NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

DANIEL MATIAS DE SOUSA

Master of Science in Electrical and Computer Engineering

MOVING HORIZON ESTIMATION SLAM FOR AGILE VEHICLES IN 3-D ENVIRONMENTS

MASTER IN ELECTRICAL AND COMPUTER ENGINEERING

NOVA University Lisbon
March, 2023

NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

DEPARTMENT OF ELECTRICAL
AND COMPUTER ENGINEERING

MOVING HORIZON ESTIMATION SLAM FOR AGILE VEHICLES IN 3-D ENVIRONMENTS

DANIEL MATIAS DE SOUSA

Master of Science in Electrical and Computer Engineering

Adviser: Bruno João Nogueira Guerreiro
Assistant Professor, Department of Electrical and Computer Eng. NOVA SCHOOL OF SCIENCE AND TECHNOLOGY

Examination Committee

Rapporteur: Doutor José António Barata de Oliveira
Associate Professor, Department of Electrical and Computer Eng. NOVA SCHOOL OF SCIENCE AND TECHNOLOGY

Adviser: Doutor Bruno João Nogueira Guerreiro
Assistant Professor, Department of Electrical and Computer Eng. NOVA SCHOOL OF SCIENCE AND TECHNOLOGY

MASTER IN ELECTRICAL AND COMPUTER ENGINEERING

NOVA University Lisbon
March, 2023

Moving Horizon Estimation SLAM for agile vehicles in 3-D environments

To my family.

Acknowledgements

I would like to express my gratitude to the following people, without whom I would not have been able to do my master's degree, as well as this dissertation. My supervisor, Prof. Bruno Guerreiro, for his knowledge, support and contribution, which I found unbelievably valuable and motivational. My friends and colleagues who supported me not only along this dissertation, but throughout the master's degree. And my biggest thanks to my family for the support and the huge investment they did on me to pursue this engineering degree.

This work was partially funded by the FCT project CAPTURE (PTDC/EEI-AUT /1732/2020) and project CTS (UIDB/EEA/00066/2020).

Abstract

The ability for a robot to be able to construct a map of the environment and recognize its position on it was one of the biggest developments in robotics. It gives them the possibility of being autonomous and safe, while creating new deployment opportunities where they were previously not feasible nor skillful enough to operate. The Simultaneous Localization and Mapping framework builds onto the perception of the robot, giving it the possibility to online calculate its trajectory and avoid obstacles. Because of it, there is now a large range of scenarios where robots can be used, ranging from a ship on open waters to a ground vehicle on mars. Moreover, the continuous development of processing units has given the possibility for previously hardware exhausting solutions to be used as an option for the localization and mapping problem.

With this in mind, the dissertation work is focused on developing a Simultaneous Localization and Mapping (SLAM) solution for a 6 Degrees of Freedom (DoF) vehicle operating on a 3D environment using Moving Horizon Estimation (MHE). Throughout the document it is presented relevant concepts to the modelling of 6 DoF vehicles as well as other approaches to the SLAM problem. It is also tested the applicability of the proposed solution, based on MHE, in a simulation environment of a 3D square-shaped corridor with stationary landmarks, whilst comparing the obtained results with other probabilistic approaches, the Extended Kalman Filter (EKF), which is commonly used but loses stability on extremely nonlinear dynamics, and the Linear Kalman Filter (LKF) sensor-based, which can also deal with the non-linearities of the system by characterizing it on the sensors frame. Each of the algorithms is simulated in MATLAB and their performance was compared considering two different Scenarios.

Keywords: Simultaneous Localization and Mapping, Moving Horizon Estimation, Extended Kalman Filter, Kalman Filter, Unmanned Aerial Vehicle

Resumo

A capacidade de um robô ser capaz de construir um mapa do ambiente que o envolve e reconhecer a sua posição no mesmo foi um dos maiores desenvolvimentos na área da robótica. Esta capacidade possibilita a que estes sejam autónomos e mais seguros, fomentando novas oportunidades de uso, para as quais a sua aplicação não era viável. A localização e mapeamento simultâneos (com acrónimo em inglês, SLAM) possibilita a perceção do robô, dando-lhe a capacidade de calcular, no momento, a sua trajetória evitando assim obstáculos. Por causa disto, há um grande leque de cenários onde os robôs podem ser aplicados, desde um navio em águas abertas até um veículo terrestre na superfície de Marte. Para além disto, o contínuo desenvolvimento de unidades de processamento possibilitou que soluções antigamente consideradas computacionalmente pesadas possam ser consideradas como solução para o problema de SLAM.

Com isto em mente, este trabalho de dissertação foca-se no desenvolvimento de uma solução SLAM baseada em estimação de horizonte movel (com acrónimo em inglês, MHE) para um veículo com 6 graus de liberdade (DoF) que opera num ambiente 3D. Ao longo do documento são apresentados conceitos relevantes para o desenvolvimento de modelos de veículos de 6 DoF bem como outras abordagens para o problema SLAM. Também é testada a aplicabilidade da solução MHE proposta num ambiente de simulação de um corredor 3D em forma de quadrado com marcos estacionários. A performance desta simulação é comparada aos resultados obtidos em outras duas abordagens probabilísticas, o filtro de kalman estendido (EKF) que é largamente utilizada mas que tem uma performance limitada aquando se encontra na presença de dinâmicas extremamente não lineares, e um filtro de kalman linear baseado nos sensores (sensor-based LKF), uma abordagem interessante que lida com as não linearidades do sistema caracterizando-o a partir do referencial do sensor. Cada um dos algoritmos é simulado no MATLAB e o seu desempenho foi comparado considerando dois cenários diferentes.

Palavras-chave: SLAM, MHE, UAV, EKF

Contents

List of Figures

List of Tables

Acronyms

C/GMRES Continuation/Generalized Minimum Residual *(pp. 7, 8)*

DoF Degrees of Freedom *(pp. ix, xi, 1, 4, 5, 7, 25, 27, 61, 69)*

EKF Extended Kalman Filter *(pp. ix, xi, xv–xvii, 1, 3–7, 9, 11, 17, 19, 27–29, 31, 35–39, 41–43, 53, 59–61, 63, 64, 66, 67, 69, 70)*

FI Full Information *(pp. 7, 18)*

GAS globally asymptotically stable *(pp. 7, 9, 51)*
GPS Global Positioning System *(p. 2)*

IEEE Institute of Electrical and Electronics Engineers *(p. 1)*
IF Information Filter *(p. 6)*
IMU Inertial Measurement Unit *(pp. 13, 26, 32)*

KF Kalman Filter *(pp. 6, 7, 9, 17, 51)*
KKT Karush-Kuhn-Tucker *(p. 9)*

LiDAR Light Detection and Ranging *(p. 1)*
LKF Linear Kalman Filter *(pp. ix, xi, xv–xvii, 1, 3, 4, 29, 51, 53, 54, 56–61, 63–67, 69, 70)*

MHE Moving Horizon Estimation *(pp. ix, xi, xv–xvii, 1–5, 7–9, 11, 18, 19, 27, 28, 31, 41–49, 59–61, 63–67, 69–71)*

PF Particle Filter *(pp. 6, 7)*
PTAM Parallel Tracking and Mapping *(p. 8)*

QP Quadratic Programming *(p. 9)*

RI-EKF Right Invariant-Extended Kalman Filter *(p. 6)*

SLAM Simultaneous Localization and Mapping *(pp. ix, xi, xv, xvi, 1–11, 13–18, 29, 31, 33, 41, 42, 44, 51, 57, 63–66, 69–71)*
SQP Successive Quadratic Programming *(pp. 4, 7, 9, 48, 49, 69, 70)*
SVD Singular Value Decomposition *(p. 55)*

UAV Unmanned Aerial Vehicle *(pp. xv, 1, 3, 4, 27)*
UKF Unscented Kalman Filter *(p. 6)*

vSLAM Visual Simultaneous Localization and Mapping *(p. 8)*

Symbols

B	Body frame *(p. 12)*
$\nabla \mathbf{f_x}$	Jacobian of $f(\hat{x}_{k\|k}, u_k)$ with respect to $x_{k\|k}$ *(p. 32)*
$\nabla \mathbf{h_x}$	Jacobian of $h(\hat{x}_{k+1\|k})$ with respect to x *(p. 33)*
$\nabla \mathbf{h_{x_L}}$	Jacobian of $h(\hat{x}_{k+1\|k})$ with respect to the landmarks part of the state vector x_{L_k} *(p. 33)*
$\nabla \mathbf{h_{x_v}}$	Jacobian of $h(\hat{x}_{k+1\|k})$ with respect to vehicles attitude part of the state vector x_{v_k} *(p. 33)*
T	Transpose *(p. 12)*
W	World frame *(p. 12)*

1 Introduction

The Simultaneous Localization and Mapping (SLAM) is a well-known robotics problem for, as the name implies, the simultaneous effort of mapping a not previously known environment and localizing the robot on the constructed map. The core of the problem, firstly introduced at the 1986 Institute of Electrical and Electronics Engineers (IEEE) Robotics and Automation Conference [1], is still a relevant issue with no final solution, as there have been a continuous research, for over three decades, on how to solve it considering distinct types of robot applications and environments. The best-known solution may differ, considering different types of robots and their surroundings.

This work is part of a project called CAPTURE [2] which views on challenging the typical airborne cargo transportation and urban mobility concepts, by reducing energy consumption. The idea behind the project is that an Unmanned Aerial Vehicle (UAV) can be able to launch and catch, on urban environments, a fixed wing aircraft with a long range mission assignment, as demonstrated in Figure 1.1. This will enable fixed wing aircrafts to deliver packages on an environment they previously could not, due to the lack of space to land safely.

On this document it is proposed a SLAM solution algorithm based on Moving Horizon Estimation (MHE) for a 6 Degrees of Freedom (DoF) vehicle in 3D space. To compare its performance and robustness, it will also be done a comparison with two other solutions: the Extended Kalman Filter (EKF) and the sensor-based Linear Kalman Filter (LKF). The proposed solution will be a probabilistic approach using the MHE method. It should be noted that this work was designed in a way that the algorithm can be implemented in any type of vehicle, as long as there are linear and angular velocity data of the body and some sort of landmark visualization, such as RGB-D cameras, Light Detection and Ranging (LiDAR) and other types of sensors. With a special emphasis on vehicles with nonlinear dynamics, such as UAV and satellites.

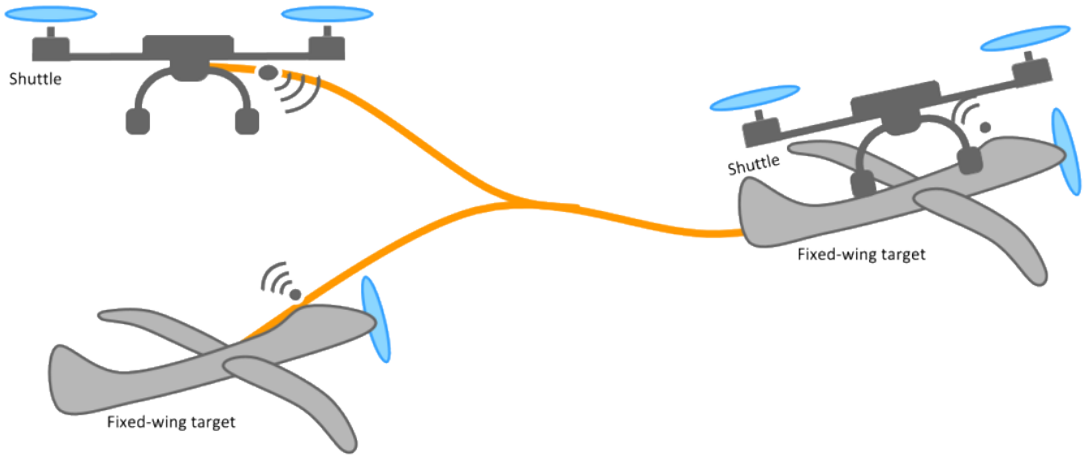


Figure 1.1: Project CAPTURE (retrieved from [2])

1.1 Motivation

Even though the SLAM has been addressed as a relative recent issue on robotics, the capability of mapping and localization was a problem that humankind had to deal with for a long time. Initially, with the help of landmarks that could be seen almost at any time and at any location, mostly stars like the north pole on the north hemisphere, and the sun. During the European Medieval Era, the ability of being able to read these maps, made of stars, was considered one of the seven mechanical arts. However, with time and the development of science and human desire of exploration, this human ability morphed from an art to a commonly taught science. In fact, the beginning of the Portuguese sea explorations was a catalyst for the boom of localisation technologies and the creation of an intercontinental map. The advancements made on navigation were critical for the hugely connected society we live in today, while also having a big impact on other key areas of human development. Therefore, it is also expected that the maturing of this concept on robotic systems will make possible their autonomy and, thus, many other new behaviours.

An autonomous robot is an intelligent system that can actuate without any human interference [3]. For mobile robots, this means that they need to have the ability of navigating any environment by themselves. For this, the robot needs to be able to precisely localise itself on the surrounding environment. Its' surroundings might be an indoor environment, where Global Positioning System (GPS) data might be non-existent or unreliable, an open sky, in which there are not many reference points to estimate the robot's position with visual sensors, or even a combination of both. Because of this, it is necessary to accurately estimate and fuse different data sources for better localization precision. The SLAM problems tries to tackle this need, but it has not been found any definitive solution, as researches are continuously developing better techniques for always emerging new challenges, with the end goal of creating robust algorithms for any scenario.

One of this algorithms is the MHE, which is an estimation method that has not been

used much on robotic systems SLAM due to its heavier computational needs. However, the tides are changing so much that robots use hardware with higher capacity and better workload, which might be sufficient for the use of such algorithm. This work aims to find out the viability of MHE on SLAM, and will compare its performance with two other solutions: a more commonly used method, the EKF, and a completely different approach using a LKF based on the sensor reference frame.

Figure 1.2: UAV (retrieved from [4])

Nonlinear dynamic vehicles are the major focus of research on SLAM applications, as there are already very robust solutions for the linear ones. Some of the vehicles with extremely high nonlinear dynamic systems are the UAVs and spacecrafts, which have seen a continuous increase on their use on the last fifteen years.

The exponential use of UAVs, also known as drones, on the past decade, can be explained by its use in many civil applications due to their ease of deployment, high mobility, low maintenance cost and ability to hover [5]. They have been applied on the agriculture industry, the mining sector, the transport branch and the security industry [6][7]. While some types of UAV's are operated through a remote control, others are completely autonomous and need a SLAM system to navigate through its environment.

Additionally, satellites have also seen a big increase on their deployment, namely nanosatellites like CubeSats, which suffered a decline after recording 300 launches in 2017, but are on track to a record breaking 633 launches in 2023 according to Kulu in [8].

1.2 Problem Statement

As mentioned before, the SLAM problem is still an open discussion with no definitive solution considering different environments. This is specially the case for vehicles with

nonlinear system dynamics. Considering the project CAPTURE, in which this thesis is integrated, the UAVs taken into account there have to deal with urban 3D environments, which is characterized by having nonlinear motion models and observation models, as it is an extremely dynamic environment with a high number of possible landmarks. In such environment it is possible that the sensing of some landmarks might be obstructed quite often.

1.3 Proposed Approach

This work studies the viability of a SLAM algorithm based on MHE applied to any type of rigid body, whilst having in consideration the high needs of computational power that the algorithm requires. The big advantage of this method is its applicability on nonlinear systems, since it does not have the linearity constraints imposed by other commonly used solutions, such as the EKF, which needs a local linearization. As such, the performance of the proposed solution will be compared, not only to a SLAM based EKF approach, but also with a sensor-based LKF.

The algorithm will be design to work on any 6 DoF robot in 3D space with known linear and angular velocities, which enables its deployment on the CAPTURE project. It will also be validated through simulation tests.

1.4 Document Outline

This document is structured as follows. Chapter 2 reviews the state of the art, mentioning probabilistic SLAM techniques, 3D visual SLAM applications and an overview over other techniques and frameworks. It also introduces the reader to nonlinear solvers, such as the Successive Quadratic Programming (SQP), active-set and interior-point. Chapter 3 introduces relevant concepts to a better understanding of the topics applied in this dissertation, such as the definition of the SLAM problem, the design of the EKF and MHE algorithms and the definitions and properties of orientation representations, such as: Euler angles, quaternions and rotation matrices. Chapter 4, presents the modelling of the dynamics of a rigid body with 6 DoF in 3D space using a quaternion approach. On Chapters 5, 6 and 7, it is presented the three approaches to tackle the described SLAM problem. The main one, an MHE algorithm with weights based on the EKF (Chapter 6), a common EKF algorithm (Chapter 5) and lastly, a sensor-based LKF approach (Chapter 7). On Chapter 8 the results of the models considered are taken into account as their differences in performance and error are showcased. The results are discussed having in mind their feasibility, robustness and applicability. Lastly, in Chapter 9, final observations are presented, as well as suggestions for future work regarding this topic.

2

State of the Art

As briefly said in the Introduction, the SLAM problem was first introduced in San Francisco at the 1986 IEEE Robotics and Automation Conference, with the recognition that probabilistic mapping was key for the development and applicability of autonomous robots in any environment. With this realization, this event turned out to be the ignition point of many of the breakthroughs achieved over the next years on trying to better define and solve the SLAM problem. The firsts major contributions to the maturing of this idea were made by Smith and Cheeseman [9], and Durrant-Whyte [10] on the articles where they define a statistical framework on how to describe and manipulate uncertainty between spatial points. During this period there were also other works that helped the development of the SLAM problem. One of them was written by Smith et al. [11], where it is described a representation for spatial information, the stochastic map, and shown that all the landmarks that one robot senses are highly correlated with each other, as every landmark estimation shares the error of the robots pose. Moreover, in [12], Cadena et al. discuss the past, present and future of SLAM. They particularly identify the key requirements of any modern SLAM application, being: robust performance, high-level understanding, resource awareness and task-driven perception the key characteristics. In light of that, this dissertation tries to tackle the robust performance issue in the context of highly nonlinear vehicles in 3D space.

Having in mind the considered objective of this work: solving the SLAM problem of a six DoF vehicle using a MHE approach, this Chapter presents solutions and methodologies on the topic. Nevertheless, and since there is a focus on different approaches to the SLAM problem, it will also be presented other methods of SLAM estimation, such as Filter-based SLAM algorithms, to which the EKF is part of. As well as an introduction to the graph-based SLAM and other frameworks of the problem, and nonlinear optimization methods.

2.1 Probabilistic approaches for SLAM

Some of the solutions for the SLAM problem are approached through probabilistic methods. These solutions are mathematical derivations of the Bayesian estimation rule [13][14]. The Bayes filters are a two-step process, the first step is to estimate the vehicle's pose and the map states, whilst the second is to compare that estimation with sensors data measurements, which will rectify the wrong predictions made on the first step [15]. This type of filters, also known as filter-based, have in consideration the uncertainties related to each of the sensors that it uses. Some of the most popular probabilistic approaches for SLAM are: Particle Filter (PF), Information Filter (IF), Kalman Filter (KF), EKF and the Unscented Kalman Filter (UKF). An overview of these filters was done by Bresson et al. [16] and Takleh et al. [17].

2.1.1 Kalman Filters

Invented by Kalman and Swerling [18][19], the KF is an optimal estimation technique for linear systems with addictive white Gaussian noise [20]. It utilizes a set of mathematical equations to obtain an efficient recursive solution of the least-squares method [21]. Due to its constraints in linearity and non-Gaussian disturbances [22][23], it is hardly used in SLAM algorithms, since the convergence of the filter cannot be guaranteed in these cases. However there has been developed derivations of the KF that have looser assumptions. The most popular ones for SLAM algorithms are the EKF and the UKF.

The EKF relaxes the linearity constraint imposed on the KF, while still considering that the processes are Gaussian [24]. It achieves this by doing local linearization of the motion and observability model [23][25], which makes it much more valuable and usable on SLAM problems, since it deals better with nonlinear robot motion. It is widely used as a SLAM solutions but it does not guarantee convergence nor consistency, as this can only be truly valid for linear models [26][27][28]. Studies on the convergence and consistency of a 3-D Right Invariant-Extended Kalman Filter (RI-EKF) SLAM were done by Zhang et al. in [29].

Another derivation of the KF, the UKF was introduced by Julier et al. in [30]. This filter, just like the EKF, performs a linearization on the system transition models. However, it achieves that by doing a stochastic linearization using a process based on a weighted statistical linear regression [24][27]. This linearization still has inconsistent results [31][28], but relaxes the Gaussian assumptions of the KF [27].

2.1.2 Particle Filters

The PF method is a set of sequential based Monte Carlo algorithms [32] that are used as an approximate inference method for calculating the posterior distribution of the system's processes, a controllable and non-totally observable Markov chain [33]. Because of this property, it is irrelevant for this type of filters whether the system is nonlinear,

non-Gaussian or both. Besides this, PFs also give the robot the capability to recover its pose even when under global uncertainty [33]. One of the most relevant realizations of PFs is the FastSLAM, which was introduced by Michael Montemerlo et al. [34]. In his work he integrates the PF and KF representations, with the use of a Rao-Blackwellized representation of the posterior, and states that the time required to update the posterior is O($MlogK$), in which M is the number of particles and K the number of landmarks, while a KF based methodology would take O(K^2). However, it is also known that the FastSLAM algorithm degenerates and loses consistency over the time [26]. Besides this, it is known that this method takes a lot more time to recover from poor initial estimations than the EKF and MHE methods [35].

There is also an improved version of FastSLAM called FastSLAM 2.0 [26], which contemplates the problems with data association of the previous [36]. Several works use this technique [37], one of them, by Saiadek et al. [38] compares the use of EKF and FastSLAM on outdoor environments, considering a Gaussian and a Non-Gaussian model of the noise.

2.1.3 Moving Horizon Estimation approach for SLAM

Another way of solving the maximum a posteriori estimate is to maximize the joint probability for a given trajectory of state values. The MHE is a constrained [39], nonlinear optimization problem, that unlike the KF and EKF, does not assume the noise to be Gaussian [40]. Similarly to the Full Information (FI), the MHE has in consideration the measured data at the current and previous steps. However, it uses a small window of measured data to make its' estimations, which makes it a more efficient and less exhaustive estimator as the FI considers every past estimation. Changing the size of the horizon creates a compromise between the computational requirements and the performance of the method. In spite of that, the MHE still has a high computational cost [35]. Moreover, the stability of the MHE for linear and nonlinear systems has been studied in [41], and by Rao et al. in [42] and [43], where it is showed that this technique is an asymptotically stable observer in a nonlinear deterministic modelling framework. Moreover, Müller M. in [44] found that the MHE for nonlinear detectable systems with bounded disturbances is robust and globally asymptotically stable (GAS).

To the best of the author's knowledge there is still a limited number of work done regarding the use of MHE on SLAM. Moreover, there are no applications of this technique on a vehicle with complex dynamics, such as vehicles with six DoF, which demonstrates the relevance of this work. However there are some use cases that should be pointed out. On the work of Kasahara et al. [45] where it's considered the effectiveness of a MHE approach to the SLAM of a tricycle in a crowded environment, they try to reduce the computational cost by applying the SQP optimization algorithm. On [46] this optimization was done using the Continuation/Generalized Minimum Residual (C/GMRES) algorithm, in which Kishimoto et al. apply the MHE technique on multi-robot SLAM, as a way to fuse

the data from all the robots. In their work they evaluate the computational cost of using the C/GMRES as better than with a quasi-Newton approach. The C/GMRES method is often used as an online optimization in nonlinear model predictive control systems [47].

Besides the use of this technique on SLAM, there are other relevant applications such as the work done by Ferrari-Trecate et al. in [48] and Bemporad et al. [49], in which they exploit the use of MHE on hybrid systems. Ferrari-Trecate et al. uses MHE as a state-smoothing algorithm whilst guaranteeing asymptotic convergence of the MHE filter. Moreover in [50], Zou et al. present the current state, opportunities and challenges of using MHE as a multi-sensor information fusion method. Additionally, another application that should be mentioned is the work done by Haseltine et al. in [51], which demonstrates examples of estimation failure in the EKF algorithm, namely when there are constraints at play, and how the MHE performs in the same conditions. They are able to conclude that the MHE algorithm consistently delivers better state estimations and greater robustness. Lastly, Farina et al. in [52] studies the use of MHE as a distributed estimator for linear constrained systems.

2.2 Graph-based SLAM

There are also other solutions which aren't based on the probabilistic approach. The Graph-based SLAM is one of them, as it is optimization-based and uses a graph to represent the mapping problem. In this formulation the nodes are the vehicle's pose and the edges the measurements between these poses. In a similar way to the filter-based formulation, the optimization-based is divided in two distinct parts. The first step predicts the constraints of the vehicle pose to the measurements of the sensor. The second part, minimizes the error between the sensor readings and the estimated positions of previous and current vehicle poses in relation to the map. The Graph-based SLAM is a well-known optimization-based approach and an introduction and tutorial about it was done by Grisetti et al. in [53].

Regarding relevant use cases, Labbe et al. studied the applicability of this method with an online global loop closure detection approach, which is useful when the drone has an unknown initial position [54]. Also, Kümmerle et al. in [55] considered the use of the graph-based SLAM formulation to fuse satellites' images and the information from the vehicle sensors.

2.3 Other Frameworks

As it was explained before, the SLAM framework is based on estimating both, the pose and the map simultaneous and cooperatively. However there are also other techniques that tackle the mapping and localization as parallel tasks, usually defined as Parallel Tracking and Mapping (PTAM) [56] they're consider to be more time-efficient. Besides that, there are also other methods called Visual Simultaneous Localization and Mapping (vSLAM) and Sensor-based SLAM. The vSLAM uses image sensors, such as RGB cameras, and

computes the camera pose using visual information of the landmarks positioning [57]. Successful uses of it can be found in [58] and [59].

The sensor-based approach tackles the filtering problem under a not so common perspective. Unlike the typical representation of the state variables in an inertial reference frame, usually called world-fixed frame, the sensor-based uses the vehicle sensor as the reference frame, thus being called sensor-based. Even though, the sensor-based is not the most widely used framework, there has been some research on its implementation over the years. One of which by Castellanos et al. in [60], where the authors use the robocentric map joining design to tackle consistency problems of the EKF-based SLAM. Further endeavours were made on the sensor-based framework, namely the one developed by Guerreiro et al. in [61]. In which the authors are able to design a GAS KF sensor-based SLAM filter for nonlinear system dynamics. This is possible due to the sensor-based approach enabling the reformulation of the nonlinear system dynamics into a linear time-varying system. Furthermore, Lourenço et al. built upon this framework, which resulted in a 3-D sensor-based GAS filter for aerial vehicles SLAM [62].

2.4 Nonlinear Optimization methods

This work focuses on solving the SLAM problem using the MHE approach, as such an optimization algorithm will be needed to find the optimal solution for solving the cost function. Even though, the MHE is an estimation method with a few static characteristics, there are also others that might differ depending the application of the algorithm, such as size, cost function, constraints and the considered models. As a result, there is not a specific numerical method to solve it [63]. Besides that, the large number of variables to be estimated and the common ill-posed formulations of the MHE can lead to powerful nonconvexities [64][65].

The SQP defines a sequence of quadratic programming sub-problems and linearizes the Karush-Kuhn-Tucker (KKT) conditions at each step, resulting in a set of linear conditions that can be used on a Quadratic Programming (QP) method [66]. This solution is known for handling better a large number of constraints [67]. A deep introduction to SQP methods can be found in [68] and [69]. Moreover, Boggs et al. in [70] present the SQP algorithm with an emphasis on large-scale nonlinear problems.

