FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Automatic State Estimation of an Over-Sensored Robotic Manipulator

João Pedro Ribeiro Moreira

U.PORTO

FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Paulo José Cerqueira Gomes da Costa

Second Supervisor: Vítor Hugo Machado Oliveira Pinto

February 20, 2021

Resumo

Nos últimos anos tem-se observado uma procura cada vez maior por manipuladores robóticos capazes de efetuar atividades complexas e versáteis. De modo a satisfazer essa necessidade, é essencial a implementação de técnicas de calibração e estimação expeditas como um primeiro passo para o correto uso do manipulador. A resolução destes problemas aumenta a performance do manipulador quando este executa tarefas de mais alto nível, tais como posicionamento de uma ferramenta ou manipulação de objetos.

Existem atualmente várias técnicas de calibração automática para uma larga gama de sensores, assim como vários filtros capazes da fusão das suas medidas. Esta dissertação tem como objetivo encontrar e implementar um conjunto destes algoritmos capazes de ser usados num manipulador genérico. As técnicas escolhidas são modulares, o que torna possível a sua utilização em diferentes configurações de manipuladores. Os requisitos para a sua utilização foram restringidos, tanto quanto possível, de modo a serem utilizáveis por todos os que tenham limitações relativas a equipamento laboratorial.

Um manipulador robótico desenvolvido especificamente para este estudo é usado como base para a implementação e teste dos diferentes métodos. Este está equipado com *encoders* incrementais, IMUs e células de carga. Um conjunto de métodos para a calibração dos sensores inerciais é descrito e as medidas calibradas são usadas em conjunto com os dados obtidos dos *encoders* para estimar a pose do manipulador.

São descritos dois modelos para a representação da pose do manipulador, os quais são usados no problema de estimação de estado. Um deles define o estado como a dinâmica dos ângulos de cada articulação. O outro usa a orientação de cada ligação do manipulador no espaço inercial. O estado do primeiro modelo é estimado usando o *Unscented Kalman Filter* e o segundo usando o *Multiplicative Extended Kalman Filter*.

Os resultados da implementação são testados e métricas da *performance* dos métodos são obtidos usando as saídas dos algoritmos e um sistema de medição externo.

Abstract

Nowadays there is an increasing demand of robotic manipulators for performing more complex and versatile tasks. In order to fulfill this need, expeditious calibration and estimation techniques are required as a first step for the correct usage of the manipulator. After solving these problems, the manipulator is better equipped to execute higher level tasks, such as generic tool placement and object manipulation.

There are currently several techniques for automatic calibration of a wide variety of sensors, as well as several filters to fuse their data into useful information. This dissertation aims at finding a subset of these algorithms that could be used in a generic manipulator and should allow for its prompt use. The techniques used were chosen with the purpose of being modular and therefore usable in a wide variety of manipulators. They also assume a minimal amount of requirements necessary for their use, making them suitable for an unequipped user.

A robotic manipulator specifically developed for this study is used to realistically test the performance of the implemented methods. It is equipped with incremental encoders, inertial measurement units and load cells. A calibration methodology for the inertial sensors is described and the calibrated measurements are used together with the information obtained from the encoders to determine the pose of the manipulator.

Two models for the representation of the pose of the manipulator are described and used in the state estimation problem. One of them defines the state vector as the dynamics of the angles of each joint. The other uses the orientation of each link in an inertial frame independently. The state of the first model is estimated with the Unscented Kalman Filter and the second one with the Multiplicative Extended Kalman Filter.

The results of implementation are tested and some performance metrics are obtained using both the algorithms' outputs and an external system.

Acknowledgments

To my father Adriano Moreira, my best teacher in engineering and in life, thank you for the many valuable lessons.

To my friends and family, thank you for giving me strength and support, especially during the times when I felt doubt.

To the bot'n'roll robotic soccer team, especially José Cruz, Inês Ribeiro and António Ribeiro, thank you for introducing me to this great world of robotics. Without you I would have never known it.

Finally, to my supervisors Paulo Costa and Vítor Pinto, I express my profound gratitude for all the aid during this dissertation and for helping me grow professionally.

João Moreira

“Man will never be enslaved by machinery
if the man tending the machine be paid enough.”

Karel Čapek

Contents

List of Figures

List of Tables

Abreviations

DC	Direct Current
DH	Denavit-Hartenberg
DOF	Degrees Of Freedom
EKF	Extended Kalman Filter
GUI	Graphical User Interface
IDE	Integrated Development Environment
IMU	Inertial Measurement Unit
KF	Kalman Filter
MEKF	Multiplicative Extended Kalman Filter
MEMS	Microelectromechanical System
PD	Proportional Derivative
PF	Particle Filter
PID	Proportional Integral Derivative
SVD	Singular Value Decomposition

Chapter 1

Introduction

1.1 Context

Robotics has been an integral part of society for several decades. It is present in the industry, the medical field, the military, among others. Wherever physical work is required, robotics often serves as a cheaper, stronger, more precise, and overall more effective alternative to human labor. In addition, robots have been proved to be capable of performing tasks that are otherwise dangerous, if not impossible, for people to do, such as search and rescue missions in hazardous environments.

Within this broad field, one of the main areas is the control of rigid structure robotic manipulators. These devices are composed of rigid sections, denoted links, with joints in between them. The joints cause the links to move relative to each another. This apparently simple setup can be used to solve a wide range of problems. It is not a coincidence that these devices are usually called robotic arms. They can perform almost any task that a human arm can, including screwing, nailing, painting, and even performing medical surgery.

Despite their utility, the use of robotic manipulators is conditioned by their setup. Initially, the different aspects of a manipulator must be defined according to its goal. These include its configuration (number and type of joints, links' shape and size, among others), number and type of sensors and actuators, micro-controller to be used, and overall design of its structure. Furthermore, software needs to be developed to allow it to fulfill its purpose.

The software requirements of a manipulator can be separated into high-level and low-level ones. High-level requirements can vary widely according to the goal, and can include processes as complicated as decision making or as simple as having a tool follow a predefined trajectory. Low-level requirements are usually more similar between different manipulators. These include, but are not limited to, sensing the environment, turning the sensor measurements into a higher level perception and controlling the different actuators. This dissertation focuses on the first two problems: sensing the environment and transforming sensor measurements into higher level information.

1.2 Motivation and Problem Definition

For a manipulator equipped with adequate sensors, having awareness of the environment is a simple task. Most sensors already have custom code libraries that allow for a seamless use of its data. Therefore, difficulty in sensing the environment is not in collecting measurements. The problem lies in making sure the sensors provide useful information. Several sensors, such as magnetometers and accelerometers, provide flawed measurements. These can be influenced by noise and interference, and can provide information that does not directly relate to the variables of interest. For example, magnetometers measure the surrounding magnetic field. For most cases, this information is useless on its own. However, given that the strength and direction of that magnetic field should be constant in the same place, it can be useful for determining orientation.

The process of transforming sensory data into useful information can be divided into two steps. The first transforms the raw measurements of sensors without changing their nature. This step is performed to better fit the measurements to the models that describe the behavior of the manipulator. This is usually dependant on parameters with unknown values. Finding these parameters is often described as calibrating the sensors and constitutes the first problem that this dissertation aims to solve. The second step is related to perception. Once the sensors provide calibrated measurements, turning them into valuable data about the state of the manipulator presents another challenge. In the mentioned magnetometer example, this corresponds to transforming the magnetic field vector into an orientation. This problem is known as state estimation and corresponds to the second problem to be solved throughout this dissertation. It is worth mentioning that this conversion should allow several sensors to provide information about the same variables of interest. When that is the case, this process is also called sensor fusion.

1.3 Objectives and Contributions

The main objective of this dissertation is to analyse and apply several calibration and estimation methods to estimate the state of a generic robotic manipulator. A physical testbed was used to implement the methods and analyse the obtained results. The two problems addressed are sensor calibration and state estimation. Both of these are solved with two constraints in mind:

- All the algorithms must be automatic and independent from external equipment;
- The algorithms should be as modular as possible.

Both constraints are imposed to allow the work developed in this dissertation to be useful for different contexts.

There are numerous solutions in the literature to the calibration and state estimation problems, and many of them are suitable in this context. Despite the best attempts at developing modular solutions, some restrictions were still imposed. Furthermore, the modelled and calibrated sensors were limited to some presented in the physical manipulator testbed. New models for other sensors can be developed to expand the usefulness of the content of this dissertation, since the solutions presented are not unique.

The work developed throughout this dissertation has resulted in the publication of one paper in an international conference [11]. Although not published, another two manuscripts were submitted, namely one to an international journal and the other to an international conference.

1.4 Document Structure

The remainder of this document is organized as follows:

- Chapter 2 [State of the Art] - Conceptual description of the subsystems composing the manipulator and initial presentation of the available solutions to the problems;
- Chapter 3 [Robotic Manipulator] - Presentation of the manipulator used for testing and development of its models;
- Chapter 4 [Inertial Measurement Units Calibration] - Development and analysis of the calibration algorithms for inertial measurement units;
- Chapter 5 [State Estimation] - Development and analysis of the state estimation algorithms;
- Chapter 6 [Conclusions] - Description of the main conclusions drawn from the results of the implemented solutions and discussion of future work.

Chapter 2

State of the Art

The design process behind the implementation of a state estimator starts by modeling the system as well as its components. More specifically, mathematical models capable of describing any of the subsystems and sensors of the robotic manipulator must be developed.

A study of the applied subsystems is performed in this chapter. The sensors used in the manipulator are explored, namely the inertial measurement unit in 2.1. Following that, the models of the DC motor and the kinematics of robotic arms are examined in sections 2.2 and 2.3, respectively. Finally, several sensor fusion techniques are studied and analysed. The first explored techniques are variants of the Kalman Filter, followed by the Particle Filte, throughout sections 2.4 and 2.5.

2.1 Inertial Measurement Unit

An inertial measurement unit (IMU) is, in its simplest form, a device composed of accelerometers, capable of measuring specific force (acceleration subtracted by the acceleration of gravity), and gyroscopes, capable of measuring angular velocity. With these, position and orientation can be determined at any moment in time, provided an initial position, velocity, and orientation.

Sometimes an IMU is also equipped with magnetometers capable of measuring magnetic fields. When this is the case, they can be used to correct drift errors, which will be mentioned later.

Most IMU sensors consist on three accelerometers and three gyroscopes. These trios of sensors are unaligned to provide measurements in all 3 spatial dimensions. However, these are not always perfectly orthogonal to one another, meaning that a measurement in one sensor might correspond to a value in more than one axis. Even if the axes were orthogonal to one another, they still might not be aligned with the chosen referential.

Another issue about modeling an IMU is the existence of parameters of its model that depend on the specific conditions under which it is working, such as temperature and magnetic interferences.

The aforementioned adversities, along with others, are the reason behind the need for the adjustment of the parameters of the IMU's model. The way these adjustments are made is by moving the IMU in a specific way and collecting data from the sensors. By associating the measurements with the movements, the parameters can be calculated. The process through which the parameters are estimated is called calibration.

2.1.1 Model

Each of the IMU's sensors depend on some parameteres, namely a gain (g) and a bias (b). The function that relates these parameters to the electric signal measured by the sensor (m) and the measured value (v) is [10, 16]:

$$v = g(m - b) \tag{2.1}$$

where v is usually expressed in m/s^2 (when measuring acceleration), rad/s (when measuring angular velocity) or μT (when measuring the magnetic field) and m is usually expressed in mV.

An especially troublesome parameter in this model is the bias. If it is not accounted for, or not correctly determined, it causes a systematic error in the measurements of the inertial sensors. This error accumulates when determining position and orientation since these are calculated by integrating the original measurements. As a consequence, a drift is created in the position and orientation state variables. This is the steady increase of a variable that should remain approximately stationary.

This is the affine relation that models single axis sensors. However, as previously stated, the IMU has several sensors. Moreover, each sensor can influence more than one axis of the chosen referential. This implies that each measurement (m_i, $i = \{x,y,z\}$) is influenced by each value (v_i, $i = \{x,y,z\}$)

$$v_i = g_{ix}(m_x - b_x) + g_{iy}(m_y - b_y) + g_{iz}(m_z - b_z) \qquad i = \{x,y,z\} \tag{2.2}$$

Assuming $V = [v_x \quad v_y \quad v_z]^T$ and $M = [m_x \quad m_y \quad m_z]^T$, the new model equation is

$$V = G(M - B^*) \tag{2.3}$$

where G is a 3x3 matrix with entry (i,j) equal to g_{ij} for $i = \{x,y,z\}$, and $B^* = [b_x \quad b_y \quad b_z]^T$. Alternatively, the previous equation is often written as

$$V = GM + B \tag{2.4}$$

with $B = -GB^*$. Considering the non-diagonal elements in the gain matrix (G), scaling factors and misalignments between the axes and with the reference frame are all accommodated.

2.1.2 Calibration

The calibration process for the mentioned model finds the parameters G and B. This is usually done by reading several measurements in a multitude of scenarios and attempting to find the parameters that match the real values with the measurements.

After modelling the sensors, as described above, the problem seems to be solvable by simply using algebra to solve the equations for the parameters. However, two problems interfere with the use of this method. First, all measurements are noisy, which means that, if there are more measurements than parameters, the system of equations probably won't be solvable. Second, the use of these equations assumes that the true measured values are known.

Regarding the first, there are several methods that take noise into account and try to find the parameters that better fit the noise model. One example is the solution to linear least squares. If the model is linear, as is the case with the IMU's sensors, this solution is the minimum variance unbiased one [18], assuming the noise is white and has a zero mean Gaussian probability distribution. Alternatively, several optimization algorithms can be used and are usually chosen when the model is not linear.

Since acquiring the real values, often referred to as *ground truth*, is theoretically impossible, instead of trying to achieve perfect real values, high precision instruments are used to obtain practically acceptable ones. When these calibration methods were originally proposed, they were designed for tactical grade IMUs. The sensors were usually fixed to high precision actuators, such as turntables or robotic manipulators, and their movement was precisely controlled by these. For example, in order to calibrate an accelerometer one could read a measurement while one of its axis was aligned vertically. When under this circumstance, that axis would be measuring the symmetric of the acceleration of gravity and the bias. Then, if turned upside down, the bias would remain the same where as the symmetric of the acceleration of gravity would flip. More specifically,

$$m_{up,i} = b_i + m_{g,i} \qquad m_{down,i} = b_i - m_{g,i} \qquad i = \{x, y, z\} \tag{2.5}$$

with $m_{g,i}$ the influence of gravity on the sensor measurement and b_i the bias of axis i. Both of these measurements are enough to find the bias along the i axis using their average:

$$\frac{m_{up,i} + m_{down,i}}{2} = \frac{b_i + m_{g,i} + b_i - m_{g,i}}{2} = \frac{2b_i}{2} = b_i \tag{2.6}$$

Using the biases, some unbiased measurements ($U = M - B$) can be calculated and used to find the gain matrix G:

$$V = GU \tag{2.7}$$

With enough measurements, G can be easily calculated with the solution to the least squares problem.

Similarly, the gyroscope can be calibrated by first finding the biases and then imposing known rotations to obtain the gain matrix. The biases can be easily acquired by averaging enough measurements while the IMU is stationary. Then, an angular velocity is imposed along an axis and matched with an unbiased measurement to find the gain. Alternatively, the unbiased measurements could be integrated, requiring the moving frame to only keep track of the overall change in orientation.

More general setups that estimate the gain and bias matrices at the same time could also be used. In that case, external systems such as optical trackers can be used to measure imprecise movements with high precision.

However, with the creation of low-cost MEMS (Microelectromechanical Systems) IMUs, the devices became more popular among hobbyists. Those without special equipment needed to find a way to calibrate their sensors without a ground truth. Several new calibration techniques that relied on the properties of the measured quantities have been developed due to this necessity.

For example, when stationary, the accelerometers should measure a vector that has constant norm. From this property, the following equation can be used to find the parameters [16]:

$$||GM + B||_2 = g \tag{2.8}$$

This equation is not linear, which means least squares cannot be directly used in this case. Alternatives include using other optimization methods or redefining the variables to make the problem linear [16]. In the absence of magnetic field disturbances, this method can also be used to calibrate magnetometers. Alternatively, after calibrating one of the sensors, the other could be calibrated by assuming that the dot product of their reference vectors (magnetic field and symmetric of the acceleration of gravity) is constant in the working region [8]. This method has the benefit of correcting misalignments between the frames of the accelerometers and the magnetometers.

The gyroscope does not have a reference vector to measure while it is standing still. The rotation of the Earth could be used, however it is negligible when compared to the magnitude of the noise of low cost MEMS gyroscopes. This absence of a reference makes it difficult to independently calibrate a gyroscope based on its static measurements. However, once the other sensors are calibrated they can be used as an acceptable ground truth. When reading a reference vector from another sensor before and after a rotation, the rotation matrix obtained by integrating the gyroscope measurements should match those vectors [2]. More specifically, consider R_ω the rotation matrix obtained from the measurements of a calibrated gyroscope and r the reference vector. Then, the equation that relates the reference vector before and after the rotation is

$$r(k+n) = R_\omega(k, k+n)r(k) \tag{2.9}$$

Since R_ω depends on the calibration parameters, this equation can be used by an optimization algorithm. Any method that finds the minimum of the objective function should find the correct parameters G and B

$$f(G, B) = \sum_k ||r(k+n) - R_\omega(k, k+n)r(k)||_2^2 \tag{2.10}$$

This method, much like the dot product one, corrects for misalignments between sensor frames since it's using another sensor as the ground truth.

2.2 DC Motor

A DC motor is usually modelled as a resistor (R), a coil (L) and a voltage source ($v_b(t)$) in series [13, 3].

$$v_s(t) = Ri(t) + L\frac{di(t)}{dt} + v_b(t) \tag{2.11}$$

In the equation, $v_s(t)$ is the voltage applied to the motor and $i(t)$ is the current that goes through it.

The resistor and the coil represent energy losses in the motor, which means that all useful energy is spent in the voltage source $v_b(t)$ This voltage source is a counter-electromotive force produced by the rotor, and therefore, it is proportional to the angular velocity with which it spins. Given that the power delivered at $v_b(t)$ is useful, then it is equal to the mechanical power delivered at the rotor. Since the useful electric power is equal to $v_b(t)i(t)$ and the mechanical power is given by the product of the torque ($T(t)$) and the angular velocity of the rotor ($\omega(t)$), the torque must be proportional to the current that goes through the stator.

$$v_b(t) = k\omega(t) \qquad\qquad T(t) = ki(t) \tag{2.12}$$

This information can be combined with the mechanical equation of the motor to define a continuous time model of the motor.

The mechanical equation relates the angular acceleration with all the torques applied to it.

$$J\frac{d\omega(t)}{dt} = T(t) - T_e(t) - B\omega(t) \tag{2.13}$$

In this equation, J is the moment of inertia, $T(t)$ is the torque produced by the motor, $T_e(t)$ is the set of opposing torques produced outside of the motor and B is the drag coefficient (making $B\omega(t)$ the drag induced torque).

Combining the electric equation (Equation (2.11)) with the mechanical one (Equation (2.13)), the following can be written.

$$L\frac{di(t)}{dt} = v_s(t) - Ri(t) - k\omega(t) \tag{2.14}$$

$$J\frac{d\omega(t)}{dt} = ki(t) - T_e(t) - B\omega(t) \tag{2.15}$$

With these equations, the resulting continuous time model of the DC motor is

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{2.16}$$

$$y(t) = Cx(t) \tag{2.17}$$

$$x(t) = \begin{bmatrix} i(t) \\ \omega(t) \end{bmatrix} \qquad u(t) = \begin{bmatrix} v_s(t) \\ T_e(t) \end{bmatrix} \qquad y(t) = \omega(t)$$

$$A = \begin{bmatrix} -{}^{R}\!/_{L} & -{}^{k}\!/_{L} \\ {}^{k}\!/_{J} & -{}^{B}\!/_{J} \end{bmatrix} \qquad B = \begin{bmatrix} {}^{1}\!/_{L} & 0 \\ 0 & -{}^{1}\!/_{J} \end{bmatrix} \qquad C = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

2.3 Robot Kinematics

Kinematics is the field of study of the motion of objects. Regarding robotic manipulators, this field is divided into two: forward and inverse kinematics. Forward kinematics is a tool for determining the position and orientation of any point of an arm in terms of the angles of the robot's joints. Inverse kinematics does the exact opposite: it finds out which angles should be in each joint in order to place a link in a certain position with a certain orientation.

An important aspect of robotic manipulators is the number of degrees of freedom (DOF). This number defines how many ways a manipulator can position and orient itself. The number of degrees of freedom a robotic arm has is equal to its number of joints. Considering there are only 3 spatial dimensions, a 6-DOF manipulator is already capable of positioning and orienting itself in (almost) any way.

The equations defining the forward kinematics of a manipulator are usually translated into matrices. These correspond to linear transformations between two Cartesian referentials, and each of the matrices represents a rotation and a translation

$$\begin{bmatrix} R & t \\ 0_{1\times3} & 1 \end{bmatrix}$$

where R is a 3x3 matrix representing the rotation and t is a 3x1 vector representing the translation.

If one of these transformations is used to find the position of a link with respect to the previous one, several transformations can be chained to find the position of any link with respect to any other. For example, the one that relates link n with the base of the arm is

$$^{n}_{0}T = {}^{1}_{0}T\,{}^{2}_{1}T \cdots {}^{n}_{n-1}T \tag{2.18}$$

where $^{i}_{i-1}T$ is the transformation between referentials $i-1$ and i.

Out of the many different ways of determining the individual transformations ($^{i}_{i-1}T$), one of the most used is the Denavit-Hartenberg (DH) method [7]. This systematic method of finding the entries of a transformation matrix takes only four parameters:

- a - distance between the previous z axis and the current one (translation along the x axis);
- α - angle between the previous z axis and the current one (rotation along the x axis);
- d - distance between the previous x axis and the current one (translation along the z axis);

- θ - angle between the previous x axis and the current one (rotation along the z axis).

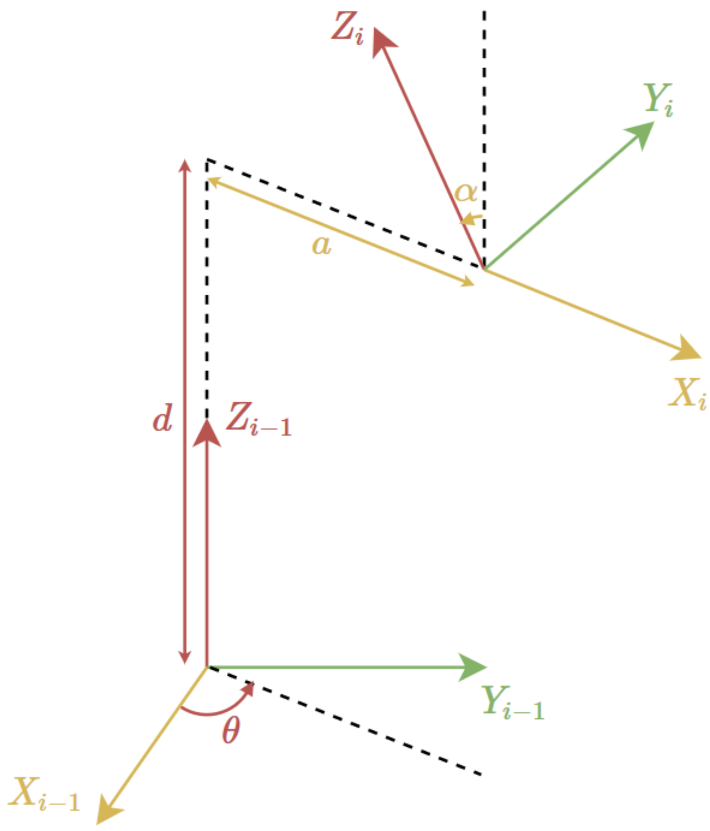


Figure 2.1: Example of a coordinate transformation with DH parameters

These parameters should be applied in the previously described order. However, the translations and the rotations along the same axis can be applied in direct or reverse order.

Once the parameters are determined, the transformation matrix is built from them.

$$T = \begin{bmatrix} \cos\theta & -\sin\theta\cos\alpha & \sin\theta\sin\alpha & a\cos\theta \\ \sin\theta & \cos\theta\cos\alpha & -\cos\theta\sin\alpha & a\sin\theta \\ 0 & \sin\alpha & \cos\alpha & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.19}$$

Since these matrices are built from only four parameters, they cannot be used to represent every possible transformation. To achieve this, at least six parameters would be needed: three for translation and three for rotation. However, the solution to this problem is relatively simple, it uses an intermediate referential. The original referential is transformed into the intermediate one and the intermediate one into a new one. This is equivalent to using eight parameters to build the transformation between the original referential and the new one.

Inverse kinematics analysis can be performed by using the inverse of the transformation matrices, followed by an extraction of the values of the angles at which the joints should be from the entries of the obtained matrices. However, this process is usually computationally expensive. An alternative way of obtaining the inverse kinematics is by geometrically analyzing the manipulator configuration. Even though it is more used, this last method has the disadvantage of not being systematic.

2.4 Kalman Filter

Once a system is modeled, its state can be approximated according to measurements taken from sensors. The state is a vector composed of all variables of interest of the system, and the way it's approximated is with sensor fusion techniques.

The techniques described below assume that the state of the system is a first-order Markov process. That is, the next state will depend on the current state, the system inputs, and some noise. Similarly to the next state, the measurements obtained from sensors are dependent on the state, the inputs, and some noise. Generically, a digital system can be mathematically described by

$$x(k+1) = f(x(k), u(k), w(k), k) \tag{2.20}$$

$$y(k) = g(x(k), u(k), v(k), k) \tag{2.21}$$

where x is the state, u the set of inputs, y the set of outputs, w the process noise and v the measurement noise. It should be noted that the system may be dependent on time (k), which means it isn't necessarily time invariant.

To estimate the state of a system, the *Kalman Filter* (KF) makes some assumptions, the first of which is the linearity of the system. The second is related to the process and measurement noises. They are both assumed to be zero-mean white noise stochastic variables. Under these assumptions, Equation (2.20) and Equation (2.21) can be rewritten

$$x(k+1) = A(k)x(k) + B(k)u(k) + w(k) \qquad w(k) \sim N(0, Q(k)) \tag{2.22}$$

$$y(k) = C(k)x(k) + D(k)u(k) + v(k) \qquad v(k) \sim N(0, R(k)) \tag{2.23}$$

where $N(\mu, \Sigma)$ is a Gaussian distribution of mean μ and covariance matrix Σ. Again, it should be noted that the system matrices (A, B, C, and D) are not time invariant. In the case of the robotic manipulator, the system must be considered time variant, because different joint angles produce different transformations between the referentials of each link (as shown in section 2.3).

If these assumptions are true, the KF is the minimum-variance unbiased estimator [18]. That is, it is the estimator that gives the most certain estimate out of the ones that do not produce a systematic error. This is the reason why it is so commonly used.

