices, representing the confidence we have in the sensor measurements. Factor Graphs can represent visually those problems and their connectivity exactly translates to the sparsity of the linear problems appearing in the optimization algorithm.

In our opinion, two equally important research directions start from here: (i) finding new solvers extending these ideas to new sets of problems, such as multi-robot or multi-hypothesis problems, and (ii) formulating new, efficient, and generalizable measurement models. This thesis mostly theoretically contributes to the direction of extending the estimation model (ii) while we mostly rely on existing algorithms (i) to solve the subsequent problems. A commonly used denomination in the context of robotics state estimation is to refer to the solver implementation as the *backend* while the algorithms processing sensor information to create residuals are referred to as *front-ends*.

The next chapters concentrate on a range of sensor modalities useful for legged robots: object pose retrieval from vision algorithms, a generalized pre-integration theory for high rate sensors, legged robot kinematics, and the application of pre-integration to legged robots force measurements.

Chapter 4

Object-level vision

Contents

The field of computer vision comprises a wide range of methods to extract information about the real world from images taken by digital cameras. Among others, this information may be the structure of the environment (mapping or object tracking), the movement of the system to which is attached the camera (localization), or both at the same time (Simultaneous Localization And Mapping, *a.k.a.* SLAM). In the context of robotics, the images are obtained one after another at a constant rate. This leads to a somewhat continuous evolution of the image appearance, which can be leveraged in various ways (feature tracking, motion models, etc.). We are here interested in solving SLAM, which is the most general application of this method and is used when a robot needs to localize itself in an a priori unknown environment. A vast literature of dedicated algorithms have been developed in this regard, from approaches based on geometrical constraints using sparse feature detection [MMT15; Fer+21] to methods leveraging photometric consistency [NLD11; ESC14; EKC17], certain methods combining ideas from both approaches [FPS14; For+16].

These algorithms aim at generalizing to any kind of environment and have their respective strengths and weaknesses [FRR15]. Although open-source code is available for

most, these pieces of software may be difficult to integrate into other frameworks. Besides, though providing quite accurate localization, maps obtained from these methods are rarely immediately useful, lacking semantic information about the scene.

In this thesis, rather than investing time and energies in new vision processing algorithms, we chose to focus our attention on legged-robot state estimation. In particular, we are interested in localizing the robot with respect to objects of interest such as stairs, tables, or the like, whose geometrical models are known. Approaches based on object-level measurement can provide an affordable alternative, and are readily available. These systems assume the presence of known objects in the scene which can be detected in the images. Dedicated algorithms can then infer the object pose relative to the camera, which can be used to build *object-level SLAM* systems.

In this chapter, we present two measurement models based on available software, that enable the implementation of this paradigm. The first one uses unique AprilTag fiducial markers [WO16] (to which we may refer to as *tags* informally) that are laid out sparsely in the scene. The second uses a deep-learning-based method [Lab+20] to obtain relative poses from an object of interest, such as a pair of stairs. Instead of artificially augmenting the environment with fiducial markers, we assume that objects of interest exist in the scene, whose CAD models are known. Both systems lack ways to estimate the uncertainty of their output and we, therefore, propose methods for the computation of their measurements outputs covariances.

In both models, we assume that we have a calibrated pinhole camera and that the images have been corrected for distortions. The camera reference frame C has its origin at the camera optical center and follows the convention X-Y-Z = Righ-Down-Front, Front being the optical axis direction, looking through the lens. This chapter starts by introducing the measurement model based on a general 6D factor used by both systems. Methods to recover the covariance of both algorithms are then given. For applications fusing these models with IMU data, please refer to Chapter 8 and Chapter 10.

4.1 Relative 6D pose factor

In this section, we will derive a general 6D relative transformation factor between a Keyframe and a object, whose poses in world frame are respectively ${}^{W}\mathbf{T}_{B} \in SE(3)$ and ${}^{W}\mathbf{T}_{O} \in SE(3)$. We denote by ${}^{a}\mathbf{T}_{b} \in SE(3)$ 6D transformations transforming a 3D point ${}^{b}\mathbf{p} \in \mathbb{R}^3$ expressed in frame b to its expression in frame a ${}^{a}\mathbf{p} = {}^{a}\mathbf{T}_{b}\,{}^{b}\mathbf{p} \in \mathbb{R}^3$.

Let us assume that an algorithm estimates directly ${}^{C}\widetilde{\mathbf{T}}_{O}$, a measurement of the pose of an element of the scene with an attached frame O with respect to the camera frame C. The kinematic chain of the problem described in Fig. 4.1 unrolls as ${}^{W}\mathbf{T}_{O} = {}^{W}\mathbf{T}_{B}\,{}^{B}\mathbf{T}_{C}\,{}^{C}\mathbf{T}_{O}$ where W and B correspond to the world and body frames. ${}^{B}\mathbf{T}_{C}$ corresponds to the extrinsic pose of the camera. Given measurement ${}^{C}\widetilde{\mathbf{T}}_{O}$, this relation can be turned into a residual relating the robot pose, the camera extrinsics, and the object pose:

$$\mathbf{e}_V({}^{W}\mathbf{T}_{B}, {}^{B}\mathbf{T}_{C}, {}^{W}\mathbf{T}_{O}) = \left[({}^{W}\mathbf{T}_{B}\,{}^{B}\mathbf{T}_{C})^{-1}\,{}^{W}\mathbf{T}_{O}\right] \ominus {}^{C}\widetilde{\mathbf{T}}_{O} \in \mathbb{R}^6 \tag{4.1}$$

$$= \underset{SE(3)}{\mathrm{Log}}\left(({}^{W}\mathbf{T}_{B}\,{}^{B}\mathbf{T}_{C}\,{}^{C}\widetilde{\mathbf{T}}_{O})^{-1}\,{}^{W}\mathbf{T}_{O}\right) \in \mathbb{R}^6 \tag{4.2}$$

where here $\mathrm{Log}_{SE(3)}$ denotes the log map on $SE(3)$ [SDA18].

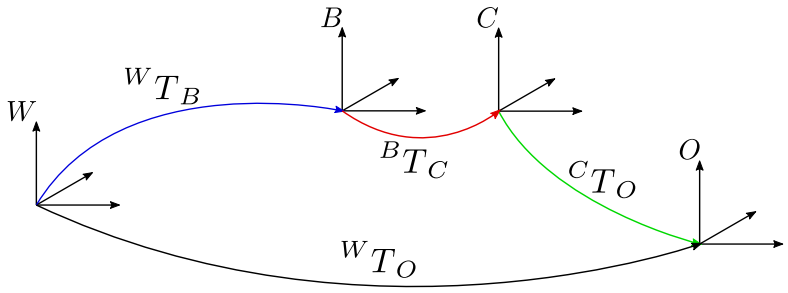


Figure 4.1: Kinematic tree of the object/camera measurement model

Depending on the nature of the problem, we can fix some of the variables of the residual. In Table 4.1, various estimation problems are modeled depending on which variable is fixed.

Table 4.1: Some possible estimation problems corresponding to fixing either of the variables. Estimated variables are checkmarked with " ✓". Fixed variables are marked with "X".

	${}^{W}\mathbf{T}_{B}$	${}^{B}\mathbf{T}_{C}$	${}^{W}\mathbf{T}_{O}$
SLAM with extrinsics calibration	✓	✓	✓
SLAM	✓	X	✓
Localization with extrinsics calibration	✓	✓	X
Localization	✓	X	X
Mapping	X	X	✓

We also assume that we have access to the covariance of this measurement $\mathbf{\Sigma}_V \in \mathbb{R}^{6\times6}$.

This factor is general enough to be used in two applications that we successively describe, based on AprilTag fiducial markers and CosyPose. For both, we explain the algorithm behind the pose measurements and propose covariance models.

4.2 Fiducial markers

4.2.1 Markers Pose estimation algorithms

Fiducial markers (such as ARToolkit [KB99], ARTag [Fia05], ArUco [Gar+14]) are widely used in robotics and Augmented Reality (AR) applications as they provide an easy way to obtain a relative 6D pose between a calibrated camera and the marker. These markers are designed so that they are uniquely and robustly identifiable in any scene [WO16; RMM18].

The pose extraction happens in two steps. First, the tag corners sub-pixel projections are extracted from the image along with the unique id of the detected tag. In this work we used the AprilTag library [WO16] for its performances and popularity within the robotics community. Second, knowing the tag size, we can compute the position of these corners in the local tag frame, as defined in Fig. 4.3. Recovering the pose of the tag from the corner projections and their local coordinates is then an instance of the Perspective-n-Point (PnP) algorithm [Gao+03], which requires at least four 2D-3D correspondences to

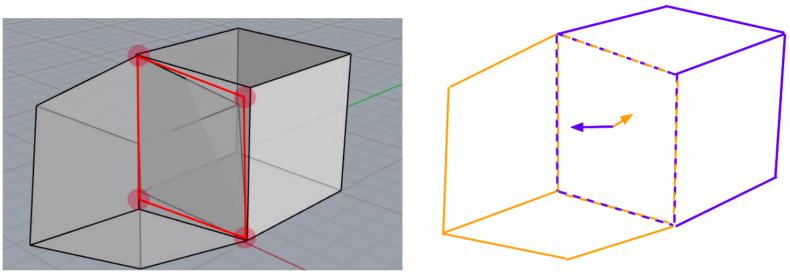


Figure 4.2: Illustration of the tag ambiguity problem. The red circles represent uncertainty in the corner detection due to pixel noise. The conic perspective projections of faces from two cubes with 120 degrees orientation difference fall withing these regions and are, therefore, hard to distinguish. Figure taken from [JMS17].

compute a unique solution the 3D pose [1]. A vast literature has been written on the topic, many methods relying on efficient analytical models [Gao+03; LMF09; CB14; TL20b], sometimes refined by a nonlinear maximum likelihood step.

4.2.2 Ambiguity in the pose estimation

A known problem of fiducial markers is the orientation ambiguity of detections in noisy images. This phenomenon may make the PnP estimation of the tag orientation jump wildly from one frame to the next. This is due to the fact that the conic perspective projection of the corners is weak, that is the difference between two symmetrical tag orientations is very close. If this difference falls below the camera pixel noise, then the two solutions are indistinguishable, as shown in Fig. 4.2. This happens especially when the tag corner projection is close to a square, that is when the tag is orthogonal to the axis passing through the camera and tag origins. We refer to this situation as tags being *fronto-parallel* with respect to the camera. This phenomenon is intensified when the tag is far from the camera as it spans a smaller region of the image.

One solution is to define the tag factor residual as the pixel error between the detected corners and their projection given the current tag pose estimate and let the optimizer find the most probable tag poses given the complete set of measurements. However, if one tag is wrongly initialized, the optimizer is not guaranteed to leave the local minimum with new measurements.

Our solution is to bring the disambiguation to the front-end side. [CB14] provides an implementation of the PnP problem that retrieves both ambiguous poses. When expressed in the camera frame, both solutions share the tag position and differ only in their orientation. We typically want to select the solution with the smallest error. However, if the reprojection errors e_1 and e_2 are too close (we test for $\frac{e_2}{e_1} < h$ with $e_1 \leq e_2$ and h an empirical threshold), we increase the rotational part of the covariance matrix by a great factor to cancel the influence of the wrong tag orientation on the overall MAP problem.

4.2.3 Covariance model

Obtaining a covariance of the estimated transformation is an important step toward the integration of these measurements in a sensor fusion algorithm.

[1] The P3P algorithm yields up to four geometrically feasible solutions, a fourth point being then used to select the right one.

Urban [ULH16, Eq. (23)] uses a careful parameterization of the homogeneous coordinates associated with 2D features to develop a maximum likelihood PnP algorithm and also proposes a covariance model based on the problem Hessian. However, the particular choice of variable parameterization makes the formulation of the covariance computation overcomplicated. We think that a simpler yet equivalent formulation is possible.

Usually, when a nonlinear mapping of noise variables from one space to another is available, we can propagate the covariance through the nonlinearity as shown in Eq. (3.32). Instead of directly obtaining Jacobians from the PnP algorithms, we found that a natural way to proceed is to take the opposite direction as follows. It should somehow be possible, knowing the marker size and the relative pose measurement, and assuming pixel noise, to recover the pose covariance $\mathbf{\Sigma_T}$. A simple model of pixel noise is to assume isotropic Gaussian noise on the pixels with variance σ_x^2. If we stack four pixel (the four tag corners) $\mathbf{x}_i = [u_i, v_i] \in \mathbb{R}^2$ in the column vector $\mathbf{x} = [\mathbf{x}_1\ \mathbf{x}_2\ \mathbf{x}_3\ \mathbf{x}_4]^\top$, we have therefore that $\mathbf{x}$ is corrupted by a Gaussian noise $\mathbf{\Sigma_x} = \sigma_x^2 \mathbf{I}_8$, where σ_x usually takes values of 1 or 2 pixels. The PnP algorithm provides us with a function pnp defined as:

$$\begin{aligned} \text{pnp}_w : \mathbb{R}^8 &\rightarrow SE(3) \\ \mathbf{x} &\rightarrow {}^C\mathbf{T}_O = \text{pnp}_w(\mathbf{x}) \end{aligned} \tag{4.3}$$

where w denotes the dependency on the width of the marker. This "pnp" function implementation depends on the specificities of the PnP algorithm used and is in general hard to analytically differentiate. Instead, if we consider the inverse function.

$$\begin{aligned} \text{proj}_w : SE(3) &\rightarrow \mathbb{R}^8 \\ {}^C\mathbf{T}_O &\rightarrow \mathbf{x} = \text{proj}_w({}^C\mathbf{T}_O)' \end{aligned} \tag{4.4}$$

"proj" maps the relative pose to the projection of the tag in the image. A rather simple Jacobian expression can be derived using the chain rule, as follows.

We denote the marker corner coordinates in the marker frame as shown in Fig. 4.3. As a convention [WO16], we order the corners counter-clockwise starting from bottom-left (looking straight at the tag).

$$\mathbf{c} = \begin{pmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \\ \mathbf{c}_3 \\ \mathbf{c}_4 \end{pmatrix} \quad \mathbf{c}_1 = \begin{pmatrix} -\frac{w}{2} \\ \frac{w}{2} \\ 0 \end{pmatrix} \quad \mathbf{c}_2 = \begin{pmatrix} \frac{w}{2} \\ \frac{w}{2} \\ 0 \end{pmatrix} \quad \mathbf{c}_3 = \begin{pmatrix} \frac{w}{2} \\ -\frac{w}{2} \\ 0 \end{pmatrix} \quad \mathbf{c}_4 = \begin{pmatrix} -\frac{w}{2} \\ -\frac{w}{2} \\ 0 \end{pmatrix}. \tag{4.5}$$

Then, assuming that images are corrected (no distortion), the pinhole camera model gives us that

$$\mathbf{x}_i = \text{eucl}(\mathbf{h}_i) = \text{eucl}(\mathbf{K}\,{}^C\mathbf{T}_O\mathbf{c}_i) \tag{4.6}$$

for each corner c_i, where h_i are the homogeneous coordinates representing the projected corners and "eucl" is the function that maps homogeneous coordinates to euclidean pixel coordinates defined as

$$\begin{aligned} &\text{eucl} : \mathbb{R}^3 \rightarrow \mathbb{R}^2 \\ &\mathbf{h} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \rightarrow \mathbf{x} = \begin{pmatrix} x/z \\ y/z \end{pmatrix}. \end{aligned} \tag{4.7}$$

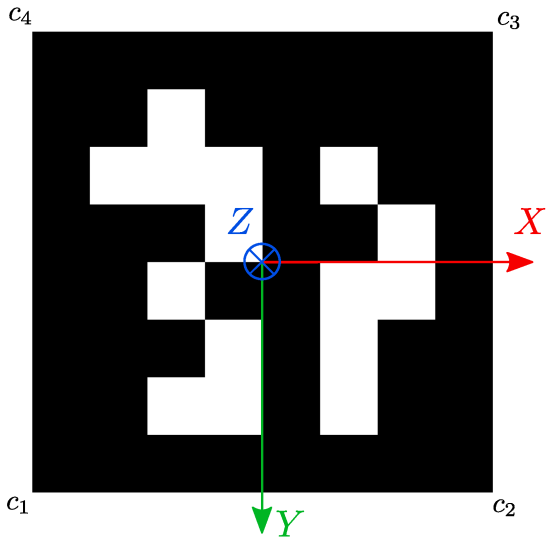


Figure 4.3: AprilTag local coordinate systems and corners conventions [WO16]

We need to compute the Jacobian of each corner projection with respect to the estimated relative pose, decomposed using the chain rule as

$$\mathbf{J}_{\mathbf{T}}^{\mathbf{x}_i} = \mathbf{J}_{\mathbf{h}_i}^{\mathbf{x}_i} \mathbf{J}_{\mathbf{T}}^{\mathbf{h}_i}. \tag{4.8}$$

Regarding the transformation, we will consider it to be an element of the composite Lie group $\langle \mathbb{R}(3) \times SO(3) \rangle$ as it is done in our solver. The expressions of those functions are, therefore, expressed as:

$$\begin{aligned}
\mathbf{h}_i &= \mathbf{K}({}^{C}\mathbf{R}_O \mathbf{c}_i + {}^{C}\mathbf{p}_O) \\
\mathbf{J}_{{}^{C}\mathbf{p}_O}^{\mathbf{h}_i} &= \mathbf{K} \\
\mathbf{J}_{{}^{C}\mathbf{R}_O}^{\mathbf{h}_i} &= -\mathbf{K}\, {}^{C}\mathbf{R}_O [\mathbf{c}_i]_\times \\
\mathbf{J}_{{}^{C}\mathbf{T}_O}^{\mathbf{h}_i} &= [\mathbf{J}_{{}^{C}\mathbf{p}_O}^{\mathbf{h}_i} \quad \mathbf{J}_{{}^{C}\mathbf{R}_O}^{\mathbf{h}_i}] = \mathbf{K}[\mathbf{I}_3 \quad -{}^{C}\mathbf{R}_O [\mathbf{c}_i]_\times]
\end{aligned} \tag{4.9}$$

while the Jacobian of the map from homogeneous coordinates to euclidean pixels is

$$\mathbf{J}_{\mathbf{h}_i}^{\mathbf{x}_i} = \begin{pmatrix} 1/z_i & 0 & -x_i/z_i^2 \\ 0 & 1/z_i & -y_i/z_i^2 \end{pmatrix}. \tag{4.10}$$

Finally, we stack the 4 Jacobians to get the full Jacobian to be used for covariance propagation:

$$\mathbf{J} \triangleq \mathbf{J}_{{}^{C}\mathbf{T}_O}^{\mathbf{x}} = \begin{pmatrix} \mathbf{J}_{{}^{C}\mathbf{T}_O}^{\mathbf{x}_1} \\ \mathbf{J}_{{}^{C}\mathbf{T}_O}^{\mathbf{x}_2} \\ \mathbf{J}_{{}^{C}\mathbf{T}_O}^{\mathbf{x}_3} \\ \mathbf{J}_{{}^{C}\mathbf{T}_O}^{\mathbf{x}_4} \end{pmatrix} \in \mathbb{R}^{8\times 6} \tag{4.11}$$

.

We therefore have the covariance propagation equation $\mathbf{\Sigma}_{\mathbf{x}} = \mathbf{J}\mathbf{\Sigma}_{\mathbf{T}}\mathbf{J}^\top$. Since $\mathbf{\Sigma}_{\mathbf{x}}$ and $\mathbf{J}$ are known, this equation must be inverted in order to recover the needed covariance $\mathbf{\Sigma}_{\mathbf{T}}$. $\mathbf{J}$ being non square and full column rank for four corners ($\mathbf{J} \in \mathbb{R}^{8\times 6}$), we can use the right pseudo inverse $\mathbf{J}^+ = (\mathbf{J}^\top\mathbf{J})^{-1}\mathbf{J}^\top$. Multiplying left and right by respectively $\mathbf{J}^+$ and

$\mathbf{J}^{+,\top}$, we obtain: $\mathbf{\Sigma_T} = \mathbf{J}^{+}\mathbf{\Sigma_x}\mathbf{J}^{+,\top}$. Given that the pixel noise covariance is isotropic as explained above, $\mathbf{\Sigma_x} = \sigma_x^2\mathbf{I}$, this equation finally simplifies to:

$$\begin{aligned}\mathbf{\Sigma_T} &= (\mathbf{J}^\top\mathbf{J})^{-1}\mathbf{J}^\top\sigma_x^2\mathbf{I}((\mathbf{J}^\top\mathbf{J})^{-1}\mathbf{J}^\top)^\top \\ &= \sigma_x^2(\mathbf{J}^\top\mathbf{J})^{-1}\mathbf{J}^\top\mathbf{J}(\mathbf{J}^\top\mathbf{J})^{-1} \\ &= \sigma_x^2(\mathbf{J}^\top\mathbf{J})^{-1}. \end{aligned} \tag{4.12}$$

Note that the derivations above are not limited to a square tag with four corners and could be used for any object defined as a set of points in its local coordinate system provided their configuration is not degenerate and makes the PnP computation possible [Gao+03]. We have therefore a compact model informed by the geometry of the measurement model with a single tuning parameter in the pixel noise σ_x.

4.2.4 Covariance visualization

It is hard to represent graphically the appearance of this uncertainty model from the full 6D covariance. However, we can inspect the positional and orientational uncertainty (respectively the 3x3 upper-left and 3x3 lower-right sub-matrices) by representing them as confidence ellipsoids.

The principle is the following. For a 3D random variable $x \sim \mathcal{N}(\mu, \mathbf{\Sigma})$ described by a multivariate normal distributions with mean $\mu \in \mathbb{R}^3$ and covariance matrix $\mathbf{\Sigma} \in \mathbb{R}^{3\times3}$, the statistics:

$$(\mathbf{x} - \mu)^\top\mathbf{\Sigma}^{-1}(\mathbf{x} - \mu) = \chi^2 \tag{4.13}$$

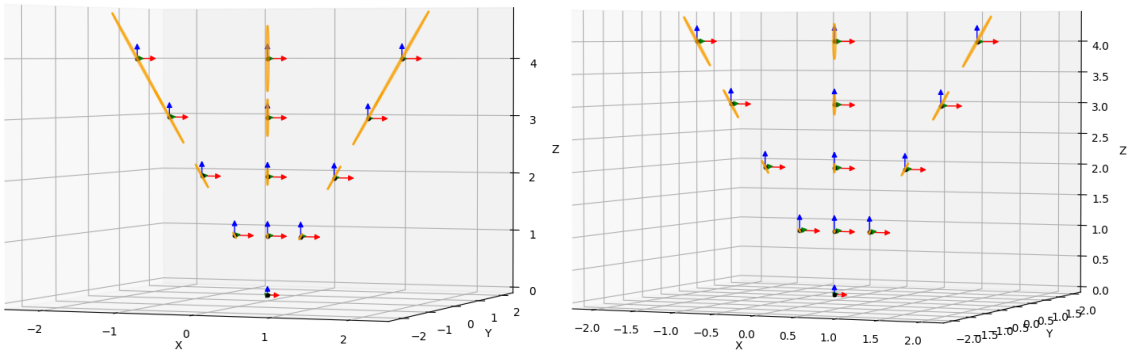
follows a chi-square distribution with 3 degrees of freedom. Since (4.13) is the equation of an ellipse centered at μ and shape governed by $\mathbf{\Sigma}$, we can then plot confidence volumes as ellipsoids by replacing the right-hand side of (4.13) by the critical value of the chi-square distribution for the desired confidence interval (we chose $\alpha = 99\%$).

We simulate a realistic scenario using calibration data from a laboratory RealSense camera, 15 cm tags and $\sigma_x = 2$ pixels. Fig. 4.4 and Fig. 4.5 show how the uncertainty on position and orientation relate to different factors.

In all cases, the covariance increases with the distance to the camera (the volume of the ellipses increases). On top of that, we can see that, for both position and orientation, the uncertainty distribution is clearly anisotropic: the uncertainty of the pose measurements is logically higher in the dimensions that lead to the least change in the aspect of the tag projection.

For the position uncertainty (Fig. 4.4), the shape of the covariance is easily interpretable: the uncertainty is the highest along the camera-tag axis. Besides, this uncertainty is not much influenced by the relative camera-tag orientation.

For the orientation uncertainty (Fig. 4.5), the shape of the covariance is harder to interpret: a high uncertainty about the orientation along one axis corresponds to a dilatation of the ellipse along this axis. For instance, in Fig. 4.5a, tags placed along the Z-axis of the camera frame have a very high uncertainty along the X and Y-axis (in camera frame). Their uncertainty ellipses seem, therefore, very flat along the Z-axis. Fig. 4.5a directly relates to our discussion about tag pose ambiguity in Section 4.2.2: tags close to fronto-parallel configuration (along the camera Z-axis) have a very high rotational uncertainty compared to tags that have a more important relative orientation as in Fig. 4.5a. On the



(a) Relative position covariances, 15 cm tags

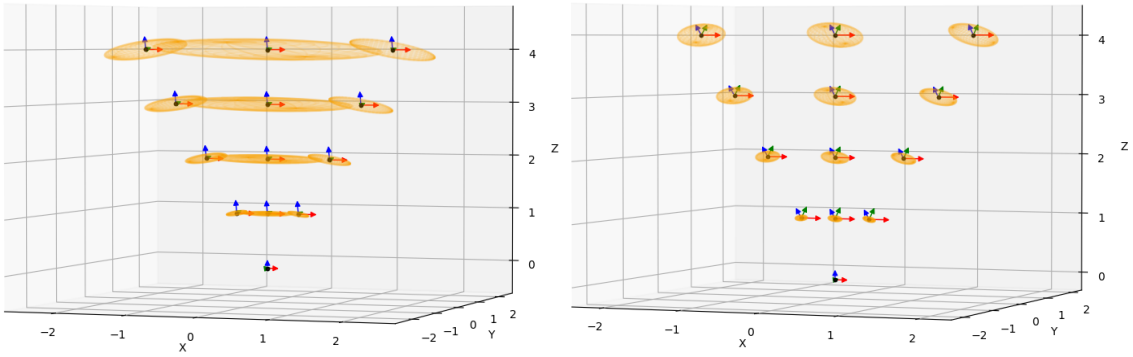
(b) Relative position covariances, 30 cm tags

Figure 4.4: Position uncertainty from the analytical covariance model. The single frame at the bottom represents the pose of the virtual camera, others are identical AprilTags placed in front of the camera at different distances but identical orientations. Covariances are represented in orange, centered at the corresponding tag position. (a) shows that the main positional uncertainty is along the camera-tag axis, which corresponds to the distance information. In (b), tags are twice the size, which shrinks their position uncertainty.

contrary, tags with higher relative orientation in Fig. 4.5a display a much lower rotational uncertainty.

In Fig. 4.6, 3 tags are projected: the first (black) is half a meter away from the camera in fronto-parallel orientation, the second (orange) 10 cm to the right, and the third (rose) 10 cm behind. We can see that the difference of appearance due to a depth-wise movement is much smaller, which explains the higher uncertainty on the depth measurement.

This covariance seems well behaved and captures the physical intuitions about the measurement uncertainties. This was leveraged in the context of a visual-inertial SLAM system implemented in Chapter 8. A quantitative comparison with [ULH16] would be interesting as well as an experimental validation campaign to further strengthen our results.



(a) Relative orientation covariances, 10° orientation (b) Relative orientation covariances, 45° orientation

Figure 4.5: Orientation uncertainty from the analytical covariance model. The single frame at the bottom represents the pose of the virtual camera, others are identical AprilTags placed in front of the camera at different distances but identical orientations. Covariances are represented in orange, centered at the corresponding tag position. In (a), tags are oriented to be nearly fronto parallel to the camera (10 degrees orientation), resulting in high orientation uncertainty due to the ambiguous perspective. (b) shows tags with the same positions but 45 degree orientations with respect to the camera, which results in lesser uncertainty.

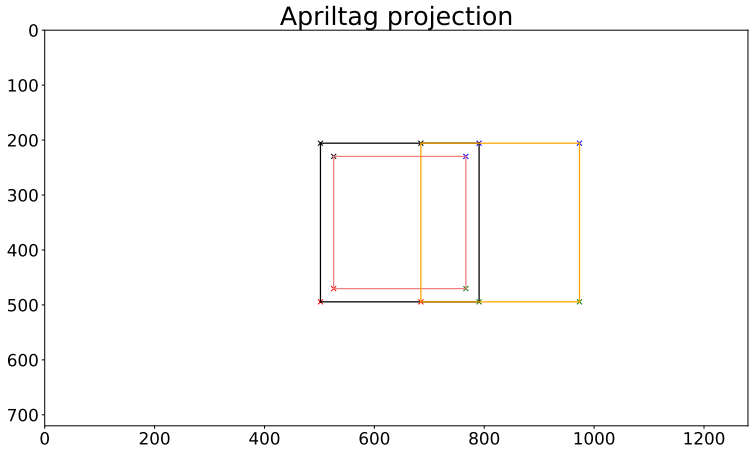


Figure 4.6: Projection of 3 tags of camera frame position: black -> (0,0,0.5), orange -> (0.1,0,0.5), rose -> (0.0,0,0.6)

4.3 Learning-based object pose estimation

In this section, we propose to integrate a deep-learning-based algorithm to object pose estimation [Lab+20] to our measurement models.

4.3.1 Object pose estimation

The problem of object pose estimation can simply be stated: given a calibrated camera and a set of known object models, detect those objects in the current image frame and extract camera-objects poses. Traditionally, competing methods were developed using either local features matching or learning-based methods. As shown by the most recent BOP challenge results [Hod+20], deep-learning methods now clearly dominate the field in terms of accuracy. At the time of the writing, the highest-ranking non-learning based pose extraction method [KD20] (object detection is deep learning based) actually uses depth measurements and is dominated by a purely RGB, deep-learning-based model [HB21], though [KD20] seems to be ten times faster [2].

In terms of accuracy, these results advocate for the use of deep-learning purely RGB image-based methods, that now provide accurate enough results for these to be used in robotics applications [Lab+21]. We use the CosyPose model of Labbe et al. [Lab+20] which at the time ranked first in the BOP challenge leaderboard. This approach mixes a new state-of-the-art single-view pose estimation algorithm with a multi-view algorithm using RANSAC and Bundle Adjustment. This system obtains precision in the order of centimeters on real objects whose 3D model is known. Its performances make it a good candidate as a direct 6D pose sensor to perform a multi-sensor fusion. In the context of legged robots, this is very useful to localize the robot relative to objects it needs to interact with, such as objects to manipulate or stairs to climb. While CosyPose is not yet able to generalize to unseen classes of objects, rapid progress is expected in this direction.

4.3.2 CosyPose

The front end of the SLAM system designed for this project includes object detection and object pose estimation. This function is provided by CosyPose [Lab+20], a deep-learning-based 6D pose estimator that reaches state-of-the-art performances for 6D object pose estimation. In the original paper, a single-view pose estimator and a multi-view algorithm were introduced. In our context, only the single-view module is used, while object tracking and mapping is handled by the SLAM framework.

CosyPose takes as input a single image I and a set of 3D models, each associated with an object label l. The camera C is assumed to be calibrated. A set of object detections is performed using the object detector Mask-RCNN [He+18]. Each 2D candidate detection is identified by an index α and associated with an object candidate O_α. Its 6D pose ${}^{C}\mathbf{T}_{O_\alpha} \in SE(3)$ relative to the camera C is then computed as follows.

The single-view procedure for pose estimation of CosyPose is an improvement over the one proposed in DeepIm [Li+19]. The general idea is to use the same neural network to iteratively converge to the object pose (Fig. 4.7). It takes as input the cropped image of the detected object bounding box and a rendered image based on the current object pose solution ${}^{C}\mathbf{T}_{O_\alpha,k-1}$ at iteration $k-1$. It returns a transformation update ${}^{O_\alpha,k-1}\mathbf{T}_{O_\alpha,k}$ that brings the rendered image closer to the cropped image. In practice, two neural networks

[2] BOP challenge leaderboard is available at https://bop.felk.cvut.cz/leaderboards/

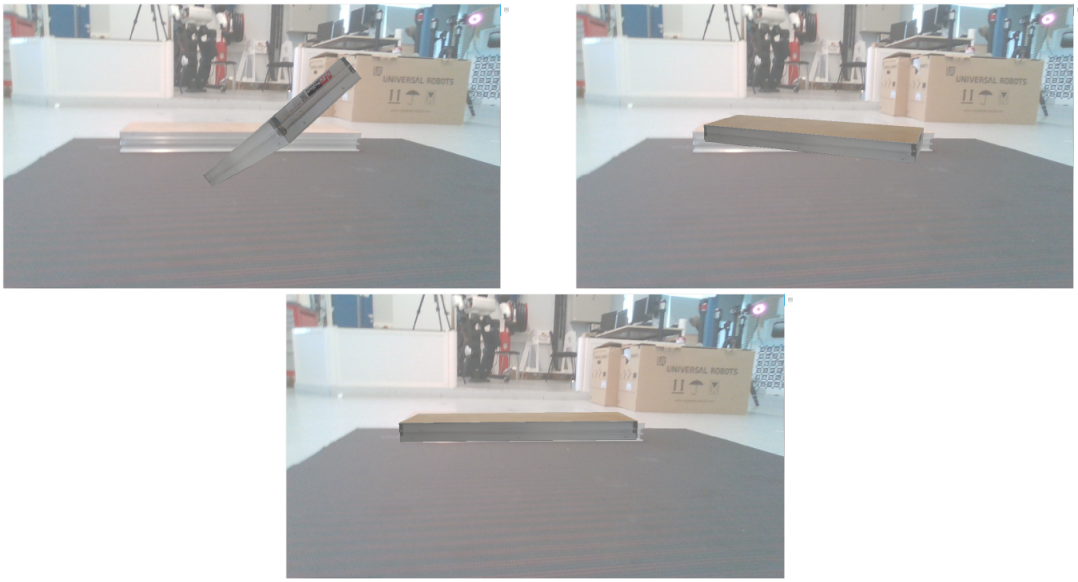


Figure 4.7: Progressive convergence of a stair step estimated pose over successive iterations of CosyPose.

with the same structure are trained independently: one for coarse pose estimation, *i.e.* the first iteration of the iterative process, and another for the refinement of the pose. The coarse network gives the first transformation ${}^{C}\mathbf{T}_{O_\alpha,0}$, and the prediction of the pose of the object is obtained by composing the N successive transformations:

$$ {}^{C}\mathbf{T}_{O_\alpha} = {}^{C}\mathbf{T}_{O_\alpha,0} \prod_{k=1}^{N} {}^{O_\alpha,k-1}\mathbf{T}_{O_\alpha,k}. \tag{4.14} $$

CosyPose reuses the neural network architecture of DeepIm with a new backbone for feature extraction, Efficient-Net [TL20a], with a spatial average pooling layer added after it. Then, it disables the optical flow sub-network during the training. A new rotation parametrization, introduced in [Zho+20], is used for the loss function, which has been shown to bring more stability during training. Then, the focal length of the cropped images is recomputed during training to fit the virtual camera of these images. Finally, the object symmetries are taken into account during training thanks to the *symmetric distance*. Each 3D model l is associated with a set of symmetries $S(l)$, that is the set of transformations that leave the aspect of the object unchanged:

$$ S(l) = \{\mathbf{S} \in SE(3) \mid \forall\, \mathbf{T} \in SE(3), \mathcal{R}(l, \mathbf{T}) = \mathcal{R}(l, \mathbf{TS})\} \tag{4.15} $$

where $\mathcal{R}(l, \mathbf{T})$ is the rendered image of the object l captured in pose M. Given a set of symmetries $S(l)$, the symmetric distance D_l measures the distance between two 6D poses represented by transformation $\mathbf{T}_1$ and $\mathbf{T}_2$. Given an object l associated to a set $\mathcal{X}_l$ of 3D points $\mathbf{x} \in \mathcal{X}_l$, we have:

$$ D_l(\mathbf{T}_1, \mathbf{T}_2) = \min_{\mathbf{S} \in S(l)} \frac{1}{|\mathcal{X}_l|} \sum_{\mathbf{x} \in \mathcal{X}_l} ||\mathbf{T}_1\mathbf{S}\mathbf{x} - \mathbf{T}_2\mathbf{x}||^2. \tag{4.16} $$

Equation (4.16) measures the average distance between the points of the object model transformed by $\mathbf{T}_1$ and $\mathbf{T}_2$ according to the symmetry that best aligns the transformed points. In practice, for continuous symmetries that are rotations around an axis (*e.g.* for a texture-less cylinder), $S(l)$ is discretized using 64 angles.

4.3.3 Empirical covariance estimation

When considering merging a tracker such as CosyPose with other sensor modalities, an important aspect is to predict covariance representing the level of confidence in the tracker estimate. Such data uncertainty is crucial to the robustness of SLAM systems involving neural networks subsystems such as [Yan+20a]. The depth-based object SLAM algorithm [Sal+13] claimed to compute a covariance matrix approximated as the inverse of the Iterative Closest Point output. The methods targeted for deep learning applications are harder to implement, especially if the goal is to use an off-the-shelf pose estimation neural network, as is the case for this paper. For instance, Bayesian Neural Networks [Jos+20] need to be trained explicitly for uncertainty prediction while Monte Carlo (MC) Dropout [GG16] requires multiple forward passes at run-time.

In our work CosySLAM [Deb+21], we present a practical implementation of a SLAM system based on the design of an off-the-shelf deep learning object pose estimation algorithm [Lab+20], which empirical results are presented in Chapter 10. To integrate these measurements with other sensors, we propose a noise model based on empirical data.

