12

Leg Odometry for SLAM

Marco Camurri and Matías Mattamala

Legged robots are becoming widespread thanks to their ability to traverse highly unstructured terrains. Their main advantage is that legs provide an active suspension that decouples the motion of the robot's body from the terrain profile [993]. This enables them to negotiate staircases, uneven terrain, and other ground obstacles that are challenging for wheeled platforms [226, 508]. Even though the SLAM algorithms reviewed in the other chapters are directly applicable to legged robots, the additional sensing introduced by the legs is a valuable new source of information that can be exploited for odometry. This is particularly important for legged locomotion control and planning, where high-frequency and low drift real-time pose and velocity estimation is required to prevent falls and failures, which is challenging.

In this chapter, we introduce the main fundaments to estimate the real-time pose and velocity of a legged robot equipped with an onboard IMU and joint sensing (position and torque). We will particularly focus on *leg odometry*, which aims to determine the relative motion of the robot's body from leg sensing. We will introduce the theory to estimate leg odometry in Section 12.2 and Section 12.3, and how to combine it with different sensor modalities within factor graphs in Section 12.4. We will conclude with a review of the open problems in the field (Section 12.6), as well as new trends that have arisen to address them (Section 12.5).

12.1 Introduction

While most of modern legged platforms carry sensors we have studied in other chapters, namely cameras (Chapter 7), LiDAR (Chapter 8), and inertial measurement units (Chapter 11), in legged platforms we can additionally exploit kinematic and dynamic information from the robot's legs to obtain measurements of the relative motion of the robot's body, a technique known as *leg odometry* . The term was introduced in analogy with the odometry of wheeled vehicles, which infer the distance travelled by measuring how much the wheels turn over time.

In contrast to wheeled vehicles, legged robots move by making and breaking contacts between their legs and the ground. Each leg stride is defined by a phase where the leg is temporarily lifted off the ground (*aerial* or *swing phase*), and then

it stays in non-slipping contact with the ground (*stance phase*). Because the legs in the stance phase are the ones responsible for propelling the robot, the leg odometry problem can be decomposed into two sub-problems:

1 *Contact estimation*—establishing what legs are in stance phase at a given period of time.
2 *Motion estimation*—determining the incremental motion from such legs during the stance phase.

In the next sections we will provide the technical details to implement leg odometry. In Section 12.2 we will explain how the relative motion can be obtained from the joint sensing of the legs, assuming that the stance legs are known. This practically allows us to consider leg odometry as a *measurement* in our SLAM or general estimation problems, in a similar way to wheel odometry or inertial preintegration. Then, Section 12.3 will describe how contact is estimated, and the main techniques adopted in practice.

12.1.1 Historical Background

Leg odometry has been directly related to the development of machines able to walk. The origins of legged robotics go back to 1950s and 1960s, with the first attempts to create transportation systems able to overcome the limitations of wheeled vehicles on rough terrain [681]. General Electric's *Walking Truck* [782], a 1400 kg machine, is one of the best examples of the human-operated machines designed in this period. With the development of new control strategies in the 1960s, the efforts shifted to smaller scale platforms that could generate automatic walking behavior [757, 758, 1104]. Raibert's monopods, bipeds, and quadrupeds developed in the late 1980s showed remarkable locomotion capabilities [909], motivating the use of legged platforms for real-world tasks. Achieving real-world autonomy has then been the main driver to develop leg odometry and state estimation systems.

The work by Roston and Krotkov presented one of the earliest uses of leg kinematics for the estimation of the motion of a legged platform—the Ambler hexapod, a 2.5 tonnes robot designed for space exploration [952]. Similar approaches were developed for other hexapods with different leg morphologies, such as RHex [675]. Later works combined leg odometry with other sensor modalities such as IMU and vision, via particle filters [216] or Kalman filters [210, 932]. The milestone work by Bloesch *et al.* [94] established the foundations for filtering-based, all terrain, kinematic-inertial odometry estimators for quadrupeds [95], and extensions to bipedal platforms [953, 881].

The DARPA Robotics Challenge (DRC), developed between 2012 and 2015, motivated the development of *whole-body* state estimation systems for humanoid robots, aiming not only to estimate a 6 DoF but also the full state of the robot's body. For this, various teams combined kinematic and dynamic information [1200] with

inertial and joint sensing, as well as exteroceptive modalities such as LiDAR [323] and stereo vision [324]. In this period most of the solutions relied on Kalman filtering but a few works also explored optimization-based solutions [1201] and factor graphs [344].

With the recent rise and commercialization of legged platforms, particularly quadrupeds, there has been a growing interest in developing more principled and resilient leg odometry and state estimation algorithms. This has been reflected in the further development of factor graph-based estimation solutions for quadrupedal [1188] and bipedal platforms [446], which have enabled its principled fusion with other sensor modalities. Other directions have been explored more fundamental challenges in modelling [26], as well as invariant estimation frameworks [448, 445]. The recent DARPA Subterranean (SubT) Challenge (2018-2021) also posed new challenges for legged state estimation in extreme scenarios, where *complementary* estimation solutions leveraged different sensor modalities across diverse legged, wheeled, and aerial platforms [566, 1282, 844]. Section 12.6 will provide further insights on the recent trends, and how they are re-shaping the research in state estimation for legged platforms.

12.1.2 Reference Frames

In Figure 12.1, we illustrate the reference frames relevant to our estimation problem. The inertial frame W and the base frame $\mathcal{F}^b$ are rigidly attached to the ground and the robot's floating base, respectively. Without loss of generality, we assume the IMU to be coincident to $\mathcal{F}^b$. A frame is attached to each end effector, corresponding to the feet ($\mathcal{F}^{f1}$ and $\mathcal{F}^{f2}$ in the example shown in Figure 12.1).

Additionally, one or more temporary inertial frames $\mathcal{F}^k$ are created when a foot comes into contact with the ground. These are coincident with the foot frame at touchdown for humanoids, or have the same Cartesian position for quadrupeds with point feet.

12.1.3 State Definition

The state of a legged robot is defined by the pose and velocity of its base, as well as the joint states. However, in this chapter we assume the joint states to be measured directly by dedicated sensors (Section 12.1.6), leaving only the pose and velocity of the robot's base as the objective of our estimation. Therefore, we use the term *state estimation* and odometry interchangeably, with the latter term being equivalent to SLAM without loop closures [142].

More formally, the robot state is defined as the set combining position, orientation, linear velocity, and angular velocity:

$$\boldsymbol{x}_k = \begin{bmatrix} \boldsymbol{t}_k & \boldsymbol{R}_k & \mathbf{v}_k & \boldsymbol{\omega}_k \end{bmatrix}^\mathsf{T} \tag{12.1}$$

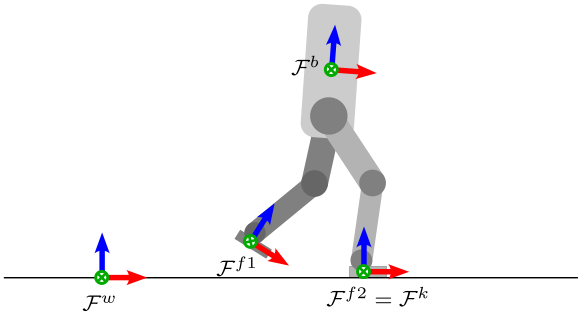


Figure 12.1 Reference frame conventions for legged robots. The world frame W is fixed to earth, while the base frame $\mathcal{F}^b$ is attached to the main chassis. Without loss of generality, the IMU frame B is not shown as it can be considered coincident with $\mathcal{F}^b$. When a foot touches the ground, a contact frame $\mathcal{F}^k$ is defined. $\mathcal{F}^k$ is rigidly attached to earth, perpendicular to the ground, and coincident with the foot frame $\mathcal{F}^f$.

where the following conventions are adopted: the robot position $\boldsymbol{t} = \boldsymbol{t}_b^{\mathrm{W}} \in \mathbb{R}^3$ and orientation $\boldsymbol{R} = \boldsymbol{R}_b^{\mathrm{W}} \in \mathrm{SO}(3)$ express the pose of the robot's base in world coordinates; the robot velocities $\mathbf{v} = \mathbf{v}_b^b$, $\boldsymbol{\omega} = \boldsymbol{\omega}_b^b \in \mathbb{R}^3$ express the base's twist, in base coordinates.

