UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

telecos BCN

IMPROVEMENT OF A POSITIONING SYSTEM FOR ASSISTED AND AUTONOMOUS DRIVING

A Degree Thesis

Submitted to the Faculty of the

Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona

Universitat Politècnica de Catalunya

by

Asier Pabolleta Martorell

In partial fulfilment

of the requirements for the degree in

***Telecommunications Technologies and Services* ENGINEERING**

Advisor: Anna Calveras Auge

Barcelona, June 2021

Abstract

In the recent years, the automotive industry has oriented its research and development efforts towards autonomous and assisted driving. These are critical systems as they must guarantee their robustness and reliability for human life, and therefore they have complex designs and involve the most innovative technologies. Location and positioning system is one of its more challenging mechanisms that requires high availability and precision. For this problem there are many technological approaches, but most common ones are GNSS-like and image recognition solutions. They both have accurate and reliable performance under favourable conditions. However, due to the variability of the environment they also have some points of failure that can result in an accident. In certain adverse conditions, to obtain a precise and reliable localization, further information is needed. Here is where Ultra Wide Band technology (UWB) with support of additional sensing mechanisms such as Inertial Navigation System (INS), can provide additional sources of information. This data fusion is done by means of Kalman Filtering, more concretely the Extended Kalman Filter algorithm, which applies for non-linear systems like the one under study. In this document, the whole positioning system's performance will be analysed and evaluated with some simulations and real scenario tests, comparing the main approaches of the Kalman Filter (linear and non-linear) and obtaining an optimized solution for the positioning mechanism. Also, some conclusions and future work tips will be provided in order to contribute the knowledge in similar areas.

Resum

En els darrers anys, la indústria de l’automòbil ha orientat els seus esforços de recerca i desenvolupament cap a la conducció autònoma i assistida. Es tracta de sistemes crítics, ja que han de garantir la seva robustesa i fiabilitat per a la vida humana i, per tant, tenen dissenys complexos i impliquen les tecnologies més innovadores. El sistema de localització i posicionament és un dels seus mecanismes més difícils que requereix una alta disponibilitat i precisió. Per a aquest problema hi ha molts enfocaments tecnològics, però els més comuns són solucions similars al GNSS i de reconeixement d’imatges. Tots dos tenen un rendiment precís i fiable en condicions favorables. Tot i això, a causa de la variabilitat de l’entorn, també presenten alguns punts de fracàs que poden provocar un accident. En certes condicions adverses, per obtenir una localització precisa i fiable, es necessita més informació. Aquí és on la tecnologia Ultra Wide Band (UWB) amb suport de mecanismes de detecció addicionals com el sistema de navegació inercial (INS), pot proporcionar fonts d'informació addicionals. Aquesta fusió de dades es fa mitjançant el filtratge de Kalman, més concretament l'algorisme del filtre Kalman estès, que s'aplica a sistemes no lineals com el que s'està estudiant. En aquest document, s’analitzarà i avaluarà el rendiment de tot el sistema de posicionament amb algunes simulacions i proves d’escenaris reals, comparant els enfocaments principals del filtre Kalman (lineal i no lineal) i obtenint una solució optimitzada per al mecanisme de posicionament. A més, es proporcionaran algunes conclusions i futurs consells de treball per tal de contribuir al coneixement en àrees similars.

Resumen

En los últimos años, la industria del automóvil ha orientado sus esfuerzos de investigación y desarrollo hacia la conducción autónoma y asistida. Estos son sistemas críticos ya que deben garantizar su robustez y fiabilidad para la vida humana, por lo que adquieren diseños complejos e involucran las tecnologías más innovadoras. El sistema de ubicación y posicionamiento es uno de sus mecanismos más desafiantes que requiere una alta disponibilidad y precisión. Para este problema existen muchos enfoques tecnológicos, pero los más comunes son las soluciones de reconocimiento de imágenes y sistemas GNSS. Ambos tienen un rendimiento preciso y confiable en condiciones favorables. Sin embargo, debido a la variabilidad del entorno también tienen algunos puntos de fallo que pueden resultar en un accidente. En determinadas condiciones adversas, para obtener una localización precisa y fiable, se necesita más información. Aquí es donde la tecnología de banda ultra ancha (UWB) con soporte de mecanismos de detección adicionales como el sistema de navegación inercial (INS), puede proporcionar fuentes adicionales de información. Esta fusión de datos se realiza mediante el filtrado de Kalman, más concretamente el algoritmo de filtro de Kalman extendido, que se aplica a sistemas no lineales como el que se está estudiando. En este documento se analizará y evaluará el desempeño de todo el sistema de posicionamiento con algunas simulaciones y pruebas de escenarios reales, comparando los principales enfoques del Filtro de Kalman (lineal y no lineal) y obteniendo una solución optimizada para el mecanismo de posicionamiento. Asimismo, se brindarán algunas conclusiones y consejos de trabajo futuro con el fin de aportar el conocimiento en áreas similares.

This thesis is entirely dedicated to family and friends who have somehow taken part in its development.

Acknowledgements

This project would not have been possible without the support of the supervisor Anna Calveras Auge, who has made available to the author of this thesis all tools and documentation needed. Also an important acknowledgement for the crew from i2CAT. They all have supported and followed this work throughout last months, and have shown a good reception of it. Special thanks to Julia Igual Nevot, whose Master’s thesis in Kalman Filtering has been an important starting point for this study.

Finally, I would also like to thank the scientific and open source community for sharing knowledge and releasing important pieces of work without which the present thesis would not have reached such depth.

Barcelona, June 2021

Revision history and approval record

Revision	Date	Purpose
0	05/06/2021	Document creation
1	10/06/2021	Document revision
2	18/06/2021	Document revision

DOCUMENT DISTRIBUTION LIST

Name	e-mail
Asier Pabolleta Martorell	asier.pabolleta@gmail.com
Anna Calveras Auge	anna.calveras@upc.edu

Written by:		Reviewed and approved by:	
Date	21/06/2021	Date	21/06/2021
Name	Asier Pabolleta Martorell	Name	Anna Calveras Auge
Position	Project Author	Position	Project Supervisor

Table of contents

Abstract....2

Resum....3

Resumen....4

Acknowledgements....6

Revision history and approval record....7

Table of contents....8

List of Figures....11

List of Tables:....12

1 Introduction....13

1.1 Motivation....13

1.2 Statement of purpose....13

1.3 Requirements and specifications....14

1.4 Methods and procedures....14

1.5 Work plan....15

1.6 Related concepts and technologies....15

1.6.1 Positioning system....16

1.6.2 Kinematic equations....16

1.6.3 Ultra Wide Band....16

1.6.4 Inertial Navigation System....18

1.6.5 Sensor fusion....19

1.6.5.1 Kalman Filter....19

1.6.5.2 Extended Kalman Filter....19

2 State of the art of the technology used or applied in this thesis:....21

2.1 Assisted and autonomous driving....21

2.2 Positioning systems on assisted and autonomous driving.....................................21

2.3 Sensor fusion...23

3 Methodology:...24

3.1 Filter design..24

3.1.1 Kalman Filter..25

3.1.1.1 Constant Velocity Model (CV)..26

3.1.1.2 Constant Velocity Model Constant Acceleration Controlled (CVCA)................28

3.1.1.3 Constant Acceleration Model (CA)..28

3.1.2 Extended Kalman Filter...30

3.1.2.1 Constant Velocity Model (CV)..31

3.1.2.2 Constant Velocity Model Constant Acceleration Controlled (CVCA)................32

3.1.2.3 Constant Acceleration Model (CA)..32

3.1.3 Algorithm code implementation..32

3.1.4 Filter input..33

3.2 Simulation..34

3.2.1 Setup and initialization...34

3.2.2 Measurements input...34

3.2.3 Study cases..35

3.2.3.1 Ideal case..35

3.2.3.2 Bad initialization..35

3.2.3.3 Optimal design...35

3.2.4 Simulation constraints..35

3.3 Real scenario test...36

3.3.1 Setup and initialization...36

3.3.2 Measurements input...37

3.3.3 Study cases..37

3.3.4 Real scenario test constraints....37

3.4 Output evaluation....37

4 Results....39

4.1 Simulation results....39

4.1.1 Ideal case....39

4.1.2 Bad initialization....40

4.1.3 Optimal design....42

4.1.4 Simulation results summary....43

4.2 Real scenario test results....43

4.2.1 Straight-way scenario test....43

4.2.2 90-degree-bend scenario test....44

4.2.3 Real tests summary....46

5 Budget....47

6 Environment Impact....48

7 Conclusions and future development:....49

Bibliography:....51

Appendices:....53

Glossary....78

List of Figures

1.1 Cartesian coordinates (a) and 3D point graphical representation (b) 16

1.2 UWB pulse in the time domain (a) and frequency domain (b) 17

1.3 UWB infrastructure scheme (a) and TWR message flow (b) 17

1.4 Euler angles 19

3.1 Univariate Gaussian distribution (a) and multivariate Gaussian distribution (b) 33

3.2 Real test scenario image 36

3.3 Straight-way scenario (a) and 90-degree-bend scenario (b) setups 36

4.1 Prediction in the XY plane for the KF CV model 40

4.2 EKF CA output plot (a) and error bar plot (b) for bad initialization 41

4.3 KF CVCA projections (a) and error bar plot (b) 42

4.4 KF CVCA projections (a) and error bar plot (b) 44

4.5 Track for KF CVCA (a) and EKF CV CVCA (b) 45

4.6 Projections (a) and error bar plot (b) for EKF CA 46

List of Tables:

3.1 Initial simulation values 34

4.1 Error statistics summary for ideal case simulation 39

4.2 Error statistics summary for bad initialization case simulation 40

4.3 Error statistics summary for optimal design case simulation 42

4.4 Error statistics summary for straight-way scenario 44

4.5 Error statistics summary for 90-degree-bend scenario 45

5.1 Budget summary table 47

1 Introduction

1.1 Motivation

Technological advances in recent years have led the automotive industry to innovate in the way of driving, which has resulted in the design, development and implementation of assisted and autonomous driving systems. These systems require high availability and precision positioning mechanisms, and so this project is focused on the improvement of the positioning mechanism of a vehicle in a rural road, where GNSS-like and image recognition positioning systems become unavailable or unreliable. Environmental conditions, as well as road characteristics, can affect system's performance in various ways, and the system design and implementation must behave appropriately to ensure its reliability. The chosen technology to improve the localization is the emerging Ultra Wide Band (UWB) technology with support of the Inertial Navigation System (INS). This technological approach has been widely used for indoor positioning (IPS). However, in this thesis the viability of UWB technology in an Outdoor Positioning System (OPS) will be evaluated.

1.2 Statement of purpose

As stated before, assisted and autonomous driving ecosystems are critical systems as they must guarantee their robustness and reliability for human life. For this reason location and positioning mechanisms become such important and crucial. Temporary unavailability or low accuracy in measures can lead into an accident. Thus, the main purpose of this project is to implement and design a reliable Outdoor Positioning System based on a sensor fusion solution implemented by means of the Extended Kalman Filter (EKF), in order to obtain an accurate positioning of a target. The integrated hybrid system will combine measures from Ultra Wide Band sensors and Inertial Measurement Units (IMU), such as accelerometers and gyroscopes, and will compute the optimal three-dimensional positioning (3D) using EKF techniques. These all will be consciously placed on the moving object in accordance with the designed system, allowing most accurate and optimal performance for such purpose.

Ultra Wide Band technology will provide absolute positioning as it will compute distance measures between anchors and a tag attached to the moving object. The anchors will be placed on both sides of the road in well known positions. When no UWB measure is available or between UWB range computation periods, INS will provide additional incremental data, such as acceleration, orientation and angular velocity values, in order to perform relative positioning updates.

Combining both sources of information in Kalman Filtering, the system must provide high precision positioning estimates for each time step. The positioning values obtained as the result must allow locating the target into the global frame with an error less than 30cm along the trajectory when required by the driving system, so short time periods between positioning updates must also be accomplished. Additionally, three-dimensional positioning (3D) will be computed as opposed to 2D positioning performed in many similar studies. In this way, the system can get more information about the real state of the target.

1.3 Requirements and specifications

A positioning system for any scope of use is supposed to have high accuracy and reliability requirements, even more when applied in autonomous driving. Therefore, the positioning system must have the ability to perform accurate positioning on outdoor environments under diverse conditions. The environment variability modelling must comprise many scenarios, such as Line of Sight (LOS), Non Line of Sight (NLOS), beam reflection and irregular path. Additionally, the positioning mechanism must be able to compute positioning on real time, so the algorithm performance and computational cost must be taken in care to design an implementable solution. Other designing aspects to consider are the whole infrastructure scalability and associated cost, but these are outside the scope of this project.

For the current positioning system under study, an UWB infrastructure must be deployed. For such purpose, the real scenario tests have been performed in a controlled environment, where UWB anchors have been consciously placed on well known locations. Additionally, an UWB tag and an IMU with an accelerometer and a gyroscope have been attached to the target. For the real case study, different data acquisition frequencies are obtained from UWB and INS, so the filtering must behave accordingly.

1.4 Methods and procedures

This project is the continuation of a previous one developed by Julia Igual Nevot as Master's Final Thesis, in which generic Kalman Filter was implemented for sensor data fusion from UWB/INS hybrid system in order to improve the location of a vehicle in rural roads. As a starting point to the current study, and in order to evaluate and compare both approaches many designing parameters have been reused. For better evaluation of the system design, linear and non-linear algorithms will be tested and compared each other, the standard Kalman Filter and the Extended Kalman Filter, respectively.

Later in this section, a brief introductory explanation on the mathematical background for the Kalman Filtering and different kinematic model designs will be provided in order to acquire a basic knowledge for later implementation of the algorithms. Additionally, in the State of the Art section in this document some external documentation and references will be given for further information. In this theoretical explanation and analytical approach for the kinematic problem under study, the non-linearity will be solved in two different ways. Each one of them will have its advantages and drawbacks that will be analysed and compared.

For the linear approach of the Kalman Filter, the results from the trilateration, using distance measures calculated from UWB ranging, and additional acceleration values from INS will be introduced into the computation of the algorithm to get optimal positioning estimates.

Alternatively, for the non-linear approach known as Extended Kalman Filter, distance measures from UWB ranging will be introduced directly into the filter, with no need of previous processing. Also, additional acceleration measurements from INS will be added.

The evaluation of both implementations will be done in two ways. First, a simulation will be performed, in which virtually generated data with added Gaussian noise will be introduced to the computation of the algorithms. Second, measures from a real scenario test will be parsed correctly and introduced into both algorithms. In this way, the performance of both approaches will be tested and compared each other in different scenarios. For each case of study descriptive statistics and visual plots will be generated to help the analysing and evaluation process.

This comparison between filter outputs is made by means of statistical parameters, such as mean, standard deviation, variance and Root Mean Square Error (RMSE). When real positioning values are known, these error statistics will be calculated between the generated output estimates and the real path. In both cases a graphical representation of the data will be done.

In order to ease the execution of the algorithm and its evaluation, a Jupyter Notebook has been developed, where some parameters can be configured interactively. All the code is written in Python 3.8 and many modules and third-party packages have been used for various purposes. All this will be detailed in the Methodology section in this document.

Finally, at the end of the document some conclusions and future work tips will be provided in order to help research in similar areas.

1.5 Work plan

The initial working plan developed in the Project Proposal and Working Plan document comprised multiple stages for the project development. First, some research about currently used technologies and methodologies for positioning systems and sensor fusion solutions was done. The State of the Art section is the result for that research. Second, the preliminary system design and analytical approach was developed. Next, in the implementation step all the code, simulations and evaluation tools were implemented in order to evaluate the system’s performance in the following stages. Finally, according to the results obtained, all the documentation and the current document have been written and released in order to share the results. The Working Plan task breakdown and Work Package relationship designed in the initial planning stage are detailed in the appendix A1. For the timing plan a Gantt diagram is provided in the appendix A2.

At the System Design stage in WP3 some difficulties related to the system modelling and analytical approach of the problem delayed the implementation stage due to their complexity. The problem under study and the algorithm itself required deep knowledge and extensive mathematical background that had to be solved correctly. Additionally, at the System Test and Validation in WP5 some technical difficulties in the scenario setup for the tests have led into a lack of precision in the measurements. However, all this difficulties have been solved accordingly, and the results and conclusions are exposed later in this document.

1.6 Related concepts and technologies

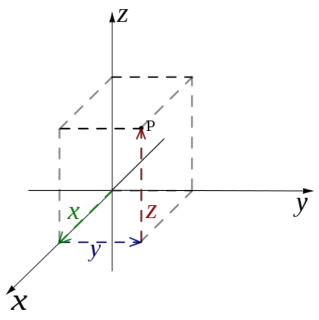
In the current section the main concepts for this project will be introduced in a general and theoretical way.

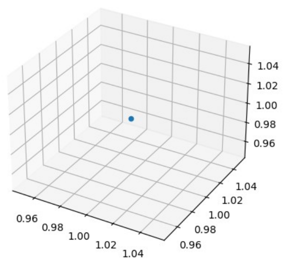
1.6.1 Positioning system

Positioning refers to locating a target on the space at known coordinates frame. Many positioning systems are defined, and the position on each is determined with different coordinates systems. For this study case, a local positioning system is defined, and positioning is expressed in Cartesian coordinates.

$$P=(x,y,z) \tag{1.1}$$

Figure 1.1: Cartesian coordinates (a) and 3D point graphical representation (b)





(a) (b)

For correct positioning of a target on the global frame an Absolute Positioning system is required at least. In this practical case, Ultra Wide Band technology is used as an Absolute Positioning system with support of the Inertial Navigation System that will serve as a Relative Positioning system. In the State of the Art section a deeper explanation on positioning systems is done.

1.6.2 Kinematic equations

Kinematic describes the motion of bodies in the space. In a kinematic problem with known initial conditions and system's geometry, the position, velocity and/or acceleration values can be calculated at any point of the trajectory. These equations will be useful when defining the Process Model for the filter. Next, are the one-dimensional expressions from kinematic equations that will be used in this project. These equations only work under constant velocity/acceleration conditions.

$$x=x_0+v\cdot t+\frac{1}{2}a\cdot t^2 \tag{1.2}$$

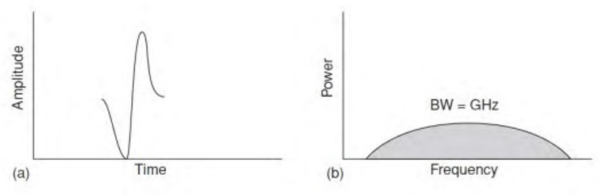
$$v=v_0+a\cdot t \tag{1.3}$$

1.6.3 Ultra Wide Band

For this brief introduction on UWB technology based positioning system, the explanations from Julia Igual Nevot's Master's Final Thesis will be referenced for more detailed explanation [1].

Ultra Wide Band technology is a wireless radio technology emerging in recent years, characterized by the use of a wider bandwidth in comparison with others (fractional bandwidth higher than 0.2 or total bandwidth larger than 500MHz). This feature allows short time duration pulses generation (picoseconds to nanoseconds) with a very low duty cycle, which translates into high precision in the time domain and a low transmission power requirement.

Figure 1.2: UWB pulse in the time domain (a) and frequency domain (b)



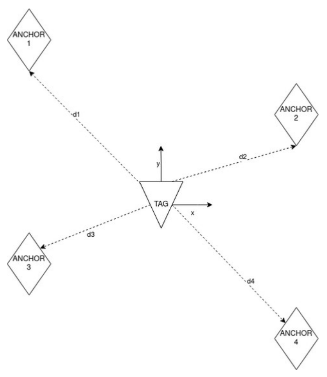
This can be used to compute high accuracy distance measurements between two UWB devices. As a result, a high accuracy absolute positioning system can be implemented. One approach for this is to use these range measurements in trilateration [2]. For a given number of i space dimensions the next formula provides the required number of range measurements to optimally compute positioning by trilateration method.

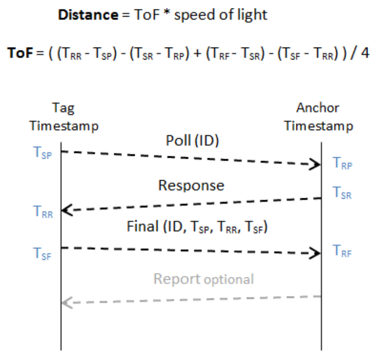
$$n_d = i + 1 \tag{1.4}$$

According to the specifications for this project, in 3D positioning 4 range measurements are required at least.

An UWB infrastructure is composed of two types of nodes: tags and anchors. Anchors are fixed nodes located in well known positions, while tag is referred as the target device attached to the moving object. Anchors are the reference nodes used to locate the tag into the global reference frame. Different range detection and localization methods between nodes have been designed. However, in this study the Two-Way-Ranging (TWR) approach will be used as it better fits the system requirements. The next figure shows an example of UWB infrastructure as well as the message flow in TWR.

Figure 1.3: UWB infrastructure scheme (a) and TWR message flow (b)





Distance = ToF * speed of light
ToF = ($(T_{RR} - T_{SP}) - (T_{SR} - T_{RP}) + (T_{RF} - T_{SR}) - (T_{SF} - T_{RR})$) / 4
Tag Timestamp
Anchor Timestamp
Poll (ID)
T_{SP}
T_{RP}
Response
T_{SR}
T_{RR}
Final (ID, T_{SP}, T_{RR}, T_{SF})
T_{SF}
T_{RF}
Report optional

(a) (b)

Each UWB range computation returns a distance value between the anchor and the tag. The next expression shows the distance measurement formula.

$$d=\sqrt{(x-a_x)^2+(y-a_y)^2+(z-a_z)^2} \qquad (1.5)$$

Where (x,y,z) is the position of the tag and (a_x,a_y,a_z) the coordinates of the anchor.

1.6.4 <u>Inertial Navigation System</u>

Inertial Navigation System is a sub-type of Dead Reckoning. Dead Reckoning is a process to compute the position of a target relative to an initial point by integrating sensors' measurements over the time. For the INS approach the used sensors are accelerometers and gyroscopes that obtain acceleration and angular rate values respectively. Inertial Measurement Units are attached to the moving object and collect measurements data with integrated accelerometer and gyroscope sensors.

Using acceleration values on each axis, the trajectory of the body in the next steps can be calculated as a kinematic problem. Additionally, integrating angular rate values from gyroscopes, the body frame to global coordinate frame orientation angles can be calculated in order to transform acceleration values in the body frame into acceleration in the global frame [3]. The transformation matrix T transforms the acceleration values on the body frame into the global frame. The next operation shows the body-to-global frame transformation of the acceleration vector.