Covariance model

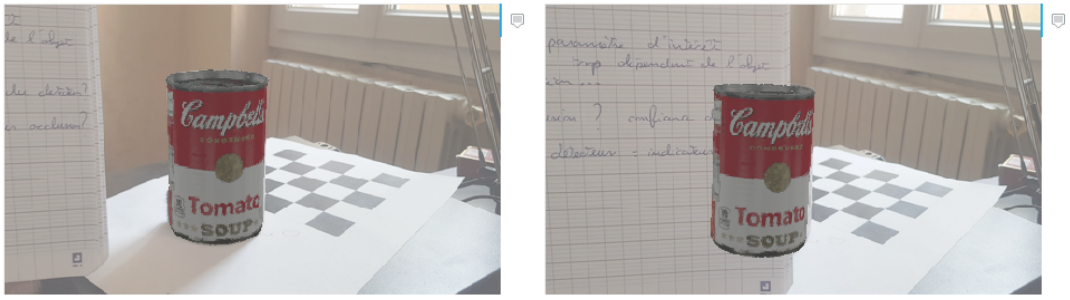
CosyPose does not provide an evaluation of its uncertainty. The two main families of solutions available to estimate uncertainties of neural network predictions consist in MC dropout [GG16] or Bayesian Neural Network (BNN) [Jos+20]. Using BNNs would require changing the architecture of CosyPose and retraining it. MC dropout would require several forward passes through the network for each iteration, which would be computationally expensive.

We need to compute the covariance without changing the architecture of the network and at an affordable cost. We propose to make an empirical error model based on polynomial regression in order to compute the covariance matrix. The idea is to parametrize the average error on each $\mathfrak{se}(3)$ component returned by CosyPose. We conducted an empirical study on several video sequences that explore the variations of the parameter set for several object types. The error is computed by comparing the $SE(3)$ transformations between the camera C and an object O returned by CosyPose ${}^{C}\mathbf{T}_{O}$ with the same transformation given by a motion capture system. We then use the error predictions of our parametric model as a proxy to the true 6 covariances during model fitting. A different model is fit for each object type due to their diversity of shapes, sizes, and textures.

The parameters used to compute the error need to represent the error sources of CosyPose as much as possible. To cover the error due to the configuration of the object in space, we need to include the 3D coordinates of the camera in the object frame. We also want to take into account some invariance that can occur by rotating around the object if it is texture-less for example. For this reason, we choose spherical coordinates of the camera origin with respect to the object frame to parametrize the model. Another source of error can be the occlusion of the object in the scene, as well as the motion blur, or any inherent noise in the image. This can affect the quality of the detection and of the pose estimation. Mask-RCNN returns a confidence score s for each detection that we will include in our model. This score is the output of the final softmax layer of the detector and can be interpreted as the level of confidence Mask-RCNN has in its prediction.

To sum up, our model is parameterized by four values: r - distance, φ - azimuth, θ - elevation, s - Mask-RCNN softmax output.

We can then compute the error of CosyPose relative to the motion capture data:



(a) $s = 0.999$ (b) $s = 0.996$

Figure 4.8: Example of values of the Mask-RCNN softmax output for various levels of occlusion. These shows the impressive robustness of the CosyPose to partial occlusions.

$$\mathbf{e} = [\mathbf{e}_t, \mathbf{e}_a] \triangleq \left[{}^{O}\widehat{\mathbf{p}}_C - {}^{O}\widetilde{\mathbf{p}}_C,\ \mathrm{Log}\left({}^{O}\widehat{\mathbf{R}}_C^{-1}\, {}^{O}\widetilde{\mathbf{R}}_C \right) \right] \tag{4.17}$$

where $\widehat{\cdot}$ and $\widetilde{\cdot}$ denote quantities obtained from the motion capture system and CosyPose respectively. Log denotes the logarithm application mapping elements of SO(3) to the $\mathbb{R}^3$ representation of its Lie algebra $\mathfrak{so}(3)$.

We want to find a polynomial function $f(r, \varphi, \theta, s) \in \mathbb{R}^6$ that returns the error given the set of training data $\{X, E\}$. For each object, we capture a set of video sequences and we compute the error with the motion capture data for each measurement. We then perform polynomial regression implemented with Scikit Learn [Ped+11]. A simple linear regression leads to a high root-mean-square error (RMSE) on a test dataset. Over degree 3, the model overfits and the high curvature of the polynomial returned high error values outside of the training data range. Thus, a degree 2 polynomial regression seemed to offer a good compromise. Quantitative results are given in the experimental validation section (see Fig. Fig. 4.9 for a few examples of fitted polynomials).

This empirical model has the drawback that it needs to be trained (ideally with real data) for each new object. If possible, the covariance model should directly result from CosyPose nominal training. We started to investigate this direction (with the MSc thesis of Cesar Debeunne) but acknowledged it as an open perspective.

Empirical covariance

As explained in Sec. II, we have trained empirical models to evaluate the covariance of the estimation of CosyPose. To validate these models, we propose to exhibit a few intuitive observations and quantitative statistical analysis. One of the parameters involved in our model is the absolute distance between the camera and the object, noted r. Our trained models show an expected behavior regarding this parameter: the global error increases when the camera moves away from the object.

Fig. 4.9 sheds light on these phenomena and gives an explicit comparison between the models of different objects. The error of the object from the YCBV dataset seems more stable and smaller than the one of the objects from the T-LESS dataset. This can be explained by the texture of the object and the absence of symmetries: T-LESS objects are known to be more challenging for pose estimation and this is confirmed by our model.

A more quantitative evaluation can be deduced from Table 4.2. The translation error seems to be captured pretty well, as the RMSE is around the centimeter. However, the angular error seems a little less predictable, especially for T-LESS objects whose orientation

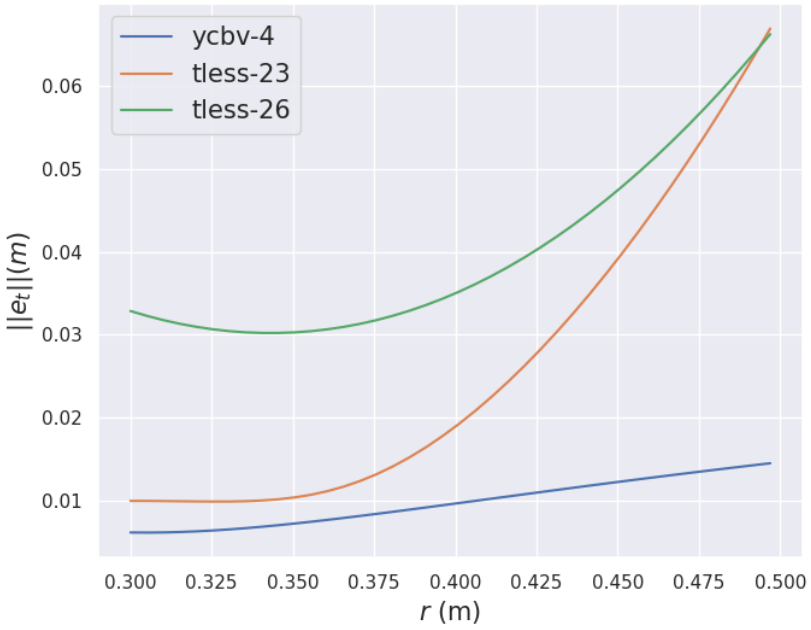


Figure 4.9: The norm of the translation error returned by the models of three different objects with respect to r, the distance between the camera and the object. The other parameters θ, ϕ and s are fixed to the average values of our training data.

estimation can suffer from an important measurement noise due to the lack of textures. The $R^2 \in [0, 1]$ score is the coefficient of determination and is often used to evaluate statistical models. Its interpretation is subject to debate and cannot conclude by itself on the absolute quality of the model. However, a score higher than 0 demonstrates that the model is more accurate than a baseline average model.

Table 4.2: A quantitative evaluation of our models, these values are computed on test samples that were not used for training.

	R^2	RMSE ang. err. (°)	RMSE trans. err. (cm)
YCBV-4	0.55	5,1	0.6
T-LESS-23	0.5	11.7	1.5
T-LESS-26	0.68	22	0.6

4.3.4 Retraining with stairs

In order to produce a realistic SLAM scenario in the context of legged robots, we retrained CosyPose with staircases present in our lab. We made a textured mesh of a stair step used in our experimental platform. This textured mesh was used both for training and using the trained model. The generation of photorealistic synthetic data was handled by BlenderProc [Den+19]. The render-and-compare loop uses PyBullet [CB21].

We generate 10.000 synthetic images that are labeled with object pose ground truths. We retrain the three modalities of CosyPose: the Mask-RCNN detector, the coarse pose estimator, and the pose refiner. We slightly tune the training parameters used for the object

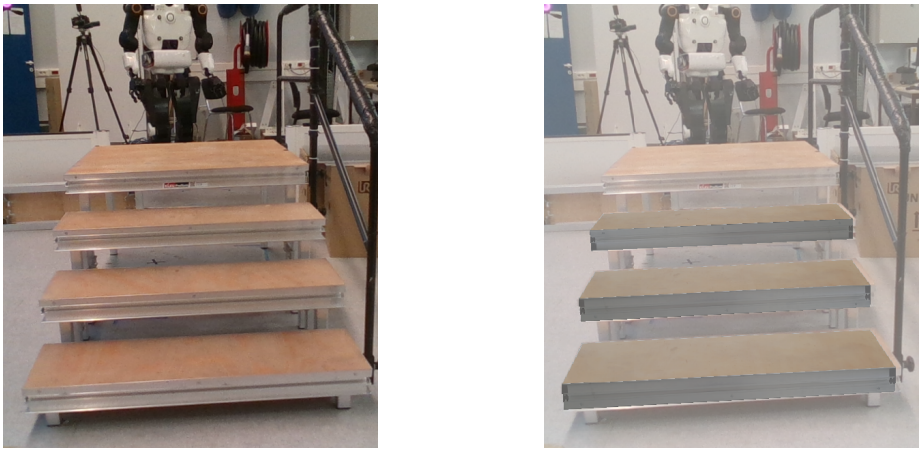


Figure 4.10: On the left, a picture of our climbing module at LAAS and on the right the 3D model of the stair projected according to CosyPose measurements on the same picture

from the BOP challenge as the scale is different: the stair is $1m \times 0.3m \times 0.07m$ whereas a T-LESS object is never wider than $10cm$. Thus, we generate training data on $10m \times 10m$ scenes, and we increase the noise used to train the refiner.

An illustration of the performances of CosyPose on a set of 3 stair steps is given on Fig. 4.10: the pose of each step is here independently estimated which leads to local accuracy but global inconsistency.

4.4 Conclusion

In this chapter, we have introduced our 2 first practical factors. Both handle a direct relation between the camera and an object of the scene, hence being related to semantic (or object-level) SLAM. The first factor, using fiducial markers, mostly has a practical interest and will be used in the first experimental setup to validate the IMU factors. It mostly boiled down to defining a new uncertainty model, that we were able to analytically derive directly from the PnP algorithm. The singularity of the 4 point projections are simply handled by rejecting rotation information when it is detected. The same factor is then extended to handle a complete object pose estimator, that we have implemented on top of the award-winning CosyPose detector. Here also we have tried to directly derivate the uncertainty model from the algorithm (by training a neural network able to predict both the mean and the covariance), but finally relied on an empirical model trained a posteriori. Beyond the practical interest of this new perception feature inserted in the MAP framework, this second work also contributed to the domain of neural-based object pose estimation, by proposing a sound and generic way of sequential estimation, i.e. aggregating measurements across time and fusing them with other sensors (IMU, etc) to improve the estimation quality. We will experimentally show in the last part of this thesis the importance of this temporal sensor fusion, as CosyPose alone, despite the quality of the algorithm, is not able to provide enough performance guarantee to be embedded in a robot feedback loop.

This chapter paves the way to extending our work to true semantic SLAM, *i.e.* mapping while directly extracting the location of an object of interest. With CosyPose, we directly identify the accurate pose of known objects. This is important in our context for two reasons. First, the recognized objects can later be used to close the localization loop

with a high degree of accuracy. Second, the object can be used (*e.g.* in a motion planner) even if out of the perception range of the robot. A direct practical use case is for the legged robot to step on an object that is typically not in the field of view (because legged robots typically do not have cameras pointing on their feet). While the inclusion on the MAP framework seems satisfactory in practice, some interesting theoretical work might focus on the insertion of raw information extracted from the trackers, instead of the overall object pose. This would avoid possible singularities to invalidate the complete pose information while partial information might benefit the MAP estimator (as we already discussed in the practical case of the PnP algorithm for fiducial markers), or even including latent information directly in the MAP. It is also important to work on a more systematic way of evaluating the uncertainty, in particular for neural-based estimators. Finally, some recent work on extending object pose estimator to classes of objects, or even to unseen objects, open very interesting directions to extend this work to semantic SLAM.

Chapter 5

Kinematics

Contents

The robot kinematic model along with encoder measurements make possible the computation of poses and spatial velocities [Fea14] (aka. twists) of reference frames attached to the robot segments relative to the base or world frame. This kinesthesis capability is of paramount importance to control multi-body systems for locomotion or other interactions with their environment. If we have information about stable contacts that the robot keeps with its environment, it is possible to infer a relative motion measurement, which is commonly called leg-odometry.

We will shortly describe the formulation of the forward-kinematics algorithm, that enables to compute relative poses and spatial velocities between parts of the robot and then show how this information can be used to define a leg-odometry factor to be added in the factor graph. The last section shows how forward-kinematics can also be used to leverage prior information about the height of the environment.

5.1 Forward kinematics

We first describe the forward-kinematics and differential kinematics algorithms by introducing notations commonly found in the legged robots literature.

The degrees of freedom (DoF) n of a poly-articulated system is the minimal number of variables that completely describe his state given the base frame. For robots with rigid segments, the DoF of the robot is the sum of the DoFs of its joints (1 for linear and rotational joints, 3 for ball joints, etc.). The state of the joints (1 angle for rotational joints) can be stacked together in the vector of joint configurations $\mathbf{q}_a \in \mathbb{R}^n$. For most simple joints, the joint configuration velocities are simply the time derivative $\dot{\mathbf{q}}_a \in \mathbb{R}^n$[1]. $\mathbf{q}_a$ and $\dot{\mathbf{q}}_a$ are typically obtained from the joint encoder measurements. Legged robots

[1] An exception is the ball joint where the state is defined as an element of $SO(3)$, thus its velocity vector is defined as an angle axis in $\mathbb{R}^3$.

are not fixed to the ground so we add a so-called "free-flyer" DoF which represents the free pose of the base with respect to the world and is modeled as an unactuated joint of 6 DoF pertaining to the $SE(3)$ group. Its velocities are therefore in the $\mathfrak{se}(3)$ Lie algebra, isomorphic to $\mathbb{R}^6$.

The whole-body state of the robot is defined by the robot configuration vector $\mathbf{q} = [\mathbf{q}_b, \mathbf{q}_a] \in SE(3) \times \mathbb{R}^n$, where

$$\mathbf{q}_b \triangleq {}^W\mathbf{T}_B = \begin{pmatrix} {}^W\mathbf{R}_B & {}^W\mathbf{p}_B \\ \mathbf{0}_3^\top & 1 \end{pmatrix} \tag{5.1}$$

is the configuration of the base, with ${}^w\mathbf{p}_b \in \mathbb{R}^3$ and ${}^W\mathbf{R}_B \in SO(3)$ are the position and orientation of the base with respect to the world. $\mathbf{q}_a \in \mathbb{R}^n$ is the configuration of the joints. $\mathbf{q}$ therefore belongs to a composite manifold $\mathcal{M}_\mathbf{q} = \langle SE(3), \mathbb{R}^n \rangle$ and the configuration velocity vector belongs to its tangent space $\mathfrak{se}(3) \times \mathbb{R}^n$ which is isomorphic to $\mathbb{R}^{6+n}$: $\mathbf{v}_q \in \mathbb{R}^{6+n}$. Those are concatenations of the base state and joint configuration vectors:

$$\mathbf{q} = [\mathbf{q}_b, \mathbf{q}_a] \in \mathcal{M}_\mathbf{q} \qquad , \qquad \mathbf{v}_q = [{}^b\nu_b, \dot{\mathbf{q}}_a] \in \mathbb{R}^{6+n} \tag{5.2}$$

where ${}^b\nu_b$ is the twist of the base frame expressed in the base frame. The computation of any segment pose of index k relative to the world is obtained using the forward-kinematics (FK) algorithm:

$${}^W\mathbf{T}_K = \text{FK}_k(\mathbf{q}). \tag{5.3}$$

The kinematic chains of most robots can be represented as a tree whose root is the world frame, nodes are the joints to which are associated reference frames (including the base, which is considered as a $SE(3)$ joint), and edges are solid segment placed between these nodes. The forward-kinematics algorithm consists of a forward pass from the root of the tree ${}^W\mathbf{T}_B$ through the joints along the path to the leaf ${}^W\mathbf{T}_K$ that we wish to obtain. Let us denote by i the index of each intermediate joint between the root and the leaf, the root (world) having index 0 and the base index 1, the forward-kinematics algorithm simply writes as a composition of successive transformations:

$${}^0\bar{\mathbf{T}}_k = {}^0\mathbf{T}_1(\mathbf{q}_b)\, {}^1\bar{\mathbf{T}}_{2'}\, {}^{2'}\mathbf{T}_2(q_2) \ldots {}^{k-1}\bar{\mathbf{T}}_{k'}\, {}^{k'}\mathbf{T}_k(q_k) \tag{5.4}$$

where ${}^{i-1}\bar{\mathbf{T}}_{i'}$ designates the transformation between joint frames $i-1$ and i when the joint configuration q_i is at the identity value and is a constant. The ${}^{i'}\mathbf{T}_i(q_i)$ designate transformations caused by a non-zero joint value q_i, as reprsented in Fig. 5.1.

It is also possible to obtain the pose relative to the base ${}^b\mathbf{T}_k$ by simply setting the base pose vector to be the identity pose $[0, 0, 0,\ 0, 0, 0, 1]$.

Similarly, the spatial velocity relative to the base ${}^b\nu_k$ can be obtained by setting the base spatial velocity part of $\mathbf{v}_q$ to zero. These algorithms are very fast to compute ($\approx 1\mu s$) using modern dedicated libraries [Car+19].

We will now see how FK can be used to derive leg-odometry for a quadruped robot.

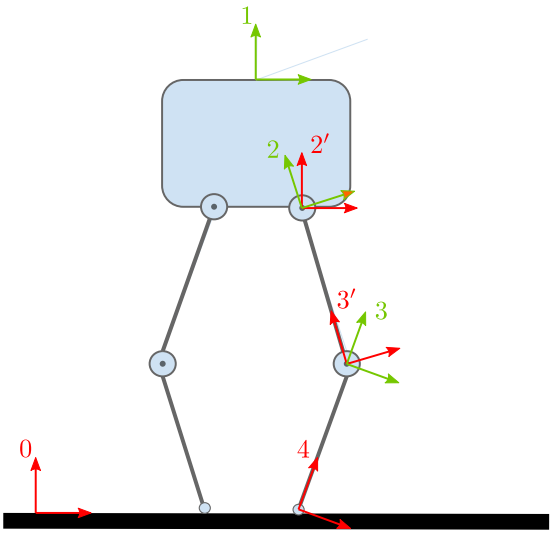


Figure 5.1: Illustration of the Forward kinematics algorithm to get the transformation of the foot 4 with respect to the world 0.

5.2 Leg odometry

We will restrict ourselves to the case of point feet as the main platform on which we experimented is a quadruped robot. Quadrupeds are usually equipped with non-articulated round feet whose contact with the ground is, in first approximation, punctual. Humanoid robots are equipped with articulated flat feet that provide richer information. In Section 2.2.1 we gave an exhaustive review of the types of leg odometry measurements that can be obtained on legged machines.

The basic idea is that we assume to have access to accurate contact detection. If the contact is held between time t_i and t_j, the contact point L (to which a frame is attached) is fixed. This can be written ${}^{W}\mathbf{p}_L^i = {}^{W}\mathbf{p}_L^j$. In practice, the round feet of our robot can slightly roll but we will neglect this phenomenon. If we unfurl the transformation chain to make the base poses appear in the equation, we obtain the identity:

$$\mathbf{p}^i + \mathbf{R}^i\,{}^{B}\widetilde{\mathbf{p}}_L^i = \mathbf{p}^j + \mathbf{R}^j\,{}^{B}\widetilde{\mathbf{p}}_L^j \tag{5.5}$$

We can immediately derive the residual $\mathbf{e}^{LO}$ expression for each foot l in contact between t_i and t_j:

$$\mathbf{e}_l^{LO}(\mathbf{p}^i, \mathbf{R}^i, \mathbf{p}^j, \mathbf{R}^j) = \mathbf{p}^i + \mathbf{R}^i\,{}^{B}\widetilde{\mathbf{p}}_L^i - (\mathbf{p}^j + \mathbf{R}^j\,{}^{B}\widetilde{\mathbf{p}}_L^j) \tag{5.6}$$

To obtain a proper measurement model, we need to model the influence of measurement inaccuracies. This equation depends on relative positions ${}^{B}\widetilde{\mathbf{p}}_L^i$ and ${}^{B}\widetilde{\mathbf{p}}_L^j$ that are computed using FK. The quality of the computation depends on several things: the robot kinematic model (calibration), deviations from the rigid body model (*e.g.* segment/joint flexibilities, backlash), encoder noise. It is tempting to propagate encoder noise $\mathbf{n}_{\mathbf{q}_a}$ through FK using the kinematic Jacobian [Blo+13b; Har+18c]:

$${}^{B}\widetilde{\mathbf{p}}_L = FK(\mathbf{q}_a + \mathbf{n}_{\mathbf{q}_a}) \approx FK(\mathbf{q}_a) + \mathbf{J}_L\mathbf{n}_{\mathbf{q}_a}, \qquad \mathbf{\Sigma}_p = \mathbf{J}_L\mathbf{\Sigma}_{\mathbf{q}_a}\mathbf{J}_L \tag{5.7}$$

However, the resolution of Solo encoders is very high: 0.002 degrees, which would account for a mere 10 *micrometers* difference. Therefore, most of the measurement errors come from the other mentioned effects. Unfortunately, those are much harder to model, and all act at the same time in an unpredictable fashion. Thus, we opt for a simple additive noise on the foot position in the world:

$$ {}^{W}\mathbf{p}_L^i = {}^{W}\mathbf{p}_L^j + \mathbf{n}_{LO} \tag{5.8} $$

where $\mathbf{n}_{LO} \sim \mathcal{N}(0, \mathbf{\Sigma}_{LO})$ denotes a Gaussian noise accounting for potential roll/slip of the foot and kinematic inaccuracies. This noise can be modeled as white noise on the foot velocity $\mathbf{n}_v$, which when integrated gives a random walk. Its variance after Δt is

$$ \mathbf{\Sigma}_{LO} = \Delta t \mathbf{\Sigma}_v \tag{5.9} $$

where ${}^{B}\tilde{\mathbf{p}}_L^i$ and ${}^{B}\mathbf{p}_L^j$ are the contact positions in base frame at times t_i and t_j acquired from $\mathbf{q}_a$ via forward-kinematics. The residual errors $\mathbf{e}_l^{LO}$ and their covariances can be added as factors to the factor graph that we wish to build.

In the literature review and Fig. 2.3, this method is referred to as *single foot matching*. To obtain a relative transformation of the robot's base frame between t_i and t_j, at least three stable feet contacts are necessary. When it is the case, it is possible to directly compute the relative transformation, which would be integrated as a separate residual, by solving an orthogonal Procrustes problem [RK91].

However, for trotting gates common for quadrupeds, the robot is most of the time standing on only two legs, which makes the Procrustes problem computation rarely applicable. Instead, we add a single factor for each foot in stable contact. In fact, when at least 3 feet are in stable contact, the information extracted from these combined factors is the same as if we had pre-computed a 6D transformation from the Procrustes problem. Besides, when fusing with other sensors (such as the IMU, see Chapter 9), 1 or 2 legs in contact already provide sufficient information to make the desired variables observable [Blo+13b].

5.3 Terrain height

If we only use leg-odometry along with other sources of odometry such as an IMU (see Chapter 9, the position of the robot drifts after some time. However, if we have prior knowledge about the foot terrain height $h \in \mathbb{R}$, it is easy to define, for each foot l in contact, the residual:

$$ \mathbf{e}_l^h(\mathbf{p}, \mathbf{R}) = (\mathbf{p} + \mathbf{R}\,{}^{b}\mathbf{p}_l)_z - h \quad , \quad e_l^h \in \mathbb{R} \tag{5.10} $$

This residual can be assigned with a variance σ_h^2 which is hand-tuned. The error and its covariance form a unary factor (only depends on one Keyframe) that can be added to the factor graph we want to build. Giving information about the height of the feet of the robot grounds the robot base height estimates and cancels the accumulation of drift in the vertical direction.

5.4 Conclusion

We have shown that leg odometry easily enters the MAP framework, using directly the FK algorithms. This is a cheap and efficient way to express the meaningful contact in-

formation in the factor graphs. As discussed in the state of the art, there are multiple ways of formulating this information as an estimation factor. We have discussed various possibilities and show why we have chosen the final formulation. There remains some space to formally analyze this choice, possibly leading to a better formulation. We will see in the last part of this thesis how it performs in practice. Here we only considered the contact information as a motion constraint. We will consider the resulting forces, ideally measured at the contact interface with a direct force measurement device, to simultaneously estimate the basis state and the centroidal quantities. But before that, we need to introduce the mathematics of pre-integration, which we will first introduce in the context of the inertial measurements.

Chapter 6

High-rate motion data pre-integration

Contents

This chapter presents a abstraction of the pre-integration, originally introduced in [LS09; For+15]. We will explain the interest of this generalization that we in part developed with Dinesh Atchuthan at the end of his Ph. D.

Sensor modalities available on robotic platforms display a great variety in the rate at which measurements are acquired. For instance, a camera may record images at 33Hz while an IMU may be updated at 1 kHz. In the context of Factor Graph estimation, this creates a challenge since residuals are defined between Keyframes selected at a relatively low frequency, at times as low as 1 Hz. One needs to integrate measurements from these high-rate sensors between Keyframes, which is not trivial in general.

In this chapter, we first describe the example of the integration of IMU measurements which motivates the development of the pre-integration theory [LS09; For+15]. We then express this pre-integration in more abstract mathematical terms to generalize it to other high-rate sensors, which was first described in Dinesh Atchuthan's thesis [Atc18]. Then, we describe a possible reformulation of the original IMU-pre-integration algorithm [For+17] by exploiting a new Lie group that we proposed in [Fou+19]. We conclude the chapter with a discussion of related works.

6.1 A motivational example: IMU integration for graph optimization

In this section, we will introduce the IMU measurement model and explain why a naive integration of these measurements in the world frame does not immediately lead to a viable factor in practice. We will then introduce the observations that lead to the development of the so-called IMU pre-integration algorithm by Lupton [LS09], which we reformulated with what we hope to be a more systematic approach. This systematic formulation will then be used in the following chapters to account for force measurements.

Let us consider the estimation of a robot base state comprised of its pose and velocity in the world frame,

$$\mathbf{x} = [{}^W\mathbf{p}_{WB}, {}^W\mathbf{v}_{WB}, {}^W\mathbf{R}_B] \triangleq [\mathbf{p}, \mathbf{v}, \mathbf{R}]. \tag{6.1}$$

We make a few hypotheses. First, we neglect effects due to the rotation of the Earth by assuming that our world frame (which is fixed with respect to the ground) is an inertial frame. This is a common simplification in robotics scenarios [For+17]. Second, without loss of generality, we assume that the IMU frame is identical to the base frame in the following developments.

The IMU measurements are known to be noisy, biased, and affected by the gravity,

$$\begin{aligned} \tilde{\boldsymbol{\omega}} &= {}^B\boldsymbol{\omega}_{WB} + \mathbf{b}_{\boldsymbol{\omega}} + \mathbf{n}_{\boldsymbol{\omega}} \\ \tilde{\mathbf{a}} &= {}^B\mathbf{a}_{WB} + \mathbf{b}_{\mathbf{a}} + \mathbf{n}_{\mathbf{a}} + {}^B\mathbf{R}_W\mathbf{g}. \end{aligned} \tag{6.2}$$

The biases $\mathbf{b} \triangleq [\mathbf{b}_{\mathbf{a}}, \mathbf{b}_{\boldsymbol{\omega}}]$ need to be estimated in order to be canceled and are thus included in the estimator state. Fusing with other sensors will help make them observable.

These biases may drift over time, more or less slowly depending on the quality of the IMU. This drift is modeled as a random walk, which is close to the observed behavior for reasonably short periods of time [HJ15].

A continuous-time dynamical model based on strap-down integration of IMU measurements can then be derived:

$$\begin{aligned} \dot{\mathbf{p}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= {}^W\mathbf{a}_{WB} \\ \dot{\mathbf{R}} &= \mathbf{R}[{}^B\boldsymbol{\omega}_{WB}]_\times \\ \dot{\mathbf{b}}_{\mathbf{a}} &= \mathbf{w}^c_{\mathbf{a}} \\ \dot{\mathbf{b}}_{\boldsymbol{\omega}} &= \mathbf{w}^c_{\boldsymbol{\omega}}, \end{aligned} \tag{6.3}$$

where $\mathbf{w}^c_{\mathbf{a}} \in \mathcal{N}(0, \mathbf{W}^c_{\mathbf{a}})$ and $\mathbf{w}^c_{\omega} \in \mathcal{N}(0, \mathbf{W}^c_{\omega})$ are the biases' random walk continuous-time white noises.

Introducing the measurement equations (6.2) in the continuous dynamics equations (6.3) and using a zero order hold explicit Euler integration scheme during δt results in the discrete-time dynamics:

$$\begin{aligned} \mathbf{p}^{k+1} &= \mathbf{p}^k + \mathbf{v}^k\delta t + \frac{1}{2}\mathbf{g}\delta t^2 + \frac{1}{2}\mathbf{R}^k(\tilde{\mathbf{a}}^k - \mathbf{b}^k_{\mathbf{a}} - \mathbf{n}^k_{\mathbf{a}})\delta t^2 \\ \mathbf{v}^{k+1} &= \mathbf{v}^k + \mathbf{g}\delta t + \mathbf{R}^k(\tilde{\mathbf{a}}^k - \mathbf{b}^k_{\mathbf{a}} - \mathbf{n}^k_{\mathbf{a}})\delta t \\ \mathbf{R}^{k+1} &= \mathbf{R}^k\,\mathrm{Exp}((\tilde{\boldsymbol{\omega}}^k - \mathbf{b}^k_{\boldsymbol{\omega}} - \mathbf{n}^k_{\boldsymbol{\omega}})\delta t) \\ \mathbf{b}^{k+1}_{\mathbf{a}} &= \mathbf{b}^k_{\mathbf{a}} + \mathbf{w}_{\mathbf{a}} \\ \mathbf{b}^{k+1}_{\boldsymbol{\omega}} &= \mathbf{b}^k_{\boldsymbol{\omega}} + \mathbf{w}_{\boldsymbol{\omega}} \end{aligned} \tag{6.4}$$

where $\mathbf{w_a} \in \mathcal{N}(0, \mathbf{W_a})$ and $\mathbf{w}_{\boldsymbol{\omega}} \in \mathcal{N}(0, \mathbf{W}_{\boldsymbol{\omega}})$ are the bias' random walk discrete-time white noises, satisfying $\mathbf{W_a} = \mathbf{W}_{\mathbf{a}}^c \delta t$ and $\mathbf{W}_{\boldsymbol{\omega}} = \mathbf{W}_{\boldsymbol{\omega}}^c \delta t$.

Now, these equations relate consecutive states with data sampled at IMU frequency. To include these measurements in our smoothing estimator, one solution would be to introduce new states at the rate of the IMU. However, the size of the problem would grow very quickly due to the high acquisition rate of IMUs. A better option is to integrate IMU measurements during extended periods, that is between Keyframes. If we simply integrate the sequence of IMU measurements $\mathcal{Z}_{im}$ between timestamps t_i and t_m by recursively applying (6.4), we obtain:

$$\begin{aligned}
\mathbf{p}^m &= \mathbf{p}^i + \sum_{k=i}^{m} \left[\mathbf{v}^k \delta t + \frac{1}{2} \mathbf{g} \delta t^2 + \frac{1}{2} \mathbf{R}^k (\tilde{\mathbf{a}}^k - \mathbf{b}_{\mathbf{a}}^k - \mathbf{n}_{\mathbf{a}}^k) \delta t^2 \right] \\
\mathbf{v}^m &= \mathbf{v}^i + \sum_{k=i}^{m} \left[\mathbf{g} \delta t + \mathbf{R}^k (\tilde{\mathbf{a}}^k - \mathbf{b}_{\mathbf{a}}^k - \mathbf{n}_{\mathbf{a}}^k) \delta t \right] \\
\mathbf{R}^m &= \mathbf{R}^i \prod_{k=i}^{m} \mathrm{Exp}((\tilde{\boldsymbol{\omega}}^k - \mathbf{b}_{\boldsymbol{\omega}}^k - \mathbf{n}_{\boldsymbol{\omega}}^k) \delta t)
\end{aligned} \tag{6.5}$$

where we assume that IMU biases stay constant during Δt_{im}

$$\mathbf{b}^k \approx \mathbf{b}^i \quad \forall k \in [i..m],$$

which makes their integration trivial and thus allows us to concentrate in the upper three lines of the model.

We are here in the position of illustrating why a naive definition of data integration leads to very bad performance. By observing (6.5) we can define a motion error, as follows. First, we naively define the motion increments or "deltas" in two ways. From the integration of the motion model,

$$\boldsymbol{\Delta}_{im}(\mathbf{b}^i, \mathbf{x}^i) = \begin{bmatrix} \boldsymbol{\Delta}\mathbf{p}_{im} \\ \boldsymbol{\Delta}\mathbf{v}_{im} \\ \boldsymbol{\Delta}\mathbf{R}_{im} \end{bmatrix} \triangleq \begin{bmatrix} \sum_{k=i}^{m} \left[\mathbf{v}^k \delta t + \frac{1}{2} \mathbf{g} \delta t^2 + \frac{1}{2} \mathbf{R}^k (\tilde{\mathbf{a}}^k - \mathbf{b}_{\mathbf{a}}^i) \delta t^2 \right] \\ \sum_{k=i}^{m} \left[\mathbf{g} \delta t + \mathbf{R}^k (\tilde{\mathbf{a}}^k - \mathbf{b}_{\mathbf{a}}^i) \delta t \right] \\ \prod_{k=i}^{m} \mathrm{Exp}((\tilde{\boldsymbol{\omega}}^k - \mathbf{b}_{\boldsymbol{\omega}}^i) \delta t) \end{bmatrix} \tag{6.6}$$

and from the difference between initial and final states:

$$\hat{\boldsymbol{\Delta}}_{im}(\mathbf{x}^i, \mathbf{x}^m) = \begin{bmatrix} \hat{\boldsymbol{\Delta}}\mathbf{p}_{im} \\ \hat{\boldsymbol{\Delta}}\mathbf{v}_{im} \\ \hat{\boldsymbol{\Delta}}\mathbf{R}_{im} \end{bmatrix} \triangleq \begin{bmatrix} \mathbf{p}^m - \mathbf{p}^i \\ \mathbf{v}^m - \mathbf{v}^i \\ \mathbf{R}^{i,T} \mathbf{R}^m \end{bmatrix}. \tag{6.7}$$

Then, we build a residual error as

$$\mathbf{e}(\mathbf{x}^i, \mathbf{x}^m, \mathbf{b}^i) = \boldsymbol{\Delta}_{im}(\mathbf{b}^i, \mathbf{x}^i) \ominus \hat{\boldsymbol{\Delta}}_{im}(\mathbf{x}^i, \mathbf{x}^m) = \begin{bmatrix} \boldsymbol{\Delta}\mathbf{p}_{im} - \hat{\boldsymbol{\Delta}}\mathbf{p}_{im} \\ \boldsymbol{\Delta}\mathbf{v}_{im} - \hat{\boldsymbol{\Delta}}\mathbf{v}_{im} \\ \mathrm{Log}(\hat{\boldsymbol{\Delta}}\mathbf{R}_{im}^{-1} \boldsymbol{\Delta}\mathbf{R}_{im}) \end{bmatrix}. \tag{6.8}$$

where $\ominus$ is the composite manifold lift operator defined in (3.26).

$\hat{\boldsymbol{\Delta}}_{im}(\mathbf{x}^i, \mathbf{x}^m)$ only depends on state variables and, thus, is cheap to compute. It corresponds to the "expected" motion of the system, given the current state estimates. $\boldsymbol{\Delta}_{im}(\mathbf{b}^i, \mathbf{x}^i)$ is the motion computed from the integration of (very many) IMU measurements during Δt_{im}. However, since we "naively" integrated in the world frame, this term

also depends on the initial state $\mathbf{x}_i$ and on IMU bias $\mathbf{b}_i$. This implies that for each update of the estimate $\mathbf{x}_i$, that is for every iteration of the MAP solver, the IMU measurements need to be re-integrated from the new $\mathbf{x}_i$ and for the new $\mathbf{b}^i$. This is highly inefficient and, therefore, not well adapted to the repeated evaluations required by nonlinear solvers.

To solve this problem, we need to re-define the deltas so that $\boldsymbol{\Delta}_{im}$ is independent of the estimated states, that is, only dependent on the measured data. This new definition reads [LS09; For+15] (proof in the annex Section B.1):

$$\boldsymbol{\Delta}_{im}(\mathbf{b}^i) \triangleq \begin{bmatrix} \sum_{k=i}^{m} \left[\boldsymbol{\Delta}\mathbf{v}^{ik}\delta t + \frac{1}{2}\boldsymbol{\Delta}\mathbf{R}^{ik}(\tilde{\mathbf{a}}^k - \mathbf{b}^i_{\mathbf{a}})\delta t^2 \right] \\ \prod_{k=i}^{m} \boldsymbol{\Delta}\mathbf{R}^{ik}\,\mathrm{Exp}((\tilde{\mathbf{a}}^k - \mathbf{b}^i_{\mathbf{a}})\delta t) \\ \prod_{k=i}^{m} \mathrm{Exp}((\tilde{\boldsymbol{\omega}}^k - \mathbf{b}^i_{\boldsymbol{\omega}})\delta t) \end{bmatrix} \tag{6.9}$$

and:

$$\hat{\boldsymbol{\Delta}}_{im}(\mathbf{x}^i, \mathbf{x}^m) \triangleq \begin{bmatrix} \mathbf{R}^{i,T}(\mathbf{p}^m - \mathbf{p}^i - \mathbf{v}^i\Delta t_{im} - \frac{1}{2}\mathbf{g}\Delta t_{im}^2) \\ \mathbf{R}^{i,T}(\mathbf{v}_m - \mathbf{v}_i - \mathbf{g}\Delta t_{im}) \\ \mathbf{R}^{i,T}\,\mathbf{R}^m \end{bmatrix}. \tag{6.10}$$

Let us now emphasize the fact that $\boldsymbol{\Delta}_{im}(\mathbf{b}^i)$ as defined in (6.9) does not depend on $\mathbf{x}_i$, contrary to (6.6).