The Kalman Filter's algorithm is usually implemented in 2 stages: prediction and filtering. The prediction stage uses the model of the system and the current state (or its estimate) to predict what the next state might be. Besides that, it determines the level of certainty of that predictions in the form of a covariance matrix. In the filtering stage, measurements are used to correct the values of the prediction stage. The influence that the measurements have in the final estimate depend on the previously calculated covariance matrix and the certainty granted to the measurements. A covariance matrix of the final estimate is also calculated as it is needed in the prediction stage. Assuming the process noise and the measurement noise are uncorrelated, being the most common situation, the estimator is implemented following these steps:

Prediction:

1. $\hat{x}(k+1|k) = A(k)\hat{x}(k|k) + B(k)u(k)$ (next state estimate)
2. $P(k+1|k) = A(k)P(k|k)A(k)^T + Q(k)$ (next state estimate's covariance)

Filtering:

1. $e(k) = y(k) - C(k)\hat{x}(k|k-1) - D(k)u(k)$ (innovation)
2. $S(k) = R(k) + C(k)P(k|k-1)C(k)^T$ (innovation's covariance)
3. $K(k) = P(k|k-1)C(k)^T S(k)^{-1}$ (Kalman gain)
4. $\hat{x}(k|k) = \hat{x}(k|k-1) + K(k)e(k)$ (current state estimate)
5. $P(k|k) = (I - K(k)C(k))P(k|k-1)$ (current state estimate's covariance)

In this algorithm $\hat{x}$ is a state estimate and the notation $(\alpha|\beta)$ refers to the instant at which the estimate corresponds (α) given measurements up to time β.

The variables calculated in steps 1, 2, and 3 of the filtering stage are merely auxiliary. The innovation $(e(k))$ is the error in the measurements, *i.e.* the difference between the expected measurements and the ones actually obtained. The innovation's covariance $(S(k))$ is used to calculate the Kalman gain $(K(k))$: a matrix which weighs the trust in the measurements against the trust in the model to find out which of these should have more influence in the filtered estimate.

These two stages are then iteratively applied to estimate the state at every time step. After the prediction stage, the current time instant is updated $(k \leftarrow k+1)$ and the filtering stage can now be applied, since $\hat{x}(k|k-1) \leftarrow \hat{x}(k+1|k)$ and $P(k|k-1) \leftarrow P(k+1|k)$. Then, the prediction stage is executed again, making this process cyclic. An initial estimate $(\hat{x}(0|-1))$ and its covariance $(P(0|-1))$ are required in order to begin the algorithm.

2.4.1 Extended Kalman Filter

The Kalman Filter assumes that the system is linear. However, the robotic manipulator is not a linear system. As seen in section 2.3, some entries of the transformation matrices are trigonometric functions of the joint's angles.

State estimation of non-linear systems is a common problem. Since the performance of the KF is such an appealing factor, many variants capable of estimating the state of non-linear systems, have been developed. One of them is the *Extended Kalman Filter*. The EKF starts by finding the required Jacobian matrices (matrices composed of partial derivatives) and uses them to describe an approximation of the system.

$$A(k) = \frac{\partial f}{\partial x}(k) \qquad B(k) = \frac{\partial f}{\partial u}(k) \qquad C(k) = \frac{\partial g}{\partial x}(k) \qquad D(k) = \frac{\partial g}{\partial u}(k) \tag{2.24}$$

Then it solves the estimation problem for the approximate system with a conventional KF.

This simple workaround removes the assurance that this is the minimum-variance unbiased estimator. However, it produces good enough results to be used in many practical applications.

Even though this filter is usable in the presented context, it displays some issues when used to estimate orientations. The main problem is in the way that orientation is represented. For example, when the orientation of a body is described by a rotation matrix, the sum of two of those matrices most likely is not a valid rotation matrix (*i.e.* is not orthogonal). So, representing an orientation by the sum of its estimate with its error is not a valid model.

$$R_\theta = R_{\hat{\theta}} + R_\varepsilon \tag{2.25}$$

This makes it difficult to apply the filtering step of the Kalman filter.

One way to avoid this problem is to assume that Equation (2.25) is valid and to modify the resulting matrix to become orthogonal again. If the orientation was being described by unit quaternions, this process would be even simpler, as the only change needed to turn the quaternion into a valid orientation would be to divide it by its magnitude. However, allowing the filter to assume that rotation matrices can merely be summed produces an incorrect model of the system, which causes the filter to over-estimate its confidence in the state estimate [6].

A different way to model the dynamics of an orientation is through matrix multiplication.

$$R_\theta = R_{\hat{\theta}} R_\eta \tag{2.26}$$

This equation always produces a valid rotation matrix, assuming there are no rounding errors, as long as the error matrix (R_η) is a rotation matrix as well. R_η can be made into a valid rotation matrix if it is built by an orientation representation that is not redundant (*i.e.* has no constraints), for example, a rotation vector η. If the state is represented by this rotation vector, then it can take any value and still produce a valid rotation. Once the rotation vector is determined, the rotation matrix changes according to Equation (2.26) and the vector is reset to zero. This extra step in the MEKF is called the reset step, and it is what allows for the orientation to be stored in the rotation matrix while the filter operations are applied to the rotation vector.

This approach to orientation estimation is called the *Multiplicative Extended Kalman Filter* (MEKF). This is often used in spacecraft orientation estimation [1]. The filter will be further explored when the robotic manipulator model is explained.

2.4.2 Unscented Kalman Filter

Another variant of the Kalman Filter is the *Unscented Kalman Filter* (UKF). This estimator doesn't assume linearity. Instead, it uses *sigma points* to describe the state's distribution. Sigma points are vectors of possible states.

These points are calculated in function of the previous estimate, the covariance matrices (of the state and noises), and some parameters. Then, they are updated according to the non-linear model of the system. Each of the points is then used, together with measurements, to calculate the Kalman gain. From this gain, the error between the expected measurements and the real ones, the predicted state, the filtered estimate, and its covariance matrix can be determined.

The UKF algorithm, in the case of additive noises, is described as follows [17]:

Sigma Points:

1. $\mathscr{X}_0(k|k) = \hat{x}(k|k)$
2. $\mathscr{X}_i(k|k) = \hat{x}(k|k) + \left(\sqrt{\frac{n}{1-\mathscr{W}_0}P(k|k)}\right)_i \qquad i = \{1, 2, ..., n\}$
3. $\mathscr{X}_i(k|k) = \hat{x}(k|k) - \left(\sqrt{\frac{n}{1-\mathscr{W}_0}P(k|k)}\right)_{i-n} \qquad i = \{n+1, n+2, ..., 2n\}$

Prediction:

1. $\mathscr{X}_i(k+1|k) = f(\mathscr{X}_i(k|k), u(k), k)$ (next sigma points)
2. $\hat{x}(k+1|k) = \sum_{i=0}^{2n}[\mathscr{W}_i\mathscr{X}_i(k+1|k)]$
3. $P(k+1|k) = \sum_{i=0}^{2n}\left[\mathscr{W}_i(\mathscr{X}_i(k+1|k) - \hat{x}(k+1|k))(\mathscr{X}_i(k+1|k) - \hat{x}(k+1|k))^T\right] + Q(k)$
4. $\mathscr{Y}_i = g(\mathscr{X}_i(k+1|k), u(k), k)$ (expected "sigma" measurements)
5. $\hat{y}(k+1|k) = \sum_{i=0}^{2n}[\mathscr{W}_i\mathscr{Y}_i]$ (expected measurements)

Filtering:

1. $e(k) = y(k) - \hat{y}(k|k-1)$
2. $P_{y,y} = \sum_{i=0}^{2n}\left[\mathscr{W}_i(\mathscr{Y}_i - \hat{y}(k|k-1))(\mathscr{Y}_i - \hat{y}(k|k-1))^T\right] + R(k)$
3. $P_{x,y} = \sum_{i=0}^{2n}\left[\mathscr{W}_i(\mathscr{X}_i(k+1|k) - \hat{x}(k|k-1))(\mathscr{Y}_i - \hat{y}(k|k-1))^T\right]$
4. $K(k) = P_{x,y}P_{y,y}^{-1}$
5. $\hat{x}(k|k) = \hat{x}(k|k-1) + K(k)e(k)$
6. $P(k|k) = P(k|k-1) - K(k)P_{y,y}K(k)^T$

In the algorithm:

- n is the length of the state vector;
- $\mathscr{W}_i$ is the weight of the i-th sigma point, *i.e.* its contribution to the calculation of the estimates and covariances;
- $-1 < \mathscr{W}_0 < 1$;
- $\sum_{i=0}^{2n} \mathscr{W}_i = 1$, that is $\mathscr{W}_i = \frac{1-\mathscr{W}_0}{2n}$ for all $i = \{1, 2, ..., 2n\}$;
- $\left(\sqrt{\frac{n}{1-\mathscr{W}_0} P(k|k)}\right)_i$ is the i-th column of the matrix $\sqrt{\frac{n}{1-\mathscr{W}_0} P(k|k)}$ and the square root matrix is obtained from the Cholesky decomposition [4].

2.5 Particle Filter

Not all estimators use the same assumptions as the KF and its variants. One of those estimators is the *Particle Filter*.

Instead of assuming linearity and noises with a Gaussian distribution, this filter assumes that the system is ergodic. That is, if enough data points (or *particles*) exist, they can be used to represent the probability distribution of the state.

By making the particles evolve according to the model (prediction) and subsequently choosing the ones that are most likely to produce the measured outputs (filtering), they should converge towards the correct state. The prediction stage is performed in one step and the filtering stage is done in the next two. Given an initial set of particles:

1. Each particle is updated according to the system's model, which isn't necessarily linear (prediction);
2. A weight is attributed to each new particle, which is proportional to the probability that the obtained measurements came from the state represented by that particle. That probability is $P(y(k)|\hat{x}_i(k))$, where $\hat{x}_i$ is the state represented by particle i (filtering);
3. New particles are sampled from the existing ones. The probability that a particle is chosen is proportional to its weight (filtering).

The estimate is obtained by the average of all particles. Since the particles represent the state distribution, other statistical values of interest, such as the covariance matrix, can be calculated from the set of particles.

The fact that the PF and the KF are both divided in two stages, prediction and filtering, is not a coincidence. These estimators are a part of a family of filters called Bayes Filters. Bayes Filters owe their name to the Bayes Theorem.

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)} \tag{2.27}$$

This theorem is used in any Bayes Filter. The equation states that, given some measurements, the probability that a certain state is the correct one ($P(x|y)$) is proportional to the probability that those measurements were generated by that state ($P(y|x)$) and the probability that the state occurs independently from the measurements ($P(x)$). $P(x)$ is denoted the *a priori* probability because it is data which is known from the model *before* any measurements are obtained. *After* having the measurements, the *a posteriori* probability ($P(x|y)$) can be calculated. The probability $P(y|x)$ is dependent on the estimator used. In the case of the KF, this probability is assumed to be given by a Gaussian distribution, whereas, in the case of the PF, it's given by the distribution of the particles.

Even though the PF makes much broader assumptions than the KF, it has other disadvantages. For instance, ergodicity is only a valid property as long as there is a large enough number of particles. This makes the algorithm take longer to run. Even when many particles are used, there is still the chance that the estimate converges into a wrong state. To avoid this behavior, an extra step can be added to the algorithm. In this extra step, a certain amount of the particles is randomly altered, independently of the system's model and the measurements. If the estimator converges into a wrong state, some randomly generated particles can end up near the correct state, causing them to weigh more and shifting the estimate to the real state. This only occurs if the randomly generated particles would produce sensor readings similar to the real ones.

Chapter 3

Robotic Manipulator

This chapter presents the robotic manipulator that is used throughout this dissertation. Initially, its hardware is described in 3.1. Then, the software that controls it is presented in 3.2. Finally, in 3.3 all the mathematical models of the manipulator and its sensors are deduced with the intent of using them for state estimation.

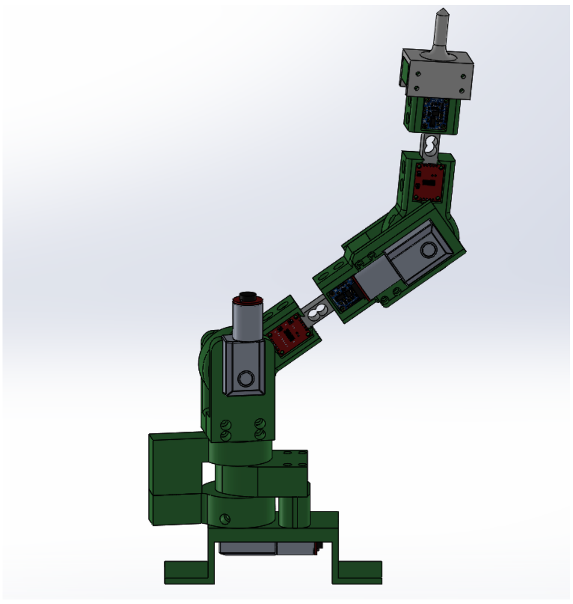
3.1 Hardware

The developed manipulator was built with the intent of studying over-sensored systems in an academic context. With this purpose in mind, the manipulator was designed based on three criteria:

- Its structural components must be 3D printable;
- The design and sensors have to be low-cost;
- It should be as modular as possible.

Considering these objectives, the manipulator presented in figure 3.1 was designed and implemented.

It uses three JGY-371 motors with worm gearbox and incremental encoder. The worm gearbox gives the motor the unique capability of self-locking, being able of saving energy while the motor is stopped. There are two TAL220 load cells, mounted on links 2 and 3, with the corresponding signal amplifiers (HX711). These make the system capable of measuring the torque applied in each link. Links 2 and 3 are also equipped with inertial measurement units. There is one MPU-9250 IMU in link 2 and one Adafruit 3463 Precision 9-DOF IMU in link 3. An Arduino Uno is used to process the data from the sensors and controls the motors with an Adafruit Motor Shield v2. Table 3.1 is a bill of materials with the price range of each component. The schematic for the electrical wiring is in Figure 3.2.


(a) 3D model

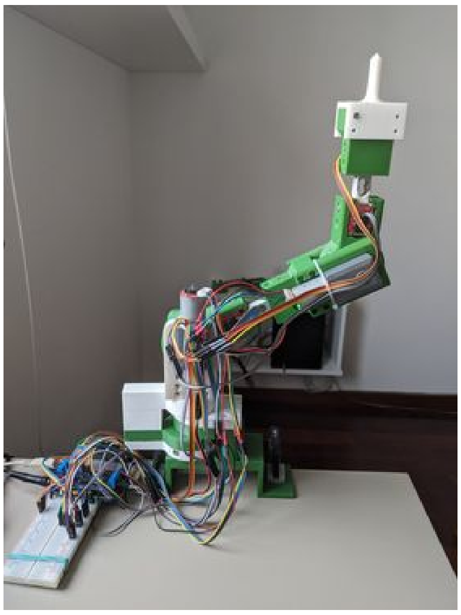

(b) Implementation

Figure 3.1: Designed robotic manipulator

Reference	Description	Price Range
TAL 220	10 Kg Load Cell	10-12 €
A000066	ARDUINO UNO REV3	20-25 €
HX711	Load Cell Amplifier	12-15 €
ADA-3463	Adafruit Precision NXP 9-DOF Breakout Board	15-18 €
MPU-9250	GY-91 10DOF MPU-9250 and BMP280 Multi-Sensor Module	10-15 €
JGY-371	12V DC Motor with worm gearbox and incremental encoder	25-30 €
ADA-1438	Adafruit Motor/Stepper/Servo Shield for Arduino v2 Kit - v2.3	20-25 €
TOTAL		184-227 €

Table 3.1: Bill of Materials of the robotic manipulator

3.2 Software

The control loop of the robotic manipulator is separated in two parts. The Arduino is used for low level tasks such as reading measurements from the sensors and controlling the motors. Serving as a central processing unit, a PC application is used, connected directly to the Arduino Uno by USB cable. This application is written in the Free Pascal programming language, specifically using the Lazarus Integrated Development Environment (IDE). It is responsible for high level tasks such as state estimation and user interaction.

More specifically, the control loop starts in the Arduino, where raw measurements from all sensors are collected. Then, the measurements are transmitted to the PC via serial communication. The application receives this data and displays it in a *Graphical User Interface* (GUI). The GUI can also display graphically the evolution of any variable of interest and interact with the user. For example, the user can move sliders to set the position or velocity of any motor and can set the PID

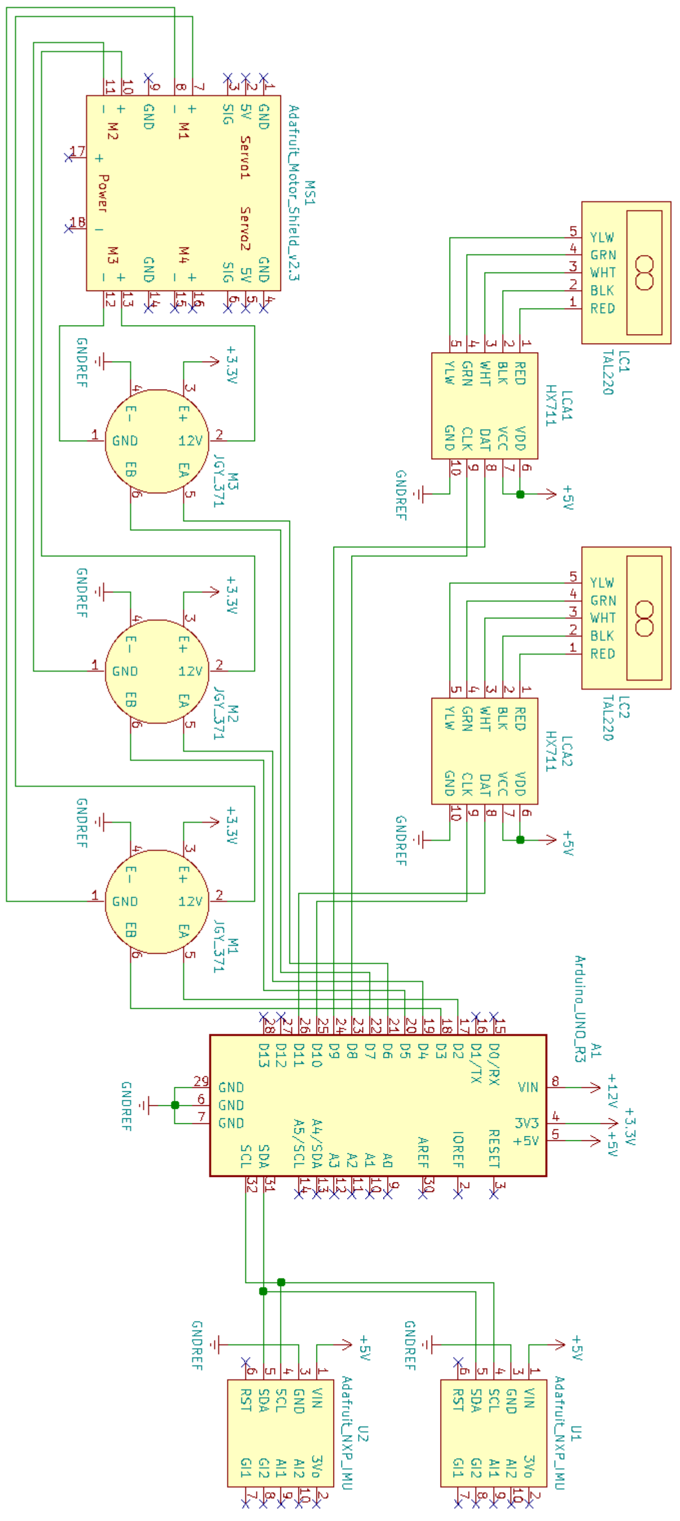
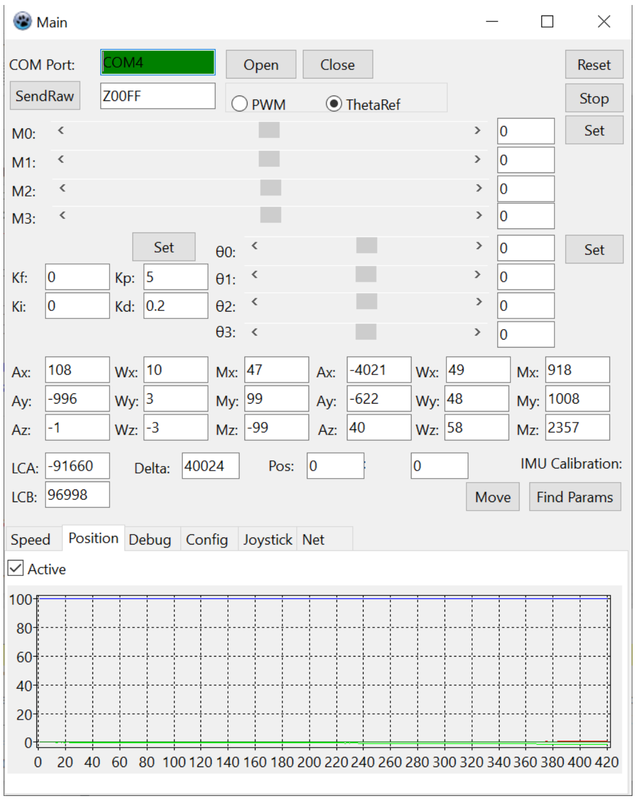


Figure 3.2: Schematic of the Robotic Manipulator

(Proportional Integral Derivative) parameters for control of the motors. Besides interacting with the user through the GUI, the application preprocesses measurements according to its calibration



COM Port: COM4
SendRaw: Z00FF
PWM [✓ ThetaRef]
M0: 0
M1: 0
M2: 0
M3: 0
θ0: 0
θ1: 0
θ2: 0
θ3: 0
Kf: 0 Kp: 5
Ki: 0 Kd: 0.2
Ax: 108 Wx: 10 Mx: 47 Ax: -4021 Wx: 49 Mx: 918
Ay: -996 Wy: 3 My: 99 Ay: -622 Wy: 48 My: 1008
Az: -1 Wz: -3 Mz: -99 Az: 40 Wz: 58 Mz: 2357
LCA: -91660 Delta: 40024 Pos: 0 0
LCB: 96998
[✓] Active

Figure 3.3: Graphical User Interface of the PC application

parameters and then uses them in state estimation algorithms. The algorithms for calibration are also run in this application. The references for the motors are then sent to the Arduino, again using serial communication, which control the motors using a PID controller. Figure 3.3 shows the GUI with which the user can interact.

3.3 Models

Throughout this section, the dynamics of the robotic manipulator and its sensors will be described. These will be decisive in the application of the filters with which the state variables will be estimated.

Initially, the equations that define the behavior of the manipulator and its sensors are written in the most general form possible. This is done with the intent of keeping the models reusable for different manipulator configurations. However, the assumption that all joints are revolute is made because it makes the model simpler and it is a constraint that the implemented manipulator follows. Before describing the models, the used nomenclature is presented. Regarding the coordinate frames, each joint uses two. For joint number i (counting from the base), the two frames are i and $i-$. Both have their origins in the rotation axis of the joint and have the z axis aligned with it. However, frame i is fixed to the link directly after the joint whereas frame $i-$ is fixed to the link directly before the joint. The frames in which the sensors take measurements can be rotated to align with the frames of the adjacent joints. This is also done to keep the nomenclature simpler.

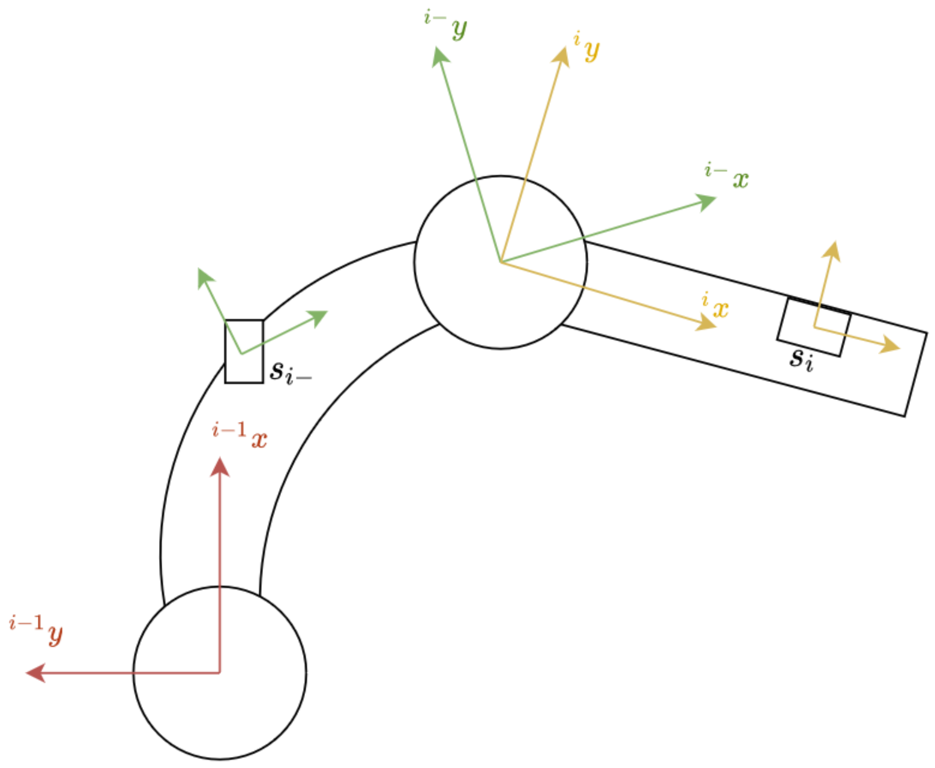


Figure 3.4: Example of part of a manipulator with frames defined by the generic model

Figure 3.4 displays a 2D example of a part of a generic manipulator following these assumptions on the frames. In the figure, i and $i-$ are the frames centered at joint i, s_i is a generic sensor with its frame aligned with i, and s_{i-} is a sensor that gives the measurements taken from sensor s_{i-1} rotated to the $i-$ frame. Note that even though the example is two-dimensional, the z-axes of the frames are aligned with the rotation axis (perpendicular to the image).

Some consequences of this nomenclature are that the transformation matrices $^{i-}_{i-1}T$ are constant and there is not any translation in the transformations $^{i}_{i-}T$. This implies that all the necessary information required to determine the pose of the manipulator can be obtained from the rotation matrices $^{i}_{i-}R$ given by

$$^{i}_{i-}R \triangleq R_{\theta_i} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 \\ \sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

The argument can be extended to the velocity and acceleration of each joint. Appendix A shows how it is possible to obtain the position, velocity, and acceleration of any joint relative to any other joint. All the required information is the rotation matrix, angular velocity, and angular acceleration between the two joints.

In the two following sections, two models are built based on this nomenclature. The first is the

Link model, in which the state is composed of the rotation matrices that represent the orientation of each link with respect to an inertial frame and their respective angular velocities. The second one is the Joint model, in which the angle, angular velocity, and angular acceleration of each joint compose the state vector. The sensors will also be modeled according to each of the state choices.

3.3.1 Link Model

As mentioned, in this model the state is composed of the rotation matrices ${}^{i}_{n}R$ and angular velocities ω_i of each link. Each of these represents the orientation of link i (with 0 being the base) with respect to an inertial frame n. The inertial frame is denoted by n because it refers to the *navigation* frame. This is a frame situated at a specific point on Earth's surface. As the planet rotates, so does the frame. Even though this frame is not inertial it can be considered approximately inertial. As demonstrated in [6], the Coriolis and centrifugal accelerations suffered by this frame are negligible.

The choice of representing orientations with respect to an inertial frame means that any error in estimating the orientation of any link doesn't propagate to the next one. It also means that inertial measurements from the IMU will be easier to model as they're independent of the previous link. On the other hand, this will make sensors like encoders, which depend on the orientation of two links, have a more complex model.