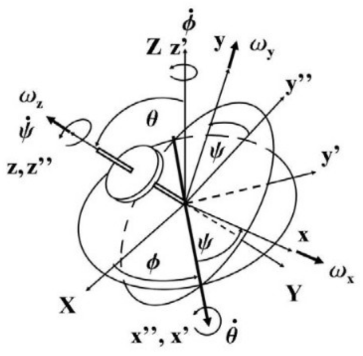
$$\begin{pmatrix} a_{xg} \\ a_{yg} \\ a_{zg} \end{pmatrix} = [T] \begin{pmatrix} a_{xb} \\ a_{yb} \\ a_{zb} \end{pmatrix} \qquad (1.6)$$

Where (a_{xg},a_{yg},a_{zg}) and (a_{xb},a_{yb},a_{zb}) are the acceleration vectors in the global and body frames, respectively. The transformation matrix (T) for the 3D case is shown below.

$$T=\begin{pmatrix} \cos\psi\cos\phi-\cos\theta\sin\phi\sin\psi & \cos\psi\sin\phi+\cos\theta\cos\phi\sin\psi & \sin\theta\sin\psi \\ -\sin\psi\cos\phi-\cos\theta\sin\phi\cos\psi & -\sin\psi\sin\phi+\cos\theta\cos\phi\cos\psi & \sin\theta\cos\psi \\ \sin\phi\sin\theta & -\cos\phi\sin\theta & \cos\theta \end{pmatrix} \qquad (1.7)$$

With the body-to-global frame transformation the global acceleration values can be obtained and thus, corresponding values of position, velocity and acceleration can be calculated in the global frame. This approach is done using Euler Angles. They describe the orientation of a rigid body with respect to a fixed coordinate system.

Figure 1.4: Euler angles



1.6.5 <u>Sensor fusion</u>

The sensor fusion is the process of combining multiple sources of information from sensors to obtain an optimal estimation on the observed variable. The resulting estimate is more accurate and so has less uncertainty than the independent sensor readings, even if the measurements are noisy, because it balances the strengths of different sensors. For sensor fusion both configurations can be implemented: Loosely Coupled (LC) and Tightly Coupled (TC). Kalman Filter is an approach for a sensor fusion problem.

1.6.5.1 <u>Kalman Filter</u>

Kalman Filter is a predict/update algorithm that uses a series of measurements observed over time, containing statistical noise, and produces optimal estimation on unknown variables. It is based on Bayesian probability and it uses Gaussian probability distributions and linear algebra to compute optimal estimates. For a theoretical approach on Kalman Filter design and implementation refer to the book *Kalman and Bayesian Filters in Python* available online [4].

1.6.5.2 <u>Extended Kalman Filter</u>

Kalman Filter applies on linear systems as the calculations are done with matrices that indicate linearity. For non-linear systems many approaches have been developed. Extended Kalman Filter is the main solution for this problem.

The EKF handles non-linearity by linearizing the system at the point of the current estimate, and then the Kalman Filter is used to filter the linearized system. The non-linearity of a problem can be found in both points, in the Process Model used to predict the state or in the measures from the Observation Model.

For the non-linear model the linear expression $\boldsymbol{F x+B u}$ in the predict step is replaced by a non-linear function $\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})$, and the linear expression $\boldsymbol{H}\bar{\boldsymbol{x}}$ in the update step is replaced by non-linear function $\boldsymbol{h}(\bar{\boldsymbol{x}})$. The linearization is done by taking the derivative of those functions, called Jacobian for the matrices, and evaluating at a point, so the corresponding matrices are obtained for the linearized Kalman Filter.

$$\boldsymbol{F} = \left. \frac{\partial \boldsymbol{f}(\boldsymbol{x}_t, \boldsymbol{u}_t)}{\partial \boldsymbol{x}} \right|_{x_t, u_t} \tag{1.8}$$

$$\boldsymbol{H} = \left. \frac{\partial \boldsymbol{h}(\bar{\boldsymbol{x}}_t)}{\partial \bar{\boldsymbol{x}}} \right|_{\bar{x}_t} \tag{1.9}$$

The Kalman Filter algorithm requires some modelling design. The custom implementations of the algorithms for linear and non-linear approaches are introduced in the Methodology section and will be particularized for each study case.

2 State of the art of the technology used or applied in this thesis:

In this section an overview of the most important positioning system approaches for autonomous driving and sensor fusion solutions is done. Also, some references to external resources and projects will be provided in order to explore other systems' advantages and throwbacks. As stated in the previous section the positioning systems is a critical mechanism for assisted and autonomous driving due to its high availability and precision requirements, and so, many research has been done in this topic among the scientific and technological community and companies to get better performance.

2.1 Assisted and autonomous driving

As a brief introduction to the assisted and autonomous driving, one of the most important automotive manufacturer in the world explains the five levels of autonomous driving defined by the experts [5]. These levels range from zero to five and they have been defined according to their relative extent of automation. Level zero means "No automation", where the driver controls every aspect about driving without the support of any driver assistance system, while in level five there is no human interaction. For our study case, as the level increases the system must provide higher reliability, as the human interaction decreases. In [6] a level 3 in autonomous driving system is introduced at a demonstration stage which comprises multiple input data sources, such as GPS, image recognition and light imaging detection and ranging (LiDAR).

In the same way, many well known companies from the automotive sector and other areas are making important research and have been able to develop some approaches. Some of these have even been commercialized. The American autonomous driving technology development company Waymo LLC, subsidiary of Alphabet Inc., the parent company of Google, operates a commercial self-driving taxi service. Similarly, famous Tesla cars also incorporate autonomous driving systems.

The next article remarks the importance of the positioning system as a part of autonomous driving ecosystems [7].

The Universitat Politècnica de Catalunya (UPC) University also has some research and development projects in this area of assisted and autonomous driving, such as the Driverless project, where students from many engineering degrees design, create and validate an autonomous car to compete in Formula Student Driverless category.

2.2 Positioning systems on assisted and autonomous driving

In the Master's Final Thesis from Julia Igual Nevot a detailed breakdown on positioning technologies is done [1], in which main characteristics for each technology are listed. The positioning systems can be divided in two categories: Absolute Positioning and Relative Positioning. Depending on whether the global or reference frame based positioning is provided.

Absolute Positioning systems do not depend on time and can compute global positioning with no need of initial configuration. This computation is done using landmarks placed in well known locations around the target. For this reason, information about the environment must be collected previously.

On the contrary, Relative Positioning systems require initial positioning configuration as they use incremental data collected from the internal state of the target, such as velocity and acceleration, to compute relative positioning. They do not require previous knowledge on the environment and can be used everywhere. However, they don't provide enough information to calculate positioning globally.

In most positioning systems for assisted autonomous driving, Absolute Positioning and Relative Positioning approaches are combined for better performance on positioning estimation. The research made by Julia Igual Nevot in the Master's Final Thesis about positioning systems for autonomous driving concludes that, in general, the precision achieved by the previous technologies in stand-alone mode is not enough for autonomous driving precision requirements, and so a sensor fusion solution is implemented.

Most widely used combination is GPS and INS. For better performance on GPS positioning similar approaches that include error correction algorithms are used instead. Differential GPS (DGPS) and Real Time Kinematic GPS (RTK) are listed as examples. In [8] a Loosely Coupled (LC) GPS/INS integration with Snap To Road (STR) correction system is designed. This approach has being tested in two real scenarios, where 90% of errors were less than 8.3 and 7.82 meters in both trajectories respectively. In [9] GPS Doppler/INS combination is done by Tightly Coupling (TC) approach for urban environments with Non Line of Sight (NLOS) and beam reflection conditions, where important improvement is done compared to the stand-alone GPS positioning. As conclusion for this study the error of heading estimation has been reduced to less than ¼ (from 15.2 meters to 3.4 meters).

Other technological approaches have been developed less frequently for positioning on outdoor environments for autonomous driving. Some of the projects discussed above used LiDAR and/or image recognition technologies or even radio detection and ranging (RADAR).

On the other hand, switching to indoor positioning systems, Ultra Wide Band technology becomes more frequent. In the next example an indoor GPS/UWB sensor fusion solution is implemented [10]. As a conclusion, it is verified that the implemented GPS/UWB solution for indoor positioning could enhance position estimation accuracy as well as being less sensitive to initial position guess. Additionally, it is stated that future work could be done in the area of multipath mitigation. For indoor environments UWB technology is widely used on positioning and many research and test have been done in different scenarios [11][12]. As an example for UWB indoor positioning approach in [12], under LOS conditions the RMSE value is 0.1 meters, while for NLOS it is increased to 0.2 meters.

For deeper knowledge about Ultra Wide Band technical specifications and ranging methods refer to Julia Igual Nevot's Master's Final Thesis [1]. There, a detailed explanation on UWB standard PHY and MAC layers is provided, as well as ranging methods breakdown and trilateration algorithms.

2.3 Sensor fusion

Sensor fusion algorithms allow optimal estimation for unknown variables as they combine multiple sources of information. In many of the projects the sensor fusion solution get better precision than stand-alone configurations. Integration of sensor measurements can be done in two ways: Loosely Coupled (LC) and Tightly Coupled (TC) configurations.

In the LC configuration a previous processing on sensor data is required in order to introduce it to the filter computation. As an example, in some approaches of GPS the position is calculated using trilateration techniques before fusing with INS data [13].

On the other hand, the TC integration fuses sensor measurements directly, without previous processing on the data obtained [14].

The standard Kalman Filter and Extended Kalman Filter can perform sensor fusioning on positioning systems and thus their performance has been tested in various projects. An analytical approach on the filtering problem is done in the free online book *Kalman and Bayesian Filters in Python* [4].

3 Methodology:

The methodology followed for this project is discussed below. First of all, the filter design and implementation will be explained. The main approaches for the positioning problem will be introduced in this section: the standard Kalman Filter and the Extended Kalman Filter. Additionally, some filtering models will be implemented to evaluate and test different designs.

In order to evaluate these designs a simulation is performed first. This simulation generates virtual data according to the input data model and evaluates the filtering performance. The setup, initialization and implementation for this is provided in the next sections. Then, a real case study is done with obtained data from a real scenario test. In both study cases, all needed designs and explanations will be provided, and in the following section the results obtained are discussed in order to evaluate the designed system's performance and make conclusions. All the code implemented for this project is available in the annex.

3.1 Filter design

The current section is probably the most critical step for the system implementation, as a good filter design will expect an optimal performance for the filtering process. At this point two approaches for the positioning problem will be designed in order to evaluate and compare each other: the standard Kalman Filter and Extended Kalman Filter.

To address the problem correctly some definitions will be done on the filtering literature. The *dynamic system* is considered a physical system whose state evolves over time, and this can be modelled by a set of differential equations. For the current study case the location of a moving object is the dynamic system, and this can be characterized by kinematic equations (1.2, 1.3). *State variables* are changing values from a dynamic system that have special relevance for the problem under study. The state variables for the positioning problem on a Euclidean space in Cartesian coordinates for three-dimensional approach are the positioning values on each axis defined by (1.1). However, depending on the filtering model to implement (Constant Velocity Model, Constant Acceleration Model, …) additional state variables can be added, such as velocity and acceleration. All these considerations are included in the *State-Space Model* design. For this, next parameters have to be designed: the State Vector ($\boldsymbol{x}$) and the State Covariance Matrix ($\boldsymbol{P}$).

The next step is designing the *Process Model*. It is a mathematical model which describes the behaviour of the system. The filter uses it to predict the state after a discrete time step (Δt). For this task the kinematic equations provided in (1.2) and (1.3) will be used. Also, the noise associated to the Process Model must be modelled in the *Process Noise Model*, as the real system operation is affected by an unknown number of factors that can't be modelled analytically. These all is considered in the *Process Model* and the parameters to be designed are the next: the State Transition Function ($\boldsymbol{F}$) and the Process Noise Matrix ($\boldsymbol{Q}$).

Optionally, in the predict step in addition to the state propagation from the Process Model, additional knowledge can be introduced as control inputs. The Constant Velocity Model Constant Acceleration Controlled (CVCA) introduces acceleration values as control inputs in a first order kinematic model. This is part of the *Control Input Model* and it is defined by the next parameters: the Control Input Function ($\boldsymbol{B}$) and the Control Input Vector ($\boldsymbol{u}$).

The better the State-Space Model and Process Model model the dynamic system operation, the better prediction will be done on the state for the next time step. For such purpose main filtering models have been designed: Constant Velocity Model (CV), Constant Velocity Model Constant Acceleration Controlled (CVCA) and Constant Acceleration Model (CA). These models comprise the State-Space Model and Process Model design. On the Process Noise model, some approaches have also been discussed from the work in the online book *Kalman and Bayesian Filters in Python*, in the section dedicated to the Kalman Filter Math [4].

Finally, the *Observation Model*, also known as the *Measurements Model*, must be designed. It is a mathematical model that comprises observable variables as observations made to the dynamic system. These observations are obtained as measurements from sensors, whose mean and variance are defined on the Measurements Vector ($\boldsymbol{z}$) and the Measurement Noise Matrix ($\boldsymbol{R}$).

The Kalman Filter computes the update step into the measurements domain, calculating the residual between the measurement and the state prior, and getting a midpoint as the state estimate according to the *Kalman Gain* (K). To be able to calculate the residual, both the state prior and the measurement must be in the same units, so the state must be converted into the measurement domain. The opposite way does not work always because most measurements are not invertible (UWB distance measurements, for example, are not convertible to position values). The Observation Model for the Kalman Filter provides the Measurement Function ($\boldsymbol{H}$) to transform the state into a measurement.

The Kalman Filter uses linear algebra to compute optimal state estimates, so non-linearities on the dynamic system must be solved somehow. In this project these non-linearities will be solved in two ways. The next sections introduce both approaches and provide required mathematical knowledge, as well as concrete designing parameters in order to evaluate both of them.

Then, the job for designing the Kalman filter is to properly design the Space-State Model ($\boldsymbol{x}, \boldsymbol{P}$), the Process Model ($\boldsymbol{F}, \boldsymbol{Q}$), the Measurement or Observation Model ($\boldsymbol{z}, \boldsymbol{R}, \boldsymbol{H}$), and optionally the Control Input Model ($\boldsymbol{B}, \boldsymbol{u}$).

3.1.1 Kalman Filter

The first approach to solve the positioning problem is the linear implementation of the Kalman Filter. This implementation requires the Loosely Coupled (LC) sensor fusion configuration, as a previous manipulation on UWB range measurements is required before the filtering step. In this configuration UWB range measurements are used for trilateration in order to obtain positioning values, so these can be introduced into a linear Kalman Filter. For the trilateration method Least Square Error (LSE) and Least Square

Error with Geometry Constraints (LSE-GC) algorithms are combined to get optimal performance. In the appendix A4 the Kalman Filter algorithm is defined. For the concrete evaluation of the designing parameters defined for the KF detailed in the appendix A3, 3 filtering models will be introduced in the next subsections.

3.1.1.1 <u>Constant Velocity Model (CV)</u>

CV Model is a first order kinematic model that models the dynamic system with the next differential equations on the univariate case.

$$x = x_0 + v \cdot t \qquad (3.1)$$

$$v = v_0 \qquad (3.2)$$

As first order model, it comprises constant velocity ($a=0$) and thus velocity is introduced into the State Vector as a state variable. Extending for the multivariate approach the State Vector and the State Covariance Matrix for three dimensions are shown below.

$$\mathbf{x} = [x \quad v_x \quad y \quad v_y \quad z \quad v_z]^T \qquad (3.3)$$

$$\boldsymbol{P} = \begin{bmatrix} \sigma_x^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_{vx}^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_y^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{vy}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_z^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{vz}^2 \end{bmatrix} \qquad (3.4)$$

The Process Model used to predict the state relies on the kinematic equations (3.1) and (3.2). Then, the State Transition Function is shown next.

$$\boldsymbol{F} = \begin{bmatrix} 1 & \Delta t & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \Delta t & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3.5)$$

For the Process Noise Model two approaches have been implemented: the Continuous Model and the Discrete Model. The Continuous Model assumes the higher order (velocity for current model) is influenced by process noise, it is no longer constant and so, it changes by a continuous time zero-mean white noise (w). As the noise changes continuously, it has to be integrated on each time step for discretization. The next expression shows the calculus for the process noise in the Continuous Noise Model.

$$Q = \int_0^{\Delta t} F(t) Q_c F^T(t)\, dt \tag{3.6}$$

Where Q_c is the continuous noise defined by the spectral density of the white noise (Φ_s). The next matrix shows the continuous noise for a single dimension.

$$Q_c = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \Phi_s \tag{3.7}$$

Finally, the resulting Process Noise Matrix for the discretized Continuous Noise Model is introduced below for three dimensions.

$$Q = \begin{bmatrix} \Delta t^3/3 & \Delta t^2/2 & 0 & 0 & 0 & 0 \\ \Delta t^2/2 & \Delta t & 0 & 0 & 0 & 0 \\ 0 & 0 & \Delta t^3/3 & \Delta t^2/2 & 0 & 0 \\ 0 & 0 & \Delta t^2/2 & \Delta t & 0 & 0 \\ 0 & 0 & 0 & 0 & \Delta t^3/3 & \Delta t^2/2 \\ 0 & 0 & 0 & 0 & \Delta t^2/2 & \Delta t \end{bmatrix} \Phi_s \tag{3.8}$$

On the other hand, the Discrete Noise Model assumes the higher order is constant during the time step, but changes for each time period and each of these is uncorrelated between periods. For this case the Process Noise Matrix is shown next, where σ_v^2 is the process noise variance.

$$Q = \begin{bmatrix} \Delta t^4/4 & \Delta t^3/2 & 0 & 0 & 0 & 0 \\ \Delta t^3/2 & \Delta t^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \Delta t^4/4 & \Delta t^3/2 & 0 & 0 \\ 0 & 0 & \Delta t^3/2 & \Delta t^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \Delta t^4/4 & \Delta t^3/2 \\ 0 & 0 & 0 & 0 & \Delta t^3/2 & \Delta t^2 \end{bmatrix} \sigma_v^2 \tag{3.9}$$

This model does not include control input values, so the parameters for the Control Input Model are evaluated to empty values.

$$B = 0 \tag{3.10}$$

$$u = 0 \tag{3.11}$$

For the Observation Model the next parameters are designed.

$$z = [z_x \quad z_y \quad z_z]^T \tag{3.12}$$

$$\boldsymbol{R}=\begin{bmatrix}\sigma_{zx}^2 & 0 & 0\\ 0 & \sigma_{zy}^2 & 0\\ 0 & 0 & \sigma_{zz}^2\end{bmatrix} \tag{3.13}$$

$$\boldsymbol{H}=\begin{bmatrix}1 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 1 & 0\end{bmatrix} \tag{3.14}$$

As it can be seen, the Observation Model contains position values from the trilateration method as mean ($\boldsymbol{z}$) and variance ($\boldsymbol{R}$). This model does not comprise acceleration values.

3.1.1.2 Constant Velocity Model Constant Acceleration Controlled (CVCA)

As the previous model, CVCA is a first order kinematic model that comprises constant velocity. However, it introduces acceleration measures as control input values in order to increase knowledge in the predict step. The Control Input Model for the CVCA modifies the next parameters from the CV model.

$$\boldsymbol{B}=\begin{bmatrix}\Delta t^2/2 & 0 & 0\\ \Delta t & 0 & 0\\ 0 & \Delta t^2/2 & 0\\ 0 & \Delta t & 0\\ 0 & 0 & \Delta t^2/2\\ 0 & 0 & \Delta t\end{bmatrix} \tag{3.15}$$

$$\boldsymbol{u}=[u_{ax} \quad u_{ay} \quad u_{az}]^T \tag{3.16}$$

The acceleration values as control inputs contribute to the state variables (positioning and velocity) according to the Control Input Function designed above. The rest of the parameters remain as for the KF CV model.

3.1.1.3 Constant Acceleration Model (CA)

CA model introduces acceleration as a state variable for a second order kinematic model, so it comprises constant acceleration in the state propagation. The kinematic equations that model the dynamic system are shown next.

$$x=x_0+v\cdot t+\frac{1}{2}at^2 \tag{3.17}$$

$$v=v_0+at \tag{3.18}$$

The State Vector and State Covariance Matrix are shown below.

$$\boldsymbol{x} = [x \quad v_x \quad a_x \quad y \quad v_y \quad a_y \quad z \quad v_z \quad a_z]^T \tag{3.19}$$

$$\boldsymbol{P} = \begin{bmatrix} \sigma_x^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_{vx}^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{ax}^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_y^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{vy}^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{ay}^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \sigma_z^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma_{vz}^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma_{az}^2 \end{bmatrix} \tag{3.20}$$

The Process Model designing is introduced next.

$$\boldsymbol{F} = \begin{bmatrix} 1 & \Delta t & \Delta t^2/2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \Delta t & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta t & \Delta t^2/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t & \Delta t^2/2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.21}$$

For the Process Noise Model both approaches have been implemented as in the previous model. Next, both the Continuous Noise Model and the Discrete Noise Model implementations are shown for second order kinematic model.

$$\boldsymbol{Q} = \begin{bmatrix} \Delta t^5/20 & \Delta t^4/8 & \Delta t^3/6 & 0 & 0 & 0 & 0 & 0 & 0 \\ \Delta t^4/8 & \Delta t^3/3 & \Delta t^2/2 & 0 & 0 & 0 & 0 & 0 & 0 \\ \Delta t^3/6 & \Delta t^2/2 & \Delta t & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \Delta t^5/20 & \Delta t^4/8 & \Delta t^3/6 & 0 & 0 & 0 \\ 0 & 0 & 0 & \Delta t^4/8 & \Delta t^3/3 & \Delta t^2/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \Delta t^3/6 & \Delta t^2/2 & \Delta t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \Delta t^5/20 & \Delta t^4/8 & \Delta t^3/6 \\ 0 & 0 & 0 & 0 & 0 & 0 & \Delta t^4/8 & \Delta t^3/3 & \Delta t^2/2 \\ 0 & 0 & 0 & 0 & 0 & 0 & \Delta t^3/6 & \Delta t^2/2 & \Delta t \end{bmatrix} \Phi_s \tag{3.22}$$

$$\boldsymbol{Q}=\begin{bmatrix} \Delta t^4/4 & \Delta t^3/2 & \Delta t^2/2 & 0 & 0 & 0 & 0 & 0 & 0 \\ \Delta t^3/2 & \Delta t^2 & \Delta t & 0 & 0 & 0 & 0 & 0 & 0 \\ \Delta t^2/2 & \Delta t & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \Delta t^4/4 & \Delta t^3/2 & \Delta t^2/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \Delta t^3/2 & \Delta t^2 & \Delta t & 0 & 0 & 0 \\ 0 & 0 & 0 & \Delta t^2/2 & \Delta t & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \Delta t^4/4 & \Delta t^3/2 & \Delta t^2/2 \\ 0 & 0 & 0 & 0 & 0 & 0 & \Delta t^3/2 & \Delta t^2 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 & \Delta t^2/2 & \Delta t & 1 \end{bmatrix}\sigma_v^2 \tag{3.23}$$

This model does not include control input values, so the parameters for the Control Input Model are shown next.

$$\boldsymbol{B}=0 \tag{3.24}$$

$$\boldsymbol{u}=0 \tag{3.25}$$

Finally, the Observation Model for the CA is introduced next.

$$\boldsymbol{z}=[z_x \quad z_{ax} \quad z_y \quad z_{ay} \quad z_z \quad z_{az}]^T \tag{3.26}$$

$$\boldsymbol{R}=\begin{bmatrix} \sigma_{zx}^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_{zax}^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{zy}^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{zay}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{zz}^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{zaz}^2 \end{bmatrix} \tag{3.27}$$

$$\boldsymbol{H}=\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.28}$$

The CA model introduces acceleration measurements in the Observation Model, unlike the CVCA model that introduces them as control input variables.