12.1.4 Legged Robot Kinematics

A legged robot is kinematically described by a main link for the body (also referred as trunk, or torso) to which one or more kinematic chains (*i.e.,* the legs) are attached. In this chapter, we consider the most common kinematic configurations adopted in practice: bipeds with 6 actuated DoF per leg and flat feet, and quadrupeds with 3 DoF per each leg and point feet (Figure 12.2). We assume that the robot has a rigid body (*e.g.,* with no articulated spine), and ignore any other upper limbs (*e.g.,* arms).

For leg odometry, we are interested in modeling the relative pose (or position) of the flat (or point) feet with respect to the robot's body. Let $\boldsymbol{q} \in \mathbb{R}^N = [q_1, \ldots, q_N]^\mathsf{T}$ be the set of joint positions of an articulated robot with N active DoF, which correspond to the angular position of revolute joints of the legs. In Figure 12.1 and for the rest of the chapter, the active DoFs of the quadruped and biped is $N = 12$, although this can be different in other platforms. The joint positions $\boldsymbol{q}$ are typically measured directly via rotary encoders placed on each joint (see Section 12.1.6.1), or indirectly using the kinematic model of the robot in addition to the readings from the encoders placed on a transmission between the motor and the joint (*e.g.,* by measuring the displacement of a hydraulic piston via a linear encoder, and calculating the corresponding angle of the revolute joint moved by the piston).

The time derivatives of the joint positions are the joint velocities $\dot{\boldsymbol{q}} = [\dot{q}_1, \ldots, \dot{q}_N]^\mathsf{T}$,

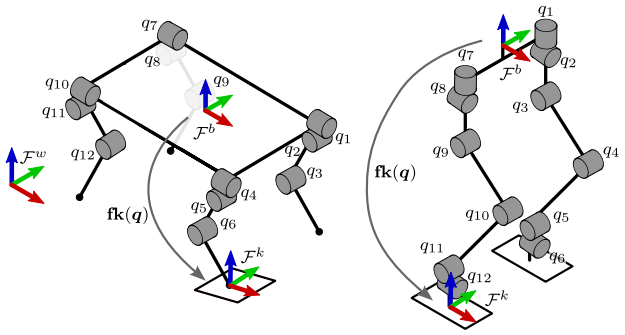


Figure 12.2 Kinematic chains of typical legged robots: quadrupeds and humanoids.

which are typically estimated by numerical differentiation of the encoder readings. In some cases, when the reading are particularly noisy, additional sensing such as IMUs placed at the links can also be used to estimate the joint velocities [1202].

Given $\boldsymbol{q}$ and a specific foot f of a humanoid robot, the *Forward kinematics* function $\mathbf{fk}(\boldsymbol{q}) : \mathbb{R}^{12} \rightarrow \mathrm{SE}(3)$ [725] maps the joint positions to the pose of the foot respect to the robot's base:

$$\boldsymbol{T}_f^b = \mathbf{fk}(\boldsymbol{q}) = \begin{bmatrix} \mathbf{f}_R(\boldsymbol{q}) & \mathbf{f}_p(\boldsymbol{q}) \\ \mathbf{0} & 1 \end{bmatrix} \tag{12.2}$$

where we defined two convenient functions $\mathbf{f}_R : \mathbb{R}^{12} \rightarrow \mathrm{SO}(3)$ and $\mathbf{f}_p : \mathbb{R}^{12} \rightarrow \mathbb{R}^3$ expressing the orientation and Cartesian position of the foot in the base frame, respectively. For quadruped robots with point feet, only $\mathbf{f}_p(\boldsymbol{q})$ is used for leg odometry, since the foot can pivot on the contact point with no change in the joint positions.

The time derivative of the forward kinematics function from (12.2) is the *Jacobian matrix* $\mathbf{J}(\boldsymbol{q}) : \mathbb{R}^{12} \rightarrow \mathbb{R}^{6\times 12}$ which can be used to compute the velocity of the end effector respect to the robot's base as follows [726]:

$$\begin{bmatrix} \mathbf{v}_f^b \\ \boldsymbol{\omega}_f^b \end{bmatrix} = \mathbf{J}(\boldsymbol{q})\dot{\boldsymbol{q}} = \begin{bmatrix} \mathbf{J}_v(\boldsymbol{q}) \\ \mathbf{J}_\omega(\boldsymbol{q}) \end{bmatrix} \dot{\boldsymbol{q}} \tag{12.3}$$

where $\mathbf{J}_v(\boldsymbol{q})$ and $\mathbf{J}_\omega(\boldsymbol{q})$ are the linear and angular part of the Jacobian respectively (note that some references [327][448] define the angular block first). Since each leg is kinematically independent from the other, $\mathbf{J}(\boldsymbol{q})$ is as a sparse block matrix, where the only two non-zero blocks $\mathbf{J}_{f,v}(\boldsymbol{q})$, $\mathbf{J}_{f,\omega}(\boldsymbol{q})$ map the subset of joint angle velocities of a leg to the linear and angular velocity of the corresponding foot f. For example, the Jacobian of the second leg of a humanoid $\mathbf{J}_2(\boldsymbol{q})$ is represented as:

$$\mathbf{J}_2(\boldsymbol{q}) = \begin{bmatrix} \mathbf{0}_6 & \bar{\mathbf{J}}_2(\boldsymbol{q}) \end{bmatrix} \tag{12.4}$$

where $\bar{\mathbf{J}}_2(\boldsymbol{q}) \in \mathbb{R}^{6\times 6}$ indicates the non-zero block of the Jacobian matrix.

The expressions from (12.2) to (12.3) are the basis to design state estimators for

legged platforms, as they relate the joint states to the robot's base motion. However, as mentioned previously, we assume we can measure the joint states directly via encoders, as explained in Section 12.1.6.

12.1.5 Legged Robot Dynamics

The dynamics of a floating-base articulated-body system can be expressed as two coupled dynamics equations, computed using Recursive Newton-Euler algorithms [327]. The first equation describes the dynamics of the floating-base body (6 DoF, underactuated), while the second describes the dynamics of the N rigid-bodies (*i.e.,* $N = 12$) attached to it through active joints (*i.e.,* active DoF). The two equations of motion can be put in matrix form as follows:

$$\boldsymbol{M}(\boldsymbol{q}) \begin{bmatrix} \dot{\mathbf{v}}_b^b \\ \dot{\boldsymbol{\omega}}_b^b \\ \ddot{\boldsymbol{q}} \end{bmatrix} + \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \begin{bmatrix} \mathbf{J}_b^\mathsf{T} \\ \mathbf{J}_q^\mathsf{T} \end{bmatrix} \boldsymbol{f} + \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} \tag{12.5}$$

where the first term $\boldsymbol{M}(\boldsymbol{q})$ is the mass matrix, which is multiplied by the stack of $\dot{\mathbf{v}} \in \mathbb{R}^3$, $\dot{\boldsymbol{\omega}} \in \mathbb{R}^3$, and $\ddot{\boldsymbol{q}} \in \mathbb{R}^{12}$ which represent the floating-base linear, floating-base angular, and active joints accelerations, respectively. The second term $\boldsymbol{h} \in \mathbb{R}^{18}$ is a bias term that accounts for Coriolis, centrifugal, and gravitational effects. Regarding the right side of (12.5), the last term describes the torques of the base, which are zero because the base is not actuated, and the torques of the active joints $\boldsymbol{\tau} \in \mathbb{R}^{12}$.