The rotation matrix for each link evolves according to the following equation

$$ {}^{i}_{n}R(k+1) = {}^{i}_{n}R(k)R_{\delta_i}(k) \tag{3.2} $$

where $R_{\delta_i}(k)$ is the rotation matrix corresponding to the rotation vector $\delta_i(k)$. $\delta_i(k)$ is the vector around which the i frame rotated from instant k to instant $k+1$. The way the rotation matrix can be acquired from the rotation vector is through matrix exponentiation, which is described in appendix B. The value of $\delta_i(k)$ is given by

$$ \delta_i(k) = T(\omega_i(k) + \varepsilon_{\omega_i}(k)) = T\omega_i(k) + \varepsilon_{\delta_i}(k) \tag{3.3} $$

with T the time interval between instants k and $k+1$, $\omega_i = {}^{i}\omega_{ni}$ is the angular velocity of link i with respect to frame n represented in frame i, and ε the error of the model.

One filter designed to find the orientation of a rigid body using an IMU is the MEKF. As mentioned in 2.4.1, the model of Equation (3.2) must be linearized in order to apply it. With that intent, the rotation matrix ${}^{i}_{n}R$ is rewritten such that the state becomes a vector around an operating point.

$$ {}^{i}_{n}R(k) = {}^{i}_{n}\hat{R}(k)R_{\eta_i}(k) \tag{3.4} $$

In this equation, ${}^{i}_{n}\hat{R}$ is the rotation matrix representing the operating point and R_{η_i} is the rotation around the operating point defined by the rotation vector η_i. As demonstrated in appendix B, R_{η_i} can be approximated by

$$ R_{\eta_i}(k) \approx I_3 + [\eta_i(k)\times] \tag{3.5} $$

Given the previous equation, the following can be deduced

$$
\begin{aligned}
{}_n^iR(k+1) &= {}_n^iR(k)R_{\delta_i}(k) && \text{(3.6a)}\\
\iff {}_n^i\hat{R}(k+1)R_{\eta_i}(k+1) &= {}_n^i\hat{R}(k)R_{\eta_i}(k)R_{\delta_i}(k) && \text{(3.6b)}\\
\iff R_{\eta_i}(k+1) &= \left({}_n^i\hat{R}(k)\hat{R}_{\delta_i}(k)\right)^T {}_n^i\hat{R}(k)R_{\eta_i}(k)R_{\delta_i}(k) && \text{(3.6c)}\\
\iff R_{\eta_i}(k+1) &= \hat{R}_{\delta_i}^T(k)R_{\eta_i}(k)R_{\delta_i}(k) && \text{(3.6d)}\\
\iff I_3+[\eta_i(k+1)\times] &= \hat{R}_{\delta_i}^T(k)R_{\delta_i}(k)+\hat{R}_{\delta_i}^T(k)[\eta_i(k)\times]R_{\delta_i}(k) && \text{(3.6e)}\\
\iff I_3+[\eta_i(k+1)\times] &= R_{\varepsilon_{\delta_i}}(k)+[\hat{R}_{\delta_i}^T(k)\eta_i(k)\times] && \text{(3.6f)}\\
\iff I_3+[\eta_i(k+1)\times] &= I_3+[\varepsilon_{\delta_i}(k)\times]+[\hat{R}_{\delta_i}^T(k)\eta_i(k)\times] && \text{(3.6g)}\\
\iff \eta_i(k+1) &= \hat{R}_{\delta_i}^T(k)\eta_i(k)+\varepsilon_{\delta_i}(k) && \text{(3.6h)}\\
\iff \eta_i(k+1) &= \hat{R}_{\delta_i}^T(k)\eta_i(k)+T\varepsilon_{\omega_i}(k) && \text{(3.6i)}
\end{aligned}
$$

which describes the model for η_i. Note that this model isn't required for the update stage of the MEKF because η_i and ε_{ω_i} are zero-mean variables. This model is only needed to find the matrices required for the filtering stage.

Finally, the angular velocity ω_i is modelled as a *random walk* that is corrected by measurements

$$\omega_i(k+1) = \omega_i(k) + \varepsilon_{\omega_i}(k) \tag{3.7}$$

3.3.1.1 Gyroscope Model

Once properly calibrated, each gyroscope should give a direct measure of one of the state variables (per link). Its measurement model is defined by the following equation

$$\omega_{g_i}(k) = \omega_i(k) + \varepsilon_{\omega_{g_i}}(k) \tag{3.8}$$

which is already linear.

3.3.1.2 Magnetometer Model

When considering a sensor like the magnetometer, all one needs to take into consideration is the orientation of the vector being measured, *i.e.* the magnetism vector. Because this vector exists within a field which is approximately constant for the neighborhood of the manipulator, the direction of the vector in the magnetometer can be considered the same as in the joint. That is $m_{m_i} = {}^im$, where m_{m_i} is the magnetism vector in magnetometer i and ${}^im$ is the magnetism vector in joint i. Given the magnetism vector in frame n (${}^nm$), which should be constant in time and only dependent on the manipulator's location on Earth, the relation between the orientation of the link and the measurement is

$$m_{m_i}(k) = {}_n^iR^T(k)\,{}^nm + \varepsilon_{m_i}(k) \tag{3.9}$$

where ε_{m_i} is the magnetometer measurement noise.

The previous equation can be linearized

$$\begin{aligned} m_{m_i}(k) &\approx R^T_{\eta_i}(k)\,{}^i_n\hat{R}^T(k)\,{}^n m + \varepsilon_{m_i}(k) \\ &= {}^i_n\hat{R}^T(k)\,{}^n m - [\eta_i(k)\times]\,{}^i_n\hat{R}^T(k)\,{}^n m + \varepsilon_{m_i}(k) \\ &= {}^i_n\hat{R}^T(k)\,{}^n m + [{}^i_n\hat{R}^T(k)\,{}^n m\times]\eta_i(k) + \varepsilon_{m_i}(k) \end{aligned} \tag{3.10}$$

where the last equality is obtained through the anti-commutative property of the cross-product ($u \times v = -v \times u$).

Note this model is no longer valid when the magnetometer suffers from interference. In case there are interferences that aren't correctable through calibration, the vector ${}^n m$ is no longer constant.

3.3.1.3 Accelerometer Model

Considering a neglectable acceleration when compared to the acceleration due to gravity, the accelerometer model is very similar to the magnetometer one. In fact, what is being measured is a rotated constant vector ($-{}^n g$). The measured specific force is given by

$$f_{a_i}(k) = -{}^i_n R^T(k)\,{}^n g + \varepsilon_{a_i}(k) \tag{3.11}$$

with ε_{a_i} the accelerometer noise.

Clearly, the condition that the IMU acceleration is negligible is false whenever the manipulator is moving. However, the variance of the accelerometers is larger when compared to the gyroscopes. This will have the effect, when applying the MEKF, of being mostly ignored when there is motion. In fact, the purpose of the accelerometer in this model, much like the purpose of the magnetometer, is to eliminate drift caused by the successive integration of the gyroscope measurements. During stationary moments, drift will be corrected. It is in those moments that the specific force measured should be the symmetric of the gravity vector, making the model valid for the filter used.

The similarity with the magnetometer model makes it simple to find the linear model of the accelerometer

$$f_{a_i}(k) = -{}^i_n\hat{R}^T(k)\,{}^n g - [{}^i_n\hat{R}^T(k)\,{}^n g\times]\eta_i(k) + \varepsilon_{a_i}(k) \tag{3.12}$$

3.3.1.4 Encoder Models

Incremental encoders measure the number of pulses produced in a time interval, which, given the resolution of the encoder, can be translated into the angular velocity of a joint. Therefore, their model is

$$\begin{aligned}
\omega_{ie_i}(k) &= \left({}^{i}\omega_{(i-1)i}(k)\right)_z + \varepsilon_{ie_i}(k) \\
&= \left({}^{i}\omega_{ni}(k) - {}^{i}\omega_{n(i-1)}(k)\right)_z + \varepsilon_{ie_i}(k) \\
&= \left(\omega_i(k) - {}_{i-1}^{i}R^T(k)\omega_{i-1}(k)\right)_z + \varepsilon_{ie_i}(k) \\
&= \left(\omega_i(k) - {}_{n}^{i}R^T(k)\,{}_{n}^{i-1}R(k)\omega_{i-1}(k)\right)_z + \varepsilon_{ie_i}(k)
\end{aligned} \tag{3.13}$$

However, it can be expanded to describe the lack of angular velocity in the other axes (imposed by the joint)

$$\begin{bmatrix} 0 \\ 0 \\ \omega_{ie_i}(k) \end{bmatrix} = \omega_i(k) - {}_{n}^{i}R^T(k)\,{}_{n}^{i-1}R(k)\omega_{i-1}(k) + \begin{bmatrix} \varepsilon_{\omega s_i,x}(k) \\ \varepsilon_{\omega s_i,y}(k) \\ \varepsilon_{ie_i}(k) \end{bmatrix} \tag{3.14}$$

where $\varepsilon_{\omega s_i}$ is the error in angular velocity due to slack between the joints that allows them to slightly rotate along other directions.

3.3.2 Joint Model

The Joint Model describes the pose of the manipulator by the angles ($\theta_i(k)$), angular velocities ($\omega_i(k)$), and angular accelerations ($\alpha_i(k)$) of each joint. These variables define the orientation of each joint relative to its previous joint instead of the n frame. This means that, for this model, the nomenclature for the state is different from that of the Link Model. That is, $\omega_i(k) = {}^{i}\omega_{(i-1)i}(k) \neq {}^{i}\omega_{ni}(k)$ and $\alpha_i(k) = {}^{i}\alpha_{(i-1)i}(k) \neq {}^{i}\alpha_{ni}(k)$.

Given that the angular acceleration $\alpha_i(k)$ is the derivative of the angular velocity $\omega_i(k)$, and that the angular velocity $\omega_i(k)$ is the derivative of the angle $\theta_i(k)$, the model can be immediately written.

$$\theta_i(k+1) = \theta_i(k) + T\omega_i(k) + \frac{T^2}{2}\alpha_i(k) + \varepsilon_{\theta_i}(k) \tag{3.15a}$$

$$\omega_i(k+1) = \omega_i(k) + T\alpha_i(k) + \varepsilon_{\omega_i}(k) \tag{3.15b}$$

$$\alpha_i(k+1) = \alpha_i(k) + \varepsilon_{\alpha_i}(k) \tag{3.15c}$$

In this model, T is once again the time interval between the instants k and $k+1$.

One advantage of this model over the Link Model is that its state vector is much smaller, with only three variables per joint. Furthermore, it's a linear model. However, because the position of a specific point in the manipulator depends on the orientation of all the preceding joints, the error of each joint builds up when estimating that position. Regarding the sensor models, the advantages of this model are the disadvantages of the previous one, and vice versa. Encoders will have a much simpler model as they give direct measurements of the state, whereas inertial measurements will depend on the states of all preceding joints. Lastly, this model can't be used to know the pose of the manipulator with respect to an inertial frame if the relation between the base frame (0) and the *n* frame isn't known. Throughout the rest of this section, the base will be considered fixed relative to frame *n* to simplify the model.

The linearization of the models presented in this section won't be calculated. This is due to the recursive nature of the models of the inertial sensors. This aspect produces complicated formulas which are configuration dependent and difficult to implement and debug. The lack of partial derivatives for those models prohibits the use of the EKF. Instead, the UKF should be used to estimate the state for this model.

3.3.2.1 Encoder Models

Given the nature of the chosen state, the encoder model is very simple

$$\omega_{ie_i}(k) = \omega_i(k) + \varepsilon_{ie_i}(k) \tag{3.16}$$

with $\varepsilon_{ie_i}(k)$ the incremental encoder measurement noise.

3.3.2.2 Magnetometer Model

Under the same assumptions as the Link Model, a magnetometer's measurements can be considered to be constant in the neighborhood of the manipulator. This means that, once again, the measurements can be considered to be obtained in the joints' frames. Given this assumption, the following can be written.

$$\begin{aligned} {}^{i}m(k) &= {}^{\;\;i}_{i-}R^T(k)\, {}^{i-}m(k) \\ &= R^T_{\theta_i}(k)\, {}^{i-}_{i-1}R^T\, {}^{i-1}m(k) \end{aligned} \tag{3.17}$$

Knowing that the measurements are noisy

$$m_{m_i}(k) = {}^{i}m(k) + \varepsilon_{m_i}(k) \tag{3.18}$$

This equation, along with the previous one, define a recursive model that can be applied as long as the magnetism vector is known in any specific link. For example, if the magnetism vector in the base (link 0) is known, the measurement from the magnetometer in link 2 is given by

$$\begin{aligned} m_{m_2}(k) &= R^T_{\theta_2}(k)\, {}^{2-}_{\;\;1}R^T\, {}^{1}m(k) + \varepsilon_{m_2}(k) \\ &= R^T_{\theta_2}(k)\, {}^{2-}_{\;\;1}R^T R^T_{\theta_1}(k)\, {}^{1-}_{\;\;0}R^T\, {}^{0}m + \varepsilon_{m_2}(k) \end{aligned} \tag{3.19}$$

3.3.2.3 Gyroscope Model

The angular velocity measured by a gyroscope is always done with respect to an inertial frame. However, given the relation

$$\begin{aligned} {}^{i}\omega_{ni}(k) &= {}^{i}\omega_{n(i-1)}(k) + {}^{i}\omega_{(i-1)i}(k) \\ &= R^T_{\theta_i}(k)\, {}^{i-}_{i-1}R^T\, {}^{i-1}\omega_{n(i-1)}(k) + {}^{i}\omega_{(i-1)i}(k) \end{aligned} \tag{3.20}$$

the gyroscope measurements can be written as a function of the angular velocity of the joint (${}^{i}\omega_{(i-1)i}$), the angle of the joint (which affects $R^T_{\theta_i}$) and the angular velocity of the previous link with respect to the inertial frame (${}^{i-1}\omega_{n(i-1)}$). With this relation, the gyroscope model can be deduced.

$$\omega_{g_i}(k) = {}^{i}\omega_{ni}(k) + \varepsilon_{g_i}(k) \tag{3.21}$$

with $\varepsilon_{g_i}(k)$ the gyroscope measurement noise and ${}^{i}\omega_{(i-1)i}(k) = [0 \quad 0 \quad \omega_i(k)]^T$.

3.3.2.4 Accelerometer Model

From appendix A, the acceleration of a joint can be written as a function of the acceleration of the previous joint

$${}^{n}a_i(k) = {}^{i-1}_{n}R(k)\,{}^{i-1}\Omega_{n(i-1)}(k)\,{}^{i-1}t_i + {}^{n}a_{i-1}(k) \tag{3.22}$$

with ${}^{i-1}t_i$ the translation vector from joint $i-1$ to joint i in the $i-1$ frame.

From this relation, the specific force in a joint can be determined

$$\begin{aligned}{}^{n}f_i(k) &= {}^{n}a_i(k) - {}^{n}g \\ &= {}^{i-1}_{n}R(k)\,{}^{i-1}\Omega_{n(i-1)}(k)\,{}^{i-1}t_i + {}^{n}a_{i-1}(k) - {}^{n}g \\ &= {}^{i-1}_{n}R(k)\,{}^{i-1}\Omega_{n(i-1)}(k)\,{}^{i-1}t_i + {}^{n}f_{i-1}(k)\end{aligned} \tag{3.23}$$

Rotating it to the joint's frame

$$\begin{aligned}{}^{i}f_i(k) &= {}^{i}_{n}R^T(k)\left({}^{i-1}_{n}R(k)\,{}^{i-1}\Omega_{n(i-1)}(k)\,{}^{i-1}t_i + {}^{n}f_{i-1}(k)\right) \\ &= {}^{i}_{i-1}R^T(k)\,{}^{i-1}\Omega_{n(i-1)}(k)\,{}^{i-1}t_i + {}^{i}f_{i-1}(k) \\ &= {}^{i}_{i-1}R^T(k)\left({}^{i-1}\Omega_{n(i-1)}(k)\,{}^{i-1}t_i + {}^{i-1}f_{i-1}(k)\right) \\ &= R^T_{\theta_i}(k)\,{}^{i-}_{i-1}R^T\left({}^{i-1}\Omega_{n(i-1)}(k)\,{}^{i-1}t_i + {}^{i-1}f_{i-1}(k)\right)\end{aligned} \tag{3.24}$$

The matrix ${}^{i-1}\Omega_{n(i-1)}(k)$ is a function of the angular velocity and angular acceleration of frame $i-1$ with respect to frame n. However, the state is composed of variables that describe the angular velocity and angular acceleration of the joints. In order to write the accelerometer measurement as a function of the state variables, the angular velocity and angular acceleration are rewritten

$${}^{i}\alpha_{ni}(k) = R^T_{\theta_i}(k)\,{}^{i-}_{i-1}R^T\,{}^{i-1}\alpha_{n(i-1)}(k) + {}^{i}\alpha_{(i-1)i}(k) \tag{3.25}$$

$${}^{i}\omega_{ni}(k) = R^T_{\theta_i}(k)\,{}^{i-}_{i-1}R^T\,{}^{i-1}\omega_{n(i-1)}(k) + {}^{i}\omega_{(i-1)i}(k) \tag{3.26}$$

with ${}^{i}\omega_{(i-1)i}(k) = [0 \quad 0 \quad \omega_i(k)]^T$ and ${}^{i}\alpha_{(i-1)i}(k) = [0 \quad 0 \quad \alpha_i(k)]^T$. Note that the influence of the angular acceleration in this model is what created the necessity for it to be included in the state vector.

With these three recursive expressions, the accelerometer can now be modeled.

$$f_{a_i}(k) = {}^{i}\Omega_{ni}(k)\,{}^{i}p_{a_i} + {}^{i}f_i(k) + \varepsilon_{a_i}(k) \tag{3.27}$$

with ${}^{i}p_{a_i}$ the position of the accelerometer in the link and $\varepsilon_{a_i}(k)$ the accelerometer measurement noise.

3.3.3 Application of the Models to the Manipulator

The previously presented models are symbolic and serve for all manipulators that follow the constraints initially mentioned. To make practical use of these models, the symbolic variables must be replaced by the appropriate constants for a specific manipulator. The one used to test the estimators has 3 degrees of freedom which allow it to position its tip in any point in space as long as it's within range. It is equipped with one incremental encoder in each joint, one IMU in link 2, and one IMU in link 3.

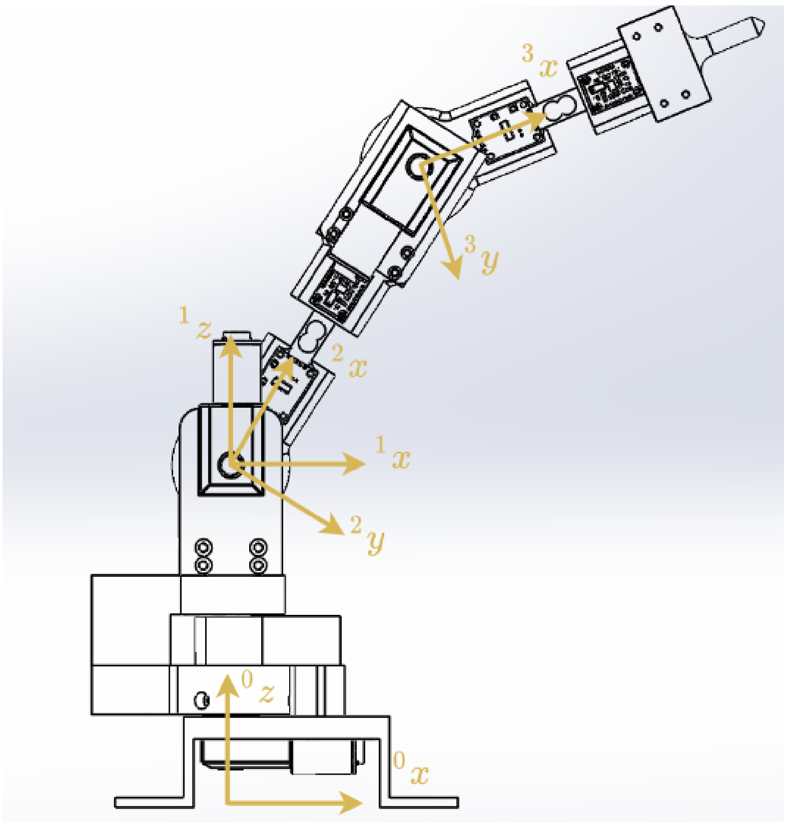


Figure 3.5: Manipulator sketch with coordinate frames

The manipulator is represented in the figure along with the frames of each joint. From this representation, the constant rotation matrices can be found. Frames 0 and $1-$ are aligned, and so are frames 2 and $3-$. This means the rotation matrices between these frames are the identity matrix

$${}^{1-}_{0}R = {}^{3-}_{2}R = I_3 \tag{3.28}$$

In fact, if all joints have their angle at zero, the only referentials which are misaligned are 1 and $2-$. Using the DH method, the rotation matrix ${}^{2-}_{1}R$ is the result of a rotation of -90° around z followed by a rotation of -90° around x.

$$ {}^{2-}_{1}R = R_{-90^\circ x} R_{-90^\circ z} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \tag{3.29} $$

The translation vectors between the frames were measured with a ruler as precisely as possible. The position of the manipulator's tip in frame 3 (${}^{3}t_{tip}$) was also determined so it could later be used to find its absolute position. The following measurements are all in millimeters.

$$ {}^{0}t_1 = \begin{bmatrix} 0 \\ 0 \\ 174.5 \end{bmatrix} \qquad {}^{1}t_2 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \qquad {}^{2}t_3 = \begin{bmatrix} 180.5 \\ 0 \\ 0 \end{bmatrix} \qquad {}^{3}t_{tip} = \begin{bmatrix} 190.5 \\ 0 \\ 0 \end{bmatrix} \tag{3.30} $$

Note that the transformation between frames 1 and 2 is purely a rotation, which implies that vector ${}^{1}t_2$ is a null vector.

The positions of the accelerometers with respect to their corresponding joints are some of the constants in the accelerometers models. These were found by measuring the position of the IMUs with a ruler and adding the offset of the MEMS device present in the datasheet.

$$ {}^{2}p_{a_2} = {}^{3}p_{a_3} = \begin{bmatrix} 94 \\ -1 \\ 11 \end{bmatrix} \tag{3.31} $$

Finally, the last required constants are the time interval of the filter (T), which has a value of $40ms$, the gravity vector in the navigation frame n, and the magnetism vector in the same frame. Assuming the base frame (0) is the same as the navigation frame (n) and that this frame follows the North-West-Up convention, the gravity vector is ${}^{n}g = [0 \quad 0 \quad -9.81]^T$. The magnetic field vector in the navigation frame was later determined using the onboard magnetometers.

Having this information, the presented models can now be used.

Chapter 4

Inertial Measurement Units Calibration

The proper use of any system assumes its sensors are correctly calibrated. The methods chosen to perform the calibration of the manipulator's sensors are presented here. The sensors used by the estimators are the incremental encoders and the IMUs. The only calibration parameter the encoders have is the relation between the number of pulses measured per revolution of the motor. This is a constant and can be found in the motor's datasheet. Calibrating the sensors of an IMU is not as simple, being the main focus of this chapter.

As discussed in 2.1, many methods for estimating the required parameters rely on a robotically actuated frame to which the IMU is attached. Even though the IMU is fixed to the manipulator, these methods are not appropriate for calibration in this context. Usually those methods assume that the movements to which the IMU is subjected are precise. However, the manipulator is not assumed to have that capability. The problem relates in part with the fact that the 3D printed plastic pieces tend to dent as the motors strain the articulations. These dents create slacks which cause the position of the motor to not have a direct relation with the angle at which each articulation is. Even if there were no slacks, finding the angle of each motor is a problem in its own. In fact, that is the goal of an estimator. Given that the estimators depend on correctly calibrated sensors, a cyclic problem emerges if this approach is attempted.

Instead, the proposed methods do not assume precise movements. The only assumption made from the manipulator is that it is capable of causing sufficiently diverse movements. The definition of "sufficiently diverse movements" is given later as it relates individually to each of the methods chosen. The way these movements are achieved is by dead-reckoning the measurements of the encoders to estimate the angles of each joint. This estimate is not accurate but is sufficient to drive the motors, and therefore the joints, to a large number of decisively different angles.

This chapter presents the implementation of the different calibration techniques, as well as their results. The implementation of all of the mentioned algorithms was done within the application described in section 3.2. The following sections are separated by the different triads of sensors that compose an IMU. Section 4.1 describes the calibration process for the accelerometers, section 4.2 for the magnetometers and section 4.3 for the gyroscopes.

Before describing the algorithms, the different parameters that the measurements depend on are presented. Generally, each triad of sensors follows the model

$$v = Gm + B + \varepsilon \tag{4.1}$$

where, m is the raw measurement from the sensor, G is the gain matrix, B is the bias vector, ε corresponds to noise from the measurements and v is the real value of the variable being measured. The bias is chosen so that ε has zero mean. This way, the expected value of each calibrated measurement doesn't have a systematic error. The gain matrix incorporates any faults that can cause linear errors, including scaling factors, misalignments between the sensors' axes and misalignments between the sensor frame and the reference frame [10]. This means that there is no redundancy in the elements of the matrix, as each entry can not be given as some combination of the other entries. In other words, the matrix G has 9 degrees of freedom.

4.1 Accelerometer Calibration

4.1.1 Parameter Estimation Algorithm

The accelerometers are calibrated under the assumption that the IMU is static. When this is the case, the vector being measured is the symmetric of the acceleration due to gravity $(-g)$ rotated

$$g' = -R_a g \tag{4.2}$$

Because the device's orientation is unknown, then so is the vector in the frame of the IMU. However, regardless of the orientation of the IMU, the vector should have the same magnitude. This property can be used to write an attitude-independent equation through which the calibration parameters can be estimated [16]

$$g'^2 = \|v\|^2 = \|Gm + B\|^2 \tag{4.3}$$

Because this equation is attitude-independent, the parameters G and B can only be determined up to a rotation matrix. To better understand this aspect, consider there is a solution to the calibration problem (G_1, B_1). If these matrices where to be left multiplied by a rotation matrix, they would produce another valid solution $(G_2, B_2) = (RG_1, RB_1)$ since all rotation matrices have determinant 1. The only difference between both solutions is that they measure the same quantity in different frames. The lack of a reference frame means that Equation (4.3) is valid for an infinite set of solutions.

A frame must be chosen in order to arrive at a specific solution. The frame is chosen to have its x-axis coincident with the x-axis of the frame of vector m (the original frame). The y-axis is chosen to be on the same plane as the xy plane of the original frame, orthogonal to the x-axis and facing the same side as the y-axis of the original frame. Note that this is not the same as saying the y-axes of both frames are coincident, as there is no guarantee that the original frame has orthogonal axes.

The z-axis is given by the right hand rule. This set of assumptions regarding the chosen frame imply that the G matrix is lower triangular with positive entries at the main diagonal.

The G matrix corresponding to this frame only has 6 non-zero entries, which correspond to its 6 DOF. In fact, considering that this matrix is determined up to a rotation matrix is equivalent to saying it has 3 DOF less than a general 3x3 matrix, which has 9 DOF. This is true because any rotation can be specified by 3 independent variables, and therefore has 3 DOF.