The active-set algorithm is also a method to have in consideration, as it is widely used to solve nonlinear optimization problems with constraints [71]. It works in a similar manner to the SQP, but with a set of active and inactive constraints. At each iteration, a QP sub-problem is solved having in consideration the set of active constraints. The algorithm terminates if this solution satisfies the non-active constraints. Otherwise, the set of active constraints is updated with inactive constraints and the process is repeated until it converges into an optimal solution [72].

Another algorithm to consider is the interior-point method. In this method, the KKT conditions regarding the inequality constraints suffer a smooth approximation as they are

handled by a logarithmic barrier function [63]. In [73] Potra et al. present and discuss the interior-point methods in more detail. On top of that, Byrd et al. in [74] formulate an interior-point algorithm for large-scale nonlinear programming.

Having considered the state-of-art of SLAM technologies, there is still the need to introduce the reader to some relevant concepts. Wich will be done on the next Chapter.

3

Conceptual Background

On this Chapter the reader is introduced to relevant concepts to the work done throughout this dissertation, namely the definitions of a SLAM problem and the EKF and MHE algorithms. Their designs and implications are also presented and discussed. Besides that, it is also presented the definitions and properties of three-dimensional rotational representations, namely the 3-dimensional rotation matrix, quaternions and the Euler angles, whilst demonstrating the conversion from one representation to the other and showcasing mathematically deductions relevant to the definition of the system dynamics in Chapter 4.

3.1 Notation

In order to better comprehend the mathematical expressions throughout this document some abbreviations were taken into account.

- Every vector is written in a bold lower case and each matrix on a bold higher case, like $\mathbf{v}$ and $\mathbf{M}$, respectively. Whilst the scalar values are represented by plain letters;
- The set of real numbers is defined by $\mathbb{R}$, and $\mathbb{R}^{n\times m}$ is the set of matrices of size $n \times m$ with real values;
- The symbols $\mathbf{0_{m\times n}}$ and $\mathbf{I_m}$ represent a matrix of zeros with dimensions $m \times n$ and an identity matrix with dimensions $m \times m$;
- A diagonal matrix of values $(a_1, a_2, ..., a_n)$ on its main diagonal is defined as $diag(a_1, a_2, ..., a_n)$ and a block diagonal matrix is denoted as $blkdiag(\mathbf{N_0}, \mathbf{N_1}, ..., \mathbf{N_i})$, in which $\mathbf{N}$ is a square matrix;
- $\mathbb{H}$ defines the quaternions group, and $\mathbb{H}_p$ the group of pure quaternions

$$\mathbb{H}_p = \left\{\mathbf{q} \in \mathbb{H} \mid \mathbf{q} = (0, \mathbf{q}_v)\right\}, \mathbf{q}_v \in \mathbb{R}^3 \tag{3.1}$$

- The sympletic group, $Sp(1)$, is the orthogonal group formed by the unitary quaternions, which is isomorphic to the unit sphere in $\mathbb{R}^4$, S^3 [75]

$$S^3 = \left\{ \mathbf{q} \in \mathbb{H} \mid |\mathbf{q}| = 1 \right\} \tag{3.2}$$

- The special orthogonal group in 3-D space is defined as

$$SO(3) := \left\{ \mathbf{R} \in \mathbb{R}^{3\times3} \mid \mathbf{R}^T\mathbf{R} = \mathbf{I}_3, det(\mathbf{R}) = 1 \right\} \tag{3.3}$$

  where T is the transpose superscript;

- $\mathbf{S}[.]$ is a map from $\mathbb{R}^3$ to the space of three-by-three skew-symmetric matrices, $so(3)$,

$$\mathbf{S}[\mathbf{v}] = \begin{bmatrix} 0 & -v_1 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}, \mathbf{v} = \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix} \in \mathbb{R}^3 \tag{3.4}$$

$$\mathbf{S}[\mathbf{v}]\mathbf{u} = -\mathbf{S}[\mathbf{u}]\mathbf{v} \tag{3.5}$$

- $\mathbf{S}'[.]$ is a map from $\mathbb{H}_p$ to $so(3)$

$$\mathbf{S}'[\mathbf{v}] = \begin{bmatrix} 0 & -q_1 & -q_2 & -q_3 \\ q_1 & 0 & q_3 & -q_2 \\ q_2 & -q_3 & 0 & q_1 \\ q_3 & q_2 & -q_1 & 0 \end{bmatrix}, \mathbf{q} = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix} \in \mathbb{H}_p \tag{3.6}$$

- ${}^{W}_{B}\mathbf{R}$ is the rotation matrix from B, the Body frame, to W, the World frame;
- Another relevant notation is the use of vector weighted norm, $\left\|\mathbf{v}\right\|_{\mathbf{\Gamma}}$ defined by

$$\left\|\mathbf{v}\right\|_{\mathbf{\Gamma}} = \mathbf{v}^T.\mathbf{\Gamma}.\mathbf{v} \tag{3.7}$$

  with $\mathbf{\Gamma} \in \mathbb{R}^{m\times m}$ and $\mathbf{v} \in \mathbb{R}^{m\times1}$.

Besides that, it is assumed the following discrete-time system

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_{k+1}) + \mathbf{n}_{u_{k+1}} \tag{3.8}$$

$$\mathbf{y}_{k+1} = \mathbf{h}(\mathbf{x}_{k+1}) + \mathbf{n}_{y_{k+1}} \tag{3.9}$$

where:

- $\mathbf{x}_k$: is the state vector, compromised of the vehicle pose and landmarks, $\mathbf{m}_i$. Normally the size of this vector is dynamic, since it changes with the number of landmarks considered;

- $\mathbf{u}_{k+1}$: is the control action at $k+1$, whose input will induce the vehicle from state $\mathbf{x}_k$to the state $\mathbf{x}_{k+1}$. This information usually comes from sensors data, such as an Inertial Measurement Unit (IMU);

- $\mathbf{m}_i$: the location of landmark number i. This vector is part of the state vector and may change at every k instant, given if a new landmark is observed or not;

- $\mathbf{y}_k$: represents the set of measurements for the landmarks that were observed at the instant k. Similarly to $\mathbf{m}_i$, this vectors size is dynamic, as its length is determined by the number of sensor readings. But it should be noticed that it doesn't necessarily need to have the same size as $\mathbf{m}_i$, since a new landmark might have been identified, or a landmark previously identified has not been seen by the sensor;

- $\mathbf{n}_{y_k}$ represents the measurement noise, and is a zero mean Gaussian motion disturbance which is considered to have time-invariant density and a standard deviation of σ_y;

- $\mathbf{n}_{u_k}$ represents a random variable called process noise, and is a zero mean Gaussian motion disturbance with time-invariant density and a standard deviation of σ_{n_u};

- $\mathbf{f}(\mathbf{x}_k, \mathbf{u}_{k+1})$ is the system model, which characterizes the motion dynamics of a vehicle given a probabilistic approach to the SLAM problem. Assumed here to be nonlinear, since that is the focus of this thesis;

- $\mathbf{h}(\mathbf{x}_{k+1})$ is the measurement function, which , on the probabilistic approach to SLAM, mirrors the transformation of the sensor readings to an actual location of a given landmark.

3.2 The SLAM Problem

The general structure of the SLAM problem takes the representation shown at Figure 3.1, where it is possible to visualize the robot, landmarks, the uncertainty related to the robots pose and landmark positioning and the correlation between the error of the estimated positions [76].

As previously stated, the SLAM problem aims to create a map of the surrounding environment and also estimate the pose of the robot in every instant. From a probabilistic point of view, there are two main resolutions for the SLAM problem with great importance depending on the use case. The online SLAM, Figure 3.2a, which involves in finding an estimation for the joint posterior density of the momentary pose and map [24]

$$p(\mathbf{x}_{k+1}|\mathbf{y}_{0:k}, \mathbf{u}_{0:k}, \mathbf{x}_0) \tag{3.10}$$

and the Full SLAM is in Figure 3.2b. In this formulation the posterior is calculated for the

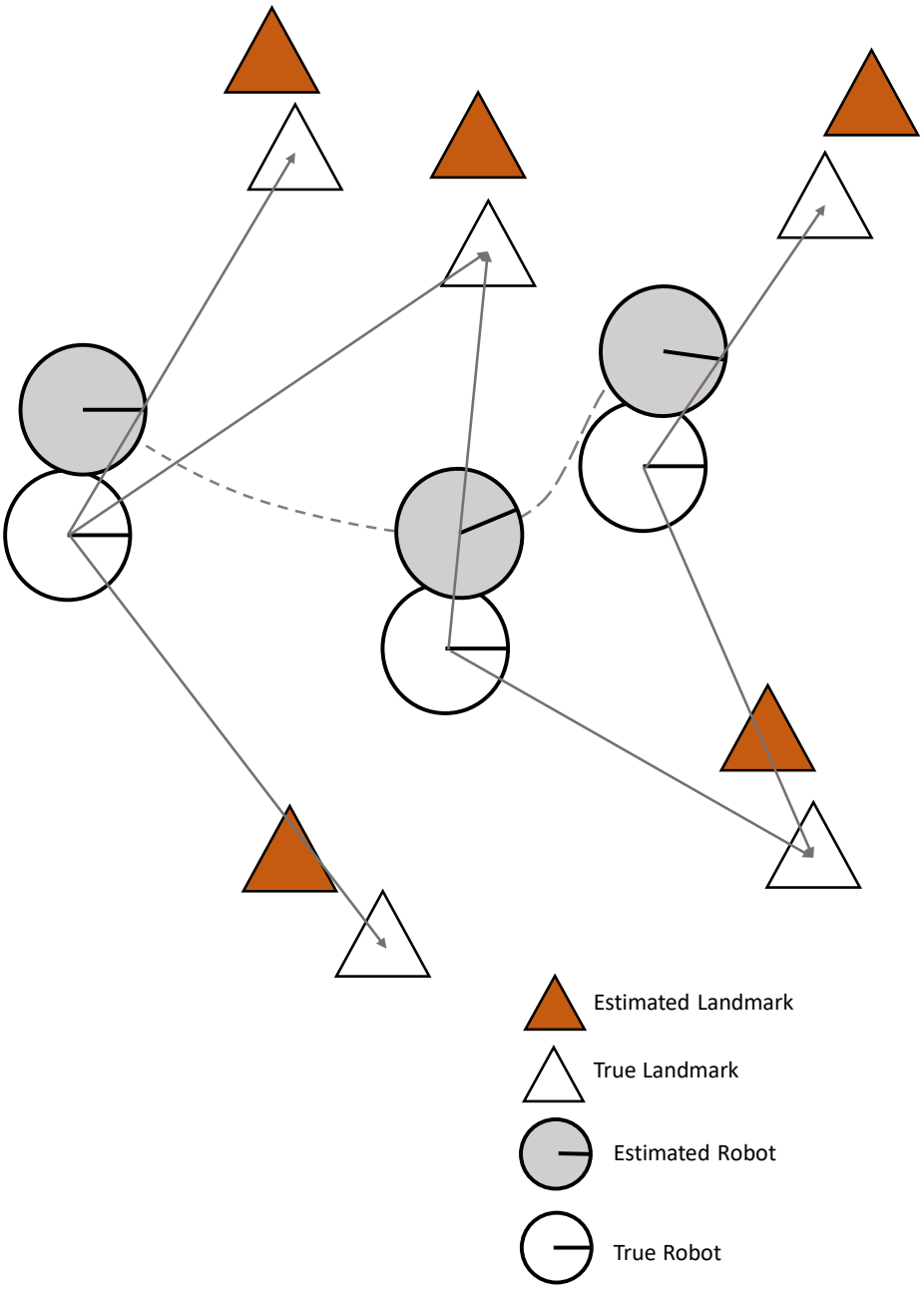
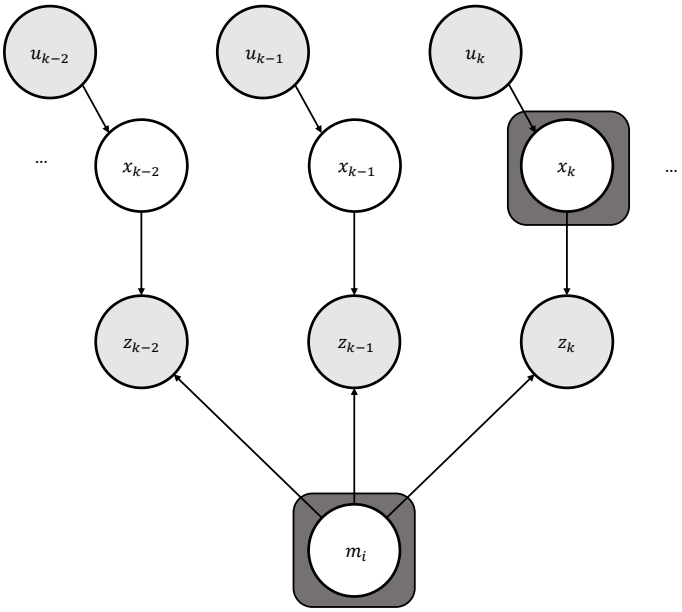
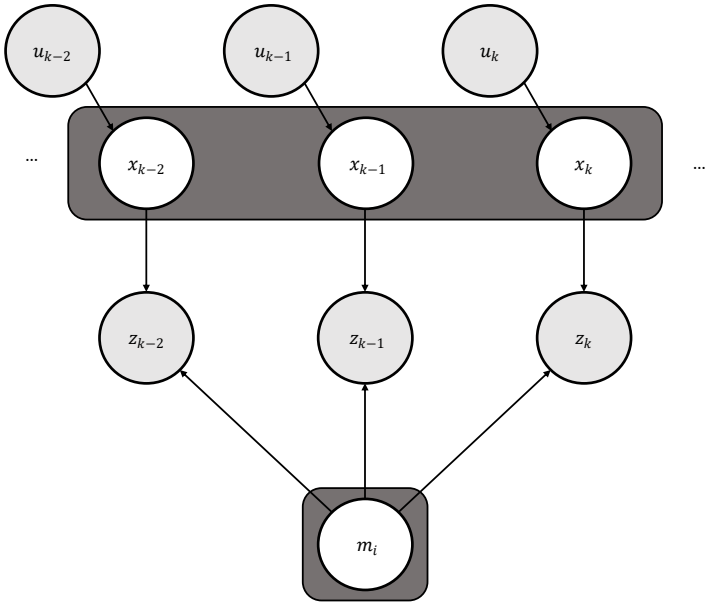


Figure 3.1: The SLAM problem (adapted from [1])



(a) Online SLAM



(b) Full SLAM

Figure 3.2: Probabilistic formulations of the SLAM (adapted from [16])

entire trajectory of $\mathbf{x}_k$ alongside the map [24]

$$p(\mathbf{x}_{0:k+1}|\mathbf{y}_{0:k}, \mathbf{u}_{0:k}, \mathbf{x}_0). \tag{3.11}$$

Taking into account the system presented in Equations (3.8) and (3.9), it should be pointed out the non-dependency between the control input and the observation sensors data. Therefore, it is possible to define a state transition model and an observation model. As so, considering the case of online SLAM, the motion model can be represented as

$$p(\mathbf{x}_{k+1}|\mathbf{x}_k, \mathbf{u}_k) \tag{3.12}$$

whilst the observation model as

$$p(\mathbf{y}_k|\mathbf{x}_k). \tag{3.13}$$

The online SLAM from a probabilistic point of view can be implemented, using the Bayes Theorem, in a two-step recursive prediction correction form. Using

$$p(\mathbf{x}_{k+1}|\mathbf{y}_{0:k}, \mathbf{u}_{0:k}, \mathbf{x}_0) = \int p(\mathbf{x}_{k+1}|\mathbf{x}_k, \mathbf{u}_k)p(\mathbf{x}_{k+1}|\mathbf{y}_{0:k}, \mathbf{u}_{0:k}, \mathbf{x}_0)d\mathbf{x}_k \tag{3.14}$$

$$p(\mathbf{x}_{k+1}|\mathbf{y}_{0:k+1}, \mathbf{u}_{0:k}, \mathbf{x}_0) = \frac{p(\mathbf{y}_{k+1}|\mathbf{x}_{k+1})p(\mathbf{x}_{k+1}|\mathbf{y}_{0:k}, \mathbf{u}_{0:k}, \mathbf{x}_0)}{p(\mathbf{y}_{k+1}|\mathbf{y}_{0:k}, \mathbf{u}_{0:k})} \tag{3.15}$$

the time-update and the measurement-update, respectively, it is possible to calculate the joint posterior $p(\mathbf{x}_{k+1}|\mathbf{y}_{0:k}, \mathbf{u}_{0:k}, \mathbf{x}_0)$.

The diagram of a probabilistic SLAM can be represented as depicted in Figure 3.3. The observed landmarks at each given instant are used to correct the measurements from the navigation sensors, which were used to make initial guesses of the current position of the vehicle. The observation data from the sensors goes through a feature extraction and a scan matching algorithm in order to identify them and characterize their quality as a feature on the map. Only high-quality features are considered. If an observation matches with an already known landmark, the position of that feature on the map is refined taking into account the current estimation of the vehicles pose and the measurement. In opposition, if the feature is not yet considered on the map (is not part of the state vector), it means that it is a new landmark and as such the state vector has to be augmented.

The reason behind the existence of the SLAM problem lies on the uncertainties inherent to every sensor, which, even though small, make the overall positional uncertainty grow larger with time. The SLAM tackles this by fusing and estimating the position through the methods mentioned before in Chapter 2. However these solutions also have some problems/considerations by their own, namely: correspondence, data association and time complexity problems [77].

- **Correspondence** might be considered the biggest problem in SLAM [78]. This builds upon the necessity that SLAM algorithms need to be able to distinguish between features, so that one landmark is not identified as another. This is especially hard when the environment information is collected by a laser sensor, which can only really differentiate landmarks by their shape and volume [77].

- **Data Association** is also a key problem to have in consideration, as it is very common for a vehicle to return to a part of the map that it had previously been on. For the loop closure to happen, the revisited landmarks have to be recognised and correctly associated with their already estimated position on the map [78].

- **Time Complexity** issues are related to how fast the whole SLAM process needs to be executed. Dealing with the uncertainty of sensors, having the right identification of landmarks and doing the data association takes time, which has to be minimum if the vehicle that the algorithm is being applied to is fast and very dynamic. As a result, algorithms have to be quick to respond to these necessities [78].

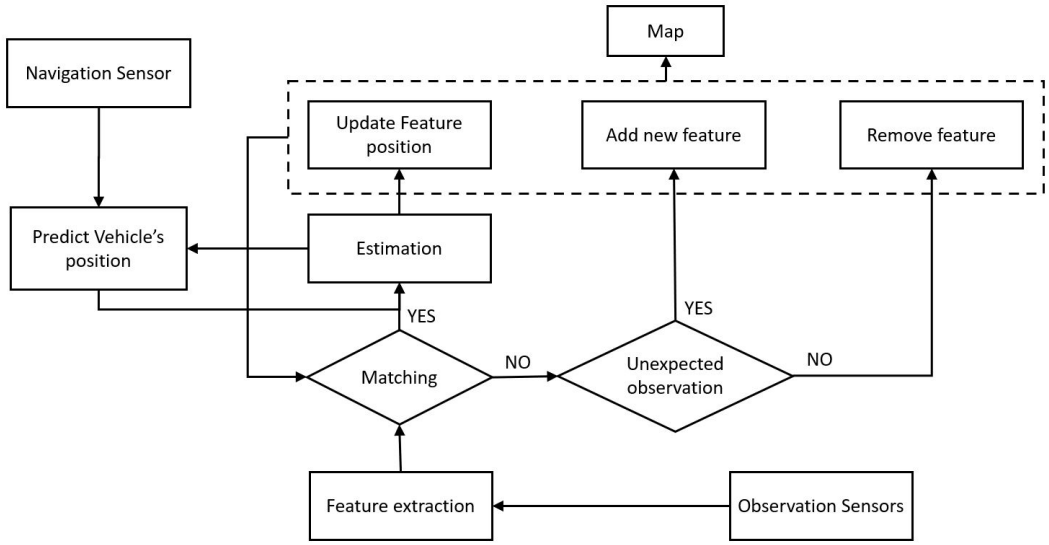


Figure 3.3: The SLAM problem as a diagram (adapted from [77])

3.3 Extended Kalman Filter

The EKF is part of a collective called *Gaussian Filters*. This type of filters are an Implementation of the Bayes Filter and all share the principle that beliefs are represented by multivariate normal distribution. This filter was derived from the KF, which is a recursive algorithm used to estimate the state of a linear system based on measurements with Gaussian noise. The major difference between them is that the EKF relaxes the linearity assumptions of the previous [23], which were very strict, whilst maintaining the belief that noise is Gaussian [24]. It achieves this by approximating, for each sampling time, the nonlinear system with a first-order Taylor series expansion. The EKF algorithm is known as efficient and computationally inexpensive, making it a popular choice for many estimation applications. Since it is an implementation of a Bayesian Filter, the EKF is comprised of two steps on its core: the prediction step and the update step. As mentioned

previously, the prediction step uses the motion model to make initial predictions, and the update step uses the observation model to correct them.

3.4 Moving Horizon Estimation

Another way of solving the maximum a posteriori estimate of the probabilistic approach to SLAM is to maximize the joint probability of a given trajectory of state values by treating it as an optimization problem. The FI estimator does this by considering every state since t_0

$$\{\mathbf{x}_0, ..., \mathbf{x}_T\} = \arg \max_{\mathbf{x}_0,...,\mathbf{x}_T} p(\mathbf{x}_0, ..., \mathbf{x}_T | \mathbf{y}_0, ..., \mathbf{y}_T). \tag{3.16}$$

However, the computational complexity of calculating the maximum posterior gets larger in each iteration, since the number of variables to be estimated grows. As a measure of tackling this issue a finite horizon can be defined so the estimation method will only consider the last N measurements.

$$\{\mathbf{x}_0, ..., \mathbf{x}_{T-N}\} = \arg \max_{\mathbf{x}_{T-N-1},...,\mathbf{x}_T} p(\mathbf{x}_{T-N}, ..., \mathbf{x}_T | \mathbf{y}_0, ..., \mathbf{y}_T). \tag{3.17}$$

This technique is called MHE and is a very well-known estimation approach for both linear and nonlinear systems. Similar to the FI estimator it is numerically equivalent to a constrained, nonlinear optimization problem: it estimates the system's state by minimizing the difference between the system's predicted output and the actual output over a certain period of time, which can also be refered to as the Horizon. The representation of such calculations is displayed in Figure 3.4.

The maximization in Equation (3.17) can be found through the minimization of the arrival cost function J

$$\min_{\mathbf{x_{T-N}},...,\mathbf{x_T}} J \tag{3.18}$$

where J has in consideration the difference between

- the relation between the estimated motion model dynamics over the horizon;
- the measured variable and the measurement model predicted value;
- the previously predicted estimations and the current estimation step.

The most common nonlinear moving horizon estimator solves the constrained optimization problem defined by

$$\min_{\mathbf{x}_k, \mathbf{n}_{u_k}, \mathbf{n}_{y_k}} \phi = \hat{l}_0(\mathbf{x}_0) + \sum_{k=0}^{H} \hat{l}_k(\mathbf{n}_{u_k}, \mathbf{n}_{y_k}) \tag{3.19}$$

$$s.t. \qquad \varphi_k(\mathbf{x}_k, \mathbf{n}_{u_k}) \geq d_k \tag{3.20}$$

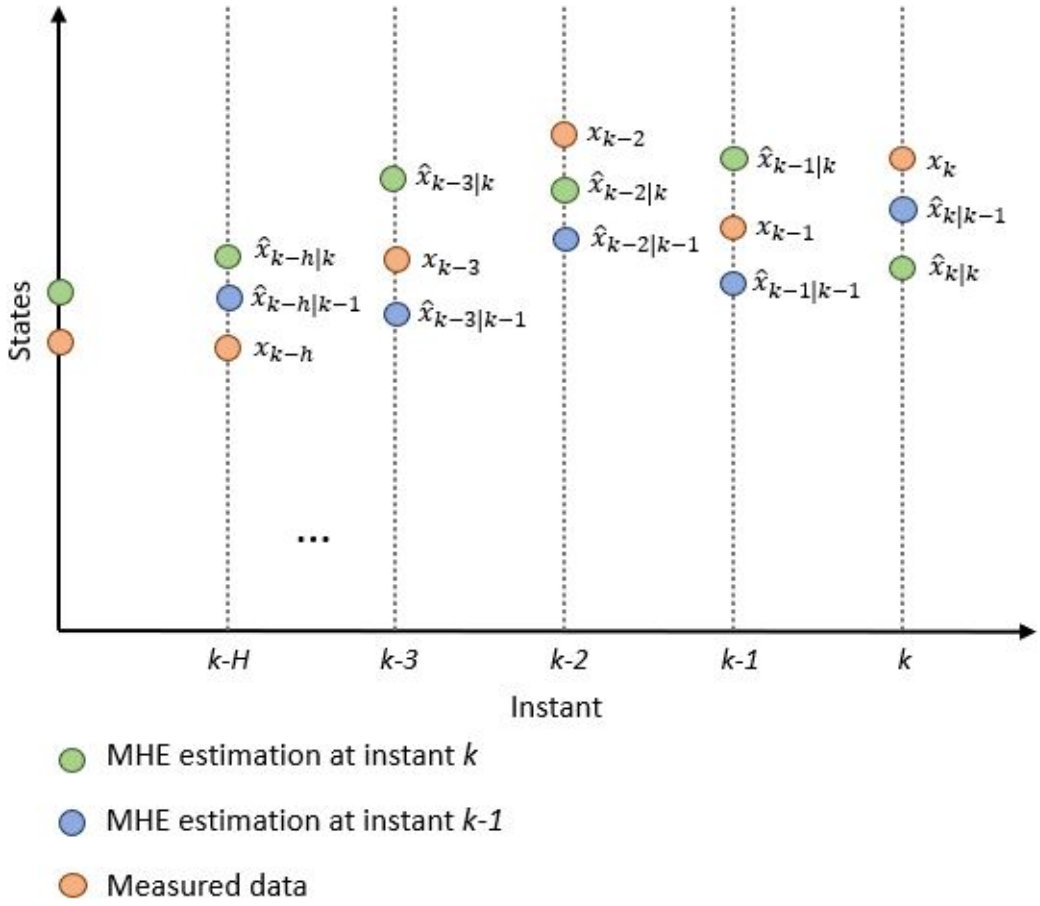


Figure 3.4: The MHE algorithm (adapted from [45])

where H is the size of the Horizon and the stage and final cost are respectively defined as weighted least squares [79], given by

$$\hat{l_0}(\mathbf{x}_0) = \frac{1}{2}(\mathbf{x}_0 - \hat{\mathbf{x}}_0)^T \hat{\mathbf{P}}_0^{-1}(\mathbf{x}_0 - \hat{\mathbf{x}}_0) \tag{3.21}$$

and

$$\hat{l_k}(\mathbf{n}_{u_k}, \mathbf{n}_{y_k}) = \frac{1}{2}\begin{bmatrix}\mathbf{n}_{u_k} - \hat{\mathbf{n}}_{u_k} \\ \mathbf{n}_{y_k} - \hat{\mathbf{n}}_{y_k}\end{bmatrix}^T \begin{bmatrix}\hat{\mathbf{Q}}_k & \hat{\mathbf{M}}_k \\ \hat{\mathbf{M}}_k^T & \hat{\mathbf{R}}_k\end{bmatrix}^{-1} \begin{bmatrix}\mathbf{n}_{u_k} - \hat{\mathbf{n}}_{u_k} \\ \mathbf{n}_{y_k} - \hat{\mathbf{n}}_{y_k}\end{bmatrix}. \tag{3.22}$$

in which $\mathbf{Q}$, $\mathbf{M}$ and $\mathbf{R}$ are weigths.