But we are not over yet. The dependency on the bias variable $\mathbf{b}^i$ in (6.9) still enforces the repeated reintegration of the measurements buffer to get $\boldsymbol{\Delta}_{im}$ each time the solver produces a new estimate of $\mathbf{b}^i$. Fortunately, because the variations in $\mathbf{b}_i$ are small, a linearized approximation can be used. The IMU measurements can be pre-integrated using the prior bias estimation at time t_i, that we note $\overline{\mathbf{b}}_i$, to give the pre-integrated delta $\overline{\boldsymbol{\Delta}}_{im} \triangleq \boldsymbol{\Delta}_{im}(\overline{\mathbf{b}}_i)$. Then, each time a new $\mathbf{b}_i$ value is computed, we can correct the delta linearly:

$$\boldsymbol{\Delta}_{im}(\mathbf{b}^i) = \overline{\boldsymbol{\Delta}}_{im} \oplus \mathbf{J}^{\boldsymbol{\Delta}_{im}}_{\mathbf{b}_i}(\mathbf{b}_i - \overline{\mathbf{b}}_i), \tag{6.11}$$

that is, without the need for re-integration. This is why this method takes the name of delta pre-integration.

With all these considerations, our residual error can be written as

$$\mathbf{e}_{im}(\mathbf{x}^i, \mathbf{x}^m, \mathbf{b}^i) = (\overline{\boldsymbol{\Delta}}_{im} \oplus \mathbf{J}^{\boldsymbol{\Delta}_{im}}_{\mathbf{b}_i}(\mathbf{b}_i - \overline{\mathbf{b}}_i)) \ominus \hat{\boldsymbol{\Delta}}_{im} \tag{6.12}$$

In this expression, $\overline{\boldsymbol{\Delta}}_{im}$ only depends on data and has been integrated only once. The rest of the operations are small and can be made as many times as necessary in the solver side.

These two observations were first made by Lupton [LS09], whose formulation relied on Euler angles, and were later formalized on $SO(3)$ using Lie theory by Forster [For+17].

We will now show how this formulation can be generalized to other high rate sensory data by abstracting a recursive implementation of the pre-integration that takes advantage of the Lie-group structure of the deltas' geometry.

6.2 Generalized pre-integration on Lie groups

Pre-integration refers to the integration of high rate proprioceptive sensory data efficiently in the context of factor graphs. The IMU pre-integration theory can be generalized to many other proprioceptive sensors as shown in [Atc18; DAS19; Fou+21].

The generalization makes use of the Lie group structure of the motion deltas, (see Section 3.3.4). Then, the motion data is interpreted as a member of the tangent space of the group.

Different Lie group structures can be chosen that satisfy the necessary properties. For the IMU, we explored the two variants: composite [For+15] and compact group (our paper [Fou+19]).

In Chapter 7, we also use this generalized approach for the pre-integration of force-torque sensors for the centroidal estimation of legged robots based on the composite Lie group approach. This work was part of our published paper [Fou+21].

This section starts by stating the need to define a well-defined delta, then defining its Lie group structure in detail, then describing the pre-integration algorithm at the front-end, and finally the definition of the factor residual at the back-end.

6.2.1 Delta definition

The definition of the deltas needs to satisfy the property that such deltas $\boldsymbol{\Delta}_{im}$ resulting from the integration of motion data from time i to m must be independent of the initial state $\mathbf{x}^i$. The general rule to achieve this stems from observing that an unbiased and unnoisy sensor measuring null data should produce null deltas. This means that the deltas would be the deviations from the sensor trajectories exhibiting null data – for example, a free-falling non-rotating frame in the case of an IMU (see Fig. 6.2). Dependence on the small-varying parameters $\mathbf{b}$ such as sensor bias or other calibration parameters is then handled through linearization.

The relation of the delta with initial and final states is then defined through the operators $\boxplus$ and $\boxminus$,

$$\boxplus : \quad \mathcal{M}_{\mathbf{x}} \times \mathcal{M}_{\boldsymbol{\Delta}} \to \mathcal{M}_{\mathbf{x}}; \qquad (\mathbf{x}_i, \boldsymbol{\Delta}_{im}) \to \mathbf{x}_m = \mathbf{x}_i \boxplus \boldsymbol{\Delta}_{im} \tag{6.13}$$

$$\boxminus : \quad \mathcal{M}_{\mathbf{x}} \times \mathcal{M}_{\mathbf{x}} \to \mathcal{M}_{\boldsymbol{\Delta}}; \qquad (\mathbf{x}_i, \mathbf{x}_m) \to \boldsymbol{\Delta}_{im} = \mathbf{x}_m \boxminus \mathbf{x}_i \,. \tag{6.14}$$

6.2.2 Description of the Lie group and the tangent space

The Delta Lie group used in the specific application has to be defined, by describing in particular:

- its identity element $\boldsymbol{\Delta}_{\mathcal{E}}$,
- its inverse element $\boldsymbol{\Delta}^{-1}$,
- the composition law $\circ$, and
- the Exp and Log operators.

Once these basic blocks are established, we shall refer to Section 3.3.4 to define the $\oplus$ and $\ominus$ operators for the group. Likewise, to deal with the uncertainty of the integrated data, Jacobians of these operators will need to be obtained, for which we follow [SDA18].

Also, we leverage the fact that motion data can easily be manipulated to be part of the tangent space of this group since it expresses either velocities or increments on the state manifolds. To account for sensor self-calibration, this motion data is first corrected using currently known calibration parameters.

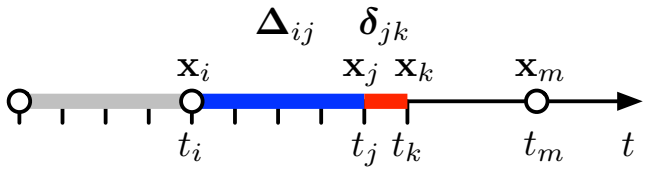


Figure 6.1: The pre-integrated delta $\boldsymbol{\Delta}_{ij}$ contains all motion from the last KF at time i, up to time j. The current delta $\boldsymbol{\Delta}_k$ contains the motion from times j to k, computed from the last IMU measurement at time k.

6.2.3 Delta pre-integration algorithm

From all the considerations exposed above, the delta pre-integration algorithm is performed recursively via the following steps.

Initialization First, $\overline{\boldsymbol{\Delta}}_{ii}$ is initialized to the null motion via the identity of the group, $\boldsymbol{\Delta}_{\mathcal{E}}$. Its covariance $\boldsymbol{\Sigma}_{\Delta}^{ii}$ and the Jacobian $\mathbf{J}_{\mathbf{b}_i}^{\boldsymbol{\Delta}_{ii}}$ are set to zero.

Calibration and retraction Second, at each reception of sensor data $\tilde{\mathbf{z}}_k$ at time k, we integrate during δt to obtain the delta corresponding to a single data sample

$$\boldsymbol{\delta}_k = f(\tilde{\mathbf{z}}_k, \overline{\mathbf{b}}_i, \delta t) \qquad \in \mathcal{M}, \tag{6.15}$$

using the calibration $\overline{\mathbf{b}}_i$ available in Keyframe i. In most cases, this function is split into the stages of data **calibration** $c()$, producing a vector in the tangent space, and **retraction** $\mathrm{Exp}()$, retracting it onto the manifold,

$$\boldsymbol{\tau}_k = c(\tilde{\mathbf{z}}_k, \overline{\mathbf{b}}_i)\delta t \qquad \in \mathbb{R}^{dim(\mathcal{M})} \tag{6.16}$$

$$\boldsymbol{\delta}_k = \mathrm{Exp}(\boldsymbol{\tau}_k) \qquad \in \mathcal{M}, \tag{6.17}$$

where the calibration, $c()$, depends on the sensor model, and the retraction, $\mathrm{Exp}()$, on the deltas Lie group.

Composition Third, this single delta is integrated onto the delta pre-integrated so far using the delta composition law

$$\overline{\boldsymbol{\Delta}}_{ik} = \overline{\boldsymbol{\Delta}}_{ij} \circ \boldsymbol{\delta}_k \qquad \in \mathcal{M}, \tag{6.18}$$

as can be seen in figure Fig. 6.1.

Covariance and Jacobian The delta covariance, as well as the Jacobian of the pre-integrated delta with respect to the calibration parameters, are also pre-integrated using the chain rule and standard covariance propagation,

$$\boldsymbol{\Sigma}_{\Delta}^{ik} = \mathbf{J}_{\boldsymbol{\Delta}_{ij}}^{\boldsymbol{\Delta}_{ik}} \boldsymbol{\Sigma}_{\Delta}^{ij} \mathbf{J}_{\boldsymbol{\Delta}_{ij}}^{\boldsymbol{\Delta}_{ik}}{}^{\top} + \mathbf{J}_{\boldsymbol{\delta}_k}^{\boldsymbol{\Delta}_{ik}} \mathbf{J}_{\mathbf{z}_k}^{\boldsymbol{\delta}_k} \boldsymbol{\Sigma}_{\mathbf{z}} \mathbf{J}_{\mathbf{z}_k}^{\boldsymbol{\delta}_k}{}^{\top} \mathbf{J}_{\boldsymbol{\delta}_k}^{\boldsymbol{\Delta}_{ik}}{}^{\top} \tag{6.19}$$

$$\mathbf{J}_{\mathbf{b}_i}^{\boldsymbol{\Delta}_{ik}} = \mathbf{J}_{\boldsymbol{\Delta}_{ij}}^{\boldsymbol{\Delta}_{ik}} \mathbf{J}_{\mathbf{b}_i}^{\boldsymbol{\Delta}_{ij}} + \mathbf{J}_{\boldsymbol{\delta}_k}^{\boldsymbol{\Delta}_{ik}} \mathbf{J}_{\mathbf{b}_i}^{\boldsymbol{\delta}_k}, \tag{6.20}$$

where $\mathbf{J}_{\mathbf{b}_i}^{\boldsymbol{\delta}_k}$, $\mathbf{J}_{\mathbf{z}_k}^{\boldsymbol{\delta}_k}$ are the Jacobians of (6.15) and $\mathbf{J}_{\boldsymbol{\Delta}_{ij}}^{\boldsymbol{\Delta}_{ik}}$, $\mathbf{J}_{\boldsymbol{\delta}_k}^{\boldsymbol{\Delta}_{ik}}$ the Jacobians of (6.18), computed according to Lie theory [SDA18], to which we give a brief introduction in Section 3.3.4.

These steps are performed recursively until a new Keyframe is added to the factor graph. When it is the case, a motion factor is created as explained in the next section.

6.2.4 Residual definition

When a new Keyframe is added to the problem at time $k = m$, a factor is created to which we pass $\overline{\boldsymbol{\Delta}}_{im}$, $\overline{\mathbf{b}}_i$, $\mathbf{J}^{\boldsymbol{\Delta}_{im}}_{\mathbf{b}_i}$, and $\boldsymbol{\Sigma}^{im}_{\Delta}$, with which the residual can be defined:

$$\mathbf{e}_{im}(\mathbf{x}^i, \mathbf{x}^m, \mathbf{b}^i) = (\overline{\boldsymbol{\Delta}}_{im} \oplus \mathbf{J}^{\boldsymbol{\Delta}_{im}}_{\mathbf{b}_i}(\mathbf{b}_i - \overline{\mathbf{b}}_i)) \ominus \hat{\boldsymbol{\Delta}}_{im} \tag{6.21}$$

with associated covariance $\boldsymbol{\Sigma}^{im}_{\Delta}$. Here, $\mathbf{b}_i$ is the current value of the sensor's calibration parameters, $\hat{\boldsymbol{\Delta}}_{im} = \mathbf{x}^m \boxminus \mathbf{x}^i$ is the expected delta between KFs, and $\{\oplus, \ominus\}$ are the plus and minus operators described in section Section 3.3.1.

In cases where the calibration parameters are subject to drift, the calibration drift is modeled as the integration of a random walk. This drift is included as a second factor in the factor graph, with residual error defined by

$$\mathbf{e}^{\mathbf{b}}_{im}(\mathbf{b}^i, \mathbf{b}^m) = \mathbf{b}^m - \mathbf{b}^i \tag{6.22}$$

with associated covariance

$$\boldsymbol{\Sigma}_{\mathbf{b},im} = \mathbf{W}^c_{\mathbf{b}} \Delta t_{im}. \tag{6.23}$$

6.2.5 Publishing the current optimal state

Interestingly, the proposed algorithm allows for the publication, at the high rate of the motion sensor and with little lag, of the current state of the robot. To do so, we take advantage of the delta pre-integrated so far, which can be corrected in case we dispose of better estimates of the calibration parameters. The current state at time t_k thus reads,

$$\mathbf{x}^k = \mathbf{x}^i \boxplus (\overline{\boldsymbol{\Delta}}_{ik} \oplus \mathbf{J}^{\boldsymbol{\Delta}_{ik}}_{\mathbf{b}_i}(\mathbf{b}_i - \overline{\mathbf{b}}_i)). \tag{6.24}$$

This current state is optimal in the following sense. First, it uses the state $\mathbf{x}^i$ of the last Keyframe, which has been eventually optimized by the back-end. Second, it uses the most updated version of the calibration parameters $\mathbf{b}_i$ to correct the pre-integrated delta, thus minimizing the dead-reckoning drift of the motion increment from Keyframe time t_i to the current time t_k.

The optimal state $\mathbf{x}^k$ is published at the rate of the motion sensor, which can reach the kHz, and might be used for closed-loop control of the robot. This is so regardless of the rate at which Keyframes are generated, and the rate at which the solver optimizes the problem. These rates, in any case, but especially if they are slow, will logically impact the accuracy of the published state.

6.3 IMU pre-integration on Lie groups

In this section, we will revisit the IMU pre-integration using the general pipeline defined in the previous section. First, we will frame the on-manifold pre-integration of Forster [For+15; For+17] as a delta pre-integration on a composite IMU delta group. Second, we will present an alternative formulation of IMU pre-integration on a new compact delta matrix Lie group $\mathcal{D}$ that we presented in [Fou+19]. We will finally discuss a few other alternatives.

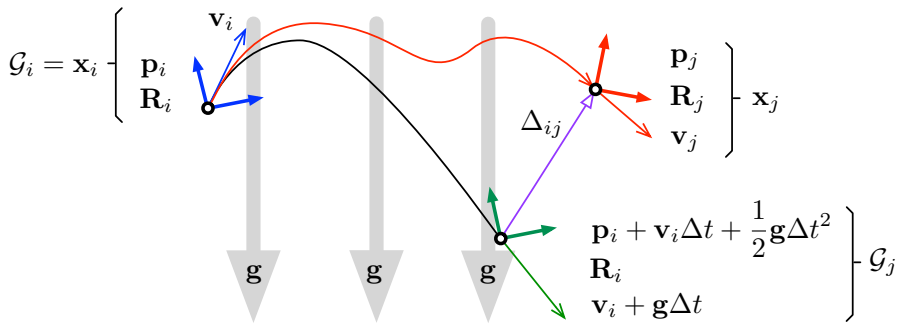


Figure 6.2: The free-falling, non-rotating frame $\mathcal{G}_t$ follows a parabolic trajectory governed only by gravity $\mathbf{g}$ and determined by the initial conditions $\mathbf{p}_i$, $\mathbf{v}_i$ and $\mathbf{R}_i$ at time i ($\mathcal{G}_i = \mathbf{x}_i$, blue). The IMU delta $\boldsymbol{\Delta}_{ij}$ between times i and j is defined as the state of the IMU at time j ($\mathbf{x}_j$, red) expressed in the free-falling frame at time j ($\mathcal{G}_j$, green).

6.3.1 IMU pre-integraton on composite Lie group

Let us come back to the IMU pre-integration problem, as stated by Forster [LS09; For+15], defined in Section 6.1, and show that we can rewrite the algorithm in terms of the generalized pre-integration described in Section 6.2.

The states involved in this integration are the base states $\mathbf{x} = [\mathbf{p}, \mathbf{v}, \mathbf{R}]$ with deltas $\boldsymbol{\Delta} = [\boldsymbol{\Delta}\mathbf{p}, \boldsymbol{\Delta}\mathbf{v}, \boldsymbol{\Delta}\mathbf{R}] \in \mathcal{M}_{\boldsymbol{\Delta}}$. The IMU produces biased and noisy measurements $\tilde{\mathbf{z}} = [\tilde{\mathbf{a}}, \tilde{\boldsymbol{\omega}}]$ of the base proper acceleration and angular velocity, with bias $\mathbf{b} = [\mathbf{b}_a, \mathbf{b}_\omega]$ and noise $\mathbf{n} = [\mathbf{n}_a, \mathbf{n}_\omega]$.

Definition of the deltas

The IMU deltas, as introduced in [LS09; For+15] can be defined [Atc18] as the motion increments, in terms of position, velocity and orientation, between the current IMU frame and another frame, that started at the IMU state at time i, $\mathbf{x}_i = (\mathbf{p}_i, \mathbf{v}_i, \mathbf{R}_i)$, and that falls freely and without rotating at the acceleration of gravity (Fig. 6.2).

This derives in the definition of the operator $\boxminus$ in (6.14) as what we detailed in (6.10), and the operator $\boxplus$ in (6.13) as being just its inverse. Full details can be found in [Atc18, Section 3.4].

Definition of the group operations

Given two IMU deltas $\boldsymbol{\Delta} = [\boldsymbol{\Delta}\mathbf{p}, \boldsymbol{\Delta}\mathbf{v}, \boldsymbol{\Delta}\mathbf{R}]$ and $\boldsymbol{\delta} = [\delta\mathbf{p}, \delta\mathbf{v}, \delta\mathbf{R}]$, the group composition law $\boldsymbol{\Delta} = \boldsymbol{\Delta} \circ \boldsymbol{\delta}$ in (6.18) is defined as

$$\boldsymbol{\Delta} \circ \boldsymbol{\delta} = \begin{bmatrix} \boldsymbol{\Delta}\mathbf{p} + \boldsymbol{\Delta}\mathbf{v}\delta t + \boldsymbol{\Delta}\mathbf{R}\delta\mathbf{p} \\ \boldsymbol{\Delta}\mathbf{v} + \boldsymbol{\Delta}\mathbf{R}\delta\mathbf{v} \\ \boldsymbol{\Delta}\mathbf{R}\delta\mathbf{R} \end{bmatrix} \tag{6.25}$$

with a group identity element composed of the identity element of its Lie groups:

$$\boldsymbol{\Delta}_{\mathcal{E}} = \begin{bmatrix} \mathbf{0}_3 \\ \mathbf{0}_3 \\ \mathbf{I}_3 \end{bmatrix}, \tag{6.26}$$

the resulting inverse being

$$\boldsymbol{\Delta}^{-1} = \begin{bmatrix} -\boldsymbol{\Delta}\mathbf{R}^\top(\boldsymbol{\Delta}\mathbf{p} + \boldsymbol{\Delta}\mathbf{v}\Delta t) \\ -\boldsymbol{\Delta}\mathbf{R}^\top\boldsymbol{\Delta}\mathbf{v} \\ \boldsymbol{\Delta}\mathbf{R}^\top \end{bmatrix} \tag{6.27}$$

The calibration function $c()$ and exponential map $\mathrm{Exp}()$ are,

$$\boldsymbol{\tau} = \begin{bmatrix} \boldsymbol{\tau}_a \\ \boldsymbol{\tau}_\omega \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{a}} - \mathbf{b}_a \\ \tilde{\boldsymbol{\omega}} - \mathbf{b}_\omega \end{bmatrix} \tag{6.28}$$

$$\boldsymbol{\delta} = \mathrm{Exp}(\boldsymbol{\tau}) = \begin{bmatrix} \frac{1}{2}\boldsymbol{\tau}_a \delta t^2 \\ \boldsymbol{\tau}_a \delta t \\ \mathrm{Exp}(\boldsymbol{\tau}_\omega \delta t) \end{bmatrix} \tag{6.29}$$

yielding $\boldsymbol{\delta} = f(\tilde{\mathbf{z}}, \mathbf{b}, \delta t)$ in (6.15) as

$$\boldsymbol{\delta}^k = \begin{bmatrix} \delta\mathbf{p} \\ \delta\mathbf{v} \\ \delta\mathbf{R} \end{bmatrix}^k = \begin{bmatrix} \frac{1}{2}(\tilde{\mathbf{a}} - \mathbf{b}_a)\delta t^2 \\ (\tilde{\mathbf{a}} - \mathbf{b}_a)\delta t \\ \mathrm{Exp}((\tilde{\boldsymbol{\omega}} - \mathbf{b}_\omega)\delta t) \end{bmatrix}^k \tag{6.30}$$

Delta pre-integration and factor residual

The concatenation of the operations above lead exactly to the Forster's algorithm [For+17]:

- Initialize $\boldsymbol{\Delta}_{ii} = \boldsymbol{\Delta}_\mathcal{E}$, $\Sigma_{\boldsymbol{\Delta}} = \mathbf{0}$, $\mathbf{J}^{\boldsymbol{\Delta}}_{\mathbf{b}} = \mathbf{0}$, and $\overline{\mathbf{b}}_i = \mathbf{b}_i$.
- Calibrate data and retract to manifold using (6.30).
- Compose $\overline{\boldsymbol{\Delta}}$ using (6.25).

Likewise, the factor residual is computed following Section 6.2.4 exactly.

6.3.2 IMU pre-integration on compact Lie group

This formulation was proposed in our published paper [Fou+19].

Delta definition

We propose a matrix form of the Lie group of IMU deltas as,

$$\boldsymbol{\Delta} = \begin{bmatrix} \boldsymbol{\Delta}\mathbf{R} & \boldsymbol{\Delta}\mathbf{v} & \boldsymbol{\Delta}\mathbf{p} \\ \mathbf{0} & 1 & \Delta t \\ \mathbf{0} & 0 & 1 \end{bmatrix} \in \mathcal{D} \subset \mathbb{R}^{5\times 5}\,. \tag{6.31}$$

The three main constituents of such deltas, namely $\boldsymbol{\Delta}\mathbf{p}$, $\boldsymbol{\Delta}\mathbf{v}$, and $\boldsymbol{\Delta}\mathbf{R}$, are defined exactly as in the composite case above. Being computed in the same manner, the definition of the $\boxminus$ and $\boxplus$ operators is a trivial manner of forming the matrix deltas from (6.10), yielding proper expressions for

$$\hat{\boldsymbol{\Delta}}_{im} = \mathbf{x}_m \boxminus \mathbf{x}_i \qquad\qquad \mathbf{x}_m = \mathbf{x}_i \boxplus \hat{\boldsymbol{\Delta}}_{im}.$$

Notice a fourth, time interval component Δt as being also part of the delta definition.

Group operations

Group composition, identity, and inverse stem from regular matrix product, identity, and inverse (with $\mathbf{\Delta R}^{-1} = \mathbf{\Delta R}^\top$),

$$\mathbf{\Delta} \circ \boldsymbol{\delta} = \begin{bmatrix} \mathbf{\Delta R}\delta\mathbf{R} & \mathbf{\Delta v} + \mathbf{\Delta R}\delta\mathbf{v} & \mathbf{\Delta p} + \mathbf{\Delta v}\delta t + \mathbf{\Delta R}\delta\mathbf{p} \\ \mathbf{0} & 1 & \Delta t + \delta t \\ \mathbf{0} & 0 & 1 \end{bmatrix} \tag{6.32}$$

$$\mathbf{\Delta}_\mathcal{E} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & 1 & 0 \\ \mathbf{0} & 0 & 1 \end{bmatrix} = \mathbf{I}_{5\times 5} \tag{6.33}$$

$$\mathbf{\Delta}^{-1} = \begin{bmatrix} \mathbf{\Delta R}^\top & -\mathbf{\Delta R}^\top\mathbf{\Delta v} & -\mathbf{\Delta R}^\top(\mathbf{\Delta p} - \mathbf{\Delta v}\Delta t) \\ \mathbf{0} & 1 & -\Delta t \\ \mathbf{0} & 0 & 1 \end{bmatrix}. \tag{6.34}$$

Comparing against (6.25–6.27) we observe that this matrix Lie group behaves equivalently to Forster's IMU deltas above.

Lie algebra $\mathfrak{d}$ and exponential map

The compact Lie group defines a different Lie algebra parametrization than Forster's, and therefore a different exponential map. The Lie algebra elements $\boldsymbol{\tau}^\wedge$ and their isomorphic Cartesian $\boldsymbol{\tau}$ have the forms

$$\boldsymbol{\tau}^\wedge = \begin{bmatrix} [\boldsymbol{\theta}]_\times & \boldsymbol{\rho} & \boldsymbol{\upsilon} \\ \mathbf{0} & 0 & \Delta t \\ \mathbf{0} & 0 & 0 \end{bmatrix} \in \mathfrak{d}, \qquad \boldsymbol{\tau} = \begin{bmatrix} \boldsymbol{\rho} \\ \boldsymbol{\upsilon} \\ \boldsymbol{\theta} \\ \Delta t \end{bmatrix} \triangleq \Delta t \begin{bmatrix} \mathbf{v} \\ \mathbf{a} \\ \boldsymbol{\omega} \\ 1 \end{bmatrix} \in \mathbb{R}^{10}, \tag{6.35}$$

with $\mathbf{v} \triangleq \dot{\mathbf{\Delta p}}$, $\mathbf{a} \triangleq \dot{\mathbf{\Delta v}}$ and $[\boldsymbol{\omega}]_\times \triangleq \mathbf{\Delta R}^\top\dot{\mathbf{\Delta R}}$. Operators $\wedge$ and $\vee$ are defined so that $\boldsymbol{\tau}^\wedge = (\boldsymbol{\tau})^\wedge$ and $\boldsymbol{\tau} = (\boldsymbol{\tau}^\wedge)^\vee$.

The exponential map transfers tangent elements to the group; the logarithmic map is its inverse,

$$\mathbf{\Delta} = \mathrm{Exp}(\boldsymbol{\tau}) \triangleq \exp(\boldsymbol{\tau}^\wedge) = \begin{bmatrix} \mathrm{Exp}(\boldsymbol{\theta}) & \mathbf{Q}\boldsymbol{\upsilon} & \mathbf{Q}\boldsymbol{\rho} + \mathbf{P}\boldsymbol{\upsilon}\Delta t \\ \mathbf{0} & 1 & \Delta t \\ \mathbf{0} & 0 & 1 \end{bmatrix} \tag{6.36}$$

$$\boldsymbol{\tau} = \mathrm{Log}(\mathbf{\Delta}) \triangleq \log(\mathbf{\Delta})^\vee = \begin{bmatrix} \mathbf{Q}^{-1}(\mathbf{\Delta p} - \mathbf{P}\mathbf{Q}^{-1}\mathbf{\Delta v}\Delta t) \\ \mathbf{Q}^{-1}\mathbf{\Delta v} \\ \mathrm{Log}(\mathbf{\Delta R}) \\ \Delta t \end{bmatrix} \tag{6.37}$$

where $\mathrm{Log}()$ is obtained by identifying terms in (6.31) and (6.36). Matrices $\mathbf{P}$ and $\mathbf{Q}$ are provided in the appendix's Section B.2.

The case of IMU data, leading to $\mathbf{v} = 0$ We showed that IMU deltas could be interpreted as the motion relative to the free-falling frame (Fig. 6.2), which has initial velocity $\mathbf{v}_i$, and the same physical analogy applies to the compact Lie group. Thus the tangent velocity $\mathbf{v} = \dot{\mathbf{\Delta p}}$ is zero at the start of the integration step. Since the exponential $\mathrm{Exp}(\boldsymbol{\nu}\Delta t)$

assumes a constant tangent vector $\boldsymbol{\nu} = (\mathbf{v}, \mathbf{a}, \boldsymbol{\omega}, 1)$ during the interval Δt, we have that $\mathbf{v} = 0$ during the full step. This gives immediately

$$\mathrm{Exp}\left(\begin{bmatrix}\mathbf{0}\\ \mathbf{a}\\ \boldsymbol{\omega}\\ 1\end{bmatrix}\Delta t\right) = \begin{bmatrix}\mathrm{Exp}(\boldsymbol{\omega}\Delta t) & \mathbf{Q}\mathbf{a}\Delta t & \mathbf{P}\mathbf{a}\Delta t^2\\ \mathbf{0} & 1 & \Delta t\\ \mathbf{0} & 0 & 1\end{bmatrix}, \tag{6.38}$$

which we will use to integrate IMU data.

Pre-integration pipeline

Here we recall the general pre-integration pipeline in the case of the compact Lie group, pointing at the minor differences with the composite Lie group case. The following pipeline of operations is performed to recursively pre-integrate IMU data into a unique measurement.

Initialization Pre-integration starts after each Keyframe with $\overline{\boldsymbol{\Delta}}_{ii} = \mathbf{I}$, $\boldsymbol{\Sigma}^{\Delta}_{ii} = \mathbf{0}$ and $\mathbf{J}^{\boldsymbol{\Delta}_{ii}}_{\mathbf{b}} = \mathbf{0}$, using $\overline{\mathbf{b}}_i = \mathbf{b}_i$ the current best estimate of the bias at time i.

Calibration At the reception of each IMU measurement $\mathbf{z}_k = (\mathbf{a}, \boldsymbol{\omega})_k$, start by correcting it with available bias estimates $\overline{\mathbf{b}}_i = (\mathbf{b}_a, \mathbf{b}_\omega)_i$, to produce the tangent vector $\boldsymbol{\tau} = \boldsymbol{\nu}\delta t$. For this, set the velocity part of $\boldsymbol{\nu}$ to zero as the IMU is by definition at zero speed with respect to the moving frame, as in (6.38). Obtain at the same time the respective Jacobians,

$$\boldsymbol{\tau}_k = \begin{bmatrix}\mathbf{0}\\ \mathbf{a}-\mathbf{b}_a\\ \boldsymbol{\omega}-\mathbf{b}_\omega\\ 1\end{bmatrix}\delta t, \qquad \mathbf{J}^{\boldsymbol{\tau}}_{\mathbf{z}} = \begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\\ 0 & 0\end{bmatrix}\delta t, \qquad \mathbf{J}^{\boldsymbol{\tau}}_{\mathbf{b}} = -\begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\\ 0 & 0\end{bmatrix}\delta t \tag{6.39}$$

Retraction Second, use the exponential map (6.38) to obtain the current delta step $\boldsymbol{\delta}^k$ in the group manifold, and obtain Jacobian

$$\boldsymbol{\delta}^k = \mathrm{Exp}(\boldsymbol{\tau}_k)\,, \qquad \mathbf{J}^{\boldsymbol{\delta}}_{\boldsymbol{\tau}} = \mathbf{J}_r(\boldsymbol{\tau}_k)\,. \tag{6.40}$$

Composition Third, use group composition (6.32) to update the pre-integrated delta; obtain Jacobians

$$\overline{\boldsymbol{\Delta}}_{ik} = \overline{\boldsymbol{\Delta}}_{ij} \circ \boldsymbol{\delta}^k\,, \qquad \mathbf{J}^{\boldsymbol{\Delta}_{ik}}_{\boldsymbol{\Delta}_{ij}} = \mathbf{Ad}^{-1}_{\boldsymbol{\delta}^k}\,, \qquad \mathbf{J}^{\boldsymbol{\Delta}_{ik}}_{\boldsymbol{\delta}^k} = \mathbf{I}\,, \tag{6.41}$$

where $\mathbf{Ad}_{\boldsymbol{\delta}}$ is the adjoint and $\mathbf{J}_r$ is the right Jacobian —see appendices Section B.2.3, Section B.2.4 and technical paper [SDA18] for reference.

Covariance Fourth, propagate the delta covariance

$$\boldsymbol{\Sigma}^{\Delta}_{ik} = \mathbf{J}^{\boldsymbol{\Delta}_{ik}}_{\boldsymbol{\Delta}_{ij}} \boldsymbol{\Sigma}^{\Delta}_{ij} \mathbf{J}^{\boldsymbol{\Delta}_{ik}}_{\boldsymbol{\Delta}_{ij}}{}^{\top} + \mathbf{J}^{\boldsymbol{\Delta}_{ik}}_{\mathbf{z}} \boldsymbol{\Sigma}_{\mathbf{z}} \mathbf{J}^{\boldsymbol{\Delta}_{ik}}_{\mathbf{z}}{}^{\top}\,, \tag{6.42}$$

with $\boldsymbol{\Sigma}_{\mathbf{z}}$ the covariance of the IMU measurements $\mathbf{z}$, and $\mathbf{J}^{\boldsymbol{\Delta}_{ik}}_{\mathbf{z}} = \mathbf{J}^{\boldsymbol{\Delta}_{ik}}_{\boldsymbol{\delta}} \mathbf{J}^{\boldsymbol{\delta}}_{\boldsymbol{\tau}} \mathbf{J}^{\boldsymbol{\tau}}_{\mathbf{z}}$ computed using the chain rule.

Jacobian Finally, integrate the Jacobian of the delta with respect to the biases

$$\mathbf{J}_{\mathbf{b}}^{\boldsymbol{\Delta}_{ik}} = \mathbf{J}_{\boldsymbol{\Delta}_{ij}}^{\boldsymbol{\Delta}_{ik}} \mathbf{J}_{\mathbf{b}}^{\boldsymbol{\Delta}_{ij}} + \mathbf{J}_{\boldsymbol{\delta}^k}^{\boldsymbol{\Delta}_{ik}} \mathbf{J}_{\tau}^{\delta} \mathbf{J}_{\mathbf{b}}^{\tau} \,. \tag{6.43}$$

Pre-integration is complete (Fig. 6.1) when $k = m$, which yields $\overline{\boldsymbol{\Delta}}_{im}$, $\boldsymbol{\Sigma}_{im}^{\Delta}$ and $\mathbf{J}_{\mathbf{b}}^{\boldsymbol{\Delta}_{im}}$.

Factor residual

Computation of the residual in the case of the Lie Delta group formulation follows the steps defined in Section 6.2. Use the pre-integrated Jacobian $\mathbf{J}_{\mathbf{b}}^{\boldsymbol{\Delta}_{im}}$ to correct the pre-integrated delta $\overline{\boldsymbol{\Delta}}_{im}$ to account for the new bias estimate $\mathbf{b}_i \neq \overline{\mathbf{b}}_i$,

$$\boldsymbol{\Delta}_{im}(\mathbf{b}_i) = \overline{\boldsymbol{\Delta}}_{im} \oplus \mathbf{J}_{\mathbf{b}}^{\boldsymbol{\Delta}_{im}} (\mathbf{b}_i - \overline{\mathbf{b}}_i) \,. \tag{6.44}$$

Use (6.10) as $\boxminus$ to compute the expected delta from $\mathbf{x}_i$ to $\mathbf{x}_m$,

$$\widehat{\boldsymbol{\Delta}}_{im}(\mathbf{x}_i, \mathbf{x}_m) = \mathbf{x}_m \boxminus \mathbf{x}_i \,. \tag{6.45}$$

Compute the residual in the tangent of $\mathcal{D}$ at $\boldsymbol{\Delta}_{im}$,

$$\begin{aligned} \mathbf{e}_{im}^{\boldsymbol{\Delta}}(\mathbf{x}_i, \mathbf{x}_m, \mathbf{b}_i) &= \widehat{\boldsymbol{\Delta}}_{im}(\mathbf{x}_i, \mathbf{x}_m) \ominus \boldsymbol{\Delta}_{im}(\mathbf{b}_i) && (6.46) \\ &= \mathrm{Log}(\boldsymbol{\Delta}_{im}(\mathbf{b}_i)^{-1} \circ \widehat{\boldsymbol{\Delta}}_{im}(\mathbf{x}_i, \mathbf{x}_m)) \in \mathbb{R}^9 \,, && (6.47) \end{aligned}$$

In this last equation, the minus operator $\ominus$ is the lift operator defined in (3.25) and specialized for this particular group.

Jacobians, uncertainty

For general functions $f : \mathcal{M} \to \mathcal{N}; y = f(x)$, we propagate uncertainty normally via the Jacobians $\mathbf{J}_x^y \triangleq \frac{\partial y}{\partial x}$, *i.e.* , $\boldsymbol{\Sigma}_y = \mathbf{J}_x^y \, \boldsymbol{\Sigma}_x \, \mathbf{J}_x^{y\top}$. These Jacobians map the tangent spaces of the manifolds $\mathcal{M}, \mathcal{N}$ at x and y, and in the case of vector spaces, they resort to the classical Jacobian. They also satisfy the chain rule, which we use extensively in our developments. Ample reference and justification of this approach can be found in [SDA18].

A comment is however necessary for the present IMU case. It relates to the uncertainty of the last component of the tangent space (6.35), which is the time Δt. This component has no uncertainty by definition. Having it in the covariances would imply singularity and result in the risk of several well-known numerical issues. We therefore systematically marginalize this time component out of the covariances, simply by removing the last row and column.