Finally, the second to last term is the most relevant for leg odometry and its factors have variable dimensions depending on the robot configuration (humanoid or quadruped) and the number of legs in contact. Let c be the number of legs in contact and d be the number of active joints per a single leg ($d = 3$ for quadrupeds, $d = 6$ for humanoids). Then, $\mathbf{J}_b \in \mathbb{R}^{dc \times 6}$ is the Jacobian matrix mapping the base twist to feet velocities, which depends on the forward kinematics of the robot and its absolute body orientation in the inertial frame W. $\mathbf{J}_q \in \mathbb{R}^{dc \times 12}$ is the stack of Jacobians described in (12.3) whose arrangement depends on the type of robot and number of contact legs. For example, a humanoid standing on both legs will have:

$$\mathbf{J}_q = \begin{bmatrix} \mathbf{J}_1(\boldsymbol{q}) \\ \mathbf{J}_2(\boldsymbol{q}) \end{bmatrix} \tag{12.6}$$

For quadrupeds, since the leg can pivot around the contact point, only the linear velocity Jacobian $\mathbf{J}_v$ is used for $\mathbf{J}_q$. For example, a quadruped standing on the first, third, and fourth leg, will have:

$$\mathbf{J}_q = \begin{bmatrix} \mathbf{J}_{1,v}(\boldsymbol{q}) \\ \mathbf{J}_{3,v}(\boldsymbol{q}) \\ \mathbf{J}_{4,v}(\boldsymbol{q}) \end{bmatrix} \tag{12.7}$$

The last term to consider is $\boldsymbol{f} \in \mathbb{R}^{dc}$ represents the collection of all the forces

and/or torques acting at each foot in contact with the ground. For a quadruped with all feet on the ground, $\boldsymbol{f}$ is the stack of four linear forces:

$$\boldsymbol{f} = \begin{bmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \\ \boldsymbol{f}_3 \\ \boldsymbol{f}_4 \end{bmatrix} \tag{12.8}$$

For a humanoid with both feet on the ground, it contains the linear forces and torques acting on the three axes of the contact point:

$$\boldsymbol{f} = \begin{bmatrix} \boldsymbol{f}_1 \\ \boldsymbol{\tau}_1 \\ \boldsymbol{f}_2 \\ \boldsymbol{\tau}_2 \end{bmatrix} \tag{12.9}$$

As explained in more detail in Section 12.3.4, we can infer leg contact using the aforementioned forces and torques.

12.1.6 Joint Sensing

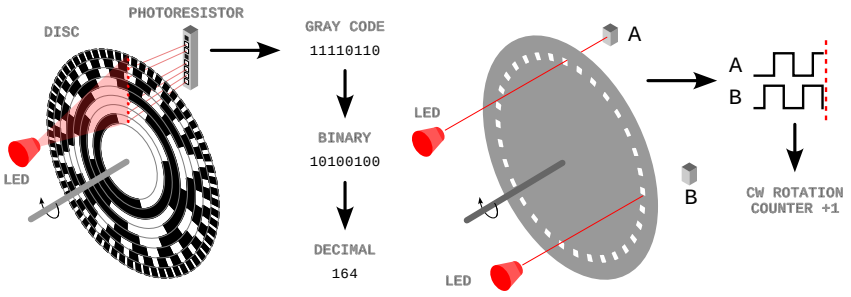
Similarly to fixed-base manipulators, the joints and the end effectors of legged robots are equipped with a variety of sensors, which are primarly used for planning and control [1018]. We briefly describe the most important sensors that are used also for leg odometry, namely encoders, force/torque sensors, and contact sensors.

12.1.6.1 Rotary Encoders

Rotary encoders are electromechanical devices that convert an angular position of a rotating shaft into an analog or digital signal. In legged robots, they enable us to measure the joint angles and determine the robot kinematics, but they can also be found in other components. For example, mechanical LiDAR use them to measure the azimuthal angle of the beam array (see Chapter 8).

Encoders can be categorized depending on the principle of operation (optical or magnetic), the type of reading (absolute or incremental), and type of output (analog or digital). The most adopted type on legged robots are absolute and relative optical digital encoders; other encoders are discussed in more detail in related literature [727].

Absolute optical digital encoders (Figure 12.3a) measure the joint angles in an absolute manner—for the same joint configuration they will provide the same sensor readings. Their operation principle is that an IR light source (*e.g.,* a Light Emitting Diode (LED)) hits an array of sensitive elements (*e.g.,* photoresistors) disposed radially on the static part of the device. In between the light source and the sensitive elements sits a disc that rotates with the shaft. The disc is divided in concentric sectors that can be either opaque or transparent. The sectors are arranged according



(a) *Optical Absolute Encoder* (b) *Optical Incremental Encoder*

Figure 12.3 Left: Principle of operation of an 8-bit optical absolute encoder. An IR light beam hits a rotating disc that masks light according to a specific pattern that encodes the angle of rotation of the disc. An array of photoresistors convert the absence or presence of the light in each sector to a binary number encoded with a Gray code. The Gray code is then translated into a decimal number representing the absolute angle of rotation. The encoder in this example has a resolution of $360/256 = 1.41$ degrees. Right: Principle of operation of an optical incremental encoder. The two photoresistors, A and B, are placed with a 90 degrees phase shift. A rising edge on A followed by a falling edge on B indicates a clockwise rotation. The change from $AB = 11$ to $AB = 10$ causes the increment of the counter.

to a pattern that encodes a specific angular range to a binary number. The binary number is ordered according to the Gray code, which maps consecutive natural numbers to binary numbers that always differ by only one bit, which reduces chances of reading errors. The process is illustrated in Figure 12.3a for an 8-bit encoder. The angular resolution of the device is determined by the number of bits (*i.e.,* the number of concentric sectors) used to make the binary word encoding the angle. For instance, for an 8-bit sensor there are 256 possible values, and then the angular resolution is $360/256 = 1.41$ degrees.

Incremental optical encoders (Figure 12.3b), conversely, measure relative angular changes with respect to the *initial configuration*—they measure a zero angle when they are turned on, and then measure the angle relative to that reference point. The angle is calculated by adding or subtracting small angle increments, depending on the direction of rotation. Instead of relying on Gray codes, they operate by using a simpler codewheel made of an opaque material with regular slots placed radially, such that a single photoresistor A produces a square wave over time when the disc rotates at constant speed. A second photoresistor B is placed at 90 degrees out of phase with the first one. The 2-bit word composing the two signals AB can have four different values at any given time, and the transition between them is used to determine the direction of rotation and whether the count has to be increased or decreased [727]. Because of their simpler construction and lower cost, high resolution incremental encoder have been used to compute the joint angle after a lower resolution absolute encoder measured the initial angle [992]. Even though

they are still in use, incremental encoders are being rapidly replaced by absolute encoders, whose technology development improves their resolution while reducing their cost.

12.1.6.2 Force and Torque Sensors

Force and torque sensors are devices that convert a linear force (applied to a point on a surface) or a mechanical torque (applied to a shaft) into an electrical signal. They are primarily used for torque control on the actuators or to sense the interaction between the end effector and the environment. In legged locomotion, each step involves forces being applied to the ground that need to be measured (directly or indirectly) so that stance legs can be identified.

The principle of operation for both type of quantities (force and torque) is typically the same, with different geometries: the internal surfaces of the sensors are shaped in a way that would slighlty deform under stress along the direction where the force needs to be measured. Glued to those surfaces is a flexible variable resistive element, the *strain gauge*. The electrical resistance of the strain gauge changes proportionally to the amount of deformation it sustains, with compression (extension) causing a reduction (increase) in resistivity. Figure 12.4 shows an example with strain gauges applied to a load cell to measure linear force [727]. To measure torque, a series of strain gauges is applied to flexible spokes of a wheel connected to a motor shaft.

To measure all forces and torques acting on an end effector, 6-axis sensors are available, containing strain gauges in a number of configurations sufficient to measure forces and torques in all directions. These are commonly used for manipulation tasks, but can be also found on humanoids feet to directly measure the interaction with the ground.

With the advent of cost effective dynamic legged robots, wich was made possible by a backdriveable motor design [552], torques can be estimated from the motor currents, which are proportional to the torque by a constant factor.

12.1.6.3 Contact Sensors

Since the main use for force and torque sensors in leg odometry is to determine the stance legs, alternative cost effective solutions are contact sensors, whose output is a binary number indicating whether a certain foot is in contact or not. This type of sensor was mainly developed for small to medium quadruped robots, whose feet consist of a spherical or circular rubber sole.