Given the structure of matrices G and B

$$G = \begin{bmatrix} G_{xx} & 0 & 0 \\ G_{yx} & G_{yy} & 0 \\ G_{zx} & G_{zy} & G_{zz} \end{bmatrix} \qquad B = \begin{bmatrix} B_x \\ B_y \\ B_z \end{bmatrix} \tag{4.4}$$

Equation (4.3) can be expanded

$$\begin{aligned} \|Gm+B\|^2 &= (G_{xx}m_x+B_x)^2+(G_{yx}m_x+G_{yy}m_y+B_y)^2+(G_{zx}m_x+G_{zy}m_y+G_{zz}m_z+B_z)^2 \\ &= G_{xx}^2m_x^2+B_x^2+2G_{xx}m_xB_x \\ &+G_{yx}^2m_x^2+G_{yy}^2m_y^2+B_y^2+2G_{yx}G_{yy}m_xm_y+2G_{yx}m_xB_y+2G_{yy}m_yB_y \\ &+G_{zx}^2m_x^2+G_{zy}^2m_y^2+G_{zz}^2m_z^2+B_z^2 \\ &+2G_{zx}G_{zy}m_xm_y+2G_{zx}G_{zz}m_xm_z+2G_{zy}G_{zz}m_ym_z \\ &+2G_{zx}m_xB_z+2G_{zy}m_yB_z+2G_{zz}m_zB_z \\ &= \left(G_{xx}^2+G_{zy}^2+G_{zx}^2\right)m_x^2+\left(G_{yy}^2+G_{zy}^2\right)m_y^2+G_{zz}^2m_z^2+B_x^2+B_y^2+B_z^2 \\ &+(G_{yx}G_{yy}+G_{zx}G_{zy})\,2m_xm_y+G_{zx}G_{zz}2m_xm_z+G_{zy}G_{zz}2m_ym_z \\ &+(G_{xx}B_x+G_{yx}B_y+G_{zx}B_z)\,2m_x+(G_{yy}B_y+G_{zy}B_z)\,2m_y+G_{zz}B_z2m_z \end{aligned} \tag{4.5}$$

The length of the measured vector can be considered 1 without loss of generality. It just implies that the resulting matrices should be multiplied by $\|g\|$ in order to obtain measurements in m/s^2. With this simplification, the previous equation can be rewritten as

$$\begin{aligned} -G_{zz}^2m_z^2 &= \left(G_{xx}^2+G_{zy}^2+G_{zx}^2\right)m_x^2+\left(G_{yy}^2+G_{zy}^2\right)m_y^2+B_x^2+B_y^2+B_z^2-1 \\ &+(G_{yx}G_{yy}+G_{zx}G_{zy})\,2m_xm_y+G_{zx}G_{zz}2m_xm_z+G_{zy}G_{zz}2m_ym_z \\ &+(G_{xx}B_x+G_{yx}B_y+G_{zx}B_z)\,2m_x+(G_{yy}B_y+G_{zy}B_z)\,2m_y+G_{zz}B_z2m_z \end{aligned} \tag{4.6}$$

which can be simplified to

$$\begin{aligned} -m_z^2 &= \left({G'_{xx}}^2+{G'_{zy}}^2+{G'_{zx}}^2\right)m_x^2+\left({G'_{yy}}^2+{G'_{zy}}^2\right)m_y^2+{B'_x}^2+{B'_y}^2+{B'_z}^2-1/G_{zz}^2 \\ &+\left(G'_{yx}G'_{yy}+G'_{zx}G'_{zy}\right)2m_xm_y+G'_{zx}2m_xm_z+G'_{zy}2m_ym_z \\ &+\left(G'_{xx}B'_x+G'_{yx}B'_y+G'_{zx}B'_z\right)2m_x+\left(G'_{yy}B'_y+G'_{zy}B'_z\right)2m_y+B'_z2m_z \end{aligned} \tag{4.7}$$

where $\alpha' = \alpha/G_{zz}$ In this form, it is evident that the equation can be solved using least squares

$$y = Fx + \varepsilon \qquad y_i = -m_z^2$$

$$F_i = \begin{bmatrix} m_x^2 \\ m_y^2 \\ 1 \\ 2m_x m_y \\ 2m_x m_z \\ 2m_y m_z \\ 2m_x \\ 2m_y \\ 2m_z \end{bmatrix}^T \qquad x = \begin{bmatrix} {G'_{xx}}^2 + {G'_{zy}}^2 + {G'_{zx}}^2 \\ {G'_{yy}}^2 + {G'_{zy}}^2 \\ {B'_x}^2 + {B'_y}^2 + {B'_z}^2 - 1/G_{zz}^2 \\ G'_{yx}G'_{yy} + G'_{zx}G'_{zy} \\ G'_{zx} \\ G'_{zy} \\ G'_{xx}B'_x + G'_{yx}B'_y + G'_{zx}B'_z \\ G'_{yy}B'_y + G'_{zy}B'_z \\ B'_z \end{bmatrix} \tag{4.8}$$

to which the best estimate of the x vector is given by

$$\hat{x} = \left(F^T F\right)^{-1} F^T y \tag{4.9}$$

The inverse in the previous equation only exists if the matrix F has full column rank. This is only possible if there are enough linearly independent rows in the matrix. Therefore, in order to find the solution, the IMU must be oriented in a sufficient number of ways. In this case, the x vector has 9 variables, which means that at least 9 orientations are required. These orientations should not be confined to a rotation in the same plane, otherwise the rows of F won't be linearly independent.

After solving the least squares problem, finding the entries of the matrices G and B is a matter of algebraic manipulation of the variables of x. When solving for the calibration parameters, it is necessary to remember the fact that the entries of the main diagonal of G are positive.

Note that each row of F (and the corresponding value in y) already contains all the variables of the quadratic vector of m. This means the F matrix already has as many columns as possible, which in turn limits the number of independent variables that can be estimated with it, *i.e.* the length of vector x. Considering x has 9 variables and that G and B are composed of 9 variables in total, it is possible to solve algebraically for those parameters. However, if matrix G was not initially considered to have $G_{xy} = G_{xz} = G_{yz} = 0$, then the number of parameters would be 12 and there would be an infinite amount of solutions for the equations given by the solution of x. Once again it is demonstrated the need to restrict the number of possible solutions due to the lack of a reference frame.

Finally, it should be noted that the lack of an external reference frame makes it impossible for the method to calibrate the sensors so as to have the IMU frame coincident with the frame of the link to which it is attached. Only an approximate solution was determined. The IMU attached to link 3 was assumed to be aligned, simply due to the design of the manipulator. Only a 180° rotation around the y-axis was used to make sure the axes were pointing roughly the same way. The other IMU is not aligned with its link, which means that even this assumption is too faulty for that case. Alignment of this IMU was done by setting the angle in the third joint to as close to zero as visibly possible before performing the calibration movements. This meant measured vector should be the same in both IMUs. Since they only differed from each other by a rotation matrix, finding this matrix was the way to align the frames. Finding a rotation matrix that relates sets of vectors is known as Wahba's problem [19]. Many solutions exist to this problem. The one chosen is given by the Singular Value Decomposition (SVD) of a matrix composed of the measured vectors [9].

4.1.2 Implementation

A set of positions was defined to satisfy the constraint of variability of orientations necessary to calibrate the accelerometers. Considering the placement of the IMUs in the manipulator, only the first two joints were moved. Moving the third joint would not cause a change in orientation for the IMU in link 2. Furthermore, it would change the orientation of the IMU in link 3 in a way that could be achieved by moving the second joint instead. This is possible since the rotation axes of the second and third joints are parallel.

In order to provide enough variability to the measurements, the base of the manipulator could not be fixed horizontally. This would cause the z-axis of the manipulator to not measure the g' vector, since any choice for the angles of the first two joints leaves the z-axis in the horizontal plane. Experiments were performed with the base fixed horizontally. This caused the z-axis of the triad to measure mostly noise and this could be such that the least squares solution would output negative variables where they were not allowed. For example, when determining the value of G_{zz}, first the value of G_{zz}^2 is found, and then the square root calculated. This produces an invalid result if G_{zz}^2 is negative. In the cases when the noise produced valid results, the calculated variance of the z-axis was about two orders of magnitude higher than the variance of the other axes. Tilting the base proved to be an effective solution to this problem.

The defined positions were automatically generated to form half of the surface of a sphere. This was done by sweeping the angles of the first two joints in a grid like the one showed in Figure 4.1. The size of the grid in the figure does not reflect the size of the grid used. When collecting measurements, the grid had 10x10 points, which means the least squares matrix F had 100 rows. In each point of the grid, the manipulator was stationary and collected 50 measurements. Each group of 50 measurements was averaged into one vector per position in the grid. This vector was then used as the input to the least squares problem. Averaging was done to remove as much noise from the measurements as possible, which means the least squares solution was obtained with more accurate values. Having the IMU stand still is a necessity to assure that the measured vector is g' and that it is not being affected by any acceleration.

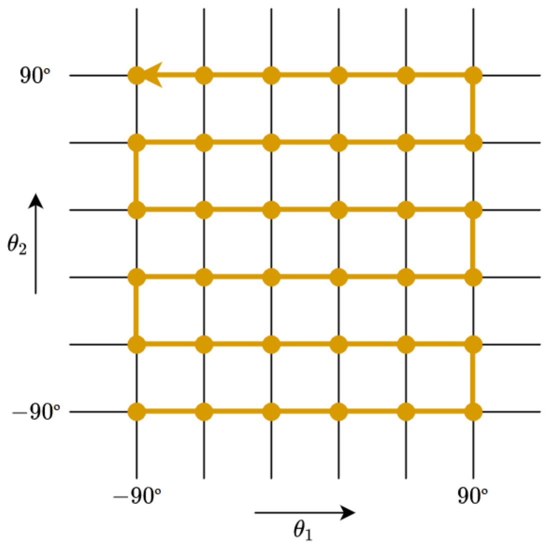


Figure 4.1: Grid of the positions of each joint during accelerometer calibration

4.1.3 Results and Analysis

To verify the validity of the model and of the calibration method, the movements were repeated 10 times resulting in 10 datasets of measurements. From these datasets, the mean parameters and their standard deviation were determined. These are presented in Table 4.1, where IMU A refers to the IMU of link 2 and IMU B to the one of link 3.

	IMU A		IMU B	
	Mean	Standard Deviation	Mean	Standard Deviation
G_{xx}	1.00E-4	1.59E-7	2.43E-4	2.03E-7
G_{yx}	-4.95E-7	2.87E-7	1.23E-6	7.86E-8
G_{yy}	1.00E-4	4.90E-7	2.47E-4	3.52E-8
G_{zx}	3.17E-6	1.68E-7	-6.91E-7	1.01E-7
G_{zy}	1.74E-6	5.22E-7	1.36E-6	5.74E-8
G_{zz}	9.96E-4	6.46E-7	2.47E-4	3.67E-7
B_x	2.44E-3	2.24E-4	-1.42E-2	7.28E-4
B_y	1.05E-2	4.81E-4	4.10E-2	5.63E-5
B_z	8.71E-2	1.34E-3	-4.47E-2	2.95E-4

Table 4.1: Mean and Standard Deviation of the Accelerometers parameters obtained from 10 datasets

As expected, the predominant entries of the gain matrix are a part of the main diagonal. The low values at the non-diagonal entries imply that the axes are mostly aligned with each other, which is expected from MEMS sensors when each triad is within the same integrated circuit.

The standard deviation of all the entries of G is about one order of magnitude lower than the value of the non-diagonal entries. The standard deviation of the bias terms is significantly higher than the gain ones. However, it is still about two orders of magnitude lower than the values of the bias. This implies the method is repeatable. Therefore, slight variations to the movements of the manipulator do not visibly modify the results.

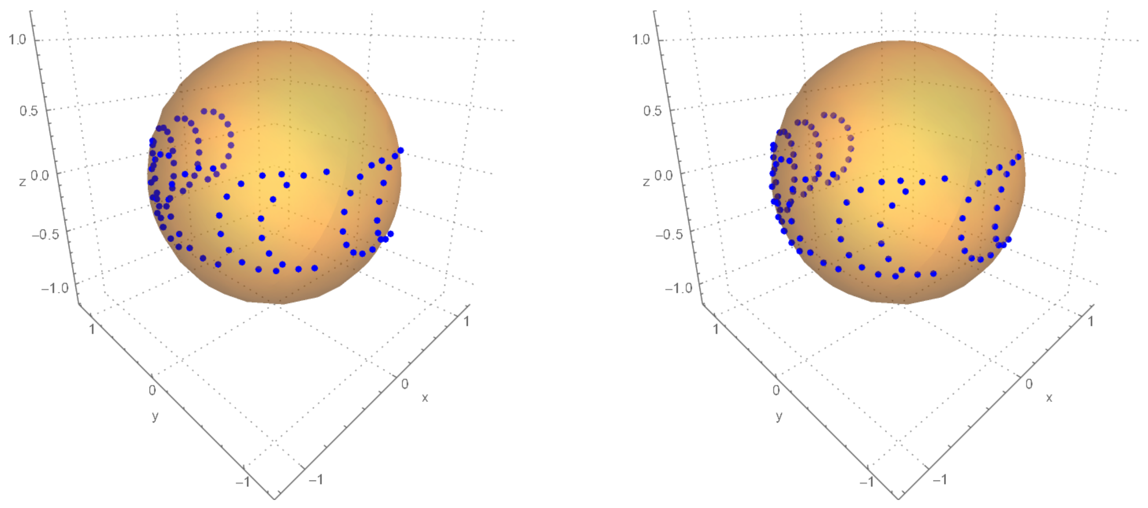
To inspect the correctness of the method, the maximum absolute errors were examined. The error in this context is the norm of a measurement subtracted by the expected norm ($1g$). For comparison, the errors of the pre-calibration measurements were also included. However, these measurements needed to be scaled in order to be comparable. The errors are presented in Table 4.2.

	IMU A	IMU B
pre-calibration	3.07E-2	5.56E-2
post-calibration	1.36E-3	2.94E-3

Table 4.2: Maximum Absolute Errors of the magnitude of each accelerometer measurement across the 10 datasets

In both IMUs, the error was reduced to the point where the norm of the g' vector did not change by more than $3mg$, or $0.0294 {}^{m}/{}_{s^2}$. Note that each of the measurements used for calculating these errors was the average of 50 other measurements, which means that the error could be higher due to noise. However, these results are enough to validate the model given its inability to correct for noise.

The measurements from one dataset were plotted to verify that the model was corresponding to the expectations of g' as the manipulator followed the movements described in Figure 4.1. These plots are shown in Figure 4.2 for IMU B.



(a) Pre-calibration Scaled Measurements (b) Post-calibration Measurements

Figure 4.2: 3D plot of the measurements of a dataset from accelerometer B before and after calibration

In the same plots there is a centered sphere with unit radius to help visualize whether the measurements have the correct norm. The pre-calibration measurements do not lie on the sphere. These are mostly shifted from the center and do not present any significant distortions. This data is congruent with Table 4.1 since the non-diagonal values of G, which should manifest as a distortion, are much smaller than the bias terms, which manifest as an offset.

Besides the comparison between the two plots, the post-calibration measurements can be analysed to verify that they make sense considering the performed movements. The most prevailing characteristic of the measurements is that they are limited in the z-axis. Recall the need to tilt the base of the manipulator to have the z-axis of the accelerometers excited by the g' vector. Without this tilt, the z-axis was confined to the horizontal plane. After tilting the base, the z-axis is still limited to a plane, however, this plane is tilted and therefore it can measure the corresponding projection of g'. The maximum value that this projection can take is given by the angle at which the base is inclined. Therefore, if the base was tilted by 90°, the projection could, at most, have unitary value, at which point the sphere could be fully covered. On the other hand, if there was no tilt, the projection of g' into the z-axis would always be 0 and the measurements would trace a circle in the xy plane. This relation between the limit of the projection and the tilt angle can be used to find the angle at which the base is tilted. This was not done since it was unnecessary, however it can be helpful in cases when it is important to find the orientation of the base.

A deeper analysis of the plot reveals that the measurements form arcs. These have their centers starting at $[0 \quad 1 \quad 0]^T$, followed by $[-1 \quad 0 \quad 0]^T$ and ending at $[0 \quad -1 \quad 0]^T$. Consider the case when $\theta_2 = -90°$ and θ_1 sweeps from $-90°$ to $90°$. Picturing the manipulator's configuration during this movement, it is evident that the y-axis always points the same way for a given θ_2 and therefore always measures the same amount of g'. However, as θ_1 changes, the x and z axes rotate around the y-axis. Therefore, the remaining projection should be exchanged between them. If θ_1 performed a full rotation, the x and z projections should form a circumference. Given that θ_1 only sweeps 180°, a semi-circumference is expected. That corresponds to the arc centered at $[0 \quad 1 \quad 0]^T$. This relation can be extended to the other arcs. There is an axis, which is the combination of the x and y axes, that always captures the same projection of g' for the same value of θ_2. That axis is perpendicular to the plane of each arc, since its projection remains constant. The axes perpendicular to that one share the remaining projection as θ_1 changes. The limitation of the θ_2 angle means that the manipulator does not point to the downward half, which causes the arcs to only cover half of the unit sphere's surface.

Note the radius of the arcs is directly related to the limitation of the z-axis. If the arcs had radius 1, then the whole sphere's surface would be covered in measurements. And if the radius was 0, the measurements would be confined to the points on the xy plane. Again, the tilt angle of the base of the manipulator can be found by analysing the radius of the arcs.

This analysis proves that the visualized measurements correspond to the expected ones. This means that the model is not obtaining good results in an unexpected manner, and is actually representing correctly the behavior of the accelerometers.

4.2 Magnetometer Calibration

4.2.1 Parameter Estimation Algorithm

The magnetometers, despite measuring a completely different quantity, share a lot of mathematical properties with the accelerometers. For example, it can also be modeled linearly by Equation (4.1). The magnetometers are also affected by magnetic disturbances sometimes referred to as *soft iron* and *hard iron* distortions [10, 16]. However, if these are constant, they are also accounted for in the model. Soft iron disturbances influence the gain matrix G whereas hard iron ones influence the bias vector b.

Another similarity shared with the accelerometers is the mathematical nature of what it measures, since what is being measured is a rotated constant vector. This means that the method used for the accelerometers should be equally valid for the magnetometers. In fact, the constraint that the manipulator is stationary is not necessary in this case.

Using this method a second time implies that the triads of accelerometers and magnetometers are calibrated independently. Recall that the misalignment between the accelerometers and the corresponding links was only approximated visually. If the same approximate rotation is used to correct for the magnetometers' misalignments, the frames of the accelerometers and the magnetometers won't be the same. In [10], this problem is solved independently for each triad by finding the rotation matrices between these frames and the gyroscope frame, making the gyroscope frame the default one from which all others are calibrated. An alternative, demonstrated in [8], uses the property of invariance of the dot product of the two vectors in the inertial frame. That is, $-g$ and v_m (the real value of the magnetic field) are constant and therefore have a constant dot product. After rotation of both vectors from the inertial frame to the IMU frame, the dot product should remain the same, since the norm of each vector remained constant as well as the angle between them. This method uses the calibrated accelerometers' measurements as a ground truth. Therefore, the frame to which the magnetometers are calibrated is the same. This method also assumes that the accelerometer measurements are read in the same moments as the magnetometer measurements, which reintroduces the need to read them while stationary.

The invariance of the dot product between both vectors can be mathematically described by

$$g' \cdot v_m = K \tag{4.10}$$

Rewriting the dot product in matrix form and expanding v_m as a function of the calibration parameters and the magnetometer measurements m_m results in

$$\begin{aligned} K &= g'^T v_m \\ &= g'^T (G m_m + B) \\ &= g'^T G m_m + g'^T B \\ &= \left(m_m^T \otimes g'^T \right) vec(G) + g'^T B \end{aligned} \tag{4.11}$$

where $\otimes$ represents the Kroenecker product and $vec(G)$ is the vector form of matrix G in which its columns are stacked into one vector with 9 variables. Since K is originally unknown, the previous expression needs to be rewritten in order to have its variables determinable by least squares

$$1 = \begin{bmatrix} m_m^T \otimes g'^T & g'^T \end{bmatrix} \begin{bmatrix} \frac{1}{K} vec(G) \\ \frac{1}{K} B \end{bmatrix} \tag{4.12}$$

In order to find G and B from $\frac{1}{K}G$ and $\frac{1}{K}B$ the magnitude of v_m must be known. However, the magnitude can be set to any value assuming that only the direction of the vector is relevant. For simplicity, the predefined magnitude is 1. If $\frac{1}{K}G$ and $\frac{1}{K}B$ are used as the gain and bias of the magnetometer triad, the measurements from calibration can be corrected to a scaled version of the real value of the magnetic field ($\frac{1}{K}v_m$). Specifically, the scale factor is $\frac{1}{K}$ and can be determined by averaging the norm of all measurements. Once this factor is calculated, G and B can be corrected. Furthermore, if the accelerometer measurements used are normalized, it is evident from Equation (4.10) that

$$K = \cos\phi \tag{4.13}$$

where ϕ is the angle between the g' and v_m measurements.

4.2.2 Implementation

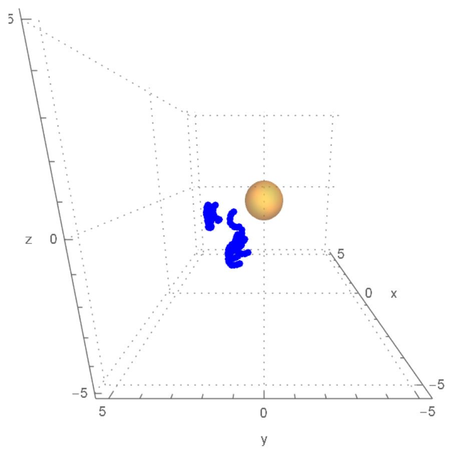
Given the criteria similarity with the accelerometers calibration, the movements performed for data collection are exactly the same. Only one restriction is different for the magnetometers. The number of parameters to be estimated is 12 (9 entries of G + 3 entries of B). This means that the F matrix from the least squares problem must have 12 rows, and consequently, the minimum number of different positions is 12. Since the accelerometers already collected measurements in 100 positions, the same movements are valid for the magnetometers. Once again, 50 measurements were collected from each position and averaged before being used in the least squares solution.

Just as with the accelerometers, the restriction that the z-axis of the IMUs is excited by the magnetic field vector is a requirement. However, this one is more easily achievable since failing to do so implied the base was tilted by the right angle along the right direction.

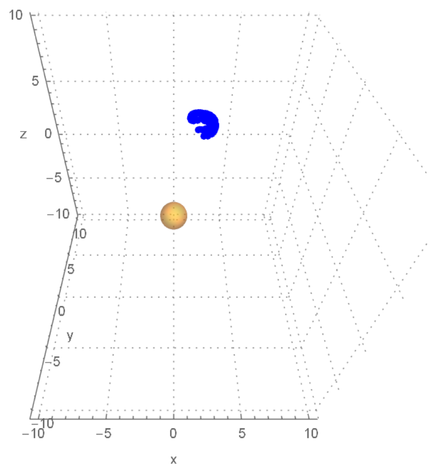
4.2.3 Results and Analysis

Before analysing the results of the calibration procedures, the pre-calibration measurements are verified to make sure the proposed linear model is capable of the necessary corrections. These were previously scaled to help match the size of the unit sphere. The plot with these measurements is represented in Figure 4.3.

The measurements from IMU B appear to suffer from a bias, much larger than the accelerometers. This is probably due to the hard iron disturbances surrounding the manipulator. Overall, the shape described by the measurements seems to fit the unit sphere if these are corrected by the model used.



(a) IMU A



(b) IMU B

Figure 4.3: 3D plot of the measurements of a dataset from both magnetometers before calibration

On the other hand, the measurements from IMU A appear to be more distorted. The pattern shown still displays arcs, which hints at the fact that these extra distortions are independent from θ_1. However, this distortion is probably not correctable by a constant matrix G. Linear and constant distortions, which G and B are able to correct, should only cause the pattern of measurements to cover an ellipsoid. The pattern shown in the figure does not correspond to that shape, which means this model is probably insufficient. In fact, it is obvious the reason for the model to be inaccurate. One of the assumptions was that the parameters G and B are constant. However, the hard iron distortions on the measurements depend on the proximity of the sensors to the materials that cause these distortions. It is evident from the design of the manipulator that IMU A is very close to such materials. The motors and their respective encoders cause a change in the surrounding magnetic field. These changes are not constant since the proximity of these elements to the IMU is dependant on the angle of the joints. Since these angles are not known, compensating for these disturbances is a difficult task. Ultimately, these disturbances were not compensated for, which left IMU A unusable. Nevertheless, the proposed calibration algorithms were still applied in an attempt to verify whether proper compensation could be achieved simply.

The first attempted calibration method was the same as the one used for the accelerometers. Applying it to IMU A resulted in invalid results. However, the application to IMU B worked and produced the measurements plotted in Figure 4.4 for one dataset and the metrics of Tables 4.3 and 4.4 for all datasets.

Figure 4.4 displays a plot similar to the one presented in Figure 4.2. The measurements follow the same logic and the model acts as expected.

Table 4.3 shows the mean and standard deviation of the calibration parameters for magnetometers B. Comparing it to Table 4.1, two main differences are noted. First, the overall order of magnitude of all metrics is higher. This is simply a consequence of the order of magnitude of the quantity being measured by the sensors. Second, the standard deviation of the G matrix entries

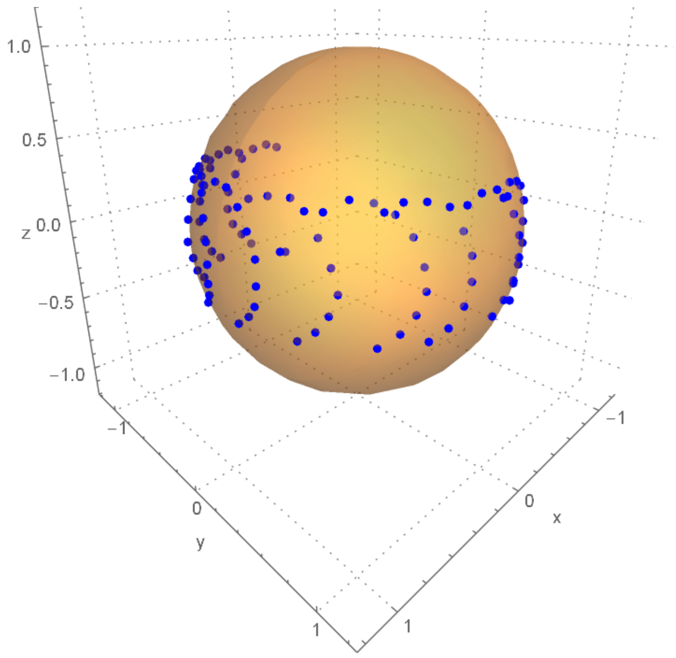


Figure 4.4: 3D plot of measurements of a dataset from magnetometers B after calibration with the first method

	Mean	Standard Deviation
G_{xx}	2.52E-3	1.31E-5
G_{yx}	3.74E-5	1.52E-5
G_{yy}	2.55E-3	9.45E-6
G_{zx}	2.45E-5	2.11E-5
G_{zy}	6.00E-5	2.05E-5
G_{zz}	3.15E-3	3.65E-5
B_x	-1.62	1.32E-2
B_y	-1.93	1.64E-2
B_z	-7.41	1.09E-1

Table 4.3: Mean and Standard Deviation of the Magnetometers parameters obtained from 10 datasets using the first calibration method

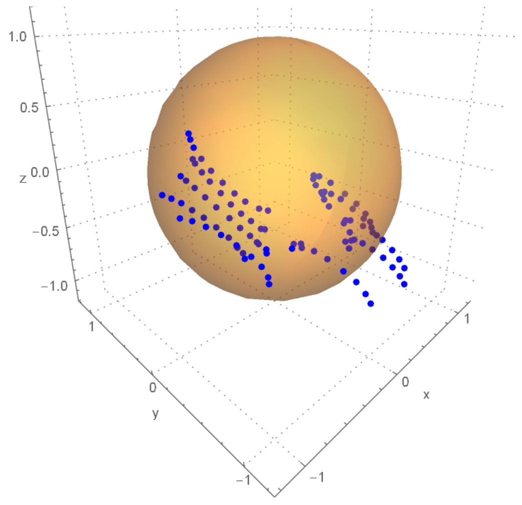
is now in the same order of magnitude as the mean of the non-diagonal entries. This hints at some sort of interference influencing the measurements. The maximum error of the norm across all datasets is 2.60E-2 which is again justified by possible interference. Later, this interference is better acknowledged and IMU B is proven unusable. Nevertheless, these techniques might still be useful in the future if these problems are solved.