This type of cost function (3.19) takes into account the state x_0, the sequence of measurement noise and the sequence of process noise in the considered horizon H, with the aim of minimizing it so that it is as close as possible to the sensor measurements whilst respecting the constraints imposed in (3.20). The criteria for a good fit is specified by (3.21) and (3.22) which state that $\hat{\mathbf{x}}_\mathbf{0}$ should be as near to $\mathbf{x}_0$ as possible and that the estimated noise sequences should also be as close as possible to their true values, respectively [79].

In this method, and in opposition to the EKF approach, there is no assumption about normally distributed noises.

3.5 3D Orientation representation

In this section are presented different forms of representing the orientation of a body in space, namely, the rotation matrices, Euler angles and quaternions.

3.5.1 Rotation Matrices

The rotation matrix is a transformation matrix of a rotation on the Euclidean space. A rotation matrix, $\mathbf{R} \in SO(3)$, can be interpreted as a rotation of angle θ around the axis defined by the unitary vector $\mathbf{u} = (u_x, u_y, u_z)$ and can be represented as

$$\mathbf{R} = \begin{bmatrix} \cos\theta + u_x^2(1-\cos\theta) & u_x u_y(1-\cos\theta) - u_z\sin\theta & u_x u_z(1-\cos\theta) + u_y\sin\theta \\ u_x u_y(1-\cos\theta) + u_z\sin\theta & \cos\theta + u_y^2(1-\cos\theta) & u_y u_z(1-\cos\theta) - u_x\sin\theta \\ u_x u_z(1-\cos\theta) + u_y\sin\theta & u_y u_z(1-\cos\theta) + u_x\sin\theta & \cos\theta + u_z^2(1-\cos\theta) \end{bmatrix} \tag{3.23}$$

or as according to Rodrigues' rotation formula [80]

$$\mathbf{R} = \mathbf{I} + (\sin\theta)\mathbf{S}[\mathbf{u}] + (1-\cos\theta)\mathbf{S}[\mathbf{u}]^2. \tag{3.24}$$

Given any rotation matrix $\mathbf{R} \in R^n$, $\mathbf{R}$ is an orthogonal matrix and therefore has the following property

$$\mathbf{R}^T = \mathbf{R}^{-1}. \tag{3.25}$$

The conversion from Euler angles to rotation matrix is also an important topic to highlight, an explanation of such transition is described by Slabaugh in [81].

3.5.2 Euler Angles

The Euler angles are a representation of a rigid body orientation on three-dimensional space, by a set of three different angles (ϕ, θ, ψ). These angles represent three successive rotations around different axes. Considering the "x-convention" (ZXZ), as it is the most common one, the orientation given by the Euler angles can be defined in the following order [82]:

- rotation of ϕ around the z-axis;
- rotation by an angle θ around the former x-axis (x-axis of the coordinate system which resulted from the first rotation);
- rotation of ψ about the former z-axis (z-axis of the coordinate system which resulted from the first two rotations).

The first and third rotation can be defined as

$$\mathbf{R_Z}(\Omega) = \begin{bmatrix} cos(\Omega) & sin(\Omega) & 0 \\ -sin(\Omega) & cos(\Omega) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.26}$$

whilst the second rotation can be established by

$$\mathbf{R_X}(\Omega) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\Omega) & sin(\Omega) \\ 0 & -sin(\Omega) & cos(\Omega) \end{bmatrix}. \tag{3.27}$$

In other notations a rotation around the y-axis might be used, which can be represented as

$$\mathbf{R_Y}(\Omega) = \begin{bmatrix} cos(\Omega) & 0 & sin(\Omega) \\ 0 & 1 & 0 \\ -sin(\Omega) & 0 & cos(\Omega) \end{bmatrix}. \tag{3.28}$$

The combination of the Euler sequence defined on the convention ZXZ, $R_{z,x,z}(\phi, \theta, \psi) = R_z(\psi)R_x(\theta)R_z(\phi)$, can be simplified by the following matrix

$$\begin{bmatrix} \cos\phi\cos\psi - \cos\theta\sin\phi\sin\psi & -\cos\phi\sin\psi - \cos\theta\cos\psi\sin\phi & \sin\theta\sin\phi \\ \cos\psi\sin\phi + \cos\phi\cos\theta\sin\psi & \cos\phi\cos\psi - \cos\theta\sin\phi\sin\psi & -\cos\theta\cos\phi \\ \sin\theta\sin\psi & \cos\psi\sin\theta & \cos\theta \end{bmatrix}. \tag{3.29}$$

3.5.3 Unit Quaternions

Quaternions, which were introduce by Hamilton, W. in [83], are 4-dimensions vectors commonly used in robotics for calculations of three-dimensional rotations. As such, they are seen as alternatives to other three-dimensional representations like the Euler angles and rotations matrices.

The quaternions are normally defined as

$$\mathbf{q} = [a, b, c, d]^T = a + bi + cj + dk \tag{3.30}$$

in which a, b, c and d are real numbers and i, j, k are the quaternions units.

Quaternions have some unique, or not so common, properties. An important characteristic to have in mind in this work is the unitary quaternion, $q \in S^3$, which is a quaternion of norm equal to one

$$\|\mathbf{q}\| = \sqrt{a^2 + b^2 + c^2 + d^2} = 1. \tag{3.31}$$

The complex conjugate of a quaternion is the same as the quaternion with its vector part multiplied by -1

$$Conj(\mathbf{q}) = \mathbf{q}^* = [a, -b, -c, -d]^T. \tag{3.32}$$

Moreover, the inverse of the quaternion is also defined as the inverse of a normal complex number

$$Inv(\mathbf{q}) = \mathbf{q}^{-1} = \frac{\mathbf{q}^*}{\|\mathbf{q}\|^2} \tag{3.33}$$

Furthermore, if $q \in S^3$ then

$$Inv(\mathbf{q}) = Conj(\mathbf{q}). \tag{3.34}$$

Considering two quaternions $\mathbf{q_0} = [a, b, c, d]^T$ and $\mathbf{q_1} = [e, f, g, h]^T$ following the nomenclature defined in equation (3.30). The addition of quaternions $\mathbf{q_0}$ and $\mathbf{q_1}$ can be seen as

$$\mathbf{q_0} + \mathbf{q_1} = [a, b, c, d]^T + [e, f, g, h]^T = [a + e, b + f, c + g, d + h]^T \tag{3.35}$$

whilst the multiplication of a scalar β and a quaternion $\mathbf{q_0}$ is

$$\beta\mathbf{q_0} = \beta[a, b, c, d]^T = [\beta * a, \beta * b, \beta * c, \beta * d]^T. \tag{3.36}$$

On the other hand, the product of two quaternions $\mathbf{q_0}$ and $\mathbf{q_1}$, also called Hamilton product, can be done as

$$\mathbf{q_0} \times \mathbf{q_1} = \begin{bmatrix} ae - bf - cg - dh \\ af + be + ch - dg \\ ag - bh + ce + df \\ ah + bg - cf + de \end{bmatrix} \tag{3.37}$$

or defined as a multiplication of matrices, using the following T transform

$$\mathbf{q_0} \times \mathbf{q_1} = T(\mathbf{q_0})\mathbf{q_1} = \begin{bmatrix} a & -b & -c & -d \\ b & a & -d & c \\ c & d & a & -b \\ d & -c & b & a \end{bmatrix} \begin{bmatrix} e \\ f \\ g \\ h \end{bmatrix} \tag{3.38}$$

or

$$\mathbf{q_0} \times \mathbf{q_1} = \bar{T}(\mathbf{q_1})\mathbf{q_0} = \begin{bmatrix} e & -f & -g & -h \\ f & e & h & -g \\ g & -h & e & f \\ h & g & -f & e \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}. \tag{3.39}$$

A rotation of an angle θ around the unit vector $\mathbf{u}$ can be defined as a quaternion [84] [85] following

$$\mathbf{q} = \cos\left(\frac{\theta}{2}\right) + \mathbf{u}\sin\left(\frac{\theta}{2}\right). \tag{3.40}$$

Moreover, given $\mathbf{q} \in S^3$, the rotation of a vector $\mathbf{v} \in \mathbb{R}^3$ by $\mathbf{q}$ is given by [84]

$$\mathbf{w} = \mathbf{q} \times \begin{bmatrix} 0 \\ \mathbf{v} \end{bmatrix} \times \mathbf{q}^*. \tag{3.41}$$

Additionally, the variation of the quaternion over time in regards to the angular velocity in the world-fixed frame is given by

$$\frac{d\mathbf{q}}{dt} = \frac{1}{2}\mathbf{w} \times \mathbf{q}. \tag{3.42}$$

Regarding the angular velocity in the body frame, consider the following equation

$$\frac{d\mathbf{q}}{dt} = \frac{1}{2}\mathbf{q} \times \mathbf{\Omega}. \tag{3.43}$$

The proof for Equations (3.42) and (3.43) is relevant in the context of this work, as such it is presented in Annex I.

Furthermore, it is possible to convert from quaternion to other known three-dimensional representations, such as Rotation Matrices and Euler angles.

Having in consideration Equation (3.41), for a value of v for each of the x,y and z axis, it is possible to get the rotation around each individual axis, $\mathbf{r_x}(\mathbf{q})$, $\mathbf{r_y}(\mathbf{q})$ and $\mathbf{r_z}(\mathbf{q})$, respectively. With that in mind, the rotation matrix can be defined [86] [84] as

$$\mathbf{R}_q(\mathbf{q}) = \begin{bmatrix} \mathbf{r_x}(\mathbf{q}) & \mathbf{r_y}(\mathbf{q}) & \mathbf{r_z}(\mathbf{q}) \end{bmatrix} \tag{3.44}$$

$$\mathbf{R}_q(\mathbf{q}) = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \tag{3.45}$$

in which $\mathbf{R}_q(.) \in SO(3)$.

For the inverse operation, of converting the rotation matrix into a quaternion, there are a few methods that can be used, such as the Shepperd's method, first introduced in [87], Hughes's method [88] or Sarabandi and Thomas's method published in [89]. This last one increases the probability of finding the original quaternion by 4% when compared to the Shepperd's method [89].

Both, rotation matrices and quaternions circumvent the singularities and discontinuities related to the 3-dimensional Euler angles by adding extra dimensions. This, however, introduces a new issue, as different values might represent the exact same rotation. In the quaternion world, $\mathbf{q}$ and $-\mathbf{q}$ would represent the same rotation [90]. Regardless, there are some works on how to avoid the singularity of the Euler angles, as showcased by Singla et al. in [91] and Kang et al. in [92].

4

System Dynamics

This Chapter details the system dynamics of a six DoF vehicle in 3D space, considering both the motion model and the measurement model. A six DoF vehicle is a vehicle that can move through and over the 3 Cartesian axis.

The system dynamics was designed in order to model the behaviour of any vehicle in 3-D space, as it considers it as the movement of an abstract rigid body. Moreover, it also defines the relation of the data collected by a set of sensors to the vehicles pose and reference frame. For a better understanding of the model considered, it should be noted that the state vector of the system

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{x}_{Vk} \\ \mathbf{x}_L \end{bmatrix} \tag{4.1}$$

can be divided into two subgroups: the state vector related to the vehicles position and quaternion orientation

$$\mathbf{x}_{Vk} = \begin{bmatrix} \mathbf{p}_k \\ q_{0,k} \\ q_{1,k} \\ q_{2,k} \\ q_{3,k} \end{bmatrix} \tag{4.2}$$

and the set of landmarks in the earth-fixed reference frame,

$$\mathbf{x}_L = \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \dots \\ \mathbf{p}_L \end{bmatrix}. \tag{4.3}$$

Moreover, every position $\mathbf{p}$ can be defined as a vector with x,y and z coordinates in the form of

$$\mathbf{p} = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}. \tag{4.4}$$

The set of landmarks $\mathbf{x}_L$ considers every l-th landmark that has been observed.

4.1 Motion Model

For the purpose of the motion model it was opted to view it as a simple rigid body. This decision enables the use of this filter to any type of vehicle, as long as the velocities at play on the rigid body are available. The proposed model works by considering the measurements of the vehicle's linear and angular velocities applied to the body. It was also considered that the sensor that returns such values, for example an IMU, is aligned with the body's reference frame and that its values are distorted by noise. As such, it is not necessary to do a transformation from the sensors frame to the robot frame. Take into account $\mathbf{v}$ and $\boldsymbol{\omega}$ as the linear and angular velocities given by

$$\mathbf{v} = \begin{bmatrix} v_x & v_y & v_z \end{bmatrix}^T \tag{4.5}$$

$$\boldsymbol{\omega} = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^T. \tag{4.6}$$

The measured angular and linear velocities, $\mathbf{v_m}(t)$ and $\boldsymbol{\omega_m}(t)$ respectively, can be defined as the sum of the actual value and the noise associated with the environment, sensor and system, as follows

$$\mathbf{v}_m(t) = \mathbf{v}(t) + \xi_v(t) \tag{4.7}$$

$$\boldsymbol{\omega_m}(t) = \boldsymbol{\omega}(t) + \xi_\omega(t). \tag{4.8}$$

Following the demonstration to get to Equation (3.43), the variation of a quaternion, $\mathbf{q} \in S^3$, imposed by the angular velocities on the vehicles' frame can be represented as

$$\dot{\mathbf{q}}(t) = \frac{1}{2}\mathbf{q}(t) \times \begin{bmatrix} 0 \\ \boldsymbol{\omega_m}(t) \end{bmatrix} \tag{4.9}$$

which can be translated into the multiplication between a matrix and a vector if it is modelled using the transformation, $\bar{\mathbf{T}}(.)$, showed in Equation (3.39). For simplicity purposes, the transformation $\bar{\mathbf{T}}$ of the angular velocity pure quaternion, will be defined as $\mathbf{S'}[.]$

$$\mathbf{S'}[\boldsymbol{\omega}_m(t)] = \bar{\mathbf{T}}\left(\begin{bmatrix} 0 \\ \frac{1}{2}\boldsymbol{\omega}_m(t) \end{bmatrix}\right). \tag{4.10}$$

The nomenclature used to define $\mathbf{S'}$ is a reference to its characteristic of being a skew-symmetric matrix, which is a square matrix that has its transpose equal to its negative

$$\mathbf{S'}^T = -\mathbf{S'}. \tag{4.11}$$

Taking into consideration equations (4.9) and (4.10), the derivative of $\mathbf{q}$ can be defined as

$$\dot{\mathbf{q}}(t) = \mathbf{S'}\big[\boldsymbol{\omega}_m(t)\big]\mathbf{q}(t). \tag{4.12}$$

Regarding the position of the robot related to the world frame, the variation of $\mathbf{p}(t)$ can be defined as

$$\dot{\mathbf{p}}(t) = {}^W_B\mathbf{R}_q\big(\mathbf{q}(t)\big)\mathbf{v}_m(t) \tag{4.13}$$

in which ${}^{W}_{B}\mathbf{R}_{q} \in SO(3)$, is the transformation of the quaternion $\mathbf{q}$ to a rotation matrix, similarly to what was displayed at Equation (3.45). In short, and having in consideration both Equation (4.12) and (4.13), the full motion model of a rigid body on 3D space can be given by

$$\begin{cases} \dot{\mathbf{p}}(t) = {}^{W}_{B}\mathbf{R}_{q}(\mathbf{q}(t))\mathbf{v}_{m}(t) \\ \dot{\mathbf{q}}(t) = \mathbf{S}'\left[\boldsymbol{\omega}_{\boldsymbol{m}}(t)\right]\mathbf{q}(t) \end{cases} \tag{4.14}$$

Considering that the former system of equations is inherently nonlinear, the data fusion of the EKF and MHE approaches will be performed in the discrete-time. The discretization of the first equation, $\mathbf{p}(t)$, uses the forward Euler method [93], as it was considered that the linear velocities on the body are constant during the considered step, Δ_t. In contrast, this consideration was not made in relation to the discretization of $\mathbf{q}(t)$, as it would be a gross approximation. It was therefore obtained considering the integral approximation method, which approximates the time-continuous differential system dynamics by a discrete infinite series, $e^{\mathbf{S}'(\boldsymbol{\omega}_{\boldsymbol{m}}(k))\Delta_t}$. The discretization results on the following system of equations

$$\begin{cases} \mathbf{p}(k+1) = \mathbf{p}(k) + \Delta_t . {}^{W}_{B}\mathbf{R}_{q}\big(\mathbf{q}(k)\big).\mathbf{v}_{m}(k) \\ \mathbf{q}(k+1) = e^{\mathbf{S}'\left[\boldsymbol{\omega}_{\boldsymbol{m}}(k)\right]_{\Delta_t}}.\mathbf{q}(k) \end{cases} \tag{4.15}$$

where the discrete time steps might be defined as $k = \Delta_t.k + t_0$ in which t_0 is the initial time and $k \in \mathbb{N}_0$.

The decision of defining the vehicles orientation by a quaternion representation over other options like the rotation matrices and Euler angles was derived by two factors. The choice of the quaternion over rotation matrices was made taken into account the number of variables in each option. While the quaternion is represented by a 4-by-1 vector, the 3-D rotation matrices are represented by a 3-by-3 matrix which has nine elements, instead of the four elements of the quaternion. This is relevant due to the number of variables in the vector state that the MHE would have to consider and estimate, which would demand a bigger work load on the algorithm that minimizes the cost function. As this method is known for its computational cost the most wise option is to make its life easier and choose the quaternion representation. The Euler angles are able to define the attitude of an object with only three variables, as it only needs to define the angles related to the rotations on the x, y and z axis. However, their use comes with the presence of a singularity whenever the angle of the second rotation is equal to 90 degrees (or 270 degrees), as that would result in an infinite number of solutions to the Euler sequence represented in the "x-convention" at Equation (3.29). On the "roll-pitch-yaw notation" of the Euler angles, this singularity appears whenever the pitch rotation is equal to the values mentioned before. Even though this work defines the vehicle in a general way, its six DoF makes it possible for the vehicle to reach such pitch values. Furthermore, this work is within the scope of research on spacecraft and of the CAPTURE project, which relies on fixed-wing aircrafts and UAVs, with that in mind, and considering that UAVs exhibit a complicated and dynamic motion

that can realistically achieve a pitch of 90 degrees, is was opted to not use the Euler angles representation.

4.2 Observation Model

The observation model consists of the Cartesian coordinates of the l-th landmark to the vehicle's visual sensor. As a result the observation model is defined by converting the l-th landmark point from the world frame to the vehicles frame. This conversion can be obtained following

$$ {}^{B}\mathbf{p}_{L,k} = \begin{bmatrix} {}^{B}x_{m,k} \\ {}^{B}y_{m,k} \\ {}^{B}z_{m,k} \end{bmatrix} = {}^{B}_{W}\mathbf{R}_{q}\Big(\mathbf{q}(k)\Big)\Big({}^{W}\mathbf{p}_{L,k} - \mathbf{p}_{V,k}\Big) \tag{4.16} $$

in which ${}^{B}_{W}\mathbf{R}_{q}\Big(\mathbf{q}(k)\Big)$ is given by ${}^{W}_{B}\mathbf{R}_{q}\Big(\mathbf{q}(k)\Big)^{T}$. In spite of that, some sensors might work with polar coordinates. If that is the case, each landmark position on the body-frame as to be transformed following

$$ \mathbf{y_{m,k}} = \begin{bmatrix} r_{m,k} \\ \theta_{m,k} \\ \phi_{m,k} \end{bmatrix} = \begin{bmatrix} \sqrt{{}^{B}x_{m,k}{}^{2} + {}^{B}y_{m,k}{}^{2} + {}^{B}z_{m,k}{}^{2}} \\ \text{atan2}\big({}^{B}y_{m,k}, {}^{B}x_{m,k}\big) \\ \text{atan2}\big({}^{B}z_{m,k}, {}^{B}x_{m,k}\big) \end{bmatrix} + \xi_{y,k} \tag{4.17} $$

where $r_{m,k}$ is the sensor reading for the distance to the landmark, $\theta_{m,k}$ the angle of the landmark to the robots x axis on the XY plain, and $\phi_{m,k}$ the angle of the landmark to the robots x axis on the XZ plain.

In addition to that, it should be mentioned that the landmarks were considered as stationary, as a result they don't have a dependency on k like the position of the vehicle. In other words, every position vector, $\mathbf{p}$, present in the landmarks set of the state vector, $\mathbf{x}_L$, follows the following assumption

$$ \dot{\mathbf{p}} = 0 \tag{4.18} $$

Every time the vector $\mathbf{y}_k$ is updated one or more landmark position measurements are available. Whenever a new landmark is observed, the state vector is augmented, in order to accommodate the brand new landmark position. The state vector regarding the landmarks, $\mathbf{x}_L$ can also be divided into two subgroups: the observed landmarks $\mathbf{x}_{L_O}$ and not observed landmarks $\mathbf{x}_{L_U}$ at each given instant k.

$$ \mathbf{x}_L = \begin{bmatrix} \mathbf{x}_{L_O} \\ \mathbf{x}_{L_U} \end{bmatrix}. \tag{4.19} $$

To solve this system of equations, different approaches were taken into account. The proposed solution based on MHE, which integrates the EKF on the estimation step is described in Chapter 6. As a basis of comparison, an approach using EKF and another

using the sensor-based LKF were developed, which are presented in Chapter 5 and 7, respectively. The choice of opting for the EKF as a benchmark algorithm, was made having in consideration its widely use on nonlinear SLAM applications, whilst the sensor-based LKF showed interesting characteristics when dealing with nonlinear dynamics. The considered scenarios will be discussed in next the Chapters regarding their advantages and limitations, computing time and robustness.

5

Extended Kalman Filter for 3-D SLAM

This work is based on the evaluation of the MHE algorithm as a method of solving the SLAM problem for a vehicle with non-linear dynamics. As such, the EKF method will be used as a comparison base for the evaluation of the MHE performance. As such, this Chapter details the use and design of an EKF algorithm considering the system defined in Chapter 4.

Even though the EKF approximates the nonlinear dynamics by a linear system using the first-order Taylor series expansion, it is widely used on similar scenarios, having proven its reliability and being known for its fast computing time.

As presented on Chapter 3 the EKF method is comprised of two steps, the prediction step and the update step, as described hereafter.

5.1 Prediction Step

The prediction step estimates the next values of the state vector based on the systems motion model and the error covariance matrix associated with such movement, according to the following equations

$$\hat{\mathbf{x}}_{k+1|k} = \mathbf{f}(\hat{\mathbf{x}}_{k|k}, \mathbf{u}_{k+1}) \tag{5.1}$$

$$\mathbf{\Sigma}_{k+1|k} = \mathbf{F}_{k+1}\mathbf{\Sigma}_{k|k}\mathbf{F}_{k+1}^{T} + \mathbf{Q}_k \tag{5.2}$$

where $\mathbf{\Sigma}_{k|k}$ represents the estimated covariance matrix and $\hat{\mathbf{x}}_{k|k}$ the estimated state vector at time $k+1$, considering the information up to time k. Whereas $\mathbf{\Sigma}_{k+1|k}$ and $\hat{\mathbf{x}}_{k+1|k}$ are estimations of the covariance matrix and state vector at time $k+1$ considering the data collected until time k. The process noise covariance $\mathbf{Q}_k \in \mathbb{R}^{7\times7}$ represents the vehicles dynamics process noise covariance and is defined as

$$\mathbf{Q}_k = \mathbf{\Gamma}.\mathbf{Q}_{\Omega,k}.\mathbf{\Gamma}^{T} \tag{5.3}$$

with

$$\mathbf{Q}_{\Omega,k} = blkdiag(\sigma^2_{\xi_u}) \tag{5.4}$$

as σ_{ξ_u} denotes the standard deviation of the process noise regarding the linear and angular velocities of the body given by the IMU sensor attached to it.

$$\xi_u = \begin{bmatrix} \xi_v & \xi_\Omega \end{bmatrix}. \tag{5.5}$$

The $\mathbf{\Gamma}$ value correspondd to the Jacobian matrix of the system function $\mathbf{f}(\mathbf{x}_{k|k}, \mathbf{u}_k)$ with respect to the measured data from the IMU at instant $k+1$

$$\mathbf{\Gamma} = \frac{\delta \mathbf{f}}{\delta \mathbf{u}} \mid_{\mathbf{u}(k+1)} . \tag{5.6}$$

Lastly, $\mathbf{F}_{k+1} \in \mathbb{R}^{(7+3n_L)\times(7+3n_L)}$ is the transition matrix and is defined as

$$\mathbf{F}_{k+1} = diag(\nabla \mathbf{f}_x, \mathbf{0}_{3n_L \times 3n_L}) \tag{5.7}$$

where $\nabla \mathbf{f}_\mathbf{x}$ is the Jacobian matrix of the system dynamics, $\mathbf{f}(\mathbf{x}_{k|k}, \mathbf{u}_k)$, regarding $\mathbf{x}_{k|k}$.

5.2 Update Step

Whenever a new measurement is received, the vector $\mathbf{y}_k$ is updated and the update step is triggered on that k instant. As mentioned before, this new measurement can be either relative to an already known landmark, therefore already consider on the state vector, or a new landmark, which will require the augmentation of the state vector, $\mathbf{x}_L$. The update step is given by the following equations

$$\hat{\mathbf{x}}_{k+1|k+1} = \hat{\mathbf{x}}_{k+1|k} + \mathbf{K}_{k+1}\tilde{\mathbf{y}}_{k+1} \tag{5.8}$$

$$\mathbf{\Sigma}_{k+1|k+1} = \mathbf{\Sigma}_{k+1|k} - \mathbf{K}_{k+1}\mathbf{S}_{k+1}\mathbf{K}_{k+1}^T \tag{5.9}$$

where $\hat{\mathbf{x}}_{k+1|k+1}$ represents the updated state vector and $\mathbf{\Sigma}_{k+1|k+1}$ the updated estimation of the covariance matrix at time $k+1$, considering the information gathered up to that instant. While the measurement residual $\tilde{\mathbf{y}}_{k+1}$, the respective residual covariance $\mathbf{S}_{k+1}$ and the Kalman gain $\mathbf{K}_{k+1}$ are defined by

$$\tilde{\mathbf{y}}_{k+1} = \mathbf{y}_k - \mathbf{h}(\hat{\mathbf{x}}_{k+1|k}) \tag{5.10}$$

$$\mathbf{S}_{k+1} = \mathbf{H}_{k+1}\mathbf{\Sigma}_{k+1|k}\mathbf{H}_{k+1}^T + \mathbf{R}_k \tag{5.11}$$

$$\mathbf{K}_{k+1} = \mathbf{\Sigma}_{k+1|k}\mathbf{H}_{k+1}^T\mathbf{S}_k^{-1} \tag{5.12}$$

in which, $\mathbf{R}_k \in \mathbb{R}^{3\times3}$ is the measurement noise covariance defined as

$$\mathbf{R}_k = diag(\sigma_{\xi_y}^2) \tag{5.13}$$

in which σ_{ξ_y} is the standard deviation of the measurement noise. Moreover, $\mathbf{H}_{k+1} \in \mathbb{R}^{(7+3n_L)\times3}$ is the measurement matrix which is described as

$$\mathbf{H}_{k+1} = \nabla \mathbf{h}_x \tag{5.14}$$

if every landmark defined on the state vector was observed. If, for example only the first landmark of state vector was visualized, the measurement matrix takes the following aspect

$$\mathbf{H}_{k+1} = \begin{bmatrix} \nabla \mathbf{h}_{x_V} & \nabla \mathbf{h}_{x_1} & \mathbf{0}_{3(n_L-1)\times 3} \end{bmatrix} \tag{5.15}$$

where $\nabla \mathbf{h}_{\mathbf{x}}$ is the Jacobian matrix of the observation model regarding $\mathbf{x}_k$, while $\nabla \mathbf{h}_{\mathbf{x_v}}$ and $\nabla \mathbf{h}_{\mathbf{x_L}}$ are the Jacobian matrix of the observation model regarding $\mathbf{x}_{V_k}$ and $\mathbf{x}_{L_k}$, respectively.