The main types of contact sensors are optical [406] or mechanical [800]. Optical contact sensors are conceptually similar to encoders: a LED-photodiode pair sense light through a small aperture. When the foot is in contact, the surface of the foot deforms enough to create a displacement of a masking panel that occludes the aperture, allowing the system to detect the contact. Mechanical contact sensors

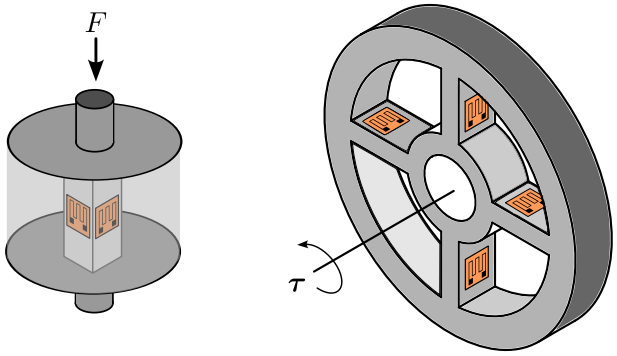


Figure 12.4 Principle of operation of force and torque sensors. A strain gauge is applied on compressible or flexible elements when under load. Inside the loadcell (on the left), a strain gauge is applied such that a compression would be detected as a reduction in resistivity. Inside the torque sensor (on the right), several strain gauges are applied at the flexible elements (the spokes of a wheel). When torque is applied, the elements would deform by flexion. The presence of multiple strain gauges allow to work out the magnitude and direction of the torque *e.g.,* by sensing a compression on one side of the spoke and an elongation on the other side.

use a simple pushbutton switch hidden inside the sole that is pressed when enough force is exerted on the foot.

The main disadvantage of contact sensors is the relatively slow response time compared to costly force/torque sensors. In addition, they suffer from the same drawbacks of F/T sensors: they require to route cables up to the foot and they are at risk of damage due to the main impacts they have to substain. For these reasons, they are mostly available only for small sized quadrupeds mostly designed for indoor operations.

12.2 Motion Estimation

Given the joint states and kinematics of the robot, we are now interested in computing the incremental motion of the robot's base. This can be mainly done in two ways: using the forward kinematics to obtain a relative pose between two time instants, or using the differential kinematics to estimate the robot's velocity instantaneously. In both cases, the underlying assumption is that a newly formed contact frame remains stationary for a certain amount of time.

12.2.1 Relative Pose Estimation

Figure 12.5 shows a simplified example of a humanoid robot walking along the zx-plane. The robot's base is represented at two consecutive time instants with the

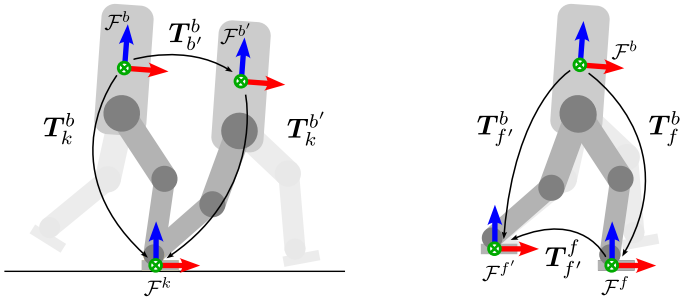


Figure 12.5 How leg odometry works with ideal contact. Left: Assuming the contact frame $\mathcal{F}^k$ is rigidly attached to the ground (in yellow), we can determine the relative motion of the robot's body. Right: Alternatively, we can represent how the leg moves with respect to the body frame (yellow) in two consecutive instants.

frames $\mathcal{F}^b$ and $\mathcal{F}^{b'}$; the foot frames and the joint positions at the same two times are defined similarly. Because the contact frame is stationary, when the foot frame and the contact frame coincide, the amount of displacement the robot's body experiences while moving forward is the same as the foot experiences moving backwards from the robot's body:

$$\boldsymbol{T}^b_{b'} = \boldsymbol{T}^b_k (\boldsymbol{T}^{b'}_k)^{-1} = (\boldsymbol{T}^f_{f'})^{-1} = (\boldsymbol{T}^b_{f'})^{-1} \boldsymbol{T}^b_f = \mathbf{fk}(\boldsymbol{q}')^{-1} \mathbf{fk}(\boldsymbol{q}) \tag{12.10}$$

(12.10) creates a mapping between the joint states and the relative pose of the robot. While a concatenation of these relative poses would effectively provide a valid motion estimate by *dead reckoning* from joint sensing only [952], this is only possible during the stance phase. Further, to be applied on robots with point feet, this requires at least three feet in contact with the ground at all times, making it impractical for quadrupedal platforms.

To overcome these issues, the standard approach for quadrupeds [94] is to augment the state in (12.1) with the positions $\boldsymbol{c}_i = \boldsymbol{t}^{\mathrm{W}}_k \in \mathbb{R}^3$ of the contact frames expressed in world coordinates and associated to each leg of the robot. For humanoids, since they have ankles, the orientation of the contact points $\boldsymbol{B}_i = \boldsymbol{R}^{\mathrm{W}}_k \in \mathrm{SO}(3)$ can also be added to the state [953].

Further, since there is no guarantee that at any given time there are a sufficient number of legs in contact (*e.g.,* a *gallop* gait has phases were all the legs are off the ground), it is also commonly assumed that an IMU is present, so the angular velocity $\boldsymbol{\omega}$ in (12.1) is disregarded, and the IMU biases are included as part of the state instead (see Chapter IMU).

With all the previous considerations, the corresponding states of interest for quadrupeds and bipeds are then defined as:

$$\boldsymbol{x}_k = \begin{bmatrix} \boldsymbol{t}_k & \boldsymbol{R}_k & \mathbf{v}_k & \mathbf{b}_k^a & \mathbf{b}_k^\omega & \boldsymbol{c}_1 & \boldsymbol{c}_2 & \boldsymbol{c}_3 & \boldsymbol{c}_4 \end{bmatrix}^\mathsf{T} \tag{12.11}$$

$$\boldsymbol{x}_k = \begin{bmatrix} \boldsymbol{t}_k & \boldsymbol{R}_k & \mathbf{v}_k & \mathbf{b}_k^a & \mathbf{b}_k^\omega & \boldsymbol{c}_1 & \boldsymbol{c}_2 & \boldsymbol{B}_1 & \boldsymbol{B}_2 \end{bmatrix}^\mathsf{T} \tag{12.12}$$

where (12.11) represents the state of a quadruped robot, whilst (12.12) corresponds to a humanoid robot. This enables us to precisely express the motion estimate relationship from (12.10) for an arbitrary i-th leg:

$$\boldsymbol{T}_k^b = \mathbf{fk}(\boldsymbol{q}) = (\boldsymbol{T})^{-1}\boldsymbol{C}_i \tag{12.13}$$

where

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \mathbf{0} & 1 \end{bmatrix} \tag{12.14}$$

is the pose of the base in the fixed frame, whereas

$$\boldsymbol{C}_i = \begin{bmatrix} \boldsymbol{B}_i & \boldsymbol{c}_i \\ \mathbf{0} & 1 \end{bmatrix} \tag{12.15}$$

is the pose of the foot contact in the fixed frame. Expanding (12.13) leads to:

$$(\boldsymbol{T})^{-1}\boldsymbol{C}_i = \begin{bmatrix} \boldsymbol{R}^\mathsf{T} & -\boldsymbol{R}^\mathsf{T}\boldsymbol{t} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{B}_i & \boldsymbol{c}_i \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}^\mathsf{T}\boldsymbol{B}_i & \boldsymbol{R}^\mathsf{T}\boldsymbol{c}_i - \boldsymbol{R}^\mathsf{T}\boldsymbol{t} \\ \mathbf{0} & 1 \end{bmatrix} \tag{12.16}$$

Rearranging (12.16), we can define the following components corresponding to the upper blocks of right-hand side matrix:

$$\mathbf{f}_p(\boldsymbol{q}) = \boldsymbol{R}^\mathsf{T}(\boldsymbol{c}_i - \boldsymbol{t}) \tag{12.17}$$

$$\mathbf{f}_R(\boldsymbol{q}) = \boldsymbol{R}^\mathsf{T}\boldsymbol{B}_i \tag{12.18}$$

where $\mathbf{f}_p(\boldsymbol{q})$ denotes the relative position change of the foot in the fixed frame, and $\mathbf{f}_R(\boldsymbol{q})$ the relative orientation change, as a function of the joint angles. Please note that for quadrupeds we only use (12.17), since we cannot obtain an orientation estimate from point feet.

(12.17) and (12.18) are the basic leg odometry expressions used as measurements within estimation frameworks such as filters or factor graphs. These will be further described in Section 12.4.