6.3.3 About the choice of the proper Lie group

Our IMU Lie group versus Forster's method

Mathematically, and disregarding methodology, the main difference between our method (Section 6.3.1) and Forster's [For+17] (Section 6.3.2) is to be found in the exponential map. To see it, let us consider small rotation increments $\boldsymbol{\theta} = \boldsymbol{\omega}\delta t$ captured at each single IMU sample. In such cases, the matrices $\mathbf{P}, \mathbf{Q}$ appearing in the exponential map (6.36)

and detailed in (B.9) can be approximated by $\mathbf{P} \approx \frac{1}{2}\mathbf{I}$ and $\mathbf{Q} \approx \mathbf{I}$. The exponential becomes,

$$\operatorname{Exp}\left(\begin{bmatrix}\mathbf{0}\\ \mathbf{a}\\ \boldsymbol{\omega}\\ 1\end{bmatrix}\delta t\right) \approx \begin{bmatrix}\operatorname{Exp}(\boldsymbol{\omega}\delta t) & \mathbf{a}\delta t & \frac{1}{2}\mathbf{a}\delta t^2\\ \mathbf{0} & 1 & \delta t\\ \mathbf{0} & 0 & 1\end{bmatrix}, \tag{6.48}$$

where we find the terms $\mathbf{a}\delta t$ and $\frac{1}{2}\mathbf{a}\delta t^2$, which are exactly the terms obtained with Forster's method (6.30). In effect, with this approximation, if we now compact all the steps (6.39–6.41) of our integration into a cumulative expression,

$$\boldsymbol{\Delta}_{ik} = \prod_{j=i+1}^{k} \operatorname{Exp}\left(\begin{bmatrix}\mathbf{0}\\ (\mathbf{a}^j - \mathbf{b}_a^j)\\ (\boldsymbol{\omega}^j - \mathbf{b}_{\boldsymbol{\omega}}^j)\\ 1\end{bmatrix}\delta t\right), \tag{6.49}$$

it is possible (although tedious) to show that both Forster's and our method are exactly equivalent when $\boldsymbol{\omega}\delta t \to 0$.

Further discussion regarding Lie group choice

Regarding "compact group" designs, there is still another proposal, the $SE_2(3)$ group proposed by [BB20; Bro+21], which can be used for IMU pre-integration. This proposal differs from ours in the following aspects:

- The $SE_2(3)$ group does not contain the time and therefore the composition law does not account for the whole integration. Some extra algebra needs to be added.
- The $SE_2(3)$ group is easier to manipulate since the closed forms for the exponential map, the adjoint, and the right-Jacobian are easier to obtain.
- Our group better separates between the delta states and the velocity of these states which depend only on the IMU data.

Therefore, it is apparent that depending on the structure of the defined Lie group, we can have different designs:

- Forster [For+15]: Composite Lie group
- Barrau [BB20]: $SE_2(3)$ compact Lie group without time
- Fourmy [Fou+19]: Compact IMU group with time

In the context of filtering, using a compact group formulation of the robot state has been proven to improve greatly the basin of convergence of the so-called Invariant Kalman Filter [BB18; Har+20]. This method tackles one of the issues of standard EKF: the Jacobians are computed around the current estimates, which leads to an inconsistency of the estimate if the filter is initialized far from the optimal. However, this issue is not so present with factor graph optimization since we repeatedly linearize around the new estimates. Nevertheless, it seems that compact Lie groups may provide slightly better performances than composite Lie groups [Bro+21], thanks to a more precise linearization

with respect to the IMU biases and a covariance propagation that better represents the geometry of the problem.

However one may ask whether these more elegant mathematical formulations and slight improvements are worth the effort. Compact designs may be seen as going against our modularity philosophy since for each new motion sensor, we need to find a new appropriate Lie group instead of being able to reuse the machinery developed for composite groups, which is vastly simpler.

6.4 Related works

Pre-integration principles were laid out by Lupton [LS09] for a smoothing-based visual-inertial estimator. His work was motivated partly by the fact that previous systems required a precise initialization of orientation and velocity (using a specialized routine) to begin to integrate IMU measurements. With this new formulation, Lupton noted that pre-integration of IMU data permitted the use of measurements immediately and delay the estimation of the initial orientation about the gravity vector in particular.

This seminal work was quickly adopted by other authors using smoothing filters [Car+14]. As pioneering as this work was, it was however limited by the use of Euler angles whose problematic geometric properties are notorious. Indelman [Ind+13] first proposed to use the exponential of the matrix rotation group instead of Euler integration and to relax the assumptions of non-rotating and flat earth of Lupton [LS09]. Forster [For+15; For+17] proposed the same formulation using the SO(3) Lie group, which was adapted to the quaternions group S^3 by Atchuthan [Atc18]. Various experiments brought to light three main problems with the Euler angle formulation, that are completely absent from the quaternion "on-manifold" formulation. Firstly, first-order integration of angular velocities using Euler angles is approximate, which leads to accumulated errors for high angular velocities or sampling rate. Secondly, the log-likelihood of the angular displacement is not invariant under the action of rigid body transformations, *e.g.* the choice of the world frame influence the results of the estimation. Finally, the well-known gimbal lock singularity of Euler angles has a consequence on the IMU noise covariance propagation, which is severely degraded when the robot trajectory comes close to the singularity. [SMK15], later improved in [QLS18] proposed to use a more precise numerical integration procedure than the default forward Euler used by Forster. Eckenhoff [EGH19] derived closed-form solutions of the pre-integration equations using various piecewise constant models.

Barrau [BB20] described a compact matrix Lie group for the propagation of pre-integration errors taking into account the earth rotation with the aerospace inertial navigation system in view. This work was later extended [Bro+21] and showed that the linearized bias update is slightly more precise than the work of Forster [For+17]. Le Gentil [LVH20] used a different trajectory parametrization framework by formulating the pre-integration algorithm in the context of Gaussian Process smoothing. Self-calibration of IMU/Camera time offsets was also developed [Yan+20b]. [LGL+21] derived a comprehensive collection of motion models depending on the various possible choices of reference frames and motion conditions.

As we saw the pre-integration theory began in the context of visual-inertial odometry. It was however adapted to other high rate sensors such as wheel odometry [Qua+19], possibly including self-calibration [DAS19]. In his thesis, Atchuthan ([Atc18, Section 4.3]) derived the general form of the pre-integration equations as a sensor agnostic form

that is integrated in the state estimation framework WOLF [Sol+21]. As previously mentioned, other teams applied pre-integration theory in the context of factor graph legged robots state estimation to derive new leg-odometry factors [Har+18c; WCF19; WCF20]. It was also applied to integrate drone dynamics to estimate external forces disturbances [Nis+19].

In Chapter 7, we propose to pre-integrate force-torque measurements that are present in some legged-robots. We show that permits to estimate the centroidal quantities of the robot as well making the kinematic bias on the center of mass measurements observable. A practical implementation is demonstrated in Chapter 9, which is based on our paper [Fou+21]

6.5 Conclusion

In this chapter, we have recalled the mathematics of the pre-integration, which is very important when handling high-frequency measurements strongly correlated with Lie-group quantities. Pre-integration was first introduced for handling IMU in the MAP framework, as we also did in our experiments. We introduced here an abstraction of the original mathematics, which allows us to more systematically generalize to other measurements. We also present a reformulation of the algorithm based on a compact Lie group. We also believe that this formulation, relying on Lie theory, gives some easier intuition to catch, at least when the reader is familiar with Lie groups.

We are now going to use the generalized pre-integration to handle force sensors, which also stream meaningful information at high (1 kHz) frequency, and integrate to to an extended MAP problem where the centroidal quantities are also estimated simultaneously to previously estimated basis state and map variables.

Chapter 7

Underactuated dynamics and centroidal states

Contents

In this chapter, we discuss how a tight coupling of the force measurements in the MAP formulation can lead to the direct estimation of centroidal quantities, *i.e.* the robot center of mass (CoM) position and velocity, and its angular momentum.

Centroidal estimation is crucial for legged robots balancing. However, this aspect has not been explored in the case of a factor graph estimator. We propose to fuse the centroidal kinematics of the robots with the pre-integration of the force-torque measurements of the platform.

We will first review the dynamical model of legged robots, then describe how we propose to use them as measurement models for quadruped and humanoid robots alike. In particular, we design a new application of the generalized pre-integration algorithm (see Section 6.2) for force-torque measurements, which constitutes one of the major theoretical contributions of this thesis.

7.1 Centroidal dynamics

The robot dynamics is described by the Lagrangian dynamics:

$$\mathbf{M}(\mathbf{q})\dot{\mathbf{v}}_q + h(\mathbf{q}, \mathbf{v}_q) = \boldsymbol{\tau}_q + \sum_l \mathbf{J}_l^\top \phi_l \tag{7.1}$$

where $\mathbf{q}, \mathbf{v}_q, \dot{\mathbf{v}}_q, \boldsymbol{\tau}_q$ are the position, velocity, acceleration and torques of the robot in configuration space, $\phi_l \triangleq [\mathbf{f}_l, \mathbf{m}_l]$ are the contact force-torques, $\mathbf{M}$ is the generalized inertia matrix, h the sum of gravity, Coriolis and centrifugal forces, and $\mathbf{J}_l$ the Jacobians of

the contact points. Because of the underactuated nature of legged robots, the configuration is often separated into $\mathbf{q} = ({}^W\mathbf{p}_B, {}^W\mathbf{q}_B, \mathbf{q}_a)$ where ${}^W\mathbf{p}_B$ is the position of the robot base in world frame (typically, the torso or in our case the IMU), ${}^W\mathbf{q}_B$ the orientation of the base body with respect to the world and $\mathbf{q}_a$ are the joint configuration of the actuated joints. The generalized velocity has a similar separation, as described in Section 5.1.

While (7.1) represents the whole dynamics, a subpart of it is of particular importance for legged robots. The centroidal dynamics is written by the two equations:

$$m\ddot{\mathbf{c}} = m\mathbf{g} + \sum_l \mathbf{f}_l \quad , \quad \dot{\mathcal{L}} = \sum_l (\mathbf{p}_l - \mathbf{c}) \times \mathbf{f}_l + \mathbf{m}_l \tag{7.2}$$

where $\mathbf{c}, \mathcal{L}$ are the position of the Center of Mass (CoM) and Angular Momentum (AM) around the CoM (both expressed in world frame), m is the robot total mass, and the $\mathbf{p}_l$ are the positions of the contact points in world frame. The centroidal dynamics is an exact part of (7.1) and more deeply reveals the underactuation: the robot can move only if applying the proper forces and torques to the environment, as the joint torques alone cannot lead to any modification of CoM or AM.

The classical approach in legged robot state estimation is to first estimate the base state and then to reconstruct the centroidal state in world frame

$$(\mathbf{c}, \dot{\mathbf{c}}, \mathcal{L}) \triangleq ({}^W\mathbf{p}_C, {}^W\mathbf{v}_C, {}^W\mathcal{L}) \tag{7.3}$$

using the joint position and velocity measurements, and the robot model. This assumes that there is no direct measurement of the centroidal state. Consequently, we are not able to recover the exact centroidal state if there is any bias in the robot model.

Yet, we can see from the centroidal dynamics that the force measurements are connected to the variation of the centroidal state. As observed in [Car+16a], a proper fusion of the force measurements with an estimation of the state of the base makes the centroidal state observable. We achieve this by adding two different kinds of information to the factor graph: centroidal kinematics and pre-integrated forces and torques. The derivation of their involved factors is presented hereafter.

7.2 Centroidal kinematics

We need to relate base states to centroidal quantities to ground their estimate. For that, we can rely on the inertial-kinematic model to compute the CoM position with respect to base frame ${}^B\mathbf{p}_C(\mathbf{q}_a) \in \mathbb{R}^3$, the CoM velocity with respect to base frame ${}^B\mathbf{v}_C(\mathbf{q}_a, \dot{\mathbf{q}}_a) \in \mathbb{R}^3$, the inertial matrix $\mathcal{I}(\mathbf{q}_a) \in \mathbb{R}^{3\times3}$, and the kinematic momentum due to gesticulation of the robot limbs $\mathcal{L}_a(\mathbf{q}_a, \dot{\mathbf{q}}_a) \in \mathbb{R}^3$. The geometrical relation between base states and centroidal states is:

$$\begin{aligned} \mathbf{c} &= \mathbf{R}\,{}^B\mathbf{p}_C + \mathbf{p} \\ \dot{\mathbf{c}} &= \mathbf{v} + \mathbf{R}({}^B\boldsymbol{\omega}_B \times {}^B\widetilde{\mathbf{p}}_C + {}^B\widetilde{\mathbf{v}}_C) \\ \mathcal{L} &= \mathbf{R}(\mathcal{I}\,{}^B\boldsymbol{\omega}_B + \mathcal{L}_a) \end{aligned} \tag{7.4}$$

By definition, computation of the CoM position from the kinodynamic model resorts to the formula

$${}^B\mathbf{p}_\mathbf{c} = \sum_{i=1}^{n} m_i\,{}^B\mathbf{p}_{\mathbf{c}_i}(\mathbf{q}_a), \tag{7.5}$$

were m_i are the individual segment masses and ${}^B\mathbf{p}_{\mathbf{c}_i}$ the levers between the base and the center of mass of each segment, computed from the robot configurations $\mathbf{q}_a$. In general, these terms are not exactly accurate as our model of the robot may not be faithful to the real system. The computation of ${}^B\mathbf{p}_\mathbf{c}$ and masses m_i are therefore biased. Worse, this bias is not constant, since it nonlinearly depends on the robot limbs configuration.

We chose a simple additive model on the CoM kinematics noisy measurements

$$ {}^B\widetilde{\mathbf{p}}_C = {}^B\mathbf{p}_C + \mathbf{b}_c + \mathbf{n}_c \,, \tag{7.6} $$

where $\mathbf{n}_c$ is a Gaussian noise and $\mathbf{b}_c$ is a varying bias that we wish to estimate.

The angular velocity from the IMU is used and its bias $\mathbf{b}_\omega$ has to be incorporated in the factor, $\widetilde{\boldsymbol{\omega}} = {}^B\boldsymbol{\omega}_B + \mathbf{b}_\omega + \mathbf{n}_\omega$. In the end, the equations used to derive the factor are:

$$ \begin{aligned} \mathbf{c} &= \mathbf{R}({}^B\widetilde{\mathbf{p}}_C - \mathbf{b}_c - \mathbf{n}_c) + \mathbf{p} \\ \dot{\mathbf{c}} &= \mathbf{v} + \mathbf{R}\left[(\widetilde{\boldsymbol{\omega}} - \mathbf{b}_\omega - \mathbf{n}_\omega) \times ({}^B\widetilde{\mathbf{p}}_C - \mathbf{b}_c - \mathbf{n}_c) + ({}^B\widetilde{\mathbf{v}}_C - \mathbf{n}_v)\right] \\ \mathcal{L} &= \mathbf{R}(\mathcal{I}(\widetilde{\boldsymbol{\omega}} - \mathbf{b}_\omega - \mathbf{n}_\omega) + \mathcal{L}_a) \end{aligned} \tag{7.7} $$

The residual $\mathbf{e}^{CK} \in \mathbb{R}^9$ is finally expressed as:

$$ \mathbf{e}^{CK}(\mathbf{p}, \mathbf{v}, \mathbf{R}, \mathbf{c}, \dot{\mathbf{c}}, \mathcal{L}, \mathbf{b}_c, \mathbf{b}_{\boldsymbol{\omega}}) = \begin{bmatrix} \mathbf{R}^\top(\mathbf{c} - \mathbf{p}) + \mathbf{b}_c - {}^B\widetilde{\mathbf{p}}_C \\ \mathbf{R}^\top(\dot{\mathbf{c}} - \mathbf{v}) - \left[(\widetilde{\boldsymbol{\omega}} - \mathbf{b}_\omega) \times ({}^B\widetilde{\mathbf{p}}_C - \mathbf{b}_p) + {}^B\widetilde{\mathbf{v}}_C\right] \\ \mathbf{R}^\top\mathcal{L} - [\mathcal{I}(\widetilde{\boldsymbol{\omega}} - \mathbf{b}_\omega) + \mathcal{L}_a] \end{bmatrix} \tag{7.8} $$

To obtain a covariance $\boldsymbol{\Sigma}_{CK}$ associated to the residual, we considered three sources of measurement noise: the CoM position measurement $\mathbf{n}_c$, the CoM velocity measurement $\mathbf{n}_v$, and the angular velocity measurement $\mathbf{n}_\omega$. We group this noises in a single noise vector $\mathbf{n} = [\mathbf{n}_c, \mathbf{n}_v, \mathbf{n}_\omega]$ with associated covariance:

$$ \boldsymbol{\Sigma}_\mathbf{n} = \begin{pmatrix} \boldsymbol{\Sigma}_p & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_v & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{\Sigma}_\omega \end{pmatrix} \tag{7.9} $$

The residual covariance is then obtained by propagating the noise $\mathbf{n}$ through the non-linearity of the residual error function (7.8), neglecting the second order noise terms found in (7.7). The linearized propagation writes $\boldsymbol{\Sigma}_{CK} = \mathbf{J}^\mathbf{e}_\mathbf{n}\boldsymbol{\Sigma}_\mathbf{n}\mathbf{J}^{\mathbf{e},\top}_\mathbf{n}$, with $\mathbf{J}^\mathbf{e}_\mathbf{n}$ the Jacobian with respect to the noise vector constructed as:

$$ \mathbf{J}^\mathbf{e}_\mathbf{n} = \begin{pmatrix} \mathbf{I} & 0 & 0 \\ [\widetilde{\boldsymbol{\omega}} - \mathbf{b}_\omega]_\times & \mathbf{I} & -[{}^B\widetilde{\mathbf{p}}_C - \mathbf{b}_p]_\times \\ 0 & 0 & \mathcal{I} \end{pmatrix} \tag{7.10} $$

This factor is added to the factor graph as a link between base states and centroidal states and, therefore, depends on almost all variables of the problem at a certain timestamp. We use this factor in the application presented in Chapter 9, whose factor graph is represented in Fig. 9.1.

7.3 Force-torque pre-integration

In this section, we apply the generalized pre-integration algorithm to the problem of using measured external forces applied on a legged robot in a smoothing estimator. We

propose to integrate the underactuated dynamics (7.2) using force-torque measurements. To this end, we derive the specificities of the pre-integration of external forces of a poly-articulated system. This is the main theoretical contribution of this chapter.

7.3.1 Newton-Euler integration

The Newton-Euler equations (7.2) relate the evolution of the CoM and AM due to gravity, external forces, and torques. In the case of a legged robot with punctual contact feet, only forces $\widetilde{\mathbf{f}}_L$ are applied at each limb contact L, with no torque. However, we will derive the equations in the general case where force-torque are available (like for humanoids) since it is then easy to just set the torque terms to zero.

We assume that at each limb contact we have access to noisy local force $\widetilde{\mathbf{f}}_L = {}^L\mathbf{f}_L + \mathbf{n}_f$ and pure torques $\widetilde{\mathbf{m}}_L = {}^L\mathbf{m}_L + \mathbf{n}_m$ measurements. To transform them into the body frame b, we compute $\widetilde{\mathbf{R}}_L \triangleq {}^B\mathbf{R}_L(\mathbf{q}_a) \in SO(3)$ and $\widetilde{\mathbf{p}}_L \triangleq {}^B\mathbf{p}_L(\mathbf{q}_a) \in \mathbb{R}^3$ from the joint configuration $\mathbf{q}_a \in \mathbb{R}^{12}$ using forward kinematics Section 5.1. The lever arm $(\mathbf{p}_L - \mathbf{c})$ in the Euler equation (7.2) uses a measurement of the CoM position in base frame ${}^B\mathbf{p}_C(\mathbf{q}_a) \in \mathbb{R}^3$. This measure is biased and noisy as explained before: we use again the measurement model (7.6). Assuming constant forces during each interval δt the integration of (7.2) yields the discrete dynamic model for the centroidal states:

$$
\begin{aligned}
\mathbf{c}^k &= \mathbf{c}^{k-1} + \dot{\mathbf{c}}^{k-1}\delta t + \frac{1}{2}\mathbf{g}\delta t^2 + \frac{1}{2m}\mathbf{R}^{k-1}\sum_L \widetilde{\mathbf{R}}_L^k(\widetilde{\mathbf{f}}_L^k - \mathbf{n}_f)\delta t^2 \\
\dot{\mathbf{c}}^k &= \dot{\mathbf{c}}^{k-1} + \mathbf{g}\delta t + \frac{1}{m}\mathbf{R}^k\sum_L \widetilde{\mathbf{R}}_L^k(\widetilde{\mathbf{f}}_L^k - \mathbf{n}_f)\delta t \\
\mathcal{L}^k &= \mathcal{L}^{k-1} + \mathbf{R}^k\sum_L \left[(\widetilde{\mathbf{p}}_L^k - \widetilde{\mathbf{p}}_C^k + \mathbf{b}_c^k + \mathbf{n}_c) \times \widetilde{\mathbf{R}}_L^k(\widetilde{\mathbf{f}}_L^k - \mathbf{n}_f) + \widetilde{\mathbf{R}}_L^k({}^L\widetilde{\mathbf{m}}_L - \mathbf{n}_m)\right]\delta t \\
\mathbf{R}^k &= \mathbf{R}^{k-1}\,\mathrm{Exp}(\widetilde{\boldsymbol{\omega}}^k - \mathbf{b}_{\boldsymbol{\omega}}^k - \mathbf{n}_{\boldsymbol{\omega}}^k)
\end{aligned}
\tag{7.11}
$$

This model is well adapted to humanoid robots with 6-axis force-torque sensors at their flat feet or quadrupeds with 3D force sensors at their (approximately) point feet. Indeed, for quadruped with point feet, no pure moment is applied by the ground of the robot. In this case, the same equations can be kept, simply setting the pure torques $\mathbf{m}$ terms to zero.

Note that the orientation of the system $\mathbf{R}$ appears in the centroidal states equations to relate the local kinematic and force measurement to the world frame in which we conduct the integration. We, therefore, have to include the dynamical model of the orientation by integrating gyroscope measurements $\widetilde{\boldsymbol{\omega}}$ together with forces and torques.

As illustrated in Section 6.1, recursively applying (7.11) and defining a naive difference in the world frame leads to an inefficient residual definition. Instead, we resort to the generalized pre-integration algorithm that we developed in Section 6.2.

7.3.2 Force-torque pre-integration on composite Lie group

This section follows the same structure and logic as the two IMU pre-integration algorithm described in Section 6.3.1 and Section 6.3.2 since we use the same generalized

pre-integration pipeline. The sensor measurements $\mathbf{z}^k$ involved in the integration are kinematics, gyro, and force-torques measurements:

$$\mathbf{z}^k = \left[\widetilde{\mathbf{p}}_C, \widetilde{\boldsymbol{\omega}}, \left[\widetilde{\mathbf{f}}_L, {}^L\widetilde{\mathbf{m}}_L, \widetilde{\mathbf{p}}_L, \widetilde{\mathbf{R}}_L\right]_{L=1..4}\right]^k \tag{7.12}$$

that we assume to be synchronized.

As we explained, $\widetilde{\mathbf{p}}_C$ and $\widetilde{\boldsymbol{\omega}}$ are corrupted by noise and biases. Thus, these biases $\mathbf{b} = [\mathbf{b}_c, \mathbf{b}_\omega]$ are added to the estimator. As we did in the IMU case, we assume them constant between two Keyframes and affected by a random walk drift.

Definition of the delta group

The state variables on which the pre-integration is applied are defined as $\mathbf{x} = [\mathbf{c}, \dot{\mathbf{c}}, \mathcal{L}, \mathbf{R}]$. Note that the rotation has to be included to be able to define residuals in the local frame.

The definition of delta and $\boxminus$ operator comprises two parts. For the force-torque pre-integration onto centroidal dynamics, we observe that a body subject to no force and torque will free-fall at the acceleration of gravity while keeping a constant angular momentum. For the orientation part, as we did for the IMU, a null angular velocity means a non-rotating frame. The deltas, defined as the motion between such free-falling frames and the current state, can therefore be written as:

$$\widehat{\boldsymbol{\Delta}}^{im} = \mathbf{x}^m \boxminus \mathbf{x}^i \triangleq \begin{bmatrix} \widehat{\boldsymbol{\Delta}\mathbf{c}}^{im} \\ \widehat{\boldsymbol{\Delta}\dot{\mathbf{c}}}^{im} \\ \widehat{\boldsymbol{\Delta}\mathcal{L}}^{im} \\ \widehat{\boldsymbol{\Delta}\mathbf{R}}^{im} \end{bmatrix} = \begin{bmatrix} \mathbf{R}^{i\top}(\mathbf{c}^m - \mathbf{c}^i - \dot{\mathbf{c}}^i \Delta t_{im} - \frac{1}{2}\mathbf{g}\Delta t_{im}^2) \\ \mathbf{R}^{i\top}(\dot{\mathbf{c}}^m - \dot{\mathbf{c}}^i - \mathbf{g}\Delta t_{im}) \\ \mathbf{R}^{i\top}(\mathcal{L}^m - \mathcal{L}^i) \\ \mathbf{R}^{i\top}\,\mathbf{R}^m \end{bmatrix}. \tag{7.13}$$

With this definition, the pre-integrated deltas are guaranteed to depend only on the sensor measurements and bias, and not on the initial state $\mathbf{x}^i$. The inverse operation $\boxplus$ is obtained as

$$\mathbf{x}^m = \mathbf{x}^i \boxplus \boldsymbol{\Delta}^{im} \triangleq \begin{bmatrix} \mathbf{c}^i + \dot{\mathbf{c}}^i \Delta t^{im} + \mathbf{R}^i \Delta \mathbf{c}^{im} + \frac{1}{2}\mathbf{g}\Delta t_{im}^2 \\ \dot{\mathbf{c}}^i + \mathbf{R}^i \Delta \dot{\mathbf{c}}^{im} + \mathbf{g}\Delta t^{im} \\ \mathcal{L}^i + \mathbf{R}^i \Delta \mathcal{L}^{im} \\ \mathbf{R}^i \boldsymbol{\Delta}\mathbf{R}^{im} \end{bmatrix} \tag{7.14}$$

Definition of the group operations

Given two centroidal deltas $\boldsymbol{\Delta} = [\boldsymbol{\Delta}\mathbf{c}, \boldsymbol{\Delta}\dot{\mathbf{c}}, \boldsymbol{\Delta}\mathcal{L}, \boldsymbol{\Delta}\mathbf{R}]$ and $\boldsymbol{\delta} = [\delta\mathbf{c}, \delta\dot{\mathbf{c}}, \delta\mathcal{L}, \delta\mathbf{R}]$, the group composition law $\boldsymbol{\Delta} = \boldsymbol{\Delta} \circ \boldsymbol{\delta}$ is defined as

$$\boldsymbol{\Delta} \circ \boldsymbol{\delta} = \begin{bmatrix} \boldsymbol{\Delta}\mathbf{c} + \boldsymbol{\Delta}\dot{\mathbf{c}}\delta t + \boldsymbol{\Delta}\mathbf{R}\delta\mathbf{c} \\ \boldsymbol{\Delta}\dot{\mathbf{c}} + \boldsymbol{\Delta}\mathbf{R}\delta\dot{\mathbf{c}} \\ \boldsymbol{\Delta}\mathcal{L} + \boldsymbol{\Delta}\mathbf{R}\delta\mathcal{L} \\ \boldsymbol{\Delta}\mathbf{R}\delta\mathbf{R} \end{bmatrix} \tag{7.15}$$

with a group identity element composed of the identity element of its components:

$$\boldsymbol{\Delta}_\mathcal{E} = \begin{bmatrix} \mathbf{0_3} \\ \mathbf{0_3} \\ \mathbf{0_3} \\ \mathbf{I}_3 \end{bmatrix}, \tag{7.16}$$

the resulting inverse being

$$\boldsymbol{\Delta}^{-1} = \begin{bmatrix} -\boldsymbol{\Delta}\mathbf{R}^\top(\boldsymbol{\Delta}\mathbf{c} + \boldsymbol{\Delta}\dot{\mathbf{c}}\Delta t) \\ -\boldsymbol{\Delta}\mathbf{R}^\top\boldsymbol{\Delta}\dot{\mathbf{c}} \\ -\boldsymbol{\Delta}\mathbf{R}^\top\boldsymbol{\Delta}\mathcal{L} \\ \boldsymbol{\Delta}\mathbf{R}^\top \end{bmatrix}. \tag{7.17}$$

The calibration function $c()$ and exponential map $\mathrm{Exp}()$ are combined to integrate one step of measurement $\tilde{\mathbf{z}}$, yielding $\boldsymbol{\delta} = f(\tilde{\mathbf{z}}, \mathbf{b}, \delta t)$ in (7.18) as

$$\boldsymbol{\delta}^k(\mathbf{z}^k, \mathbf{b}^i, \delta t) = \begin{bmatrix} \frac{1}{2m}\sum_l \widetilde{\mathbf{R}}_L^k\, \widetilde{\mathbf{f}}_L^k \delta t^2 \\ \frac{1}{m}\sum_l \widetilde{\mathbf{R}}_L^k\, \widetilde{\mathbf{f}}_L^k \delta t \\ \sum_l \left[(\widetilde{\mathbf{p}}_L^k - (\widetilde{\mathbf{p}}_C^k - \mathbf{b}_c^i)) \times \widetilde{\mathbf{R}}_L^k\, \widetilde{\mathbf{f}}_L^k + \widetilde{\mathbf{R}}_L^k\, \widetilde{\mathbf{m}}_L^k)\right] \delta t \\ \mathrm{Exp}((\widetilde{\boldsymbol{\omega}}^k - \mathbf{b}_\omega^i)\delta t) \end{bmatrix} \tag{7.18}$$

Delta pre-integration and factor residual

The concatenation of the operations above lead exactly the algorithm we implemented in [Fou+21]:

- Initialize $\boldsymbol{\Delta}_{ii} = \boldsymbol{\Delta}_{\mathcal{E}}$, $\Sigma_{\boldsymbol{\Delta}} = \mathbf{0}$, $\mathbf{J}_{\mathbf{b}}^{\boldsymbol{\Delta}} = \mathbf{0}$, and $\overline{\mathbf{b}}_i = \mathbf{b}_i$.
- Calibrate data and retract to manifold using (7.18).
- Compose $\overline{\boldsymbol{\Delta}}$ using (7.15).

The factor residual is again computed following Section 6.2.4 exactly.

Comment on the link with IMU composite pre-integration

The reader will have noticed the high similarity between the centroidal motion delta group defined here and the composite IMU delta Lie group defined in Section 6.3.1 (*e.g.* between equations (7.13) and (6.10)). If we only consider the center of mass, its velocity, and the orientation of the base equations, they are in fact the same. Indeed, the force sensors provide local second-order information on the CoM position, which is akin to the IMU acceleration measurements.

Where the problems differ however is in the inclusion of the angular momentum (the second part of the Newton-Euler equations (7.2)). This equation introduces the dependence on the CoM lever bias $\mathbf{b}_c$, which is also present in the centroidal kinematics residual (7.8). [Rot+15] showed that including the same set of measurements on centroidal quantities as us, the state space centroidal system with centroidal kinematics bias is made observable, provide we have a prior estimation of the base states. In our case, base states are state variables estimated in conjunction with the centroidal states. However, if the base states are made observable by other sensor sources, then the same observability analysis should apply to the centroidal states in our case. This is confirmed by experimental results presented in Chapter 9.

7.4 Conclusion

In this chapter, we have proposed the first step toward a whole-body estimator based on MAP. On the first hand, we have introduced centroidal states into the MAP problem, in addition to the previous decision variables, classical estimated in SLAM. The centroidal states are related to the basis states through the centroidal kinematics, which is typically used in state-of-the-art estimators. While the same relation is here used, the classical centroidal estimator only loosely couples these two estimated quantities, then being unable to exploit other observations on the centroidal states to also contribute to the estimation of the basis. Then we draw the relation between the force measurements and the centroidal states, which enriches the MAP factor graph and enables us to take advantage of the tight coupling between basis and centroidal states. The force sensors are related to the centroidal states in a somehow similar relation to the IMU is connected to the base state. As shown previously on more theoretical estimation work, the force sensors provide the observability condition to unbias the COM position despite (unavoidable) kinematic and inertial calibration problem of the robot model.

We will show in the experimental part of this thesis (Chapter 9) how this tight coupling can be implemented on a quadruped robot, although a significant experimental work remains to be done to demonstrate its interest in practice. As previously said, this contribution is a first step toward building a whole-body estimator, *i.e.* an estimator that would be able to estimate all robot related quantities by fusing any available sensors, in particular, grid sensors such as robot skin and distributed IMUs.

Part II

Applications

Chapter 8

Visual-inertial SLAM with fiducial markers

Contents

8.1 Introduction

In this chapter, we are looking for a solution to localize a humanoid robot indoors, with sufficient accuracy to navigate on some stairs, grasp a handrail, or walk on a 30-cm wide beam. As the robot is going to come back again and again in the same environment, we would like to benefit from loop-closure information and localization with respect to some known landmarks. While our final goal is to merge in the optimal estimator the measurements coming from all the sensors of the robot, we focus here on contributions validating the use of visual-inertial localization and mapping on a humanoid robot navigating indoors in a 3D environment.

For the visual factor, we rely on AprilTags [WO16], while proposing a practical contribution to avoid ambiguity issues in the pose estimation of the tags, as described in Section 4.2. For the inertial factor, we build upon Forster pre-integration [For+17] and propose an original, by exhibiting a compact Lie group (described in Section 6.3.2) that is suitable for optimal estimation. This formulation, although leading to very similar formulas for the inertial factors, enables a generalization to the other high-frequency factors that would typically arise in the humanoid contact (leg odometry based on coders, force

sensors, etc). Both inertial and visual factors are processed in a factor graph formalizing a MAP problem.

This first application is typically addressed in the literature as Visual-Inertial estimation and expressed by various approaches in several robotics domains. We will first discuss the state-of-the-art related to this experimental context to more accurately define the experimental stakes.

We then formalize the inertial SLAM estimation problem using factor definitions given in the previous part. Most of the chapter will be presenting experimental results on datasets obtained with a visual-inertial sensor (VIS) carried by a human operator and the HRP-2 humanoid robot.

8.2 Related works

The difficulty in fusing inertial, kinematics, and exteroceptive measurements stems from the disparity in the properties of each data source. Inertial and kinematic measurements come at high frequency (typically 100 Hz to 1 kHz) and are cheap to process, while images and laser scans are obtained at some few frames per second and are expensive to process. On the other hand, inertial measurements are quickly deprecated while images and scans provide absolute information. This implies a rigorous synchronization between the sensors with the risk of decreasing the performances of the inertial estimation when images and laser scans are not carefully merged.

These difficulties explain that the first works to merge proprioceptive and exteroceptive sensors for legged localization have been with staggered approach, first fusing inertial and kinematic measurements at high frequency, and then correcting the localization drift with absolute localization computed from the camera and/or the LIDAR with low bandwidth and higher delay [Nob+17; Fal+14].

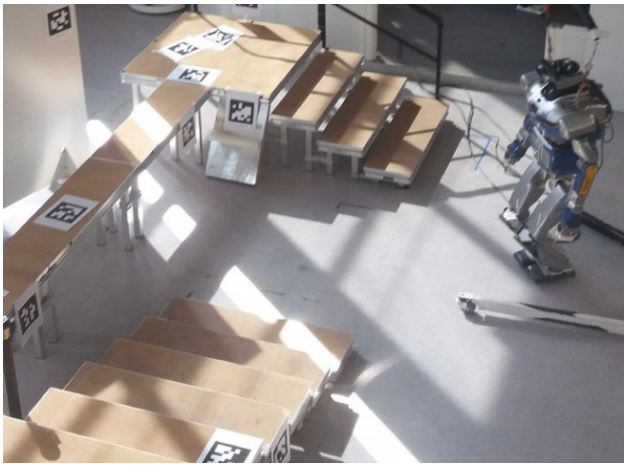
Quite recently, several concurrent approaches have been proposed to merge all relevant data in a unique estimator. Following the recent results in UAVs localization [For+17; Leu+15], optimal estimation structured by a factor graph seems to be a suitable framework to formulate the fusion. In [Har+18c], a graph-SLAM is proposed to fuse inertial, kinematics, and visual data. Inertial measurements are considered using Forster's pre-integration factors [For+17], which is recalled in Section 6.3.1. Kinematics data are included using a 6D factor which is also pre-integrated, taking into account the hybrid nature of the contact dynamics using an event-based approach. Visual factors are also expressed as 6D factors obtained by visual odometry. Results are reported on sequences of a few meters with motion-capture ground truth. In [WCF19], the graph-SLAM also considers inertial measurements through Forster's pre-integration, while kinematic measurements are pre-treated by the robot low-level system [Blo+17] and integrated directly as 6D factors without further consideration. Finally, the visual information is included as 2D pixel reprojection factors in the image space, obtained from feature tracking (KLT [BM04]). Impressive experimental results are demonstrated with long outdoor sequences, using a ground truth obtained from off-line LIDAR reconstruction.

The pros and cons of these two approaches come from the choice of the factors, but the similarities are possibly more important than the differences. Both use a plain Forster pre-integration [For+17]. Using either visual odometry or feature tracking, both systems cannot natively benefit from the information brought by loop closure and would fail to exploit known map information. In both cases, the kinematic factor is straightforward to write as a 6D probabilistic constraint. Finally, both works can account for the very

different sensor frequencies, while providing a good estimate at the higher frequency if needed.

As for the specific problem that we wish to solve in this chapter, two AprilTag based visual-inertial SLAM systems have been implemented in the previous years. In [NBB16], the authors rely on an EKF in which state propagation is naturally handled by the IMU and each marker detection is used in an update step where the reprojection error of its 4 corners provides an 8D innovation vector. A closer solution to ours was very recently proposed in [HZG19] and is also based on graph SLAM optimization benefiting from Forster's IMU pre-integration from GTSAM. As explained previously, the AprilTag factor formulation is different from ours and the algorithm is tested on large datasets consisting only of smooth motions.

(a) HRP2 robot with which were conducted the experiments. The head was replaced by our visual-inertial sensor.



(b) Experimental room with parkour, HRP-2 and fiducial markers.

Figure 8.1: Experimental setup. (a): installation of the visual-inertial sensor. (b): experimental space.