5.3 Simulation results

In this Section the proposed solution is applied to a drone on a simulated environment, which helps on evaluating its' performance and robustness as a solution for the SLAM problem. The simulations were done using MATLAB 2021a program.

5.3.1 Simulation environment

The simulated environment is a 3-D square shaped corridor, with 48 landmarks along its sides and edges. The simulation takes 215 seconds to be complete and the drone has an average speed of 0.1 m/s. This velocity is quite low, however it only considers one iteration per second. A faster vehicle would require a higher frequency of estimation. The simulation system disturbances are zero-mean white Gaussian noise and have the following standard deviations

$$\boldsymbol{\sigma}_{\xi_u} = \begin{bmatrix} \boldsymbol{\sigma}_{\xi_v} & \boldsymbol{\sigma}_{\xi_\omega} \end{bmatrix} \tag{5.16}$$

$$\boldsymbol{\sigma}_{\xi_v} = \begin{bmatrix} 0.015 & 0.015 & 0.015 \end{bmatrix} m/s \tag{5.17}$$

$$\boldsymbol{\sigma}_{\xi_\omega} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} deg/s \tag{5.18}$$

$$\boldsymbol{\sigma}_{\xi_y} = \begin{bmatrix} 5.0 & 5.0 & 5.0 \end{bmatrix} cm \tag{5.19}$$

This will be considered Scenario 1, in which the rigid-body does a loop on the horizontal plane. This environment simulates the scenario where a vehicle moves over a corridor on the XY plane. The linear and angular velocities regarding this movement are presented in Figures 5.1 and 5.2, as well as the inherent noise regarding this measurements which have the standard deviation presented in the previous equations. From Figure 5.1 is possible to understand that the body only has a linear velocity on its' x axis, as the others are exclusively noise. On Figure 5.2 it is possible to visualize the angular velocities that make the horizontal loop, which appear on the z axis.

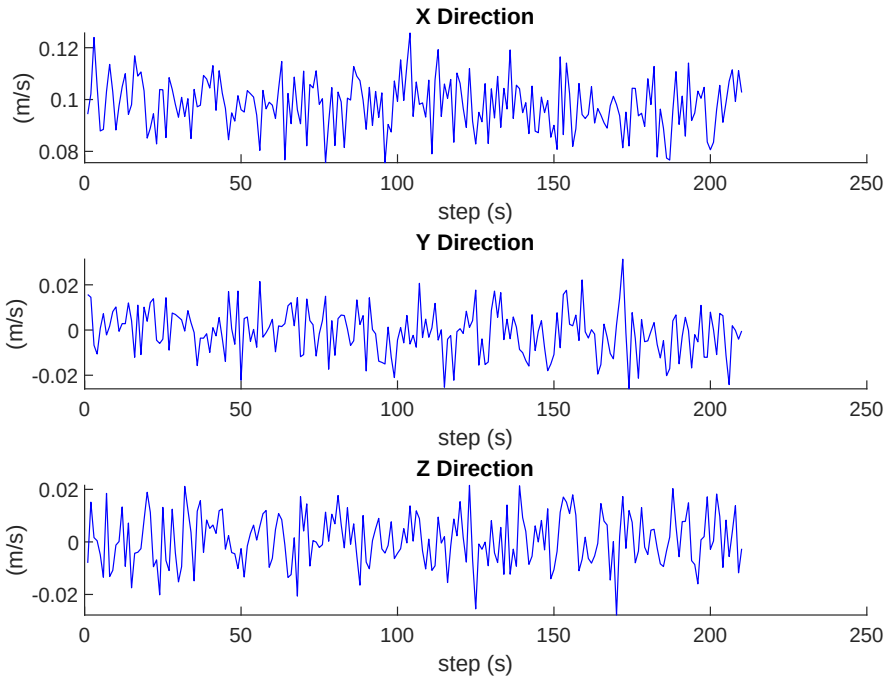


Figure 5.1: Scenario 1 - input values of linear velocities

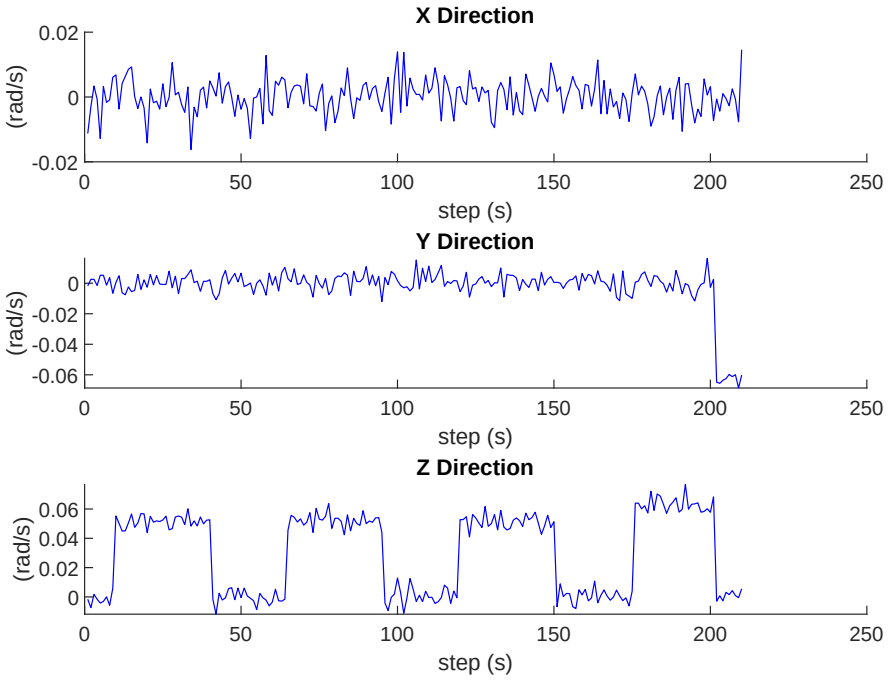


Figure 5.2: Scenario 1 - input values of angular velocities

5.3.2 Results

The true and estimated trajectories of the simulation in the Scenario 1 are presented in Figure 5.3, as well as in Figure 5.4 however projected on the XY plane, being the true trajectory represented in purple and the estimation in blue. Moreover it is also possible to visualize the resulting map and the final vehicle position after the vehicle trajectory in the square shaped corridor. Every estimated position is associated with two ellipses as a characterization of their 3-sigma bounded ellipsoid representation of the covariance matrices. The green ellipses are related to the covariances of the landmarks, whilst the pink one is related to the vehicle pose. Analyzing Figures 5.3 and 5.4 it is possible to see that the estimated trajectory, as well as the landmarks positions, are near their true values. This can be corroborated through Figures 5.5 and 5.6. As the first shows the true position (in red) and the estimate of the vehicle (in brown) on the x,y and z axis, and also the respective 3-sigma bound regarding the covariance matrix of the position error over the simulation period. Similarly, Figure 5.7 also has in consideration the 3-sigma bounds but instead of representing the actual estimated values, it presents the estimation error for a better visualization. From these Figures it is possible to see that the estimation is always within the 3-sigma bounds and recognise where the closed loop happens, as in step 192 the estimation is corrected, thus the error goes towards zero and the 3-sigma bounds get narrower and closer to the estimation., which translates to a smaller uncertainty regarding the position of the vehicle.

Additionally, Figure 5.7 shows the estimated orientation error, from which it is possible to identify the relation between the position error and the error regarding the orientation, specially when the loop closure happens. This error was calculated following the next Equation

$$e_R = \frac{1}{2} S^{-1}\Big[\mathbf{R}_q(\mathbf{q})^T \mathbf{R}_q(\hat{\mathbf{q}}) - \mathbf{R}_q(\hat{\mathbf{q}})^T \mathbf{R}_q(\mathbf{q})\Big] \tag{5.20}$$

where $\mathbf{S}^{-1}[.]$ is the inverse operation of $\mathbf{S}[.]$ as

$$\mathbf{S}^{-1}\Big[\mathbf{S}[\mathbf{w}]\Big] \longrightarrow \mathbf{w}. \tag{5.21}$$

In top of that, it is also possible to recognise the relation between the error of the estimated vehicle position and the landmarks position, as one influences the other. This connection can be visualized in Figure 5.8, where it is depicted the sum of the estimation error of each landmark on the x,y and z axis of the 3 dimensional Cartesian plane. It is also visible that the observation of an already seen landmark does a backwards propagation for the landmarks estimation, which happens on the case of a close loop event and results in a closer to reality estimation of the map.

Overall, the results obtained detail a good approximation of the EKF filter to the true values of landmarks and vehicle position, confirming why the EKF estimation filter is so widely used.

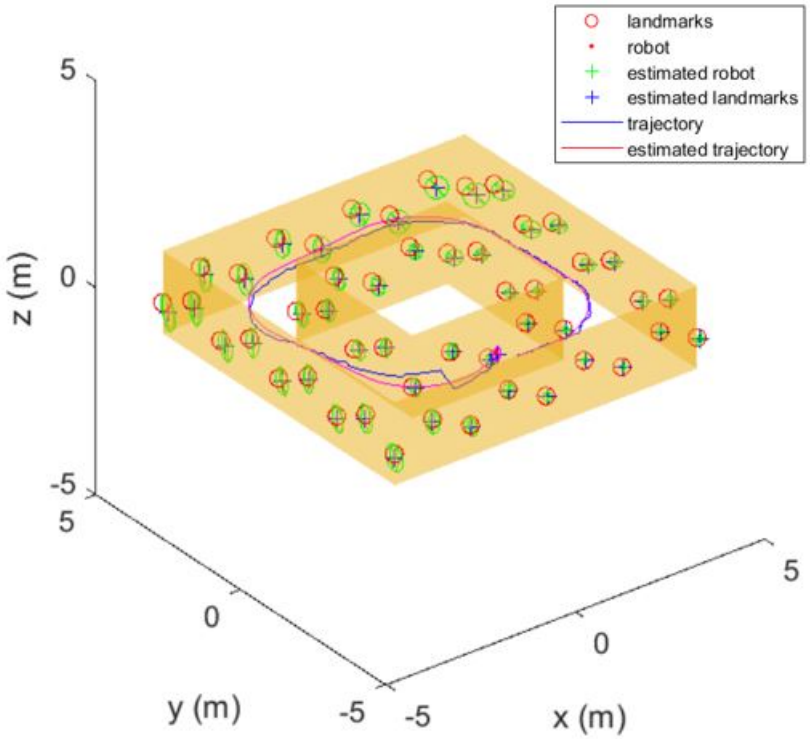


Figure 5.3: Scenario 1 - Simulation result with the EKF method

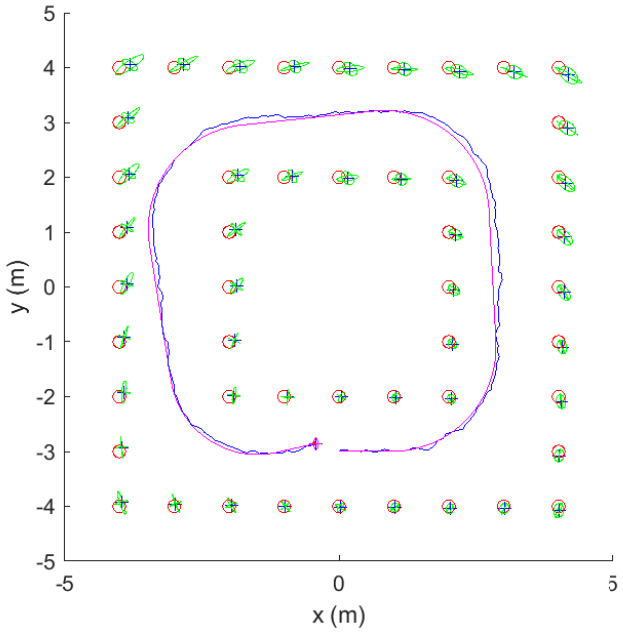


Figure 5.4: Scenario 1 - Simulation result with the EKF method, XY plane

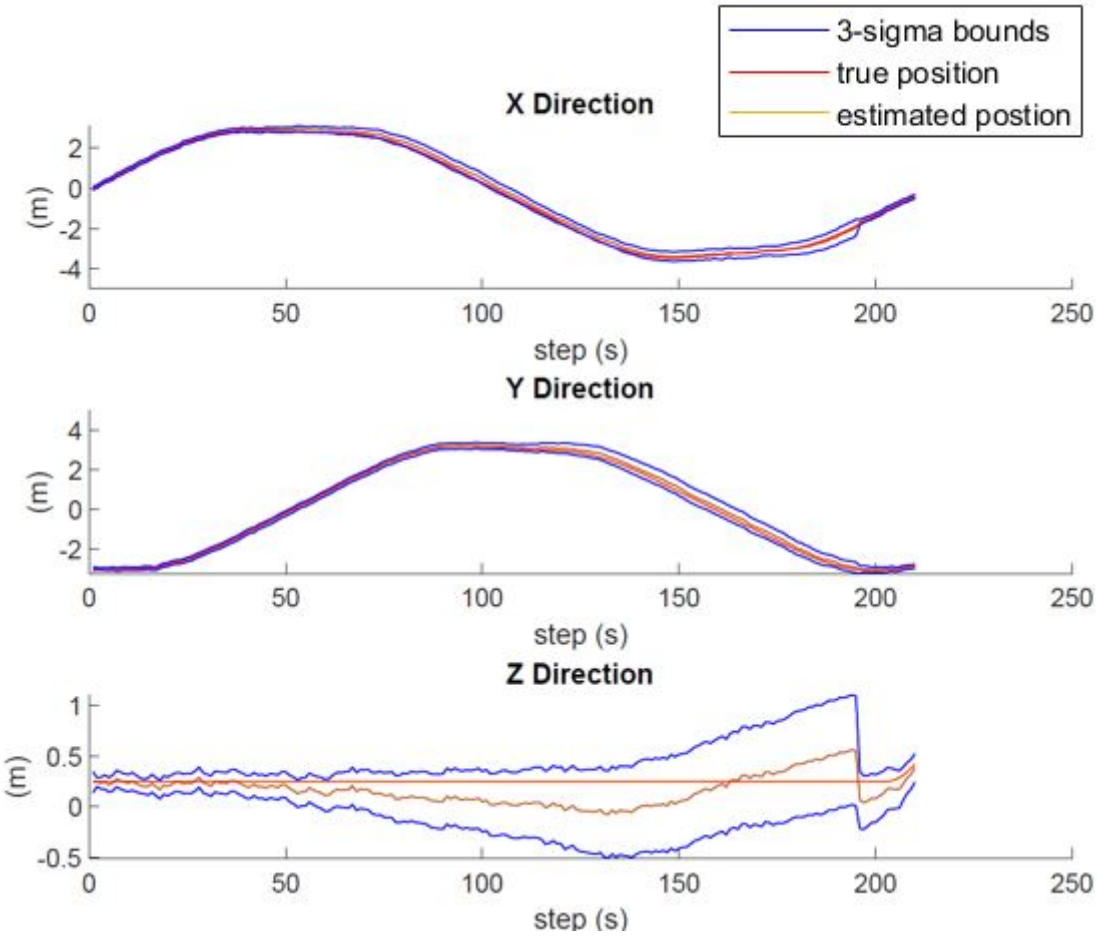


Figure 5.5: Scenario 1 - Vehicle true and estimated position with 3-sigma bounds using the EKF method

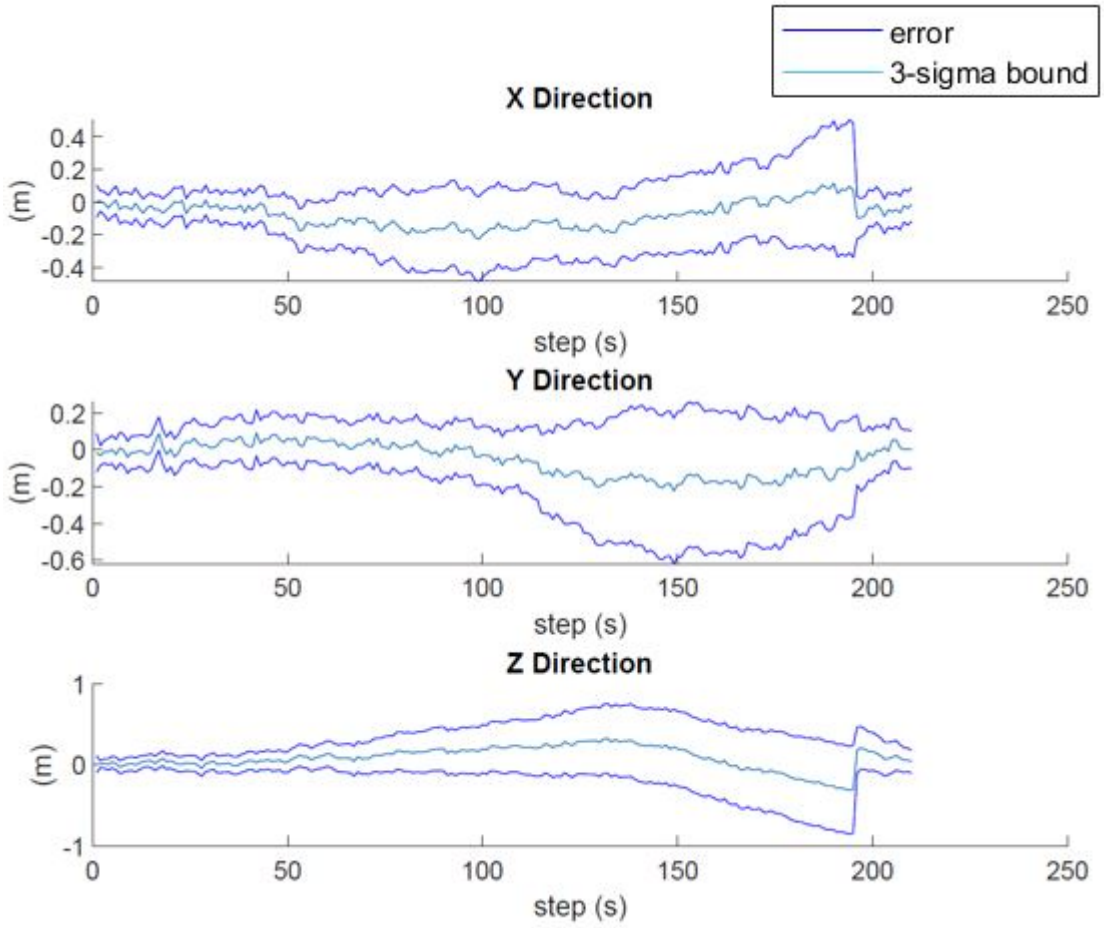


Figure 5.6: Scenario 1 - Vehicle position error with 3-sigma bounds using the EKF method

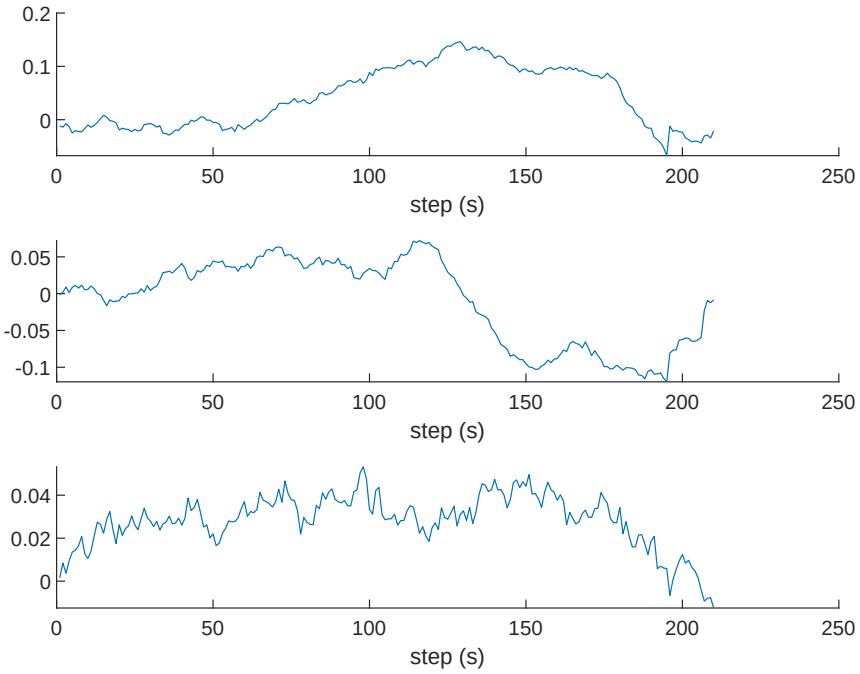


Figure 5.7: Scenario 1 - Vehicle orientation error with the EKF method

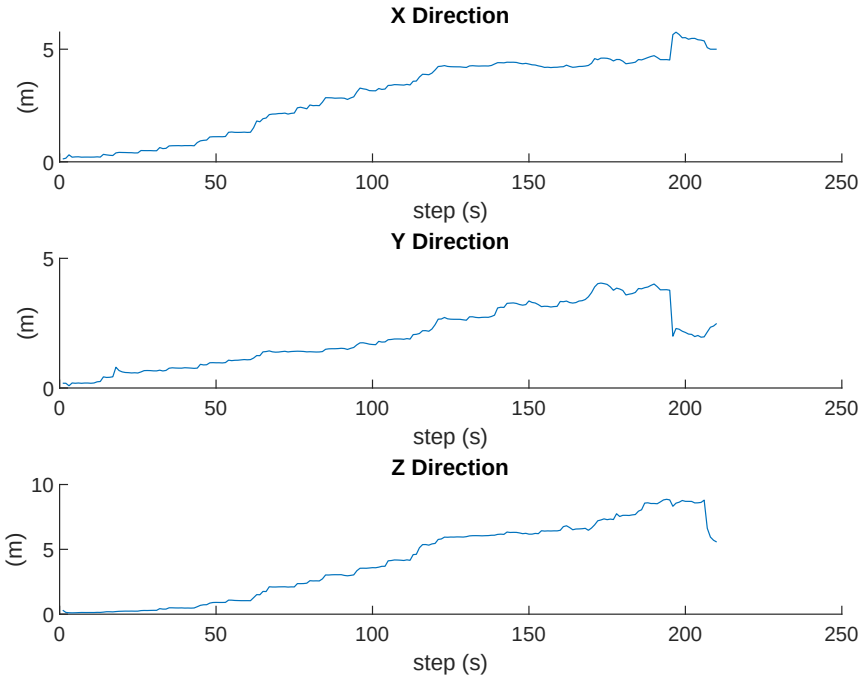


Figure 5.8: Scenario 1 - Sum of the landmarks position error with the EKF method

6

Moving Horizon Estimation for 3-D SLAM

The MHE method, as presented on Chapter 2, deals better with nonlinearities than other algorithms, such as the EKF which is a widely used estimation method for nonlinear systems. This is due to the fact that the MHE can inherently deal with nonlinear dynamics, and solutions like the EKF have to process linearized version of it. Moreover, there are no assumptions regarding the type of noise on this method.

On this chapter, it is presented in full detail the design and use of a probabilistic MHE algorithm as a solution to the SLAM problem. As it was taken a probabilistic approach to the MHE, there is the need to know the covariances at play, which were calculated using the EKF algorithm. With that in mind, this Chapter was divided into three Sections: Section 6.1 design of the MHE algorithm, Section 6.2 integration of the EKF in the MHE filter and Section 6.3 the respective simulation.

6.1 MHE

Differently than the EKF, the MHE estimates the previous robot positions, $\mathbf{p}_k$, in each iteration, in a similar manner to the full information estimator, but with a restricted horizon H of past data. As such, the estimated vector, $\hat{\mathbf{x}}$, of the MHE is given as

$$\hat{\mathbf{x}} = \begin{bmatrix} \mathbf{x}_{V,k} \\ \mathbf{x}_{V,k-1} \\ \dots \\ \mathbf{x}_{V,k-H} \\ \mathbf{x}_L^{'} \end{bmatrix} \tag{6.1}$$

which is comprised of vehicle positions, $\mathbf{x}_V$, over time and the landmarks position, $\mathbf{x}_L^{'}$. Moreover, the $\mathbf{x}_L^{'}$ is not the same as the one presented in the system dynamics in Chapter 4, as the one here presented is the collection of observed landmarks O in the horizon H, as such a subset of the broader $\mathbf{x}_L$ vector. This subset is used because the complete one grows as fast as new landmarks are visualized. This is already a challenge on every approach to

solving the SLAM problem, however in the use case of the MHE it's even more relevant as it will have a bigger impact on the velocity of the algorithm that finds the best solution to the cost function. As such, $\mathbf{x}_L^{'}$ is created dynamically having in consideration the vector $\mathbf{x}_L$ and the horizon. It changes in each iteration having in account the landmarks visualized in the past H instants, which corresponds to the size of the MHE window.

With that in mind, the MHE cost function was designed following the equation (3.19), defined as

$$\underset{\mathbf{x}_k, \boldsymbol{\omega}_k, \mathbf{v}_k}{min} \phi = \hat{l}_0(\mathbf{x}_0) + \sum_{k=0}^{N} \hat{l}_k(\boldsymbol{\xi}_{\omega_k}, \boldsymbol{\xi}_{v_k}) \tag{6.2}$$

which tries to minimize the differences between the true and estimated measurement noise, as well as the true and estimated system noise, whilst considering the state vector of the previous iteration. Moreover, as it was considered a probabilistic approach to the MHE, the weights were defined as the covariances of the system dynamics. Which resulted on the following cost function

$$J = \sum_{k=k_0-H}^{k_0} \left\| \hat{\mathbf{p}}_{k+1} - f(\hat{\mathbf{p}}_k, \mathbf{u}_k) \right\|^2_{\mathbf{Q}_k^{-1}} + \sum_{k=k_0-H}^{k_0} \sum_{j \in O_k} \left\| (h(\hat{\mathbf{p}}_k, \hat{\mathbf{p}^{'}}_{L_j}) - \mathbf{y}_{j,k}) \right\|^2_{\mathbf{R}^{-1}} + \left\| \hat{\mathbf{x}} - \hat{\mathbf{x}}^{-} \right\|^2_{\boldsymbol{\Sigma}_k^{-1}}, \tag{6.3}$$

in which k_0 is the current SLAM step, O_k is the set of observed landmarks at time k, $\hat{\mathbf{x}}$ is the current state vector estimation and $\hat{\mathbf{x}}^{-}$ the previous state vector estimation. The first term in equation (6.3) has in consideration the error of the vehicles motion dynamics, while the second term evaluates the error related to the measurements taken of each observed landmark. The last term is the arrival cost, which has in account the previous estimation. Moreover, a constrain on the initial iteration of the MHE was set up, in order to guarantee that the optimization algorithm would converge to the initial position, which is known. Another relevant constraint would be the restriction of the observed landmarks position at a given instant to be within the sensors range.