3.1.2 Extended Kalman Filter

The second approach to solve the positioning problem is the non-linear version of the Kalman Filter in the Tightly Coupled (TC) sensor fusion configuration. In this implementation the UWB ranges are introduced directly into the filtering algorithm with the acceleration measurements. The non-linearity is then solved by the EKF approach.

The non-linearity of a problem for the Kalman Filter can be found in the Process Model or in the Observation Model, the State Transition Function ($\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})$) and the Measurement Function ($\boldsymbol{h}(\bar{\boldsymbol{x}})$) respectively. In this case they become non-linear functions. Then, the EKF linearizes the problem on a point ($\boldsymbol{x}_t, \boldsymbol{u}_t$) by calculating the partial derivative with respect to the state variables. For matrices, the Jacobian is calculated for both functions.

$$\boldsymbol{F}=\left.\frac{\partial f(\boldsymbol{x}_t,\boldsymbol{u}_t)}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_t,\boldsymbol{u}_t} \tag{3.29}$$

$$\boldsymbol{H}=\left.\frac{\partial h(\bar{\boldsymbol{x}})}{\partial \bar{\boldsymbol{x}}}\right|_{\bar{x}_t} \tag{3.30}$$

Then, the EKF algorithm uses the Kalman Filter equations with the linearized problem. The algorithm remains the same, but for this case the State Transition Matrix ($\boldsymbol{F}$) and the Measurement Matrix ($\boldsymbol{H}$) are approximated by the first partial derivatives, the Jacobian, evaluated at the current state. The general algorithm for the EKF is shown in the appendix A5.

In this concrete case under study, the Process Model remains the same way as in the previous implementation on the linear filter. The non-linearity problem in this scenario is found in the Observation Model, as the UWB ranging returns measurement distances modelled by the expression in (1.5). So, the Measurement Function becomes a non-linear expression instead of a matrix.

Therefore, to solve the non-linear problem, the Measurement Function is linearized by evaluating its partial derivative at the state prior. To do so, the Jacobian of the Measurement Function is calculated with respect to the state variables (3.34). Then, this linearized Measurement Function will be introduced to the KF algorithm as the Measurement Matrix ($\boldsymbol{H}$).

For the concrete evaluation of the designing parameters 3 filtering models will be defined in the next subsections. For each filtering model the differences between the linear algorithm will be provided in order to shorten the extension of this document.

3.1.2.1 Constant Velocity Model (CV)

The State-Space Model and the Process Model remain unchangeable, so the State Vector ($\boldsymbol{x}$), the State Covariance Matrix ($\boldsymbol{P}$), the State Transition Function ($\boldsymbol{F}$) and the Process Noise Matrix ($\boldsymbol{Q}$) will not be modified from the linear implementation.

Otherwise, in the Observation Model little change will be done in the Measurement Function, as it becomes a non-linear function.

$$\boldsymbol{h}(\bar{\boldsymbol{x}})=[\sqrt{(x-a_x)^2+(y-a_y)^2+(z-a_z)^2}] \tag{3.31}$$

The linearization is done by calculating the Jacobian of the expression above with respect to the state variables and evaluated at the state prior. This way, the linear Measurement Matrix is obtained.

$$\boldsymbol{H}=\begin{bmatrix} \frac{x-a_x}{\sqrt{(x-a_x)^2+(y-a_y)^2+(z-a_z)^2}} & 0 & \frac{y-a_y}{\sqrt{(x-a_x)^2+(y-a_y)^2+(z-a_z)^2}} & 0 & \frac{z-a_z}{\sqrt{(x-a_x)^2+(y-a_y)^2+(z-a_z)^2}} & 0 \end{bmatrix} \tag{3.32}$$

3.1.2.2 Constant Velocity Model Constant Acceleration Controlled (CVCA)

The State-Space Model and the Process Model remain unchangeable, so the State Vector ($\boldsymbol{x}$), the State Covariance Matrix ($\boldsymbol{P}$), the State Transition Function ($\boldsymbol{F}$) and the Process Noise Matrix ($\boldsymbol{Q}$) will not be modified from the linear implementation. The Control Input Function ($\boldsymbol{B}$) and the Control Input Vector ($\boldsymbol{u}$) will not be modified either.

In the Observation Model the same modification for the EKF CV Model will be done (3.31 and 3.32).

3.1.2.3 Constant Acceleration Model (CA)

The State-Space Model and the Process Model remain unchangeable, so the State Vector ($\boldsymbol{x}$), the State Covariance Matrix ($\boldsymbol{P}$), the State Transition Function ($\boldsymbol{F}$) and the Process Noise Matrix ($\boldsymbol{Q}$) will not be modified from the linear implementation.

In the Observation Model some modifications will be done in the Measurement Function ($\boldsymbol{h}(\bar{\boldsymbol{x}})$), and so in its linear approximation for the Measurement Matrix ($\boldsymbol{H}$). The current model takes acceleration values as measurements variables. Therefore, the Measurement Function is implemented as shown next.

$$\boldsymbol{h}(\bar{\boldsymbol{x}})=\begin{bmatrix} \sqrt{(x-a_x)^2+(y-a_y)^2+(z-a_z)^2} \\ z_{ax} \\ z_{ay} \\ z_{az} \end{bmatrix} \tag{3.33}$$

$$\boldsymbol{H}=\begin{bmatrix} \frac{x-x_a}{\sqrt{(x-x_a)^2+(y-y_a)^2+(z-z_a)^2}} & 0 & 0 & \frac{y-y_a}{\sqrt{(x-x_a)^2+(y-y_a)^2+(z-z_a)^2}} & 0 & 0 & \frac{z-z_a}{\sqrt{(x-x_a)^2+(y-y_a)^2+(z-z_a)^2}} & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.34}$$

3.1.3 Algorithm code implementation

The implementation on the filtering algorithm is done in an interactive Jupyter Notebook that can be found in the annex. The programmatically approach for the filtering implementation is to create a Python class that provides the parameters, as vectors and matrices required to the filter computation according to the designed filtering model. To create a model, a Python class must be created and this class must implement the abstract methods from *AbstractModel* class in *models/abstract_model.py*. These models are stored in the modules *kf_models.py* and *ekf_models.py* under the *models/* package. The concrete implementation on each model parameters has been done according to the previous sections. The UML diagram for Python classes that implement filtering models is shown in the appendix A7. Also, the blocks diagrams that shows the filtering process and the configurable parameters list is provided in A8. This implementation is available on the

Jupyter Notebook created for filtering and evaluation, and it can be run interactively by executing the code cells.

3.1.4 Filter input

The Kalman Filtering algorithms use Gaussian probability distributions and linear algebra to compute optimal state estimates. So, the input data model assumes that the noisy measurements from sensors can be modelled by Gaussian distributions characterized with two parameters, mean and variance. For the multivariate case of the Gaussian probability distribution the next expression is used.

$$N(\mu,\Sigma) \tag{3.35}$$

Where μ,Σ are the mean vector and covariance matrix for the multivariate Gaussian in n dimensions.

$$\mu=\begin{bmatrix}\mu_1\\ \mu_2\\ \dots\\ \mu_n\end{bmatrix} \tag{3.36}$$

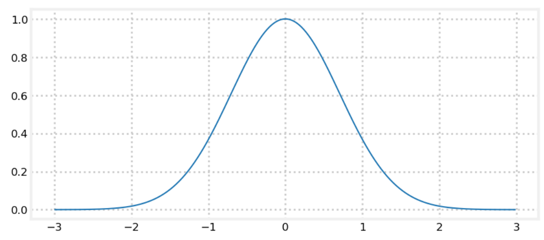
$$\Sigma=\begin{bmatrix}\sigma_1^2 & \sigma_{12} & \dots & \sigma_{1n}\\ \sigma_{21} & \sigma_2^2 & \dots & \sigma_{2n}\\ \dots & \dots & \dots & \dots\\ \sigma_{n1} & \sigma_{n2} & \dots & \sigma_n^2\end{bmatrix} \tag{3.37}$$

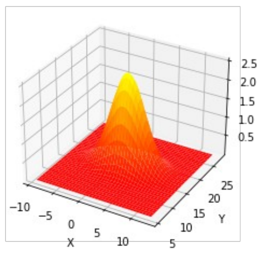
Finally, the multivariate normal (or Gaussian) distribution has the next mathematical expression.

$$f(\boldsymbol{x},\mu,\Sigma)=\frac{1}{\sqrt{(2\Pi)^n|\Sigma|}}\exp\left[\frac{-1}{2}(\boldsymbol{x}-\mu)^T\Sigma^{-1}(\boldsymbol{x}-\mu)\right] \tag{3.38}$$

A graphical visualization of normal distributions is shown next.

Figure 3.1: Univariate Gaussian distribution (a) and multivariate Gaussian distribution (b)





(a) (b)

So, the filtering process in Kalman Filtering introduces sensors' measurements as inputs in the update step. These can be modelled by Gaussian distributions as mean and variance values. Additionally, the time step is introduced in the filter for the predict step.

3.2 Simulation

A simulation allows to imitate a real-world process operation in a totally controlled virtual environment, where customizable parameters can be easily changed and evaluated. For such purpose a model on system input has been designed. For handling all code a Python project and Jupyter Notebook have been developed, where code can be easily run and modified interactively.

3.2.1 Setup and initialization

The previous step before running the simulation is to configure the working environment. For the execution of the simulations, Python 3.x and Jupyter Notebook server must be installed. Please, refer to the Jupyter Project official web page and follow the installation steps [15]. Additionally, other Python modules and packages are required for miscellaneous purposes. These all are listed in the *requirements.txt* file in the project root folder. To install all the dependencies with *pip* [16] open a terminal and run the following command on the project root folder:

```
$ pip install -r requirements.txt
```

3.2.2 Measurements input

The first step in the simulation process is to generate noisy measurements data for filtering input. The data format generated depends on the filter implementation, so it changes for the standard KF and EKF. The implemented functions for this purpose are in the module *utils/random_stream.py*.

The simulated measures for the UWB ranges and acceleration values contain Gaussian noise. The parameters for the measurements noise, as well as other parameters for the input data can be configured interactively in the generator inside the Jupyter Notebook. In the appendix A6 a flowchart for the generator's operation process and a table of configurable parameters are provided for better understanding on simulated data generation and configuration.

The generator takes initial position, velocity and acceleration for each axis, as well as other configurable parameters such as variances for UWB/INS measures, time step between measurements and sample count, and generates corresponding noisy measurements for the filtering input according to kinematic equations in (1.2) and (1.3). This implementation for the generator allows comparing both KF and EKF approaches as the obtained measurements simulate real scenario operation. The next table shows the initial values given to the simulation. The rest of the parameters will be configured on each study case defined later in this section.

Table 3.1: Initial simulation values

Parameter	Value	Units
init_pos	(0, 0, 1)	m
init_vel	(1.8, 0.05, 0)	m/s
init_acc	(0, 1, 0)	m/s^2

dt	1	s
count	100	
anc	(120, 50, 1),	m
	(250, 300, 1),	
	(80, 400, 1.2),	
	(0, 200, 0.8)	

3.2.3 Study cases

In order to evaluate and compare each implementation some study cases have been designed. Each study case will evaluate some filtering parameters and will check how the filtering responds to this changes. The concrete evaluation on each parameter is described in the appendix A9 in order to get reproducible data. The results are shown in the next section on this document and conclusions will be listed in the corresponding section according to the results obtained.

3.2.3.1 Ideal case

For this first study case ideal measurements will be generated, with no added noise. Also a perfect initialization on the state will be done. This test allows to evaluate how precise the Process Model is in the predict step, as in the update step the measures will be ideal.

3.2.3.2 Bad initialization

To evaluate how fast the filter converges into the real value, a bad initialization will be done in the initial state estimate.

3.2.3.3 Optimal design

Finally, an optimal design will be done to the filter. For this case, an uncertain initialization will be done and measurements variance will be modelled correctly.

3.2.4 Simulation constraints

The simulation has many benefits before a real implementation of a design, as it allows to control many of the parameters. The best the simulation models the real system operation, the best the results emulate a real test. However, it gets almost impossible to consider all the variables of a real scenario test. For this reason, in the simulation process some approximations and simplifications are done with respect to the real dynamic system. The next list enumerates some of this simulation constraints that have to be taken into account in the conclusions section:

- It is supposed to get a higher variability of a dynamic system in a real scenario: irregular path, changes of direction, velocity and acceleration, etc.
- The noise for the sensor measurements is not ideally Gaussian with zero mean.
- The positioning of the anchors in a virtual scenario is defined, not measured. So it can be defined ideally.

- Number of samples and error on UWB ranging measurements depend on the environmental conditions (LOS, NLOS, beam reflection, outliers, …).
- Different data acquisition rates for real sensors.

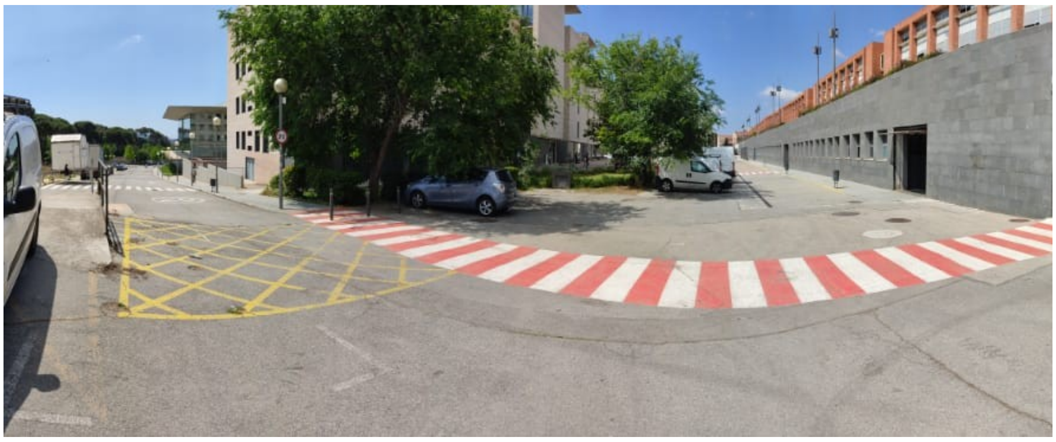
3.3 Real scenario test

On the current section the setup and configuration for the real scenario tests are introduced. The results from the real tests in addition with the result from the simulations will validate the system design.

3.3.1 Setup and initialization

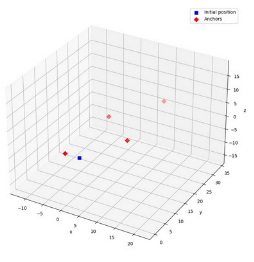
The current real tests have been carried out in the campus of Universitat Politècnica de Catalunya University in Campus Nord, Barcelona. The concrete scenarios comprise a straight way and a 90 degree bend.

Figure 3.2: Real test scenario image

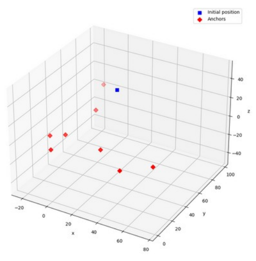


The UWB hardware used for this tests is the In-Circuit radino32 DW1000 [17]. According to the manufacturer it can provide 10 cm-level accuracy. Additionally, GY-521 IMU device [18] based in the MPU-6050 sensor [19] will be used. This sensor contains 3-axis gyroscope and 3-axis accelerometer. For the real test two scenarios have been configured.

Figure 3.3: Straight-way scenario (a) and 90-degree-bend scenario (b) setups



(a)



(b)

In (a) UWB ranging measures and acceleration updates are obtained each 55ms in a 40-meters-length straight-way trajectory. However, in (b) different data acquisition rates are obtained from UWB ranging and the accelerometer. UWB ranges are computed each second (1Hz), while acceleration updates are obtained each 100ms (10Hz). This constraint is related to the UWB infrastructure deployment, to test whether with less UWB updates the positioning system can maintain its accuracy and availability levels.

3.3.2 Measurements input

The UWB ranging distance measurements and acceleration values are obtained in a csv file after the data acquisition stage. Before introducing to the filter the acceleration data must be transformed from the body frame to the global frame reference in order to compute positioning updates. For this refer to the equations in (1.6) and (1.7). Additionally, for the LC configuration in the standard KF implementation trilateration must be done with UWB range measurements before the filtering process. The input measurements csv files can be found in the *samples/* folder and the parsers used for raw data in *utils/_parser.py* module. For these real scenario tests *{kf, ekf}recto2.csv* and *10Hz-{kf, ekf}-irampa.csv* files will be used. Each one of them is parsed according to the filtering implementation (KF or EKF) for one of the scenarios (a) or (b) mentioned above.

3.3.3 Study cases

For both real tests the same models in the simulation will be used (CV, CVCA, CA). However, the initial configuration parameters will vary for better operation on the data filtering. The initial configuration and setup values for both scenarios (a) and (b) are described in the appendices A11 and A12 respectively. For both uncertain initialization will be done and the measurements error will be modelled according to the sensor specifications with slight variance. For the Process Noise Model the Continuous model will be used instead, as the real system operation comprises continuous noise unlike the simulation scenario.

3.3.4 Real scenario test constraints

The real test results serve as a validation method for a system design. However, the current scenario tests have many inaccuracies that should have to be solved in future tests to evaluate correctly the system performance. First, the test environment comprised many challenges as there were many objects and obstacles that could interfere in the UWB signal, such as buildings, cars and EMI from electronic devices. Next, important error was made in the anchors positioning. Additionally, initial positioning and orientation values were approximated.

3.4 Output evaluation

In order to evaluate the filter performance between different designs and operations, two evaluating approaches will be implemented. The code implemented for this purpose is found in the function *evaluate_filter()* developed in the Jupyter Notebook.

First, some visualizations will be created on the data. Filter output, prior, real path and measurements (when presented) will be graphically plotted in a 3D plot. Additionally, 2D projections will be provided.

Next, error will be calculated between real path and filter output for each axis. In addition, total three-dimensional error and plane error (2D) will be calculated, too. The next expression shows the computation of the error in a multidimensional approach.

$$\boldsymbol{e}=\boldsymbol{r}-\boldsymbol{x}=\begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix} \tag{3.39}$$

Where $\boldsymbol{r}$ is the real path, $\boldsymbol{x}$ the output of the filter and $\boldsymbol{e}$ is the computed error. So, the error module is calculated next.

$$|\boldsymbol{e}|=\sqrt{e_x^2+e_y^2+e_z^2} \tag{3.40}$$

$$|\boldsymbol{e_{2D}}|=\sqrt{e_x^2+e_y^2} \tag{3.41}$$

It is important to distinguish total error and 2D error, as the error in z axis is not such critical for the positioning problem for autonomous driving applications, and usually gets higher order values as the anchors and the trajectory are in the same horizontal plane.

Finally, some descriptive statistics on the error values are performed, such as mean, min and max values. Additionally, Root Mean Square Error is calculated.

$$RMSE=\sqrt{\frac{1}{N}\sum e^2} \tag{3.42}$$

The graphical representation of the error is also provided with bar plots and histograms for each axis.

4 Results

On the current section the results obtained from the simulations and the real scenario tests will be provided. After the filtering process, the *evaluate_filter()* function implemented in the project takes measurements stream, filtered data and real path, and computes some visualizations and descriptive statistics to evaluate the filter performance. These results shown below contain plots and visualizations for these data streams, as well as error statistics as a summary to make correct conclusions on the system performance. For deeper analysis on the evaluation outputs refer to the appendices.

4.1 Simulation results

The results for the simulations are divided in corresponding study cases defined in the Methodology section. In the appendix A10 some outputs and error plots are shown for each study case.

4.1.1 Ideal case

The ideal case simulation introduces ideal measurements data (with no added noise) to the filter. This study case allows evaluating the Process Model's systematic error on the dynamic system modelling for each implementation. In the optimal case, the Process Model perfectly models the dynamic system and the predict step behaves ideally. However, all models are approximations for real systems, and then the Process Noise Model comprise these unmodelled factors. The next table shows the error statistics.