12.2.2 Velocity Estimation

The differential kinematics function from (12.3) can be used get a direct velocity measurement from each stance leg. This approach is widely adopted on quadrupeds [95, 149, 586] and less frequently on bipeds [1089]. The advantages are that velocity measurements can be easily (pre)integrated into a filter (or factor); they do not retain any history to avoid position error build up [323] and they do not need to keep track of extra states (contact poses or positions). However, we must point out

that the joint velocities are usually numerically differentiated from joint positions, possibly degrading the estimation performance due to rounding errors.

Following a similar procedure as with the relative pose measurements, we aim to describe the velocity relationships that hold while a leg in rigid contact with the ground moves. First, we observe that the contact point k must be stationary for stance legs, hence the velocity of the contact point seen from the fixed frame must be zero:

$$\mathbf{v}_k^{\mathrm{W}} = \mathbf{0}. \tag{12.19}$$

Furthermore, the velocity of the contact point k must coincide with the velocity of the the foot f, also described by the Jacobian matrix (12.3):

$$\mathbf{v}_k^b = \mathbf{v}_f^b = \mathbf{J}_v(\boldsymbol{q})\dot{\boldsymbol{q}}. \tag{12.20}$$

Since the angular velocity of the robot is measured by the IMU, we focus on the linear velocity only. From (12.19) and (12.20), we can determine the robot's body velocity as:

$$\begin{aligned} \mathbf{v}_k^{\mathrm{W}} &= \mathbf{v}_b^{\mathrm{W}} + \boldsymbol{\omega}_b^{\mathrm{W}} \times \boldsymbol{t}_k^b + \mathbf{v}_k^b \\ \mathbf{0} &= \mathbf{v}_b^{\mathrm{W}} + \boldsymbol{\omega}_b^{\mathrm{W}} \times \mathbf{f}_p(\boldsymbol{q}) + \mathbf{J}_v(\boldsymbol{q})\dot{\boldsymbol{q}} \\ \mathbf{v}_b^{\mathrm{W}} &= -\boldsymbol{\omega}_b^{\mathrm{W}} \times \mathbf{f}_p(\boldsymbol{q}) - \mathbf{J}_v(\boldsymbol{q})\dot{\boldsymbol{q}} \end{aligned} \tag{12.21}$$

where we take into account the additional linear velocity caused by the lever arm between the base and the foot [278]. (12.21) maps the absolute velocity of the robot to the forward and differential kinematics functions and can therefore be used as a measurement update or factor in a graph, as we will describe in Section 12.4. Note that since we conventionally express velocities in body coordinates, these can be obtained by using the robot orientation $\boldsymbol{R} = \boldsymbol{R}_b^{\mathrm{W}}$.

Up to now we assumed that the stance legs are known. In the next section, we describe different methods to identify them.

12.3 Contact Estimation

The definition of contact estimation varies with the application. For example, in collaborative robotics, the end effector of a manipulator might be considered in contact as soon as it is "touching" something, *i.e.,* there is a non negligible external force exerted on it. For leg odometry, a foot can be considered in contact only when the contact point is stationary over time; on robots with point feet, this means ensuring it does not slip.

Technically, a foot does not slip when the vertical component of the Ground Reaction Force (GRF), f_z, is within the friction cone [954]:

$$\sqrt{f_x^2 + f_y^2} \leq \mu_{x,y} f_z \tag{12.22}$$

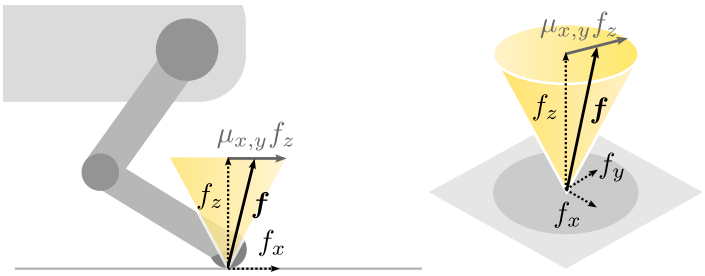


Figure 12.6 The contact point on a quadruped's leg. A leg can be considered in stance when the force $\boldsymbol{f} = [f_x, f_y, f_z]^\top$ applied by the foot stays within the friction cone.

where f_x and f_y are the tangential components of the GRF with respect to the contact plane, which depend on the local morphology of the terrain. $\mu_{x,y}$ is the friction coefficient, which depends on the mechanical properties of the ground and the foot touching it.

Humanoids with flat feet can also exert a torque on the ground. This introduces an additional condition to the non-slipping condition (12.22), which requires that the foot does not *rotate*:

$$\begin{bmatrix} -\tau_y / f_z \\ \tau_x / f_z \end{bmatrix} \leq \begin{bmatrix} CoP_x \\ CoP_y \end{bmatrix} \tag{12.23}$$

$$|\tau_z| \leq \mu_z f_z \tag{12.24}$$

where τ is the contact torque, μ_z is the rotational coefficient of friction, and CoP_x, CoP_y denote upper limits of the components of the *center of pressure*, which define the contact support polygon bounds that are functions of contact surface geometry.

Since a sufficiently-high normal force f_z would guarantee that inequalities ((12.22)-(12.24)) are satisfied regardless of the other contact wrench (force and torque) dimensions, the most adopted approach for contact estimation is to simply threshold f_z. Then, the only differences from an implementation point of view are how the force is measured/estimated (*i.e.,* contact sensors, F/T sensors, joint sensing, IMUs), and the specific characteristics of the robot.

12.3.1 With Contact Sensors

Contact sensors implicitly threshold f_z in hardware, as they are tuned such that the binary signale they provided is only activated when the measured force exceeds the nominal f_z. This is the simplest case, as the leg odometry can directly rely on the binary state provided by these sensors.

12.3.2 With Force/Torque Sensors

When F/T sensors are present on the foot, f_z can be measured directly over time. This permits to associate specific force patterns to events that are not just binary. For example, a small but rising force that lasts for more than a certain time is an indication that the foot is striking the ground but not yet in stance. Conversely, a force that falls below a certain value (*e.g.,* half of the expected load for one leg) means the the foot is about to break the contact and it is therefore not reliable. In both cases, the information coming from that leg need to be discarded or its associated uncertainty increased [323].

12.3.3 With IMUs

As seen in Chapter 11, IMUs are inexpensive sensors that provide acceleration and rotational velocity measurements. While we tipically use them to measure these quantities with respect to the robot's body, we can also use them on the legs or feet. Since any force applied to the foot (*e.g.,* during a touch down) would cause a change to its acceleration, some works used them to implicitly detect the stance legs [1225, 954, 736]. The main advantage of this approach is that the sensor is not sustaining an impact directly, so it is less likely to break, at the cost of additional signal processing to effectively detect such acceleration changes.

12.3.4 From Joint Torque Sensing

While it might seem straightforward to add additional sensors at the feet to detect contact, the different robot morphologies, design, and integration challenges might not make it always possible. In this case, the GRF can be estimated from the joint torques by exploiting the robots's dynamics (see (12.1.5)).

Using a quadruped platform as an example, we can exploit the block-wise structure of $\mathbf{J}_q$ to compute the force at the end effector from (12.5) as:

$$\boldsymbol{f}_i = -(\bar{\mathbf{J}}_{i,v}^{\mathsf{T}})^{-1}\big(\boldsymbol{\tau}_i - \boldsymbol{h}_{q,i} - \boldsymbol{F}^{\mathsf{T}}\dot{\mathbf{v}}\big) \tag{12.25}$$

where: $\boldsymbol{f}_i \in \mathbb{R}^3$ and $\boldsymbol{\tau}_i \in \mathbb{R}^3$ are the GRF and the torque leg i; $\bar{\mathbf{J}}_{i,v}$ is the non-zero block i-th foot Jacobian $\mathbf{J}_v$ (which for quadrupeds is a square matrix); $\boldsymbol{F} \in \mathbb{R}^{3\times3}$ is one of the blocks of the mass matrix; $\boldsymbol{h}_{i,q} \in \mathbb{R}^3$ is the vector of centrifugal/Coriolis/gravity torques for leg i.