Additionally, the weights $\mathbf{R}, \mathbf{Q}_k, \boldsymbol{\Sigma}_k$ are the measurement noise, process noise and system EKF covariances, respectively. These matrices will be calculated before each iteration of the MHE, as an EKF algorithm will run in series with it. This interaction is depicted in Figure 6.1, in which is noted that both filters receive sensor and measurement data and propagate their own estimation, whilst the MHE waits for the covariance matrices of the EKF. The choice of designing a probabilistic MHE, thus using the covariances matrices as weights, was made having in account the robustness of this approach in comparison to a user defined or hand tailored algorithm. As this method has in consideration the real probabilities of each movement and visualization, it guarantees a better overall estimation on the long run. The choice of the EKF for calculating the covariance matrices was made having into account the stability and, more relevantly, the quickness of the algorithm, as it had to be considerably faster than the MHE to not create a bigger delay on the estimation filter.

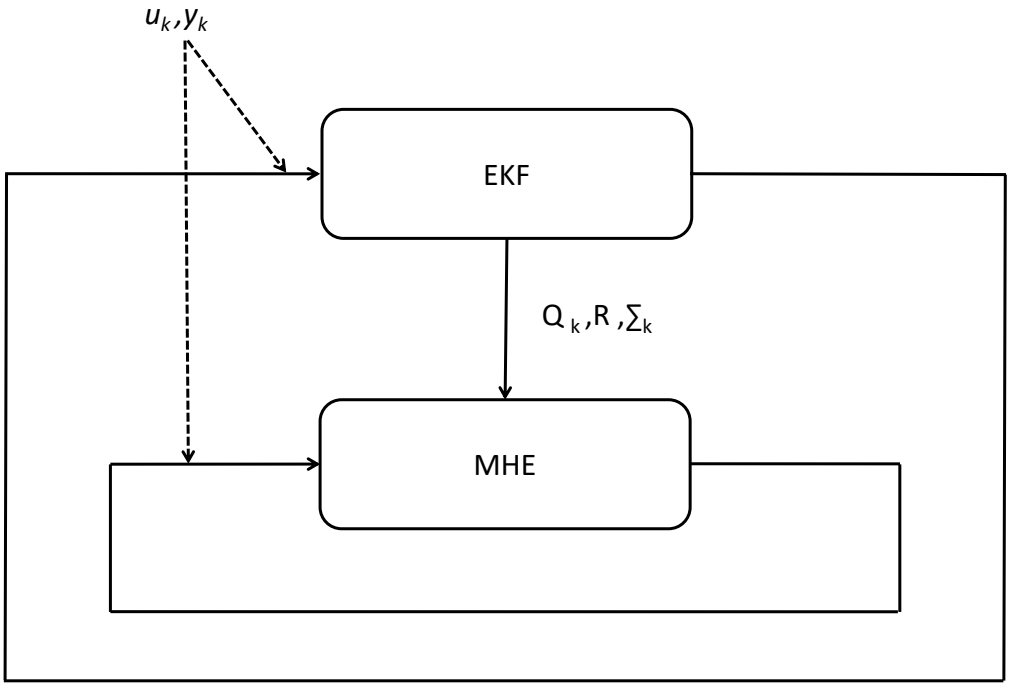


Figure 6.1: MHE+EKF Algorithm

6.2 EKF-based weights

As mentioned before, on this iteration of the filter the weights on the MHE cost function are covariance matrices associated with the system and measurement dynamics. To achieve this it was designed an EKF similar to the one presented at Equations (5.1) to (5.15) from the previous Chapter.

While the inverse of the process noise covariance matrix $\mathbf{Q}_k \in \mathbb{R}^{7\times7}$ is the weight used to influence the estimated positions to follow the motion model, the inverse of the measurement noise $\mathbf{R}_k \in \mathbb{R}^{3\times3}$ has an impact regarding the observation model. Lastly, the system covariance matrix $\mathbf{\Sigma}$ considers the importance of the each of the previous state estimation values. Additionally, the system covariance matrix is considered to be $\mathbf{\Sigma}_{k+1|k+1}$ if the update step of the EKF was executed at instant $k+1$. Whenever that does not happen (there are no measurements at $k+1$) it is consider the covariance matrix obtained at the prediction step of the EKF, $\mathbf{\Sigma}_{k+1|k}$.

Note that at every iteration of the MHE, it has to be considered the values of the covariances on the time window H.

6.3 Simulation results

On this Section it is presented the simulation environment considered, Scenario 1, and the results obtained on such environment using the MHE filter. A discussion regarding the performance of the estimation will also be done.

6.3.1 Simulation environment

The simulated environment is identical to the one described on Chapter 5, a 3-D square shaped corridor, with forty-eight landmarks along its sides and edges. A simulation of 215 seconds has to be complete by a vehicle with an average linear speed of 0.1 m/s which does a horizontal loop. The simulated system disturbances are zero-mean white Gaussian noise and have the follow standard deviations

$$\sigma_{\xi_u} = \begin{bmatrix} \sigma_{\xi_v} & \sigma_{\xi_w} \end{bmatrix} \tag{6.4}$$

$$\sigma_{\xi_v} = \begin{bmatrix} 0.015 & 0.015 & 0.015 \end{bmatrix} m/s \tag{6.5}$$

$$\sigma_{\xi_w} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} deg/s \tag{6.6}$$

$$\sigma_{\xi_y} = \begin{bmatrix} 5.0 & 5.0 & 5.0 \end{bmatrix} cm \tag{6.7}$$

and the input velocities are precisely the same as the ones considered in Scenario 1 and presented in Figures 5.1 and 5.2.

6.3.2 Results

For the initial test of the MHE filter it was considered a window of 3 past steps, $H = 3$. The overall result of Scenario 1 is depicted in figures 6.2 and 6.3, in which it is possible to visualize the true positions of the vehicle and of each landmark, and their MHE estimated value. Moreover, it also depicts the actual trajectory and its estimate, as they are represented in red and blue, respectively. From these Figures it can be concluded that the MHE estimation followed the dynamics of the vehicle considerably well, which can also be corroborated by Figure 6.4 which shows the ABS error of the position of the vehicle during the simulation period, $\tilde{\mathbf{p}}_k = |\hat{\mathbf{p}}_k - \mathbf{p}_k|$. Moreover, the orientation error, which was calculated using Equation (5.20), during the simulation is presented at Figure 6.5. It can be observed that the velocity along the x axis and the rotation on the z axis had a bigger impact on the orientation error regarding the x and y axis. Additionally, the sum of the error of each estimated landmark position in the x,y and z axis is depicted in Figure 6.6. In which is visible that the error is continuously growing, which can be explained by the weak capability of the MHE to do loop closures. As a result of these figures it is possible to conclude the good performance of the algorithm, and that, even though subtle, it was able to identify that the loop closure occurs in the step 195, at which the error related to the estimated position of the vehicle decreases in a tenuous manner. Moreover, since the MHE is known for not being the fastest algorithm on solving the estimation problem, Figure 6.7 displays the time that the algorithm took at each step. This value is considerably high when considering its use as a SLAM approach, however there are more optimized ways of solving it, as it will be discussed next.

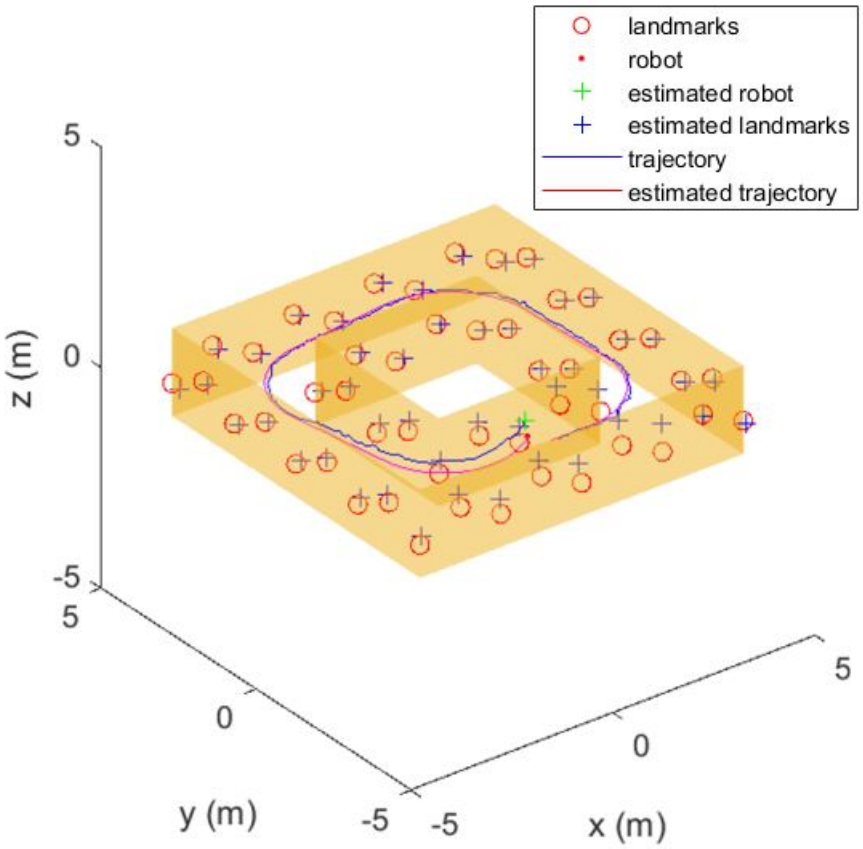


Figure 6.2: Scenario 1 - Simulation result with the MHE method

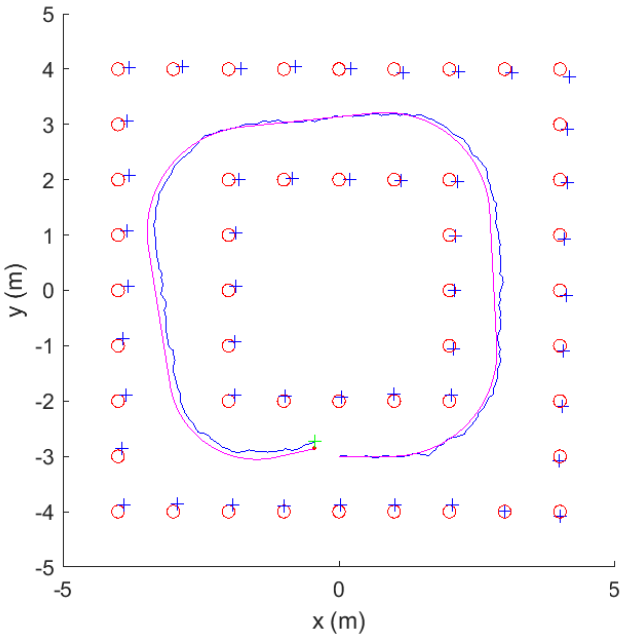


Figure 6.3: Scenario 1 - Simulation result with the MHE method, XY plane

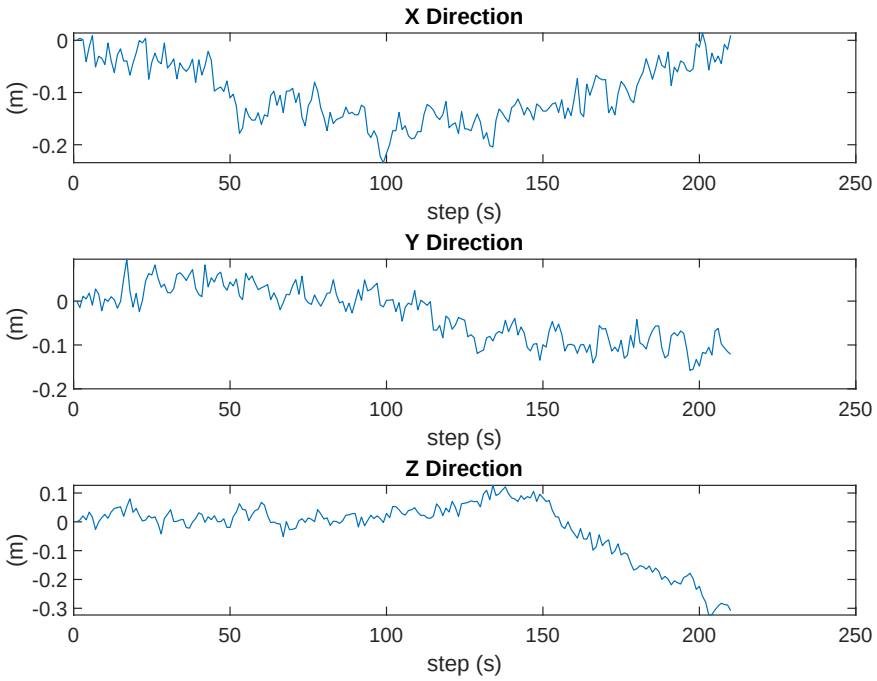


Figure 6.4: Scenario 1 - Vehicle's position error with the MHE method

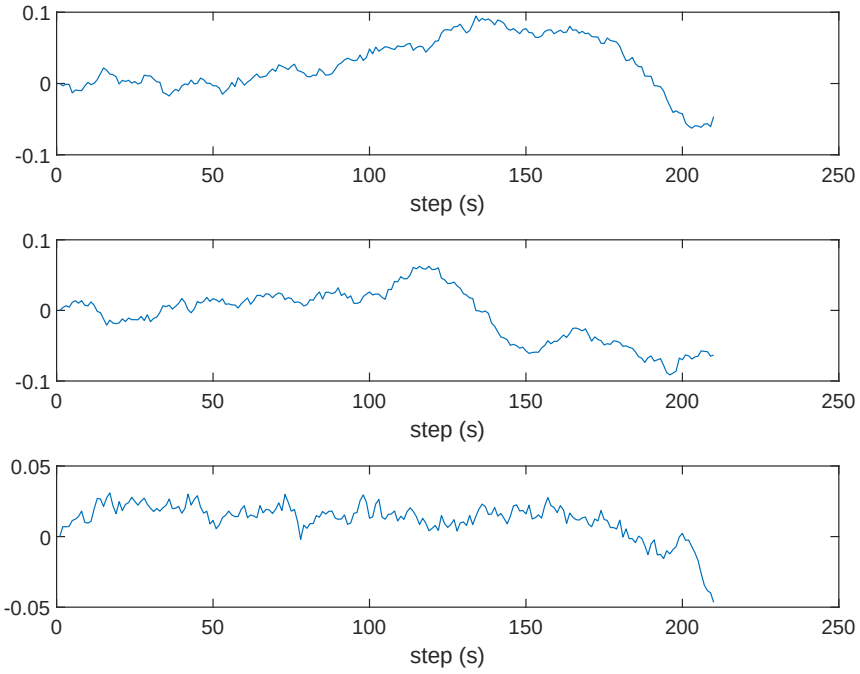


Figure 6.5: Scenario 1 - Vehicles' orientation error with the MHE method

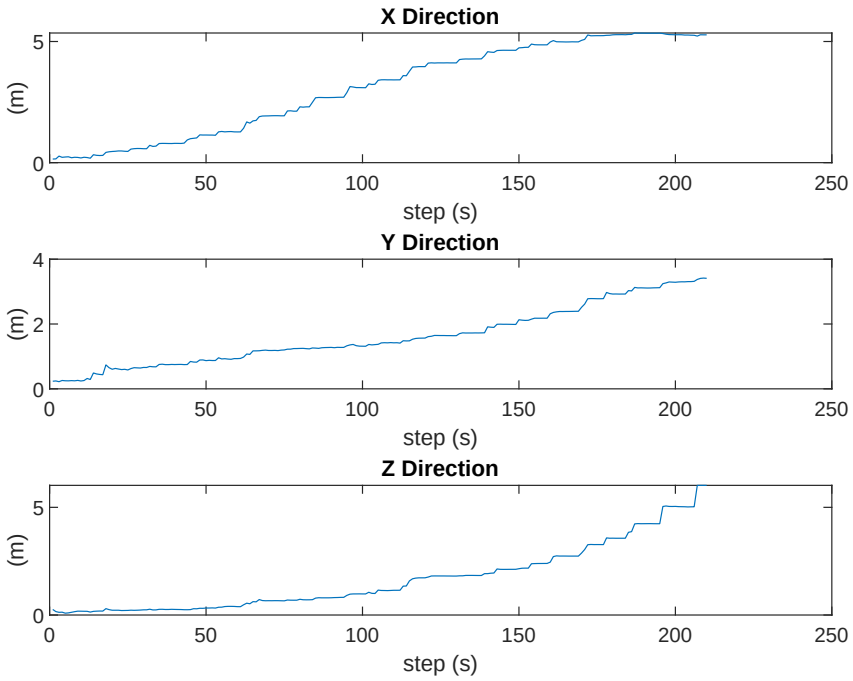


Figure 6.6: Scenario 1 - Sum of the landmarks position error with the MHE method

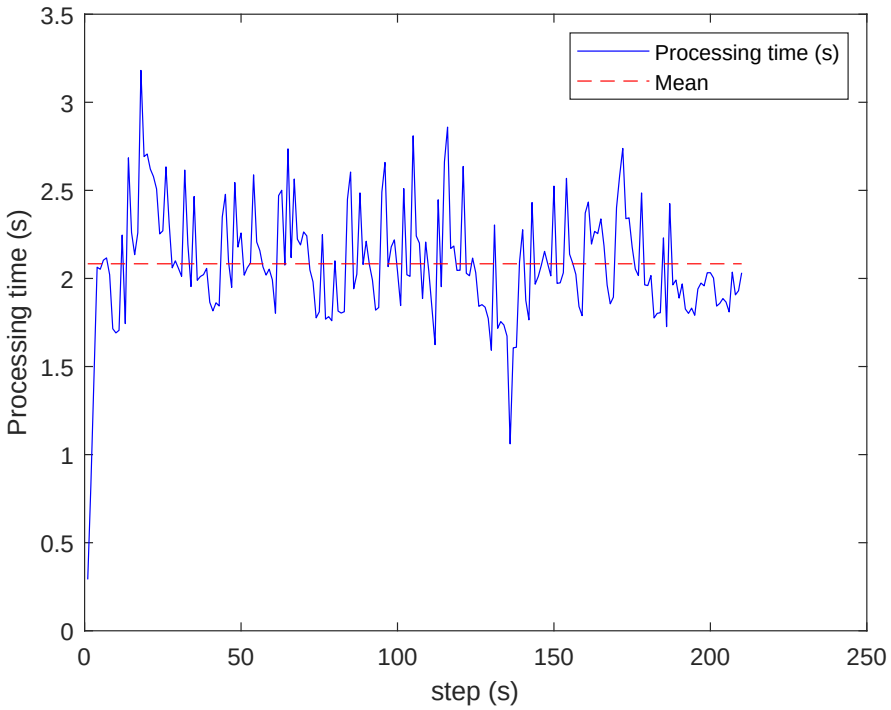


Figure 6.7: Scenario 1 - Processing time at each step for the MHE method

Table 6.1: Scenario 1 - Comparison of the vehicle final position error, sum of landmarks error, average processing time and peak processing time of each optimization algorithm on solving the MHE cost function considering four observed landmarks at each iteration and a horizon of 10 iterations

Method	ABS error of $\tilde{\mathbf{p}}_{k_f}$	Sum of LMs ABS error	Mean t_{CPU}	Max t_{CPU}
Active-set	0.4429 m	19.7292 m	30.2316 s	42.2277 s
SQP	0.4429 m	20.1721 m	22.106 s	29.2181 s
interior-point	0.4429 m	19.7292 m	30.527 s	42.3698 s
bnb	0.4429 m	19.923 m	35.342 s	43.4451 s
bmibnb	0.4429 m	19.923 m	38.782 s	45.2945 s

The optimization algorithm can have an influence on how fast it reaches a solution. To understand which one would be best suited for this specific application, it was considered an horizon of $H = 10$ and compared the error, average computation time per step and the peak computation time per step for MATLABs' fmincon using active-set, SQP and interior-point algorithms, as demonstrated in Table 6.1. Other solvers were also included, such as the bnb and bmibnb, which were integrated in MATLAB using the yalmip package [94]. The algorithms gurobi and mosek, which have free access for academia, and KKTQP could not deal with the polynomial characteristics of the cost function, whilst bmibnb and bnb displayed a big delay compared to the options above, as it uses an upper and lower solver. The Table 6.1 shows that the SQP algorithm has not only a lower average processing time per step, but also a lower peak processing time per step, in relation to the other approaches, whilst maintaining an equal or similar error on the estimated positions.

Another consideration to have in mind, is the influence of the optimization algorithm on the overall stability and performance of the filter. The cost function in equation 6.3 has local minimums, which makes the work of the optimization algorithm hard as they might not guarantee an optimal solution. An approach to solving this issue, would be the use of parallel computing to run multiple instances of the nonlinear optimization algorithm and choosing the one with the lowest cost function value. However, the implementation of parallel computing on MATLAB introduces a noticeable delay when compared to a single run of the optimization algorithm. As such, this option was considered but disregarded as a potential solution for this scenario.

Regardless of the optimization algorithm being used, considering a fewer number of landmarks also makes the MHE algorithm simpler, and as result more efficient. However, this might result on a worse estimation if the noise regarding the sensors is relatively big. Besides that, the size of the horizon is also a big factor on computational complexity as it introduces more constraints and estimation variables into the cost function. With that in mind, Table 6.2 displays the relation between size of the horizon and number of landmarks with the time that each step take using the fmincon's SQP optimization algorithm. Analyzing the Table it is possible to note that the horizon and number of landmarks influences not only the positional error, but also the average processing time.

The bigger the horizon the better for the MHE to adapt to abrupt dynamics such as the rotation at each edge of the square-shaped corridor. Moreover, the number of landmarks identified in each step also guarantees a better estimation of the pose, as they help to filter the noise associated to them.

Table 6.2: Scenario 1 - Influence of the size of the Horizon and the number of landmarks considered on the MHE cost function execution time and final vehicle position error

H	n_O	ABS error of $\tilde{\mathbf{p}}_{k_f}$	Mean t_{CPU}
3	4	0.4789 m	2.086 s
3	8	0.4392 m	4.271 s
5	4	0.4501 m	3.525 s
5	8	0.4445 m	10.803 s
10	4	0.4429 m	22.106 s
10	8	0.4429 m	36.237 s
15	4	0.4429 m	35.004 s
15	8	0.4429 m	47.938 s

Considering the data displayed at tables 6.1 and 6.2, it is considered that the SQP algorithm with a limit of 4 landmarks at each iteration and an horizon of 3 steps is sufficient to our considered scenario, since the lost in performance to other sets of parameters is not relevant enough.

7

Sensor-based LKF for 3-D SLAM

On this Chapter it is presented another solution to the SLAM problem: the LKF sensor-based approach. This methodology turns a nonlinear dynamic system into an almost linear one by revolutionizing the standard SLAM approach, as the system is described in the visual sensor's frame. This characteristic enables the use of the standard KF on the estimation of the landmarks position, as there is no need to linearize the dynamics of the vehicle. As such, the system dynamics designed and presented in Chapter 4 cannot be used in this framework. Instead, the design is based on the work of Lourenço et al. in [62], which presents a 3D sensor-based GAS filter with positive experimental results. The Chapter is divided in: system dynamics, KF design and simulation results.

7.1 Sensor-based system Dynamics

Let the following equations define the position of the landmark and its respective variation in time regarding the sensor frame

$$\mathbf{p}_L = \mathbf{R}^T \Big({}^W\mathbf{p}_L(t) - {}^W\mathbf{p}_V(t) \Big) \tag{7.1}$$

$$\dot{\mathbf{p}}_L(t) = -\mathbf{v}(t) - \mathbf{S}\Big[\boldsymbol{\omega}_m(t)\Big]\mathbf{p}_L(t) \tag{7.2}$$

where $\mathbf{R}(t) \in SO(3)$ is the rotation from the body-frame to the earth-fixed frame that satisfies $\dot{\mathbf{R}}(t) = \mathbf{R}S[\boldsymbol{\omega_m}(t)]$, in which $\boldsymbol{\omega}_m(t) \in \mathbb{R}^3$ is the angular velocity applied on the body [95], and $\mathbf{v}(t) \in \mathbb{R}^3$ is the linear velocity on the body. As mentioned before, on the considered environment it is assumed that the landmarks are fixed, as such the velocities of the landmarks on the sensor-frame will be the negative value of the velocity of the vehicle.

The angular velocity $\boldsymbol{\omega}_m(t)$ is defined differently than the one considered previously, at Chapter 4. In this formulation, $\boldsymbol{\omega}_m(t)$ only considers the actual angular velocities and the biased of the gyros measurements $\mathbf{b}_w(t) \in \mathbb{R}^3$, as in

$$\boldsymbol{\omega}_m(t) = \boldsymbol{\omega}(t) + \mathbf{b}_w(t). \tag{7.3}$$

As a result, the noise on the angular velocity measurements is considered in the system noise and not directly on the measured values, as it was in the previous formulation. Exploiting this equality and considering that $\mathbf{S}[\mathbf{a}]\mathbf{b} = -\mathbf{S}[\mathbf{b}]\mathbf{a}$, equation 7.2 can be defined as

$$\dot{\mathbf{p}}_L(t) = -\mathbf{v}(t) - \mathbf{S}\Big[\mathbf{p}_L(t)\Big]\mathbf{b}_w(t) - \mathbf{S}\Big[\boldsymbol{\omega}_m(t)\Big]\mathbf{p}_L(t). \tag{7.4}$$

The variables $\mathbf{b}_w(t)$ and $\mathbf{v}(t)$ are included on the state vector as part of the vehicle state, $\mathbf{x}_v(t) = \begin{bmatrix}\mathbf{v}^T(t) & \mathbf{b}_w^T(t)\end{bmatrix}^T \in \mathbb{R}^{n_v}$, which are considered to be constant

$$\dot{\mathbf{x}}_v(t) = 0 \tag{7.5}$$

and, as such, Equation (7.4) can be rewritten as

$$\dot{\mathbf{p}}_L(t) = \mathbf{A}_{lv}\Big(\mathbf{p}_L(t)\Big)\mathbf{x}_v(t) - \mathbf{S}\Big[\boldsymbol{\omega}_m(t)\Big]\mathbf{p}_L(t) \tag{7.6}$$

with

$$\mathbf{A}_{lv}(\mathbf{p}_L(t)) = \begin{bmatrix}-\mathbf{I}_3 & -\mathbf{S}\big[\mathbf{p}_L(t)\big]\end{bmatrix}. \tag{7.7}$$

Similarly, to the previous system dynamics it is possible to define the landmarks into two different subgroups, the visible, O, and the non-visible landmarks, U. As a result, the vector state of the landmarks can be described as the following

$$\mathbf{x}_l(t) = \begin{bmatrix}\mathbf{x}_{l_O}^T(t) & \mathbf{x}_{l_U}^T(t)\end{bmatrix} \tag{7.8}$$

in which $\mathbf{x}_{l_O}(t)$ is the collection of observed landmark positions $\mathbf{p}_L(t)$, and $\mathbf{x}_{l_U}(t)$ the collection of unobserved landmark positions, at the instant t.