Table 4.4 shows the mean and standard deviation of the angle between the g' vector and the magnetism vector. This metric relates to the dot product invariance calibration method and is already used to corroborate the correctness of this method. The low standard deviation for the angle implies that the frame of the accelerometers is already roughly aligned with the frame of the magnetometers.

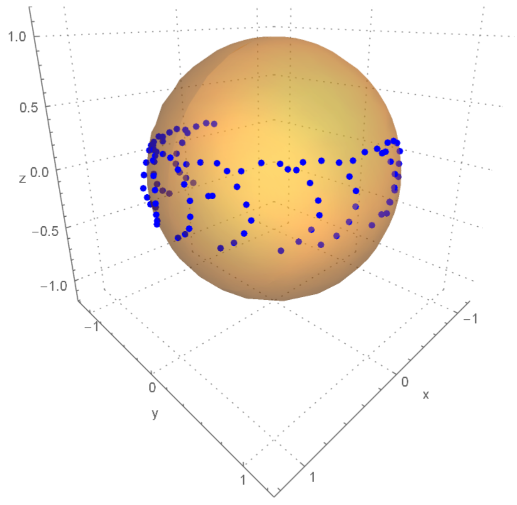
	Angle (°)
Mean	141.24
Standard Deviation	1.73

Table 4.4: Mean and Standard Deviation of the Angle between g' and the magnetism vector for 10 datasets when calibrating with the first method

When using the second method - the dot product invariance method - a solution always exists. Unlike with the previous method, even IMU A has defined results for the best estimates of G and B. Their validity is checked in the plot of Figure 4.5.



(a) IMU A



(b) IMU B

Figure 4.5: 3D plot of the measurements of a dataset from both magnetometers after calibration with the second method

As expected, the corrected measurements from IMU A do not correspond to valid magnetic field vectors. The shape of the arcs is maintained and the points stay relatively close to the surface of the unit sphere. However, the pattern present in the accelerometer measurements is not the one visualized in this case. IMU B seems to follow that pattern, although there seems to be a slight shift from the center.

Table 4.5 shows the mean and standard deviation of the parameters calculated using the second method. Regarding IMU A, the G matrix entries have values that correspond to an extreme transformation between the frames of the pre-calibration measurements and the post-calibration ones. The non-diagonal entries are of the same order of magnitude as the diagonal ones, and some of them are even higher. Even more impressive is the negative value for G_{zz}, which implies that the frame was flipped and now follows better the left-hand rule than the right-hand rule. The variance of these terms is also higher than for IMU B, which is a typical consequence of trying to fit to data that does not follow the model. Slight variations to the movements of each dataset result, as expected, in fitting the model to particular datasets.

	IMU A		IMU B	
	Mean	Standard Deviation	Mean	Standard Deviation
G_{xx}	2.00E-3	1.61E-4	2.51E-3	1.97E-5
G_{xy}	1.17E-2	1.30E-4	6.97E-5	1.90E-5
G_{xz}	3.45E-3	2.57E-4	1.11E-4	3.32E-5
G_{yx}	3.80E-3	5.25E-4	-4.51E-5	1.79E-5
G_{yy}	3.56E-3	3.33E-4	2.53E-3	1.02E-5
G_{yz}	-1.01E-2	4.57E-4	2.53E-4	5.10E-5
G_{zx}	4.67E-3	5.26E-4	-4.27E-5	2.84E-5
G_{zy}	-6.13E-3	2.42E-4	2.02E-4	4.67E-5
G_{zz}	-5.98E-3	2.40E-4	2.60E-3	9.76E-5
B_x	-1.36	4.00E-2	-1.90	9.21E-2
B_y	-2.22	6.76E-2	-2.44	1.24E-1
B_z	-5.16E-1	7.46E-2	-6.17	2.22E-1

Table 4.5: Mean and Standard Deviation of the Magnetometers parameters obtained from 10 datasets using the second calibration method

The columns for IMU B display parameters which are similar to the ones calculated using the first method. The most noticeable difference is the existence of non-zero values in the upper right corner of the G matrix. Nevertheless, the main diagonal has significantly higher values than the non-diagonal entries. However, some entries, such as G_{xz}, G_{yz} and G_{zy}, are only 1 order of magnitude lower. This suggests that either the axes of the magnetometers are not as orthogonal as the first method displays or that the magnetometers and the accelerometers are not well aligned with each other.

The maximum absolute errors of the norm of the measured vector, using the second method, are 4.50E-1 for IMU A and 7.18E-2 for IMU B. The expected norm is, just as before, 1. The high error for IMU A is of no surprise considering the plot in Figure 4.5. The error for IMU B is slightly higher using this method. This result is surprising given that the parameters are calculated directly using the least squares solution, which gives the unbiased minimum variance solution. The first method uses least squares to find variables that algebraically relate to the parameters, but not the parameters themselves. Therefore, the second method should show better results, even if the metric used for calibration isn't the norm of the vector.

Table 4.6 is the equivalent of Table 4.4 for the second calibration method. That is, it shows the mean and standard deviation of the angle between the g' vector and the magnetic field vector.

IMU A, once again, demonstrates data without any value. Its mean is completely different from the more reliable value found by the second method and its variance is high.

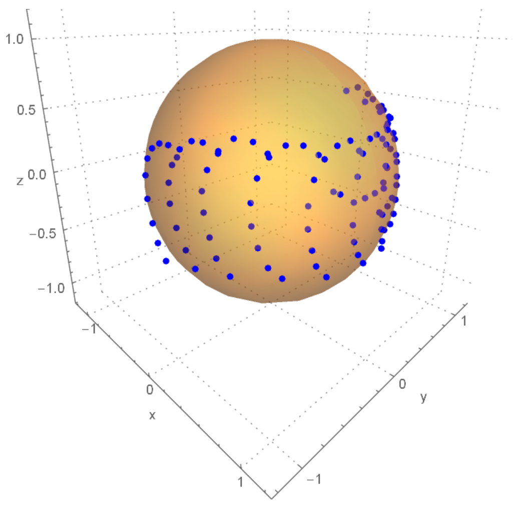
The angles measured using the calibrated IMU B have a mean similar to when the second calibration procedure was used. However, the variance is higher in this case. At first glance, from the data in this table and the errors of the norm, it would appear that the first method is more suitable for calibration. An alternative explanation is that the problem affecting IMU A also affects IMU B but at a smaller scale. That is, even though IMU B has more clearance from materials that

	Angle (°)	
	IMU A	IMU B
Mean	39.22	144.12
Standard Deviation	12.63	2.34

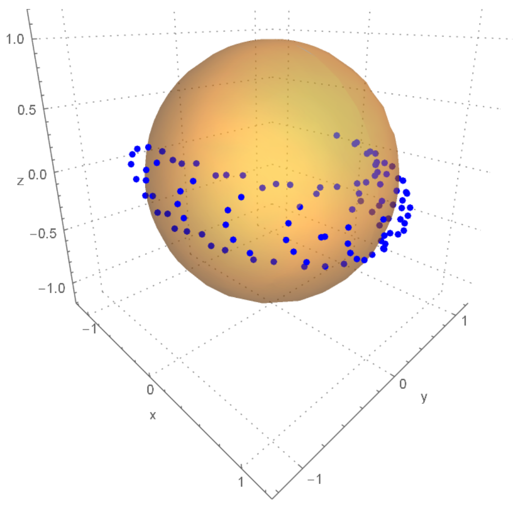
Table 4.6: Mean and Standard Deviation of the Angle between g' and the magnetism vector for 10 datasets when calibrating with the second method

might cause magnetic interference, it might not be sufficient. Perhaps the magnetic field is still influenced by these materials at that distance. If that were the case, the measured vector would not always have the same orientation relative to the g' vector, which would invalidate the dot product invariance method.

To test this hypothesis, an extra dataset was collected for IMU B. The difference between this dataset and the previous ones is that the movements were performed by joints 1 and 3 instead of 1 and 2. This choice of joints should cause the same orientations on IMU B, while emphasising the distortions. Figure 4.6 shows plots of the measurements of the extra dataset after calibration with each method.



(a) First calibration method



(b) Second calibration method

Figure 4.6: 3D plot of the measurements of the extra dataset from magnetometers B after calibration with both methods

The results of the first method are visually acceptable for the most centered arcs. These arcs correspond to small values of θ_3, that is, the cases when there is more distance between the magnetometers and the sources of magnetic distortion. When θ_3 approximates $-90°$ or $90°$, the measured vector starts leaving the surface of the unit sphere. The second method, when trying to minimize the error of the dot product, changes the direction of the measured vector to compensate for the varying norm. Overall, both models fail when acting on θ_3. This effectively invalidates the use of IMU B whenever all joints are used.

4.3 Gyroscope Calibration

4.3.1 Parameter Estimation Algorithm

Unlike the previous sensors, the gyroscope does not measure any quantity when standing still. Some high precision gyroscopes can measure the rotation of the Earth, however this is not the case with low-cost devices. This lack of a constant vector implies that the calibration can not be performed without subjecting the gyroscopes to motion. Unfortunately, if this motion is not precise, a ground truth is required. The upside of this characteristic is that the bias of the measurements, which is the most critical parameter of the gyroscope as it is the one responsible for drift, can be determined by collecting measurements while stationary. In this case, the unbiased measurements of the gyroscopes are obtained by subtracting the average of the stationary measurements to each measurement. During long term operations, the bias may change significantly, however, during testing, the amount of time the manipulator is used is reduced. When using it for longer periods of time, the bias should be included in the estimation algorithms so it is corrected as it slowly changes.

Considering that the different triads of sensors should have aligned frames, this can be accomplished by having them use the same reference. Since there is the need for a ground truth for the gyroscopes calibration, both of these problems can be solved by using calibrated accelerometer measurements. If these are used as the ground truth, the problem of calibrating the gyroscopes reduces to finding a formula that relates the gyroscope measurements, its model parameters and the calibrated accelerometer measurements. Then a suitable optimization algorithm must be used to find the parameters.

The gyroscope measurements can be integrated to find the rotation between two positions p and $p+1$. This integration can be approximated by consecutively multiplying the rotation matrices determined at each time step

$$R_{p+1,p} = \prod_{k=k_{p+1}}^{k_p} R_{k+1,k} \tag{4.14}$$

Each individual rotation can be found by exponentiation of the angular velocity vector, as shown in Appendix B

$$R_{k+1,k} = exp(T v_g) \tag{4.15}$$

with T the sample period and v_g the vector that the calibrated gyroscopes should read. Assuming the measurements are already corrected for bias, v_g is

$$v_g = G u_g \tag{4.16}$$

where u_g are the unbiased measurements. It is evident that the rotation matrix $R_{p+1,p}$ is dependent on the gyroscopes unbiased measurements u_g and the gain matrix G. The g' vector, measured by the accelerometers, is influenced by this rotation

$$g'(k_{p+1}) = R_{p+1,p}\, g'(k_p) + \varepsilon_g \tag{4.17}$$

with ε_g the error between the vector after rotation and its expected value. With these equations, the relation between the calibrated accelerometer measurements (g'), the unbiased gyroscope measurements (u_g) and the calibration parameters for the gyroscope (G) is established.

This relation not linear, which means this problem cannot be solved using linear least squares. Alternatively, as suggested in [2], the Downhill Simplex optimization method is used. This algorithm, when attempting to determine n parameters, keeps track of $n+1$ points in a n-dimensional space. Each of these points represents a solution to the problem and is attributed a score related to how good of a solution it is. The score is calculated with an objective function, which the method tries to minimize. At each step, this iterative algorithm replaces the worst scoring point by a new point. The new point is obtained by finding different points that are determined as linear combinations of all existing points. Then, the best of the candidate points is chosen based on its score. Eventually, the method should converge to a minimum of the objective function. When this happens, the best scoring point is considered the optimal solution to the problem. Details of the Downhill Simplex optimization method can be found in [12].

For calibrating the gyroscope, the objective function chosen was

$$F(G) = \sum_p \|g'(k_{p+1}) - R_{p+1,p}\, g'(k_p)\|_2^2 \tag{4.18}$$

and the points have as coordinates the 9 entries of the G matrix.

4.3.2 Implementation

The motion pattern used for calibrating the other sensors is not ideal for gyroscope calibration. Each IMU should turn a considerable amount between consecutive positions to maximize the change of g'. If the rotation is small, then $g'(k_{p+1})$ is similar to $g'(k_p)$ and the noise in the measurements is enough to justify the negligible change. This means that the calibration algorithm would fit G to the noise, as well as the actual rotations.

In order to prevent this, another movement pattern was used. The new positions were selected with the intent of exciting all axes. The pattern is repetitive and is constituted by 4 stages per repetition:

1. $\theta_1 = 0^\circ$; $\theta_2 = 0^\circ$; moving to the next position excites the z-axis
2. $\theta_1 = 0^\circ$; $\theta_2 = 90^\circ$; moving to the next position excites the y-axis
3. $\theta_1 = 90^\circ$; $\theta_2 = 90^\circ$; moving to the next position excites the z-axis
4. $\theta_1 = 90^\circ$; $\theta_2 = 0^\circ$; moving back to the initial position excites the x-axis

This cycle is repeated 10 times, giving rise to 40 positions and therefore 39 collections of measurements from the gyroscopes. Some measurements are also collected when the manipulator is stationary to calculate the bias. During those stationary moments, accelerometer measurements are read to find the g' vector, similarly to how it is done during accelerometer calibration. Again, during these stationary moments, each sensor collects 50 measurements.

Once again, θ_3 is not changed as it is redundant in terms of orientation change.

The initial matrix G chosen for the Downhill Simplex optimization method was the identity matrix multiplied by a constant. The identity matrix assumes full alignment, which should be relatively close to the final solution. The scaling was done to approach the final solution as well, drastically reducing the execution time of the algorithm.

The termination condition for the method was convergence of the points. More specifically, if the best score was sufficiently close to the second worst score, the algorithm ends. The value of the proximity threshold chosen is 10^{-12}, as increasing it did not alter the final score significantly. This implies that, if the method converges to the absolute minimum, the best estimate of G will never have any entry that is more than 10^{-12} away from the real value of that entry.

4.3.3 Results and Analysis

During the execution of the optimization algorithm, the scores of the different points were recorded as they evolved.

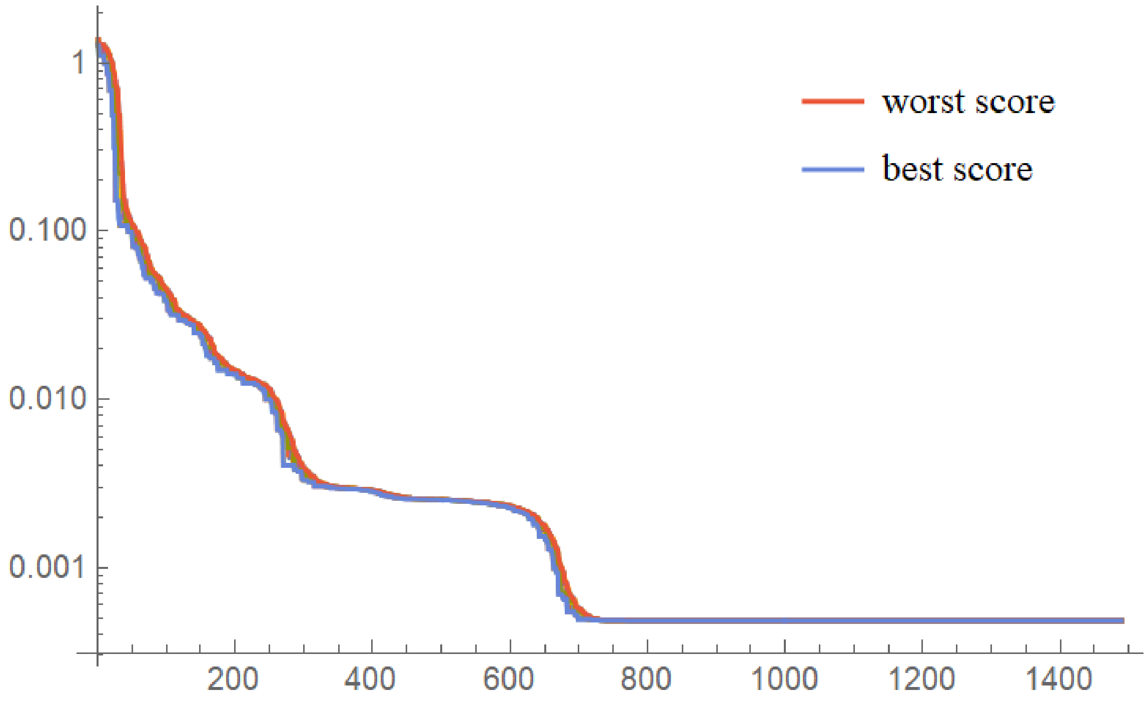
Figure 4.7 shows plots of the evolution of normalized scores for one dataset of each IMU, as well as the scores of the best and worst points. The plots were normalized to show the square of the average error. By taking the square root of a normalized score, the average error is found for each dataset. Recall that the error is the norm of the difference between a vector $g'(k_{p+1})$ and the expectation of that vector after rotating $g'(k_p)$ by $R_{p+1,p}$. This difference vector can be translated into an angle error. That is, if $\varepsilon_g = g'(k_{p+1}) - R_{p+1,p}\, g'(k_p)$, then the norm of ε_g can be translated into the angle between $g'(k_{p+1})$ and $R_{p+1,p}\, g'(k_p)$ using the fact that the vectors $g'(k_{p+1})$, $R_{p+1,p}\, g'(k_p)$ and ε_g form an isosceles triangle.

The plots seem to behave differently. For IMU A all the scores are close to each other and they seem to evolve together, whereas for IMU B they start far apart and converge to the same value. This is due to the different order of magnitude of the parameters visible in Table 4.7. The best solution for IMU B is somewhere to the interior of the starting points, therefore they will converge to a point approximately around their centroid. On the other hand, since the best solution for IMU A is a whole different order of magnitude, the initial points will move relatively close to each other towards the best.

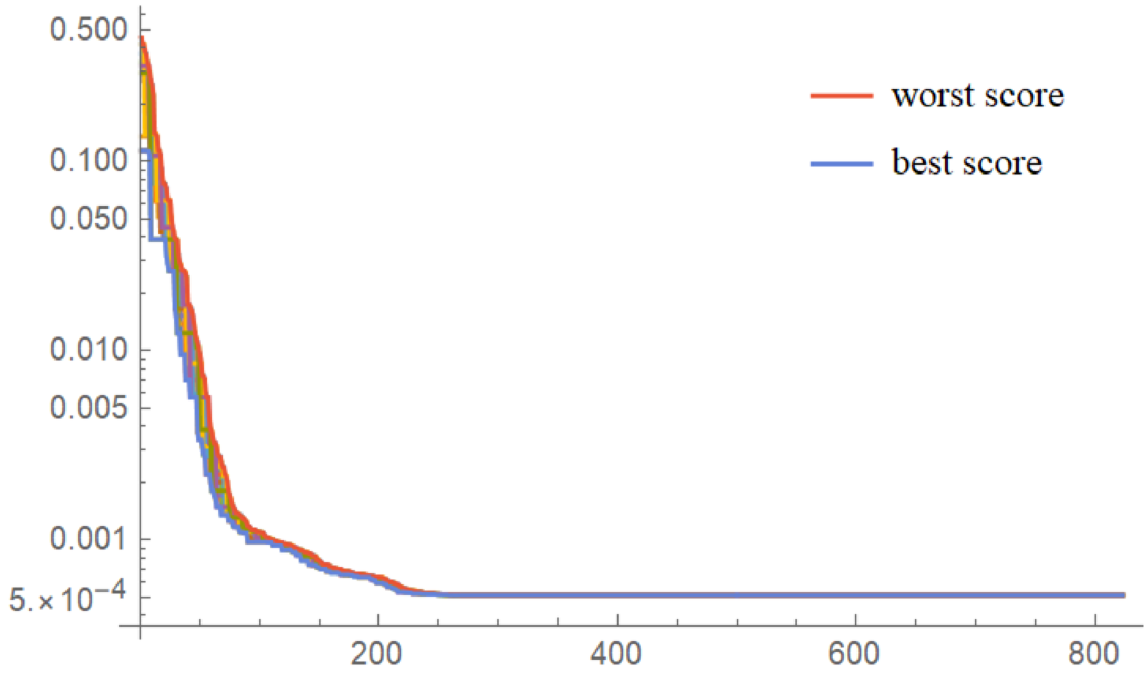
Table 4.7 shows the mean and standard deviation of the calibration parameters across the 10 datasets. In both cases, the main diagonal of G is dominant relative to the non-diagonal entries, once again, as expected from a MEMS device that has its axes mostly orthogonal.

One noteworthy case is visible in entry G_{zz} of IMU A. This value is negative since this IMU does not follow the right hand rule, that is, one of its axis is flipped. As a consequence, one of its axis has a negative coefficient along the main diagonal, in this case, the z-axis.

In both IMUs, the standard deviation of the entries of G is roughly in the same order of magnitude as the mean of the non-diagonal elements, which means their actual value is in fact insignificant in regards to the noise of the sensors. The standard deviation of the entries of B is about one order of magnitude lower than their mean. This is a relatively high value which is probably related with a low number of measurements used to calculate it. However, it produces accurate enough



(a) IMU A



(b) IMU B

Figure 4.7: Evolution of normalized scores of the downhill simplex optimization method for one dataset for each IMU

solutions for its purpose considering that minor drifts will be easily correctable by the sensors that measure quantities more related to the steady-state of the manipulator's model.

Finally, the average and maximum error, as described above, are presented in Table 4.8. Given

	IMU A		IMU B	
	Mean	Standard Deviation	Mean	Standard Deviation
G_{xx}	1.812E-3	1.76E-5	1.22E-4	1.78E-6
G_{xy}	-4.37E-5	6.63E-6	-2.70E-6	9.52E-7
G_{xz}	1.87E-5	7.43E-6	1.32E-7	9.14E-7
G_{yx}	1.79E-5	6.41E-6	5.76E-6	7.70E-7
G_{yy}	1.62E-3	2.25E-5	1.47E-4	1.48E-6
G_{yz}	9.38E-7	9.07E-6	-5.23E-7	7.82E-7
G_{zx}	-4.88E-6	3.18E-6	-1.27E-6	1.42E-7
G_{zy}	-2.03E-5	3.59E-6	1.78E-6	1.39E-7
G_{zz}	-1.75E-3	3.58E-6	1.37E-4	3.25E-7
B_x	-1.04E-2	5.38E-4	-1.06E-2	1.84E-3
B_y	-3.54E-3	1.29E-4	-1.00E-2	9.91E-4
B_z	-8.41E-3	1.25E-4	-8.27E-3	2.25E-4

Table 4.7: Mean and Standard Deviation of the Gyroscopes parameters obtained from 10 datasets

	IMU A		IMU B	
	Average	Maximum	Average	Maximum
Norm (g)	2.59E-2	3.04E-2	2.58E-2	2.73E-2
Angle (°)	1.48	1.74	1.48	1.56

Table 4.8: Average and Maximum Errors for the gyroscope measurements across the 10 datasets, given as the norm of the error vectors ε_g and their corresponding angle

that the sample period used was $0.04s$ and the number of measurements between positions is never less than 1200 for IMU A and 700 for IMU B, then the average measurement is incorrect by less than $0.031^{\circ}/s$ for IMU A and $0.053^{\circ}/s$ for IMU B.

Chapter 5

State Estimation

The state to be determined, in the context of this work, is the pose of the manipulator. The pose of a rigid body is usually defined as its position and orientation. In this case, the pose is a subset of variables used to describe how the different links are oriented. Note that, using the transformations described in Appendix A, the position of each point of the manipulator relative to its base can be found using just the orientation of each link.

The orientations are represented according to the models introduced in section 3.3. These two different descriptions of pose, the Link Model and the Joint Model, will be used with different filters. The Link Model will have its state estimated using the MEKF, whereas the UKF will be used to estimate the state of the Joint Model. Both estimators were implemented in the Pascal programming language, so that they could be used with the application described in section 3.2.

The MEKF was implemented using the algorithm in section 2.4 with the additions pointed out in section 2.4.1. The linear versions of the Link Model are described in section 3.3.1. More specifically, the manipulator model used is given by Equation (3.6) and Equation (3.7), and the sensor models are described by the following equations:

- 3.8 for the gyroscopes;
- 3.10 for the magnetometers;
- 3.12 for the accelerometers;
- 3.14 for the encoders.

The description of the UKF algorithm is in section 2.4.2. As previously mentioned, the models for this filter are not required to be linear. These are described in section 3.3.2. The manipulator is modeled by Equations (3.15) and the sensors by equations:

- 3.16 for the encoders;
- 3.17 and 3.18 for the magnetometers;
- 3.20 and 3.21 for the gyroscopes;

- 3.24, 3.25, 3.26 and 3.27 for the accelerometers.

For implementation, the variances of the models' noises had to be determined. The gyroscope and accelerometer variances were calculated with the information of the calibration datasets. The variances of the state variables were determined empirically, as well as the variances of the encoder noises. The criteria for choosing those variances was to obtain a trade-off between noisy values of the state variables and quickness in achieving a steady-state after transitions. The variances were composed of diagonal matrices since the cross variance between different values was negligible. The empirical values obtained for the variances of the state variables were:

- 4E-5 for $\varepsilon_{\eta_{ij}}$, with $i = \{1,2,3\}$ and $j = \{x,y,z\}$ (Link Model);
- 1E-3 for $\varepsilon_{\omega_{ij}}$, with $i = \{1,2,3\}$ and $j = \{x,y,z\}$ (Link Model);
- 1E-7 for ε_{θ_i}, ε_{ω_i} and ε_{α_i}, with $i = \{1,2,3\}$ (Joint Model);

The sensors' variances are presented in Table 5.1 in their respective units ($({}^{rad}/_{s})^2$ for gyroscopes and encoders and $({}^{m}/_{s^2})^2$ for accelerometers).