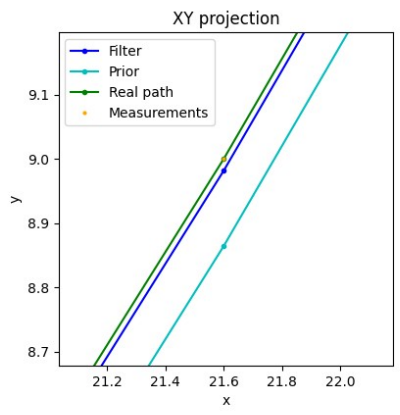
Table 4.1: Error statistics summary for ideal case simulation

	Kalman Filter						Extended Kalman Filter						UWB stand-alone	
	CV		CVCA		CA		CV		CVCA		CA			
	err2d	err	err2d	err	err2d	err	err2d	err	err2d	err	err2d	err	err2d	err
mean	0.087	0.168	~0	0.125	0.001	0.123	0.015	0.070	~0	~0	~0	~0	~0	0.121
std	0.010	0.116	0.002	0.136	0.002	0.128	0.031	0.059	0.005	0.005	0.005	0.005	~0	0.141
max	0.092	0.658	0.011	0.652	0.012	0.600	0.159	0.218	0.092	0.092	0.089	0.089	0.001	0.680
RMSE	0.088	0.204	0.002	0.184	0.002	0.177	0.034	0.091	0.005	0.005	0.005	0.005	~0	0.185

In general, the error values obtained are low. However, it must be considered that the measurements are ideal. The stand-alone UWB positioning for the ideal case has optimal performance on the XY plane. However, for the three-dimensional case higher error values are obtained due to the error in the Z axis. The reason for obtaining higher order error values on the Z axis is that the trajectory, as well as the anchors are oriented along the horizontal XY plane, and thus positioning on z becomes more difficult to solve as there is low variability in the z coordinates of the anchors' positions and the trajectory.

The input data for the simulation contains acceleration of value 1m/s^2 in the Y axis. As it can be checked the errors for the KF CV and EKF CV implementations are the highest errors comparing to the other models for the same algorithm. This can be easily seen in the appendix A10.1 for the KF CV error bar plot, where there is a constant error in the y coordinates. The reason for this is that the CV model is a first order kinematic model and thus, it does not include acceleration in its computation. In this way, the prediction step is always behind the accelerated movement in the y coordinate. This behaviour can be shown in the next figure.

Figure 4.1: Prediction in the XY plane for the KF CV model



Finally, in optimal conditions it can be seen that the Extended Kalman Filter operation for CVCA and CA models enhances the performance for the positioning problem as the error obtained is very small. The better filtering operation with slightly difference is for the EKF CA implementation. However, it must to be analysed how the filters react to noisy data. This will be done in the next study cases.

4.1.2 Bad initialization

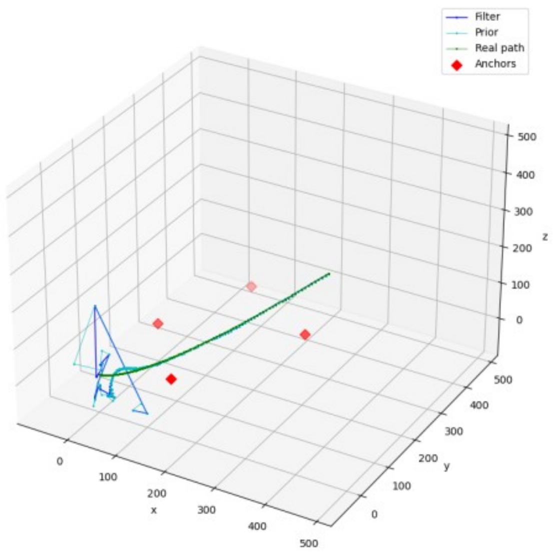
For the current study case a bad initialization in the state mean and covariance has been configured. A bad initialization on the state supposes a bad estimate on the state for the filter initialization, as well as little uncertainty in the state. This way the filter is initialized on a bad location with high belief on the wrong value. Additionally, unlike the previous study case, the current measurements are no longer ideal and contain added Gaussian noise. This test allows evaluating how fast the filter corrects the bad initial state estimate and converges into the real value of the trajectory.

Table 4.2: Error statistics summary for bad initialization case simulation

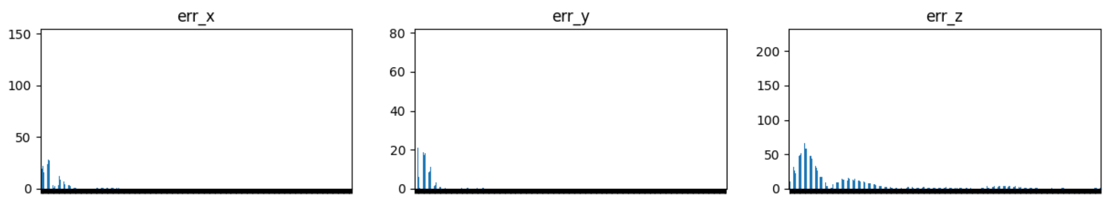
	Kalman Filter						**Extended Kalman Filter**						**UWB stand-alone**	
	CV		**CVCA**		**CA**		**CV**		**CVCA**		**CA**			
	err2d	**err**	**err2d**	**err**	**err2d**	**err**	**err2d**	**err**	**err2d**	**err**	**err2d**	**err**	**err2d**	**err**
mean	0.579	2.417	0.553	2.407	0.934	2.499	1.490	4.769	1.490	4.579	2.313	8.843	0.109	2.395
std	2.008	2.297	2.012	2.293	2.531	2.494	11.653	16.796	11.655	16.847	12.220	21.197	0.064	2.260
max	13.749	13.778	13.745	13.774	14.335	14.365	166.549	233.477	166.531	234.440	166.426	224.757	0.359	10.375
RMSE	2.080	3.326	2.077	3.316	2.685	3.521	11.734	17.440	11.735	17.437	12.422	22.944	0.127	3.285

The results show bad performance for EKF models in bad initialization conditions. In the appendix A10.4 error bar plots show high error for first iterations of the filter. However, after many iterations the filter converges to the actual value with lower error. This could be an expected result, as for each iteration distance measurements are used, instead of trilateration results for the KF. Then, each distance measurement implies infinite number of positions in a sphere of the same radius around the anchor used to compute the ranging measurement. For this reason, after a bad initialization the filter delays many iterations of the algorithm until it converges into the real path.

Figure 4.2: EKF CA output plot (a) and error bar plot (b) for bad initialization



(a)



(b)

Otherwise, the KF models in LC configuration have better performance on bad initialization conditions. The reason for this is that in the first iteration a positioning value from trilateration is introduced directly. For the first iterations of the filter the output converges more rapidly to the real values. The error plots for KF models in the appendix A10.3 show this behaviour in the horizontal coordinates. For the CVCA and CA models the initial transient stage is slightly shorter than for the CV model, as they both introduce acceleration values. However, for the KF CA model in the Y axis, some error disturbances that oscillate in values close to 30cm can be seen.

The best performance for the sensor fusion solution in bad initialization scenario is obtained by KF CVCA. In this approach the maximum error value is obtained on the first iterations, and the error is rapidly reduced on subsequent iterations of the filter.

Finally, it can be checked that the UWB stand-alone configuration works much better than sensor fusion solutions. This is because a bad initialization affects on the filter performance increasing the error for the predict step. A solution for this is to make an uncertain initialization. This way the filter converges more rapidly as the first updates are more confident than the predictions for initial estimates.

4.1.3 Optimal design

The last study case is supposed to emulate a real scenario design. For this purpose an uncertain initialization is done. Also, measurements' noise is correctly modelled by the variance. The next table shows the error statistics summary.

Table 4.3: Error statistics summary for optimal design case simulation

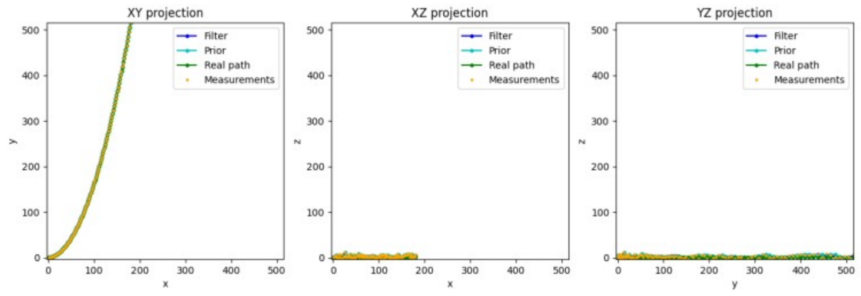
	Kalman Filter						Extended Kalman Filter						UWB stand-alone	
	CV		CVCA		CA		CV		CVCA		CA			
	err2d	err	err2d	err	err2d	err	err2d	err	err2d	err	err2d	err	err2d	err
mean	0.124	2.030	0.101	2.021	0.182	1.867	0.162	2.454	0.159	2.356	0.122	1.925	0.109	2.395
std	0.053	1.644	0.053	1.637	0.090	1.347	0.136	2.325	0.136	2.359	0.090	2.101	0.064	2.260
max	0.255	7.177	0.295	7.123	0.387	5.365	1.236	13.881	1.199	13.869	0.889	12.701	0.359	10.375
RMSE	0.135	2.607	0.114	2.596	0.203	2.298	0.211	3.378	0.209	3.332	0.151	2.848	0.127	3.285

As seen in the previous study cases, the error in the Z axis is higher than in the other axis, because of the small variability of the anchors positions along that axis, resulting in less information contained in the third coordinate. However, the horizontal error on the XY plane will be evaluated first, which is more critical on autonomous driving systems.

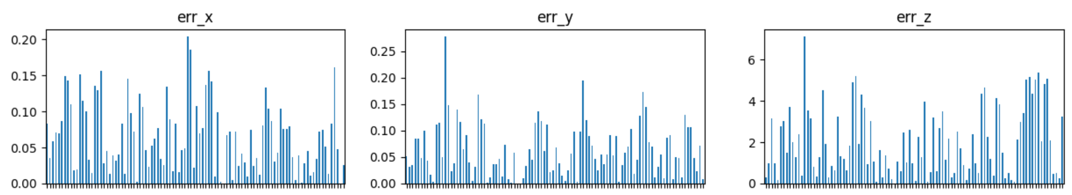
For this scenario the KF CVCA model gets the best performance due to its maximum error and RMSE values. However, KF CV model has similar results and should have to be considered. The UWB stand-alone configuration shows little worse operation in 2D, but for the three-dimensional approach the error increases considerably.

The next figure shows the projections for the filter output and the error bar plot for each axis. Most error values for X and Y axis are below 20 cm.

Figure 4.3: KF CVCA projections (a) and error bar plot (b)



(a)



(b)

The EKF approach shows low error mean and standard deviation on CVCA and CA models. However, the maximum error achieved can not be allowed for positioning in an autonomous driving system. Refer to the error plots in the appendix A10.6 for better interpretation on the error for the EKF approaches.

4.1.4 Simulation results summary

The results have showed that in the current simulated scenarios the sensor fusion solution performs better than the UWB positioning in stand-alone configuration. Moreover, it reduces considerably the error in the Z axis due to the low variability of the samples along this axis. However, the performance along the Z axis could be enhanced placing the anchors at different heights along the road. Future improvements and considerations will be discussed later in this document.

Additionally, the simulations conclude that the KF performs better than the EKF in many scenarios, as long as there are enough range measurements for trilateration. The lack of ranging distance measurements and the existence of outliers in measures have not been covered for these simulations, in which case the EKF operation will be increased in comparison to the KF's, as it is not required a minimum number of distance measurements for positioning with EKF in TC configuration. For the EKF models, the CVCA has better error statistics than the other approaches on the 3 study cases. Additionally, without taking into account the initial stage of the EKF filter until converging to the real value, error statistics obtained prove that it could be useful in some scenarios.

Finally, it has been shown that the performance of the models that include acceleration values, such as CVCA and CA, gets better. Generally, the CVCA model has had the best performance for the simulations. However, the samples generated did not contain high variability in acceleration, and thus modelling the dynamic system in a first order kinematic model has been accurate. But in some scenarios with high variations on acceleration and big changes on direction it may be required to implement a second order model containing acceleration.

4.2 Real scenario test results

In the current section the same procedure from the simulations will be followed in order to evaluate the filter performance. The results for both tests are shown next as error statistics summary. However, in the appendix A13 more extensive error analysing results are provided, such as error bar plots and histograms.

4.2.1 Straight-way scenario test

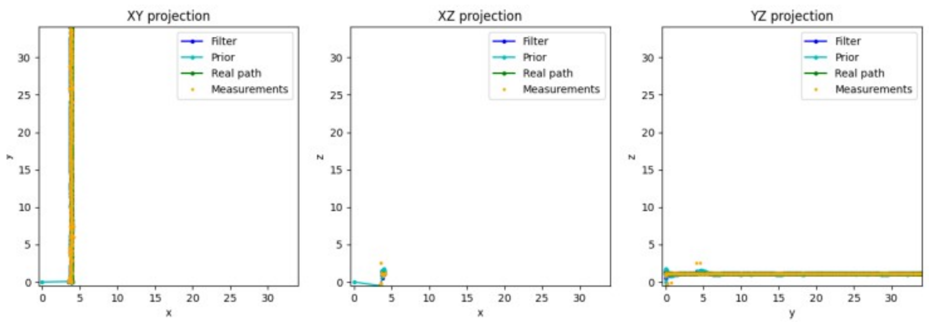
In this scenario setup a straight-way trajectory is followed by the tag in LOS conditions, so no outliers have been found in the measurements data. The error summary for each filtering approach is shown below.

Table 4.4: Error statistics summary for straight-way scenario

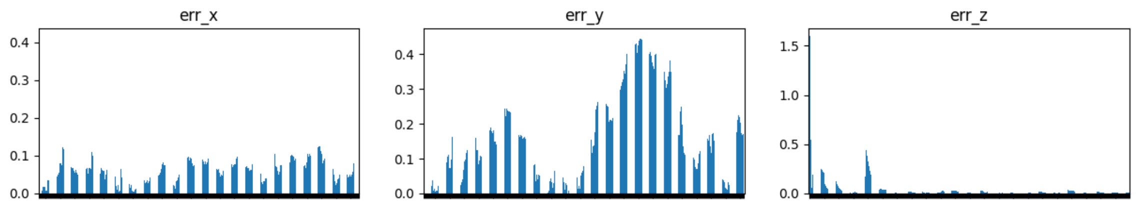
	Kalman Filter						Extended Kalman Filter						UWB stand-alone	
	CV		CVCA		CA		CV		CVCA		CA			
	err2d	err	err2d	err	err2d	err	err2d	err	err2d	err	err2d	err	err2d	err
mean	0.220	0.244	0.186	0.211	2.597	2.624	0.240	0.574	0.217	0.605	1.897	2.573	0.226	0.245
std	0.148	0.177	0.115	0.156	1.429	1.403	0.213	0.467	0.212	0.456	0.952	0.551	0.149	0.218
max	0.544	1.650	0.454	1.650	5.201	5.215	4.046	5.533	4.046	5.533	4.046	5.533	0.576	1.650
RMSE	0.265	0.302	0.218	0.262	2.963	2.974	0.321	0.740	0.303	0.757	2.122	2.631	0.270	0.328

For the current scenario the best performance is obtained with KF CVCA model. However, EKF CVCA also shows good performance on error mean and RMSE. The problem for the EKF implementation is that it takes many iterations until it finally converges to the actual value. This behaviour can be checked in error plots for EKF in A13.2.

Figure 4.4: KF CVCA projections (a) and error bar plot (b)



(a)



(b)

As it can be seen in the figure above, the error values on X axis are below 20cm. However, higher errors are obtained for the Y coordinate with a remarkable increase at the second half of the trajectory.

4.2.2 90-degree-bend scenario test

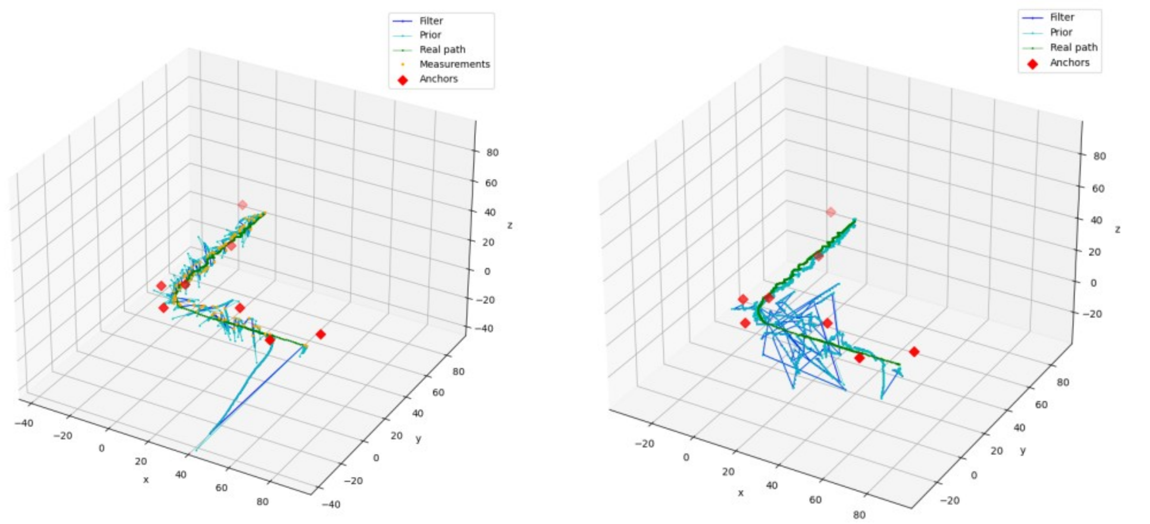
For this test, the UWB measurements have been affected by the environment factors that have caused bad operation on filtering. Next, the error statistics are shown.

Table 4.5: Error statistics summary for 90-degree-bend scenario

	Kalman Filter						Extended Kalman Filter						UWB stand-alone	
	CV		CVCA		CA		CV		CVCA		CA			
	err2d	err	err2d	err	err2d	err	err2d	err	err2d	err	err2d	err	err2d	err
mean	8.772	10.021	9.500	10.891	9.958	12.125	8.637	9.709	8.759	10.181	10.054	11.680	8.787	9.633
std	6.347	5.912	7.828	8.391	9.056	12.780	6.430	7.335	6.495	7.527	5.245	4.816	6.280	5.739
max	30.172	30.186	49.373	58.759	68.716	102.414	43.969	49.856	44.053	50.247	24.210	29.632	23.458	23.506
RMSE	10.826	11.634	12.308	13.746	13.458	17.613	10.767	12.167	10.903	12.660	11.339	12.634	10.780	11.197

As it can be checked, the error statistics have increased from previous tests. This is due to the measurement outliers in the data acquisition stage, were the environmental conditions caused bad measurements on distances from UWB. Even if the error values are unaffordable for precision positioning purposes, the EKF filter performs better on the first straight stretch of the track, while it diverges on the curve due to the outliers in measurements.

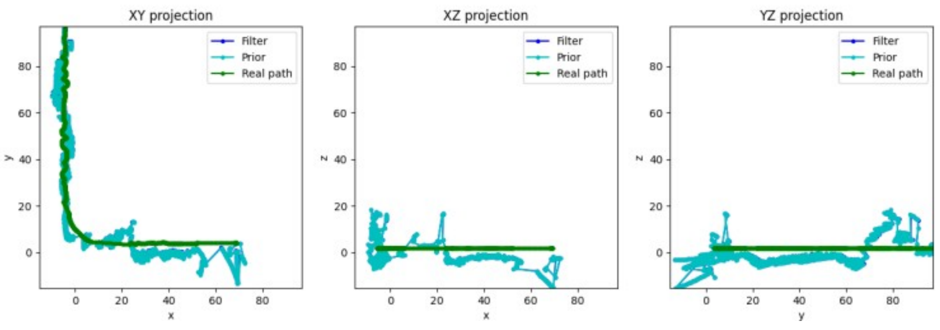
Figure 4.5: Track for KF CVCA (a) and EKF CV CVCA (b)

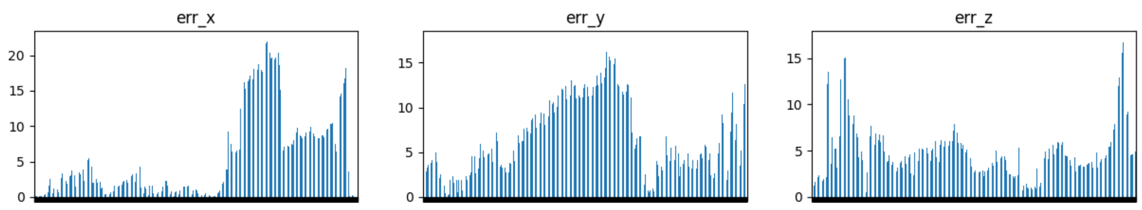


(a) (b)

As an interesting result in A13.4 the EKF CA model shows better operation on the outliers section than the rest of the implementations.

Figure 4.6: Projections (a) and error bar plot (b) for EKF CA





4.2.3 Real tests summary

In general, due to the environment effects on the UWB ranging operation, the tests performed in real scenarios have not reached the precision requirements for the positioning purposes on autonomous driving systems. However, they are intended to be a starting point on the development and implementation process of the UWB technology for outdoor scenarios.

According to the filtering models implemented, the results have shown better performance for first order kinematic models. However, in the last scenario due to higher variability on the trajectory direction the second order kinematic model has obtained better results.

In real tests the Process Noise Model must be designed accordingly to each Process Model implemented. However, these optimization on Process Noise Model has not been done; instead, a comparison between Process Models under similar conditions was one of the main goals. Thus, the performance could have been enhanced with a more accurate design on dynamic system modelling.

5 Budget

The system cost evaluation and resources optimization is outside of the scope of this thesis. However, some approximated cost values on testing scenario setup and deployment will be detailed in the next table for a 100 meters road. It should be highlighted that these are guide values and that more detailed study must be done for this system implementation on production environments.

Table 5.1: Budget summary table

Item/task/component	Unit	Cost/unit	Total amount
UWB Anchors + Tag	8+1	34.90€	314.1€
IMU GY-521-MPU6050	1	4.29€	4.29€
Installation and setup	6 h	15€	90€
System design and modelling	60 h	25€	1500€
		TOTAL COST	1908.39€

One of biggest drawback of this positioning system approach based on Ultra Wide Band technology is that it requires a large investment on a dedicated UWB infrastructure, as it requires multiple UWB anchors placed on both sides of the road. Therefore, its deployment should be restricted to areas where main positioning approaches become unreliable, such as rural roads and tunnels.

6 Environment Impact

The UWB technology infrastructure deployment does not imply a huge environmental impact beyond the need to produce required hardware. However, as this technology has wide frequency spectrum (over 500MHz) the overlap with other radio signals should be evaluated in the future in order to avoid interferences.

7 Conclusions and future development:

In this thesis a solution for outdoor positioning system has been designed and evaluated. This hybrid system combines UWB and INS technologies in a sensor fusion solution by means of Kalman Filter. For better evaluation on the system performance both approaches of the Kalman Filter have been studied: the standard Kalman Filter and the Extended Kalman Filter. Additionally, some filtering models have been designed and tested for each study case.