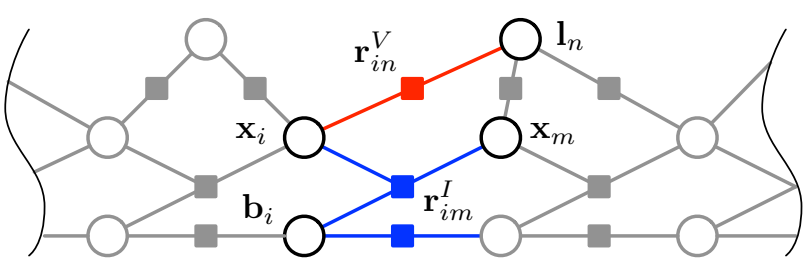


Figure 8.2: Factor graph supporting the estimator used in this chapter, involving state blocks corresponding to keyframes $\mathbf{x}_i = (\mathbf{p}_i, \mathbf{v}_i, \mathbf{R}_i)$, biases $\mathbf{b}_i$ and landmark poses $\mathbf{l}_n$. IMU factors (**blue**) relate consecutive Keyframes and the IMU biases. The lower branch controls bias drift along time. Visual factors (**red**) relate landmarks with poses $(\mathbf{p}_i, \mathbf{R}_i)$.

8.3 Problem statement

As mentioned in the tutorial on MAP estimation (Chapter 3), the problem is well represented as a bipartite graph, where one type of node refers to the variables, and the other type called *factors* represents probabilistic constraints between variables, produced by the measurements. In the case of landmark-based visual-inertial SLAM (see Fig. 8.2), $\mathbf{x}$ includes robot poses and velocities $\mathbf{x} = (\mathbf{p}, \mathbf{v}, \mathbf{R})$ and IMU biases $\mathbf{b}$, both at selected Keyframes along the trajectory, and landmark poses $\mathbf{l} \in SE(3)$. Biases are considered constant between Keyframes. In line with the recent works on the subject, we write the MAP optimization as the least-squares minimization (Fig. 8.2),

$$\mathcal{X}^* = \arg\min_{\mathcal{X}} \sum_i \left\| \mathbf{r}_i^I(\mathcal{X}) \right\|^2_{\mathbf{\Sigma}_i^I} + \sum_j \left\| \mathbf{r}_j^V(\mathcal{X}) \right\|^2_{\mathbf{\Sigma}_j^V} , \tag{8.1}$$

with $\{\mathbf{r}^I, \mathbf{\Sigma}^I\}$ and $\{\mathbf{r}^V, \mathbf{\Sigma}^V\}$ indicating the residuals and covariances of respectively the inertial (IMU) and visual factors. These residuals are computed differently depending on the nature of the measurements and the state blocks they relate to. The AprilTag factor is described in this thesis Chapter 4 and the IMU factor in Chapter 6. The factor graph is then implemented in the WOLF framework [Sol+21], using Ceres [AM+] as the backend solver. A ROS-based demo with installation instructions can be found in this page.

8.4 Experimental setup

We have gathered several datasets in the experimental arena of the humanoid robots at LAAS-CNRS, a 3D environment about $10m \times 5m$ made of flat floor, stairs, and a 30cm wide beam. The robot environment was augmented with about 20 fiducial "AprilTag" markers (of about 20 cm width). The tags have been randomly dispatched in the environment. They are fixed during a run but may vary significantly between two sets of data, and their locations are not calibrated —that is, we do not have ground truth localization of the tags.

Each dataset is composed of 3 sequences:

- a sequence of RGB images captured at 33 Hz
- a sequence of IMU measures captured at 200 Hz
- a sequence of motion-capture (MoCap) at 200 Hz measurements used as ground truth.

The visual-inertial sensor (VIS) is comprised of a Memsic IMU running at 200 Hz and an Imagine Source camera. IMU and camera are hardware synchronized: the image acquisition is triggered by a micro-controller (STM32) synchronized with the IMU. We have validated that there is less than a 2 ms synchronization error by the hardware (shutter time) and that this delay is stable. The camera and the IMU are collocated, with less than 10 cm of distance between IMU and camera focal. The camera-to-IMU extrinsics parameter was calibrated using the Kalibr library [FRS13]. In each sequence, we have taken care that the camera is navigating in a comfortably-dense field of tags, even if it may not have always a tag in its field of view. The motion-capture data have been obtained from a calibrated 3D marker attached to the camera and are synchronized in post-process by maximizing the velocity norm cross-correlation between MoCap and estimated state sequences.

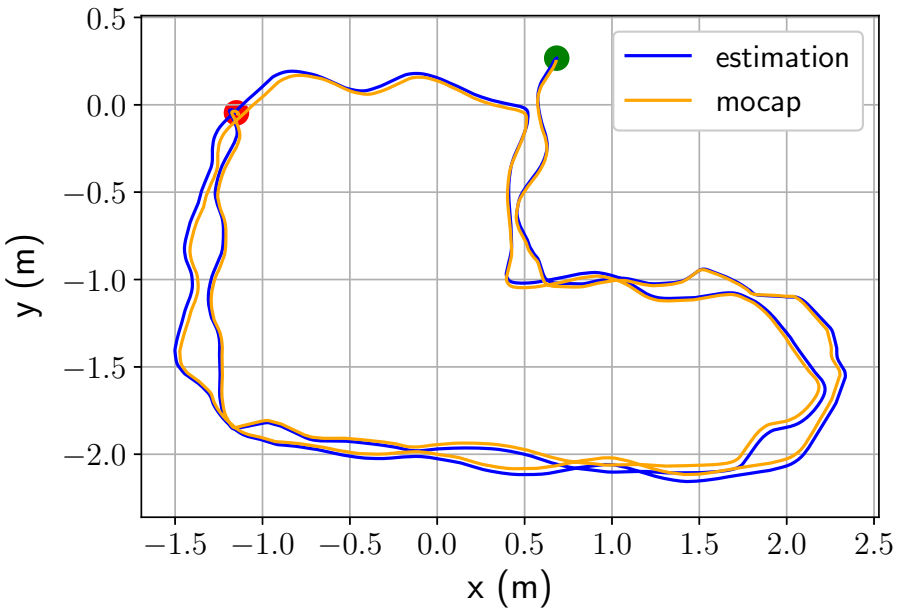


Figure 8.3: Two loops of the experimental field with camera in hand

Table 8.1: Datasets description and results

Description	Duration	Length	MTE[1]	STE[2]
Handheld loop	59.0 s	20.6 m	29.0	11.3
HRP2 turns then walks	59.9 s	12.87 m	30.9	15.7
HRP2 climbs stairs	47.1 s	6.25 m	13.9	6.3
HRP2 descends stairs	19.39s	2.62m	30.4	11.8

[1] Mean translation error [mm]
[2] Std. dev. of translation error [mm]

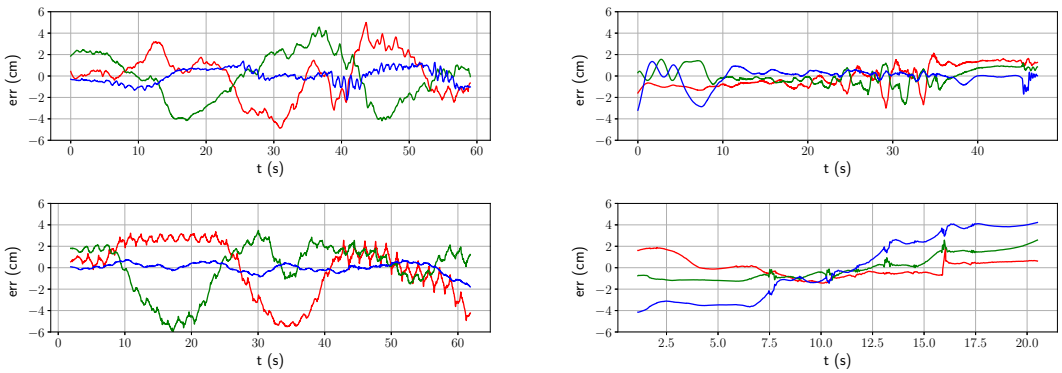


Figure 8.4: Translation estimation error (cm) as a function of time (s). In the clock-wise sens, starting from the top-left corner, the datasets are handheld camera, stairs climbing, stairs descending and walking on flat ground. RGB colors correspond to xyz axes.

8.5 Results

8.5.1 Absolute localization

We consider four datasets that are summarized in Table 8.1. They cover different tasks on which a consistent estimation of the robot movement is necessary. The first one is a relatively long sequence consisting of two loops with the VIS handheld. This is used to test the long-term localization of the robot, which is interesting for navigation. Secondly, we made the LAAS Gepetto team HRP-2 walk and turn around on a short distance to evaluate the resilience of the filter to the vibrations of the robot. Finally, two more challenging datasets are recorded while the robot is climbing and descending stairs. Especially on the latter, the locomotion causes impacts that on one hand bring the IMU close to its dynamic range saturation, and on the other hand, provoke images with motion blur. Note that during these experiments, the estimator was not used for feedback control. To compare our results with the ground truth, we used methods described in [ZS18] to align trajectories given that 4 DoFs are unobservable in VI estimation. For each case, Keyframes are created at a frequency of 6.6 Hz (every 5 images) if tags are detected in the corresponding image.

Fig. 8.4 presents a quantitative evaluation of the translational errors. In all cases, our estimator achieves errors consistently below a few centimeters. The biggest errors are obtained for the walking datasets where the two humps correspond to phases where the robot is turning on itself and sees landmarks that will not be seen again later in the trajectory.

8.5.2 High-rate velocity estimation

A high rate estimation of a humanoid robot velocity and in particular of its center of mass is critical for balance controllers. It can be recovered from motion capture through numerical differentiation of the positions, but this results in a quite noisy time series. It is especially visible when hard impacts make the robot when descending stairs for instance.

Fig. 8.5 presents a zoom-in one part of the stairs descending trajectory. Here, within 2.5 seconds, HRP-2 lowers its base and then its foot touches the next step. The velocity is thus first negative until the impact of the foot on the step. Then, the estimated velocity follows a familiar oscillatory damped system behavior while the mocap estimation is more erratic.

The high rate estimate of the velocity is obtained by integrating IMU measurements with optimally estimated biases from the last Keyframe optimized by the solver, as explained in Section 6.2.5. The smoothness of the trajectory is a good indicator that the problem has converged to a consistent state.

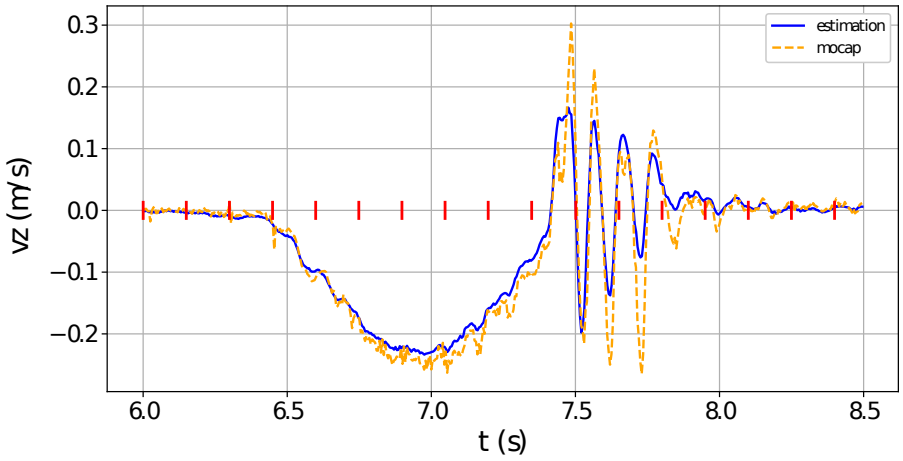


Figure 8.5: Descending stairs (one step): the robot first lowers its base and then touches the next step with results in vibrations from the impact. Vibrations at approximately 10 Hz visible in both the motion capture ground truth and optimal state estimation. Note that the estimation in this case has a smoother curve. Keyframes are introduced at 6.6 Hz (every $\approx$ 150 ms) and are represented by vertical red bars. State in between is obtained by applying the pre-integrated delta from the last Keyframe to the current time to the estimate of the last Keyframe state estimate (see Section 6.2.5).

8.6 Conclusion

This chapter reports our first experimental results, validating our IMU observation model (using pre-integration), that we assembled into an observable estimator using fiducial markers. The implementation reproduces what now is a classical formulation of visual-inertial odometry, but has been the first demonstration on a humanoid at the time of publication, to the best of our knowledge. We have been able to obtain an accurate estimation of the humanoid robot HRP-2 in a 3D terrain. The IMU enables the estimator to have excellent accuracy and high bandwidth in the resulting estimation. In particular, the impact of the feet on the stairs, and ensuing vibrations, clearly appears as a relevant movement in the estimator output. On the other hand, the fiducial markers enable the filter to display a global localization, by closing the loop when previously-seen markers are observed again while the robot walks along a loop in the experimental room. This experimentally validates the relevance of fusing these two information sources for localizing a legged robot. In particular, the filter provides a full consistency, on the opposite to loosely-coupled filters where the global localization (SLAM-like) is not guaranteed to be consistent with the local estimation of the base state.

Since this work, a few interesting MAP-based estimators have been deployed on legged robots, as we are going to discuss in the next chapters. For the implementation we reported here, we mostly missed some visual odometry (*i.e.* low level geometrical visual features) that would have helped the estimator to have a better mid-frequency accuracy (between the high-frequency provided by the IMU and the low-frequency provided by the sparse fiducial tracker). The experimental results were running in real-time by using a fixed window of Keyframes active in the estimator and removing the older ones. The approach may be improved by introducing a marginalization procedure as done in some classical visual-inertial systems [Leu+15].

In this chapter, we have not yet added the information coming from the foot contact. Moreover, the fiducial markers are not strongly related to any relevant part of the robot environment, which makes it difficult to use the localization information in a contact planner. In the next chapters, we are going to show, on one hand, how this estimator extends to fully account for the contact information and, on the other hand, how it may obtain information about parts of the environment, that a motion planner could directly use, by replacing the visual front-end.

Chapter 9

Centroidal estimation

Contents

In this chapter, we propose a tightly-coupled estimator of the base and centroidal states that fuses IMU, kinematics, centroidal kinematics, and force-torque measurements. This work was first presented in our published paper [Fou+21].

9.1 Introduction

As mentioned in the literature review, centroidal states are key to the control of legged robots. Indeed, they provide rich information about the general behavior of the system and can be used to check for the stability of the system. The centroidal state estimation literature is rich, especially for humanoid robots (see Section 2.3). To our knowledge, all of the centroidal estimators proposed in the literature can be classified as loosely-coupled estimators (following the terminology introduced in Section 2.2.2): they rely on prior knowledge of a base state. For instance, Piperakis [PKT18] describes the whole pipeline: first, an inertial-kinematics EKF estimates the base state, then an EKF uses this fixed base state to obtain centroidal states. Since the factor graph optimization framework that we use theoretically reaches its full potential when the maximum number of cross-correlation are considered, we did not find this solution satisfactory. Here, we rather propose a tightly-coupled estimator that jointly estimates base and centroidal states using IMU, kinematics, and force measurements.

To the best of our knowledge, this is the first time an estimator tightly couples forces, IMU, and proprioception to estimate both base and centroidal quantities. This estimator

finds its roots in visual-inertial SLAM as described in the previous chapter, and in the observability studies of centroidal quantities, already discussed in Section 7.1. We then skip here a discussion of related works which has been sufficiently covered, to directly describe the underlying factor before exhibiting the experimental results.

9.2 Problem statement

We define here an estimator capable of observing both base states (position, orientation, velocity) and centroidal states (CoM, CoM velocity, angular momentum) from proprioception only. We assume that the following measurements are available on the robot: IMU data, kinematics (leg odometry), centroidal kinematics, and force-torque sensors. Since the IMU measurements (6.2) and CoM local position from the kinematics (7.6) are biased, we also need to integrate them to the estimated variables.

Bloesch [Blo+13b] showed that the base states of legged robots and IMU biases are observable using a tightly-coupled estimator based on IMU and kinematics measurements. On the other end, Rotella [Rot+15] showed that the centroidal state of a robot and a bias on the centroidal kinematics can be obtained by fusing force-torque sensor measurements with centroidal kinematics, based on prior knowledge of the base states. We join the two problems into a single state estimation problem, represented by the factor graph of Fig. 9.1.

In this estimator, we have two motion sensors, IMU and force-torque, whose measurements we pre-integrate as explained in Section 6.3.1 and Section 7.3 respectively. Base states and centroidal states are tightly constrained by the centroidal kinematics factor, which relates almost all estimation variables at a given Keyframes.

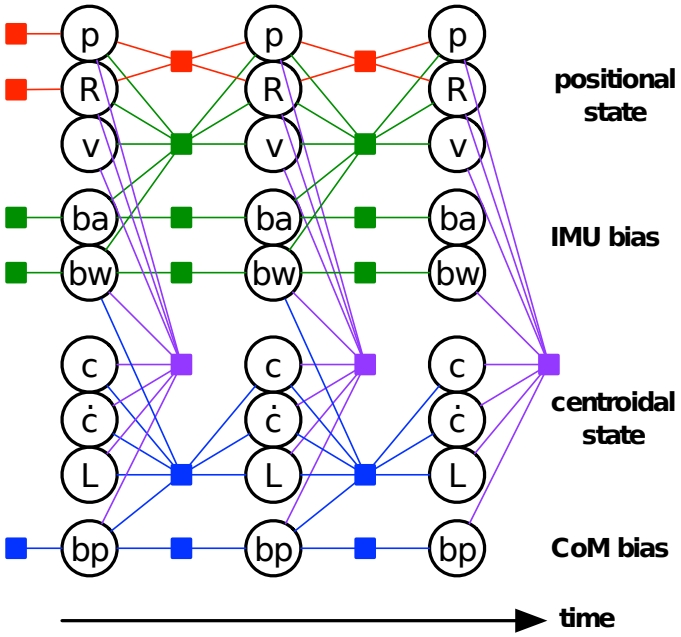


Figure 9.1: Factor graph representation of the tightly-coupled base-centroidal state estimation problem. Each round node corresponds to an estimated state variable. Each square corresponds to a factor, that is a measurement residual. The colors of the residuals are as follows. **Red**: leg odometry (Section 5.2), **green**: IMU pre-integration and IMU bias drift (Section 6.2.4), **purple**: centroidal kinematics (Section 7.2), **blue**: force-torque pre-integration and centroidal kinematics bias drift (Section 7.3).

9.3 Experimental setup

This estimator is implemented in WOLF [Sol+21], with necessary kinematics and dynamics quantities computing with Pinocchio software [Car+19], and is experimentally validated on the Solo-12 quadruped robot.

As is often the case with quadruped robots, Solo-12 is not equipped with three-axis force sensors at its feet. Yet, in order to validate the present method, it is possible to reconstruct the contact forces based on the robot dynamics equation (7.1). Knowing the robot configuration and derivatives $\mathbf{q}, \mathbf{v}_q, \dot{\mathbf{v}}_q$, and joint torques $\boldsymbol{\tau}$ (from motor currents), recovering forces from this equation results in solving an over-determined linear system. Some of these quantities are hard to obtain directly since they depend on the state being estimated (*e.g.* base orientation) or on numerical differentiation ($\ddot{\mathbf{q}}_a$). For these reasons, we pre-calculated these forces by benefiting from an internal filter of the 3DM-CX5-25 IMU for the base orientation, centered window differentiation of encoder speed measurements for the articulation acceleration $\ddot{\mathbf{q}}_a$, and neglecting the influence of linear velocity. Fig. Fig. 9.2 shows an example of the force reconstruction of one leg using the robot proprioceptive sensors.

This movement and another one are displayed in the video https://peertube.laas.fr/videos/watch/16822d27-3557-4e35-9a0d-ce5b0aea4c27. The configuration trajectories were obtained using task space inverse dynamics [Pre+16] and applied to the robot with joint-level admittance control.

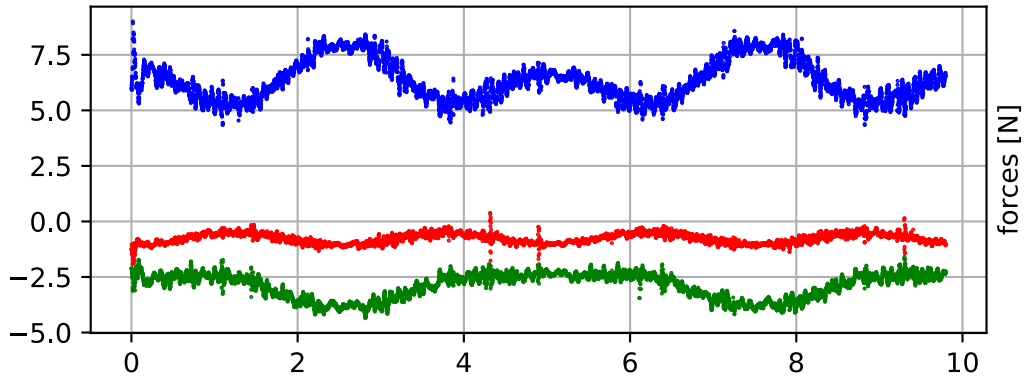


Figure 9.2: Force estimation on X-Y-Z axis (r-g-b) of one Solo-12 leg expressed in world frame using proprioceptive sensors during the *sinXYZ* trajectory

9.4 Results

9.4.1 Base estimation through inertial kinematic fusion

First, to validate the use of our kinematic factor, we include uniquely the IMU and leg-odometry factors to obtain an Inertial Kinematics estimator which conceptually includes the same information as estimators such as [Blo+13b]. In Fig. 9.3, we compare our state estimation at 1 kHz with motion capture (Mo-Cap) up-sampled from 200 Hz to 1 kHz. Velocity in the base frame is also shown in Fig. 9.4.

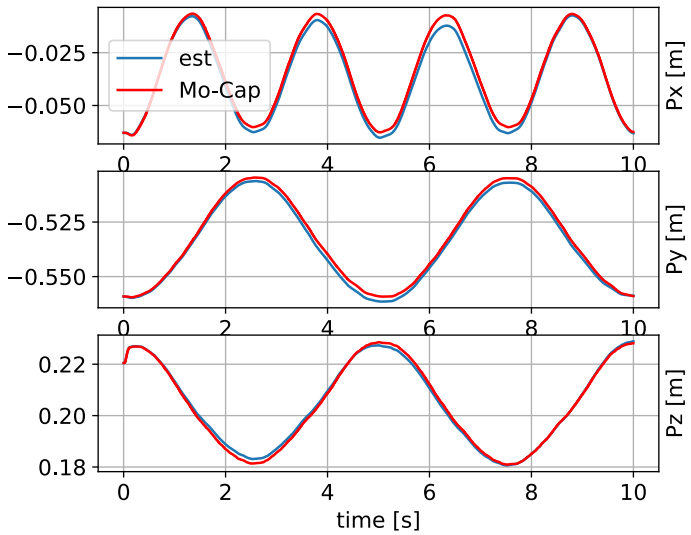


Figure 9.3: *sinXYZ* trajectory base position from the IMU+Kinematics (IK) estimator (blue) vs Mo-Cap (red)

Artificially removing contact factors (considering only 1, 2, or 3 feet in contact) can help us gain confidence in the use of this kinematic factor in situations where we rarely have all feet in contact, like for example with trotting gaits. In Fig. 9.5, we can see that only considering 1 foot in contact during the whole trajectory results in a drifting position, but as soon as 2 or more feet are in contact, the system is constrained enough for the drift to remain below around 5mm on all axes.

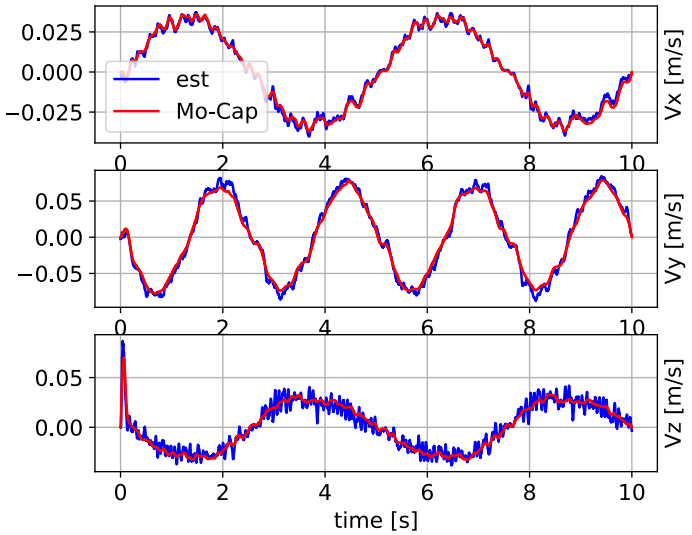


Figure 9.4: *sinXYZ* trajectory base velocity in base frame from the IMU+kinematics estimator (blue) vs Mo-Cap (red)

9.4.2 Centroidal estimation

Now, on the same trajectory, we deploy the full estimator with all factors described in the factor graph in Fig. 9.1 to jointly estimate the base and centroidal quantities. A ground truth on the centroidal quantities is difficult to obtain since no direct sensor can provide us with this information contrary to the base state. We can however validate our method by comparing it to a two-step procedure: first, estimate the base state with a state Kalman filter as implemented in [Ble+18a], then compute the centroidal quantities directly from the robot kinematic model. The full estimator should be able to infer a bias on the ${}^{B}\tilde{\mathbf{p}}_{C}$ measure so we artificially add a constant disturbance in the robot dynamic model on the lever of the base link of [0.03, 0.06, 0.04] cm, which then corresponds to a CoM bias of [-0.0197, -0.0394, 0.0263] cm. Fig. 9.8 shows that the bias estimated with our method closely matches the introduced bias. Fig. 9.9 shows a comparison between the base and CoM reconstruction with our method and with the two-step base Kalman filter with geometric CoM. Note that the base-CoM difference on the z-axis reflects the fact that the limbs of the robot naturally lower its CoM. The estimated CoM velocity closely follows the velocity of the base as shown in Fig. 9.7.

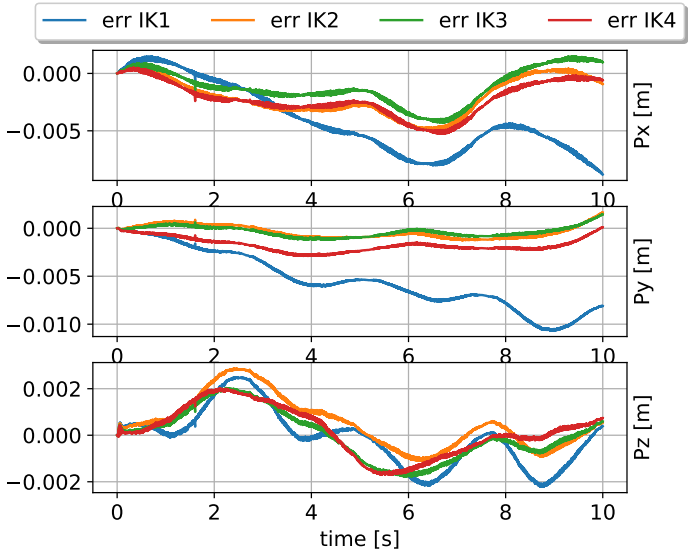


Figure 9.5: *sinXYZ* trajectory base position error with different numbers of feet used for the leg-odometry factors: 1 (blue), 2 (orange), 3 (green), 4 (red) from the IMU+kinematics estimator

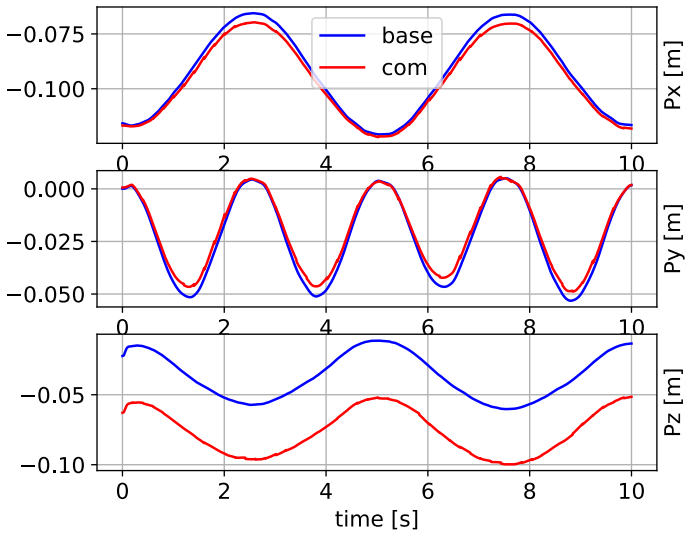


Figure 9.6: Base position (blue) vs CoM (red) from the full estimator

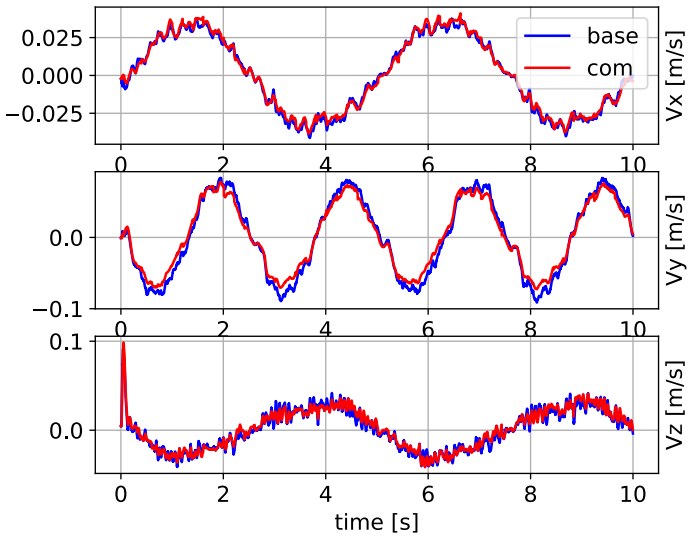


Figure 9.7: Base (blue) vs CoM (red) velocities from the full estimator

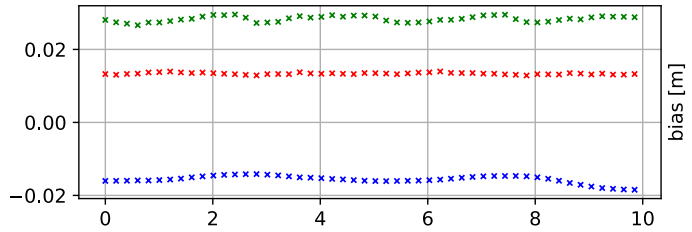


Figure 9.8: Estimation of bias on CoM measurement from the full estimator along x-y-z axis (red-green-blue) in base frame

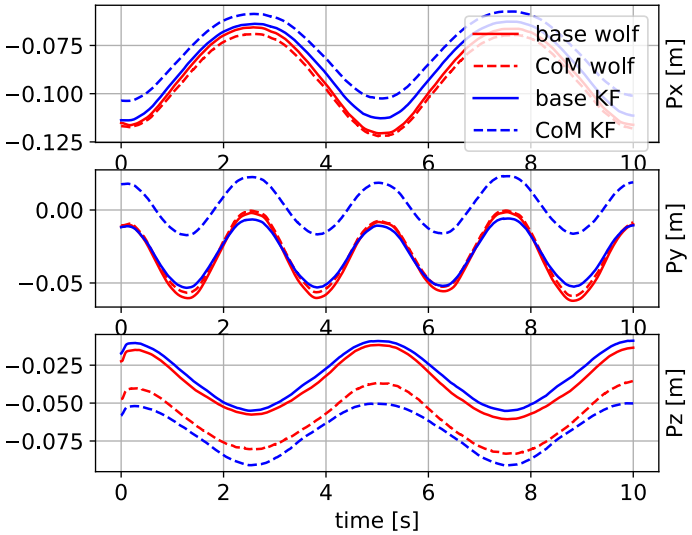


Figure 9.9: Comparison of the estimates between decoupled the Kalman filter base estimator and geometric CoM reconstruction (blue) and the tightly-coupled estimator presented in this paper (red) on the *sinXYZ* trajectory with artificial base link CoM bias.

9.5 Discussion

On the theoretical side, the correlation of the IMU and forces pre-integration models (due to the common gyroscope data integration) might also be investigated. One of the hypotheses involved in factor graph MAP estimation is the conditional independence of measurement data (see (3.11) in Section 3.1 for more details). Using the same measurements for both factors violates this assumption. Our model is not an exception in MAP estimation: for example, [WCF19] uses integrates the velocity output of an inertial-kinematic filter to obtain an odometry factor, while using IMU data in an IMU-preintegration factor. In our case, a solution would be to derive an algorithm pre-integrating IMU and force data at the same time. This choice would be at the cost of the modularity of the use of two separate factors.

While the results correspond to the expectation, several aspects of the experimental protocol would benefit from improvements in the future. First, the experimental platform we used is not the most appropriate to implement this type of estimator. Indeed, Solo-12 does not have force-sensors at the feet, which required the force estimation procedure described in Section 9.3. The inconvenience of this method is two-fold. Firstly, the estimated forces make heavy use of the IMU acceleration measurements which makes the forces pre-integration correlated with IMU pre-integration. Secondly, the force estimation gave us good performances only for rather slow trajectories, which prevented their use for walking trajectories for instance. We could have turned our interest to our humanoid robot Talos (see Fig. 2.1) which features force-torque sensors at its feet. However, at the time, Solo-12 was still a safer choice for several issues with Talos, that have since been partially addressed (kinematic calibration, higher flexibility of the structure, biased feet force sensors).

Nevertheless, Solo-12 allowed us to easily generate datasets based on simple quasi-static trajectories. We since recorded more datasets with walking trajectories based on the walking controller [Léz+21], including the robot proprioceptive measurements as well as images from an onboard camera (See Chapter 11 for more details). Evaluation of our inertial-kinematic estimator on these new trajectories is ongoing (see Chapter 11).

9.6 Conclusion

To the best of our knowledge, the work presented in this chapter corresponds to the first demonstration of the feasibility of tightly coupling the estimation of both the robot basis and its centroidal state. Beyond the practical limitations of our experimental setup, it opens the road to more ambitious filters able to provide a consistent by design estimation of the quantities needed for balance control and navigation, which we believe to be essential for reaching safe locomotion on 3d terrains. Our main effort is now to extend this work to also integrate advanced exteroceptive measurements able to provide meaningful information about the surrounding environment, as will be explained in the next chapter. We then see exciting perspectives in extending this estimator to multiple other information sources on the robot's whole body, to tightly couple the estimation of a more complete state.

Chapter 10

Cosy SLAM

Contents

In this chapter, we present an object-level visual-inertial SLAM system based on the deep-learning-based pose estimation CosyPose [Lab+20]. We present experimental results from our paper [Deb+21].

10.1 Introduction

Navigation of legged robots using onboard exteroceptive sensors has gained a lot of traction in recent years due to their progressive deployment for industrial applications [Bel+18]. For repeated travels, the map-less teach and repeat methods [FB10; Mat+21] avoid the need for a metric and globally coherent localization by benefiting from the knowledge of a human operator. This works very well for applications in which such supervision is available, and a global map is not. Building a metric and semantic map of the environment may be useful to automatize navigation and exploration in larger environments. Besides, standard objects whose CAD model is known (such as gauges, valves, stairs, etc.) may often appear.

Many representations of the environment are possible depending on the needs of the system. In [Fal+14] a prior map defined as a LIDAR point cloud is used in a Gaussian particle filter to localize a humanoid robot and perform online foot planning. Fankhauser [Fan+14] takes as an input an external odometry source to produce efficient robot-centric elevation maps while [Kim+20] builds a global height map to navigate through cluttered environments. Other approaches [WCF21] rely on a tight fusion between proprioceptive

and exteroceptive sensors to make the odometry more robust. A full overview of these systems can be found in our literature review (Section 2.4).

Figure 10.1: Experimental setup: a RealSense D435i is mounted on the Solo robot that localizes itself with respect to stairs. A motion capture system provides ground truth of the robot pose.

These approaches build metric maps that do not usually leverage the presence of known assets in the scene, although a few examples in the computer vision literature exist. In [Sal+13], the authors develop one of the first object-level SLAM algorithms from a depth sensor. Using a voting process based on point cloud descriptors, a simultaneous recognition and pose estimation of known objects was performed and included as factors in a graph optimization estimator. The method benefited from an active search of the objects in the scene, the detection being done in the SLAM loop. Aside from the robot trajectory and poses of objects, [SWD20] also proposes to optimize the object shapes using a differentiable rendering engine. In such approaches, objects need to be detected, classified, and their relative pose with respect to the camera has to be integrated into the estimator. On the other end, a work like [PL15] uses a semi-dense mono camera SLAM algorithm to produce a scale-ambiguous feature map. Then, a descriptor-based multi-view object proposal is performed as a post-processing step.

As described in Section 4.3.1, deep-learning-based object detection systems have now reached an accuracy that makes them candidates for mobile robotics applications. Applications range from robot manipulation, like sorting known objects, or localization with respect to known assets. For this last application, however, the direct output of CosyPose is not sufficient for two reasons. First, many objects have strong symmetries, which makes CosyPose orientation estimation jumps from one to the other depending on the frames. This output has, therefore, to be filtered using prior knowledge about the world or the robot's movements. Second, the robot needs to keep a memory of objects it has seen when they go out of its field of view.

We present in this chapter a practical implementation of the proposed MAP formulation fusing inertial measurements and object-level visual features estimated by CosyPose.

The stake is to demonstrate that a centroidal filter is able to merge information typically used for balance control (IMU related) with features typically needed by a high-level planner, *e.g.* the stair steps locations extracted by the object pose estimator. Beyond the practical validation for legged robots, this also demonstrates that the single view CosyPose can be extended to a sequential tracker able to benefit from complementary measurements such as an IMU.

To integrate CosyPose measurements with other sensors, we used the noise model based on empirical data, presented in Section 4.3.3. We also detail here the pragmatic implementation of heuristics to circumvent outliers in the network output. Experimental validations were conducted with a visual-inertial system, first handheld then mounted on a quadruped robot. Finally, we fine-tuned the pre-trained models to perform stairs localization, as explained in Section 4.3.4.