Gyroscopes	Variable	$\varepsilon_{\omega_{g_A x}}$	$\varepsilon_{\omega_{g_A y}}$	$\varepsilon_{\omega_{g_A z}}$	$\varepsilon_{\omega_{g_B x}}$	$\varepsilon_{\omega_{g_B y}}$	$\varepsilon_{\omega_{g_B z}}$
	Variance	3.4E-6	6.5E-6	1.1E-5	9.7E-6	8.4E-6	1.3E-5
Accelerometers	Variable	$\varepsilon_{a_{Ax}}$	$\varepsilon_{a_{Ay}}$	$\varepsilon_{a_{Az}}$	$\varepsilon_{a_{Bx}}$	$\varepsilon_{a_{By}}$	$\varepsilon_{a_{Bz}}$
	Variance	6.7E-6	3.1E-5	2.3E-5	8.8E-5	1.9E-4	7.0E-5
Encoders	Variable	ε_{ie_1}	$\varepsilon_{\omega s_1}$	ε_{ie_2}	$\varepsilon_{\omega s_2}$	ε_{ie_3}	$\varepsilon_{\omega s_3}$
	Variance	1E-3	1E-3	1E-3	1E-3	1E-3	1E-3

Table 5.1: Variance of the errors of the sensors

After implementing the filters, each of the sensor models were tested individually (along with the manipulator model) to verify that they were working correctly. Most of these tests were performed with the manipulator stationary or while following this set of movements:

- $\theta_1 = 0°$; $\theta_2 = 0°$; $\theta_3 = 0°$ @ $0s$
- $\theta_1 = 0°$; $\theta_2 = 90°$; $\theta_3 = 0°$ @ $5s$
- $\theta_1 = 0°$; $\theta_2 = 90°$; $\theta_3 = -90°$ @ $10s$
- $\theta_1 = 90°$; $\theta_2 = 90°$; $\theta_3 = -90°$ @ $15s$
- $\theta_1 = 90°$; $\theta_2 = 0°$; $\theta_3 = 0°$ @ $20s$
- $\theta_1 = -90°$; $\theta_2 = 0°$; $\theta_3 = 0°$ @ $25s$
- $\theta_1 = -90°$; $\theta_2 = 90°$; $\theta_3 = 0°$ @ $30s$
- $\theta_1 = -90°$; $\theta_2 = 90°$; $\theta_3 = -90°$ @ $35s$
- $\theta_1 = 0°$; $\theta_2 = 90°$; $\theta_3 = -90°$ @ $40s$

- $\theta_1 = 0^\circ$; $\theta_2 = 0^\circ$; $\theta_3 = 0^\circ$ @ $45s$

These movements were controlled using a PD (Proportional-Derivative) controller. The error that served as input to this controller is the difference between the references described above and the dead-reckoning of the encoders' measurements. On their own, the encoders do not provide an accurate ground truth. However, their measurements are reliable enough to verify the correctness of the models.

Once this process was complete, all the sensors were used to test the accuracy of the estimators at finding the position of the tip of the manipulator. This accuracy test was performed using a structure with strategically located points. This structure is displayed in Figure 5.1. By manually

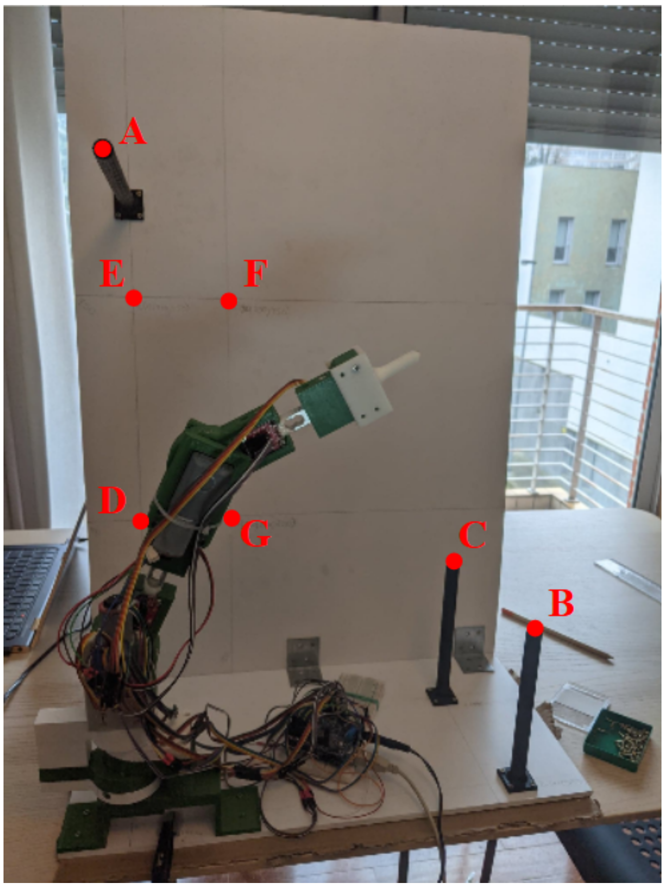


Figure 5.1: Structure used for accuracy tests with its interest points marked

driving the tip of the manipulator to each of the structure's points of interest and calculating the position from the estimated state, the error of the methods can be found for the different points. The tip was guided to the points in alphabetical order, starting at point A, going through all the others, and finishing at A again. The coordinates of the presented points, in millimeters, are:

- A: $(0;0;545.5)$
- B: $(371;0;174.5)$
- C: $(340.5;141.5;174.5)$

- D: $(0;224;200)$
- E: $(0;224;450)$
- F: $(100;224;450)$
- G: $(100;224;200)$

This extensive analysis is performed for the estimator of the Link Model in section 5.1 and for the estimator of the Joint Model in section 5.2.

5.1 Link Model Estimation

5.1.1 Correctness Analysis

To verify the correctness of the sensor models, the estimated state of the Link Model, which is composed of rotation matrices, had to be converted into the angles of the joints. These angles could then be compared to the baseline created by dead-reckoning of the encoder measurements. Appendix B mentions how to convert a rotation matrix into a rotation vector. The matrix that should be converted to a vector is ${}_{i-}^{\;i}R(k)$ for joint i. This can be obtained from the state matrices as

$${}_{i-}^{\;i}R(k) = {}_{i-1}^{i-}R^{T}\, {}_{n}^{i-1}R^{T}(k)\, {}_{n}^{i}R(k) \tag{5.1}$$

Since the rotation of joint i is around the z-axis of the i frame for all joints, the expected rotation vectors should have their x and y components (θ_{ix} and θ_{iy}) close to zero and the z component (θ_{iz}) should match the baseline angle (θ_{ib}).

5.1.1.1 Gyroscope Model

Using just the gyroscopes, the orientation of link 1 remains static throughout the execution of the test. This is a consequence of the lack of an IMU in that link. Under these circumstances, the orientations of each link are independent. Therefore, there is nothing preventing adjacent links to be rotated by an axis different than the one allowed by the joints. This is evident in the graph of Figure 5.2.

This graph shows the evolution of the rotation vector θ_2 and the baseline for the first two angles (θ_{1b} and θ_{2b}). Note that θ_{1b} was included here since there would be no relation between this angle and the one estimated by the Link Model since the estimated orientation of link 1 is static.

However, there should be a relation between θ_{1b} and θ_{2x} because, when $\theta_2 = 0$, a rotation of θ_1 is equivalent to a rotation of link 2 around its x-axis. This association is evident in the time intervals when $\theta_1 \neq 0$, that is $[15;40]s$. Note this relation is even stronger in the interval $[20;30]s$, which is when $\theta_2 = 0$.

The relation between θ_{2b} and θ_{2z} is also evident in the intervals $[5;20]s$ and $[30;45]s$. Again, because the rotation vector is not always pointing along the z-axis of frame 2, the association is mostly present when $\theta_1 = 0$, which is in intervals $[5;15]s$ and $[40;45]s$. Although, even in these

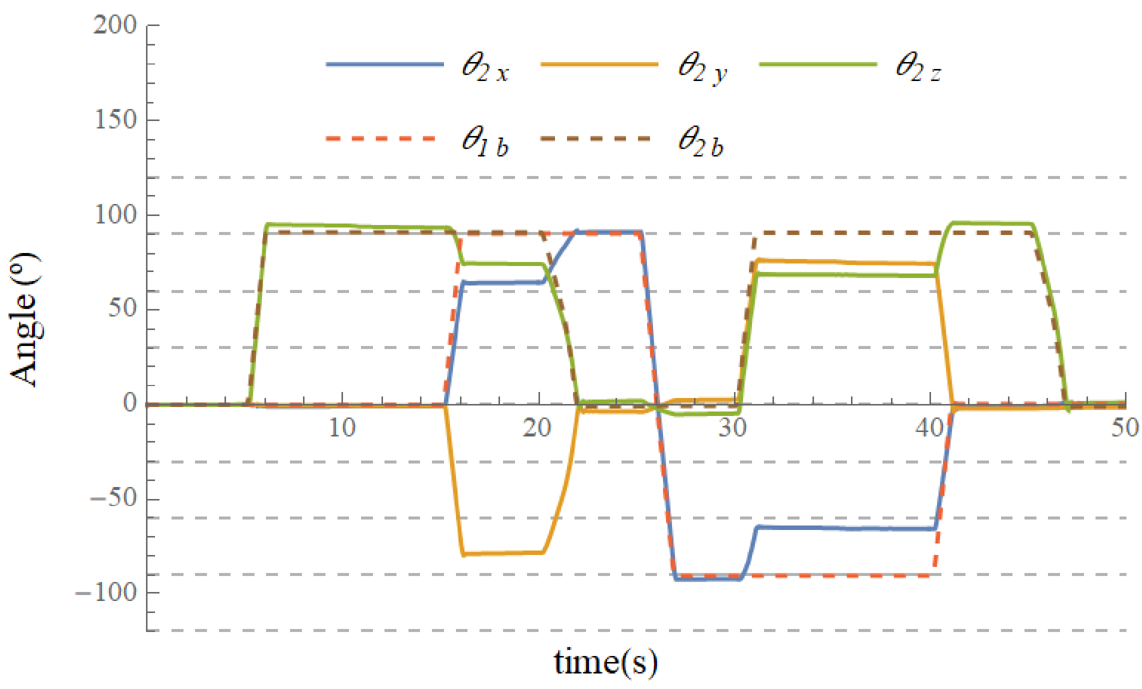


Figure 5.2: Evolution of rotation vector 2 when testing the Link Model using only gyroscope measurements

intervals, the value of θ_{2z} seems to be slightly higher than that of θ_{2b} by about 5°. This slight overshoot is visible when performing the test, which means the gyroscope is being more accurate than the encoders. The reason why the encoder measures a smaller value than the real one is that the movement performed at instants $5s$ and $30s$ is so fast that the encoder misses some pulses. This kind of error is correctable by the other sensors.

In the moments when $\theta_1 \neq 0$ and $\theta_2 \neq 0$, the rotation vector points along a direction that also has a y component, which is visible in the intervals $[15;20]s$ and $[30;40]s$. These also correspond to the intervals when there is not a direct correlation between the baseline variables and the corresponding rotation vector components.

Figure 5.3 shows the graph with the components of rotation vector 3 (θ_{3x}, θ_{3y} and θ_{3z}) and the baseline for the angle of the third joint (θ_{3b}). Assuming the orientation of link 2 is being correctly estimated, this rotation vector should only have a z component. As expected, θ_{3z} roughly matches with θ_{3b}.

However, the x and y components of the rotation vector are still significant during some periods. This discrepancy is noticeable after $15s$, when the first joint starts rotating out of the 0° angle. This error is justifiable by a slight tilt of the manipulator in one direction that should not be allowed by the manipulator construction. This is due to too large clearance between some parts. Nevertheless it is relevant to have the model notice these imprecisions, as they influence the estimate of the position of the manipulator's tip. The reason the error only appears at that moment is related to the joint that moves. Once θ_1 varies, inertia in the manipulator can cause it to tilt to a different side from the original. Overall, the x and y components are small when compared to the

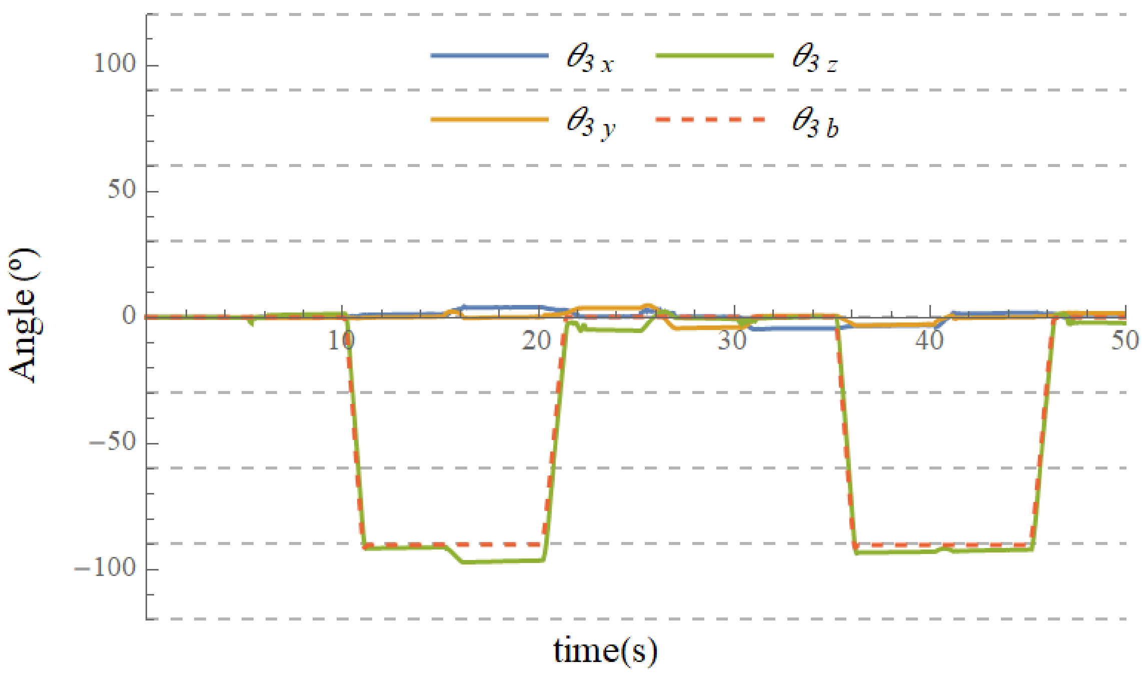


Figure 5.3: Evolution of rotation vector 3 when testing the Link Model using only gyroscope measurements

length of the rotation vector, which is approximately equal to θ_{3z}.

5.1.1.2 Accelerometer Model

Similarly to the gyroscope test, the orientation of link 1 remained static throughout the execution of all movements for the test using only accelerometers. This is again due to the lack of an IMU in link 1. However, because the gravity acceleration vector was approximately parallel to the axis of rotation of joint 1, the angle of this joint has no relation with any component of the available rotation vectors. Much like the previous test, the system is not fully observable. Nevertheless, the remaining angles can be estimated using the information from the accelerometers.

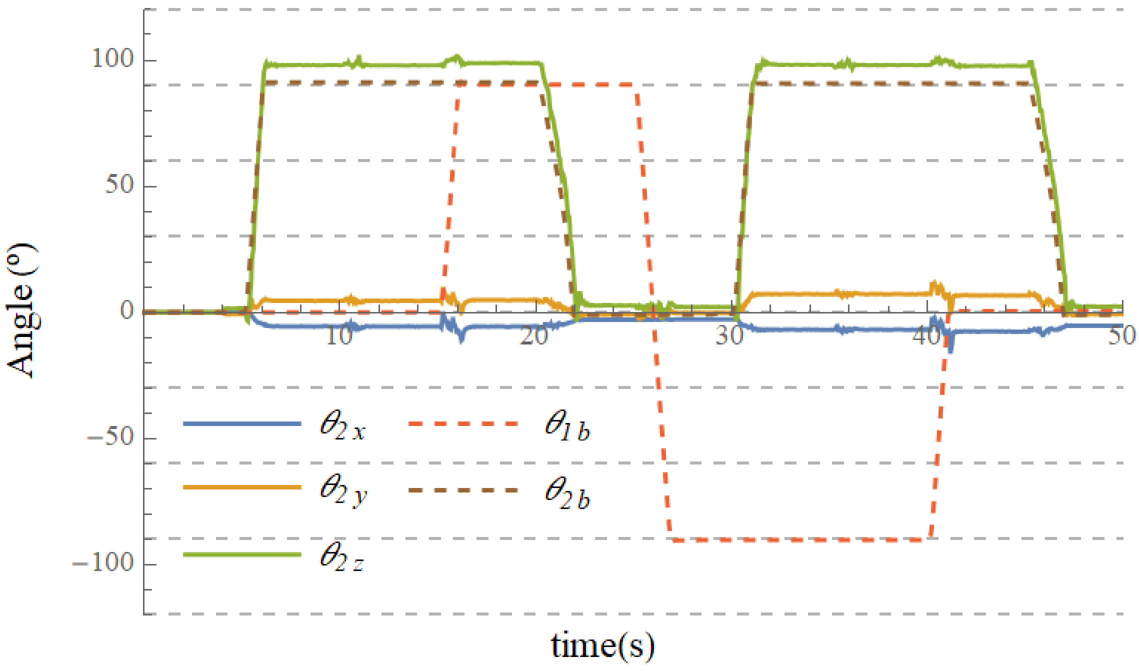


Figure 5.4: Evolution of rotation vector 2 when testing the Link Model using only accelerometer measurements

Figure 5.4 shows the evolution of the components of rotation vector 2 along with the baseline for the angles of joints 1 and 2. The baseline for joint 1 (θ_{1b}) is only included in the graph to show that it is not related to the rotation vector.

As expected, the angle of joint 2 is coincident with θ_{2z}. Furthermore, there are peaks in acceleration during the transitions, which influence the rotation vector in a visible way. The accelerometer measurements justify the encoder errors that were found using the gyroscope test, having obtained an overshoot of θ_{2z}.

An unexpected detail is the small value for the θ_{2x} and θ_{2y} variables that starts being noticed at around $5s$. This error is probably associated with imperfections in the initial angle of joint 3. The initial gravity acceleration vector, and therefore the one expected in the inertial frame, is calculated as the average of the first 100 measurements of the accelerometers of IMU B. If θ_3 is slightly different from zero, then the initial gravity vector for link 2 will be inconsistent with the one from link 3. Consequently, the orientation of link 2 changes slightly to match the initial acceleration vector.

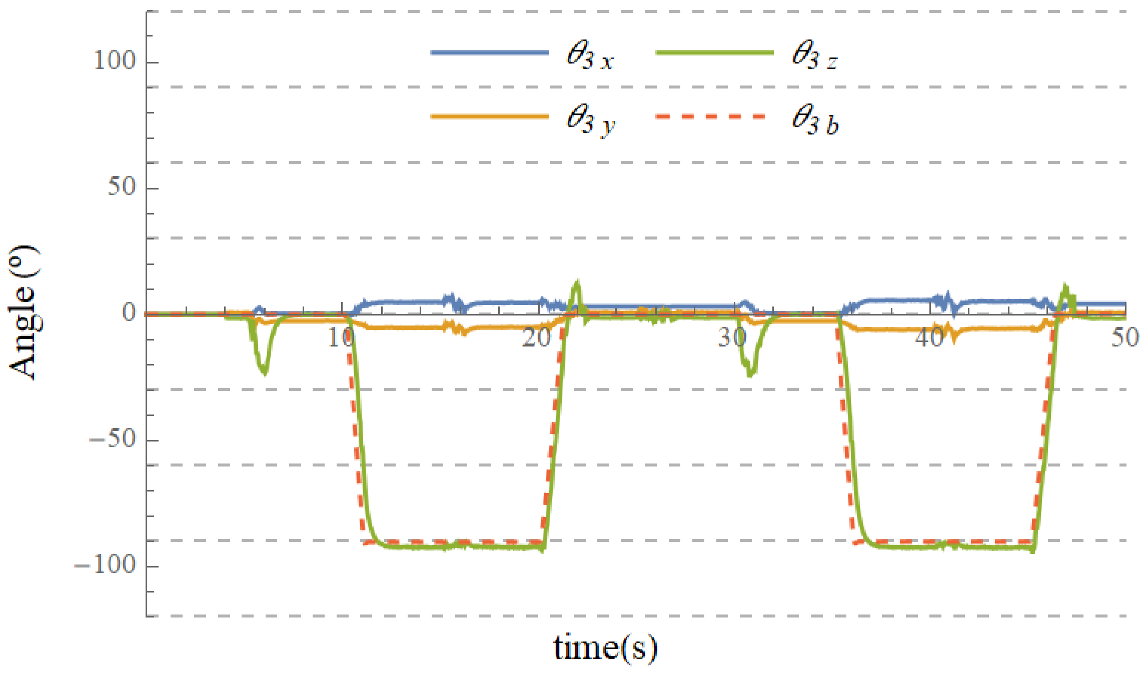


Figure 5.5: Evolution of rotation vector 3 when testing the Link Model using only accelerometer measurements

The graph in Figure 5.5 displays the coherence between θ_{3z} and θ_{3b}. It also demonstrates that the initial error of θ_3 does not affect the orientation of link 3, since the slight orientation change of link 2 produce errors in θ_{2x} and θ_{2y} that are exact opposites of θ_{3x} and θ_{3y}. The peaks of acceleration during transitions are also more noticeable in this link as a consequence of it being further from the base.

5.1.1.3 Encoder Model

If the encoder model is correctly implemented, the z component of the rotation vectors obtained from the orientation estimates of each link should perfectly match the baseline angles since the baseline is obtained using the encoders. This is mostly verified in Figures 5.6, 5.7 and 5.8.

There are slight changes noticeable when a different encoder detects movement. For example, at $10s$, which is when θ_3 changes, θ_2 is slightly affected by that movement. This is a natural consequence of the way ω_i was modelled. Since it follows a random walk model, there should be some inertia when altering the orientation of a link. For example, if θ_3 changes, then the expectations of the model for links 2 and 3 to stand still, force θ_2 to also change. However, the lack of a change in that joint detected by its encoder quickly reverts this small effect.

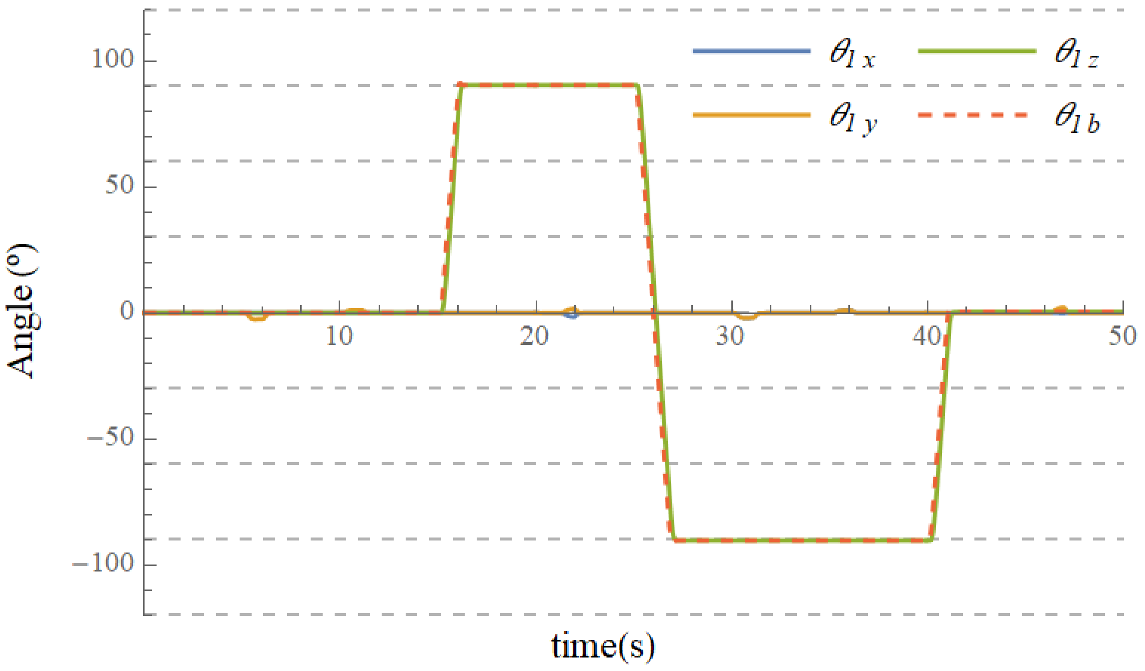


Figure 5.6: Evolution of rotation vector 1 when testing the Link Model using only encoder measurements

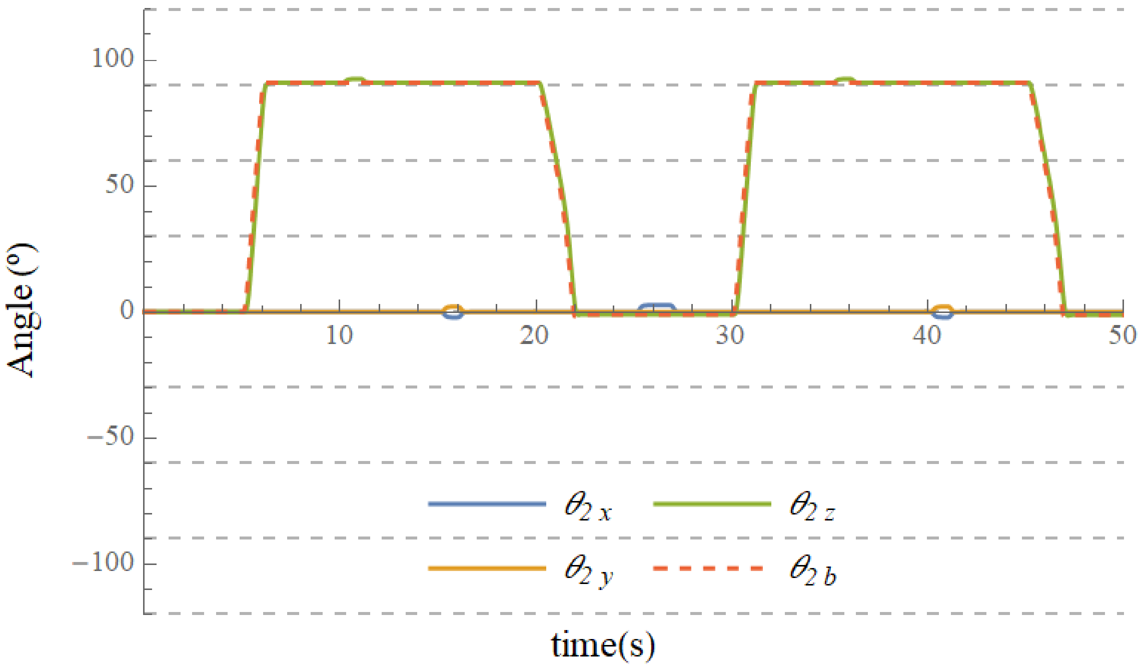


Figure 5.7: Evolution of rotation vector 2 when testing the Link Model using only encoder measurements

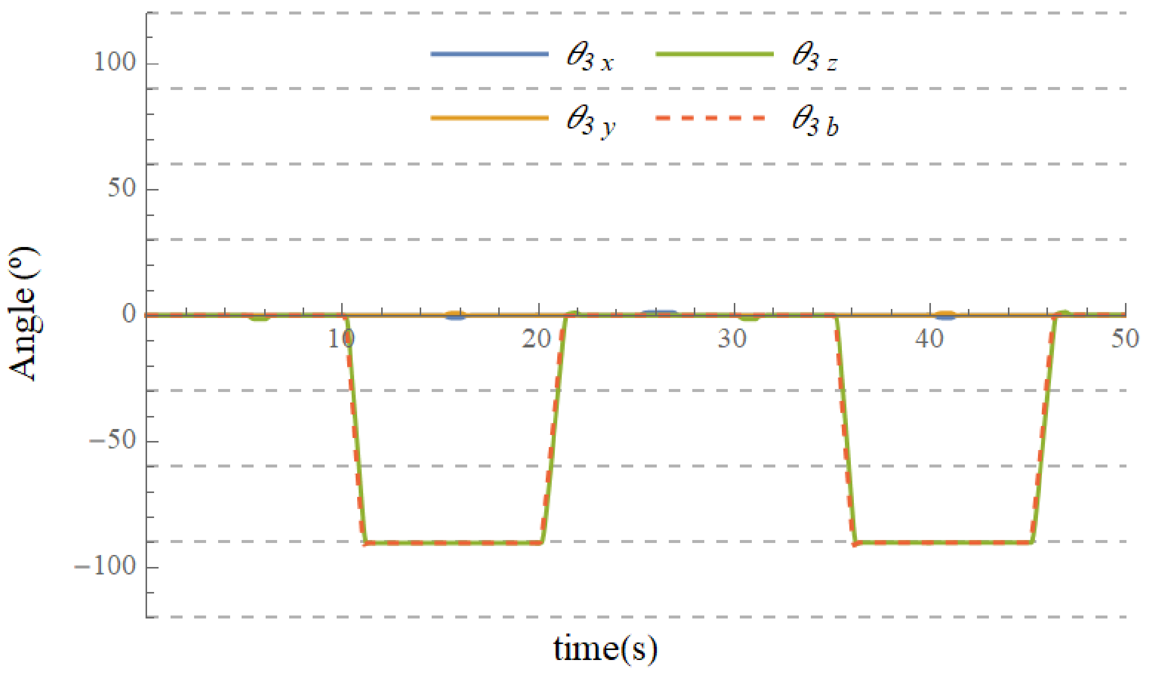


Figure 5.8: Evolution of rotation vector 3 when testing the Link Model using only encoder measurements

5.1.1.4 Full Model

Combining the measurements of all available sensors produces the estimates depicted in Figures 5.9, 5.10 and 5.11.