Considering the complete state vector as

$$\mathbf{x}(t) = \begin{bmatrix}\mathbf{x}_v^T(t) & \mathbf{x}_l^T(t)\end{bmatrix}^T \tag{7.9}$$

the system dynamics can be defined as

$$\begin{cases}\dot{\mathbf{x}}(t) = \mathbf{A}(\mathbf{x}_l)\mathbf{x}(t) \\ \mathbf{y}(t) = \mathbf{x}_{l_O}(t)\end{cases} \tag{7.10}$$

where

$$\mathbf{A}(\mathbf{x}_l) = \begin{bmatrix}\mathbf{0}_{n_v} & \mathbf{0}_{n_v \times n_l} \\ \mathbf{A}_{lv}\big(\mathbf{x}_l(t)\big) & \mathbf{A}_l(t)\end{bmatrix} \tag{7.11}$$

and $\mathbf{A}_l(t)$ and $\mathbf{A}_{lv}(\mathbf{x}_l(t))$ are defined as the influence of $\mathbf{x}_l$ and $\mathbf{x}_v$ on the position of the landmarks, following

$$\mathbf{A}_l(t) = diag\Big(-\mathbf{S}\big[\boldsymbol{\omega}_m(t)\big], ..., -\mathbf{S}\big[\boldsymbol{\omega}_m(t)\big]\Big) \tag{7.12}$$

$$\mathbf{A}_{lv}\big(\mathbf{x}_l(t)\big) = \begin{bmatrix}\mathbf{A}_{lv}\big(\mathbf{x}_1(t)\big) & ... & \mathbf{A}_{lv}\big(\mathbf{x}_{n_l}(t)\big)\end{bmatrix}. \tag{7.13}$$

The discretization of the system was done using the Euler method and can be given by

$$\begin{cases} \mathbf{x}_{k+1} = \mathbf{F}_k \mathbf{x}_k + \mathbf{n}_F \\ \mathbf{y}_{k+1} = \mathbf{H}_{k+1} \mathbf{x}_{k+1} + \mathbf{n}_H \end{cases} \tag{7.14}$$

where $\mathbf{F}_k$ is the transition matrix and $\mathbf{H}_{k+1}$ the measurement matrix. Considering that the state vector is arranged as $\mathbf{x}_k = \begin{bmatrix} \mathbf{x}_v^T & \mathbf{x}_{l_O}^T & \mathbf{x}_{l_U}^T \end{bmatrix}^T$ in each step k and that $\mathbf{F}_k = \mathbf{I}_{n_x} + \delta t . \mathbf{A}(\mathbf{y}_k, \hat{\mathbf{x}}_{l_U k})$, and

$$\mathbf{H}_{k+1} = \begin{bmatrix} \mathbf{0}_{n_{lO} \times n_v} & \mathbf{I}_{n_{lO}} & \mathbf{0}_{n_{lO} \times n_{lU}} \end{bmatrix} \tag{7.15}$$

since it works as a selection of the observed set of landmarks.

The noise vector of the system, ξ_F, and the noise vector of the measurements, ξ_H, are zero-mean white Gaussian noise with a vector of standard deviations of σ_F and σ_H, respectively. This discretized system estimates the unobserved landmarks position based on their previous estimation, whilst using the sensors readings for the observed ones.

7.2 KF design

The application of a Kalman filter on the discretized system displayed at Equation (7.14) can be done through the following prediction and update steps.

7.2.1 Prediction Step

Similar to the EKF on Chapter 5, the prediction step estimates the next values of the state vector based on the systems motion model and the error covariance matrix associated with such movement, as such it can be defined as a estimation of $k+1$ based on k, according to the following equations

$$\begin{cases} \hat{\mathbf{x}}_{k+1} = \mathbf{F}_k \hat{\mathbf{x}}_k + \xi_F \\ \mathbf{\Sigma}_{k+1|k} = \hat{\mathbf{F}}_k \mathbf{\Sigma}_{k|k} \hat{\mathbf{F}}_k^T + \mathbf{Q}_k \end{cases} \tag{7.16}$$

in which, the Jacobian of the system dynamics, $\hat{\mathbf{F}}_k$, is defined as

$$\hat{\mathbf{F}}_k = \begin{bmatrix} \mathbf{I}_{(n_v + n_{l_O})} + \Delta_t \mathbf{A}(\mathbf{y}_k) & \mathbf{0}_{(n_v + n_{l_O}) \times n_{l_U}} \\ \begin{bmatrix} \mathbf{A}_{lv}(\hat{x}_{l_U}) & \mathbf{0}_{n_{l_U} \times n_{l_O}} \end{bmatrix} & -diag\left(\mathbf{S}[\boldsymbol{\omega}_m - \hat{\mathbf{b}}_\omega]\right) \end{bmatrix} \tag{7.17}$$

and $\mathbf{Q}_k = diag(\xi_F \xi_F^T)$.

7.2.2 Update Step

The update step corrects the estimation done on the prediction step of the visualized landmarks. The update step of the LKF is given by the following set of equations

$$\hat{\mathbf{x}}_{k+1|k+1} = \hat{\mathbf{x}}_{k+1|k} + \mathbf{K}_{k+1}\tilde{\mathbf{y}}_{k+1} \tag{7.18}$$

$$\mathbf{\Sigma}_{k+1|k+1} = \mathbf{\Sigma}_{k+1|k} - \mathbf{K}_{k+1}\mathbf{H}_{k+1}\mathbf{\Sigma}_{k+1|k} \tag{7.19}$$

in which $\hat{\mathbf{x}}_{k+1|k+1}$ is the updated state vector, $\mathbf{\Sigma}_{k+1|k+1}$ the updated covariance matrix, $\tilde{\mathbf{y}}_{k+1}$ the measurement residual, $\mathbf{S}_{k+1}$ the residual covariance and finally $\mathbf{K}_{k+1}$ the Kalman gain. These values can be defined by

$$\tilde{\mathbf{y}}_{k+1} = \mathbf{y}_{k+1} - \mathbf{H}_{k+1}\hat{\mathbf{x}}_{k+1|k} \tag{7.20}$$

$$\mathbf{S}_{k+1} = \mathbf{H}_{k+1}\mathbf{\Sigma}_{k+1|k}\mathbf{H}_{k+1}^T + \mathbf{R}_k \tag{7.21}$$

$$\mathbf{K}_{k+1} = \mathbf{\Sigma}_{k+1|k}\mathbf{H}_{k+1}^T\mathbf{S}_k^{-1} \tag{7.22}$$

and $\mathbf{R}_k = diag(\xi_H\xi_H^T)$.

7.3 Simulation results

On this Section it is presented the simulation considering the Scenario 1, and the results obtained on such environment using the sensor-based LKF filter. Additionally, it will be discussed its performance regarding the estimation.

7.3.1 Simulation environment

The simulated environment is identical to the one described in the previous Chapter, a 3-D square shaped corridor, with forty-eight landmarks along its sides and edges. A simulation of 250 seconds to be complete and the drone has an average speed of 0.1 m/s. The simulation system disturbances are zero-mean white Gaussian noise and have the following standard deviations

$$\sigma_{\xi_F} = \begin{bmatrix} \sigma_{\xi_v} & \sigma_{\xi_w} \end{bmatrix} \tag{7.23}$$

$$\sigma_{\xi_v} = \begin{bmatrix} 0.015 & 0.015 & 0.015 \end{bmatrix} m/s \tag{7.24}$$

$$\sigma_{\xi_w} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} deg/s \tag{7.25}$$

$$\sigma_H = \begin{bmatrix} 5.0 & 5.0 & 5.0 \end{bmatrix} cm. \tag{7.26}$$

The vehicle values of linear and angular velocities over time are exactly the same as the ones used in Chapters 5 and 6, as represented in Figures 5.1 and 5.2.

7.3.2 Results

To give meaning to the estimation on the sensor-based approach, the values estimated can be converted into the earth-fixed frame. As such, at each step the approximate rotation and translation need to be calculated having in consideration the position of the visible landmarks at the present and previous step. Which can be found solving the optimization problem [96]

$$(\delta\mathbf{R}^*_{k+1}, \delta\mathbf{p}^*_{k+1}) = \arg \min_{\delta\mathbf{R}_{k+1}, \delta\mathbf{p}_{k+1}} \frac{1}{n_{\delta_{k+1}}} \sum_{i \in O} \|\mathbf{e}_{i_{k+1}}\|^2 \tag{7.27}$$

where $e_{i_{k+1}}$ is the error defined by

$$\mathbf{e}_{L_{k+1}} = \mathbf{p}_{L_{k+1}} - \delta\mathbf{R}_{k+1}\mathbf{p}_{L_k} - \delta\mathbf{p}_{V_{k+1}}. \tag{7.28}$$

The problem defined in (7.27) can be solved by means of the Singular Value Decomposition (SVD), in order to find the rotation and translation between the sets of visible landmarks at $k+1$ and k. The SVD can decompose a matrix into three matrices, $\mathbf{U}_{k+1}$ and $\mathbf{V}_{k+1}$ orthogonal and a diagonal matrix with singular values $\mathbf{D}_{k+1}$. The centroid of both sets of landmarks at the current and previous step are given by

$$\boldsymbol{\mu}_{k+1} = \frac{1}{n_{Om}} \sum_{i \in O_L} \mathbf{p}_{i_{k+1}} \tag{7.29}$$

$$\boldsymbol{\mu}_k = \frac{1}{n_{O_m}} \sum_{i \in O_L} \mathbf{p}_{i_k} \tag{7.30}$$

in which O_L is the set of landmarks visualized in both instances. Considering that

$$\boldsymbol{\Pi}_{k+1} = \left(\mathbf{x}_{l_O k+1} - \left[\boldsymbol{\mu}_{k+1}\right]\right)\left(\mathbf{x}_{l_O k} - \left[\boldsymbol{\mu}_k\right]\right)^T \tag{7.31}$$

and that

$$\mathbf{U}_{k+1}\mathbf{D}_{k+1}\mathbf{V}_{k+1} = SVD\left(\boldsymbol{\Pi}_{k+1}\right) \tag{7.32}$$

it is possible to find the optimal rotation and translation through

$$\begin{cases} \delta\mathbf{R}^*_{k+1} = \mathbf{V}_{k+1}\mathbf{U}^T_{k+1}, & \text{if } det(\mathbf{V}_{k+1}\mathbf{U}^T_{k+1}) \geq 0 \\ \delta\mathbf{R}^*_{k+1} = diag\left(\begin{bmatrix} 0 & 0 & -1 \end{bmatrix}\right)\mathbf{V}_{k+1}\mathbf{U}^T_{k+1}, & \text{if } det(\mathbf{V}_{k+1}\mathbf{U}^T_{k+1}) < 0 \end{cases} \tag{7.33}$$

$$\delta\mathbf{p}^*_{k+1} = -\delta\mathbf{R}^*_{k+1}\boldsymbol{\mu}^T_{k+1} + \boldsymbol{\mu}^T_k. \tag{7.34}$$

It should be taken into account Equation (7.33), specially to the determinant of $\mathbf{V}_{k+1}\mathbf{U}^T_{k+1}$. If negative, the rotation obtained is in fact a reflection which has no physical meaning, and the last column of $\mathbf{V}_{k+1}$ has to be multiplied by -1.

The true and estimated trajectories, represented by red and blue respectively, can be evaluated through Figures 7.1 and 7.2, from which it is possible to recognise that the estimation, even if not very smooth, is proximate to reality.

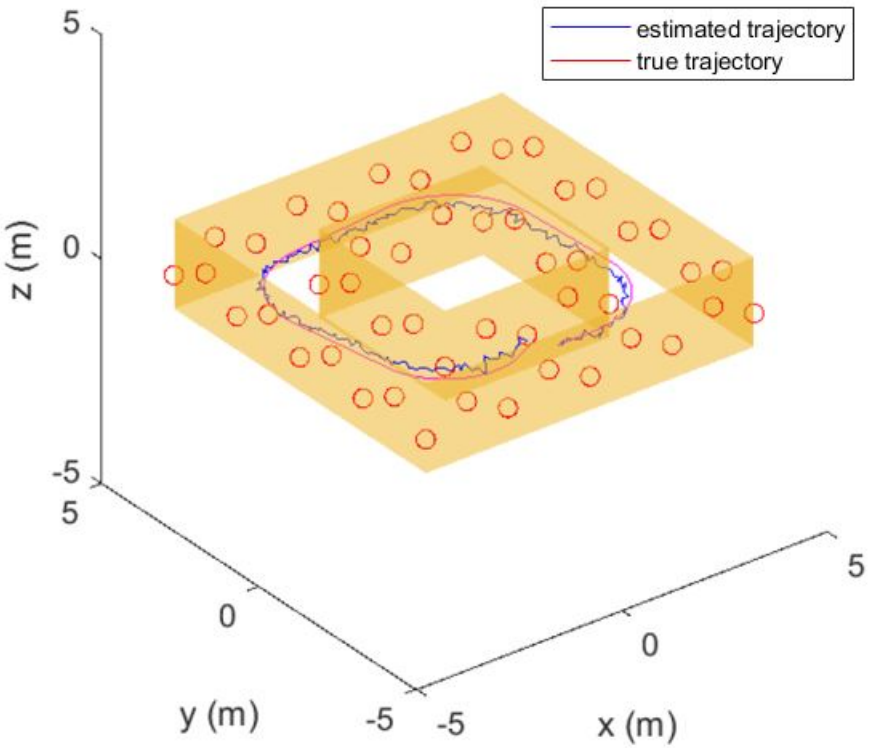


Figure 7.1: Scenario 1 - Vehicle position error with the LKF sensor-based method

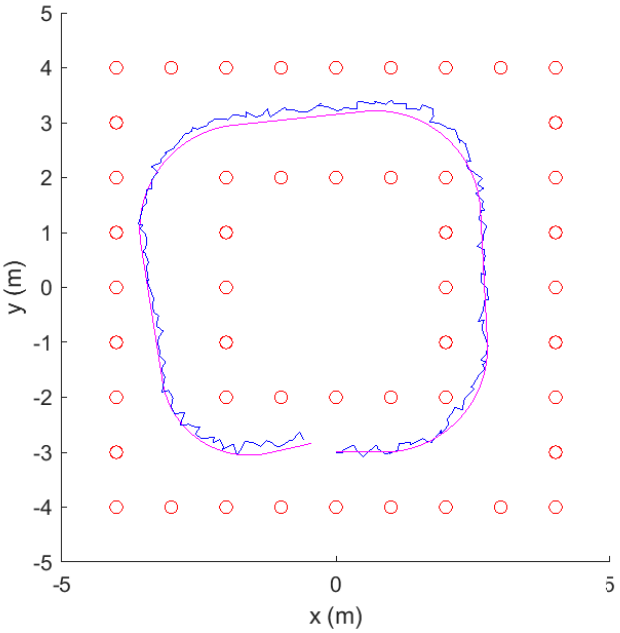


Figure 7.2: Scenario 1 - Vehicle orientation error with the LKF sensor-based method

In order to better understand the performance of this filter, the evolution of the estimated vehicle position error is presented in Figure 7.3, in which is possible to visualize that there is a considerable small error regarding the estimation. Moreover, the orientation error, calculated using equation (5.20), is displayed in Figure 7.4. The orientation error is also within acceptable values, as the changes in angular velocity introduce error, but not enough to destabilize the estimation. This results are in accordance with the work of Lourenço et al. in [62] and Guerreiro et al. in [96] wich present and validate the relevance of this approach as a solution to the SLAM problem. However, the assumption of a constant linear velocity introduces errors to the dynamics of the system whenever there is a big change in angular velocity. Moreover, the transformation of the sensor-based information into the world-frame through Equations (7.33) and (7.34), introduces errors on the estimated robot position.

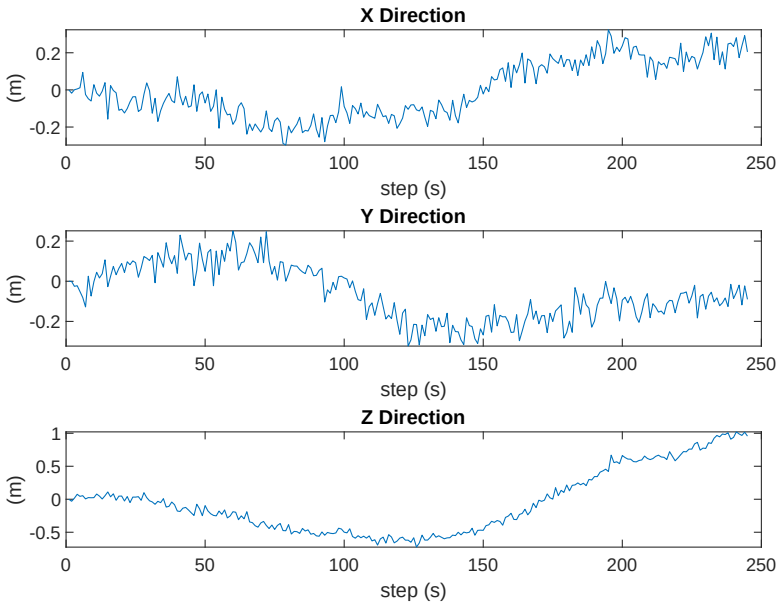


Figure 7.3: Scenario 1 - Vehicle position error with the LKF sensor-based method

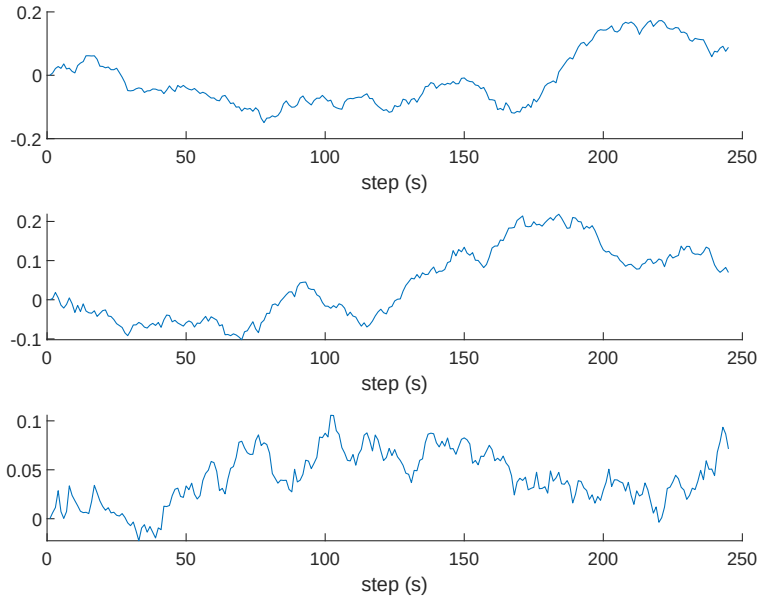


Figure 7.4: Scenario 1 - Vehicle orientation error with the LKF sensor-based method

8

Comparative results and discussion

This Chapter introduces a more challenging 3-D scenario to compare the algorithms presented in the previous Chapters, aiming at comparing the performance and robustness of each the EKF, MHE and LKF approaches to SLAM.

A first set of results involves the comparison and discussion of the results presented separately in Chapters 5, 6, and 7 in a joint manner. These results use the Scenario 1, as depicted in Figure 6.2. For the formulation of the environment it was considered that the MHE had an horizon equal to three, the number of landmarks to be used on each iteration was limited to five and the weights of its cost function were the weights of the EKF covariance matrices. On Figure 8.1 and 8.2 it is possible to evaluate which algorithm performed the best regarding the estimation of the vehicle position and orientation, which was calculated taking in consideration equation (5.20). The sum of the absolute error of the EKF position was of 64.39 whilst the error using the MHE was 46.99 and the LKF was 75.59. Taking into consideration these values it is possible to recognise that each of the algorithms performed well under the defined circumstances, however the MHE had the best overall estimation. Regardless, it is not reasonable to compare the performance of the EKF and MHE algorithms to the LKF, as the LKF uses a completely different formulation of its dynamics from the other two. However, from the three it is the only filter capable of directly estimate the vehicle velocity. It is also possible to recognize that the error pattern for the EKF and for the MHE has similar dynamics. Which is expected to an extent, since in certain conditions the MHE is equivalent to the EKF, as explained by Rao et al. in [43]. Moreover, in Figure 8.3 it is possible to compare the performance of the MHE and EKF algorithms regarding the error of the landmarks position. From which it is possible to conclude that the EKF had a worse overall performance, as it had a bigger error in the x and y axis, whilst having better estimation of the z axis.

To validate the experiment on more challenging conditions, another simulation environment was set up: Scenario 2. Similarly to the previous one, it has a loop on the XY plane, but it was added another one on the XZ plane: after a horizontal loop, the vehicle returns to its' initial position and does a vertical loop. This scenario not only simulates the dynamics of the vehicle on a corridor on the XY plane, but also considers the Z axis,

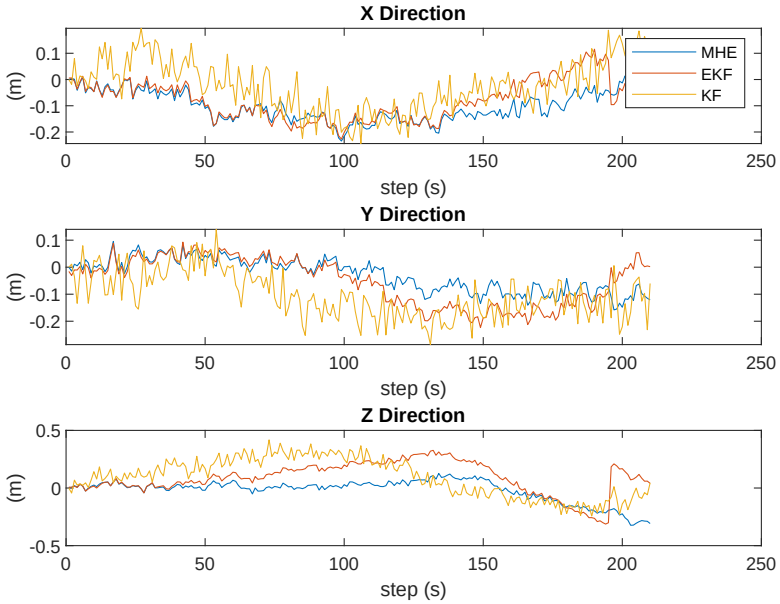


Figure 8.1: Scenario 1 - Vehicle position error between the MHE, EKF and LKF algorithms

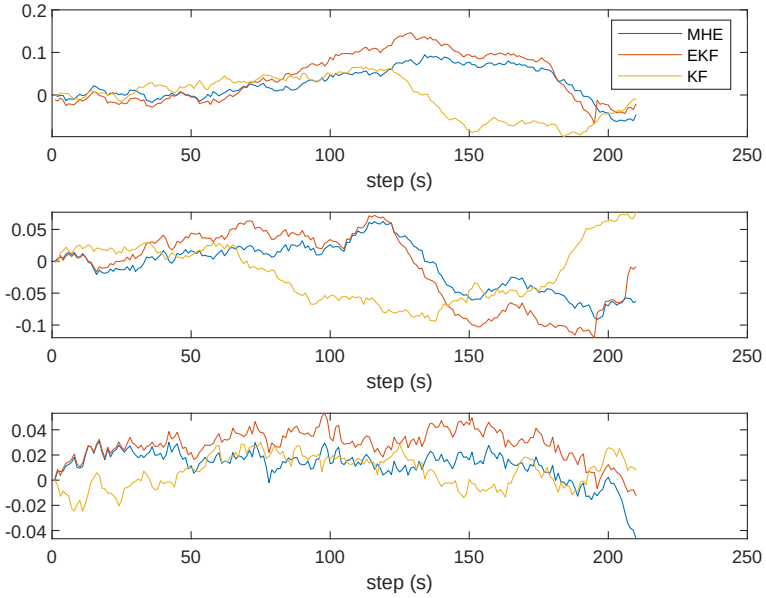


Figure 8.2: Scenario 1 - Comparison of the vehicle orientation error between the MHE, EKF and LKF algorithms

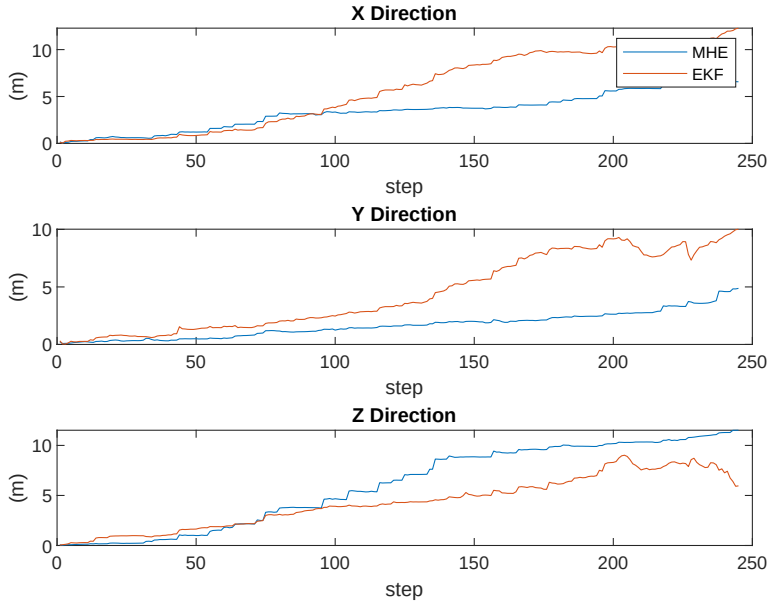


Figure 8.3: Scenario 1 - Comparison of the estimated landmarks position error between the MHE and the EKF algorithms

which is relevant having in account that any 6 DoF vehicle can move over the 3 Cartesian axis. During this scenario the linear and angular velocities suffered from a similar noise as previously, as the standard deviations were the ones previously defined. These input values are depicted in Figures 8.4 and 8.5 during the simulation period in Scenario 2 of 485 seconds. In Figure 8.4 it is possible to note that the linear motion of the rigid-body mainly moves on the x direction, as the other axis are only disturbed by noise. On the other hand, in Figure 8.5 the changes in direction during loops, which happen as rotations on the z axis, are noticeable, as well as the one in the y axis which represents the change between loops.