10.2 Implementation of the Visual Inertial filter

10.2.1 Factor Graph formulation

This section will be very short so as not to repeat ourselves: the problem has the same structure as the AprilTag-IMU VI system presented in Chapter 8, thus it as the same factor graph Fig. 8.2. The AprilTag pose measurement model is replaced by the CosyPose measurement model, presented in Section 4.3.

10.2.2 Data association and Outlier rejection

A key part of our SLAM system is the association of landmarks with the rejection of erroneous pose estimates. First of all, each object is associated with a label α so that a detection can only match a landmark with the same label. Then, the position of the robot is propagated by integrating the IMU measurements with the current biases estimates. Thus, each detected object pose can be transformed in the world frame using the propagated robot state. We check if this pose is similar to the one of a landmark with the same label with a threshold on the distance between the poses in $SE(3)$. If a detection does not match any landmark then a new landmark is created.

CosyPose can return poses of objects that are not included in the scene because of false detections, of Mask-RCNN, or wrong pose estimations (most often due to object symmetries). To handle these outlier detections, each landmark is associated with a score c that corresponds to its repeatability over time:

$$c = \frac{n_f}{\Delta t} \tag{10.1}$$

Δt is the time since the landmark initialization and n_f is the number of factors associated with it. The lowest scores are filtered with a threshold determined empirically and the associated landmarks are removed from the map.

10.3 Experimental validation

We have produced datasets in the robotic experimental arena at LAAS-CNRS in Toulouse. This is a 3D environment about $10m \times 5m$ made of flat floors, stairs and beams. The robot

environment was augmented with objects of the datasets that were used to train CosyPose. Each dataset is composed of three data sources:

- A sequence of RGB images (30 Hz)
- A sequence of IMU measurements (200 Hz)
- A sequence of motion capture (MoCap) measurements (200 Hz), used as ground truth

We recorded two types of datasets: one for the uncertainty models and one for SLAM experiments. For the uncertainty models, reflective MoCap markers were attached to the object to obtain the ground truth of their pose. For the SLAM, only the camera was tracked. We used the monocular RGB camera and the Bosh BMI085 IMU of an Intel RealSense L515 Camera for handheld trajectories. The Intel RealSense d435i was used with the same modalities for the experiments on the quadruped robot Solo [Gri+20] as shown in Fig. 10.1. The extrinsic calibration between the IMU and the camera was provided by Intel and the delays observed between IMU and Camera measurements were negligible. Our datasets are publicly available at https://homepages.laas.fr/mfourmy/icra22_cosyslam.

10.3.1 Object level VI-SLAM

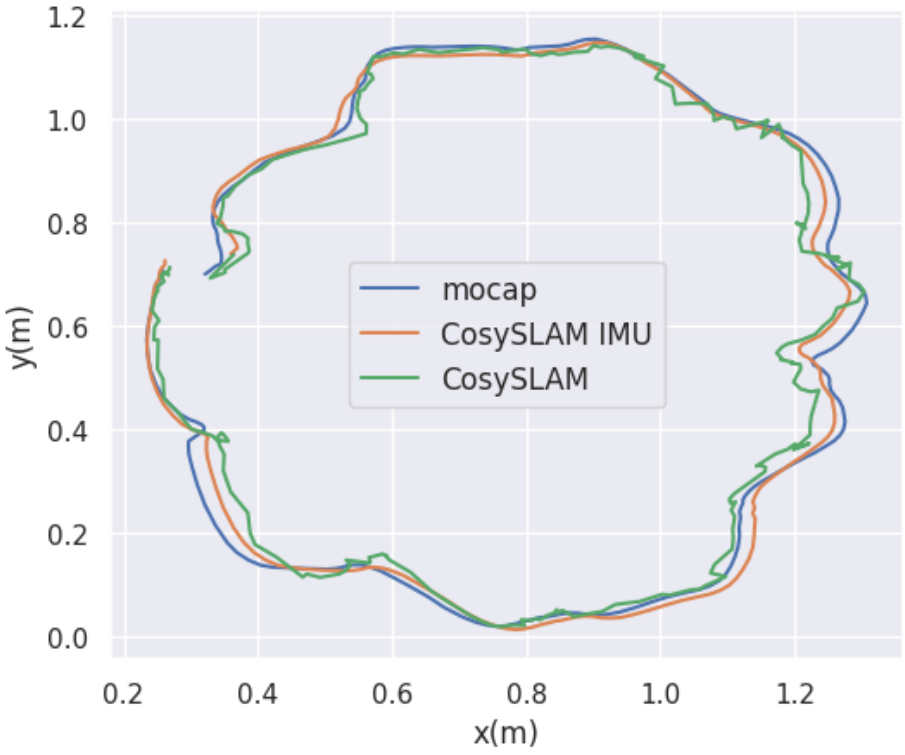


Figure 10.2: Comparison between the MoCap, the output of CosySLAM with visual factors only and the output of CosySLAM with IMU fusion on the circular trajectory.

In order to validate the performances of the fusion of CosyPose estimates and inertial measurements, we evaluated three scenarios with the camera held by hand and T-LESS objects[1] in the scene. The first one is a short and slow trajectory, *i.e.* an ideal scenario. The second one is a slow but long trajectory, to validate the consistency of our system over time. The last one is a highly dynamic scenario with a lot of motion that can blur some frames and lose sight of objects for more extended periods. Moreover, T-LESS objects being the most difficult objects for pose estimation with CosyPose, they may return many outliers and noisy measurements. This is therefore a challenging dataset to test the robustness of our algorithm. Keyframes are selected at 10 Hz, only if objects are detected in the images.

Table 10.1: Datasets description and results of the hand held videos

Scenario	Length(m)	Duration(s)	MTE[1](cm)	STE[2](cm)
V-only - Circular	3.7	23.7	3.8	1.6
V-only - Short	2.5	12	3.8	2.4
V-only - Dynamic	3.5	17.8	7.8	4.0
V-IMU - Circular	3.7	23.7	1.9	0.7
V-IMU - Short	2.5	12	1.9	0.5
V-IMU - Dynamic	3.5	17.8	1.7	1.2

[1] Mean translation error
[2] Standard deviation of translation error

It is interesting to analyze the gains brought by the IMU fusion. The most evident observation is that the output trajectory is smoother, which gives more consistency to the result (Fig. 10.2). But we can notice that the mean translation error (MTE) is also reduced (Table 10.1). Indeed, the motion model is more precise thanks to IMU data. This makes the outlier rejection more efficient than the visual-only CosySLAM which makes a zero velocity assumption between Keyframes.

10.3.2 Localization and Mapping of stairs by Solo

With our retrained model (Section 4.3.4) we were able to perform SLAM in our experimental area, without augmenting it with other objects. We recorded video sequences including stairs with a camera fixed on a Solo robot (Fig. 10.3). A stair has three discrete symmetries that are hard to handle for an object pose estimator and the images provided by Solo were noisy because of the walk. These scenarios are challenging for our SLAM system, but it maps successfully the stairs and the error on the position of the base of Solo remains reasonable (Table 10.2).

[1] T-LESS is one of the datasets for which CosyPose is trained by default and whose object can be bought in the Czech Republic [Hod+17]. It features several small electric devices, whose symmetry at lack of texture make them an interesting benchmark for realistic scenarios.

Table 10.2: Datasets description and results of the videos taken on Solo

Scenario	Length(m)	Duration(s)	MTE(cm)	STE(cm)
V-IMU - Approach	1.3	18.7	2.0	0.9
V-IMU - Module	1.3	15.5	2.4	1.5

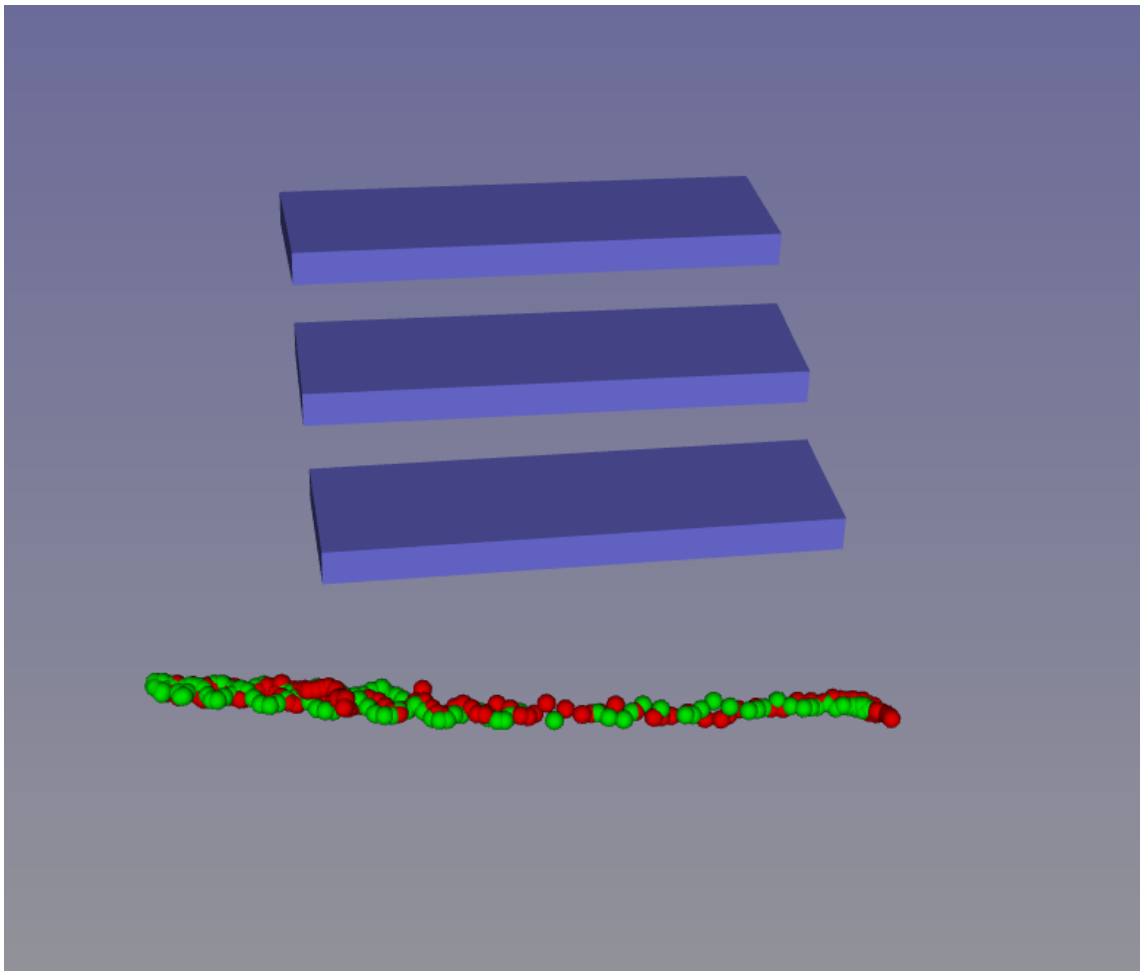


Figure 10.3: This trajectory was recorded on Solo walking along a climbing module made of three stairs using a walking controller [Léz+21]. The green dots represent the trajectory of Solo provided by the MoCap, and the red dots the one produced by our visual-inertial SLAM. The blue rectangles represent the map of the SLAM made of stairs.

10.4 Conclusion

This chapter presents the first step toward a semantic reconstruction of the robot environment. We see in this method the potential to provide multi-contact locomotion controllers [Car+17] with their needed contact surface reconstructions. By leveraging models of known objects in the scene, one may imagine localizing staircases, their handrail, door handles, etc. that the robot is required to interact with. This work was conducted during the MSc thesis of Cesar Debeunne, who was a passionate collaborator.

In the case where few or no object of interest are in the field of view, the IMU handles the estimation as a strap-down integration, which is viable for a few seconds. After this delay, extra sources of information are required to avoid drift in the estimates. One

solution would be to increase the robustness of the visual front-end by including residuals using classical 2D salient points features, as visual odometry, as already discussed in Chapter 8. In this context, objects in the scene would be useful both for providing semantic and surface information, as well as providing loop-closures.

The covariance model that we developed also requires more development. In particular, it required important experimental work to build the dataset of CosyPose errors. Collecting data for each individual object is very time-consuming. Besides, the models are likely to overfit the obtained dataset and generalize poorly to datapoints outside of its boundaries.

To improve this, we could work on several aspects. First, the ideal method would be to find a general model, as we did for the AprilTag PnP algorithm (Section 4.2.3), directly computable from the CosyPose network. Second, if we show that this is not feasible, we could try to train the covariance model in simulation which provides a vastly superior diversity in situations, though lacking some of the artifacts of reality. CosyPose itself is only trained in simulation. For this, we could use a reduced model as presented in this chapter or retrain Cosypose using methods specialized for neural networks [Jos+20]. Third, we could work on the formal definition of an uncertainty model, e.g. using a Bayesian formulation, that we also would have to fit empirical data. This would potentiality enable to quantify this quality of the predictions when extrapolating outside of the training domain (for example, if trained on a range 2-4 meters, the current model would diverge without warning if triggered at 5m, while a Bayesian model would predict a the variance and and uncertainty on this prediction).

Finally, we are eager to validate this formulation jointly with the tight estimation of Chapter 10 and are getting prepared for that as explained in next chapter.

Chapter 11

A new proprioceptive and vision dataset

This short chapter presents a dataset that we recently produced with Gepetto team's quadruped robot Solo-12. This dataset was acquired in our experimental space in LAAS-CNRS (see Fig. 11.1) and includes several walking trajectories using the controller [Léz+21]. We remote-controlled the robot in a scene augmented with AprilTags and elements from the T-less dataset, recording proprioceptive and exteroceptive measurements.

Figure 11.1: Solo quadruped in the experimental room.

The recorded sensor measurements include joint encoders, joint currents, the IMU from Solo onboard sensing as well as RGB camera and IMU from a RealSense D435i. Controller logs (such as the planned feet contacts timings) are also recorded. The RealSense was fixed at the front of the robot, slightly down-facing (30 degrees). A summary of available data is reported in Table 11.1. Calibration data using a fiducial marker grid were recored before the experiments to obtain the image distortion, camera-IMU relative transformation, and potential time-shift of the RealSense system. A sample image of one of the trajectories is shown in Fig. 11.2. External video recordings of each experiment were taken.

All data from Solo's onboard sensing is hardware synchronized. Similarly, the RealSense IMU and RGB images streams are synchronized. However, we do not have the

Table 11.1: Summary of available data sources from Solo-12 onboard sensing and RealSense D435i

Type	Source	Details	Frequency
Joint currents	Solo-12		1 kHz
IMU	Solo-12	3DM-CX5-25 LORD Microstrain	1 kHz
RGB images	Solo-12	1920 × 1080 rolling shutter	30 Hz
IMU	RealSense	Bosch BMI055	200 Hz
Ground truth	Qualysis Motion Capture		200 Hz

possibility to hardware synchronize the RealSense with Solo yet. Instead, we plan to rely on the correlation between Solo's IMU and the RealSense IMU to synchronize the streams of data in a post-processing step. To this end, we proceeded to a synchronization procedure at the beginning of each trajectory by tapping on the robot a few times. This precise signal should be enough to align both IMU time series and, thus, the rest of the sensor streams.

Recorded trajectories (2 minutes each) include motions of increasing difficulty, with forward-backward walking in a straight line, a square path around the scene, and loopy trajectories. With this dataset, we plan to implement an estimator fusing inertial, kinematics, and object-level transformation based on our previous work. This will serve to benchmark an integration work destined to obtain an estimator for both balance control states (orientation, velocity) and global localization through object-level SLAM. The loopy trajectories in particular will serve to display the behavior of the estimator when closing longer loops.

Figure 11.2: Example of an image captured with the RealSense attached to Solo while walking (with AprilTag detections).

Chapter 12

Conclusion

The objective of this thesis was to develop a new class of state estimators for legged robots able to fuse proprioceptive and exteroceptive sensors. We advocated for the use of tightly-coupled formulations where all measurements and state variables are integrated into a single estimation problem. This type of estimator enables a greater accuracy and the estimation of extra parameters by leveraging all sensors correlations. It also makes use of a sound mathematical formulation to extend generic estimators to more diverse and numerous perception sources, toward whole-body estimation. The accent was put on the necessary modularity of the formulations to make the mathematical developments extendable to new types of sensors. This was achieved through the use of Maximum a Posteriori estimation, modeled as Factor Graph optimization. All mathematical formulations used extensively the smooth manifold and Lie group theories that best represent the geometry of the state variables. We also emphasize the generalizability of the developed measurement models.

12.1 Contributions

We developed a range of measurement models targeted at sensors present on legged platforms. These measurement models were integrated as factors in three factor-graph-based estimators.

We implemented a general factor for camera-based object-level pose estimations based on object pose estimators. First, the AprilTag library was used to **obtain poses of fiducial markers** augmenting the scene with unique landmarks. We proposed a new analytical model to estimate the covariance of these measurements. We discussed the problem of the orientation ambiguity of these markers and proposed a practical method to alleviate this problem. The anisotropic nature of the pose uncertainty was highlighted in simulated experiments. Second, we used **a deep-learning-based framework to obtain the pose of known objects in the scene**. A covariance model based on empirical data was obtained through experiments. We also showed that we could fine-tune the model to elements of the scene of interest for legged-robots navigation such as stairs.

We then leverage the pre-integration formulation to include high-frequency sensors such as IMU in the factor graph. We propose **a more systematic formulation of this idea, generalizable to other high-rate proprioceptive sensors**. Within this framework, we proposed a new IMU pre-integration algorithm **based on a compact "delta" Lie-group**. This approach was theoretically compared to the seminal work on IMU pre-

integration, based on composite delta Lie groups. We then leveraged the generalized pre-integration method to **extend the pre-integration to force-torque sensors** present at the end-effectors of legged robots. We then proposed a first step toward whole-body estimation, by extending the factor graph approach to also **estimate centroidal quantities** (center of mass position and velocity, and angular momentum) by merging centroidal kinematics and force measurements. This makes it possible to accurately estimate the centroidal states, despite unavoidable biases in the kinematic model.

These measurement models were all integrated into the WOLF framework [Sol+21] to which we contributed, joining our effort with the IRI team, which was recently released open-source. WOLF provided the necessary modularity to formulate various estimation problems. It then allowed us to implement three different applications of the proposed theory, which we used to experimentally validate our models.

First, we developed **a visual-inertial SLAM system based on IMU pre-integration and the AprilTag pose measurement model**. This estimator was validated through a series of experiments conducted at LAAS on the HRP-2 humanoid robot. We showed that the system provided both: localization and mapping with centimetric precision for the locomotion of the robot on flat terrain and stairs, and a high-rate, smooth estimation of the base velocity. Both were evaluated against a motion capture ground truth.

Second, we proposed **a tightly-coupled algorithm for simultaneous estimation of the base and centroidal states of the robot**. This estimator combined the pre-integration of IMU and force-torque data, centroidal kinematics, and leg odometry. We showed that the estimator enables to estimate the bias on CoM kinematics measurements, which is sometimes ignored by controllers in first approximation. Experimental validation was conducted on simple trajectories performed on the Solo-12 quadruped robot.

Third, we implemented **an object-level visual-inertial SLAM system based on a deep-learning framework for object pose estimation**. This system used the same IMU pre-integration as well as our empirical model of the object pose uncertainty. We showed that the trajectory of the system and objects in the scene could be recovered and that the IMU contributed to the robustness of the system by alleviating the instability of the pose estimation. We proposed as a proof of concept to perform visual-inertial SLAM using stairs elements as landmarks, for which we retrained the neural network. This dataset was recorded on the Solo-12 quadruped robot.

Finally, we recorded a new dataset on Solo-12 quadruped for following works on the fusion of inertial, kinematics, and vision data.

12.2 Perspectives

This work laid the foundations of a general factor-graph-based estimation framework for legged robotics. However, a lot of alleys have yet to be explored to obtain an estimator usable on any legged platform. Here we present a few of the next projects that we wish to undertake.

12.2.1 Short term

We developed on one hand an object-level visual-inertial system and on the other hand a proprioceptive estimator for the sense of balance. A short-term goal would be to merge both estimators in one, providing simultaneously a high-rate estimation for control and

a non-drifting localization based on SLAM, which should serve also as an input for gait and/or contact planning.

We began to work in this direction by recording a dataset using Solo-12 as an experimental platform. For this experiment, we will concentrate on estimating the base state by including inertial, kinematics, and object pose measurements.

We also plan to integrate the system in a feedback loop with the current controller of Solo-12. For the moment, instabilities in the solver convergence times prevented us from obtaining a hard-real-time estimate for the proprioceptive estimator, while the AprilTag based visual-inertial SLAM system works in real-time. One of the solutions would be a hyperparameter search on the numerous options provided by the Ceres solver [AM+]. Another is to search for the most appropriate size of the graph, that is the frequency of Keyframe creation and their number. Marginalization of older states and sparsification procedures should also be investigated.

12.2.2 Mid term

In a second time, we would like to further strengthen the environmental perception of our system. This would be done by developing or integrating a vision system based on general geometric constraints. Many of the most mature vision-based SLAM systems are based on sparse feature extraction. This will be the first venue that we explore.

Our ideal realization would then be an integrated demo on Solo-12 with an odometry estimator based on kinematics, IMU, and a sparse feature KLT [BM04] tracking-based vision front-end. This system would then easily be extended with either of our object-level SLAM algorithms to provide loop closures and, therefore, a global localization, together with high bandwidth gravity related state estimation.

The reproducibility of experimental results is a common issue in many scientific fields, which is especially notable in the robotics community. Many benchmarks including IMU, vision, and LIDAR sensors are nowadays available [Bur+16; Cor+18; KMH19; ZCF21]. However, to the best of our knowledge, very few legged robots datasets are available [FS20; Ahm+21] and none include both proprioceptive and exteroceptive sensors. The Open Dynamic Robot Initiative [Gri+20], through the development of open-hardware robots (such as Solo-12) and open-source low-level controllers aims at making a robotic platform easily accessible to many institutions. One could imagine that a dataset to benchmark estimation algorithms on such a platform would be useful to the community as a whole.

12.2.3 Longer term

We will here develop a few ideas and reflections that our work on a general estimator for legged robots inspired.

Following the endeavor to develop an estimator fusing as many sources of information, we could turn our attention to a *whole-body* state estimation. As we have seen, the standard proprioception sensor set for legged robots is an IMU attached to the robot base and encoders at the joints (possibly along with joint torque, motor current, and force sensors at the end effector). This set of sensors is enough to implement proprioceptive odometry of the base but is quite limited when it comes to perceiving finer information about the state of the robot and its environment. In particular, the kinesthesis sense of artificial-legged systems is far from the subtleties of their biological counterparts, partly

because of the rigid segment assumptions. One solution is to augment the robot with other sensors measuring these flexibilities. Recent solutions have demonstrated the applicability of such methods by placing IMUs in each segment of an exoskeleton. One might also imagine adding strain gauges to measure directly the segments' deflections.

The sense of touch is for the moment also underrepresented in legged robotics. Some teams propose to add artificial skins that are implemented as strain cells sheets, which provides a higher density haptic feedback than strain-gauges. These systems in particular demonstrated a great potential for human-robot interactions. A higher-quality sense of touch may also provide richer information about the nature of the contacts between the robot and its environment, be it for locomotion (slip detection, terrain nature, etc.), or for manipulation (object surface analysis, 3D object position estimation).

On the other end of the spectrum, one could imagine methods targeted toward robots with limited sets of sensors that an observability study based on our formulation would help to optimize. For instance, Solo-12 is too small to be equipped with high-fidelity, multiple-axis strain gauges such as those found on humanoid robots. Such a robot may be equipped with cheap one-dimensional strain-gauges, which would require new measurement model formulations, with application in centroidal estimation.

With a limited set of sensors, the choice of their nature and placement on the robotic platform is crucial, which is limited by the nature of its initial design. On the control side, the concept of co-design is gaining traction. The goal is to optimize the design of a robot to certain criteria, such as energy efficiency or dynamic capabilities, exploring the design space guided by the simulated control of the platform. A dual concept could be applied to optimize the design of the platform to maximize the efficiency of estimation algorithms.

Our visual-inertial SLAM work based on CosyPose object position is a first step in deriving semantic information from the environment. On this line of research, we may imagine including more general information about the scene objects by leveraging latent space representations of these deep-learning algorithms. This might enable us to deal with the problem caused by the symmetries of these objects more properly. Work is also currently conducted on the side of the CosyPose team to better generalize the pose estimation to objects on which the model has not been trained. More broadly, the dialog between trained models and optimal estimation has a lot of potentials, both in robotics and computer vision communities. On one hand, learned factors could be trained with the end goal of improving the result of the estimation algorithm. On the other end, injecting priors based on physical models of the world, such as the dynamics of legged systems, can be used to improve the results of vision-based algorithms for dynamic scenes and human pose reconstruction.

Finally, the notion of state feedback may be rethought entirely. Model-based controllers rely on a limited set of physically grounded states to compute the robot control inputs. However, some recent learning-based systems do not require this kind of information, relying instead on latent space representations of the robot state learned in simulation. Some of these algorithms abstract the concepts of robot and environment as a single latent state vector resulting from a tight coupling of raw proprioceptive and exteroceptive measurements. While demonstrating impressive results in practice, those representations might have the downside of not being interpretable by a human operator and other systems, making the communication and fault detection harder to manage.

Appendix A

Covariance of the MAP estimate

An important question to ask is how confident are we in our MAP estimate. This can inform us on the quality of the performed estimation. In particular, in the case of parameter identification, a certain degree of excitation, namely variability in the data, is necessary to obtain reliable values. Unfortunately, as we will see, the computation of the covariance on the posterior is a costly process that is, therefore, rarely used in online estimation.

First we will review the process to obtain the posterior's covariance, known as the Laplacian approximation. Then we will give a simple example to illustrate the nature of the approximation and compare it to an exact inference of the posterior distribution.

A.1 Laplace approximation

Even if our priors and measurement models are Gaussian distributions, the posterior distribution is in general non-Gaussian (unless all measurement models are linear). The region near the peak of the posterior is, however, often nearly Gaussian in shape. The curvature around the mode is described by the Hessian of the posterior negative log-likelihood at the MAP estimated state. It can be shown that he Hessian is the *information matrix* of the problem[1]. Thus, finding the covariance of the MAP estimate resolves to inverting a sparse positive definite matrix.

$$p(\mathcal{X}|\mathcal{Z}) \sim \mathcal{N}(\mathcal{X}^{MAP}, \mathbf{H}^{-1}) \tag{A.1}$$

This is referred to as the quadratic or Laplace approximation ([McE18, Section 2.4.2]). Obtaining an approximation of the full posterior is then a two-step process: first, find the mode of the posterior (the MAP), then "fit a Gaussian" on this mode.

The full covariance is however rarely computed for a few reasons. First, few algorithms rely on this information (a notable exception being active feature matching [Dav+07], which is not longer used in state of the art vision systems). Second, computing the Hessian's inverse is too costly to be realized "in the loop". Sometimes, only parts of the covariance are computed using the Schur-complement method [Kon05], which is costly but much less than the full inverse. The covariance computation is then often left for offline evaluations of the estimation.

[1] The proof of this statement is out of the scope of this document and can be found in [Pen18, Section 5.1]

To contrast this approach with the aforementioned approximations, variational inference as applied in Barfoot et al. [BFY20] for instance fit both the mean and the information matrix of a Gaussian model as a result of a single optimization problem.

A.2 Example: stereoscopic depth estimation

We can illustrate the Laplace approximation with a one-dimensional toy problem (borrowed from Barfoot [Bar17, Section 4.1.1]). The problem is stated as estimating the depth $x \in \mathbb{R}$ of a landmark in the scene with a nonlinear camera model (see Fig. A.1)

$$y = \frac{fb}{x} + n_y \tag{A.2}$$

where $y = u - v$ is a disparity measurement (u and v are pixels corresponding to the

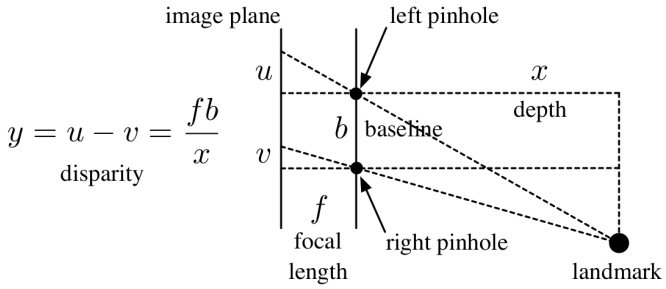


Figure A.1: Stereo depth estimation toy model [Bar17]

projection of the landmark in each camera), f is the focal length of the cameras (in pixels), b is the horizontal distance between cameras (the baseline, in meters), and $n_y \sim \mathcal{N}(0, \sigma_y^2)$ is the measurements noise (in pixels), assumed to be Gaussian. We also assume that we have prior knowledge about the estimated value x_p, with a standard deviation of σ_p.

To ground the problem, we will assign sensible values to the problem (same as [Bar17]):

$$x_{true} = 22\ m, \quad x_p = 20\ m, \quad \sigma_p = 3\ m$$
$$f = 400\ pixels, \quad b = 0.1\ m, \quad \sigma_y = 0.3\ pixels$$

where x_{true} is the true depth that we seek to estimate.

Notice that the prior standard deviation is quite big, assuming a great uncertainty about the prior value. We simulate noisy measurements by drawing samples $\{y_{i\in[1..m]}\}$ of the measurement model using x_{true} (we drew $m = 10$ measurements). For a low dimensional problem such as this one, it is possible to compute the full posterior distribution $p(x|\mathbf{y})$ by numerical integration, which is referred to as the "grid approximation" by [McE18]. We can make this computation almost arbitrarily precise since the computations are quite cheap. The prior and density distribution are represented in Fig. A.2. Applying the Laplacian approximation to compute a posterior approximation involves minimizing the negative log-likelihood:

$$\frac{1}{2\sigma_p^2}(x - x_p)^2 + \frac{1}{2\sigma_y^2}\sum_{i=1}^{m}(\frac{fb}{x} - y_i) \tag{A.3}$$

Finding the MAP and approximating the covariance deviation of x, we can plot the MAP posterior along with its numerical computation in Fig. A.2. Both computations result in largely overlapping functions: the main mass of the real posterior density function

is captured by the Laplacian approximation. However, notice that the real posterior distribution is not symmetrical contrary to the prior it derives from. This means that, contrary to its Gaussian approximation, the mean of the of the real posterior is not equal to its mode.

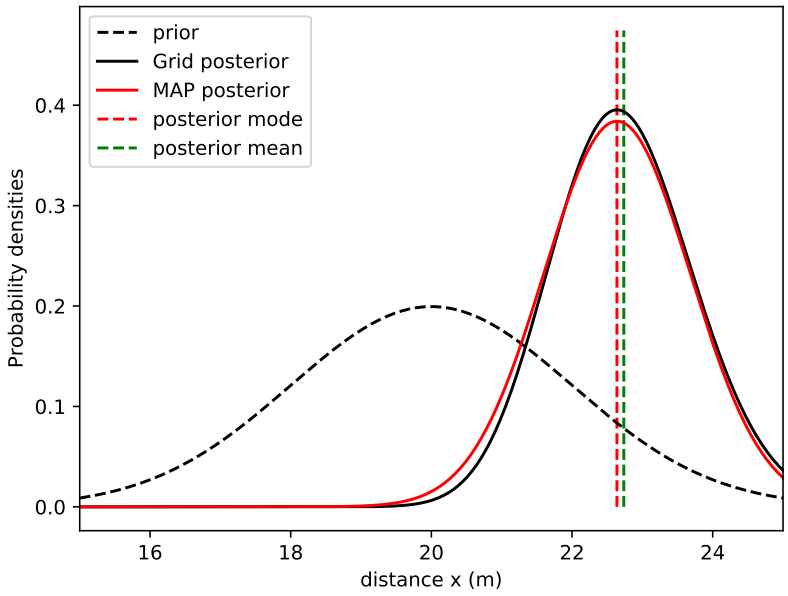


Figure A.2: Representation of the posterior distribution inference for the 1D depth estimation problem. Dotted black: prior on x, continuous black: numerical "grid" integration of the posterior, continuous red: Laplacian approximation of the posterior. Vertical lines: red=MAP, green=mean of the posterior. The true value (22 m) is reasonably close to the MAP given the variance of the posterior distribution.

Appendix B

Pre-integration on Lie groups

B.1 Justification of Forster's delta formulas

We detail here how the delta formulas of equations (6.9) and (6.10) can be derived from (6.5).

The rotation one is easy to get, multiplying by $\mathbf{R}^{i,T}$:

$$\boldsymbol{\Delta}\mathbf{R}_{im} \triangleq \mathbf{R}^{i,T}\,\mathbf{R}^{m} = \prod_{k=i}^{m} \mathrm{Exp}((\widetilde{\boldsymbol{\omega}}^{k} - \mathbf{b}_{\boldsymbol{\omega}}^{k} - \mathbf{n}_{\boldsymbol{\omega}}^{k})\delta t) \tag{B.1}$$

The velocity is as well quite easy, by defining $\Delta t_{im} \triangleq \sum_{k=i}^{m} \delta t = (m-i)\delta t$:

$$\boldsymbol{\Delta}\mathbf{v}_{im} \triangleq \mathbf{R}^{i,T}(\mathbf{v}_m - \mathbf{v}_i - g\Delta t_{im}) = \prod_{k=i}^{m} \boldsymbol{\Delta}\mathbf{R}_{ik}\,\mathrm{Exp}((\tilde{\mathbf{a}}^{k} - \mathbf{b}_{\mathbf{a}}^{k} - \mathbf{n}_{\mathbf{a}}^{k})\delta t) \tag{B.2}$$

The position delta requires more calculations. First, inject equation (B.2) in the position equation of (6.5) then rearrange and reorder the terms.

$$\begin{aligned}
\mathbf{p}^m - \mathbf{p}^i &= \sum_{k=i}^{m}\left[(\mathbf{v}^i + \mathbf{g}\Delta t_{ik} + \mathbf{R}^i\boldsymbol{\Delta}\mathbf{v}_{ik})\delta t + \frac{1}{2}\mathbf{g}\delta t^2 + \frac{1}{2}\,\mathbf{R}^k(\tilde{\mathbf{a}}^k - \mathbf{b}_{\mathbf{a}}^k - \mathbf{n}_{\mathbf{a}}^k)\delta t^2\right] \\
&= \mathbf{v}^i\Delta t_{im} + \mathbf{g}\sum_{k=i}^{m}\Delta t_{ik}\delta t + \frac{1}{2}\mathbf{g}\sum_{k=i}^{m}\delta t^2 + \mathbf{R}^i\sum_{k=i}^{m}\left[\boldsymbol{\Delta}\mathbf{v}_{ik}\delta t + \frac{1}{2}\boldsymbol{\Delta}\mathbf{R}_{ik}(\tilde{\mathbf{a}}^k - \mathbf{b}_{\mathbf{a}}^k - \mathbf{n}_{\mathbf{a}}^k)\delta t^2\right] \\
&= \mathbf{v}^i\Delta t_{im} + \mathbf{g}\sum_{k=i}^{m}(\Delta t_{ik}\delta t + \frac{1}{2}\delta t^2) + \mathbf{R}^i\sum_{k=i}^{m}\left[\boldsymbol{\Delta}\mathbf{v}_{ik}\delta t + \frac{1}{2}\boldsymbol{\Delta}\mathbf{R}_{ik}(\tilde{\mathbf{a}}^k - \mathbf{b}_{\mathbf{a}}^k - \mathbf{n}_{\mathbf{a}}^k)\delta t^2\right]
\end{aligned} \tag{B.3}$$

We will simplify the gravity term that we will call $\mathcal{G}t_{im}$. Noting that:

$$\sum_{k=i}^{m} k = \sum_{k=1}^{m} k - \sum_{k=1}^{i-1} k = \frac{m(m-1)}{2} - \frac{i(i-1)}{2}$$

we can deduce that:

$$
\begin{aligned}
\mathcal{G}t_{im} &= \delta t \sum_{k=i}^{m} (\Delta t_{ik} + \frac{1}{2}\delta t) = \delta t \sum_{k=i}^{m} ((k-i)\delta t + \frac{1}{2}\delta t) \\
&= \delta t \Big[\sum_{k=i}^{m} [k\delta t - i(m-i)\delta t] + \frac{1}{2}\Delta t_{im} \Big] \\
&= \delta t \Big[\frac{\delta t}{2}(m(m-1) - i(i-1) - 2i(m-1)) + \frac{1}{2}\Delta t_{im} \Big] \\
&= \delta t \Big[\frac{\delta t}{2}(i^2 - 2im + m^2 + i - m) + \frac{1}{2}\Delta t_{im} \Big] \\
&= \delta t \Big[\frac{1}{2}(m-i)^2 dt - \frac{1}{2}(m-i)\delta t + \frac{1}{2}\Delta t_{im} \Big] \\
&= \frac{1}{2}(m-i)^2 dt^2 = \frac{1}{2}\Delta t_{im}^2
\end{aligned}
$$

Multiplying by $\mathbf{R}^{i,T}$ and reordering the terms, we can finally define a position delta quantity:

$$
\boldsymbol{\Delta}\mathbf{p}_{im} \triangleq \mathbf{R}^{i,T}(\mathbf{p}^m - \mathbf{p}^i - \mathbf{v}^i \Delta t_{im} - \frac{1}{2}\mathbf{g}\Delta t_{im}^2) = \sum_{k=i}^{m} \Big[\boldsymbol{\Delta}\mathbf{v}_{ik}\delta t + \frac{1}{2}\boldsymbol{\Delta}\mathbf{R}_{ik}(\tilde{\mathbf{a}}^k - \mathbf{b}_{\mathbf{a}}^k - \mathbf{n}_{\mathbf{a}}^k)\delta t^2 \Big] \tag{B.4}
$$

B.2 Elements of the compact IMU delta matrix Lie group

B.2.1 Tangent space and Lie algebra $\mathfrak{d}$

[SDA18] Following [SDA18], the tangent space of $\mathcal{D}$ at the point $\boldsymbol{\Delta}$ is found by taking the time derivative of the group constraint, $\boldsymbol{\Delta}^{-1}\boldsymbol{\Delta} = \mathbf{I}$. Noting $\dot{\bullet} \triangleq \frac{\partial \bullet}{\partial t}$, this yields after a few manipulations

$$
\boldsymbol{\Delta}^{-1}\dot{\boldsymbol{\Delta}} = \begin{bmatrix} [\boldsymbol{\omega}]_\times & \boldsymbol{\Delta}\mathbf{R}^\top \mathbf{a} & \boldsymbol{\Delta}\mathbf{R}^\top(\mathbf{v} - \boldsymbol{\Delta}\mathbf{v}) \\ \mathbf{0} & 0 & 1 \\ \mathbf{0} & 0 & 0 \end{bmatrix}, \tag{B.5}
$$

with $\mathbf{v} \triangleq \dot{\boldsymbol{\Delta}}\mathbf{p}$, $\mathbf{a} \triangleq \dot{\boldsymbol{\Delta}}\mathbf{v}$ and $[\boldsymbol{\omega}]_\times \triangleq \boldsymbol{\Delta}\mathbf{R}^\top\dot{\boldsymbol{\Delta}}\mathbf{R}$. The Lie algebra $\mathfrak{d}$ is the tangent space at the identity $\boldsymbol{\Delta} = \mathbf{I}$. Its elements $\boldsymbol{\nu}^\wedge \triangleq \dot{\boldsymbol{\Delta}}|_{\boldsymbol{\Delta}=\mathbf{I}}$ and their isomorphics $\boldsymbol{\nu}$ in Cartesian space are given by,

$$
\boldsymbol{\nu}^\wedge = \begin{bmatrix} [\boldsymbol{\omega}]_\times & \mathbf{a} & \mathbf{v} \\ \mathbf{0} & 0 & 1 \\ \mathbf{0} & 0 & 0 \end{bmatrix} \in \mathfrak{d} \quad \underset{\wedge}{\overset{\vee}{\rightleftharpoons}} \quad \boldsymbol{\nu} = \begin{bmatrix} \mathbf{v} \\ \mathbf{a} \\ \boldsymbol{\omega} \\ 1 \end{bmatrix} \in \mathbb{R}^{10}. \tag{B.6}
$$

This tangent $\boldsymbol{\nu}^\wedge$ corresponds to the 'velocity' of the group element. Any point in the Lie algebra can be obtained after moving at constant velocity during a period Δt, that is, $\boldsymbol{\tau}^\wedge = \boldsymbol{\nu}^\wedge \Delta t \in \mathfrak{d}$ —see (6.35).