The results from this simulations, evaluated as error mean, standard deviation, maximum error and RMSE have proved that the best models to describe the simulated trajectory are first order kinematic models. These models include constant velocity for the state propagation estimate. However, the acceleration values from an accelerometer have improved many of the operations for the CVCA model. Additionally, they highlight the sensor fusion as a better approach for the improvement of positioning instead of stand-alone configurations.

The Extended Kalman Filter algorithm has shown a good performance on the simulations for the ideal case. However, under bad initialization conditions the filter delayed converging into the real value that resulted in high errors for the first iterations of the filter. This behaviour shows the importance of the initialization of the filter, specially for the EKF. For the rest of the tests, generally, the non-linear approach has shown slightly worse performance than the KF. This result can be expected, as the EKF performs an approximation for the non-linear problem as a linearization evaluated at a point. However, the EKF can perform updates with single range measurements, while the KF requires a minimum of samples to perform the trilateration.

First and second kinematic Process Models are precise enough for tracking the positioning of a moving object. However, more research and test must be done in positioning of faster objects and dynamic systems with higher order variability. Additionally, a deeper study on Process Noise Models must be done in order to improve the modelling. According to the Observation Model, the measurements' precision and availability must be enhanced in order to get better performance of the system. Also, for better measurements in the testing stage, the anchors must be globally located in well-known positions with high precision.

In general, it can be concluded that the standard Kalman Filter algorithm with the Constant Velocity Model Constant Acceleration Controlled (CVCA) gets better position estimates for the real scenario tests. However, the generic implementations of both algorithms are not precise enough for positioning on autonomous driving. For such purpose, improvement on the algorithms must be done in order to detect outliers in the measurements and reject them. In addition, it could be interesting mixing both linear and non-linear approaches of the Kalman Filter. In this case, when there are not enough ranging measurements for trilateration with the KF, or the error increases, the EKF can perform these updates.

It has been proved that the outdoor environments' variability, such as NLOS and multipath propagation conditions, deteriorates the UWB ranging performance and generates outliers on the filter input, that result into bad operation. Future work could be done in the mitigation of the environmental effects on UWB ranging.

Bibliography:

1. Igual Nevot, J. (2020, October). Improving location of vehicles in rural roads (Projecte Final de Màster Oficial). UPC, Escola Tècnica Superior d'Enginyeria de Telecomunicació de Barcelona, Departament d'Enginyeria Telemàtica. Retrieved from http://hdl.handle.net/2117/340252

2. Ultra Wide Band System – Trilateration. March 2016 [Online] Available: http://cstwiki.wtb.tue.nl/index.php?title=Ultra_Wide_Band_System_-_Trilateration. [Accessed: 19 June 2021]

3. J. Peraire, S. Widnall. "Lecture L29 – 3D Rigid Body Dynamics". MIT, 2009. [Online] Available: https://ocw.mit.edu/courses/aeronautics-and-astronautics/16-07-dynamics-fall-2009/lecture-notes/MIT16_07F09_Lec29.pdf. [Accessed: 17 June 2021]

4. Roger Labbe. "Kalman and Bayesian Filters in Python". [Online] Available: https://nbviewer.jupyter.org/github/rlabbe/Kalman-and-Bayesian-Filters-in-Python/blob/master/table_of_contents.ipynb. [Accessed: 17 June 2021]

5. Mathias Hartwig. "Autonomous Driving". 2020. [Online] Available: https://www.bmw.com/en/automotive-life/autonomous-driving.html. [Accessed: 17 June 2021]

6. K. Min, S. Han, D. Lee, D. Choi, K. Sung and J. Choi, "SAE Level 3 Autonomous Driving Technology of the ETRI," 2019 International Conference on Information and Communication Technology Convergence (ICTC), 2019, pp. 464-466, doi: 10.1109/ICTC46691.2019.8939765.

7. Joel Gibson. "Six Ways Autonomous Driving is Relying on Precise Positioning". November 2019. [Online] Available: https://www.wardsauto.com/industry-voices/six-ways-autonomous-driving-relying-precise-positioning. [Accessed: 17 June 2021]

8. M. L. Cherif, J. Leclère and R. J. Landry, "Loosely coupled GPS/INS integration with snap to road for low-cost land vehicle navigation: EKF-STR for low-cost applications," 2018 IEEE/ION Position, Location and Navigation Symposium (PLANS), 2018, pp. 275-282, doi: 10.1109/PLANS.2018.8373391.

9. K. Takeyama, Y. Kojima and E. Teramoto, "Trajectory estimation improvement based on time-series constraint of GPS Doppler and INS in urban areas," Proceedings of the 2012 IEEE/ION Position, Location and Navigation Symposium, 2012, pp. 700-705, doi: 10.1109/PLANS.2012.6236946.

10. Kian Meng Tan and Choi Look Law, "GPS and UWB Integration for indoor positioning," 2007 6th International Conference on Information, Communications & Signal Processing, 2007, pp. 1-5, doi: 10.1109/ICICS.2007.4449630.

11. F. Wu and Z. Liu, "Research on UWB / IMU Fusion Positioning Technology in Mine," 2020 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS), 2020, pp. 934-937, doi: 10.1109/ICITBS49701.2020.00207.

12. G. Schroeer, "A Real-Time UWB Multi-Channel Indoor Positioning System for Industrial Scenarios," 2018 International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2018, pp. 1-5, doi: 10.1109/IPIN.2018.8533792.

13. Catany Isern, M. (2015, August). Real-Time Robust Loosely-Coupled GPS-Aided PDR (Projecte/Treball Final de Carrera). UPC, Escola Tècnica Superior d'Enginyeria de Telecomunicació de Barcelona, Departament de Teoria del Senyal i Comunicacions. Retrieved from http://hdl.handle.net/2117/79244

14. X. Chen, J. Yu and M. Gu, "Study on tightly-coupled GPS/SINS integrated navigation system by using software GPS receiver," 2011 IEEE International Instrumentation and Measurement Technology Conference, 2011, pp. 1-4, doi: 10.1109/IMTC.2011.5944155.

15. Jupyter Project. https://jupyter.org/

16. Python Package Index. https://pypi.org/

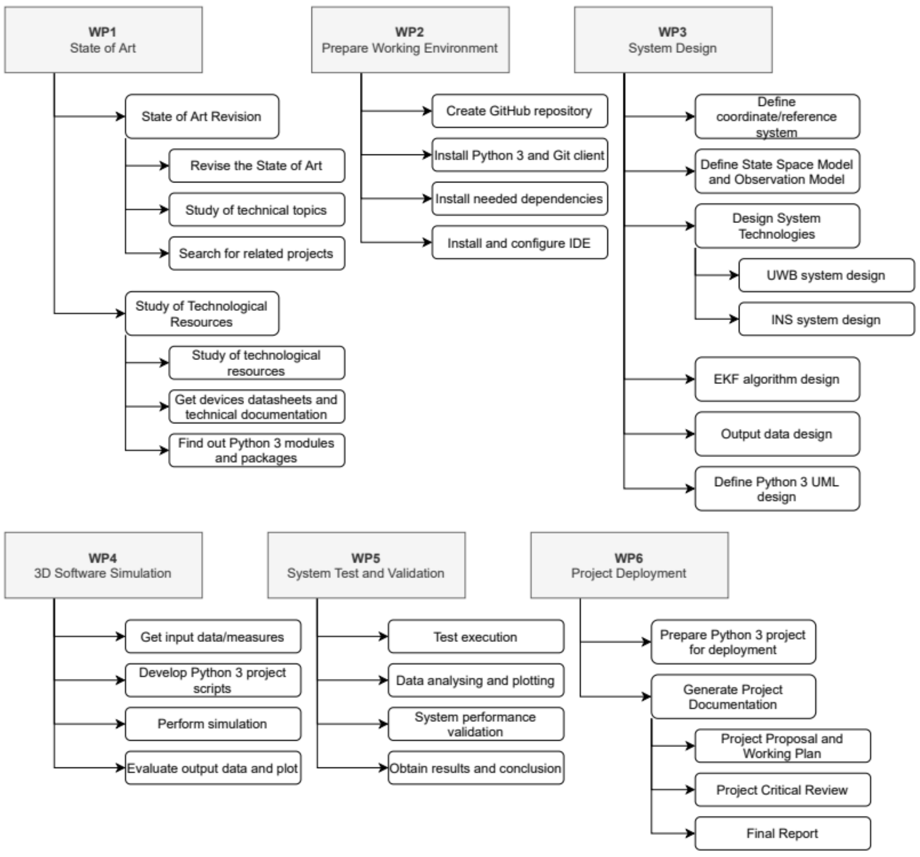
17. in Circuit. Radino32 DW1000. https://wiki.in-circuit.de/index.php5?title=radino32_DW1000

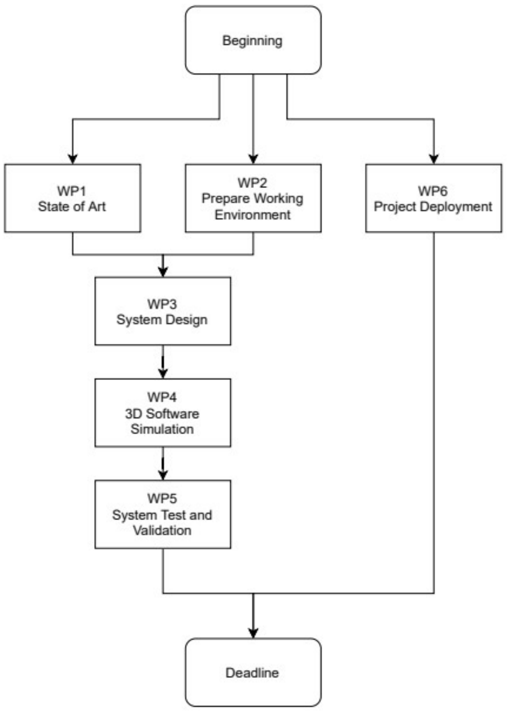
18. GY-521-MPU6050. https://www.addicore.com/GY-521-MPU6050-p/170.htm

19. MPU-6050 https://invensense.tdk.com/products/motion-tracking/6-axis/mpu-6050/

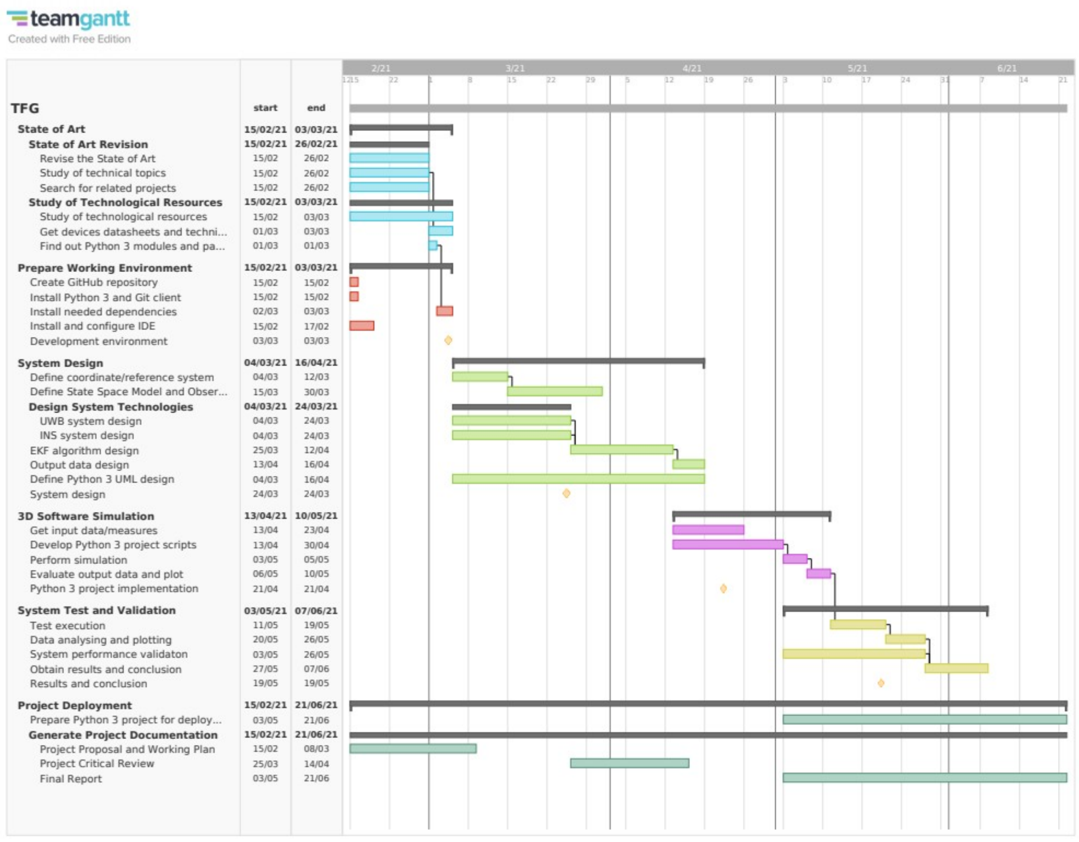
Appendices:

A1. Work Breakdown Structure and Work Packages relationship


WP1
State of Art
State of Art Revision
Revise the State of Art
Study of technical topics
Search for related projects
Study of Technological Resources
Study of technological resources
Get devices datasheets and technical documentation
Find out Python 3 modules and packages
WP2
Prepare Working Environment
Create GitHub repository
Install Python 3 and Git client
Install needed dependencies
Install and configure IDE
WP3
System Design
Define coordinate/reference system
Define State Space Model and Observation Model
Design System Technologies
UWB system design
INS system design
EKF algorithm design
Output data design
Define Python 3 UML design
WP4
3D Software Simulation
Get input data/measures
Develop Python 3 project scripts
Perform simulation
Evaluate output data and plot
WP5
System Test and Validation
Test execution
Data analysing and plotting
System performance validation
Obtain results and conclusion
WP6
Project Deployment
Prepare Python 3 project for deployment
Generate Project Documentation
Project Proposal and Working Plan
Project Critical Review
Final Report



A2. Gantt diagram



TFG	start	end
State of Art	**15/02/21**	**03/03/21**
State of Art Revision	**15/02/21**	**26/02/21**
Revise the State of Art	15/02	26/02
Study of technical topics	15/02	26/02
Search for related projects	15/02	26/02
Study of Technological Resources	**15/02/21**	**03/03/21**
Study of technological resources	15/02	03/03
Get devices datasheets and techni...	01/03	03/03
Find out Python 3 modules and pa...	01/03	01/03
Prepare Working Environment	**15/02/21**	**03/03/21**
Create GitHub repository	15/02	15/02
Install Python 3 and Git client	15/02	15/02
Install needed dependencies	02/03	03/03
Install and configure IDE	15/02	17/02
Development environment	03/03	03/03
System Design	**04/03/21**	**16/04/21**
Define coordinate/reference system	04/03	12/03
Define State Space Model and Obser...	15/03	30/03
Design System Technologies	**04/03/21**	**24/03/21**
UWB system design	04/03	24/03
INS system design	04/03	24/03
EKF algorithm design	25/03	12/04
Output data design	13/04	16/04
Define Python 3 UML design	04/03	16/04
System design	24/03	24/03
3D Software Simulation	**13/04/21**	**10/05/21**
Get input data/measures	13/04	23/04
Develop Python 3 project scripts	13/04	30/04
Perform simulation	03/05	05/05
Evaluate output data and plot	06/05	10/05
Python 3 project implementation	21/04	21/04
System Test and Validation	**03/05/21**	**07/06/21**
Test execution	11/05	19/05
Data analysing and plotting	20/05	26/05
System performance validaton	03/05	26/05
Obtain results and conclusion	27/05	07/06
Results and conclusion	19/05	19/05
Project Deployment	**15/02/21**	**21/06/21**
Prepare Python 3 project for deploy...	03/05	21/06
Generate Project Documentation	**15/02/21**	**21/06/21**
Project Proposal and Working Plan	15/02	08/03
Project Critical Review	25/03	14/04
Final Report	03/05	21/06

A3. Kalman Filtering design parameters

Predict step

State Vector	$\boldsymbol{x}$	Contains state variables to estimate from the dynamic system	$\boldsymbol{x}=[x_1,x_2,\dots x_{dimx}]^T$
State Covariance Matrix	$\boldsymbol{P}$	Defines state variables' covariance. It can be translated as the belief in the state estimate	$\boldsymbol{P}=\begin{bmatrix} var(x_1) & cov(x_1,x_2) & \dots & cov(x_1,x_N) \\ cov(x_2,x_1) & var(x_2) & \dots & cov(x_2,x_N) \\ \dots & \dots & \dots & \dots \\ cov(x_N,x_1) & cov(x_N,x_2) & \dots & var(x_N) \end{bmatrix}$
Control Input Function	$\boldsymbol{B}$	Transforms control input values in a contribution to state variables	$\boldsymbol{B}$ (must be designed according to Control Input Model)
Control Input Vector	$\boldsymbol{u}$	Contains control input variables	$\boldsymbol{u}=[u_1,u_2,\dots u_{dimu}]^T$
State Transition Function	$\boldsymbol{F}$	Models the state propagation over time	$\boldsymbol{F}$ (must be designed according to Process Model)
Process Noise Matrix	$\boldsymbol{Q}$	Models Process Model's noise	$\boldsymbol{Q}$ (must be designed according to Process Model)

Update step

Measurement Vector	$\boldsymbol{z}$	Contains measurement variables observed from the dynamic system	$\boldsymbol{z}=[z_1,z_2,\dots,z_{dimz}]^T$
Measurement Noise Matrix	$\boldsymbol{R}$	Contains measurement variables' variances. It can be translated as the belief in the measurements	$\boldsymbol{R}=\begin{bmatrix} var(z_1) & 0 & \dots & 0 \\ 0 & var(z_2) & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & var(z_{dimz}) \end{bmatrix}$
Measurement function	$\boldsymbol{H}$	Transforms state variables into the measurement domain	$\boldsymbol{H}$ (must be designed according to Measurement Model)

A4. Kalman Filter algorithm

Initialization

1. Initialize the state of the filter ($\boldsymbol{x}$)
2. Initialize our belief in the state ($\boldsymbol{P}$)

Predict

1. Use Process Model to predict state at the next time step (prior, $\bar{\boldsymbol{x}}$)

$$\bar{\boldsymbol{x}} = \boldsymbol{F}\,\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \qquad \text{A4.1}$$

2. Adjust belief to account for the uncertainty in prediction

$$\bar{\boldsymbol{P}} = \boldsymbol{F}\,\boldsymbol{P}\,\boldsymbol{F}^{T} + \boldsymbol{Q} \qquad \text{A4.2}$$

Update

1. Get a measurement ($\boldsymbol{z}$) and associated belief about its accuracy ($\boldsymbol{R}$)
2. Compute residual ($\boldsymbol{y}$) between estimated state (prior) and measurement

$$\boldsymbol{y} = \boldsymbol{z} - \boldsymbol{H}\,\bar{\boldsymbol{x}} \qquad \text{A4.3}$$

3. Compute scaling factor (Kalman Gain, $\boldsymbol{K}$) based on whether the measurement or prediction is more accurate

$$\boldsymbol{K} = \bar{\boldsymbol{P}}\,\boldsymbol{H}^{T}\left(\boldsymbol{H}\,\bar{\boldsymbol{P}}\,\boldsymbol{H}^{T} + \boldsymbol{R}\right)^{-1} \qquad \text{A4.4}$$

4. Set state between the prediction and measurement based on scaling factor

$$\boldsymbol{x} = \bar{\boldsymbol{x}} + \boldsymbol{K}\,\boldsymbol{y} \qquad \text{A4.5}$$

5. Update belief in the state based on how certain we are in the measurement

$$\boldsymbol{P} = \left(\boldsymbol{I} - \boldsymbol{K}\,\boldsymbol{H}\right)\bar{\boldsymbol{P}} \qquad \text{A4.6}$$

A5. Extended Kalman Filter algorithm

Initialization

1. Initialize the state of the filter ($\boldsymbol{x}$)
2. Initialize our belief in the state ($\boldsymbol{P}$)

Predict

1. Use Process Model to predict state at the next time step (prior, $\bar{\boldsymbol{x}}$)

$$\bar{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \tag{A5.1}$$

2. Adjust belief to account for the uncertainty in prediction

$$\bar{\boldsymbol{P}} = \boldsymbol{F}\,\boldsymbol{P}\,\boldsymbol{F}^T + \boldsymbol{Q} \tag{A5.2}$$

Update

1. Get a measurement ($\boldsymbol{z}$) and associated belief about its accuracy ($\boldsymbol{R}$)
2. Compute residual ($\boldsymbol{y}$) between estimated state (prior) and measurement

$$\boldsymbol{y} = \boldsymbol{z} - \boldsymbol{h}(\bar{\boldsymbol{x}}) \tag{A5.3}$$

3. Compute scaling factor (Kalman Gain, $\boldsymbol{K}$) based on whether the measurement or prediction is more accurate

$$\boldsymbol{K} = \bar{\boldsymbol{P}}\,\boldsymbol{H}^T \left(\boldsymbol{H}\,\bar{\boldsymbol{P}}\,\boldsymbol{H}^T + \boldsymbol{R}\right)^{-1} \tag{A5.4}$$

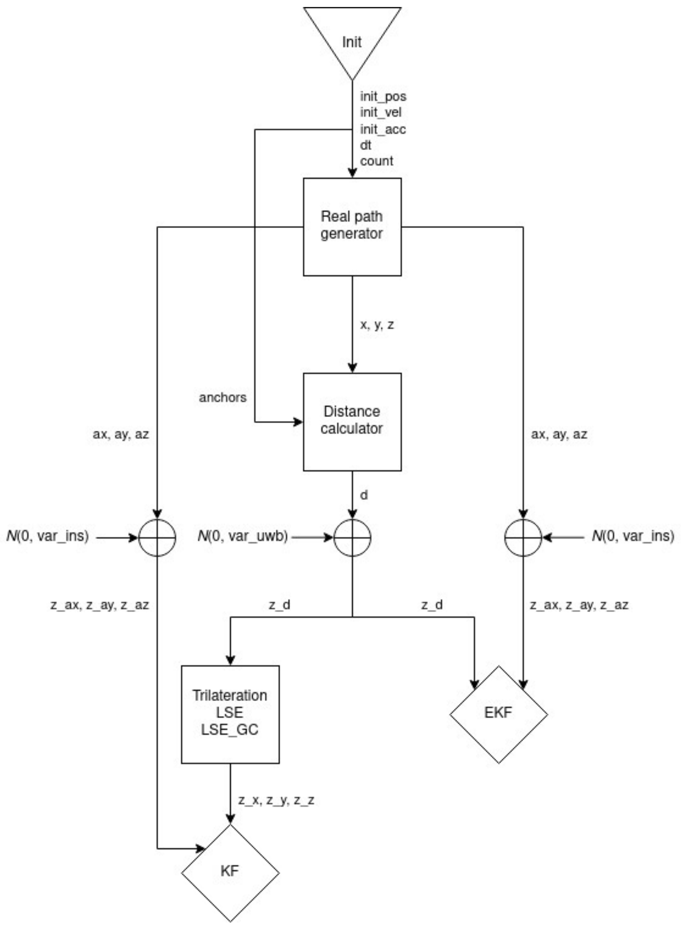
4. Set state between the prediction and measurement based on scaling factor

$$\boldsymbol{x} = \bar{\boldsymbol{x}} + \boldsymbol{K}\,\boldsymbol{y} \tag{A5.5}$$