The results of the three algorithms are discussed regarding this new Scenarion. The Evolution of the trajectory for the EKF, MHE and the LKF can be observed on Figures 8.6, 8.7, 8.8, 8.9, 8.10 and 8.11 where it can be seen both loops and that the filters follow the true trajectory dynamics quite well. Moreover, on Figures 8.12 and 8.13 it is displayed the vehicle estimated EKF, MHE and LKF position and orientation, respectively. Through these simulations it is possible to analyse that overtime the estimation of the MHE algorithm gets worse than the estimation of the EKF, which did not happen previously. This can be explained by the capability of the EKF algorithm to close the loop, as it identifies that the landmarks at the end of the first loop are the same as the ones it started the simulation with, which have a smaller covariance associated to them, and corrects the positioning of the vehicle and the previous landmarks. This also happens to an extent in MHE, but it is only able to correct estimations of the previous landmarks that are still on the horizon and gets a worse performance with the increase of the horizon H. Since, the bigger the

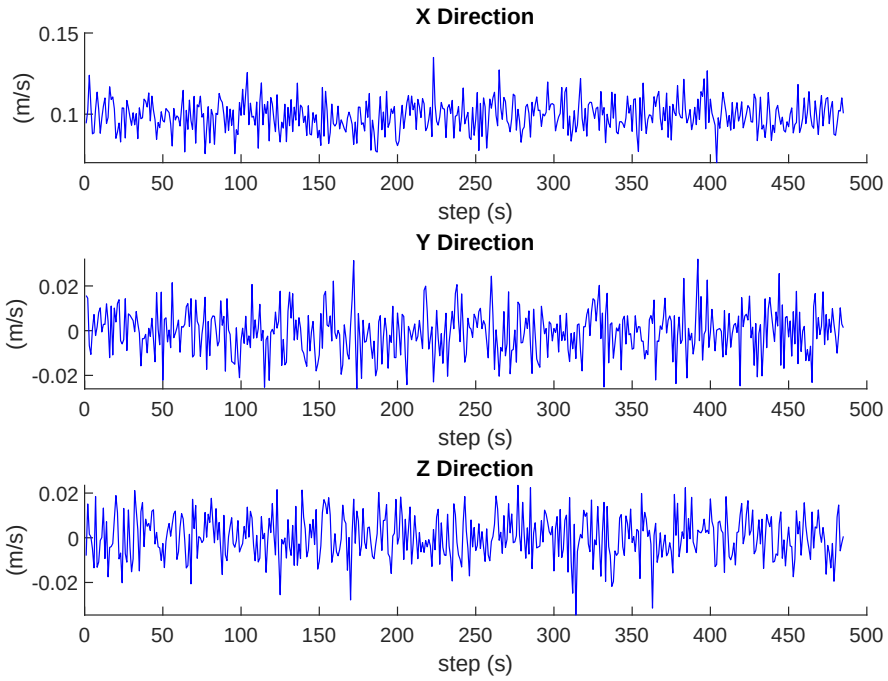


Figure 8.4: Scenario 2 - input values of linear velocities

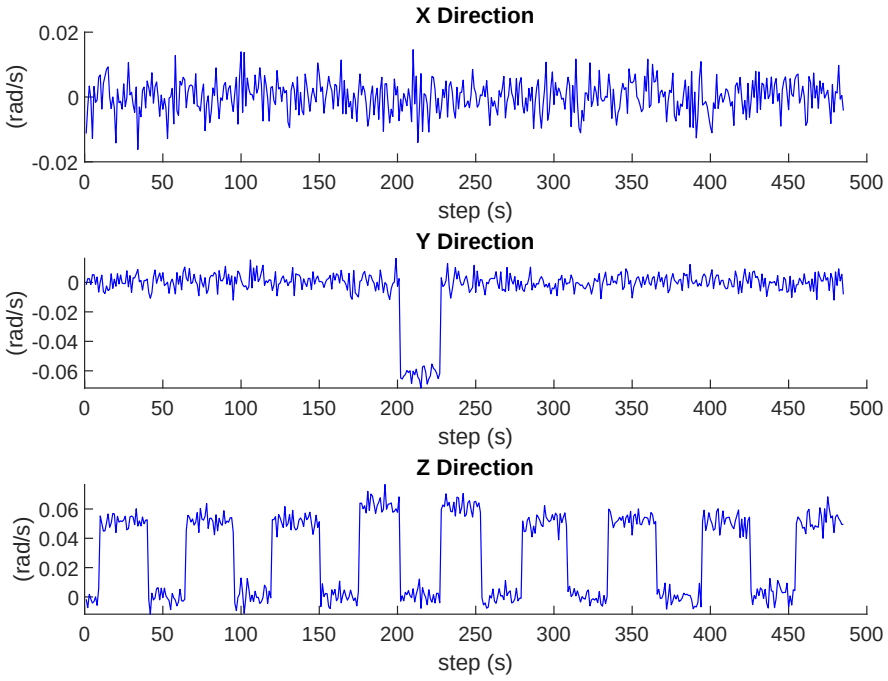


Figure 8.5: Scenario 2 - input values of angular velocities

window is, the less relevance a single landmark will have in the optimal solution for the cost function J, result in a tenuous effect on the estimation. Moreover, from Figure 8.13 can also be understood that, even though the pose estimation of the LKF is on par with the others, its orientation error was more influenced by noise, specially on the z axis. This can be a result of the transformation from the sensor-based to the world-frame. The evolution of the error related to the estimated landmarks position is depicted in Figure 8.14, from which it is possible to identify the closing loop of the EKF and the propagation of the error on the MHE, specially on the z axis.

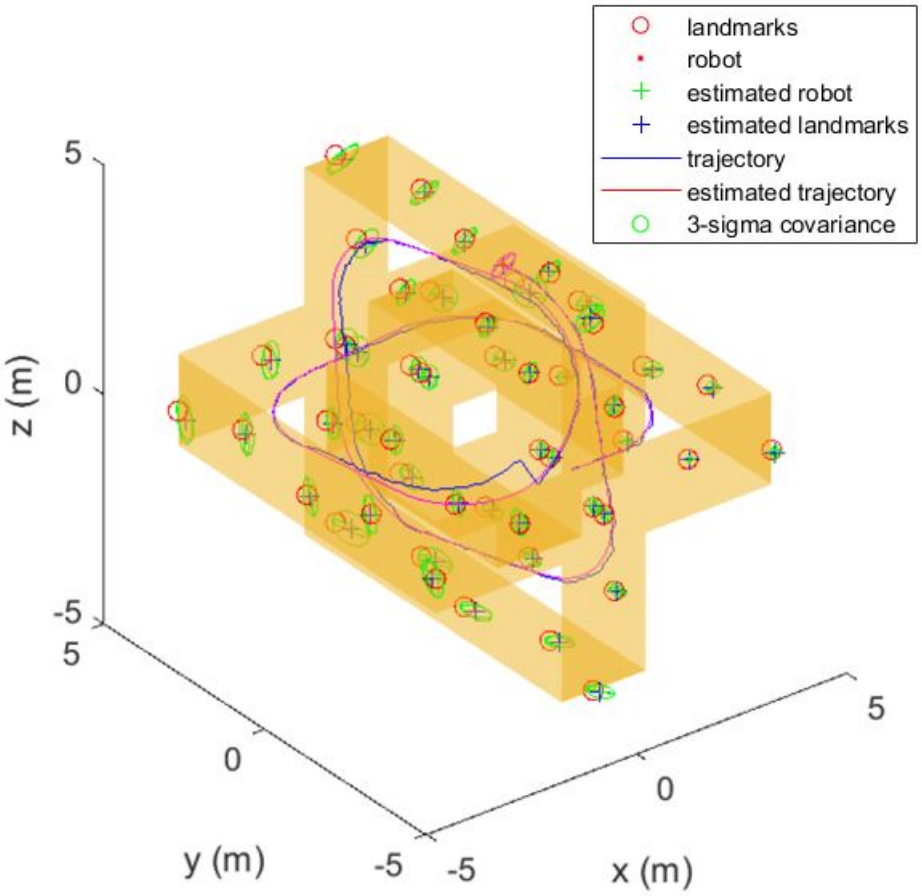
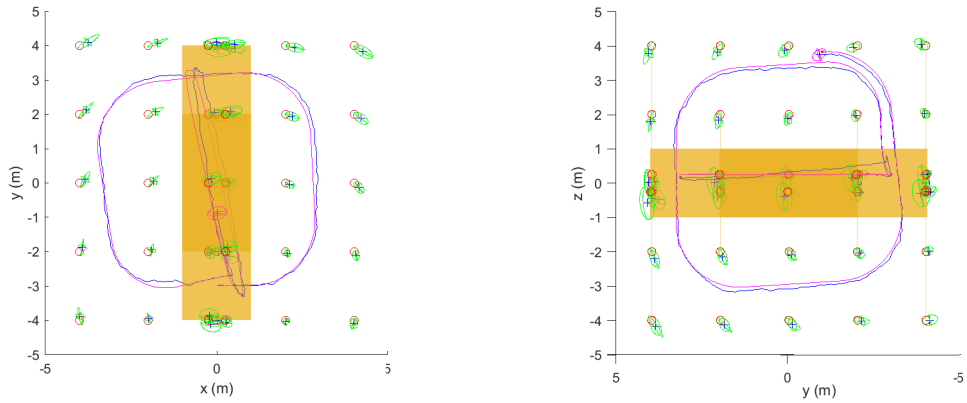


Figure 8.6: Scenario 2 - EKF-based SLAM method

Lastly, the computation time, considering Scenarios 1 and 2, for each of the approaches is presented in Table 8.1, in which can be observed that the computation time of the MHE is extremely big when compared to the other two. In which it is noticeable that the MHE presented an average computation time 100 times bigger than the EKF and 1000 times bigger than the LKF, which can be explained by having to solve a non-linear optimization problem on the MHE approach. Additionally, it is possible to recognise that the overall estimation time grows larger with the duration of the simulation. This can be explained by the influence of the accumulated noise on the estimation process.



(a) XY plane (b) YZ plane

Figure 8.7: Scenario 2 - EKF-based SLAM method

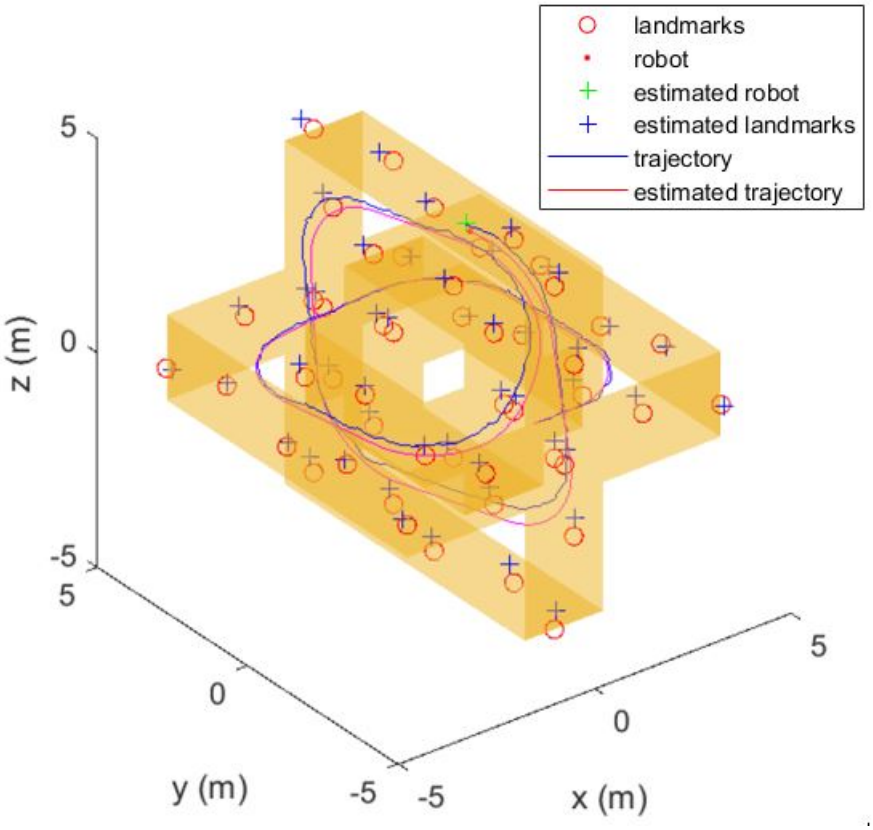
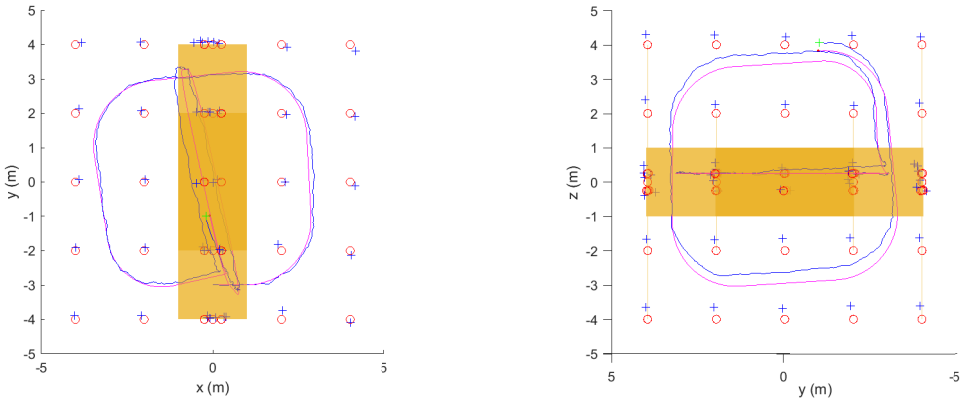


Figure 8.8: Scenario 2 - MHE-based SLAM method

Table 8.1: Computation times of EKF, MHE and LKF in Scenarios 1 and 2

	Scenario 1	Scenario 2
EKF	0.0196 s	0.0218 s
MHE	2.751 s	2.974 s
LKF	0.0022 s	0.0058 s



(a) XY plane (b) YZ plane

Figure 8.9: Scenario 2 - MHE-based SLAM method

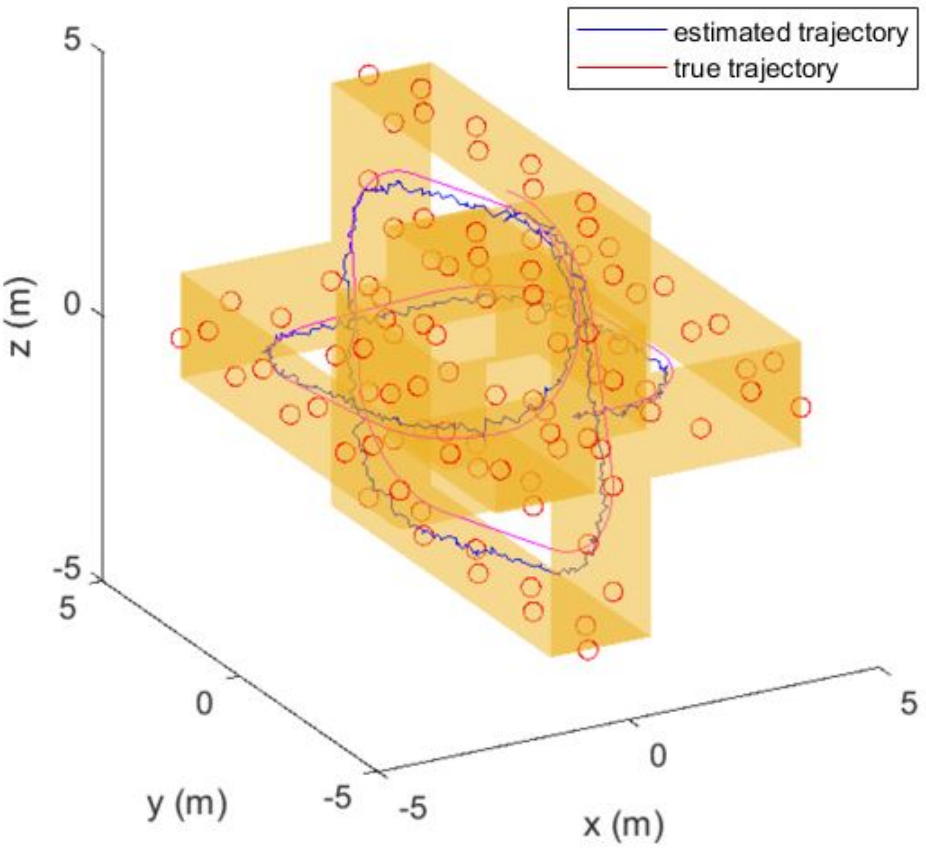
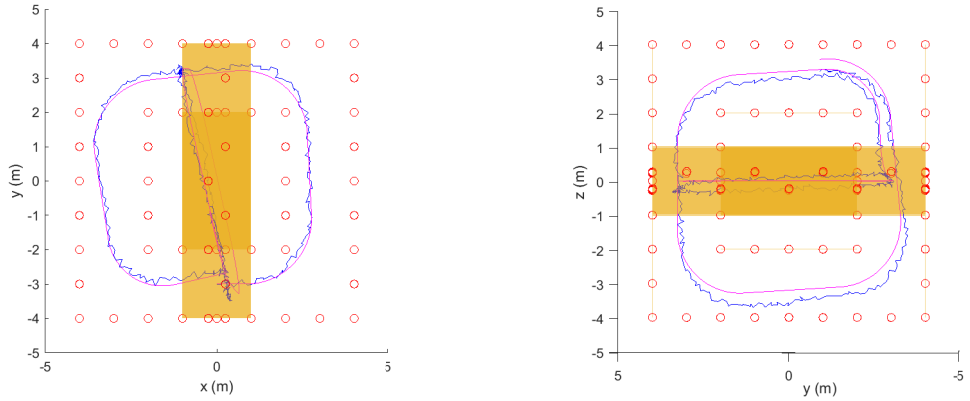


Figure 8.10: Scenario 2 - LKF-based SLAM method



(a) XY plane (b) YZ plane

Figure 8.11: Scenario 2 - LKF-based SLAM method

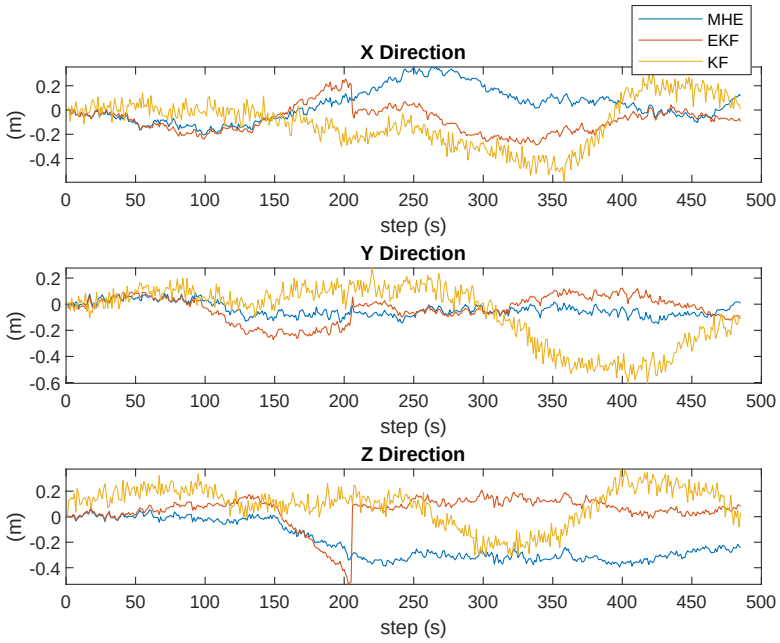


Figure 8.12: Scenario 2 - Comparison of the vehicle positional error between the MHE, EKF and LKF algorithms

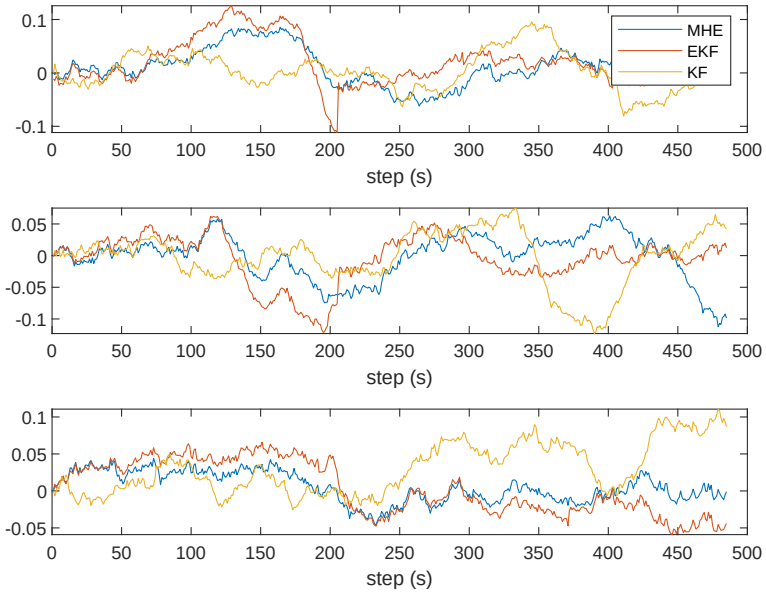


Figure 8.13: Scenario 2 - Comparison of the vehicle orientation error between the MHE, EKF and LKF algorithms

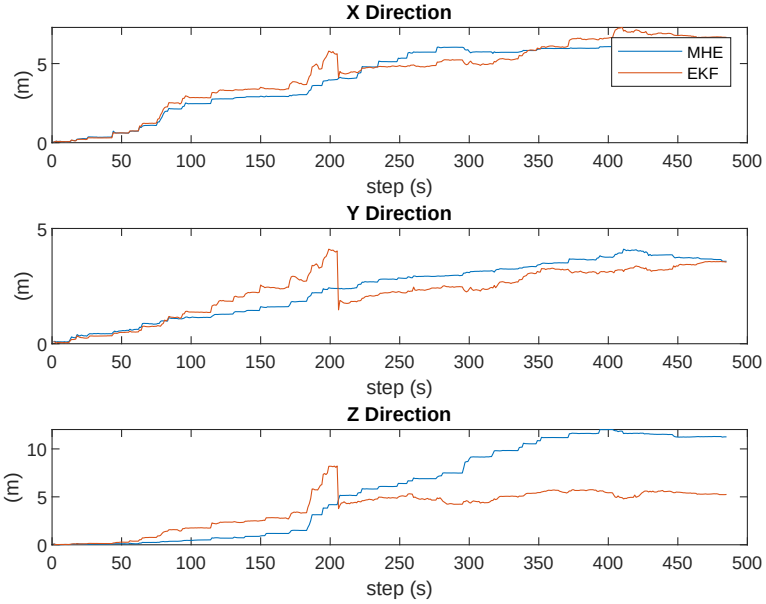


Figure 8.14: Scenario 2 - Comparison of the sum of the estimated landmark error between the MHE and the EKF algorithms

9

Conclusions

The objective of this thesis was to investigate the feasibility, through simulation, of the MHE algorithm as a solution to the SLAM problem of vehicles in 3D space with six DoF, as that work, to the best of the author's knowledge, had never been done.

During this document it was presented not only information regarding the history and the current state of the art on the SLAM problem, but also discussed the role of the MHE, EKF and sensor-based LKF on solving the SLAM formulation. As such three different approaches were developed and taken into consideration: an EKF, a MHE and a sensor-based LKF algorithms.

The system dynamics of the first two was design as depicted in Chapter 4, in which the model of a rigid body with six DoF was created. Besides that, the measurement model was made having into account that the input values were the position of the landmarks related to the rigid-body frame. As for the movement of the body, linear and angular velocities are needed to estimate its position.

The choice of using the EKF as one of the approaches, was made having in consideration that it is widely used in general estimation and, in particular, in the SLAM framework, when the dynamics of the system are not linear. However it is not an optimal solution, as showed throughout this document, as it does a linearization of the nonlinear dynamics. The results obtained showcased that, the estimation error was not considerably big when considering the disturbances, but was bigger than the results of the MHE.

The MHE was the actual case study of this dissertation, as its application as a SLAM approach is still a maturing concept, which shows the relevance of the work done in this dissertation. The MHE was designed having in mind a probabilistic approach based on the covariance values of the system. For that, the integration of an EKF filter was made, in which it was calculated the process covariance matrix as well as the system covariance matrix. The measurement covariance matrix was constant, as the observation model was only affected by the sensors error. Regarding the performance of the MHE it was possible to conclude that the MHE is heavily constrained by the time that it takes to solve its cost function. A few different optimization algorithms, such as yalmip's bnb and bmibnb, and MATLAB's active-set, SQP and interior-point, were tested in order to reduce the

computation time. The best result was achieved with the MATLAB's SQP algorithm, as it not only guaranteed a smaller average processing time, but it also displayed the smaller maximum processing time. Moreover, it was also stated that the size of the considered past data window H also has a big impact on the estimation time, as it introduces more variables and constraints to be estimated, which might result in a stronger non-convex optimization problem. Additionally, it was also tested the influence of the number of observed landmarks on the estimation error and processing time, which resulted on finding that, for the considered Scenarios 1 and 2, a number of four landmarks was sufficient for the estimation and guaranteed a drastically improvement on the solving time of the cost function J. From Scenario 1 it is possible to recognise the good performance of the MHE when compared to the other solutions. However, on Scenario 2 it can be recognised the importance of the loop closure capability, specially of the EKF, as it is possible to see the correction of its position and landmark estimations. As mentioned before, this capability is almost nonexistent on the probabilistic approach to MHE as the importance of each observation is exclusively defined by the weights, in this case the covariance matrices.

Lastly, for the sensor-based LKF approach it had to be defined another formulation of the system dynamics, as they have to be constructed regarding the sensor frame. Even though, the results were transformed from the body-frame to the world-frame, which introduces further errors in the estimation, it was possible to recognise the value of this type of approach, as the estimated position and orientation are on par with the values obtained on the other two methods.

From the results, it is possible to conclude the satisfactory performance of the three approaches considered in this document as solutions to the SLAM problem, having the MHE displayed a better estimation when there are no loop closures.

In light of the results obtained for the MHE, the author recommends the use of this method when dealing with a less dynamic vehicle, as they require a much lower estimation frequency. This is the case for the tricycle in the work of Kasahara et al. [45], which used the same optimization algorithm (SQP) as the one here proposed as the best to tackle the MHE SLAM problem. Nevertheless, it should be pointed that the MHE was not optimized to its fullest, had not this project been done on MATLAB2021a. An application running online would require the proposed filter to be designed on a compiled language, such as Fortran or C/C++, for quicker and more efficient usage of CPU resources. Besides that, other third-party optimization algorithms might be worth considering, as the ones used in this work were either free to use or had an academic license.

To conclude, even though the application of the MHE is partially conditioned by the computing power onboard of the robot, the author believes that in the future that will not be a constraint and this algorithm will be used in numerous applications where nonlinearities and dynamic environments are present.

9.1 Future Research

As the author believes in the success of this method as a solution to the SLAM problem, it also recognises that further investigation should be done to make it more robust. One of the aspects that should be worked on is the inability of the probabilistic MHE of not being able to correct past estimations when it reaches a loop closure event. A backwards approach to the MHE could be triggered in such events as it is done in other SLAM algorithms. This backwards algorithm would be running in parallel with the other instance of the MHE in order to not extend the computation time. Besides that, the development of a sensor-based approach to the MHE should be considered, as it would reduce the complexity of the cost function, resulting in a more reliable estimation when dealing with harsh dynamics. Lastly, a real-time experiment with real data and optimized libraries would be an interesting information to corroborate the results found throughout this dissertation.

Bibliography

[1] H. Durrant-Whyte and T. Bailey. "Simultaneous localization and mapping: part I". In: *IEEE Robotics Automation Magazine* 13.2 (2006), pp. 99–110. DOI: 10.1109/MRA.2006.1638022 (cit. on pp. 1, 14).

[2] *Project CAPTURE*. Accessed: 2022-12-17. URL: http://capture.isr.tecnico.ulisboa.pt/ (cit. on pp. 1, 2).

[3] F. Fahimi. *Autonomous robots*. Springer, 2009 (cit. on p. 2).

[4] M. Hassan. *pixabay Database*. https://pixabay.com/. Accessed: 2023-01-25 (cit. on p. 3).

[5] S. Hayat, E. Yanmaz, and R. Muzaffar. "Survey on Unmanned Aerial Vehicle Networks for Civil Applications: A Communications Viewpoint". In: *IEEE Communications Surveys Tutorials* 18.4 (2016), pp. 2624–2661. DOI: 10.1109/COMST.2016.2560343 (cit. on p. 3).

[6] H. Shakhatreh et al. "Unmanned Aerial Vehicles (UAVs): A Survey on Civil Applications and Key Research Challenges". In: *IEEE Access* 7 (2019), pp. 48572–48634. DOI: 10.1109/ACCESS.2019.2909530 (cit. on p. 3).

[7] J. Kim et al. "Unmanned Aerial Vehicles in Agriculture: A Review of Perspective of Platform, Control, and Applications". In: *IEEE Access* 7 (2019), pp. 105100–105115. DOI: 10.1109/ACCESS.2019.2932119 (cit. on p. 3).