B.2.2 The exponential map

Eq. (B.5) can be written as $\dot{\boldsymbol{\Delta}} = \boldsymbol{\Delta} \cdot \boldsymbol{\nu}^\wedge$. This is an ordinary differential equation whose integral for constant $\boldsymbol{\nu}$ yields the exponential map [SDA18], $\boldsymbol{\Delta}(t) = \exp(\boldsymbol{\nu}^\wedge t)$. This gives a direct expression of the integral of information of the type $(\mathbf{v}, \mathbf{a}, \boldsymbol{\omega})$ onto the deltas

manifold. The closed form of the exponential map is obtained through Taylor expansion (see *e.g.* [SDA18] for examples). At $t = \Delta t$ we have,

$$\boldsymbol{\Delta}(\Delta t) = \exp(\boldsymbol{\nu}^{\wedge}\Delta t) \triangleq \sum_{n} \frac{1}{n!}(\boldsymbol{\nu}^{\wedge}\Delta t)^{n} \,. \tag{B.7}$$

Exploiting the cyclic pattern of the powers of $[\boldsymbol{\omega}]_{\times}$, this results in

$$\exp\left(\begin{bmatrix} [\boldsymbol{\omega}]_{\times} & \mathbf{a} & \mathbf{v} \\ \mathbf{0} & 0 & 1 \\ \mathbf{0} & 0 & 0 \end{bmatrix}\Delta t\right) = \begin{bmatrix} \exp([\boldsymbol{\omega}]_{\times}\Delta t) & \mathbf{Q}\mathbf{a}\Delta t & \mathbf{Q}\mathbf{v}\Delta t + \mathbf{P}\mathbf{a}\Delta t^{2} \\ \mathbf{0} & 1 & \Delta t \\ \mathbf{0} & 0 & 1 \end{bmatrix} \tag{B.8}$$

with (we skip proofs for space reasons)

$$\mathbf{Q}(\boldsymbol{\theta}) = \mathbf{I} + \frac{1-\cos\theta}{\theta}[\mathbf{u}]_{\times} + \frac{\theta - \sin\theta}{\theta}[\mathbf{u}]_{\times}^{2} \tag{B.9}$$

$$\mathbf{P}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{I} + \frac{\theta - \sin\theta}{\theta^{2}}[\mathbf{u}]_{\times} + \frac{\cos\theta + \frac{1}{2}\theta^{2} - 1}{\theta^{2}}[\mathbf{u}]_{\times}^{2} \,, \tag{B.10}$$

where $\boldsymbol{\theta} = \boldsymbol{\omega}\Delta t$, $\theta = \|\boldsymbol{\theta}\|$ and $\mathbf{u} = \boldsymbol{\theta}/\theta$ form the angle-axis representation of the rotation step $\boldsymbol{\omega}\Delta t$.

B.2.3 The adjoint and small adjoint matrices

Following the general methodology explained in [SDA18], the adjoint matrix is obtained by identifying the linear terms in $\mathbf{Ad}_{\boldsymbol{\Delta}}\boldsymbol{\tau} = (\boldsymbol{\Delta\tau}^{\wedge}\boldsymbol{\Delta}^{-1})^{\vee}$. We get after long but relatively easy calculations,

$$\mathbf{Ad}_{\boldsymbol{\Delta}} = \begin{bmatrix} \boldsymbol{\Delta}\mathbf{R} & -\boldsymbol{\Delta}\mathbf{R}\Delta t & [\boldsymbol{\Delta}\mathbf{p} - \boldsymbol{\Delta}\mathbf{v}\Delta t]_{\times}\boldsymbol{\Delta}\mathbf{R} & \boldsymbol{\Delta}\mathbf{v} \\ \mathbf{0} & \boldsymbol{\Delta}\mathbf{R} & [\boldsymbol{\Delta}\mathbf{v}]_{\times}\boldsymbol{\Delta}\mathbf{R} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{\Delta}\mathbf{R} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & 1 \end{bmatrix} \in \mathbb{R}^{10\times 10} \,. \tag{B.11}$$

Similarly, from [Ead18] the small adjoint matrix can be computed by identifying the linear terms in $\mathbf{ad}_{\boldsymbol{\tau}}\boldsymbol{\sigma} = (\boldsymbol{\tau}^{\wedge}\boldsymbol{\sigma}^{\wedge} - \boldsymbol{\sigma}^{\wedge}\boldsymbol{\tau}^{\wedge})^{\vee}$ which for $\boldsymbol{\tau} = (\boldsymbol{\rho}, \boldsymbol{\upsilon}, \boldsymbol{\theta}, \Delta t) \in \mathfrak{d}$ yields,

$$\mathbf{ad}_{\boldsymbol{\tau}} = \begin{bmatrix} [\boldsymbol{\theta}]_{\times} & -\mathbf{I}\Delta t & [\boldsymbol{\rho}]_{\times} & \boldsymbol{\upsilon} \\ \mathbf{0} & [\boldsymbol{\theta}]_{\times} & [\boldsymbol{\upsilon}]_{\times} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & [\boldsymbol{\theta}]_{\times} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & 0 \end{bmatrix} \in \mathbb{R}^{10\times 10} \,. \tag{B.12}$$

B.2.4 The right Jacobian

The right Jacobian $\mathbf{J}_r$ is the Jacobian of $\mathrm{Exp}()$ as described in [SDA18]. Lacking at the moment a closed form for it, we take the general methodology for the left Jacobian described in [Ead18], and transform it to the right using $\mathbf{J}_r(\boldsymbol{\tau}) = \mathbf{J}_l(-\boldsymbol{\tau})$ [SDA18],

$$\mathbf{J}_r(\boldsymbol{\tau}) = \mathbf{J}_l(-\boldsymbol{\tau}) = \sum_{i} \frac{\mathbf{ad}_{-\boldsymbol{\tau}}{}^{i}}{(i+1)!} = \sum_{i} \frac{(-\mathbf{ad}_{\boldsymbol{\tau}})^{i}}{(i+1)!} \,. \tag{B.13}$$

This sum can be truncated at the desired degree of accuracy.

Abstract

Legged robots are complex mechanisms whose stable behavior depends on the proper estimation of several different quantities that must be observed at high speed and accuracy. While the robot state can mostly not be observed directly by any existing sensors, it is typically reconstructed by fusing very diverse sensor modalities. These sources of information differ significantly on their acquisition frequencies, the nature of the data, and the computational processing cost. While state estimation by sensor fusion is a common topic to most robotic platforms, it offers a particular challenge in legged robotics by their peculiar dynamics. For legged robots, on the one hand, the robot state is needed to maintain its sense of balance and locomote safely, and, on the other hand, a precise representation of the environment is required for navigation and interaction.

The current approach for many legged systems, in the literature, solve these problems independently, using cascades of estimators that may neglect some of the correlation present in the data. This artificial decoupling acts as strong priors that enable simple estimators to handle the estimation of each part of the cascade and stabilize the behavior of the overall estimation scheme. On the other hand, designing a cascade involves a lot of specialized work that hardly generalizes to new scenarios or new sensor modalities. In this thesis, we rather defend the idea of building a single tightly-coupled estimator capable of estimating all quantities needed by the robot. For this goal, the framework of a-posteriori estimation, formalized as a factor graph, is very suitable. This assertion does not come as a surprise, as factor graphs are nowadays highly popular in the SLAM literature, yet they are still under-represented in legged robots systems.

In this thesis, we investigate a few avenues that we believe are crucial to achieving these goals. First, we develop tailored sensor measurement models with attention to the correct mathematical formulation involving Lie theory. Second, we propose visual-inertial systems based on object-level detection that provide relative transformation between the camera and the objects. We provide covariance models for two kinds of objects: the first one is an analytical model for fiducial markers ; the second is an empirical model for deep-learning-based object pose estimation. Third, we handle high-rate sensors by developing a generalization of the IMU pre-integration theory. We propose a new formulation of the IMU pre-integration based on compact Lie groups. Fourth, we show that pre-integration can also be applied to use force-torque sensors found on legged robots. By fusing it with a leg-kinematics based odometry and IMU, we show that this new formulation makes possible the tightly-coupled estimation of centroidal quantities within the context of Factor Graph estimation.

The proposed theoretical ideas are implemented in a coherent estimation framework, extending the factor-graph software Wolf. Each new modality is validated in a dedicated experimental setup that allowed us to quantify its interest and relevance for legged robotics.

Keywords

Factor Graph, estimation, legged robots, SLAM, pre-integration, IMU, force sensor, vision

Résumé

Les robots à pattes sont des mécanismes complexes dont la stabilité dépend de la bonne estimation de plusieurs quantités qui doivent être observées à grande vitesse et avec précision. Bien que l'état du robot ne puisse généralement pas être observé directement par des capteurs, il est généralement possible de le reconstruire en fusionnant plusieurs capteurs à condition d'être capable de bénéficier des modalités très diverses qu'ils offrent. C'est bien sur un sujet commun à la plupart des plateformes robotiques, mais le défi posé par l'estimation d'état par fusion de capteurs se renouvelle dans le cadre de la robotique à patte à cause de la dynamique particulière de ces systèmes. D'une part, leur équilibre dépend intimement de l'estimation correcte de leur état ; d'autre part, ils ont besoin d'une représentation fine de leur environnement pour y naviguer et y interagir.

A ce jour, l'approche actuelle dominante les robots à pattes est de résoudre indépendamment plusieurs problèmes d'estimation, en utilisant des cascades d'estimateurs qui peuvent négliger une partie des corrélations présente dans les données. Ce découplage artificiel agit comme des *a priori* forts qui permettent à des estimateurs simples de gérer chaque partie de la cascade et de stabiliser le comportement du schéma d'estimation global. Ce pragmatisme implique néanmoins beaucoup de travail spécialisé qui ne se généralise guère à de nouveaux scénarios ou à de nouvelles modalités de capteurs. Dans cette thèse, nous défendons l'idée de construire un unique estimateur capable d'estimer toutes les quantités nécessaires au robot de manière étroitement couplée. Le cadre de l'estimation a-postériori, formalisé sous la forme d'un graphe de facteurs, est alors très approprié pour formaliser l'approche. Cette affirmation n'est pas une surprise, car les graphes de facteurs sont aujourd'hui très populaires dans la littérature SLAM. Ils sont cependant encore sous-représentés pour les robots à pattes.

Dans cette thèse, nous étudions quelques pistes qui nous semblent cruciales pour atteindre cet objectif. Tout d'abord, nous développons des modèles de mesure de capteurs en prêtant attention à la formulation mathématique correcte fondée sur la théorie des groupes de Lie. Deuxièmement, nous proposons des systèmes visuels inertiels basés sur des algorithmes de détection d'objets, qui fournissent une transformation relative entre la caméra et les objets. Nous fournissons des modèles de covariance pour deux types d'objets : le premier est un modèle analytique pour les marqueurs fiduciaires ; la deuxième est un modèle empirique pour l'estimation de pose d'objet basée sur l'apprentissage en profondeur. Troisièmement, nous traitons les capteurs à haute fréquence en développant une généralisation de la théorie de pré-intégration IMU. Nous proposons une nouvelle formulation de la pré-intégration IMU basée sur des groupes de Lie compacts. Quatrièmement, nous montrons que la pré-intégration peut également être appliquée pour utiliser les capteurs de forces trouvés sur les robots à pattes. En la fusionnant avec une odométrie basée sur la cinématique des jambes et une IMU, nous montrons que cette nouvelle formulation rend possible l'estimation étroitement couplée des quantités centroïdale dans le contexte de l'estimation des graphes de facteurs.

Les idées théoriques proposées sont mises en œuvre dans un cadre d'estimation cohérent, étendant le logiciel de graphe de facteurs Wolf. Chaque nouvelle modalité est validée dans un montage expérimental dédié qui nous a permis de quantifier son intérêt et sa pertinence pour la robotique à pattes.

Mots clefs

Estimation, graphes de facteurs, robots à pattes, SLAM, pre-integration, IMU, capteur de force, vision

Bibliography

[AM+] Sameer Agarwal, Keir Mierle, et al. *Ceres Solver*. http://ceres-solver.org (cit. on pp. 22, 30, 31, 94, 119).

[Ahm+21] Ahmadreza Ahmadi, Tønnes Nygaard, Navinda Kottege, David Howard, and Nicolas Hudson. "Semi-supervised gated recurrent neural networks for robotic terrain classification". In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 1848–1855 (cit. on p. 119).

[Ahn+12] SungHwan Ahn, Sukjune Yoon, Seungyong Hyung, Nosan Kwak, and Kyung Shik Roh. "On-board odometry estimation for 3D vision-based SLAM of humanoid robot". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2012, pp. 4006–4012 (cit. on p. 20).

[APL08] Yannick Aoustin, Franck Plestan, and Vincent Lebastard. "Experimental comparison of several posture estimation solutions for biped robot rabbit". In: *2008 IEEE International Conference on Robotics and Automation*. IEEE. 2008, pp. 1270–1275 (cit. on p. 13).

[Atc18] Dinesh Atchuthan. "Towards new sensing capabilities for legged locomotion using real-time state estimation with low-cost IMUs". Theses. Université Paul Sabatier - Toulouse III, Oct. 2018. URL: https://tel.archives-ouvertes.fr/tel-02088756 (cit. on pp. 65, 68, 72, 78).

[Aul+08] Josep Aulinas, Yvan Petillot, Joaquim Salvi, and Xavier Llado. "The SLAM problem: a survey". In: *Artificial Intelligence Research and Development* (2008), pp. 363–371 (cit. on p. 19).

[Aw+96] ST Aw, T Haslwanter, GM Halmagyi, IS Curthoys, RA Yavor, and MJ Todd. "Three-dimensional vector analysis of the human vestibuloocular reflex in response to high-acceleration head rotations. I. Responses in normal subjects". In: *Journal of neurophysiology* 76.6 (1996), pp. 4009–4020 (cit. on p. 3).

[AVN08] Ko Ayusawa, Gentiane Venture, and Yoshihiko Nakamura. "Identification of the inertial parameters of a humanoid robot using unactuated dynamics of the base link". In: *Humanoids 2008-8th IEEE-RAS International Conference on Humanoid Robots*. IEEE. 2008, pp. 1–7 (cit. on p. 17).

[AVN14] Ko Ayusawa, Gentiane Venture, and Yoshihiko Nakamura. "Identifiability and identification of inertial parameters using the underactuated base-link dynamics for legged multibody systems". In: *The International Journal of Robotics Research* 33.3 (2014), pp. 446–468 (cit. on p. 17).

[BCS21] François Bailly, Justin Carpentier, and Philippe Souères. "Optimal Estimation of the Centroidal Dynamics of Legged Robots". In: *ICRA 2021-IEEE International Conference on Robotics and Automation*. 2021 (cit. on p. 19).

[Bai+19] François Bailly, Justin Carpentier, Bruno Watier, and Philippe Soueres. "Recursive estimation of the human body's center of mass and angular momentum derivative". In: *CMBBE 16th Internat. Symposium on Computer Methods in Biomechanics and Biomedical Engineering*. 2019 (cit. on p. 19).

[Bai+18] François Bailly, Emmanuelle Pouydebat, Bruno Watier, Vincent Bels, and Philippe Souères. "Should mobile robots have a head?" In: *Conference on Biomimetic and Biohybrid Systems*. Springer. 2018, pp. 28–39 (cit. on p. 3).

[BM04] Simon Baker and Iain Matthews. "Lucas-kanade 20 years on: A unifying framework". In: *International journal of computer vision* 56.3 (2004), pp. 221–255 (cit. on pp. 23, 92, 119).

[Bar17] Timothy D Barfoot. *State estimation for robotics*. Cambridge University Press, 2017 (cit. on p. 122).

[BFY20] Timothy D Barfoot, James R Forbes, and David J Yoon. "Exactly sparse Gaussian variational inference with application to derivative-free batch nonlinear state estimation". In: *The International Journal of Robotics Research* 39.13 (2020), pp. 1473–1502 (cit. on pp. 28, 122).

[BB18] Axel Barrau and Silvere Bonnabel. "Invariant kalman filtering". In: *Annual Review of Control, Robotics, and Autonomous Systems* 1 (2018), pp. 237–257 (cit. on pp. 13, 77).

[BB20] Axel Barrau and Silvere Bonnabel. "A mathematical framework for IMU error propagation with applications to preintegration". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 5732–5738 (cit. on pp. 77, 78).

[BD67] JF Bellantoni and KW Dodge. "A square root formulation of the Kalman-Schmidt filter." In: *AIAA journal* 5.7 (1967), pp. 1309–1314 (cit. on p. 7).

[Bel+18] C Dario Bellicoso et al. "Advances in real-world applications for legged robots". In: *Journal of Field Robotics* 35.8 (2018), pp. 1311–1326 (cit. on p. 107).

[BL15] Mehdi Benallegue and Florent Lamiraux. "Estimation and stabilization of humanoid flexibility deformation using only inertial measurement units and contact information". In: *International Journal of Humanoid Robotics* 12.03 (2015), p. 1550025 (cit. on pp. 16, 18).

[Bla19] José-Luis Blanco-Claraco. "A Modular Optimization Framework for Localization and Mapping." In: *Robotics: Science and Systems*. 2019 (cit. on p. 24).

[Ble+18a] Gerardo Bledt, Matthew J Powell, Benjamin Katz, Jared Di Carlo, Patrick M Wensing, and Sangbae Kim. "Mit cheetah 3: Design and control of a robust, dynamic quadruped robot". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 2245–2252 (cit. on pp. 11, 13, 14, 21, 103).

[Ble+18b] Gerardo Bledt, Patrick M Wensing, Sam Ingersoll, and Sangbae Kim. "Contact model fusion for event-based locomotion in unstructured terrains". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 1–8 (cit. on pp. 14, 15).

[Blo17] Michael Bloesch. "State estimation for legged robots-kinematics, inertial sensing, and computer vision". PhD thesis. ETH Zurich, 2017 (cit. on p. 13).

[Blo+17] Michael Bloesch, Michael Burri, Hannes Sommer, Roland Siegwart, and Marco Hutter. "The two-state implicit filter recursive estimation for mobile robots". In: *IEEE Robotics and Automation Letters* 3.1 (2017), pp. 573–580 (cit. on p. 92).

[Blo+13a] Michael Bloesch, Christian Gehring, Péter Fankhauser, Marco Hutter, Mark A Hoepflinger, and Roland Siegwart. "State estimation for legged robots on unstable and slippery terrain". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2013, pp. 6058–6064 (cit. on pp. 11, 14).

[BH18] Michael Bloesch and Marco Hutter. "Technical Implementations of the Sense of Balance". In: *Journal: Humanoid Robotics: A Reference* (2018), pp. 1489–1517 (cit. on pp. 10, 11).

[Blo+13b] Michael Bloesch et al. "State estimation for legged robots-consistent fusion of leg kinematics and IMU". In: *Robotics* 17 (2013), pp. 17–24 (cit. on pp. 10, 11, 13, 17, 24, 61, 62, 100, 102).

[Bon+19] V Bonnet et al. "Overview on dynamic identification methods of floating base anthropomorphic structures". In: *Computer Methods in Biomechanics and Biomedical Engineering* 22.sup1 (2019), S471–S473 (cit. on p. 17).

[Bon+18] Vincent Bonnet, André Crosnier, Gentiane Venture, Maxime Gautier, and Philippe Fraisse. "Inertial Parameters Identification of a Humanoid Robot Hanged to a Fix Force Sensor". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 4927–4932 (cit. on p. 17).

[BBV04] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004 (cit. on pp. 30, 32, 40).

[Bro+21] Martin Brossard, Axel Barrau, Paul Chauchat, and Silvère Bonnabel. "Associating uncertainty to extended poses for on lie group IMU preintegration with rotating Earth". In: *IEEE Transactions on Robotics* (2021) (cit. on pp. 77, 78).

[Bur+16] Michael Burri et al. "The EuRoC micro aerial vehicle datasets". In: *The International Journal of Robotics Research* (2016). DOI: 10.1177/0278364915620033. eprint: http://ijr.sagepub.com/content/early/2016/01/21/0278364915620033.full.pdf+html. URL: http://ijr.sagepub.com/content/early/2016/01/21/0278364915620033.abstract (cit. on p. 119).

[Cad+16] Cesar Cadena et al. “Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age”. In: *IEEE Transactions on robotics* 32.6 (2016), pp. 1309–1332 (cit. on p. 19).

[Cam17] M Camurri. “Multisensory state estimation and mapping on dynamic legged robots”. In: *Istituto Italiano di Tecnologia and Univ. Genoa* (2017) (cit. on p. 13).

[Cam+20] Marco Camurri, Milad Ramezani, Simona Nobili, and Maurice Fallon. “Pronto: A multi-sensor state estimator for legged robots in real-world scenarios”. In: *Frontiers in Robotics and AI* 7 (2020), p. 68 (cit. on p. 21).

[Cam+17] Marco Camurri et al. “Probabilistic contact estimation and impact detection for state estimation of quadruped robots”. In: *IEEE Robotics and Automation Letters* 2.2 (2017), pp. 1023–1030 (cit. on p. 15).

[Car+14] Luca Carlone, Zsolt Kira, Chris Beall, Vadim Indelman, and Frank Dellaert. “Eliminating conditionally independent sets in factor graphs: A unifying perspective based on smart factors”. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, pp. 4290–4297 (cit. on p. 78).

[Car+16a] Justin Carpentier, Mehdi Benallegue, Nicolas Mansard, and Jean-Paul Laumond. “Center-of-mass estimation for a polyarticulated system in contact—A spectral approach”. In: *IEEE Transactions on Robotics* 32.4 (2016), pp. 810–822 (cit. on pp. 18, 19, 82).

[Car+16b] Justin Carpentier, Steve Tonneau, Maximilien Naveau, Olivier Stasse, and Nicolas Mansard. “A versatile and efficient pattern generator for generalized legged locomotion”. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 3555–3561 (cit. on p. 17).

[Car+17] Justin Carpentier et al. “Multi-contact locomotion of legged robots in complex environments–the loco3d project”. In: *RSS Workshop on Challenges in Dynamic Legged Locomotion*. 2017, 3p (cit. on pp. 19, 112).

[Car+19] Justin Carpentier et al. “The Pinocchio C++ library: A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives”. In: *2019 IEEE/SICE International Symposium on System Integration (SII)*. IEEE. 2019, pp. 614–619 (cit. on pp. 10, 60, 101).

[CL85] Raja Chatila and Jean-Paul Laumond. “Position referencing and consistent world modeling for mobile robots”. In: *Proceedings. 1985 IEEE International Conference on Robotics and Automation*. Vol. 2. IEEE. 1985, pp. 138–145 (cit. on p. 1).

[CHG11] Annett Chilian, Heiko Hirschmüller, and Martin Görner. “Multisensor data fusion for robust pose estimation of a six-legged walking robot”. In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2011, pp. 2497–2504 (cit. on p. 13).

[CED08] Jose A Cobano, Joaquin Estremera, and P Gonzalez De Santos. “Location of legged robots in outdoor environments”. In: *Robotics and Autonomous Systems* 56.9 (2008), pp. 751–761 (cit. on pp. 11, 13).

[CB14] Toby Collins and Adrien Bartoli. "Infinitesimal plane-based pose estimation". In: *International journal of computer vision* 109.3 (2014), pp. 252–286 (cit. on p. 46).

[Col+20] Mirco Colosi et al. "Plug-and-Play SLAM: A Unified SLAM Architecture for Modularity and Ease of Use". In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2020, pp. 5051–5057 (cit. on p. 24).

[Cor+18] Santiago Cortés, Arno Solin, Esa Rahtu, and Juho Kannala. "ADVIO: An authentic dataset for visual-inertial odometry". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 419–434 (cit. on p. 119).

[CB21] Erwin Coumans and Yunfei Bai. *PyBullet, a Python module for physics simulation for games, robotics and machine learning*. http://pybullet.org. 2016–2021 (cit. on p. 56).

[Dav+07] Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. "MonoSLAM: Real-time single camera SLAM". In: *IEEE transactions on pattern analysis and machine intelligence* 29.6 (2007), pp. 1052–1067 (cit. on pp. 20, 22, 23, 121).

[De 96] Paolo De Leva. "Adjustments to Zatsiorsky-Seluyanov's segment inertia parameters". In: *Journal of biomechanics* 29.9 (1996), pp. 1223–1230 (cit. on p. 17).

[Deb+21] César Debeunne et al. "CosySLAM: tracking contact features using visual-inertial object-level SLAM for locomotion". In: (2021) (cit. on pp. 5, 54, 107).

[Del12] Frank Dellaert. *Factor graphs and GTSAM: A hands-on introduction*. Tech. rep. Georgia Institute of Technology, 2012 (cit. on p. 24).

[Del+99] Frank Dellaert, Dieter Fox, Wolfram Burgard, and Sebastian Thrun. "Monte carlo localization for mobile robots". In: *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*. Vol. 2. IEEE. 1999, pp. 1322–1328 (cit. on pp. 19, 28).

[DK06] Frank Dellaert and Michael Kaess. "Square Root SAM: Simultaneous localization and mapping via square root information smoothing". In: *The International Journal of Robotics Research* 25.12 (2006), pp. 1181–1203 (cit. on p. 39).

[DK+17] Frank Dellaert, Michael Kaess, et al. "Factor graphs for robot perception". In: *Foundations and Trends® in Robotics* 6.1-2 (2017), pp. 1–139 (cit. on pp. 30, 39).

[Den+19] Maximilian Denninger et al. "BlenderProc". In: *arXiv preprint arXiv:1911.01911* (2019) (cit. on p. 56).

[DAS19] Jeremie Deray, Juan Andrade-Cetto, and Joan Solà. "Joint on-manifold self-calibration of odometry model and sensor extrinsics using pre-integration". In: *European Conference on Mobile Robots*. 2019 (cit. on pp. 68, 78).

[DWE17] Kevin Doherty, Jinkun Wang, and Brendan Englot. "Bayesian generalized kernel inference for occupancy map prediction". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 3118–3124 (cit. on p. 20).

[DL19] Jing Dong and Zhaoyang Lv. "miniSAM: A Flexible Factor Graph Nonlinear Least Squares Optimization Framework". In: *arXiv preprint arXiv:1909.00903* (2019) (cit. on p. 39).

[Don+16] Jing Dong, Mustafa Mukadam, Frank Dellaert, and Byron Boots. "Motion Planning as Probabilistic Inference using Gaussian Processes and Factor Graphs." In: *Robotics: Science and Systems*. Vol. 12. 2016, p. 4 (cit. on p. 39).

[Du+21] Guoguang Du, Kai Wang, Shiguo Lian, and Kaiyong Zhao. "Vision-based robotic grasping from object localization, object pose estimation to grasp estimation for parallel grippers: a review". In: *Artificial Intelligence Review* 54.3 (2021), pp. 1677–1734 (cit. on p. 19).

[Ead18] Ethan Eade. *Derivative of the Exponential Map*. Tech. rep. 2018 (cit. on p. 127).

[EGH19] Kevin Eckenhoff, Patrick Geneva, and Guoquan Huang. "Closed-form preintegration methods for graph-based visual–inertial navigation". In: *The International Journal of Robotics Research* 38.5 (2019), pp. 563–586 (cit. on p. 78).

[ELF97] David W Eggert, Adele Lorusso, and Robert B Fisher. "Estimating 3-D rigid body transformations: a comparison of four major algorithms". In: *Machine vision and applications* 9.5 (1997), pp. 272–290 (cit. on p. 11).

[EKC17] Jakob Engel, Vladlen Koltun, and Daniel Cremers. "Direct sparse odometry". In: *IEEE transactions on pattern analysis and machine intelligence* 40.3 (2017), pp. 611–625 (cit. on p. 43).

[ESC14] Jakob Engel, Thomas Schöps, and Daniel Cremers. "LSD-SLAM: Large-scale direct monocular SLAM". In: *European conference on computer vision*. Springer. 2014, pp. 834–849 (cit. on p. 43).

[Eng+14] Johannes Englsberger et al. "Overview of the torque-controlled humanoid robot TORO". In: *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE. 2014, pp. 916–923 (cit. on p. 14).

[Fal16] Maurice Fallon. "Perception and estimation challenges for humanoid robotics: DARPA Robotics Challenge and NASA Valkyrie". In: *Unmanned/Unattended Sensors and Sensor Networks XII*. Vol. 9986. International Society for Optics and Photonics. 2016, p. 998602 (cit. on p. 20).

[Fal+14] Maurice F Fallon, Matthew Antone, Nicholas Roy, and Seth Teller. "Drift-free humanoid state estimation fusing kinematic, inertial and lidar sensing". In: *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE. 2014, pp. 112–119 (cit. on pp. 10, 11, 13, 14, 17, 20, 21, 92, 107).

[Fal+15] Maurice F Fallon et al. "Continuous humanoid locomotion over uneven terrain using stereo fusion". In: *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. IEEE. 2015, pp. 881–888 (cit. on p. 21).

[Fan+14] Peter Fankhauser, Michael Bloesch, Christian Gehring, Marco Hutter, and Roland Siegwart. "Robot-centric elevation mapping with uncertainty estimates". In: *Mobile Service Robotics*. World Scientific, 2014, pp. 433–440 (cit. on pp. 21, 107).

[FBH18] Peter Fankhauser, Michael Bloesch, and Marco Hutter. "Probabilistic terrain mapping for mobile robots with uncertain localization". In: *IEEE Robotics and Automation Letters* 3.4 (2018), pp. 3019–3026 (cit. on p. 21).

[Fea14] Roy Featherstone. *Rigid body dynamics algorithms*. Springer, 2014 (cit. on pp. 8, 12, 59).

[Fen+15] Siyuan Feng, Eric Whitman, X Xinjilefu, and Christopher G Atkeson. "Optimization-based full body control for the darpa robotics challenge". In: *Journal of Field Robotics* 32.2 (2015), pp. 293–312 (cit. on p. 13).

[Fer+21] Maxime Ferrera, Alexandre Eudes, Julien Moras, Martial Sanfourche, and Guy Le Besnerais. "OV2 SLAM: A Fully Online and Versatile Visual SLAM for Real-Time Applications". In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 1399–1406 (cit. on pp. 23, 43).

[Fia05] Mark Fiala. "ARTag, a fiducial marker system using digital techniques". In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 2. IEEE. 2005, pp. 590–596 (cit. on p. 45).

[FS20] Geoff Fink and Claudio Semini. "The DLS quadruped proprioceptive sensor dataset". In: *Int. Conf. Ser. on Climbing and Walking Robots, Moscow, Russia*. 2020, pp. 1–8 (cit. on p. 119).

[Fla+17] Thomas Flayols, Andrea Del Prete, Patrick Wensing, Alexis Mifsud, Mehdi Benallegue, and Olivier Stasse. "Experimental evaluation of simple estimators for humanoid robots". In: *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE. 2017, pp. 889–895 (cit. on pp. 11, 13, 16).

[Foc+15] Michele Focchi, Victor Barasuol, M. Frigerio, D. Caldwell, and C. Semini. "Slip Detection and Recovery for Quadruped Robots". In: *ISRR*. 2015 (cit. on p. 14).

[For+17] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza. "On-Manifold Preintegration for Real-Time Visual–Inertial Odometry". In: *IEEE Transactions on Robotics* 33.1 (2017), pp. 1–21. DOI: 10.1109/TRO.2016.2597321 (cit. on pp. 23, 24, 65, 66, 68, 71, 73, 76, 78, 91, 92).

[For+15] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. "IMU preintegration on manifold for efficient visual-inertial Maximum a Posteriori estimation". In: Georgia Institute of Technology. 2015 (cit. on pp. 65, 68, 69, 71, 72, 77, 78).

[FPS14] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. "SVO: Fast semi-direct monocular visual odometry". In: *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2014, pp. 15–22 (cit. on pp. 24, 43).

[For+16] Christian Forster, Zichao Zhang, Michael Gassner, Manuel Werlberger, and Davide Scaramuzza. "SVO: Semidirect visual odometry for monocular and multicamera systems". In: *IEEE Transactions on Robotics* 33.2 (2016), pp. 249–265 (cit. on p. 43).

[Fou+19] Mederic Fourmy, Dinesh Atchuthan, Nicolas Mansard, Joan Sola, and Thomas Flayols. "Absolute humanoid localization and mapping based on IMU Lie group and fiducial markers". In: *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*. IEEE. 2019, pp. 237–243 (cit. on pp. 5, 65, 69, 71, 73, 77).

[Fou+21] Médéric Fourmy, Thomas Flayols, Pierre-Alexandre Léziart, Nicolas Mansard, and Joan Solà. "Contact Forces Preintegration for Estimation in Legged Robotics using Factor Graphs". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 1372–1378 (cit. on pp. 5, 11, 68, 69, 79, 86, 99).

[FRR15] Jorge Fuentes-Pacheco, José Ruiz-Ascencio, and Juan Manuel Rendón-Mancha. "Visual simultaneous localization and mapping: a survey". In: *Artificial intelligence review* 43.1 (2015), pp. 55–81 (cit. on p. 43).

[FB10] Paul Furgale and Timothy D Barfoot. "Visual teach and repeat for long-range rover autonomy". In: *Journal of Field Robotics* 27.5 (2010), pp. 534–560 (cit. on pp. 19, 20, 107).

[FRS13] Paul Furgale, Joern Rehder, and Roland Siegwart. "Unified temporal and spatial calibration for multi-sensor systems". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2013, pp. 1280–1286 (cit. on p. 94).

[GG16] Yarin Gal and Zoubin Ghahramani. "Dropout as a bayesian approximation: Representing model uncertainty in deep learning". In: *international conference on machine learning*. PMLR. 2016, pp. 1050–1059 (cit. on p. 54).

[Gan+20] Lu Gan, Ray Zhang, Jessy W Grizzle, Ryan M Eustice, and Maani Ghaffari. "Bayesian spatial kernel smoothing for scalable dense semantic mapping". In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 790–797 (cit. on p. 20).

[Gao+03] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. "Complete solution classification for the perspective-three-point problem". In: *IEEE transactions on pattern analysis and machine intelligence* 25.8 (2003), pp. 930–943 (cit. on pp. 45, 46, 49).

[Gar+14] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco José Madrid-Cuevas, and Manuel Jesús Marin-Jiménez. "Automatic generation and detection of highly reliable fiducial markers under occlusion". In: *Pattern Recognition* 47.6 (2014), pp. 2280–2292 (cit. on p. 45).

[Gas+05] Bernd Gassmann, Franziska Zacharias, J Marius Zollner, and Rüdiger Dillmann. "Localization of walking robots". In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. IEEE. 2005, pp. 1471–1476 (cit. on pp. 11, 13).

[GS13] Martin Görner and Annett Stelzer. "A leg proprioception based 6 DOF odometry for statically stable walking robots". In: *Autonomous Robots* 34.4 (2013), pp. 311–326 (cit. on p. 13).

[Gri+20] F. Grimminger et al. "An Open Torque-Controlled Modular Robot Architecture for Legged Locomotion Research". In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 3650–3657. DOI: 10.1109/LRA.2020.2976639 (cit. on pp. 8, 10, 110, 119).