5. Update belief in the state based on how certain we are in the measurement

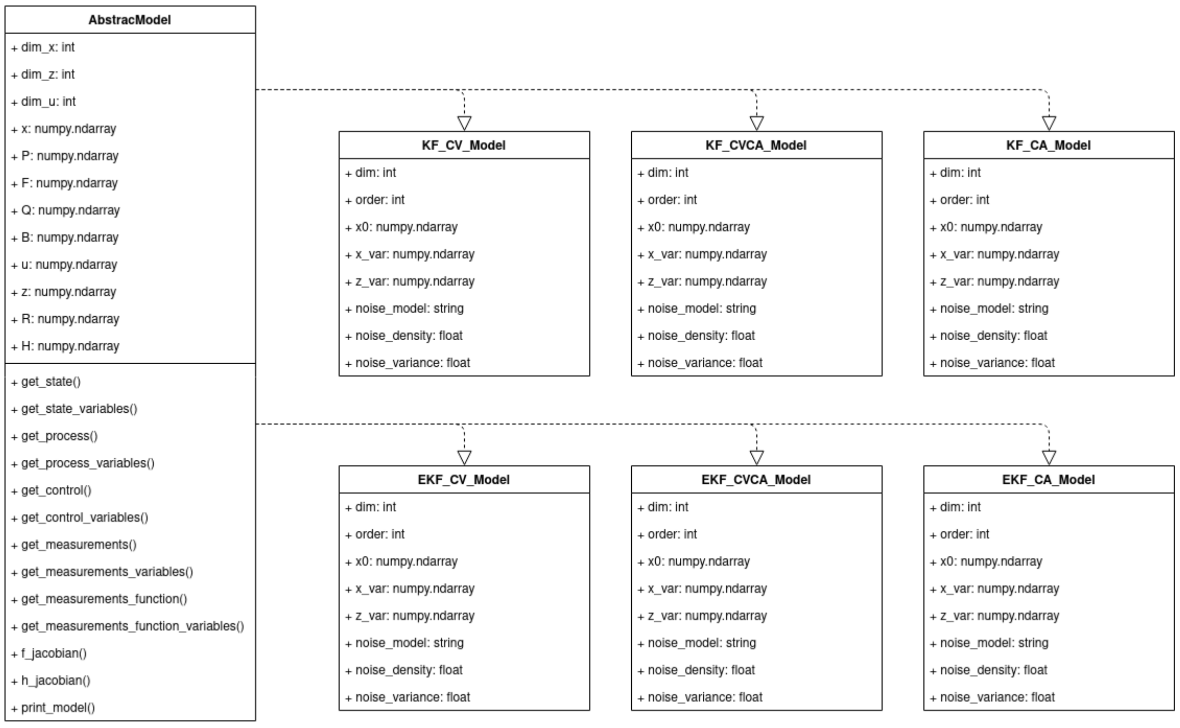
$$\boldsymbol{P} = (\boldsymbol{I} - \boldsymbol{K}\,\boldsymbol{H})\,\bar{\boldsymbol{P}} \tag{A5.6}$$

A6. Measurements generator flowchart and configurable parameters table

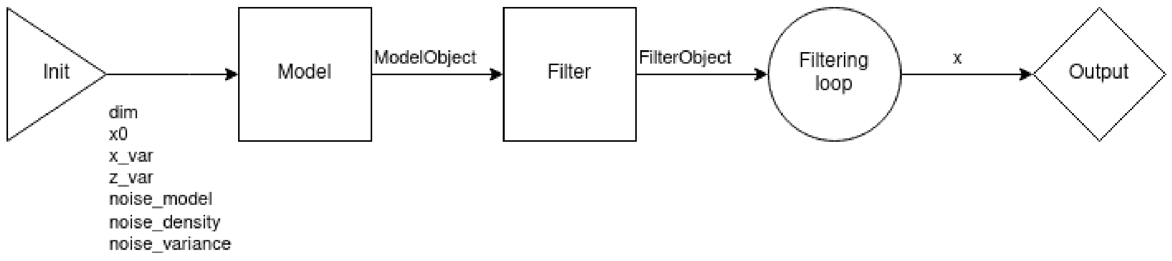


Parameter	Type	Meaning	Default value
dim	int	Number of spatial dimensions	
init_pos	[], int, float	Initial positioning coordinates	
init_vel	[], int, float	Initial velocity on each axis	
init_acc	[], int, float	Initial acceleration on each axis	
var_uwb	float	Variance of measures from UWB ranging	
var_ins	float	Variance of measures from INS	
anc	[]	Anchors position coordinates	
dt	float	Time differential	1
count	int	Number of samples to generate	100

A7. UML diagram for filtering models



A8. Filtering algorithm in Python and configurable parameters list



Parameter	Type	Meaning	Default value
x0	[]	Initial State Vector	
x_var	[]	State variables' variances	
z_var	[]	Measurements' variances	
model	{'CV', 'CVCA', 'CA'}	Filtering model to use	
noise_model	{'ContinuousModel', 'DiscreteModel'}	Process Noise Model to use	
noise_density	float	Process Noise spectral density for Continuous Noise Model	1
noise_variance	float	Process Noise variance for Discrete Noise Model	1

A9. Parameter configuration for each study case

The configurable parameters for the generator and the filter algorithm for each study case.

Ideal case

Table A9.1: Generator parameters for ideal case

std_uwb	0
std_ins	0

Table A9.2: Filter parameters for ideal case

	CV	CVCA	CA
x0	[0, 1.8, 0, 0.05, 1, 0]	[0, 1.8, 0, 0.05, 1, 0]	[0, 1.8, 0, 0, 0.05, 1, 1, 0, 0]
x_var	[1, 1, 1, 1, 1, 1]	[1, 1, 1, 1, 1, 1]	[1, 1, 1, 1, 1, 1, 1, 1, 1]
z_var	KF: [0.5^2, 0.5^2, 0.5^2]	KF: [0.5^2, 0.5^2, 0.5^2]	KF: [0.5^2, 0.1^2, 0.5^2, 0.1^2, 0.5^2, 0.1^2]
	EKF: [0.1^2]	EKF: [0.1^2]	EKF: [0.1^2, 0.1^2, 0.1^2, 0.1^2]
noise_model	Discrete Model	Discrete Model	Discrete Model
noise_variance	0.1	0.1	0.1

Bad initialization

Table A9.3: Generator parameters for bad initialization

std_uwb	0.1
std_ins	0.1

Table A9.4: Filter parameters for bad initialization

	CV	CVCA	CA
x0	[120, 0, -50, 0, 7, 0]	[120, 0, -50, 0, 7, 0]	[120, 0, 0, -50, 0, 0, 7, 0, 0]
x_var	[1, 1, 1, 1, 1, 1]	[1, 1, 1, 1, 1, 1]	[1, 1, 1, 1, 1, 1, 1, 1, 1]
z_var	KF: [0.5^2, 0.5^2, 0.5^2]	KF: [0.5^2, 0.5^2, 0.5^2]	KF: [0.5^2, 0.1^2, 0.5^2, 0.1^2, 0.5^2, 0.1^2]
	EKF: [0.1^2]	EKF: [0.1^2]	EKF: [0.1^2, 0.1^2, 0.1^2, 0.1^2]
noise_model	Discrete Model	Discrete Model	Discrete Model
noise_variance	0.1	0.1	0.1

Optimal design

Table A9.5: Generator parameters for optimal design

std_uwb	0.1
std_ins	0.1

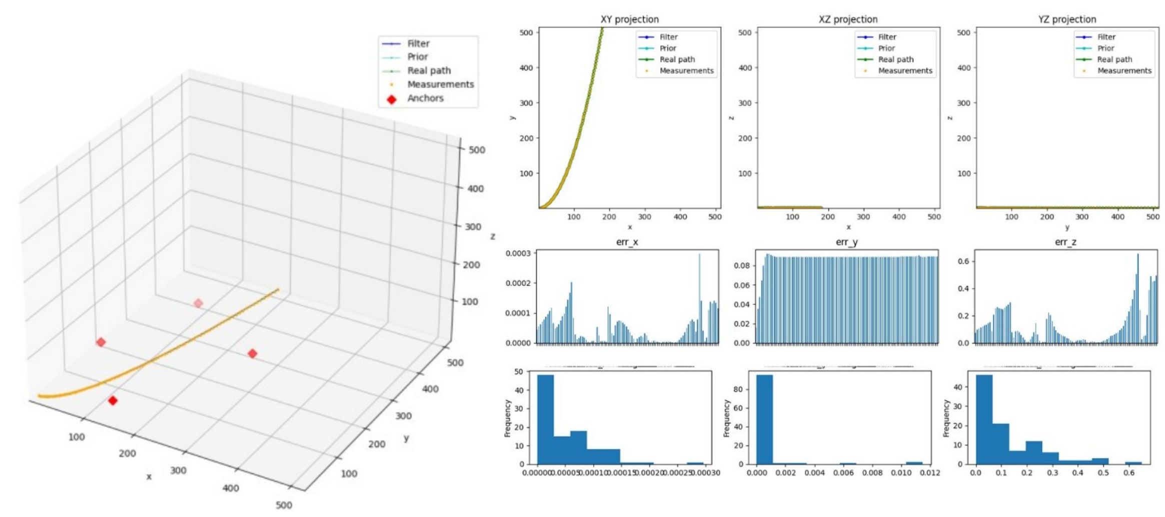
Table A9.6: Filter parameters for optimal design

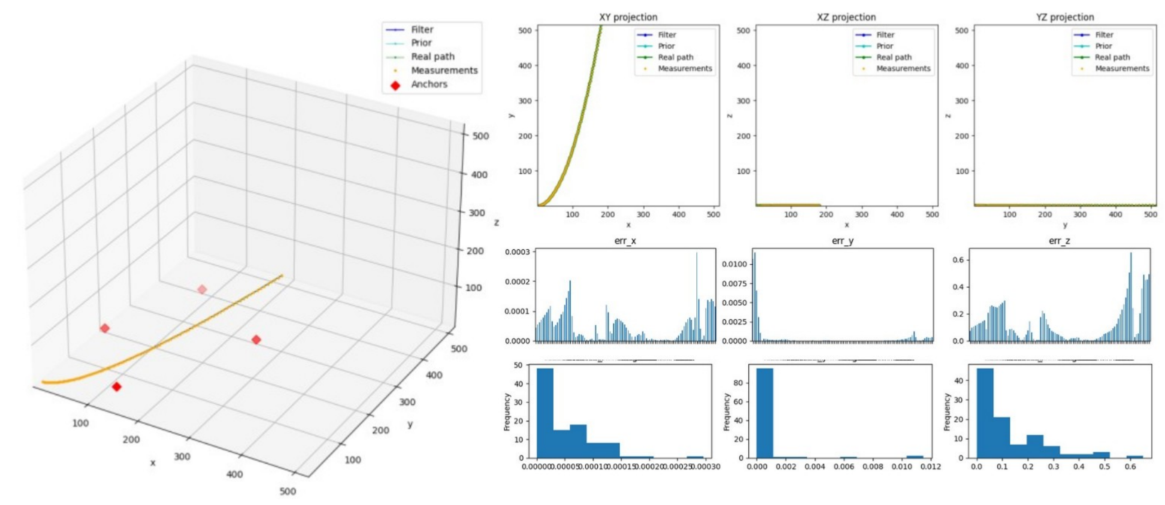
	CV	**CVCA**	**CA**
x0	[0, 0, 0, 0, 0, 0]	[0, 0, 0, 0, 0, 0]	[0, 0, 0, 0, 0, 0, 0, 0, 0]
x_var	[20^2, 10^2, 20^2, 10^2, 20^2, 10^2]	[20^2, 10^2, 20^2, 10^2, 20^2, 10^2]	[20^2, 10^2, 3^2, 20^2, 10^2, 3^2, 20^2, 10^2, 3^2]
z_var	KF: [0.5^2, 0.5^2, 0.5^2]	KF: [0.5^2, 0.5^2, 0.5^2]	KF: [0.5^2, 0.1^2, 0.5^2, 0.1^2, 0.5^2, 0.1^2]
	EKF: [0.1^2]	EKF: [0.1^2]	EKF: [0.1^2, 0.1^2, 0.1^2, 0.1^2]
noise_model	Discrete Model	Discrete Model	Discrete Model
noise_variance	0.1	0.1	0.1

A10. Simulation outputs

Ideal case

Figure A10.1: KF CV (a), CVCA (b) and CA (c) plots for ideal case simulation





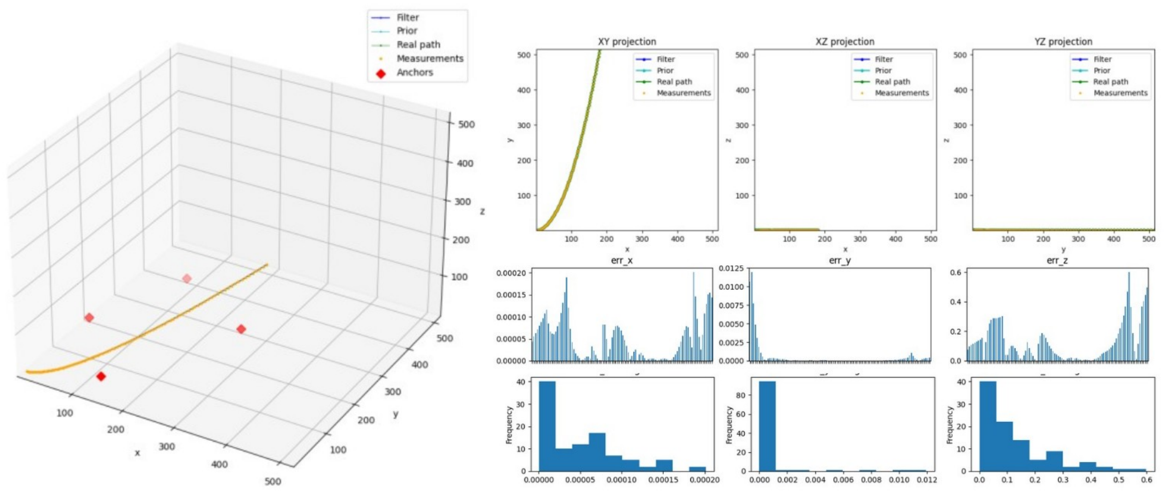
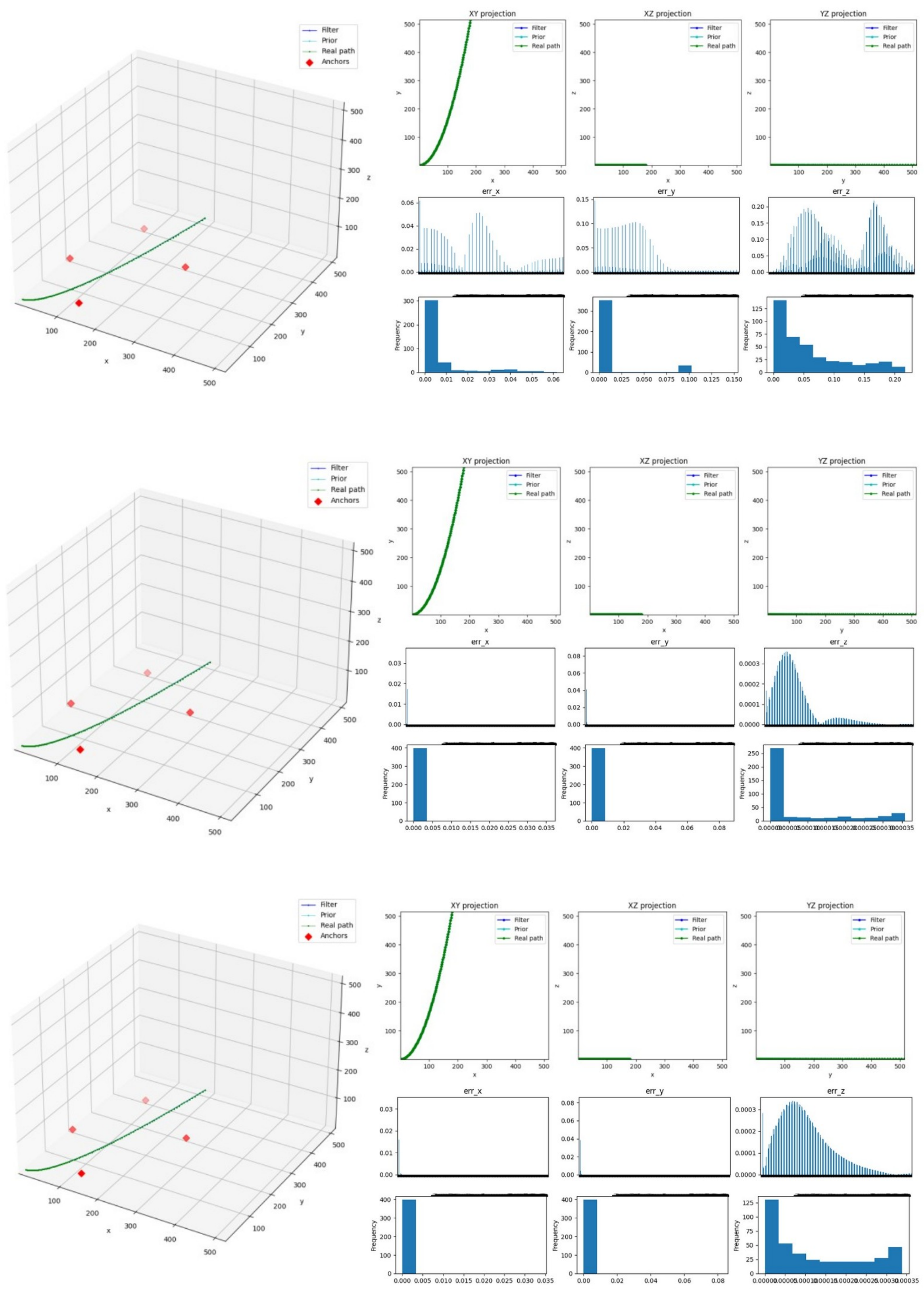
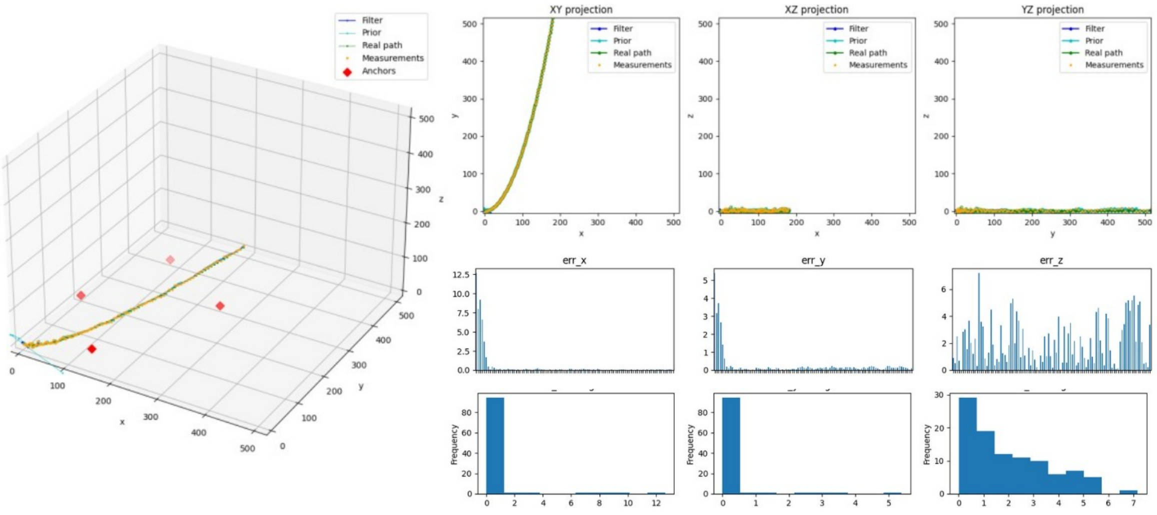


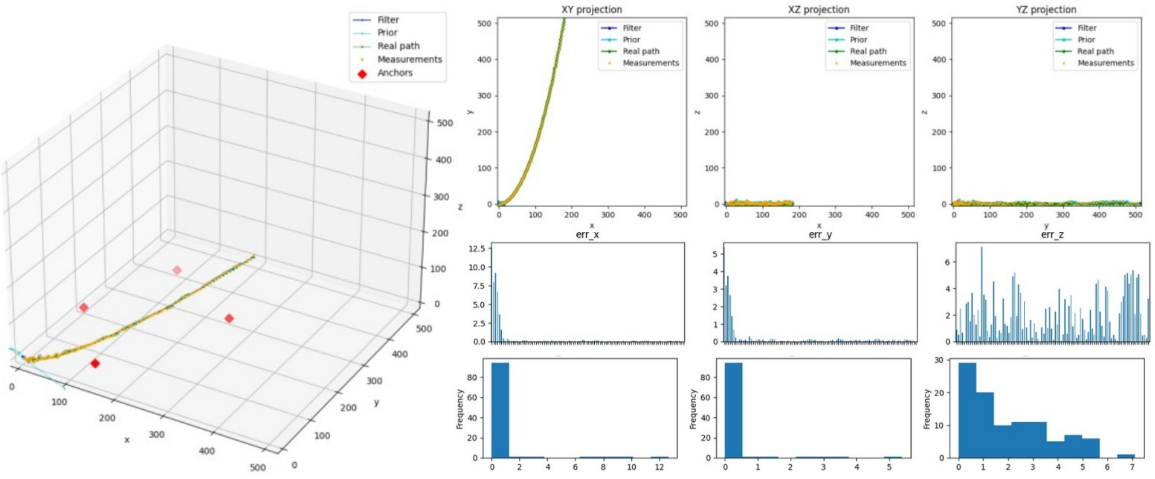
Figure A10.2: EKF CV (a), CVCA (b) and CA (c) plots for ideal case simulation