Note that the estimate for $\boldsymbol{f}_i$ can only be in base coordinates. However, to recover the actual GRF there are two pieces of information missing:

- the local inclination of the terrain, which the orientation of the contact force depends on. While the ankle joints of a humanoid can give a good approximation, for quadrupeds the orientation of the contact frame cannot be determined without

exteroceptive sensing, but can be inferred heuristically from the other feet in contact (*e.g.,* by fitting a plane through them) [337]
- the friction coefficient, which the horizontal components of the force depend on. The friction coefficient depends on the material the robot is stepping on, so it can only be known a priori or inferred from the amount of slippage the robot is experiencing [521].

In general, establishing the contact states from joint sensing remains an open problem, and several techniques have been developed to detect contact in a probabilistic fashion, (*e.g.,* by combining also kinematics as well as dynamics of the robot [505]) or by using learning methods (see Section 12.6.1).

12.4 Leg Odometry for Estimation Problems

Now that we have specified the main steps required to obtain leg odometry measurements, in this section we describe how to integrate them into an estimation framework to solve the state estimation problem. The predominant estimation solutions are based on filtering approaches, which combine IMU and leg odometry at the high frequency required for closed-loop control. The factor graph-based smoothing approaches described in the previous chapters have only been adopted on legged platforms in the last few years. However, their focus has not been on providing estimates for control but rather lower frequency estimates for mapping, which benefit from *slower* sensors such as LiDARs or cameras.

Regardless of the method, for optimally fusing leg odometry with other sensor modalities we need to quantify its associated uncertainties. This is the first topic we will cover before presenting the filtering and smoothing approaches.

12.4.1 Encoder Noise Propagation

The main source of uncertainties in leg odometry are the robot's joints encoders, which measure the joint positions and are affected by noise. This noise can be modeled as an additive zero-mean Gaussian term $\boldsymbol{\eta}_q \in \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_q)$, such that the true value $\boldsymbol{q}$ and the measured value $\tilde{\boldsymbol{q}}$ are related as follows:

$$\tilde{\boldsymbol{q}} = \boldsymbol{q} + \boldsymbol{\eta}_q \tag{12.26}$$

Since the forward and differential kinematics functions involve rotations, they are therefore nonlinear and they will not preserve the Gaussian properties of the encoder noise. However, as done in previous chapters, we can consider a first-order approximation—which is locally linear and preserves Gaussianity—using the the Jacobian function [446]:

$$\mathbf{f}_p(\boldsymbol{q} + \boldsymbol{\eta}_q) \approx \mathbf{f}_p(\boldsymbol{q}) + \mathbf{J}_c(\boldsymbol{q})\boldsymbol{\eta}_q \tag{12.27}$$

where $\mathbf{J}_c(\boldsymbol{q})$ is the *body manipulator Jacobian*, *i.e.,* the same as the manipulator Jacobian $\mathbf{J}(\boldsymbol{q})$ but expressed in the contact frame.

The same Gaussianity assumption can also be applied to velocity measurements affected by encoder noise $\boldsymbol{\eta}_q$ and encoder velocity noise $\boldsymbol{\eta}_{\dot{q}}$ [1188], considering that:

$$\mathbf{J}(\boldsymbol{q}+\boldsymbol{\eta}_q)(\dot{\boldsymbol{q}}+\boldsymbol{\eta}_{\dot{q}}) \approx \mathbf{J}(\boldsymbol{q})\dot{\boldsymbol{q}} + \frac{\partial}{\partial \boldsymbol{q}}\left(\mathbf{J}(\boldsymbol{q})\dot{\boldsymbol{q}}\right)\boldsymbol{\eta}_q + \mathbf{J}(\boldsymbol{q})\boldsymbol{\eta}_{\dot{q}} \tag{12.28}$$

From (12.27) and (12.28), the extra terms multiplied by the noise terms can simply be grouped into a single term, since they are all linear combinations of a Gaussian term for a given encoder measurement.

12.4.2 Factor Graph Smoothing

To generate locomotion behaviors while avoiding falls and other catastrophic failures, legged robots have strict real-time control and high-frequency state estimation requirements. Historically, those requirements were met by using nonlinear variants of the Kalman Filter, such as the Extended Kalman Filter (EKF) [94, 149], the Unscented Kalman Filter (UKF) [95], or the Invariant-EKF [448, 1250, 676]. These types of filter would typically fuse together high-frequency sensor data, such as inertial and kinematics, to feed the controller. Exteroceptive sensor updates within the control loop have also been demonstrated [150] but those are normally relegated to mapping and planning purposes.

One limitation of Kalman filtering-based methods is that they are designed to have a process model in addition to the measurement model. When such a model is not available, it is usually replaced by a constant velocity model or, more often, IMU propagation. This suggests that factor graph-based methods are a more general approach, as they consider both process models and measurement models in a general manner—as a relationship between states and measurements.

Factor graph-based methods for legged systems mainly differ in the number of estimated states and the time horizon. When only two consecutive states are considered, a factor graph resembles a filter; the Two-State Implicit Filter (TSIF) [98] is an instance of this case. When the window is increased, instead of estimating only the most current state as the TSIF, the factor graph has the ability to correct a history of past states within a time window. The frequency of the states considered is a design decision: adding more frequent states at a high frequency (*e.g.,* IMU rate) simplifies the design of the estimator but it requires to reduce the time window to keep the computational requirements bounded. Conversely, longer time horizons with a fixed number of states can be achieved by preintegrating measurements, as showed for IMU measurements in Chapter 11.

We next present two examples from the related literature that illustrate how preintegration theory and the leg odometry concepts previously introduced are

leveraged in a factor graph estimation framework. We particularly focus on the case of contact preintegration for bipeds [447], and velocity bias preintegration for quadrupeds [1188]. In both cases, the measurements from Section 12.2 will be reformulated in terms of residuals and covariances for the factors of the graph.

12.4.2.1 Contact Preintegration

Contact preintegration aims to integrate the relative motion increments from the kinematics of a humanoid robot, and add them as factors that link two humanoid states, defined as in (12.12). This idea was presented by Hartley *et al.* [447], and the proposed factor graph is shown in Figure 12.7a.

The factors are generally standard: a prior factor (in black) anchors the graph, while a preintegrated IMU factor (orange) introduces the motion prior from the IMU. Additionally, for humanoids we add a forward kinematics factor (in green) that constrains the pose of the contact frames, while the contact preintegration factor (in blue) encodes the relative motion between contact states of the two legs.

Forward Kinematics Factor The forward kinematics factor relates the pose of the contact frame at the feet to the pose of the robot, both expressed in the inertial frame, via the forward kinematics of the stance leg.

By plugging the encoder noise from (12.27) into the relative pose measurement of (12.17) and (12.18), we can define the following residual and covariance for the forward kinematics factor:

$$\mathbf{r}F = \mathrm{Log}\left(\boldsymbol{C}_i^{-1}\,\boldsymbol{T}\,\mathbf{fk}(\tilde{\boldsymbol{q}})\right) \tag{12.29}$$

$$\boldsymbol{\Sigma}_{\mathcal{F}} = \mathbf{J}_c(\tilde{\boldsymbol{q}})\,\boldsymbol{\Sigma}_q\,\mathbf{J}_c^{\mathsf{T}}(\tilde{\boldsymbol{q}}) \tag{12.30}$$

where the residual enforces that the difference between the contact frame and the robot frame are close to the forward kinematics, given the uncertainty propagated from the encoders' noise.

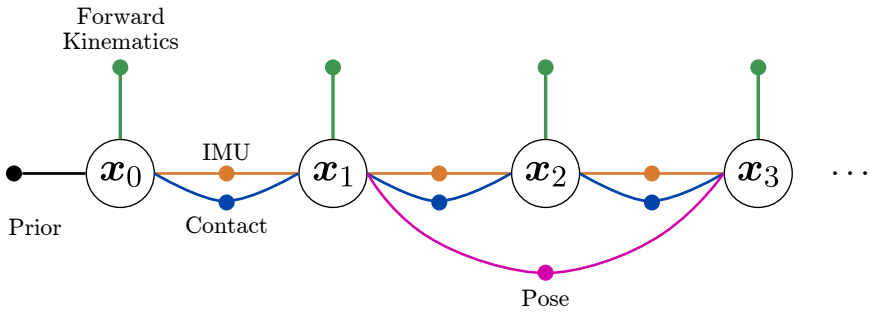
Contact Preintegration Factor The contact preintegration factor adds an additional contraint on the contact point. Ideally, if there is no slip on the stance leg and the pose of the contact frame should remain unaltered; in practice, slip occurs and can be modeled as Gaussian noise added to the velocities of the contact point. The contact preintegration factor models how the contact point can change between two time instants due to this noise.