[Gri+11] Giorgio Grisetti, Rainer Kümmerle, Hauke Strasdat, and Kurt Konolige. "g2o: A general framework for (hyper) graph optimization". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China*. 2011, pp. 9–13 (cit. on pp. 22, 40).

[GS12] Ozlem Gur and ULUÇ Saranli. "Model-based proprioceptive state estimation for springmass running". In: *Robotics: Science and Systems VII* (2012), pp. 105–112 (cit. on p. 13).

[Gus+02] Fredrik Gustafsson et al. "Particle filters for positioning, navigation, and tracking". In: *IEEE Transactions on signal processing* 50.2 (2002), pp. 425–437 (cit. on p. 28).

[Har+18a] Omar Harib et al. "Feedback control of an exoskeleton for paraplegics: Toward robustly stable, hands-free dynamic walking". In: *IEEE Control Systems Magazine* 38.6 (2018), pp. 61–87 (cit. on pp. 8, 16).

[Har+20] Ross Hartley, Maani Ghaffari, Ryan M Eustice, and Jessy W Grizzle. "Contact-aided invariant extended Kalman filtering for robot state estimation". In: *The International Journal of Robotics Research* 39.4 (2020), pp. 402–430 (cit. on pp. 11, 13, 77).

[Har+18b] Ross Hartley, Maani Ghaffari Jadidi, Lu Gan, Jiunn-Kai Huang, Jessy W Grizzle, and Ryan M Eustice. "Hybrid contact preintegration for visual-inertial-contact state estimation using factor graphs". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 3783–3790 (cit. on pp. 11, 20, 24).

[Har+18c] Ross Hartley et al. "Legged robot state-estimation through combined forward kinematic and preintegrated contact factors". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 4422–4429 (cit. on pp. 11, 20, 24, 61, 79, 92).

[HB21] Rasmus Laurvig Haugaard and Anders Glent Buch. "SurfEmb: Dense and Continuous Correspondence Distributions for Object Pose Estimation with Learnt Surface Embeddings". In: *arXiv preprint arXiv:2111.13489* (2021) (cit. on p. 52).

[HZG19] Guoping He, Shangkun Zhong, and Jifeng Guo. "A lightweight and scalable visual-inertial motion capture system using fiducial markers". In: *Autonomous Robots* 43.7 (2019), pp. 1895–1915 (cit. on p. 93).

[He+18] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. *Mask R-CNN*. 2018. arXiv: 1703.06870 [cs.CV] (cit. on p. 52).

[Hen+17] Bernd Henze, Alexander Dietrich, Máximo A Roa, and Christian Ott. "Multi-contact balancing of humanoid robots in confined spaces: Utilizing knee contacts". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 697–704 (cit. on p. 19).

[HA17] Ayonga Hereid and Aaron D Ames. "Frost: Fast robot optimization and simulation toolkit". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 719–726 (cit. on p. 10).

[Hig75] Walter T Higgins. "A comparison of complementary and Kalman filtering". In: *IEEE Transactions on Aerospace and Electronic Systems* 3 (1975), pp. 321–325 (cit. on p. 13).

[Hil+14] Aharon Bar Hillel, Ronen Lerner, Dan Levi, and Guy Raz. "Recent progress in road and lane detection: a survey". In: *Machine vision and applications* 25.3 (2014), pp. 727–745 (cit. on p. 2).

[Hod+17] Tomáš Hodan, Pavel Haluza, Štepán Obdržálek, Jiri Matas, Manolis Lourakis, and Xenophon Zabulis. "T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects". In: *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE. 2017, pp. 880–888 (cit. on p. 111).

[Hod+20] Tomáš Hodaň et al. "BOP Challenge 2020 on 6D Object Localization". In: *European Conference on Computer Vision Workshops (ECCVW)* (2020) (cit. on p. 52).

[Hor+14] Armin Hornung, Stefan Oßwald, Daniel Maier, and Maren Bennewitz. "Monte Carlo localization for humanoid robot navigation in complex indoor environments". In: *International Journal of Humanoid Robotics* 11.02 (2014), p. 1441002 (cit. on pp. 20, 21).

[Hor+13] Armin Hornung, Kai M Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. "OctoMap: An efficient probabilistic 3D mapping framework based on octrees". In: *Autonomous robots* 34.3 (2013), pp. 189–206 (cit. on p. 20).

[HG21] Jiunn-Kai Huang and Jessy W Grizzle. "Efficient anytime clf reactive planning system for a bipedal robot on undulating terrain". In: *arXiv preprint arXiv:2108.06699* (2021) (cit. on p. 21).

[HJ15] AA Hussen and IN Jleta. "Low-cost inertial sensors modeling using Allan variance". In: *International Journal of Computer, Electrical, Automation, Control and Information Engineering* 9.5 (2015), pp. 1237–1242 (cit. on p. 66).

[Hut+16] Marco Hutter et al. "Anymal-a highly mobile and dynamic quadrupedal robot". In: *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE. 2016, pp. 38–44 (cit. on p. 7).

[Hwa+16] Jemin Hwangbo, Carmine Dario Bellicoso, Péter Fankhauser, and Marco Hutter. "Probabilistic foot contact estimation by fusing information from dynamics and differential/forward kinematics". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 3872–3878 (cit. on pp. 14, 15).

[Ila+17] Viorela Ila, Lukas Polok, Marek Solony, and Pavel Svoboda. "SLAM++-A highly efficient and temporally scalable incremental SLAM framework". In: *The International Journal of Robotics Research* 36.2 (2017), pp. 210–230 (cit. on pp. 22, 40).

[Ind+13] V. Indelman, S. Williams, Michael Kaess, and F. Dellaert. "Information Fusion in Navigation Systems via Factor Graph Based Incremental Smoothing". In: *Robotics and Autonomous Systems* 61.8 (Aug. 2013), pp. 721–738 (cit. on p. 78).

[Jen+19] Fabian Jenelten, Jemin Hwangbo, Fabian Tresoldi, C Dario Bellicoso, and Marco Hutter. "Dynamic Locomotion on Slippery Ground". In: *IEEE Robotics and Automation Letters* 4.4 (2019), pp. 4170–4176 (cit. on p. 14).

[JBD12] Yong-Dian Jian, Doru C Balcan, and Frank Dellaert. "Generalized subgraph preconditioners for large-scale bundle adjustment". In: *Outdoor and Large-Scale Real-World Scene Analysis*. Springer, 2012, pp. 131–150 (cit. on p. 32).

[JMS17] P. Jin, P. Matikainen, and S. S. Srinivasa. "Sensor fusion for fiducial tags: Highly robust pose estimation from single frame RGBD". In: *2017 IEEE/RSJ IROS*. 2017 (cit. on p. 46).

[Joh+15] Matthew Johnson et al. "Team IHMC's lessons learned from the DARPA robotics challenge trials". In: *Journal of Field Robotics* 32.2 (2015), pp. 192–208 (cit. on pp. 11, 13).

[Jos+20] Laurent Valentin Jospin, Wray Buntine, Farid Boussaid, Hamid Laga, and Mohammed Bennamoun. "Hands-on Bayesian Neural Networks–a Tutorial for Deep Learning Users". In: *arXiv preprint arXiv:2007.06823* (2020) (cit. on pp. 54, 113).

[Kae+12] Michael Kaess, Hordur Johannsson, Richard Roberts, Viorela Ila, John J Leonard, and Frank Dellaert. "iSAM2: Incremental smoothing and mapping using the Bayes tree". In: *The International Journal of Robotics Research* 31.2 (2012), pp. 216–235 (cit. on pp. 22, 40).

[Kaj+01] Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kazuhito Yokoi, and Hirohisa Hirukawa. "The 3D linear inverted pendulum mode: A simple modeling for a biped walking pattern generation". In: *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)*. Vol. 1. IEEE. 2001, pp. 239–246 (cit. on pp. 17, 18).

[Kal60] Rudolph Emil Kalman. "A new approach to linear filtering and prediction problems". In: (1960) (cit. on pp. 7, 11, 13).

[KMH19] Mike Kasper, Steve McGuire, and Christoffer Heckman. "A benchmark for visual-inertial odometry systems employing onboard illumination". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2019, pp. 5256–5263 (cit. on p. 119).

[KB99] Hirokazu Kato and Mark Billinghurst. "Marker tracking and hmd calibration for a video-based augmented reality conferencing system". In: *Proceedings 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR'99)*. IEEE. 1999, pp. 85–94 (cit. on p. 45).

[Kim+20] Donghyun Kim et al. “Vision aided dynamic exploration of unstructured terrain with a small-scale quadruped robot”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 2464–2470 (cit. on pp. 21, 107).

[Kim+21] Joon-Ha Kim et al. “Legged Robot State Estimation With Dynamic Contact Event Information”. In: *IEEE Robotics and Automation Letters* 6.4 (2021), pp. 6733–6740 (cit. on pp. 11, 24).

[KM09] Georg Klein and David Murray. “Parallel tracking and mapping on a camera phone”. In: *2009 8th IEEE International Symposium on Mixed and Augmented Reality*. IEEE. 2009, pp. 83–86 (cit. on p. 22).

[KD07] Alexander Kleiner and Christian Dornhege. “Real-time localization and elevation mapping within urban search and rescue scenarios”. In: *Journal of Field Robotics* 24.8-9 (2007), pp. 723–745 (cit. on p. 21).

[KF09] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009 (cit. on pp. 28, 39).

[KKN09] J Zico Kolter, Youngjun Kim, and Andrew Y Ng. “Stereo vision and terrain modeling for quadruped robots”. In: *2009 IEEE International Conference on Robotics and Automation*. IEEE. 2009, pp. 1557–1564 (cit. on p. 21).

[KD20] Rebecca König and Bertram Drost. “A hybrid approach for 6DoF pose estimation”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 700–706 (cit. on p. 52).

[Kon05] Kurt Konolige. “SLAM via variable reduction from constraint maps”. In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. IEEE. 2005, pp. 667–672 (cit. on p. 121).

[Koo+16] Twan Koolen et al. “Design of a momentum-based control framework and application to the humanoid robot atlas”. In: *International Journal of Humanoid Robotics* 13.01 (2016), p. 1650007 (cit. on pp. 10, 19, 20).

[Kul+21] Mihir Kulkarni et al. “Autonomous Teamed Exploration of Subterranean Environments using Legged and Aerial Robots”. In: *arXiv preprint arXiv:2111.06482* (2021) (cit. on p. 19).

[Kwa+09] Nosan Kwak, Olivier Stasse, Torea Foissotte, and Kazuhito Yokoi. “3D grid and particle based SLAM for a humanoid robot”. In: *2009 9th IEEE-RAS International Conference on Humanoid Robots*. IEEE. 2009, pp. 62–67 (cit. on p. 20).

[Lab+20] Yann Labbé, Justin Carpentier, Mathieu Aubry, and Josef Sivic. “Cosypose: Consistent multi-view multi-object 6d pose estimation”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 574–591 (cit. on pp. 44, 52, 54, 107).

[Lab+21] Yann Labbé, Justin Carpentier, Mathieu Aubry, and Josef Sivic. “Single-view robot pose and joint angle estimation via render & compare”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 1654–1663 (cit. on p. 52).

[LBB22] Marc Lambert, Silvere Bonnabel, and Francis Bach. “The recursive variational Gaussian approximation (R-VGA)”. In: *Statistics and Computing* 32.1 (2022), pp. 1–24 (cit. on p. 28).

[LVH20] Cedric Le Gentil, Teresa Vidal-Calleja, and Shoudong Huang. "Gaussian process preintegration for inertial-aided state estimation". In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 2108–2114 (cit. on p. 78).

[LAP11] Vincent Lebastard, Yannick Aoustin, and Franck Plestan. "Estimation of absolute orientation for a bipedal robot: Experimental results". In: *IEEE Transactions on Robotics* 27.1 (2011), pp. 170–174 (cit. on p. 13).

[Lee+20] Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. "Learning quadrupedal locomotion over challenging terrain". In: *Science robotics* 5.47 (2020) (cit. on p. 19).

[LB16] John J Leonard and Alexander Bahr. "Autonomous underwater vehicle navigation". In: *Springer Handbook of Ocean Engineering* (2016), pp. 341–358 (cit. on p. 7).

[LMF09] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. "Epnp: An accurate o (n) solution to the pnp problem". In: *International journal of computer vision* 81.2 (2009), p. 155 (cit. on p. 46).

[Leu+15] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. "Keyframe-based visual–inertial odometry using nonlinear optimization". In: *The International Journal of Robotics Research* 34.3 (2015), pp. 314–334 (cit. on pp. 23, 92, 97).

[Lev44] Kenneth Levenberg. "A method for the solution of certain non-linear problems in least squares". In: *Quarterly of applied mathematics* 2.2 (1944), pp. 164–168 (cit. on p. 32).

[Léz+21] Pierre-Alexandre Léziart, Thomas Flayols, Felix Grimminger, Nicolas Mansard, and Philippe Souères. "Implementation of a Reactive Walking Controller for the New Open-Hardware Quadruped Solo-12". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 5007–5013 (cit. on pp. 13, 14, 106, 112, 115).

[Li+19] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. "DeepIM: Deep Iterative Matching for 6D Pose Estimation". In: *International Journal of Computer Vision* 128.3 (Nov. 2019), pp. 657–678. ISSN: 1573-1405. DOI: 10.1007/s11263-019-01250-9. URL: http://dx.doi.org/10.1007/s11263-019-01250-9 (cit. on p. 52).

[LKK05] Pei-Chun Lin, Haldun Komsuoglu, and Daniel E Koditschek. "A leg configuration measurement system for full-body pose estimates in a hexapod robot". In: *IEEE Transactions on robotics* 21.3 (2005), pp. 411–422 (cit. on pp. 11, 13).

[LKK06] Pei-Chun Lin, Haldun Komsuoglu, and Daniel E Koditschek. "Sensor data fusion for body state estimation in a hexapod robot with dynamical gaits". In: *IEEE Transactions on Robotics* 22.5 (2006), pp. 932–943 (cit. on pp. 11, 13).

[Lin+21] Tzu-Yuan Lin, Ray Zhang, Justin Yu, and Maani Ghaffari. "Deep Multi-Modal Contact Estimation for Invariant Observer Design on Quadruped Robots". In: *arXiv preprint arXiv:2106.15713* (2021) (cit. on pp. 9, 13, 15).

[Loe04] H-A Loeliger. "An introduction to factor graphs". In: *IEEE Signal Processing Magazine* 21.1 (2004), pp. 28–41 (cit. on p. 39).

[LM97] Feng Lu and Evangelos Milios. “Globally consistent range scan alignment for environment mapping”. In: *Autonomous robots* 4.4 (1997), pp. 333–349 (cit. on p. 22).

[LGL+21] Yarong Luo, Chi Guo, Jingnan Liu, et al. “The Unified Mathematical Framework for IMU Preintegration in Inertial-Aided Navigation System”. In: *arXiv preprint arXiv:2111.09100* (2021) (cit. on p. 78).

[LS09] Todd Lupton and Salah Sukkarieh. “Efficient Integration of Inertial Observations into Visual SLAM without Initialization”. In: *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2009 (cit. on pp. 23, 65, 66, 68, 72, 78).

[Ma+12] Jeremy Ma, Sara Susca, Max Bajracharya, Larry Matthies, Matt Malchano, and Dave Wooden. “Robust multi-sensor, day/night 6-DOF pose estimation for a dynamic legged vehicle in GPS-denied environments”. In: *2012 IEEE International Conference on Robotics and Automation*. IEEE. 2012, pp. 619–626 (cit. on pp. 11, 13).

[Mar63] Donald W Marquardt. “An algorithm for least-squares estimation of nonlinear parameters”. In: *Journal of the society for Industrial and Applied Mathematics* 11.2 (1963), pp. 431–441 (cit. on p. 32).

[Mas+20a] Carlos Mastalli, Ioannis Havoutis, Michele Focchi, Darwin G Caldwell, and Claudio Semini. “Motion planning for quadrupedal locomotion: Coupled planning, terrain mapping, and whole-body control”. In: *IEEE Transactions on Robotics* 36.6 (2020), pp. 1635–1648 (cit. on p. 21).

[Mas+15] Carlos Mastalli, Ioannis Havoutis, Alexander W Winkler, Darwin G Caldwell, and Claudio Semini. “On-line and on-board planning and perception for quadrupedal locomotion”. In: *2015 IEEE International Conference on Technologies for Practical Robot Applications (TePRA)*. IEEE. 2015, pp. 1–7 (cit. on p. 21).

[Mas+20b] Carlos Mastalli et al. “Crocoddyl: An efficient and versatile framework for multi-contact optimal control”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 2536–2542 (cit. on p. 19).

[MCF22] Matas Mattamala, Nived Chebrolu, and Maurice Fallon. “An Efficient Locally Reactive Controller for Safe Navigation in Visual Teach and Repeat Missions”. In: *arXiv preprint arXiv:2201.03938* (2022) (cit. on p. 20).

[Mat+21] Matias Mattamala, Milad Ramezani, Marco Camurri, and Maurice Fallon. “Learning Camera Performance Models for Active Multi-Camera Visual Teach and Repeat”. In: *arXiv preprint arXiv:2103.14070* (2021) (cit. on pp. 20, 107).

[McE18] Richard McElreath. *Statistical rethinking: A Bayesian course with examples in R and Stan*. Chapman and Hall/CRC, 2018 (cit. on pp. 121, 122).

[McG+85] Leonard A McGee, Stanley F Schmidt, Leonard A Mcgee, and Stanley F Sc. “Discovery of the Kalman Filter as a Practical Tool for Aerospace and”. In: *Industry,” National Aeronautics and Space Administration, Ames Research*. Citeseer. 1985 (cit. on p. 7).

[MM76] Harry McGurk and John MacDonald. “Hearing lips and seeing voices”. In: *Nature* 264.5588 (1976), pp. 746–748 (cit. on p. 3).

[MS83] M Alex Meredith and Barry E Stein. “Interactions among converging sensory inputs in the superior colliculus”. In: *Science* 221.4608 (1983), pp. 389–391 (cit. on p. 3).

[ML81] FA Miles and SG Lisberger. “Plasticity in the vestibulo-ocular reflex: a new hypothesis.” In: *Annual review of neuroscience* (1981) (cit. on p. 3).

[MD05] Lee M Miller and Mark D’esposito. “Perceptual fusion and stimulus coincidence in the cross-modal integration of speech”. In: *Journal of Neuroscience* 25.25 (2005), pp. 5884–5893 (cit. on p. 3).

[Mio+20] Nina Miolane et al. “Geomstats: A Python Package for Riemannian Geometry in Machine Learning”. In: *Journal of Machine Learning Research* 21.223 (2020), pp. 1–9 (cit. on p. 33).

[Mon+02] Michael Montemerlo, Sebastian Thrun, Daphne Koller, Ben Wegbreit, et al. “FastSLAM: A factored solution to the simultaneous localization and mapping problem”. In: *Aaai/iaai* 593598 (2002) (cit. on p. 28).

[MMT15] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. “ORB-SLAM: a versatile and accurate monocular SLAM system”. In: *IEEE transactions on robotics* 31.5 (2015), pp. 1147–1163 (cit. on pp. 23, 43).

[NBB16] Michael Neunert, Michael Bloesch, and Jonas Buchli. “An open source, fiducial based, visual-inertial motion capture system”. In: *2016 19th International Conference on Information Fusion (FUSION)*. IEEE. 2016 (cit. on p. 93).

[NLD11] Richard A Newcombe, Steven J Lovegrove, and Andrew J Davison. “DTAM: Dense tracking and mapping in real-time”. In: *2011 international conference on computer vision*. IEEE. 2011, pp. 2320–2327 (cit. on p. 43).

[Nil84] Nils J. Nilsson. “Shakey the Robot”. In: 1984 (cit. on p. 1).

[Nis+19] Barza Nisar, Philipp Foehn, Davide Falanga, and Davide Scaramuzza. “Vimo: Simultaneous visual inertial model-based odometry and force estimation”. In: *IEEE Robotics and Automation Letters* 4.3 (2019), pp. 2785–2792 (cit. on pp. 23, 79).

[Nob+17] Simona Nobili et al. “Heterogeneous sensor fusion for accurate state estimation of dynamic legged robots”. In: (2017) (cit. on pp. 21, 92).

[NKT15] Francesco Nori, Naveen Kuppuswamy, and Silvio Traversaro. “Simultaneous state and dynamics estimation in articulated structures”. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2015, pp. 3380–3386. DOI: 10.1109/IROS.2015.7353848 (cit. on p. 4).

[Ori+12] Giuseppe Oriolo, Antonio Paolillo, Lorenzo Rosa, and Marilena Vendittelli. “Vision-based odometric localization for humanoids using a kinematic EKF”. In: *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*. IEEE. 2012, pp. 153–158 (cit. on p. 20).

[Ori+16] Giuseppe Oriolo, Antonio Paolillo, Lorenzo Rosa, and Marilena Vendittelli. “Humanoid odometric localization integrating kinematic, inertial and visual information”. In: *Autonomous Robots* 40.5 (2016), pp. 867–879 (cit. on p. 20).

[Ped+11] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830 (cit. on p. 55).

[Pen18] Roger D Peng. “Advanced statistical computing”. In: *Work in progress* (2018) (cit. on p. 121).

[Pfi+00] Hanspeter Pfister, Matthias Zwicker, Jeroen Van Baar, and Markus Gross. “Surfels: Surface elements as rendering primitives”. In: *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. 2000, pp. 335–342 (cit. on p. 22).

[PL15] Sudeep Pillai and John Leonard. “Monocular slam supported object recognition”. In: *arXiv preprint arXiv:1506.01732* (2015) (cit. on p. 108).

[PKT18] Stylianos Piperakis, Maria Koskinopoulou, and Panos Trahanias. “Nonlinear state estimation for humanoid robot walking”. In: *IEEE Robotics and Automation Letters* 3.4 (2018), pp. 3347–3354 (cit. on pp. 18, 99).

[PT16] Stylianos Piperakis and Panos Trahanias. “Non-linear ZMP based state estimation for humanoid robot locomotion”. In: *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE. 2016, pp. 202–209 (cit. on p. 18).

[Pre+16] Andrea Del Prete, Nicolas Mansard, Oscar E Ramos, Olivier Stasse, and Francesco Nori. “Implementing Torque Control with High-Ratio Gear Boxes and without Joint-Torque Sensors”. In: *Int. Journal of Humanoid Robotics*. 2016, p. 1550044. URL: https://hal.archives-ouvertes.fr/hal-01136936/document (cit. on p. 101).

[QLS18] Tong Qin, Peiliang Li, and Shaojie Shen. “Vins-mono: A robust and versatile monocular visual-inertial state estimator”. In: *IEEE Transactions on Robotics* 34.4 (2018), pp. 1004–1020 (cit. on pp. 23, 78).

[Qua+19] Meixiang Quan, Songhao Piao, Minglang Tan, and Shi-Sheng Huang. “Tightly-coupled monocular visual-odometric slam using wheels and a mems gyroscope”. In: *IEEE Access* 7 (2019), pp. 97374–97389 (cit. on p. 78).

[RRM97] S Chenchal Rao, Gregor Rainer, and Earl K Miller. “Integration of what and where in the primate prefrontal cortex”. In: *Science* 276.5313 (1997), pp. 821–824 (cit. on p. 3).

[RMA19] Jacob Reher, Wen-Loong Ma, and Aaron D Ames. “Dynamic walking with compliance on a cassie bipedal robot”. In: *2019 18th European Control Conference (ECC)*. IEEE. 2019, pp. 2589–2595 (cit. on p. 21).

[RH11] Michal Reinstein and Matej Hoffmann. “Dead reckoning in a dynamic quadruped robot: Inertial navigation system aided by a legged odometer”. In: *2011 IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 617–624 (cit. on pp. 11, 13).

[RMM18] Francisco J Romero-Ramirez, Rafael Muñoz-Salinas, and Rafael Medina-Carnicer. “Speeded up detection of squared fiducial markers”. In: *Image and vision Computing* 76 (2018), pp. 38–47 (cit. on p. 45).

[RK91] Gerald P Roston and Eric P Krotkov. *Dead reckoning navigation for walking robots*. Tech. rep. CARNEGIE-MELLON UNIV PITTSBURGH PA ROBOTICS INST, 1991 (cit. on pp. 11, 62).

[Rot+14] Nicholas Rotella, Michael Bloesch, Ludovic Righetti, and Stefan Schaal. "State estimation for a humanoid robot". In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2014, pp. 952–958 (cit. on pp. 10, 11, 13, 24).

[Rot+15] Nicholas Rotella, Alexander Herzog, Stefan Schaal, and Ludovic Righetti. "Humanoid momentum estimation using sensed contact wrenches". In: *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. IEEE. 2015, pp. 556–563 (cit. on pp. 19, 86, 100).

[Rot+] Nicholas Rotella, Sean Mason, Stefan Schaal, and Ludovic Righetti. "IMU-based joint state estimation for humanoid control". In: (cit. on pp. 10, 11, 16).

[RSR18] Nicholas Rotella, Stefan Schaal, and Ludovic Righetti. "Unsupervised contact learning for humanoid estimation and control". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 411–417 (cit. on pp. 9, 15).

[Rou+19] Tomáš Rouček et al. "Darpa subterranean challenge: Multi-robotic exploration of underground environments". In: *International Conference on Modelling and Simulation for Autonomous Systesm*. Springer, Cham. 2019, pp. 274–290 (cit. on p. 19).

[Sal+13] Renato F. Salas-Moreno, Richard A. Newcombe, H. Strasdat, P. Kelly, and A. Davison. "SLAM++: Simultaneous Localisation and Mapping at the Level of Objects". In: *2013 IEEE Conference on Computer Vision and Pattern Recognition* (2013), pp. 1352–1359 (cit. on pp. 54, 108).

[SB04] Philippe Sardain and Guy Bessonnet. "Forces acting on a biped robot. Center of pressure-zero moment point". In: *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 34.5 (2004), pp. 630–637 (cit. on p. 18).

[SF11] Davide Scaramuzza and Friedrich Fraundorfer. "Visual odometry [tutorial]". In: *IEEE robotics & automation magazine* 18.4 (2011), pp. 80–92 (cit. on p. 19).

[SF16] Johannes Lutz Schönberger and Jan-Michael Frahm. "Structure-from-Motion Revisited". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016 (cit. on p. 22).

[Sco+17] Raluca Scona, Simona Nobili, Yvan R Petillot, and Maurice Fallon. "Direct visual SLAM fusing proprioception for a humanoid robot". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 1419–1426 (cit. on pp. 9, 22).

[SMK15] Shaojie Shen, Nathan Michael, and Vijay Kumar. "Tightly-coupled monocular visual-inertial fusion for autonomous flight of rotorcraft MAVs". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2015, pp. 5303–5310 (cit. on p. 78).

[SSM62] Gerald L Smith, Stanley F Schmidt, and Leonard A McGee. *Application of statistical filter theory to the optimal estimation of position and velocity on board a circumlunar vehicle*. National Aeronautics and Space Administration, 1962 (cit. on p. 7).

[Sol12] Joan Sola. "Quaternion kinematics for the error-state KF". In: *Laboratoire dAnalyse et dArchitecture des Systemes-Centre national de la recherche scientifique (LAAS-CNRS), Toulouse, France, Tech. Rep* (2012) (cit. on p. 35).

[Sol17] Joan Sola. *Course on SLAM*. Tech. rep. Technical Report IRI-TR-16-04, Institut de Robòtica i, 2017 (cit. on pp. 30, 31).

[Sol+21] Joan Sola et al. "WOLF: A modular estimation framework for robotics based on factor graphs". In: *arXiv preprint arXiv:2110.12919* (2021) (cit. on pp. 4, 5, 24, 30, 79, 94, 101, 118).

[SDA18] Joan Solà, Jérémie Deray, and Dinesh Atchuthan. "A micro Lie theory for state estimation in robotics". In: (2018) (cit. on pp. 24, 34–37, 44, 69, 70, 75, 76, 126, 127).

[Sta+06] Olivier Stasse, Andrew J Davison, Ramzi Sellaouti, and Kazuhito Yokoi. "Real-time 3d slam for humanoid robot considering pattern generator information". In: *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2006, pp. 348–355 (cit. on p. 20).

[Sta+17] Olivier Stasse et al. "TALOS: A new humanoid research platform targeted for industrial applications". In: *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE. 2017, pp. 689–695 (cit. on pp. 8, 14, 16).

[Ste11] Benjamin J Stephens. "State estimation for force-controlled humanoid balance using simple models in the presence of modeling error". In: *2011 IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 3994–3999 (cit. on p. 18).

[SMD12] Hauke Strasdat, José MM Montiel, and Andrew J Davison. "Visual SLAM: why filter?" In: *Image and Vision Computing* 30.2 (2012), pp. 65–77 (cit. on p. 23).

[SWD20] Edgar Sucar, Kentaro Wada, and Andrew Davison. "NodeSLAM: Neural object descriptors for multi-view shape reconstruction". In: *2020 International Conference on 3D Vision (3DV)*. IEEE. 2020, pp. 949–958 (cit. on p. 108).

[Sug+06] Tadashi Sugihara, Mark D Diltz, Bruno B Averbeck, and Lizabeth M Romanski. "Integration of auditory and visual communication information in the primate ventrolateral prefrontal cortex". In: *Journal of Neuroscience* 26.43 (2006), pp. 11138–11147 (cit. on p. 3).

[Tan+18] Jie Tan et al. "Sim-to-real: Learning agile locomotion for quadruped robots". In: *arXiv preprint arXiv:1804.10332* (2018) (cit. on p. 19).

[TL20a] Mingxing Tan and Quoc V. Le. *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. 2020. arXiv: 1905.11946 [cs.LG] (cit. on p. 53).

[TL20b] George Terzakis and Manolis Lourakis. "A consistently fast and globally optimal solution to the perspective-n-point problem". In: *European Conference on Computer Vision*. Springer. 2020, pp. 478–494 (cit. on p. 46).

[Thr+04] Sebastian Thrun, Yufeng Liu, Daphne Koller, Andrew Y Ng, Zoubin Ghahramani, and Hugh Durrant-Whyte. "Simultaneous localization and mapping with sparse extended information filters". In: *The international journal of robotics research* 23.7-8 (2004), pp. 693–716 (cit. on p. 7).

[Thr+06] Sebastian Thrun et al. “Stanley: The robot that won the DARPA Grand Challenge”. In: *Journal of field Robotics* 23.9 (2006), pp. 661–692 (cit. on p. 2).

[Ton+18] Steve Tonneau, Andrea Del Prete, Julien Pettré, Chonhyon Park, Dinesh Manocha, and Nicolas Mansard. “An efficient acyclic contact planner for multiped robots”. In: *IEEE Transactions on Robotics* 34.3 (2018), pp. 586–601 (cit. on p. 21).

[Ton+20] Steve Tonneau, Daeun Song, Pierre Fernbach, Nicolas Mansard, Michel Taix, and Andrea Del Prete. “SL1M: Sparse L1-norm Minimization for contact planning on uneven terrain”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 6604–6610 (cit. on p. 21).

[Tri+99] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. “Bundle adjustment—a modern synthesis”. In: *International workshop on vision algorithms*. Springer. 1999, pp. 298–372 (cit. on pp. 19, 22).

[ULH16] Steffen Urban, Jens Leitloff, and Stefan Hinz. “Mlpnp-a real-time maximum likelihood solution to the perspective-n-point problem”. In: *arXiv preprint arXiv:1607.08112* (2016) (cit. on pp. 47, 50).

[Urm+08] Chris Urmson et al. “Autonomous driving in urban environments: Boss and the urban challenge”. In: *Journal of Field Robotics* 25.8 (2008), pp. 425–466 (cit. on p. 2).

[Vig+18] Matthieu Vigne, Antonio El Khoury, Matthieu Masselin, Florent Di Meglio, and Nicolas Petit. “Estimation of multiple flexibilities of an articulated system using inertial measurements”. In: *2018 IEEE Conference on Decision and Control (CDC)*. IEEE. 2018, pp. 6779–6785 (cit. on pp. 10, 16, 17).

[Vig+22] Matthieu Vigne, Antonio El Khoury, Marine Pétriaux, Florent Meglio, and Nicolas Petit. “MOVIE: a Velocity-aided IMU Attitude Estimator for Observing and Controlling Multiple Deformations on Legged Robots”. In: (2022) (cit. on p. 16).

[Vil+22] Nahuel Villa, Pierre Fernbach, Nicolas Mansard, and Olivier Stasse. “Addressing flexibility in biped locomotion with robust control and closed-loop model-predictive control”. In: *International Conference on Robotics and Automation (ICRA)*. 2022 (cit. on pp. 10, 16).

[WM86] KENNETHJ Waldron and ROBERTB McGhee. “The adaptive suspension vehicle”. In: *IEEE Control Systems Magazine* 6.6 (1986), pp. 7–12 (cit. on p. 11).

[WVH01] Eric A Wan, Rudolph Van Der Merwe, and Simon Haykin. “The unscented Kalman filter”. In: *Kalman filtering and neural networks* 5.2007 (2001), pp. 221–280 (cit. on p. 7).

[WO16] John Wang and Edwin Olson. “AprilTag 2: Efficient and robust fiducial detection”. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2016 (cit. on pp. 44, 45, 47, 48, 91).

[Whe+12] Thomas Whelan, Michael Kaess, Maurice Fallon, Hordur Johannsson, John Leonard, and John McDonald. “Kintinuous: Spatially extended kinectfusion”. In: (2012) (cit. on p. 21).

[Whe+16] Thomas Whelan, Renato F Salas-Moreno, Ben Glocker, Andrew J Davison, and Stefan Leutenegger. "ElasticFusion: Real-time dense SLAM and light source estimation". In: *The International Journal of Robotics Research* 35.14 (2016), pp. 1697–1716 (cit. on p. 22).

[Wie06a] P-B Wieber. "Holonomy and nonholonomy in the dynamics of articulated motion". In: *Fast motions in biomechanics and robotics*. Springer, 2006, pp. 411–425 (cit. on p. 17).

[Wie00] Pierre-Brice Wieber. "Modélisation et commande d'un robot marcheur anthropomorphe". PhD thesis. ANRT [diff.], 2000 (cit. on p. 17).

[Wie06b] Pierre-Brice Wieber. "Trajectory free linear model predictive control for stable walking in the presence of strong perturbations". In: *2006 6th IEEE-RAS International Conference on Humanoid Robots*. IEEE. 2006, pp. 137–142 (cit. on p. 17).

[Win+15] Alexander W Winkler, Carlos Mastalli, Ioannis Havoutis, Michele Focchi, Darwin G Caldwell, and Claudio Semini. "Planning and execution of dynamic whole-body locomotion for a hydraulic quadruped on challenging terrain". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2015, pp. 5148–5154 (cit. on p. 21).

[WCF19] David Wisth, Marco Camurri, and Maurice Fallon. "Robust legged robot state estimation using factor graph optimization". In: *IEEE Robotics and Automation Letters* 4.4 (2019), pp. 4507–4514 (cit. on pp. 24, 79, 92, 106).

[WCF20] David Wisth, Marco Camurri, and Maurice Fallon. "Preintegrated velocity bias estimation to overcome contact nonlinearities in legged robot odometry". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 392–398 (cit. on pp. 11, 24, 79).

[WCF21] David Wisth, Marco Camurri, and Maurice Fallon. "VILENS: Visual, inertial, lidar, and leg odometry for all-terrain legged robots". In: *arXiv preprint arXiv:2107.07243* (2021) (cit. on pp. 4, 24, 107).

[Won+20] Jeremy Nathan Wong, David Juny Yoon, Angela P Schoellig, and Timothy D Barfoot. "Variational inference with parameter learning applied to vehicle trajectory estimation". In: *IEEE Robotics and Automation Letters* 5.4 (2020), pp. 5291–5298 (cit. on p. 28).

[XA12] X Xinjilefu and Christopher G Atkeson. "State estimation of a walking humanoid robot". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2012, pp. 3693–3699 (cit. on p. 18).

[XFA14] X Xinjilefu, Siyuan Feng, and Christopher G Atkeson. "Dynamic state estimation using quadratic programming". In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2014, pp. 989–994 (cit. on p. 18).

[XFA15] X Xinjilefu, Siyuan Feng, and Christopher G Atkeson. "Center of mass estimator for humanoids and its application in modelling error compensation, fall detection and prevention". In: *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. IEEE. 2015, pp. 67–73 (cit. on p. 18).

[XFA16] X Xinjilefu, Siyuan Feng, and Christopher G Atkeson. "A distributed mems gyro network for joint velocity estimation". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 1879–1884 (cit. on p. 16).

[Xin+14] X Xinjilefu, Siyuan Feng, Weiwei Huang, and Christopher G Atkeson. "Decoupled state estimation for humanoids using full-body dynamics". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, pp. 195–201 (cit. on pp. 11, 16).

[Yan+20a] Nan Yang, Lukas von Stumberg, Rui Wang, and Daniel Cremers. "D3vo: Deep depth, deep pose and deep uncertainty for monocular visual odometry". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 1281–1292 (cit. on p. 54).

[Yan+19] Shuo Yang, Hans Kumar, Zhaoyuan Gu, Xiangyuan Zhang, Matthew Travers, and Howie Choset. "State estimation for legged robots using contact-centric leg odometry". In: *arXiv preprint arXiv:1911.05176* (2019) (cit. on p. 9).

[Yan+20b] Yulin Yang, Benzun Pious Wisely Babu, Chuchu Chen, Guoquan Huang, and Liu Ren. "Analytic combined imu integration (aci 2) for visual inertial navigation". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 4680–4686 (cit. on p. 78).

[ZCF21] Lintong Zhang, Marco Camurri, and Maurice Fallon. "Multi-camera LiDAR inertial extension to the newer college dataset". In: *arXiv preprint arXiv:2112.08854* (2021) (cit. on p. 119).

[ZS18] Zichao Zhang and Davide Scaramuzza. "A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 7244–7251 (cit. on p. 96).

[Zho+20] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. *On the Continuity of Rotation Representations in Neural Networks*. 2020. arXiv: 1812.07035 [cs.LG] (cit. on p. 53).