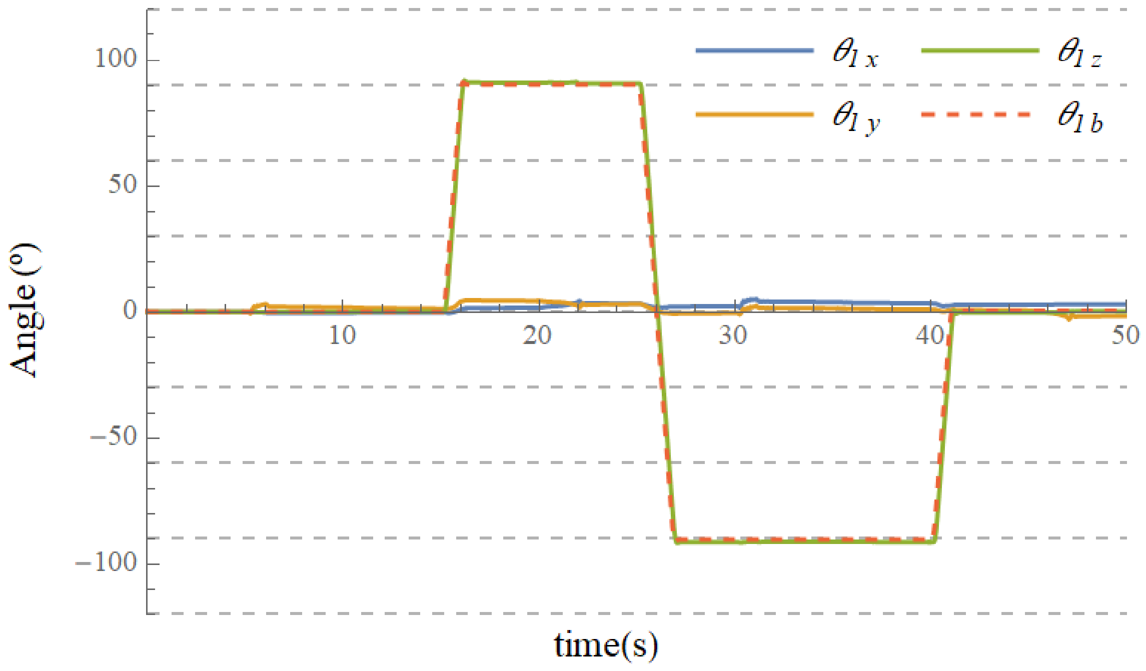


Figure 5.9: Evolution of rotation vector 1 when testing the Link Model using all measurements

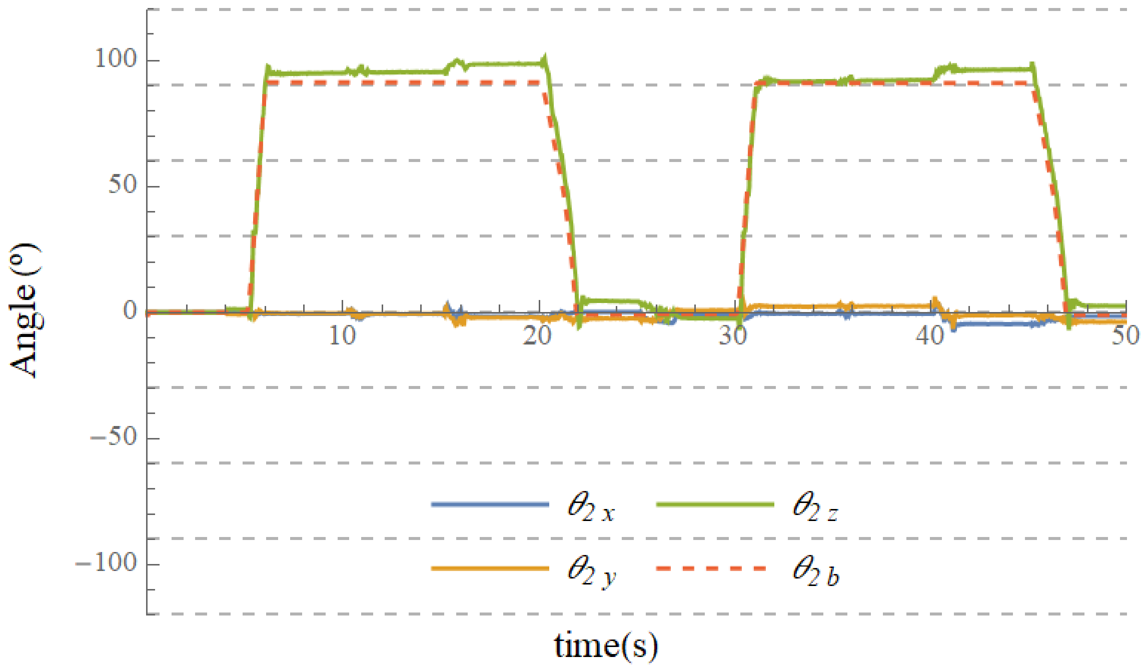


Figure 5.10: Evolution of rotation vector 2 when testing the Link Model using all measurements

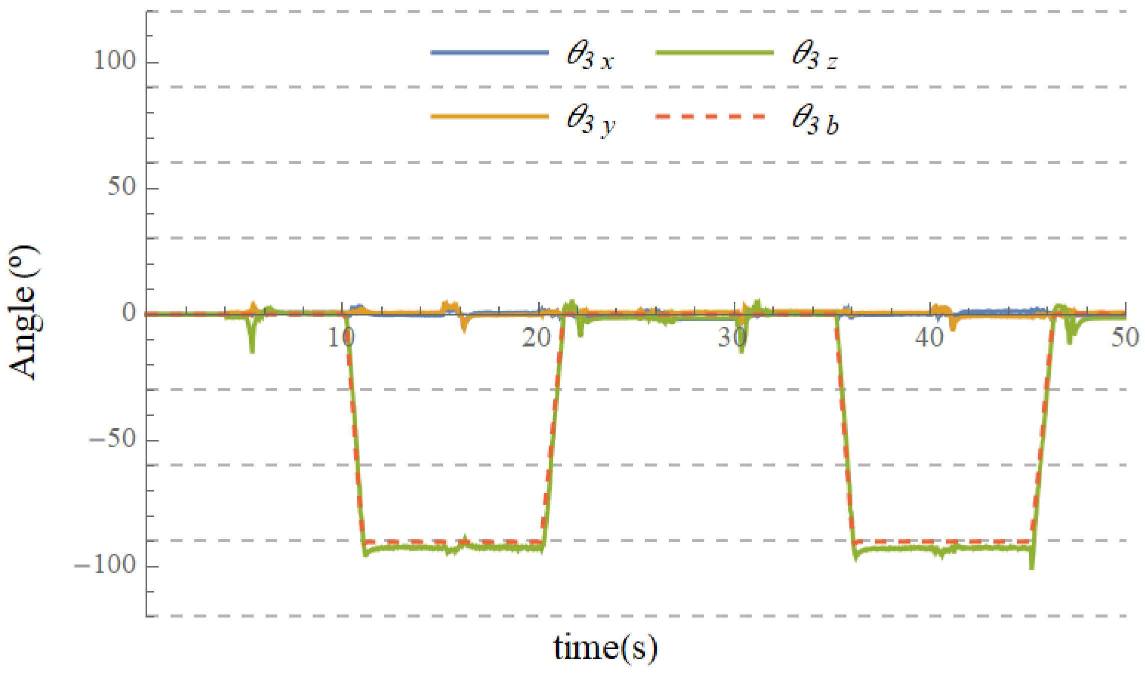


Figure 5.11: Evolution of rotation vector 3 when testing the Link Model using all measurements

Clearly, the full model detects the overshoot of θ_2 as well as the diminute values of the x and y components of the rotation vectors that the encoders can not measure. Furthermore, this model is capable of estimating the value of θ_1 since it limits the relative orientations of adjacent links and attributes changes along the x-axis of links 2 and 3 to the first encoder's measurements. This allows the model to estimate in a level that it is not capable of when using only the gyroscopes and the accelerometers, making the system fully observable.

Table 5.2 displays a summary of the previously discussed errors of the Link Model when using individual sensors.

Sensors used	Problem	Probable Cause
Gyroscopes	Static θ_1 rotation vector	Missing IMU in link 1
	No direct relation of the components of θ_2 with θ_{1b} and θ_{2b}	Static orientation of link 1 causes the orientation of link 2 to have 2 DOF
Accelerometers	Static θ_1 rotation vector	Missing IMU in link 1
	Significant values for the components θ_{2x} and θ_{2y}	Initial misalignment of link 2 relative to link 3
	Significant noise on θ_{3z} during transitions	Large distance from the base of the manipulator
Encoders	Low noise on θ_{2b}	"Inertia" of the model causes the orientations of the links to fall behind
	Overshoot not detected	Fast angular velocity of a joint causes the encoder to miss pulses

Table 5.2: Summary of the errors of the Link Model using individual sensors

5.1.2 Accuracy Test

The accuracy test was performed using the Link Model under the previously specified conditions.

Point	B	C	D	E	F	G	A
Norm of the Error (mm)	9.50	12.69	21.40	20.25	43.79	56.64	40.75
Norm of the Error projected on the xy-plane (mm)	9.49	10.60	19.32	16.19	42.48	54.98	40.04
Norm of the Error projected on the z-axis (mm)	0.28	6.98	9.20	12.16	10.64	13.62	7.57

Table 5.3: Norm of the errors of the accuracy test performed on the Link Model

The norm of the error vectors was collected for each point and is displayed in Table 5.3. All the values from the table are presented in millimeters.

The maximum error has a norm of $5.7cm$, which is quite large in the scale of the manipulator. In order to analyse the source of these errors, the horizontal and vertical projections of the errors were included in the table. Note that most of the error is located in the horizontal projection. Furthermore, this error has a tendency to increase, except when reaching points E and A.

All of these errors are caused by a drift of θ_1. All sensors with the exception of the accelerometers measure a changing quantity. That implies that the accelerometers are the only sensors capable of correcting for sources of drift. Given that the acceleration of gravity is almost parallel to the rotation axis of θ_1, drift on this angle cannot be compensated for. This drift was not noticeable in the correctness test since the duration of that test was much shorter. On the other hand, the accuracy test was performed much slower, by manually commanding the motors to reach the desired points. This allows for a small biases to accumulate and create an increasing error in the horizontal projection. The reason why this error is decreased when reaching points E and A is that the error of the tip's position becomes less dependant on the value of θ_1 as the manipulator approaches a vertical pose. In the limit, when $\theta_2 = \theta_3 = 0$, the value of θ_1 does not change the position of the tip.

Nevertheless, the error in the vertical component is still significant, even though it is considerably smaller than the error in the horizontal component. The maximum error on the vertical component is around $14mm$.

The drift problem is solvable using another sensor that measures a quantity that directly relates to the orientations instead of relating to their rate of change. One example of such a sensor is the magnetometer. If the calibration problem of the magnetometers was solved, this sensor would serve as a viable solution to the drift problem.

Point	B	C	D	E	F	G	A
Error in θ_1 (°)	1.85	5.49	13.74	18.06	19.28	23.27	23.74
Error in θ_2 (°)	4.48	1.85	6.54	10.66	16.63	17.20	9.17
Error in θ_3 (°)	1.35	2.80	4.61	4.54	3.40	26.81	3.59

Table 5.4: Error of the angles of the joints during the accuracy test for the Link Model

A confirmation on the cause of the problem can be observed in Table 5.4. From it, it becomes more evident that the error in θ_1 increases and that it is the most responsible joint for the accumulation of error. This table was obtained with the angles the manipulator should have in order to reach the real positions. These were calculated by means of the inverse kinematics of the manipulator.

5.2 Joint Model Estimation

5.2.1 Correctness Analysis

Unlike with the previous model, a conversion is not required to compare the results with the baseline provided by the encoders. The state of the Joint Model already includes the best estimates of each angle.

5.2.1.1 Gyroscope Model

Figure 5.12 shows the estimates of the joints' angles obtained with the Joint Model using only gyroscope measurements. There is clearly an aberrant behavior which manifests at the first sign of movement. When θ_2 changes at $5s$, the estimate of θ_2 (θ_{2est}) falls short of the baseline (θ_{2b}) and the estimate of θ_3 (θ_{3est}) also changes. This error in θ_{3est} persists after rotation of that joint at $10s$. Later, when θ_1 changes at $15s$, the error in both estimates (θ_{2est} and θ_{3est}) is corrected.

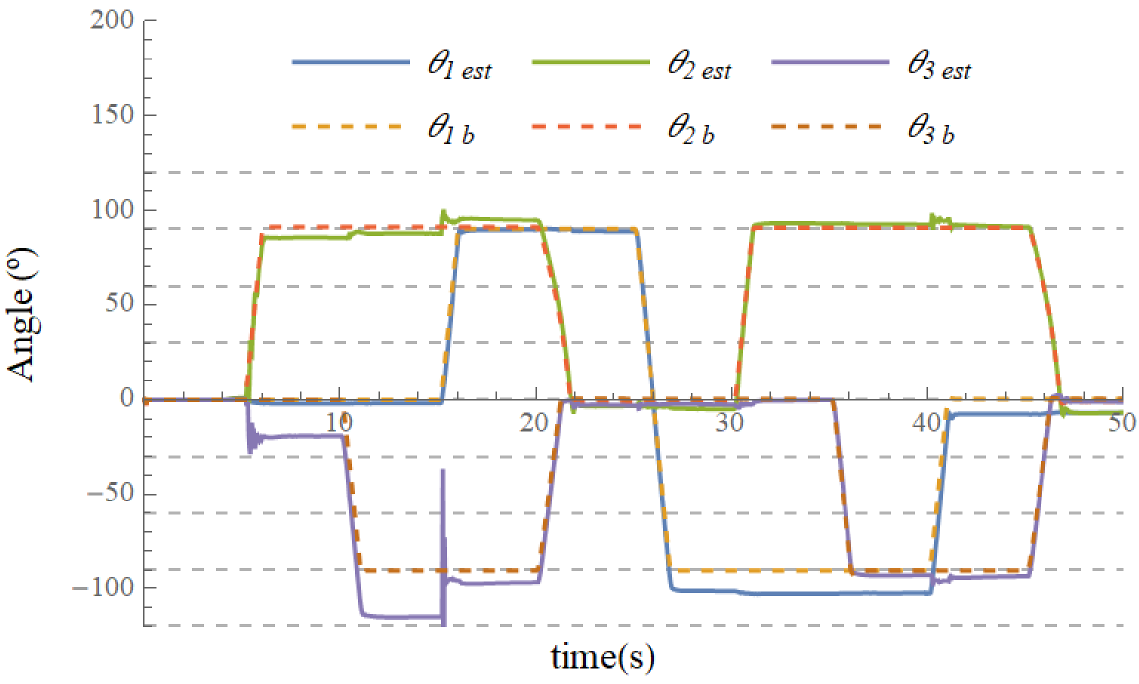


Figure 5.12: Evolution of estimates of the joints' angles when testing the Joint Model using gyroscope measurements

These errors are the first sign of a flaw in the Joint Model. This model assumes that the pose of the manipulator is fully describable by the angles of each joint. However, as the results of the Link Model have shown, there can be slight misalignments between the links which result in a slightly different pose. These misalignements are not accounted for by the Joint Model. In this case, the consequence is that the small difference of the rotation axes of joints 2 and 3 results in a different amount of rotation measured in the axes of the inclined joints.

The errors are corrected when θ_1 changes because the different IMUs measure that rotation in different axes. Assuming $\theta_2 \approx 90°$ and $\theta_3 \approx -90°$, the rotation of joint 1 should be measured mostly in the y-axis of IMU A and on the x-axis of IMU B. Therefore, the only way the model can justify this change is by quickly altering its estimates to the actual values of θ_2 and θ_3. The full model avoids this flaw using the accelerometers to correct the steady-state value of these estimates.

5.2.1.2 Accelerometer Model

The accelerometer test results present an even stronger evidence of the flaw in the Joint Model.

The graph in Figure 5.13 was obtained from the first two seconds of execution of a static test. Despite the absence of movements, the results imply that the manipulator is spinning its first joint at high speed.

Once again, the culprit behind this problem is the tilt that causes a misalignment of the links from their expected frames. This tilt causes a part of the gravity acceleration vector to be projected into the yz-plane of the IMUs. However, if the manipulator was perfectly vertical, the same vector should only be noticed in the x-axis of the IMUs. Since there is no acceleration expected in

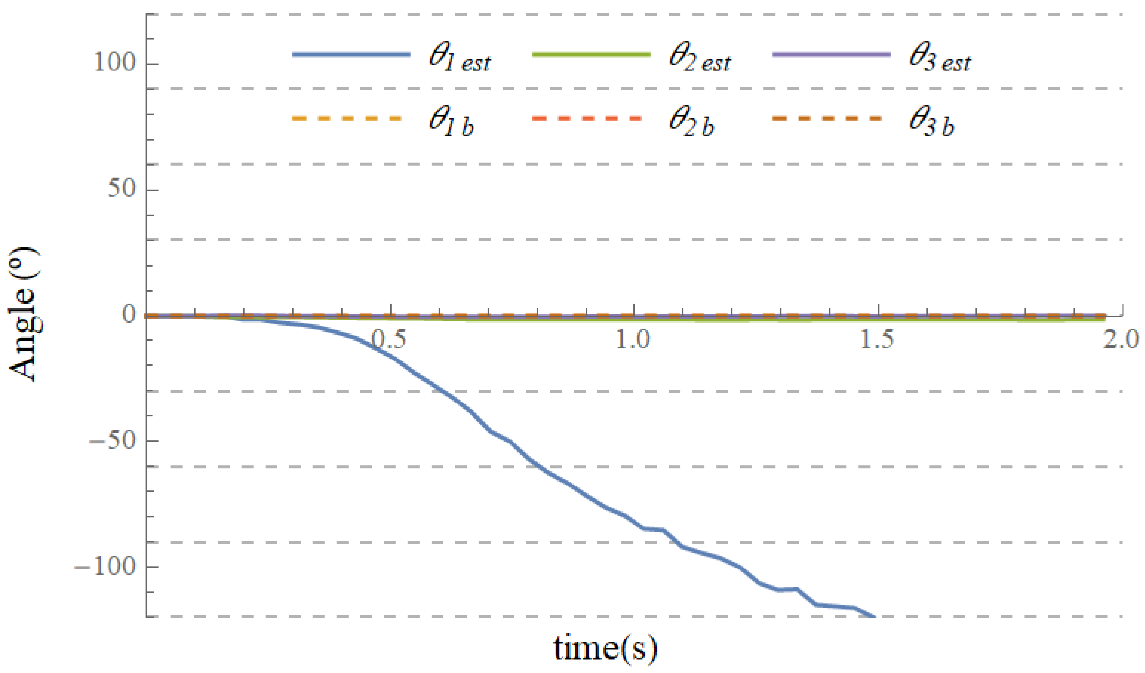


Figure 5.13: Evolution of estimates of the joints' angles during a static test of the Joint Model using accelerometer measurements

that plane from that vector, the acceleration in that plane is justified as centrifugal acceleration. Considering the distance of the IMUs to the axis of rotation, the angular velocity must be high in order to justify the small acceleration measured in the horizontal plane. This angular velocity translates to a quickly changing value of θ_1.

When testing the model using only the accelerometer and gyroscope measurements of IMU B, a curious case arises. This can be observed in Figure 5.14. Since there is no limitation to the orientation of link 2 from sensor measurements but there is a limitation to the angular velocity imposed by the gyroscope, the model justifies a high centrifugal acceleration by moving the IMU farther from the axis of rotation. The way the model justifies this distancing is by assuming θ_2 approaches $-90°$. θ_3 tend towards $90°$ in order to maintain the expected orientation of link 3.

The solution adopted for this problem was to deny the model the ability to justify the acceleration in the horizontal plane as centrifugal acceleration. In other words, the accelerometer model was simplified to assume the only acceleration measured is due to the acceleration of gravity

$$^{i}g(k) = R^{T}_{\theta_i}(k)\,{}^{i-}_{i-1}R^{T\ i-1}g(k) \tag{5.2a}$$

$$f_{a_i}(k) = -{}^{i}g(k) + \varepsilon_{a_i}(k) \tag{5.2b}$$

This is the same assumption as the one used in the Link Model. This new model is similar to the model of the magnetometer since it is equivalent to measuring a rotated vector that is static in inertial space. The results of this corrected model to the test with the usual movements are

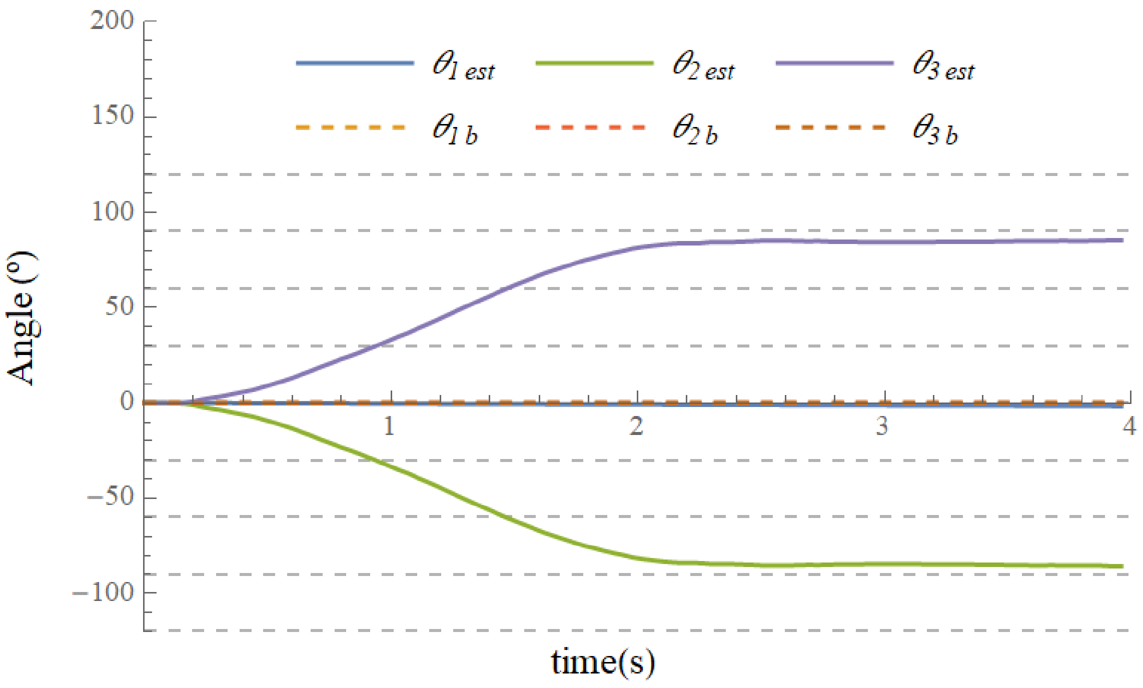


Figure 5.14: Evolution of estimates of the joints' angles during a static test of the Joint Model using only measurements from IMU B

presented in Figure 5.15. Note this model is still incapable of finding the value of θ_1.