Technically, given two consecutive states $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ at times t_i and t_j, respectively, the following relationship holds:

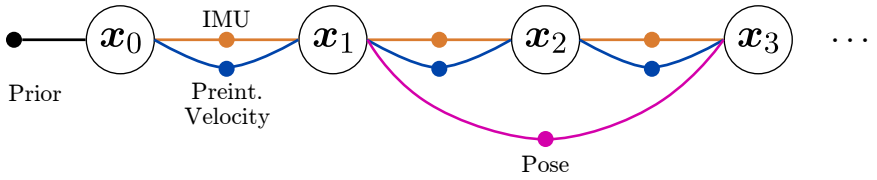
$$\Delta\tilde{\boldsymbol{B}}_{ij} = \boldsymbol{B}_i^{\mathsf{T}}\boldsymbol{B}_j\mathrm{Exp}\left(\delta\boldsymbol{\theta}_{ij}\right) = \mathbf{I} \tag{12.31}$$

$$\Delta\tilde{\boldsymbol{c}}_{ij} = \boldsymbol{B}_i^{\mathsf{T}}(\boldsymbol{c}_j - \boldsymbol{c}_i) + \delta\boldsymbol{d}_{ij} = \mathbf{0} \tag{12.32}$$

where we have used the rotational and translational components of the contact



(a) *Factor graph with preintegrated contacts*



(b) *Factor graph with preintegrated velocity*

Figure 12.7 Factor graph formulations for preintegrated contact (top) and velocity (bottom). Additional measurements (*e.g.,* from exteroceptive sensors, which constrain two states) can be easily added as additional factors (magenta).

frames $\boldsymbol{C}_i$ and $\boldsymbol{C}_j$ as stated in (12.15). The terms $\delta\boldsymbol{\theta}_{ij}$ and $\delta\boldsymbol{d}_{ij}$ are *preintegrated contact noise* terms, which are introduced to model the uncertainty on the contact point velocity as a zero-mean Gaussian variable [447]. The preintegrated contact factor is then given by the following rotation and translation residuals and covariance:

$$\mathbf{r}C = \begin{bmatrix} \mathrm{Log}\left(\boldsymbol{B}_i^{\mathsf{T}}\boldsymbol{B}_j\right) \\ \boldsymbol{B}_i^{\mathsf{T}}(\boldsymbol{c}_j - \boldsymbol{c}_i) \end{bmatrix} \tag{12.33}$$

$$\boldsymbol{\Sigma}_{\mathcal{C}} = \begin{bmatrix} \boldsymbol{\Sigma}_w & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_v \end{bmatrix} \Delta t_{ij} \tag{12.34}$$

where we have rearranged and stacked (12.31) and (12.32) into a single vector residual. The covariance $\boldsymbol{\Sigma}_{\mathcal{C}}$ is made of the time integration of the contact angular covariance $\boldsymbol{\Sigma}_w$ and linear velocity covariance $\boldsymbol{\Sigma}_v$ over Δt_{ij}.

As a last note, an important limitation of the contact preintegration factor is that it is only valid for the same stance leg, during the stance phase. Henceforth, it is not valid for the switching dynamics of a legged platform. This has been addressed in follow up work [446], by modeling and properly handling the contact frame switches, enabling preintegration among different legs.

12.4.2.2 Velocity Preintegration

As mentioned previously, the kinematics of point feet platforms—such as quadruped robots—cannot constrain the relative 6 DoF between two states. This impedes the use of the forward kinematic and contact preintegration factors recently introduced. Alternatively, we can exploit the linear velocity measurements from leg odometry, reviewed in Section 12.2.2, and preintegrate them to obtain additional factors that constraint the relative change of the robot's pose [1188, 586].

Velocity Preintegration Factor This factor assumes that the instant linear velocity of the body can be determined from leg odometry using (12.28) and it is affected by Gaussian noise terms $\boldsymbol{\eta}_v$ and $\boldsymbol{\eta}_\omega$:

$$\tilde{\mathbf{v}} = -\mathbf{J}_v(\boldsymbol{q})\dot{\boldsymbol{q}} - \boldsymbol{\omega} \times \mathbf{f}_p(\boldsymbol{q}) + \boldsymbol{\eta}_v \tag{12.35}$$

Assuming the robot has constant body linear velocity between times t_i and t_j, we can preintegrate the velocity measurements to obtain:

$$\Delta\tilde{\boldsymbol{t}}_{ij} = \Delta\boldsymbol{t}_{ij} + \delta\boldsymbol{p}_{ij} = \sum_{k=i}^{j-1} \left[\Delta\tilde{\boldsymbol{R}}_{ik}\tilde{\mathbf{v}}_k\Delta t\right] + \delta\boldsymbol{p}_{ij} \tag{12.36}$$

where, similarly to the preintegrated contact factors, $\delta\boldsymbol{p}_{ij}$ is a *preintegrated velocity noise* term [1187, 586]. Then, the preintegrated velocity factor and associated covariance are given by:

$$\mathbf{r}V = \boldsymbol{R}_i^\mathsf{T}\left(\boldsymbol{t}_j - \boldsymbol{t}_i\right) - \Delta\boldsymbol{t}_{ij} \tag{12.37}$$

$$\boldsymbol{\Sigma}_{\mathcal{V},ij} = \sum_{j-1}^{k=i} \boldsymbol{\Sigma}_{\mathcal{V},ik} + \boldsymbol{A}\boldsymbol{\Sigma}_v\boldsymbol{A}^\mathsf{T} \tag{12.38}$$

with the matrix $\boldsymbol{A} = \Delta\tilde{\boldsymbol{R}}_{ik}\Delta t$.

12.4.2.3 Handling of Multiple Measurements

In the previous sections we have considered only one measurement per leg, without considering what to do when multiple legs are in contact at the same time. The presence of multiple legs in contact potentially provides redundancy and robustness, but increases the risk of inconsistencies (*e.g.,* when different legs provide conflicting information). The simplest approach adopted by some works [323] is to pick only the leg that is deemed to be most reliable, discarding the information from the others.

Another intuitive approach is to treat each leg (and their measurements) independently. This is easier when the contact poses (or positions) are explicitly part of the state [447, 586]. When this is not the case, the velocity measurements simultaneously acquired by all legs in stance can still be treated as independent, but it is often preferrable to average them into one single measurement to reduce the computational load on the filter or factor graph [1188].

12.4.3 Integration with Exteroceptive Sensors for SLAM

As we have seen in the previous sections, leg odometry provides an additional way to compute incremental motion between two consecutive states. Their main use is to improve the odometry estimate, such that the SLAM system building on top of it (*e.g.,* a pose graph) can benefit from low drift edges between nodes, which translate into less abrupt corrections during loop closures.

The integration of additional sensors, such as LiDAR and cameras, is naturally handled by both filtering and smoothing approaches by simply adding more measurements to the former and factors to the latter. There are however subtle details to be considered while doing so. Fusing measurements from multiple independent sources, each one operating at different frequencies, levels of noise, and failure rates, is not trivial.

If a sensor modality breaks the zero-mean Gaussian noise assumption, or fails completely, the status of the filter (or factor graph) can be compromised. For this reason, also motivated by the DARPA SubT challenge, there has been a surge in loosely coupled methods that run different subsystems in parallel (*e.g.,* Visual-Inertial, Legged-Inertial, and LiDAR-Inertial) while triaging their outputs and select the best estimate from each subsystem [307, 566].

The alternative to loosely coupled methods are tightly coupled ones. In [1188] a fixed-lag smoother was used to fuse leg odometry with IMU, cameras and LiDAR in the same factor graph. In this case, to overcome the problems related to inconsistencies between the different types of factors or sensor failures, the triaging happens directly into the factors: if a sensor modality fails, the factor is simply not added to the graph. In addition, to handle noise that is not zero-mean Gaussian, robust cost functions can be used within factors.

12.5 Open Challenges

In previous sections we have covered how to generally perform leg odometry and fuse it with other sensor modalities. We made a number of assumptions that are often not valid in practice and challenging situations that are still unaddressed. We briefly introduce them here.