Bad initialization

Figure A10.3: KF CV (a), CVCA (b) and CA (c) plots for bad initialization case simulation





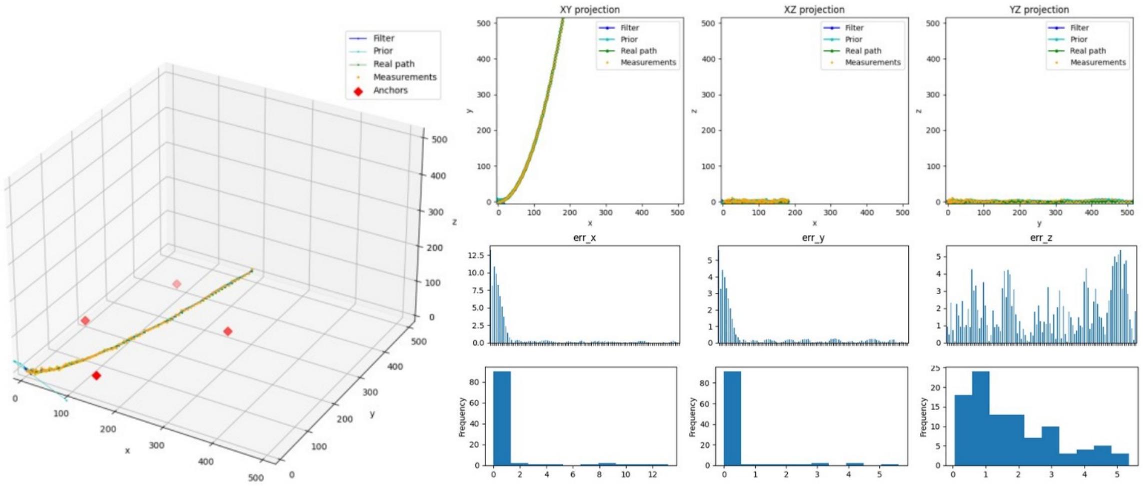
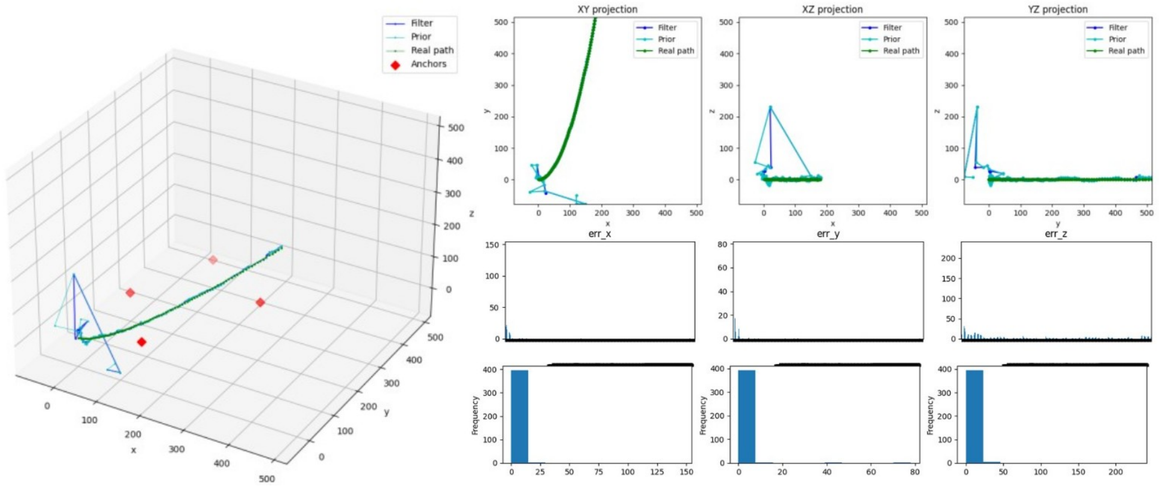
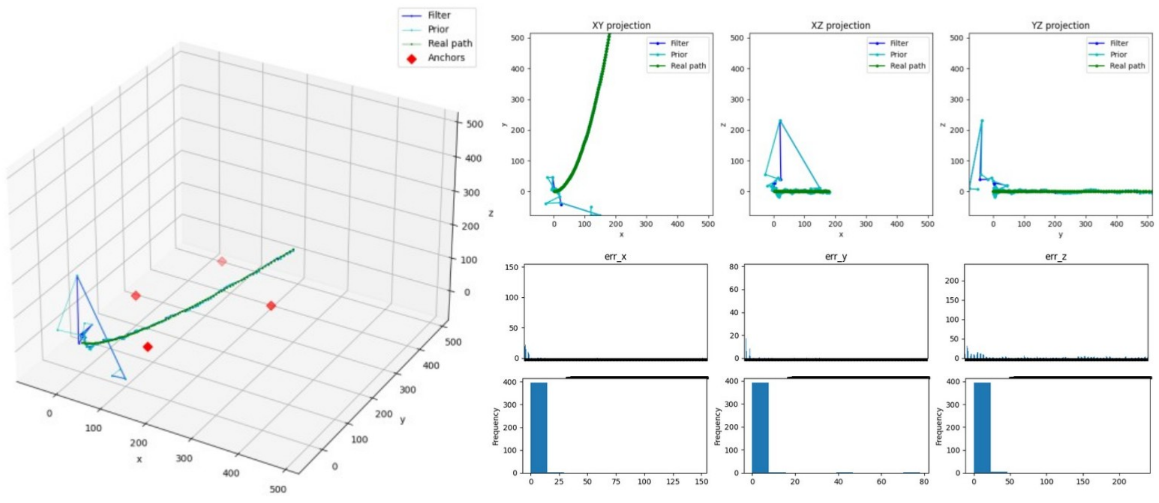
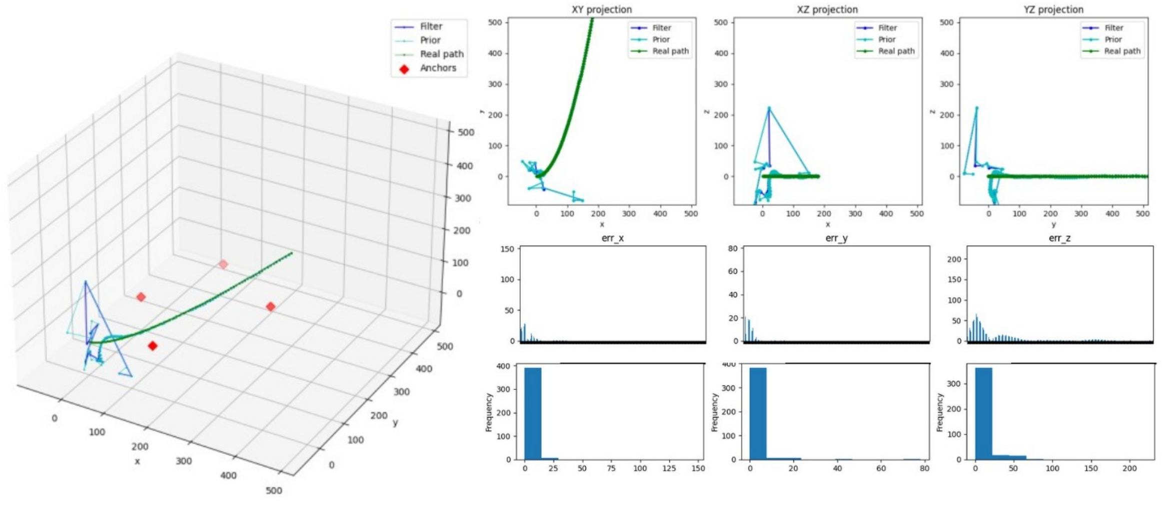


Figure A10.4: EKF CV (a), CVCA (b) and CA (c) plots for bad initialization case simulation







Optimal design

Figure A10.5: KF CV (a), CVCA (b) and CA (c) plots for optimal design case simulation

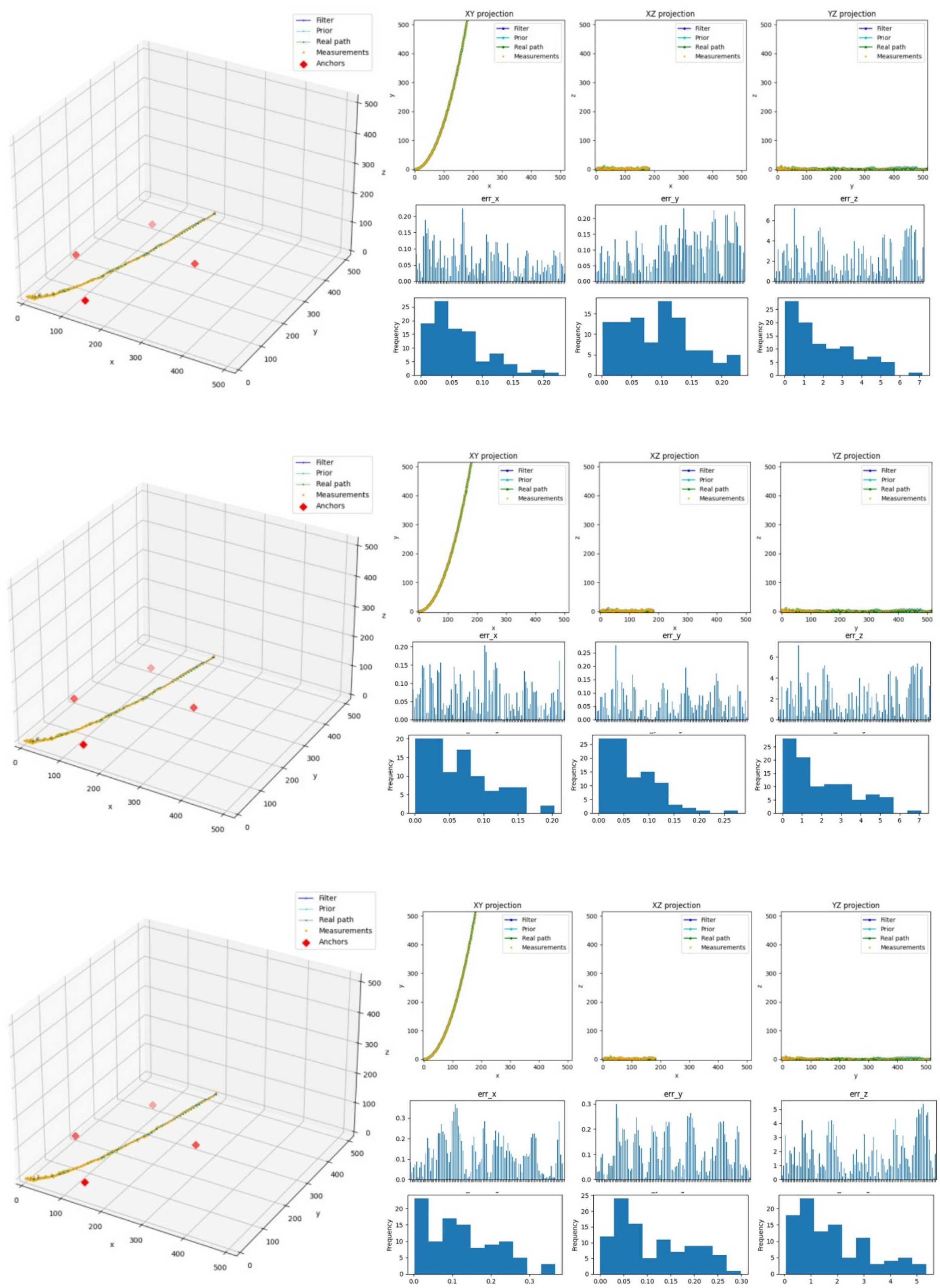
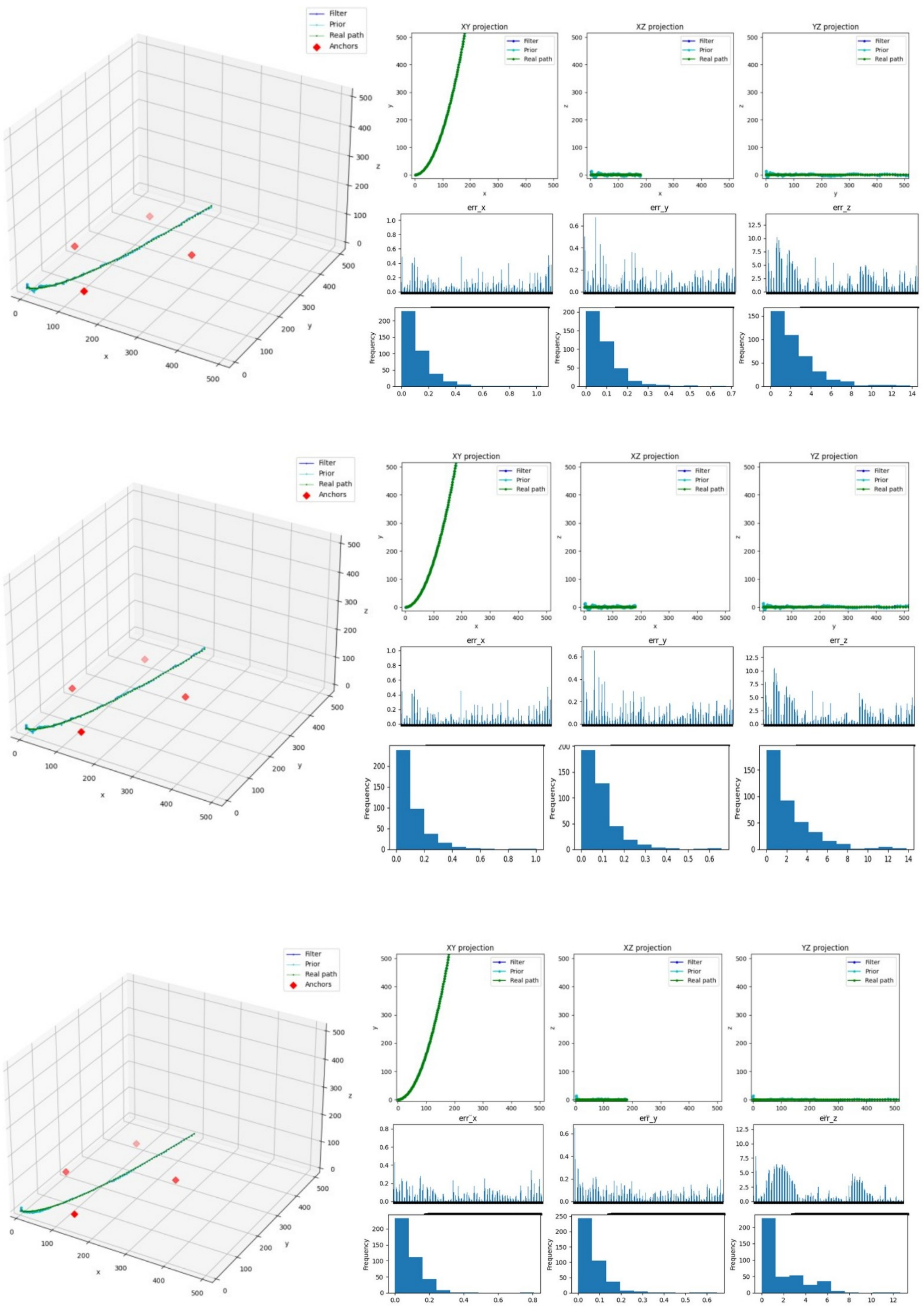
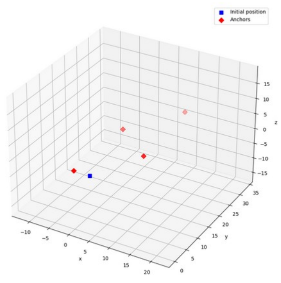


Figure A10.6: EKF CV (a), CVCA (b) and CA (c) plots for optimal design case simulation

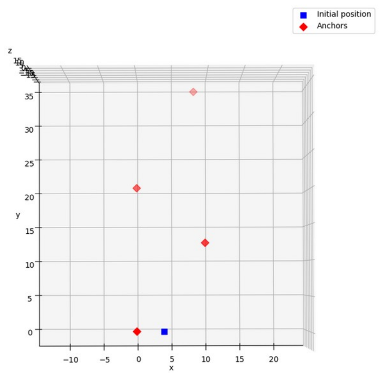


A11. Real test (a). Straight way scenario setup and filter design parameter evaluation

Figure A11.1: Anchors locations in 3D plot (a) and aerial view (b) for the straight-way scenario



(a)



(b)

Figure A11.2: Real path plotting for the straight-way scenario

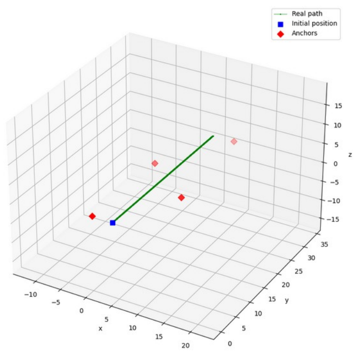


Table A11.1: Anchors positioning coordinates for the straight-way scenario

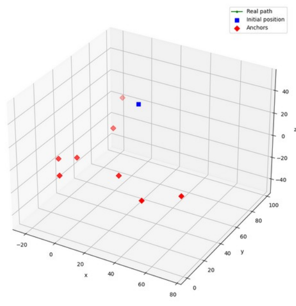
Anchor ID	Location
6ECD	(0.06, 0, 1.1)
7331	(9.7, 12.5, 1.1)
6EC7	(0.06, 20.24, 1.1)
1111	(8.10, 33.86, 1.1)

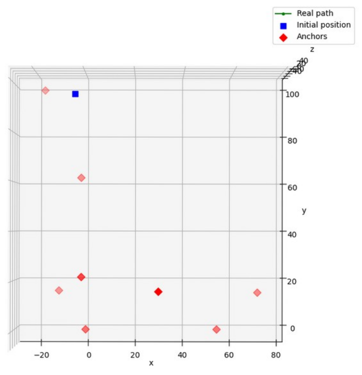
Table A11.2: Filter parameters for the straight-way scenario

	CV	**CVCA**	**CA**
x0	(0, 0, 0, 0, 0, 0)	(0, 0, 0, 0, 0, 0)	(0, 0, 0, 0, 0, 0, 0, 0, 0)
x_var	(20^2, 10^2, 20^2, 10^2, 20^2, 10^2)	(20^2, 10^2, 20^2, 10^2, 20^2, 10^2)	(20^2, 10^2, 3^2, 20^2, 10^2, 3^2, 20^2, 10^2, 3^2)
z_var	KF: (0.5^2, 0.5^2, 0.5^2)	KF: (0.5^2, 0.5^2, 0.5^2)	KF: (0.5^2, 0.1^2, 0.5^2, 0.1^2, 0.5^2, 0.1^2)
	EKF: (0.25^2)	EKF: (0.25^2)	EKF: (0.25^2, 0.1^2, 0.1^2, 0.1^2)
noise_model	Continuous model	Continuous model	Continuous model
noise_density	1	1	1

A12. Real test (b). 90 degree bend scenario setup and filter design parameter evaluation

Figure A12.1: Anchors locations in 3D plot (a) and aerial view (b) for the 90-degree-bend scenario





(a)

(b)

Figure A12.2: Real path plotting for the 90-degree-bend scenario

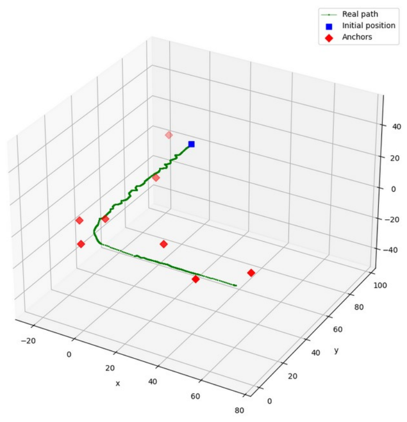


Figure A12.3: Measurements plotting for the 90 degree bend scenario

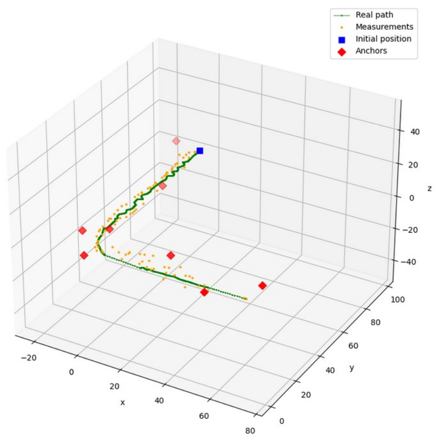


Table A12.1: Anchors positioning coordinates for the 90 degree bend scenario

Anchor ID	Location
6ECD	(29.65, 15.37, 3.46)
6EC7	(-10.86, 15.83, 2.10)
1111	(-1.78, 21.31, 2.86)
7B6F	(0, 0, 2.3)
7BDB	(70.01, 15.02, 2.20)
7331	(53.41, 0, 2.22)
7B30	(-1.78, 61.74, 2.57)
7B22	(-16.5, 97.3, 2.66)

Table A12.2: Filter parameters for the 90 degree bend scenario

	CV	CVCA	CA
x0	(-4.6, 0, 96, 0, 1.8, 0)	(-4.6, 0, 96, 0, 1.8, 0)	(-4.6, 0, 0, 96, 0, 0, 1.8, 0, 0)
x_var	(20^2, 10^2, 20^2, 10^2, 20^2, 10^2)	(20^2, 10^2, 20^2, 10^2, 20^2, 10^2)	(20^2, 10^2, 3^2, 20^2, 10^2, 3^2, 20^2, 10^2, 3^2)
z_var	KF: (0.5^2, 0.5^2, 0.5^2)	KF: (0.5^2, 0.5^2, 0.5^2)	KF: (0.5^2, 0.1^2, 0.5^2, 0.1^2, 0.5^2, 0.1^2)
	EKF: (0.25^2)	EKF: (0.25^2)	EKF: (0.25^2, 0.1^2, 0.1^2, 0.1^2)
noise_model	Continuous model	Continuous model	Continuous model
noise_density	1	1	1