[8] E. Kulu. "Nanosatellite Launch Forecasts-Track Record and Latest Prediction". In: (2022) (cit. on p. 3).

[9] R. C. Smith and P. Cheeseman. "On the representation and estimation of spatial uncertainty". In: *The international journal of Robotics Research* 5.4 (1986), pp. 56–68 (cit. on p. 5).

[10] H. F. Durrant-Whyte. "Uncertain geometry in robotics". In: *IEEE Journal on Robotics and Automation* 4.1 (1988), pp. 23–31 (cit. on p. 5).

[11] R. Smith, M. Self, and P. Cheeseman. "Estimating uncertain spatial relationships in robotics". In: *Autonomous robot vehicles*. Springer, 1990, pp. 167–193 (cit. on p. 5).

[12] C. Cadena et al. "Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age". In: *IEEE Transactions on Robotics* 32.6 (2016), pp. 1309–1332. DOI: 10.1109/TRO.2016.2624754 (cit. on p. 5).

[13] T. Bayes. "LII. An essay towards solving a problem in the doctrine of chances. By the late Rev. Mr. Bayes, FRS communicated by Mr. Price, in a letter to John Canton, AMFR S". In: *Philosophical transactions of the Royal Society of London* 53 (1763), pp. 370–418 (cit. on p. 6).

[14] G. Bresson et al. "Simultaneous localization and mapping: A survey of current trends in autonomous driving". In: *IEEE Transactions on Intelligent Vehicles* 2.3 (2017), pp. 194–220 (cit. on p. 6).

[15] J. Mullane et al. "A random set formulation for Bayesian SLAM". In: *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2008, pp. 1043–1049 (cit. on p. 6).

[16] G. Bresson et al. "Simultaneous Localization and Mapping: A Survey of Current Trends in Autonomous Driving". In: *IEEE Transactions on Intelligent Vehicles* 2.3 (2017), pp. 194–220. DOI: 10.1109/TIV.2017.2749181 (cit. on pp. 6, 15).

[17] T. T. O. Takleh et al. "A brief survey on SLAM methods in autonomous vehicle". In: *International Journal of Engineering & Technology* 7.4 (2018), pp. 38–43 (cit. on p. 6).

[18] R. E. Kalman. "A new approach to linear filtering and prediction problems". In: (1960) (cit. on p. 6).

[19] R. E. Kalman et al. "Contributions to the theory of optimal control". In: *Bol. soc. mat. mexicana* 5.2 (1960), pp. 102–119 (cit. on p. 6).

[20] R. J. Meinhold and N. D. Singpurwalla. "Understanding the Kalman filter". In: *The American Statistician* 37.2 (1983), pp. 123–127 (cit. on p. 6).

[21] H. W. Sorenson. "Least-squares estimation: from Gauss to Kalman". In: *IEEE spectrum* 7.7 (1970), pp. 63–68 (cit. on p. 6).

[22] G. Welch, G. Bishop, et al. "An introduction to the Kalman filter". In: (1995) (cit. on p. 6).

[23] F. LEWIS. "Optimal estimation: With an introduction to stochastic control theory((Book))". In: *New York, Wiley-Interscience, 1986, 386* (1986) (cit. on pp. 6, 17).

[24] S. Thrun, W. Burgard, and D. Fox. *Probabilistic robotics*. MIT Press, 2006 (cit. on pp. 6, 13, 16, 17).

[25] M. I. Ribeiro. "Kalman and extended kalman filters: Concept, derivation and properties". In: *Institute for Systems and Robotics* 43.46 (2004), pp. 3736–3741 (cit. on p. 6).

[26] T. Bailey et al. "Consistency of the EKF-SLAM Algorithm". In: *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2006, pp. 3562–3568. DOI: 10.1109/IROS.2006.281644 (cit. on pp. 6, 7).

[27] E. Wan and R. Van Der Merwe. "The unscented Kalman filter for nonlinear estimation". In: *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*. IEEE. 2000, pp. 153–158. DOI: 10.1109/ASSPCC.2000.882463 (cit. on p. 6).

[28] S. Julier and J. Uhlmann. "A counter example to the theory of simultaneous localization and map building". In: *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*. Vol. 4. 2001, 4238–4243 vol.4. DOI: 10.1109/ROBOT.2001.933280 (cit. on p. 6).

[29] T. Zhang et al. "Convergence and Consistency Analysis for a 3-D Invariant-EKF SLAM". In: *IEEE Robotics and Automation Letters* 2.2 (2017), pp. 733–740. DOI: 10.1109/LRA.2017.2651376 (cit. on p. 6).

[30] S. J. Julier and J. K. Uhlmann. "New extension of the Kalman filter to nonlinear systems". In: *Signal processing, sensor fusion, and target recognition VI*. Vol. 3068. Spie. 1997, pp. 182–193 (cit. on p. 6).

[31] G. P. Huang, A. I. Mourikis, and S. I. Roumeliotis. "On the complexity and consistency of UKF-based SLAM". In: *2009 IEEE International Conference on Robotics and Automation*. 2009, pp. 4401–4408. DOI: 10.1109/ROBOT.2009.5152793 (cit. on p. 6).

[32] N. Metropolis and S. Ulam. "The monte carlo method". In: *Journal of the American statistical association* 44.247 (1949), pp. 335–341 (cit. on p. 6).

[33] S. Thrun. "Particle Filters in Robotics." In: *UAI*. Vol. 2. Citeseer. 2002, pp. 511–518 (cit. on pp. 6, 7).

[34] M. Montemerlo et al. "FastSLAM: A factored solution to the simultaneous localization and mapping problem". In: *Aaai/iaai* 593598 (2002) (cit. on p. 7).

[35] J. B. Rawlings and B. R. Bakshi. "Particle filtering and moving horizon estimation". In: *Computers Chemical Engineering* 30.10 (2006). Papers form Chemical Process Control VII, pp. 1529–1541. ISSN: 0098-1354. DOI: https://doi.org/10.1016/j.compchemeng.2006.05.031. URL: https://www.sciencedirect.com/science/article/pii/S0098135406001566 (cit. on p. 7).

[36] M. Montemerlo et al. "FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges". In: *IJCAI*. Vol. 3. 2003, pp. 1151–1156 (cit. on p. 7).

[37] R. Sim et al. "Vision-based SLAM using the Rao-Blackwellised particle filter". In: *IJCAI Workshop on Reasoning with Uncertainty in Robotics*. Vol. 14. 1. 2005, pp. 9–16 (cit. on p. 7).

[38] J. Z. Sasiadek, A. Monjazeb, and D. Necsulescu. "Navigation of an autonomous mobile robot using EKF-SLAM and FastSLAM". In: *2008 16th Mediterranean Conference on Control and Automation*. 2008, pp. 517–522. DOI: 10.1109/MED.2008.4602213 (cit. on p. 7).

[39] C. V. Rao and J. B. Rawlings. "Constrained process monitoring: Moving-horizon approach". In: *AIChE Journal* 48.1 (2002), pp. 97–109. DOI: https://doi.org/10.1002/aic.690480111. eprint: https://aiche.onlinelibrary.wiley.com/doi/pdf/10.1002/aic.690480111. URL: https://aiche.onlinelibrary.wiley.com/doi/abs/10.1002/aic.690480111 (cit. on p. 7).

[40] D. G. Robertson, J. H. Lee, and J. B. Rawlings. "A moving horizon-based approach for least-squares estimation". In: *AIChE Journal* 42.8 (1996), pp. 2209–2224 (cit. on p. 7).

[41] K. R. Muske and J. B. Rawlings. "Nonlinear moving horizon state estimation". In: *Methods of model based process control*. Springer, 1995, pp. 349–365 (cit. on p. 7).

[42] C. V. Rao. *Moving horizon strategies for the constrained monitoring and control of nonlinear discrete-time systems*. The University of Wisconsin-Madison, 2000 (cit. on p. 7).

[43] C. Rao, J. Rawlings, and D. Mayne. "Constrained state estimation for nonlinear discrete-time systems: stability and moving horizon approximations". In: *IEEE Transactions on Automatic Control* 48.2 (2003), pp. 246–258. DOI: 10.1109/TAC.2002.808470 (cit. on pp. 7, 59).

[44] M. A. Müller. "Nonlinear moving horizon estimation in the presence of bounded disturbances". In: *Automatica* 79 (2017), pp. 306–314. ISSN: 0005-1098. DOI: https://doi.org/10.1016/j.automatica.2017.01.033. URL: https://www.sciencedirect.com/science/article/pii/S0005109817300432 (cit. on p. 7).

[45] T. Kasahara et al. "Comparative Experiments of Moving Horizon Estimation based SLAM in Indoor Environment". In: *2019 12th Asian Control Conference (ASCC)*. IEEE. 2019, pp. 1131–1136 (cit. on pp. 7, 19, 70).

[46] Y. Kishimoto, K. Takaba, and A. Ohashi. "Moving Horizon Multi-Robot SLAM Based on C/GMRES Method". In: *2019 International Conference on Advanced Mechatronic Systems (ICAMechS)*. 2019, pp. 22–27. DOI: 10.1109/ICAMechS.2019.8861681 (cit. on p. 7).

[47] T. Ohtsuka. "A continuation/GMRES method for fast computation of nonlinear receding horizon control". In: *Automatica* 40.4 (2004), pp. 563–574 (cit. on p. 8).

[48] G. Ferrari-Trecate, D. Mignone, and M. Morari. "Moving horizon estimation for hybrid systems". In: *IEEE Transactions on Automatic Control* 47.10 (2002), pp. 1663–1676. DOI: 10.1109/TAC.2002.802772 (cit. on p. 8).

[49] A. Bemporad, D. Mignone, and M. Morari. "Moving horizon estimation for hybrid systems and fault detection". In: *Proceedings of the 1999 American Control Conference (Cat. No. 99CH36251)*. Vol. 4. 1999, 2471–2475 vol.4. DOI: 10.1109/ACC.1999.786492 (cit. on p. 8).

[50] L. Zou et al. "Moving horizon estimation meets multi-sensor information fusion: Development, opportunities and challenges". In: *Information Fusion* 60 (2020), pp. 1–10. ISSN: 1566-2535. DOI: https://doi.org/10.1016/j.inffus.2020.01.009. URL: https://www.sciencedirect.com/science/article/pii/S1566253519310024 (cit. on p. 8).

[51] E. L. Haseltine and J. B. Rawlings. "Critical Evaluation of Extended Kalman Filtering and Moving-Horizon Estimation". In: *Industrial & Engineering Chemistry Research* 44.8 (2005), pp. 2451–2460. DOI: 10.1021/ie034308l. eprint: https://doi.org/10.1021/ie034308l. URL: https://doi.org/10.1021/ie034308l (cit. on p. 8).

[52] M. Farina, G. Ferrari-Trecate, and R. Scattolini. "Distributed Moving Horizon Estimation for Linear Constrained Systems". In: *IEEE Transactions on Automatic Control* 55.11 (2010), pp. 2462–2475. DOI: 10.1109/TAC.2010.2046058 (cit. on p. 8).

[53] G. Grisetti et al. "A tutorial on graph-based SLAM". In: *IEEE Intelligent Transportation Systems Magazine* 2.4 (2010), pp. 31–43 (cit. on p. 8).

[54] M. Labbe and F. Michaud. "Online global loop closure detection for large-scale multi-session graph-based SLAM". In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2014, pp. 2661–2666 (cit. on p. 8).

[55] R. Kümmerle et al. "Large scale graph-based SLAM using aerial images as prior information". In: *Autonomous Robots* 30 (2011), pp. 25–39 (cit. on p. 8).

[56] G. Klein and D. Murray. "Parallel Tracking and Mapping for Small AR Workspaces". In: *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*. 2007, pp. 225–234. DOI: 10.1109/ISMAR.2007.4538852 (cit. on p. 8).

[57] T. Taketomi, H. Uchiyama, and S. Ikeda. "Visual SLAM algorithms: A survey from 2010 to 2016". In: *IPSJ Transactions on Computer Vision and Applications* 9.1 (2017), pp. 1–11 (cit. on p. 9).

[58] A. J. Davison. "Real-time simultaneous localisation and mapping with a single camera". In: *Computer Vision, IEEE International Conference on*. Vol. 3. IEEE Computer Society. 2003, pp. 1403–1403 (cit. on p. 9).

[59] J. Engel, T. Schöps, and D. Cremers. "LSD-SLAM: Large-scale direct monocular SLAM". In: *European conference on computer vision*. Springer. 2014, pp. 834–849 (cit. on p. 9).

[60] J. A. Castellanos et al. "Robocentric map joining: Improving the consistency of EKF-SLAM". In: *Robotics and autonomous systems* 55.1 (2007), pp. 21–29 (cit. on p. 9).

[61] B. J. Guerreiro et al. "Globally asymptotically stable sensor-based simultaneous localization and mapping". In: *IEEE Transactions on Robotics* 29.6 (2013), pp. 1380–1395 (cit. on p. 9).

[62] P. Lourenço et al. "Simultaneous localization and mapping for aerial vehicles: a 3-D sensor-based GAS filter". In: *Autonomous Robots* 40.5 (2016), pp. 881–902 (cit. on pp. 9, 51, 57).

[63] T. A. Johansen. "Introduction to nonlinear model predictive control and moving horizon estimation". In: *Selected topics on constrained and nonlinear control* 1 (2011), pp. 1–53 (cit. on pp. 9, 10).

[64] A. N. Tikhonov et al. *Solutions of ill-posed problems*. Vh Winston, 1977 (cit. on p. 9).

[65] V. M. Zavala, C. D. Laird, and L. T. Biegler. "A fast moving horizon estimation algorithm based on nonlinear programming sensitivity". In: *Journal of Process Control* 18.9 (2008). Selected Papers From Two Joint Conferences: 8th International Symposium on Dynamics and Control of Process Systems and the 10th Conference Applications in Biotechnology, pp. 876–884. ISSN: 0959-1524. DOI: https://doi.org/10.1016/j.jprocont.2008.06.003. URL: https://www.sciencedirect.com/science/article/pii/S0959152408001091 (cit. on p. 9).

[66] J. Nocedal and S. J. Wright. *Numerical optimization*. Springer, 1999 (cit. on p. 9).

[67] P. T. Boggs and J. W. Tolle. "Sequential quadratic programming". In: *Acta numerica* 4 (1995), pp. 1–51 (cit. on p. 9).

[68] P. E. Gill and E. Wong. "Sequential quadratic programming methods". In: *Mixed integer nonlinear programming*. Springer, 2011, pp. 147–224 (cit. on p. 9).

[69] R. Fletcher. "The Sequential Quadratic Programming Method". In: *Nonlinear Optimization: Lectures given at the C.I.M.E. Summer School held in Cetraro, Italy, July 1-7, 2007*. Ed. by G. Di Pillo and F. Schoen. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 165–214. ISBN: 978-3-642-11339-0. DOI: 10.1007/978-3-642-11339-0_3. URL: https://doi.org/10.1007/978-3-642-11339-0_3 (cit. on p. 9).

[70] P. T. Boggs and J. W. Tolle. "Sequential quadratic programming for large-scale nonlinear optimization". In: *Journal of computational and applied mathematics* 124.1-2 (2000), pp. 123–137 (cit. on p. 9).

[71] E. Wong. *Active-set methods for quadratic programming*. University of California, San Diego, 2011 (cit. on p. 9).

[72] K. G. Murty and F.-T. Yu. *Linear complementarity, linear and nonlinear programming*. Vol. 3. Citeseer, 1988 (cit. on p. 9).

[73] F. A. Potra and S. J. Wright. "Interior-point methods". In: *Journal of Computational and Applied Mathematics* 124.1 (2000). Numerical Analysis 2000. Vol. IV: Optimization and Nonlinear Equations, pp. 281–302. ISSN: 0377-0427. DOI: https://doi.org/10.1016/S0377-0427(00)00433-7. URL: https://www.sciencedirect.com/science/article/pii/S0377042700004337 (cit. on p. 10).

[74] R. H. Byrd, M. E. Hribar, and J. Nocedal. "An Interior Point Algorithm for Large-Scale Nonlinear Programming". In: *SIAM Journal on Optimization* 9.4 (1999), pp. 877–900. DOI: 10.1137/S1052623497325107. eprint: https://doi.org/10.1137/S1052623497325107. URL: https://doi.org/10.1137/S1052623497325107 (cit. on p. 10).

[75] J. Gallier and J. Gallier. "The Quaternions and the Spaces $\mathbb{S}^3$, SU (2), SO (3), and $\mathbb{RP}^3$". In: *Geometric Methods and Applications: For Computer Science and Engineering* (2001), pp. 248–266 (cit. on p. 12).

[76] M. Dissanayake et al. "A solution to the simultaneous localization and map building (SLAM) problem". In: *IEEE Transactions on Robotics and Automation* 17.3 (2001), pp. 229–241. DOI: 10.1109/70.938381 (cit. on p. 13).

[77] A. R. Khairuddin, M. S. Talib, and H. Haron. "Review on simultaneous localization and mapping (SLAM)". In: *2015 IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*. 2015, pp. 85–90. DOI: 10.1109/ICCSCE.2015.7482163 (cit. on pp. 16, 17).

[78] G. Dissanayake et al. "A review of recent developments in simultaneous localization and mapping". In: *2011 6th International Conference on Industrial and Information Systems*. IEEE. 2011, pp. 477–482 (cit. on pp. 16, 17).

[79] J. B. Jørgensen. "Moving horizon estimation and control". In: (2004) (cit. on p. 19).

[80] H. Cheng and K. Gupta. "An historical note on finite rotations". In: (1989) (cit. on p. 20).

[81] G. G. Slabaugh. "Computing Euler angles from a rotation matrix". In: *Retrieved on August* 6.2000 (1999), pp. 39–63 (cit. on p. 20).

[82] E. W. Weisstein. "Euler angles". In: *https://mathworld. wolfram. com/* (2009) (cit. on p. 20).

[83] B. A. Rosenfeld. *A history of non-Euclidean geometry: Evolution of the concept of a geometric space*. Vol. 12. Springer Science & Business Media, 2012 (cit. on p. 21).

[84] E. Fresk and G. Nikolakopoulos. "Full quaternion based attitude control for a quadrotor". In: *2013 European control conference (ECC)*. IEEE. 2013, pp. 3864–3869 (cit. on pp. 22, 23, 84).

[85] J. S. Dai. "Euler–Rodrigues formula variations, quaternion conjugation and intrinsic connections". In: *Mechanism and Machine Theory* 92 (2015), pp. 144–152. ISSN: 0094-114X. DOI: https://doi.org/10.1016/j.mechmachtheory.2015.03.004. URL: https://www.sciencedirect.com/science/article/pii/S0094114X15000415 (cit. on p. 22).

[86] F. L. Markley and J. L. Crassidis. *Fundamentals of spacecraft attitude determination and control*. Vol. 1286. Springer, 2014 (cit. on p. 23).

[87] S. W. Shepperd. "Quaternion from rotation matrix". In: *Journal of guidance and control* 1.3 (1978), pp. 223–224 (cit. on p. 23).

[88] P. C. Hughes. *Spacecraft attitude dynamics*. Courier Corporation, 2012 (cit. on p. 23).

[89] S. Sarabandi and F. Thomas. "Accurate computation of quaternions from rotation matrices". In: *International Symposium on Advances in Robot Kinematics*. Springer. 2018, pp. 39–46 (cit. on p. 23).

[90] J. Diebel et al. "Representing attitude: Euler angles, unit quaternions, and rotation vectors". In: *Matrix* 58.15-16 (2006), pp. 1–35 (cit. on p. 23).

[91] P. Singla, D. Mortari, and J. L. Junkins. "How to avoid singularity for Euler angle set". In: *Proceedings of the AAS Space Flight Mechanics Conference, Hawaii*. 2004 (cit. on p. 23).

[92] C. W. Kang and C. G. Park. "Euler angle based attitude estimation avoiding the singularity problem". In: *IFAC Proceedings Volumes* 44.1 (2011), pp. 2096–2102 (cit. on p. 23).

[93] B. Biswas et al. "A discussion on Euler method: A review". In: *Electronic Journal of Mathematical Analysis and Applications* 1.2 (2013), pp. 294–317 (cit. on p. 27).

[94] *Yalmip*. https://yalmip.github.io/. Accessed: 2023-02-12 (cit. on p. 48).

[95] S. Zhao. "Time Derivative of Rotation Matrices: A Tutorial". In: *CoRR* abs/1609.06088 (2016). arXiv: 1609.06088. URL: http://arxiv.org/abs/1609.06088 (cit. on p. 51).

[96] B. J. Guerreiro et al. "Sensor-based simultaneous localization and mapping — Part II: Online inertial map and trajectory estimation". In: *2012 American Control Conference (ACC)*. 2012, pp. 6334–6339. DOI: 10.1109/ACC.2012.6315296 (cit. on pp. 55, 57).

[97] J. Solà. "Quaternion kinematics for the error-state Kalman filter". In: (2017-10) (cit. on p. 83).

[98] E. A. Coutsias and L. Romero. "The quaternions with an application to rigid body dynamics". In: (2004) (cit. on pp. 83, 84).

[99] P. Betsch and R. Siebert. "Rigid body dynamics in terms of quaternions: Hamiltonian formulation and conserving numerical integration". In: *International journal for numerical methods in engineering* 79.4 (2009), pp. 444–473 (cit. on p. 83).

[100] R. Mahony, T. Hamel, and J.-M. Pflimlin. "Complementary filter design on the special orthogonal group SO(3)". In: *Proceedings of the 44th IEEE Conference on Decision and Control*. 2005, pp. 1477–1484. DOI: 10.1109/CDC.2005.1582367 (cit. on p. 83).

I

Annex 1 Use of quaternions on rigid body dynamics

In this annex, the proof of the quaternion variation in terms of angular velocity is addressed based on the notions presented in [97][98][99] [100].

The quaternion, first introduced by William Rowan Hamilton, is a system that extends the complex numbers. This relation can be particularly explained using the Cayley-Dickson construction: given two complex numbers $A = a + bi$ and $C = c + di$, then $Q = A + Cj$ is a number in the quaternion space $\mathbb{H}$. Defining $k := ij$, Q is equal to

$$Q = a + bi + cj + dk \in \mathbb{H} \tag{I.1}$$

in which $\{i, j, k\}$ are the quaternion units and $\{a, b, c, d\} \in \mathbb{R}\}$

Other representation of the quaternion, is the sum of a scalar and a vector in 3D space, in the form of a vector with dimension equal to four:

$$Q := \begin{bmatrix} q_\alpha \\ \mathbf{q}_v \end{bmatrix} \tag{I.2}$$

I.0.1 Proof: Variation of a quaternion

Considering ${}^{W}\mathbf{x}'$ as the coordinate vector of a vehicle on the world-fixed frame and ${}^{B}\mathbf{x}'$ the coordinate vector of the vehicle on his own body's frame, we get that

$${}^{W}\mathbf{x}' = \mathbf{Q}(t)\,{}^{B}\mathbf{x}' \tag{I.3}$$

in which $\mathbf{Q}(t)$ is an orthogonal transformation. The time evolution of the equation, can be given by

$$\frac{d\,{}^{W}\mathbf{x}'}{dt} = \frac{d\mathbf{Q}}{dt}\,{}^{B}\mathbf{x}' + \mathbf{Q}\frac{d\,{}^{B}\mathbf{x}'}{dt} \tag{I.4}$$

Knowing that

$${}^{B}\mathbf{x}' = \mathbf{Q}^T\,{}^{W}\mathbf{x}' \tag{I.5}$$

equation (I.4) can be rewritten as

$$\frac{d\,{}^{W}\mathbf{x}'}{dt} = \frac{d\mathbf{Q}}{dt}\mathbf{Q}^{T}\,{}^{W}\mathbf{x}' + \mathbf{Q}\frac{d\,{}^{B}\mathbf{x}'}{dt} \tag{I.6}$$

Besides that, the coordinate vector of the vehicle on his own body frame does not change over time

$$\frac{d\,{}^{B}\mathbf{x}'}{dt} = 0, \tag{I.7}$$

as such equation (I.6) takes the following representation

$$\frac{d\,{}^{W}\mathbf{x}'}{dt} = \frac{d\mathbf{Q}}{dt}\mathbf{Q}^{T}\,{}^{W}\mathbf{x}' \tag{I.8}$$

Since $\mathbf{Q}(t)$ is orthogonal, $\mathbf{Q}\mathbf{Q}^{T} = 1$, and as such, $\dot{\mathbf{Q}}\mathbf{Q}^{T}$ is a skew-symmetric matrix. Defining

$$\dot{\mathbf{Q}}\mathbf{Q}^{T} = \mathbf{S}[\mathbf{w}] \tag{I.9}$$

we get that

$$\frac{d\,{}^{W}\mathbf{x}'}{dt} = \mathbf{w} \times {}^{W}\mathbf{x}' \tag{I.10}$$

The vector $\mathbf{w}$ is the instantaneous angular velocity in the world frame.

Consider now the relation between the coordinates of a vector in body and space coordinates, given by the unit quaternion $\mathbf{q}(t)$ [84] as in equation (3.41)

$${}^{W}\mathbf{x} = \mathbf{q}.\,{}^{B}\mathbf{x}.\mathbf{q}^{*} \tag{I.11}$$

in which ${}^{W}\mathbf{x} = [0, {}^{W}\mathbf{x}']$ and ${}^{W}\mathbf{x} = [0, {}^{B}\mathbf{x}']$ are pure quaternions. Given the assumption at (I.7), the evolution of the equation over time can be depicted as [98]

$$\frac{d\,{}^{W}\mathbf{x}}{dt} = \frac{d\mathbf{q}}{dt}.\,{}^{B}\mathbf{x}.\mathbf{q}^{*} + \mathbf{q}.\,{}^{B}\mathbf{x}.\frac{d\mathbf{q}}{dt} = (\frac{d\mathbf{q}}{dt}.\mathbf{q}^{*}).\,{}^{W}\mathbf{x} + {}^{B}\mathbf{x}.(\mathbf{q}.\frac{d\mathbf{q}^{*}}{dt}) \tag{I.12}$$

Having in mind that $(\frac{d\mathbf{q}}{dt}.\mathbf{q}^{*})$ is a pure quaternion, it will be defined that $\mathbf{v} = (\frac{d\mathbf{q}}{dt}.\mathbf{q}^{*}) = [0, \mathbf{v}']$. Since the conjugate of pure quaternions its equal to their negation, we have

$$\frac{d\,{}^{W}\mathbf{x}}{dt} = 2\mathbf{v} \times {}^{W}\mathbf{x} \tag{I.13}$$

or

$$\frac{d\,{}^{W}\mathbf{x}'}{dt} = 2\mathbf{v}' \times {}^{W}\mathbf{x}' \tag{I.14}$$

Having in consideration the result of (I.10), it is possible to define

$$\mathbf{w} = 2\mathbf{v} = [0, 2\mathbf{v}'] \tag{I.15}$$

which results in defining the variation of the quaternion over time in regards to the angular velocity

$$\frac{d\mathbf{q}}{dt} = \frac{1}{2}\mathbf{w}\mathbf{q} \tag{I.16}$$

Regarding the angular velocity in the body frame $\mathbf{\Omega}$, we have that $\mathbf{\Omega} = \mathbf{q}^{*}\mathbf{w}\mathbf{q}$ or that $\mathbf{w} = \mathbf{q}\mathbf{\Omega}\mathbf{q}^{*}$, which results in

$$\frac{d\mathbf{q}}{dt} = \frac{1}{2}\mathbf{q}\mathbf{\Omega} \tag{I.17}$$