12.5.1 Leg Deformation

The leg odometry equations we have seen throughout the chapter all assumed that the robot was a perfectly rigid body. When this assumption is not valid, leg odometry measurements will be biased, because the forward kinematics function computes the ideal position of the end effector and not the real one (see Figure 12.8). Similarly, when the contact point does not move, but forces are applied to it such that the legs bend, the joint angles change. When the problem occurs for short peri-

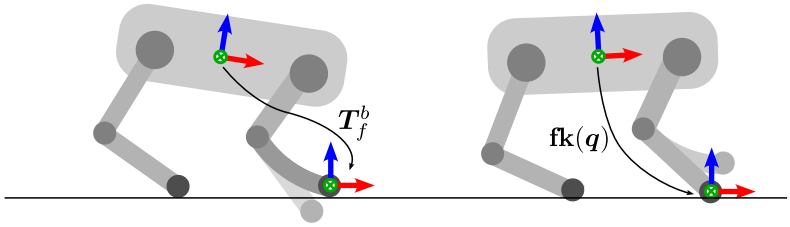


Figure 12.8 Example of leg deformation on a quadruped. The real transformation between the robot base and the contact point is shown at the left. Since the forward kinematics assumes the robot's legs are rigid, it incorrectly estimates and upward motion, as shown on the right.

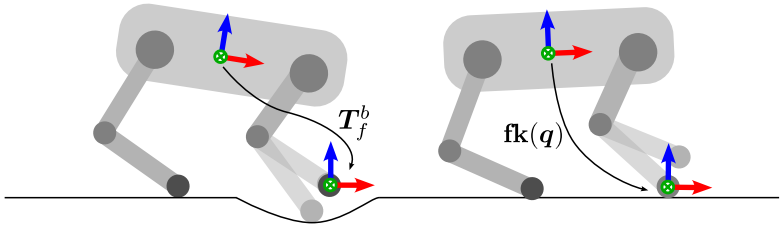


Figure 12.9 Example of ground deformation with a quadruped. The robot initially touches the ground which is flat. While keeping the contact state on, the ground deforms and the foot sinks into it (left). As the joint angles change while the foot goes down, this motion is interpreted as an upward motion from the initial touchdown point (right)

ods of time, detecting the impact by analyzing the force profile and rejecting the measurements during those periods is a strategy adopted in the past [149].

Since bipeds tend to have longer legs, the problem of leg flexibility can be even worse on such platforms. One way to approach it would be to exploit the correlation between the leg load and the flexibility (intuitively, the more a leg is loaded, the more it will flex) by carefully modelling the bending properties of the robot considering its structural geometry and properties. This approach is however complex and cannot be generalized well.

Instead, the most adopted approach is to integrate additional IMU sensors located on the links and estimate the link orientation compared to the joint readings [1132].

12.5.2 Non-rigid Contacts and Slippage

If the robot is walking on soft or collapsible ground, when the contact is first detected before the ground starts its deformation (Figure 12.9, left). Then, the leg stretches penetrating the terrain. Because the contact point is assumed to be stationary, this is interpreted by the forward kinematics as an upward motion (Figure 12.9, right). Even if the effect is similar to leg deformation, in this case it is the

assumption of zero velocity of the contact point to be broken [322], since the contact point moves downwards as the ground deforms. This problem is similar to slippage, when a foot is considered in contact with the ground but the forces it exerts on the terrain violate (12.22) and/or (12.24).

In this case, an exteroceptive sensor such as a camera is needed to make the velocity of the contact point observable in non-degenerate motions and robot configurations [1089]. The velocity of the contact can be explicitly tracked as an additional state in [1188], or as the derivative of the feet positions [586].

12.6 New Trends and Paradigm Shifts

In this last section, we focus on some of the recent trends in state estimation and legged robotics. While some of them address part of the open challenges discussed in the previous section, such as contact estimation, others also represent significant paradigm shifts in the current techniques—particularly those aided by learning algorithms.

12.6.1 Learning-based Contact Estimation

We discussed how contact estimation assumes rigid contact which is easily violated by situations such as slippage, soft terrain, or leg deformation. Given the challenges of accurately modeling these problems, it has been proposed to use data-driven methods for contact estimation. These approaches generally aim to learn a binary signal that determines when contact was established. This has been demonstrated in a supervised manner by learning contact classifiers [149] but also in an unsupervised fashion via clustering [953], where proprioceptive sensing (joint and inertial) provide the main signals for the models.

With the rise of deep learning methods, it has been proposed to use neural network architectures to determine the contact state of the feet. This has shown better generalization of to a wider set of structured and unstructured environments [676]. Vision-based haptic sensors, which capitalize on the progress of machine learning and computer vision [648], are another direction that shows promise to improve force and contact estimates [1009].

12.6.2 End-to-End Learning

While high-frequency leg odometry and proprioceptive state estimation were developed to achieve closed-loop model-based locomotion control, the current progress in reinforcement learning (RL) has challenged their necessity. RL-based locomotions controllers showed that only the body velocity and orientation are required for locomotion, which can be provided explicitly by a standard proprioceptive state estimators [506], or even raw data from joint and inertial sensing [645, 769].

While the latter might question the need for state estimation, this is explained by the manner these particular RL-based controllers are trained. The training objective aims to track velocity commands, emulating the way in which the robot will be controlled by a human operator or planning system. To achieve this, the locomotion controller only needs to know the orientation of the base with respect to the gravity vector, as well as the instantaneous body velocity. As seen in Chapter IMU, these quantities are fully observable from inertial data, hence can be implicitly estimated during training.

In contrast, another current trend in locomotion learning aims to achieve advanced mobility skills to navigate the world, by learning locomotion controllers that are able to traverse different obstacles and reach goals relative to a starting position—*robot parkour* being an example [1306, 472]. Achieving this navigation behavior does require access to an odometry estimate, since the robot needs to keep track of the progress towards the goal in an inertial frame. This suggests that state estimation is still needed to achieve more complex locomotion and navigation tasks.

A few works have proposed to learn state estimation as part of the locomotion policy learning process [523], and explicitly estimate variables such as the body velocity, feet height, and contact state. While this has only been used for locomotion purposes, it can be a promising alternative to obtain more accurate leg odometry estimates for proprioceptive state estimation or odometry factors in SLAM.

12.6.3 Humanoid Robots

Humanoid robots embed part of the dreams that have motivated the development of robotics—creating artificial agents able to do the *dull, dangerous, and dirty* tasks that humans prefer not to do. Having a *human-like body* should—in principle—enable them to seamlessly work in human-oriented environments, using tools, devices, and even vehicles designed for people's use.

The DARPA Robotics Challenge, briefly introduced in Section 12.1.1, has been one of the main efforts in this direction. The diverse set of tasks, involving locomotion on rough terrain but also tool handling and driving vehicles, presented several challenges towards this goal. However, after it ended in 2015, most of the efforts in legged robotics focused on quadrupedal platforms instead—which presented clear advantages in control and robustness, motivating their adoption for industrial inspection and monitoring. It was not until 2021 when a commercial interest in humanoid platforms arose again, motivated by the optimism and fast-pacing progress in artificial intelligence (AI), as well as the success of quadrupedal robots.

Diverse companies have recently aimed to develop new humanoid platforms as a way to *embody* AI systems in the real world. Humanoid robots have been targeted to solve complex tasks in delivery, warehousing, and manufacturing—working side-by-side with people, in highly demanding environments. This presents different

challenges that push the topics covered in this chapter in directions currently unexplored. Problems such as long-term, accurate and reliable whole-body estimation need to be solved for humanoid robots to be able to achieve tasks in last-mile delivery problems. Intermittent contacts, from the feet but also the trunk, arms, and hands are expected when handling parcels or other objects in a warehouse setting. Similarly, compliance is required when operating close to people in order to be safe—this also relaxes the rigid contact assumptions we made in this chapter.

Acknowledgment

The authors thank Michele Focchi (University of Trento) for his advice in preparing parts of this chapter.

PART THREE

FROM SLAM TO SPATIAL AI

II

Prelude

Marc Pollefeys Ayoung Kim, Frank Dellaert,
Timothy Barfoot, Luca Carlone, and Daniel Cremers

This prelude will be completed after all corrections.