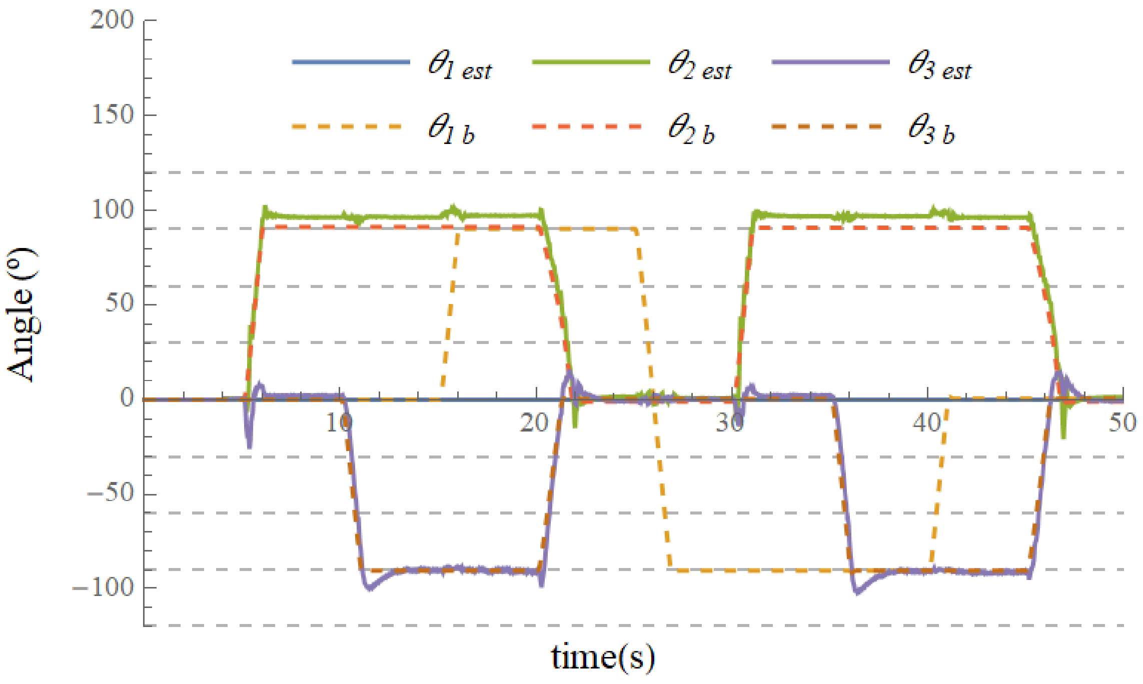


Figure 5.15: Evolution of estimates of the joints' angles when testing the corrected Joint Model using accelerometer measurements

5.2.1.3 Encoder Model

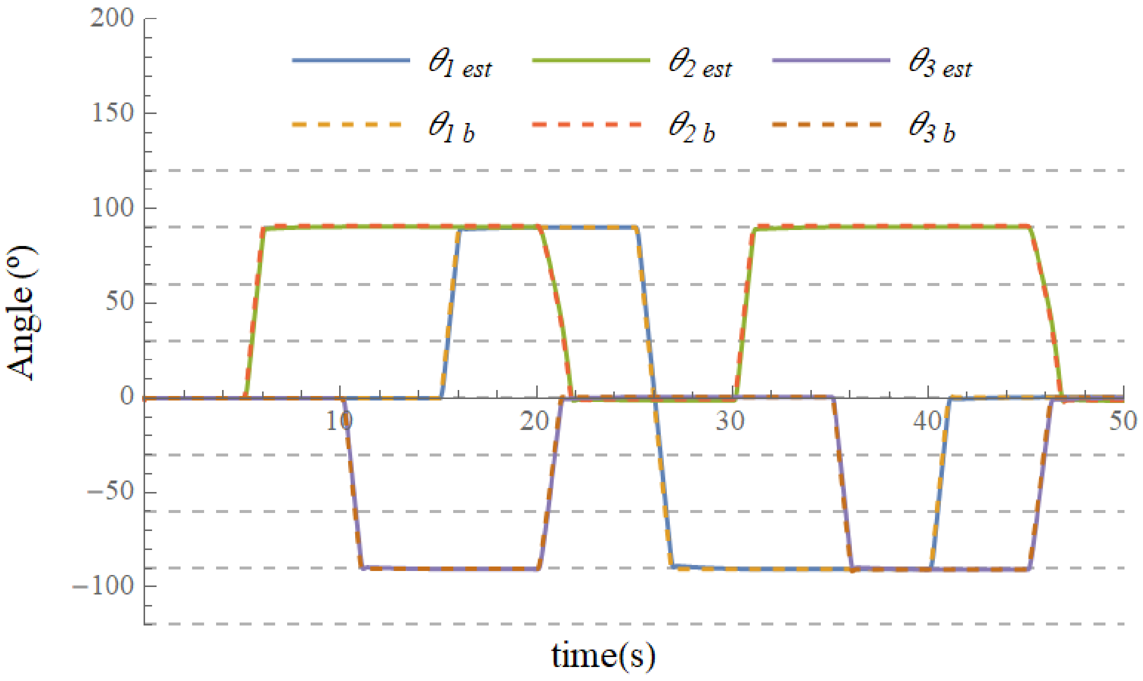


Figure 5.16: Evolution of estimates of the joints' angles when testing the Joint Model using encoder measurements

The graph from Figure 5.16 displays the expected: the estimates of the Joint Model using only encoders follow the baseline obtained from the same encoders. This is an obvious outcome which has its limitations. The measurements from the encoders do not constitute a reliable ground truth, and therefore should not be considered an accurate estimate. For example, this model is incapable of detecting overshoots like the ones observed in Figures 5.12 and 5.15 of the results of the gyroscope and accelerometer models.

5.2.1.4 Full Model

Finally, the results of the full model to the same test are presented in Figure 5.17. The gyroscope problem of the initially incorrect steady-state estimates of θ_2 and θ_3 is removed and the overshoots are detected. It is worth mentioning a last time that this model makes a dangerous assumption, and therefore it should be used carefully.

A summary of the errors observed when using individual sensors is displayed in Table 5.5.

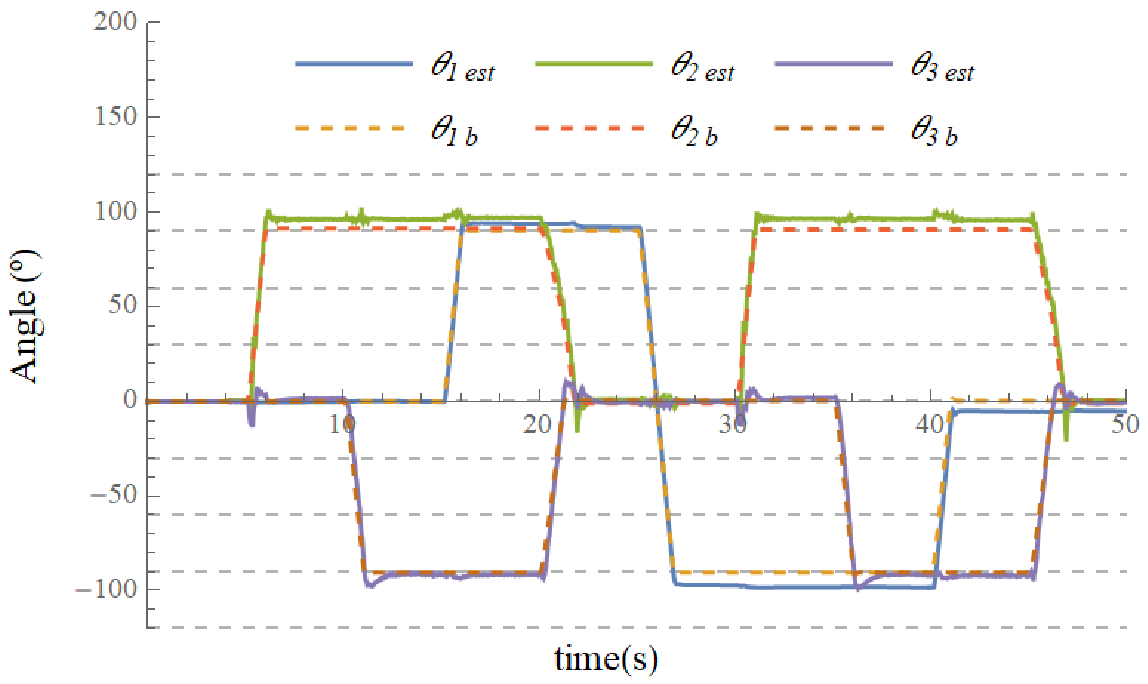


Figure 5.17: Evolution of estimates of the joints' angles when testing the Joint Model using all measurements

Sensors used	Problem	Probable Cause
Gyroscopes	Incorrect amount of rotation estimated for θ_2 and θ_3	Misalignment of the rotation axes caused by a misalignment of the links' orientations
Accelerometers	Fast change of θ_1	A small acceleration projected into the horizontal plane is obtained from the tilt in the links and justified as centrifugal acceleration
	(After correction) inability to estimate θ_1	Gravity acceleration vector parallel to rotation axis of joint 1
	(After correction) significant noise on θ_{3est} during transitions	Large distance from the base of the manipulator
Encoders	Overshoot not detected	Fast angular velocity of a joint causes the encoder to miss pulses

Table 5.5: Summary of the errors of the Joint Model using individual sensors

5.2.2 Accuracy Test

The errors observed during the accuracy test can be found in Table 5.6. Much like as with the Link Model, the drift problem is maintained since the sensors used are the same.

A notable difference is the overall performance of this filter relative to the other. The maximum error was around 5.7*cm* for the Link Model whereas the error for the Joint Model always stays

Point	B	C	D	E	F	G	A
Norm of the Error (mm)	1.50	29.83	7.59	4.45	24.30	35.60	2.12
Norm of the Error projected on the xy-plane (mm)	1.06	29.80	6.44	3.67	23.94	35.55	2.12
Norm of the Error projected on the z-axis (mm)	1.07	1.37	4.02	2.52	4.19	1.90	0.01

Table 5.6: Norm of the errors of the accuracy test performed on the Joint Model

below $3.6cm$. This improvement is not drastic. However, if the effects of drift are set aside, the vertical projection of the error goes from a maximum of $13.6mm$ in the Link Model to $4.2mm$ in the Joint Model. Apparently, the freedom given to the Link Model of tilting the links independently, allows it to tilt to far off the real orientations. These extra degrees of freedom cause the drift to project to the other axes allowing the maximum error to be higher.

Point	B	C	D	E	F	G	A
Error in θ_1 (°)	0.16	4.63	0.93	0.85	5.55	8.33	10.01
Error in θ_2 (°)	0.83	0.61	0.31	0.08	1.69	0.01	0.02
Error in θ_3 (°)	1.94	1.60	2.22	1.03	4.56	0.75	0.68

Table 5.7: Error of the angles of the joints during the accuracy test for the Joint Model

Table 5.7 presents the errors of the joints' angles. Roughly the same conclusions can be taken from this table and Table 5.4, since this is the equivalent for the Joint Model.

While this problem persists, it is inadequate to consider any model better than the other. However, in a situation when drift is necessarily present, the Joint model obtains better results. Overall, the error of this model, when not considering drift, is always below $1cm$, which given the limited precision with which the structure was assembled, means this model might have even better performance when tested with a different ground truth.

Chapter 6

Conclusions

The growing need for robotic manipulators creates a demand for their fast configuration. In this dissertation, several aspects of this setup were discussed, and solutions for this problem were provided.

First of all, a bibliographic revision was performed to explore the subsystems that composed the manipulator and some existing solutions were analysed. Afterwards, a description of the manipulator was provided, as well as the main criteria behind the choice of the proposed solutions. These were selected with the purpose of allowing this work to be as generic as possible. Then, the calibration of inertial sensors was explored and a few solutions were implemented and analysed. Finally, the state estimation problem was approached under the assumption that all sensors, with the exception of the magnetometers, were correctly calibrated. Two models were proposed and their results were compared.

From the calibration results, it became evident that the magnetometers could not be easily modelled since there were components in the manipulator that interfered with the measurements in a way that depended on their proximity to the magnetometers. Since this behavior was too difficult to model for an initial approach, the usage of these sensors was discarded. The remaining sensors were accurately calibrated.

The accelerometers were calibrated using the property that the vector that they measure is constant when the manipulator is static. From this, the solution to least squares was used to find the calibration parameters. After correcting the measurements with those parameters, they fit precisely on a unit sphere's surface, which proved the proposed model was representing the behavior of the accelerometers correctly.

The gyroscopes were calibrated with the aid of the already calibrated accelerometers. By integrating the rotations measured by the gyroscopes, a relation between the accelerometer vectors and the gyroscope calibration parameters was defined. This relation was used as an objective function in an optimization algorithm in order to find those parameters. The low value of the final objective function, as well as the behavior of the optimization algorithm, proved the calibration was successful.

During the implementation of the filters, the correctness of the two used models was verified with extensive tests to the individual sensors. These revealed flaws in each individual sensor. However, when combined, these flaws were corrected. The only one which was uncorrectable by sensor fusion was the one from the Joint Model, which originated from the accelerometers measuring a small horizontal acceleration. This acceleration was caused by a slight tilt of the manipulator, however the model justified it as centrifugal acceleration, which caused the model to portray the movement of the manipulator as rapidly spinning along one axis. This problem was corrected by simplifying the accelerometers' model, by removing its ability to consider centrifugal acceleration.

Despite this flaw on the Joint Model, it still outperformed the Link Model in an accuracy test. This test was performed resorting to an external structure with specified points marked in multiple strategically predefined locations. The estimate of the position of the manipulator's tip was compared to the coordinates of the points of interest when the tip was driven to those points. Both models suffered from drift around an axis. Despite that fact, during the execution of the accuracy test, the Joint Model was able to keep its maximum vertical error under $1cm$. This was the error that was not affected by drift, and therefore represents the accuracy the filter would have achieved if the drift problem was not present.

A possible solution to this issue would be to find a way to correctly calibrate the magnetometers. These would provide measurements that would compensate for drift in the direction in which it existed. Besides an improved calibration technique, the proposed models can be expanded to work with other sensors. Alternatively, new sensors could be used to estimate other aspects of the manipulator other than its pose. For example, the presented manipulator was equipped with load cells capable of measuring force. These can be used to further develop the pose estimators or to help detect collisions or constant strains. These are just some examples of how the work in this dissertation could be further developed in the future.

Appendix A

Kinematics Transformation Matrices

Given two referentials u and v, related at a certain instant by the transformation matrix ${}^{v}_{u}T$, the coordinates of a point p in the u frame (${}^{u}p$) can be written as a function of ${}^{v}_{u}T$ and ${}^{v}p$. More specifically

$$ {}^{u}\tilde{p} = {}^{v}_{u}T\,{}^{v}\tilde{p} \tag{A.1} $$

where $\tilde{p} = [p^T \quad 1]^T$. Equation A.1 can be written as

$$ {}^{u}p = {}^{v}_{u}R\,{}^{v}p + {}^{u}t_v \tag{A.2} $$

with ${}^{v}_{u}R$ a 3×3 rotation matrix and ${}^{u}t_v$ a vector pointing from the origin of the u frame to the origin of the v frame. Taking the derivative of ${}^{u}p$

$$ {}^{u}\dot{p} = {}^{v}_{u}\dot{R}\,{}^{v}p + {}^{v}_{u}R\,{}^{v}\dot{p} + {}^{u}\dot{t}_v \tag{A.3} $$

Given the derivative of ${}^{v}_{u}R$

$$ {}^{v}_{u}\dot{R} = {}^{v}_{u}R[{}^{v}\omega_{uv}\times] \tag{A.4} $$

with proof in [21], expression A.3 can be joined with A.1 to describe the kinematic behavior of ${}^{u}p$ up to its velocity:

$$ {}^{u}\dot{\tilde{p}} = {}^{v}_{u}\dot{T}\,{}^{v}\dot{\tilde{p}} \tag{A.5} $$

with $\dot{\tilde{p}} = [\dot{p}^T \quad p^T \quad 1]^T$ and

$$ {}^{v}_{u}\dot{T} \triangleq \begin{bmatrix} {}^{v}_{u}R & {}^{v}_{u}R[{}^{v}\omega_{uv}\times] & {}^{u}\dot{t}_v \\ 0_{4\times 3} & \multicolumn{2}{c}{\begin{bmatrix} \quad {}^{v}_{u}T \quad \end{bmatrix}} \end{bmatrix} \tag{A.6} $$

In A.4, ${}^{v}\omega_{uv}$ is the angular velocity of referential v with respect to u given in the v frame. $[\ \cdot\ \times]$ is an operator that gives a 3×3 matrix from a vector. This operator can be used to solve a cross product

$$ a \times b = [a\times]b = -[b\times]a \tag{A.7} $$

and therefore is defined as

$$[a\times] \triangleq \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \tag{A.8}$$

with $a = [a_1 \quad a_2 \quad a_3]^T$.

As expected, the acceleration of ${}^u p$ can also be written in terms of ${}^v p$ and another matrix.

$${}^u\dddot{p} = {}^v_u\ddot{T}\,{}^v\dddot{p} \tag{A.9}$$

where $\dddot{p} = [\ddot{p}^T \quad \dot{p}^T \quad p^T \quad 1]^T$ and

$${}^v_u\ddot{T} \triangleq \begin{bmatrix} {}^v_uR & \begin{matrix} 2\,{}^v_uR[{}^v\omega_{uv}\times] & {}^v_uR\,{}^v\Omega_{uv} & {}^u\ddot{t}_v \end{matrix} \\ 0_{7\times3} & \left[\qquad\qquad {}^v_u\dot{T} \qquad\qquad \right] \end{bmatrix} \tag{A.10}$$

with ${}^v\Omega_{uv} = [{}^v\alpha_{uv}\times] + [{}^v\omega_{uv}\times]^2$ and ${}^v\alpha_{uv}$ the derivative with respect to time of ${}^v\omega_{uv}$, *i.e.* the angular acceleration. The top 3 rows of ${}^v_u\ddot{T}$ are found by taking the derivative in equation A.3.

Appendix B

Exponential Map

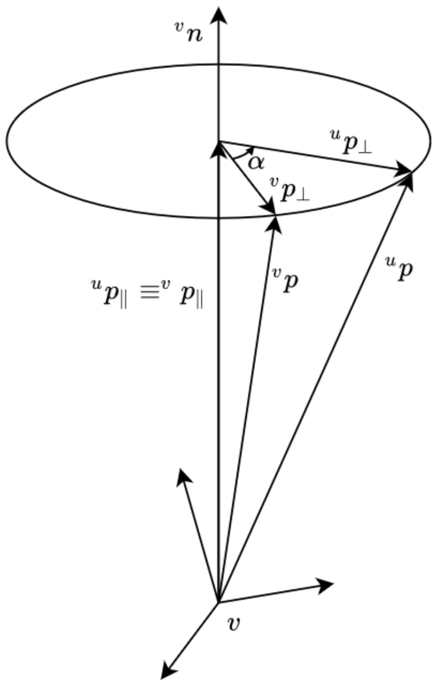


Figure B.1: Generic rotation around a vector ^{v}n

Assume a vector p in a frame v is rotated clockwise α radians around a vector n with norm 1. This new rotated vector has the same coordinates as the initial vector in a frame u, if that frame is a rotation of frame v of α radians along the same vector n (counterclockwise). Therefore, this new vector is ^{u}p. The relation between ^{v}p and ^{u}p is depicted in figure B.1. This relation between frames is what is called the *rotation vector* representation of an orientation.

The new vector ^{u}p can be written as a function of ^{v}p, the angle α and ^{v}n. To show that relation, the vector ^{v}p is initially decomposed into two components: one parallel to ^{v}n and one perpendicular

$$^{v}p = {}^{v}p_{\parallel} + {}^{v}p_{\perp} \tag{B.1}$$

The parallel component is given by

$$ {}^{v}p_{\|} = ({}^{v}p \cdot {}^{v}n)\,{}^{v}n \tag{B.2}$$

and the perpendicular by

$$ {}^{v}p_{\perp} = {}^{v}p - {}^{v}p_{\|} = {}^{v}p - ({}^{v}p \cdot {}^{v}n)\,{}^{v}n \tag{B.3}$$

Decomposing ${}^{u}p$ in the same manner gives

$$\begin{aligned} {}^{u}p &= {}^{u}p_{\|} + {}^{u}p_{\perp} \\ &= {}^{v}p_{\|} + ({}^{v}p_{\perp}\cos\alpha + ({}^{v}n \times {}^{v}p)\sin\alpha) \\ &= ({}^{v}p \cdot {}^{v}n)\,{}^{v}n + ({}^{v}p - ({}^{v}p \cdot {}^{v}n)\,{}^{v}n)\cos\alpha + ({}^{v}n \times {}^{v}p)\sin\alpha \\ &= {}^{v}p\cos\alpha + ({}^{v}p \cdot {}^{v}n)\,{}^{v}n(1-\cos\alpha) + ({}^{v}n \times {}^{v}p)\sin\alpha \end{aligned} \tag{B.4}$$

Considering the following expansion of multiple cross-products

$$ a \times (b \times c) = (c \cdot a)b - (a \cdot b)c \tag{B.5}$$

the previous expression can be further developed

$$\begin{aligned} {}^{u}p &= {}^{v}p\cos\alpha + ({}^{v}n \times ({}^{v}n \times {}^{v}p) + {}^{v}p)\,(1-\cos\alpha) + ({}^{v}n \times {}^{v}p)\sin\alpha \\ &= {}^{v}p + ({}^{v}n \times {}^{v}p)\sin\alpha + ({}^{v}n \times ({}^{v}n \times {}^{v}p))\,(1-\cos\alpha) \\ &= \left(I_3 + \sin\alpha[{}^{v}n\times] + (1-\cos\alpha)[{}^{v}n\times]^2\right){}^{v}p \end{aligned} \tag{B.6}$$

Given that frames u and v have the same origin, then, by definition

$$ {}^{v}_{u}R = I_3 + \sin\alpha[{}^{v}n\times] + (1-\cos\alpha)[{}^{v}n\times]^2 \tag{B.7}$$

This expression can also be written as the exponential of the matrix $\alpha[{}^{v}n\times]$

$$\begin{aligned} e^{\alpha[{}^{v}n\times]} &= \sum_{k=0}^{\infty} \frac{1}{k!}(\alpha[{}^{v}n\times])^k \\ &= I_3 + \alpha[{}^{v}n\times] + \frac{1}{2!}\alpha^2[{}^{v}n\times]^2 - \frac{1}{3!}\alpha^3[{}^{v}n\times] - \frac{1}{4!}\alpha^4[{}^{v}n\times]^2 + \cdots \\ &= I_3 + \left(\alpha - \frac{1}{3!}\alpha^3 + \cdots\right)[{}^{v}n\times] + \left(\frac{1}{2!}\alpha^2 - \frac{1}{4!}\alpha^4 + \cdots\right)[{}^{v}n\times]^2 \\ &= I_3 + \sin\alpha[{}^{v}n\times] + (1-\cos\alpha)[{}^{v}n\times]^2 \end{aligned} \tag{B.8}$$

This is why the relation between a rotation vector ($\alpha\,{}^{v}n$) and a rotation matrix (${}^{v}_{u}R$) is said to be given by an *exponential map*.

The inverse relation, that is, the conversion from rotation matrix to rotation vector, can be easily found by analysis of each element of ${}^v_uR$ as a function of the variables of the previous expression

$$ {}^v_uR = \begin{bmatrix} 1+(1-c)({}^vn_x^2-1) & (1-c)\,{}^vn_x\,{}^vn_y - s\,{}^vn_z & (1-c)\,{}^vn_x\,{}^vn_z + s\,{}^vn_y \\ (1-c)\,{}^vn_x\,{}^vn_y + s\,{}^vn_z & 1+(1-c)({}^vn_y^2-1) & (1-c)\,{}^vn_y\,{}^vn_z - s\,{}^vn_x \\ (1-c)\,{}^vn_x\,{}^vn_z - s\,{}^vn_y & (1-c)\,{}^vn_y\,{}^vn_z + s\,{}^vn_x & 1+(1-c)({}^vn_z^2-1) \end{bmatrix} \tag{B.9} $$

with $s = \sin\alpha$ and $c = \cos\alpha$. From this representation, the following equations can be written, with which the variables α, ${}^vn_x$, ${}^vn_y$ and ${}^vn_z$ can be found

$$ {}^v_uR_{xx} + {}^v_uR_{yy} + {}^v_uR_{zz} = 1 + 2c \tag{B.10} $$
$$ {}^v_uR_{zy} - {}^v_uR_{yz} = 2s\,{}^vn_x \tag{B.11} $$
$$ {}^v_uR_{xz} - {}^v_uR_{zx} = 2s\,{}^vn_y \tag{B.12} $$
$$ {}^v_uR_{yx} - {}^v_uR_{xy} = 2s\,{}^vn_z \tag{B.13} $$

Throughout this document, equation B.7 will be used to represent deviations of orientations by a rotation vector. More specifically, orientations will be represented by

$$ R = \hat{R}R_\eta \tag{B.14} $$

where R_η is the matrix representing the deviation from a linearization point $\hat{R}$. The rotation vector corresponding to matrix R_η is η.

Furthermore, B.7 can be linearized around $\alpha = 0$ in order to build a linear model for orientations.

$$ R_\eta \approx I_3 + [\eta\times] \tag{B.15} $$

References

[1] John L Crassidis, F Landis Markley, and Yang Cheng. Survey of nonlinear attitude estimation methods. *Journal of guidance, control, and dynamics*, 30(1):12–28, 2007.

[2] WT Fong, SK Ong, and AYC Nee. Methods for in-field user calibration of an inertial measurement unit without external equipment. *Measurement Science and technology*, 19(8):085202, 2008.

[3] Victor Hugo Garcia-Rodriguez, Ramon Silva-Ortigoza, Eduardo Hernandez-Marquez, Jose Rafael Garcia-Sanchez, Mario Ponce-Silva, and Griselda Saldana-Gonzalez. A dc motor driven by a dc/dc boost converter-inverter: Modeling and simulation. In *2016 International Conference on Mechatronics, Electronics and Automotive Engineering (ICMEAE)*, pages 78–83. IEEE, 2016.

[4] Nicholas J Higham. *Analysis of the Cholesky decomposition of a semi-definite matrix*. Oxford University Press, 1990.

[5] Jeroen D Hol. *Sensor fusion and calibration of inertial sensors, vision, ultra-wideband and GPS*. PhD thesis, Linköping University Electronic Press, 2011.

[6] Manon Kok, Jeroen D Hol, and Thomas B Schön. Using inertial sensors for position and orientation estimation. *arXiv preprint arXiv:1704.06053*, 2017.

[7] Serdar Kucuk and Zafer Bingul. *Robot kinematics: Forward and inverse kinematics*. INTECH Open Access Publisher, 2006.

[8] Xiang Li and Zhi Li. A new calibration method for tri-axial field sensors in strap-down navigation systems. *Measurement Science and technology*, 23(10):105105, 2012.

[9] F Landis Markley. Attitude determination using vector observations and the singular value decomposition. *Journal of the Astronautical Sciences*, 36(3):245–258, 1988.

[10] J Metge, R Mégret, A Giremus, Y Berthoumieu, and T Décamps. Calibration of an inertial-magnetic measurement unit without external equipment, in the presence of dynamic magnetic disturbances. *Measurement Science and Technology*, 25(12):125106, 2014.

[11] João Moreira, Vítor H. Pinto, José Gonçalves, and Paulo Costa. Using a low-cost robotic manipulator towards the study of over-sensored systems and state estimation. In *Eighth International Conference on Technological Ecosystems for Enhancing Multiculturality*, TEEM'20, page 69–74, New York, NY, USA, 2020. Association for Computing Machinery.

[12] John A Nelder and Roger Mead. A simplex method for function minimization. *The computer journal*, 7(4):308–313, 1965.

[13] Vítor H Pinto, José Gonçalves, and Paulo Costa. Modeling and control of a dc motor coupled to a non-rigid joint. *Applied System Innovation*, 3(2):24, 2020.

[14] Stuart Russell and Peter Norvig. *Artificial Intelligence A Modern Approach Third Edition*. Prentice Hall, 2010.

[15] Isaac Skog and Peter Händel. Calibration of a mems inertial measurement unit. In *XVII IMEKO world congress*, pages 1–6. Citeseer, 2006.

[16] John C Springmann and James W Cutler. Attitude-independent magnetometer calibration with time-varying bias. *Journal of Guidance, Control, and Dynamics*, 35(4):1080–1088, 2012.

[17] Gabriel A Terejanu. Unscented kalman filter tutorial. *University at Buffalo, Buffalo*, 2011.

[18] Michel Verhaegen and Vincent Verdult. *Filtering and system identification: a least squares approach*. Cambridge university press, 2007.

[19] Grace Wahba. A least squares estimate of satellite attitude. *SIAM review*, 7(3):409–409, 1965.

[20] Xiaolong Zhang, Eelis Peltola, and Jouni Mattila. Angle estimation for robotic arms on floating base using low-cost imus. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1458–1465. IEEE, 2018.

[21] Shiyu Zhao. Time derivative of rotation matrices: A tutorial. *arXiv preprint arXiv:1609.06088*, 2016.