A13. Real scenario test results

(a) Straight-way scenario

Figure A13.1: KF CV (a), CVCA (b) and CA (c) plots for straight-way scenario

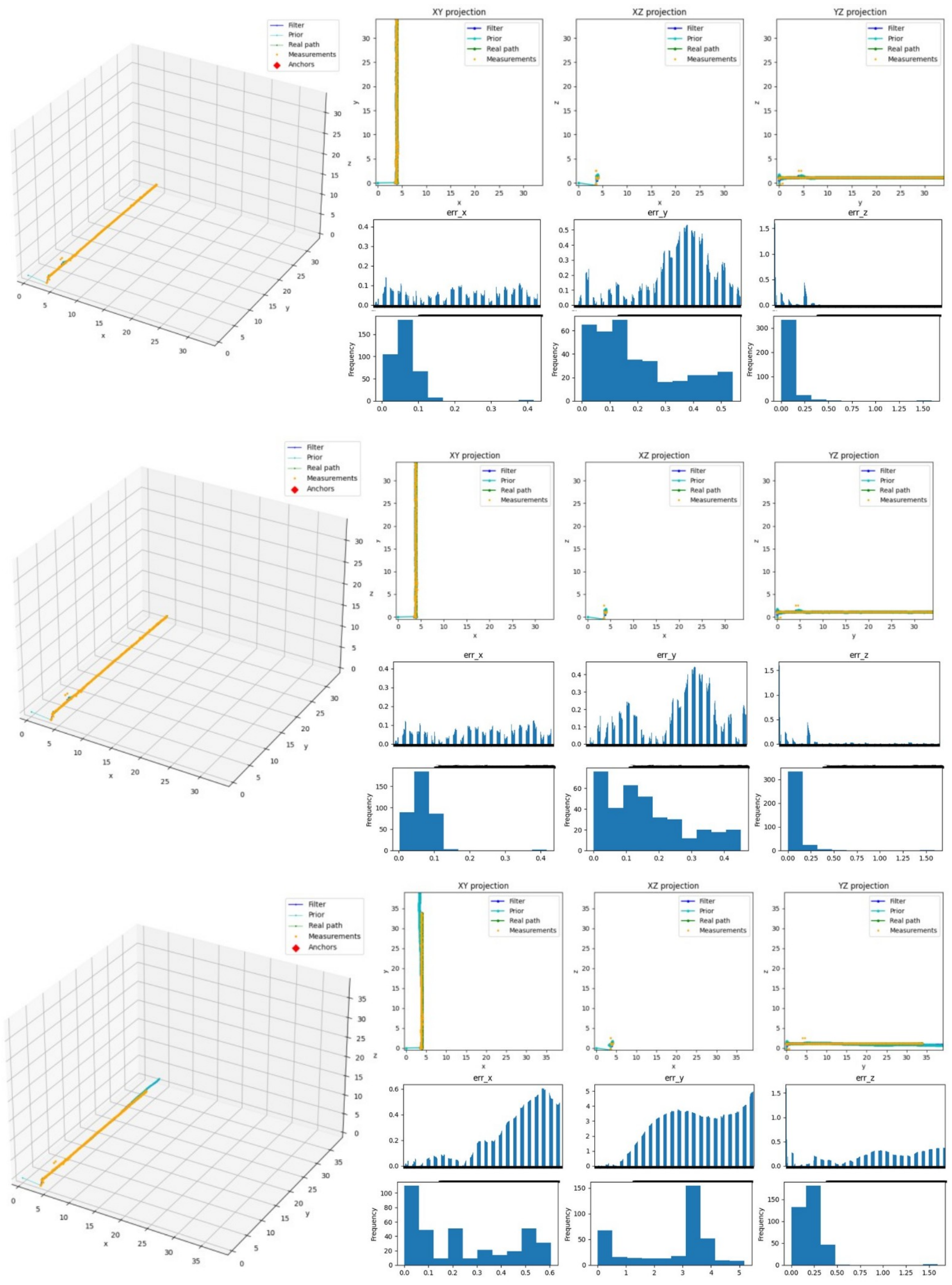
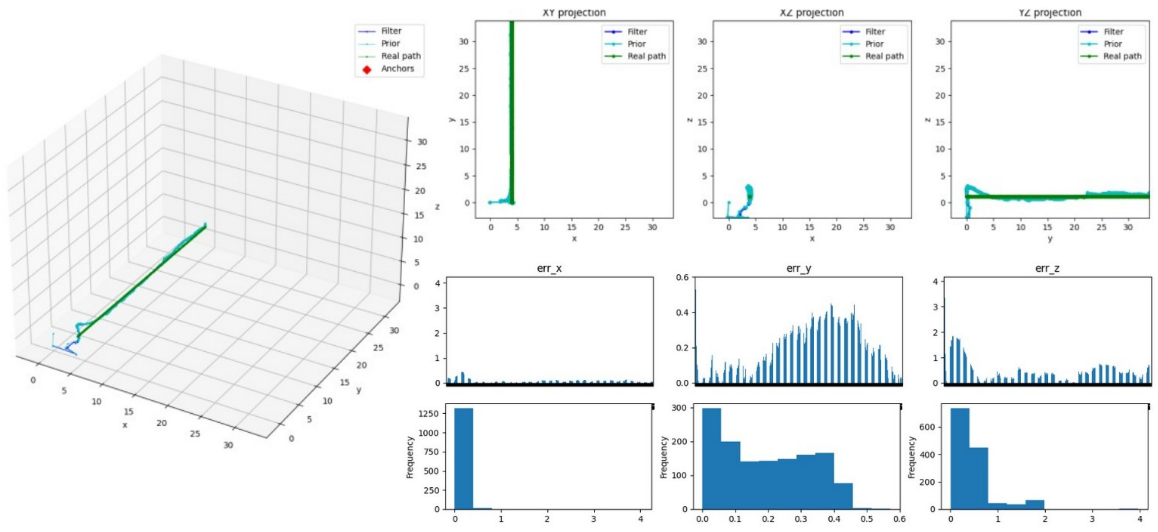
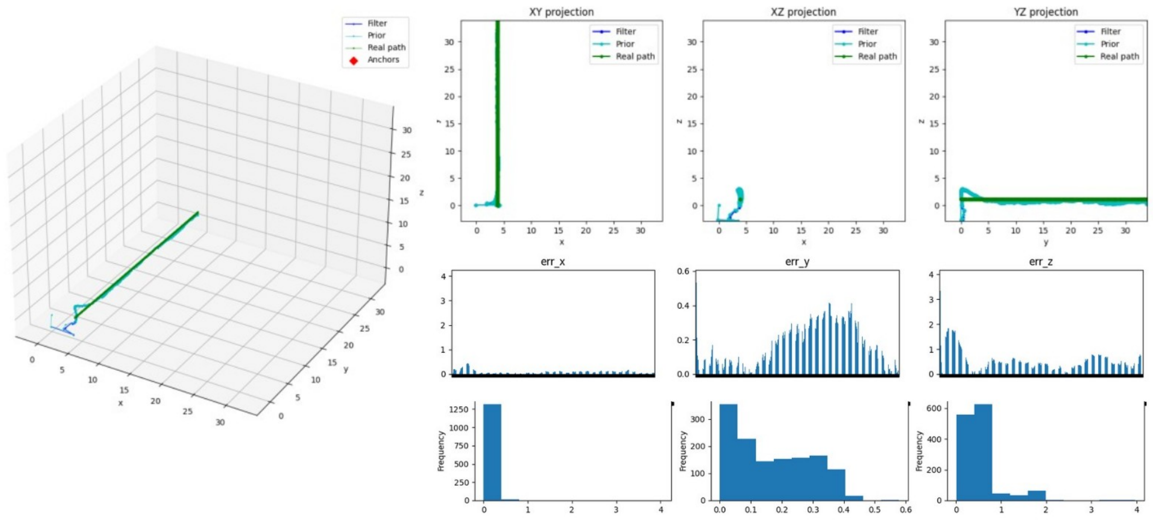
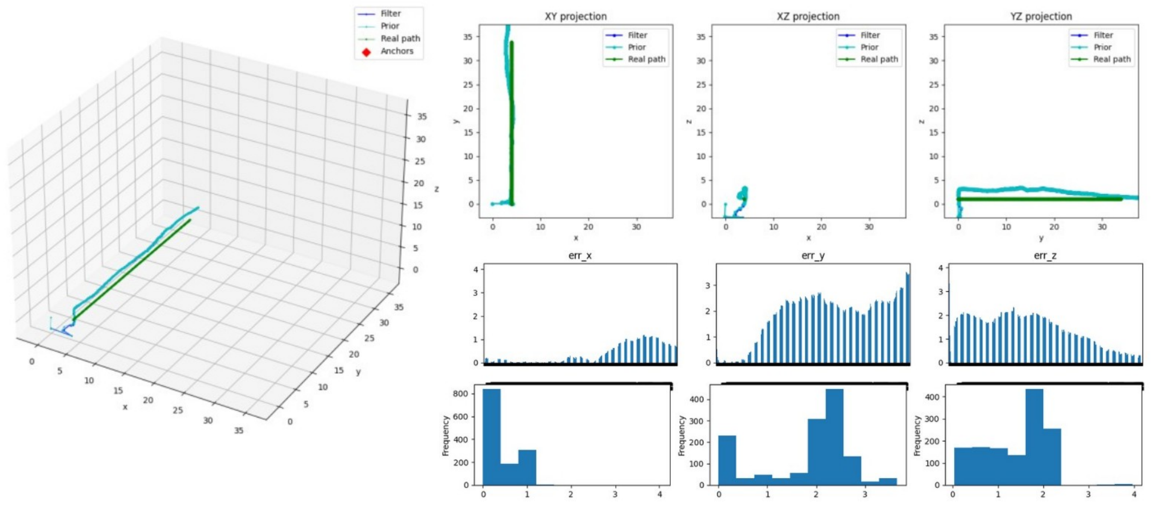


Figure A13.2: EKF CV (a), CVCA (b) and CA (c) plots for straight-way scenario

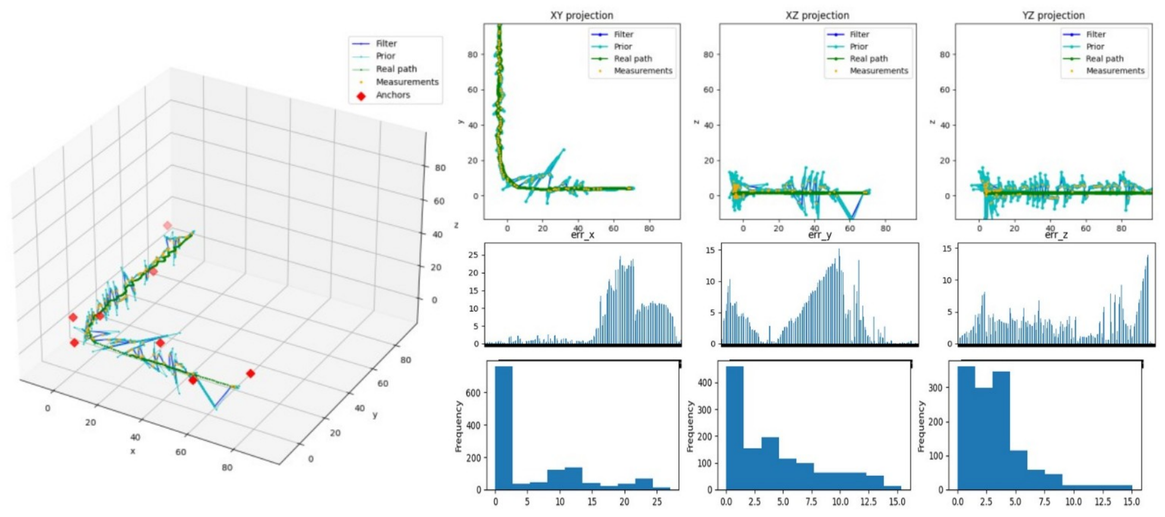


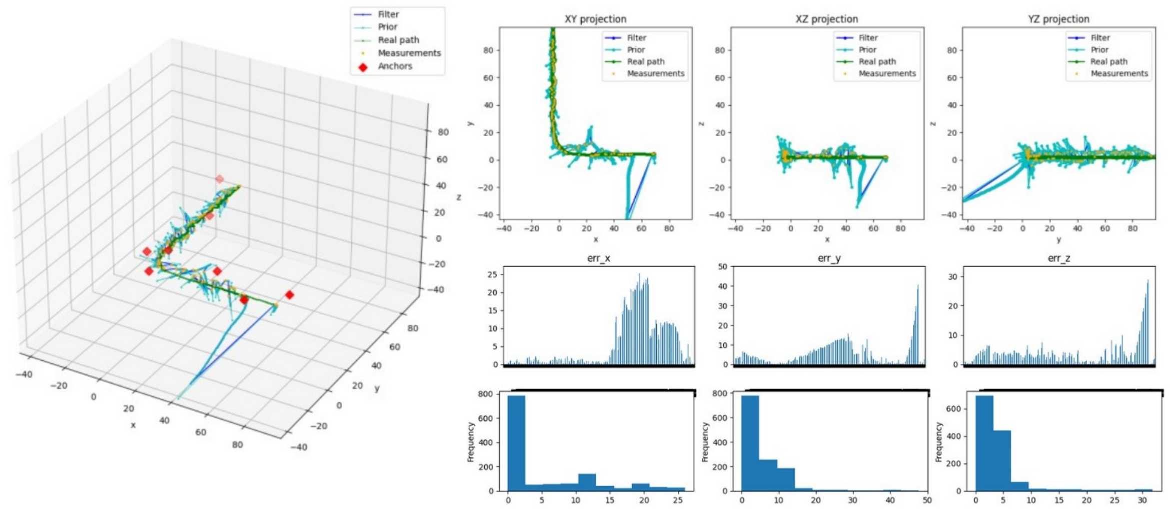




(b) 90-degree-bend scenario

Figure A13.3: KF CV (a), CVCA (b) and CA (c) plots for 90-degree-bend scenario





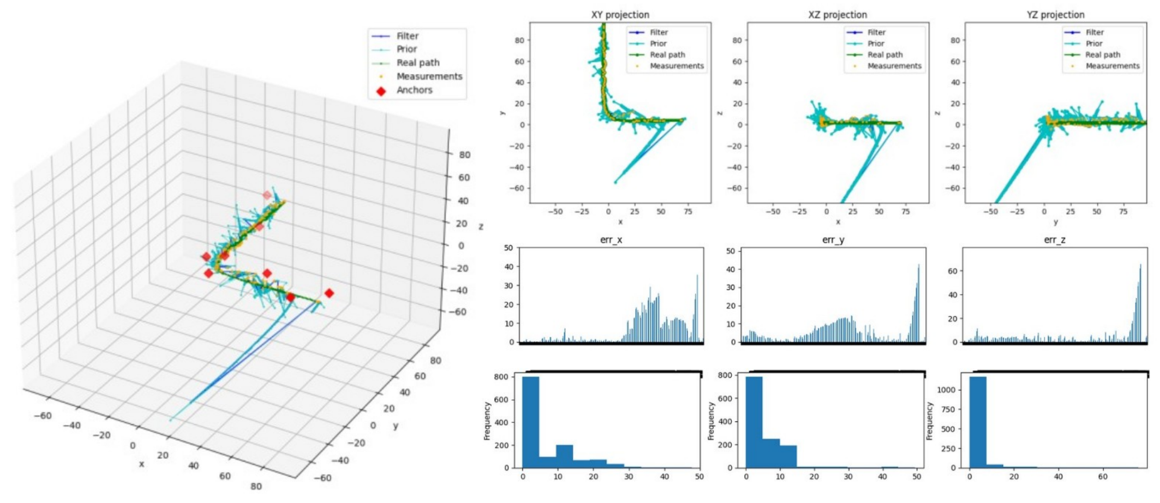
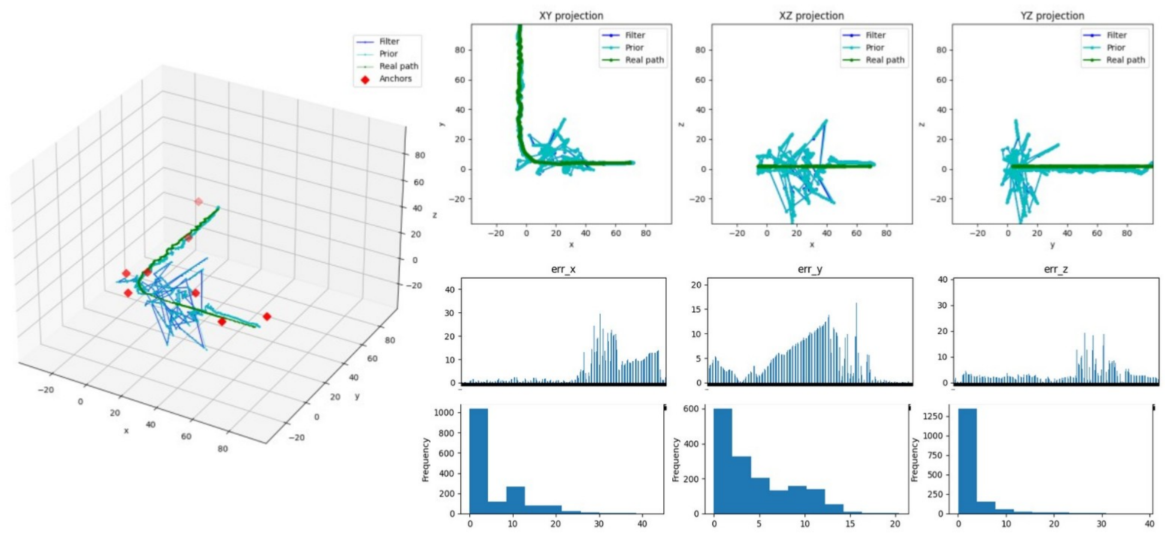
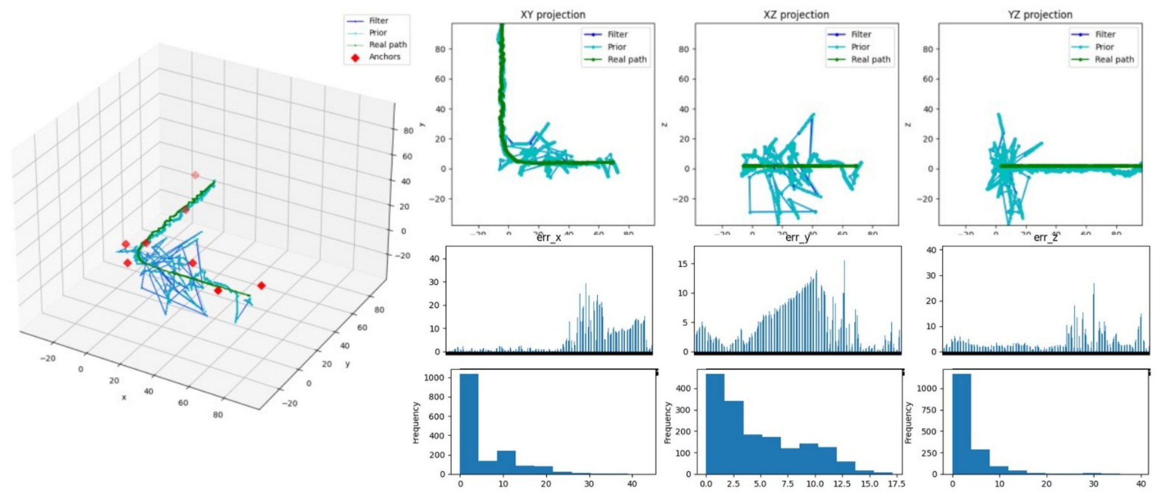
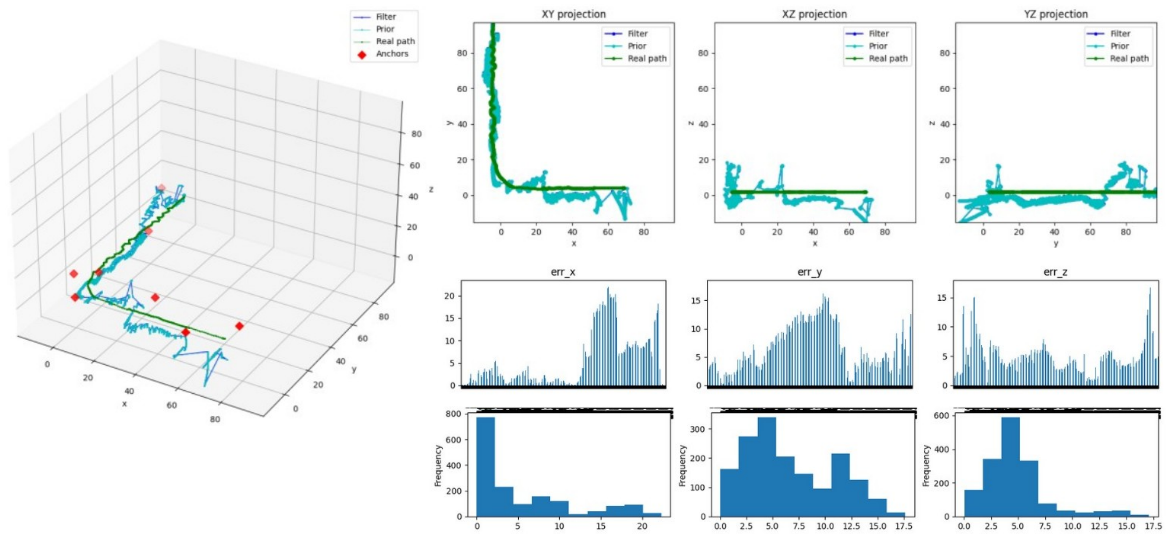


Figure A13.4: EKF CV (a), CVCA (b) and CA (c) plots for 90-degree-bend scenario







Glossary

2D	Bidimensional
3D	Three-dimensional
AP	Absolute Positioning
CA	Constant Acceleration
CV	Constant Velocity
CVCA	Constant Velocity Constant Acceleration
DGPS	Differential GPS
EKF	Extended Kalman Filter
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
IPS	Indoor Positioning System
KF	Kalman Filter
LC	Loosely Coupled
LiDAR	Light imaging Detection and Ranging
LOS	Line of Sight
LSE	Least Square Error
LSE-GC	Least Square Error with Geometry Constraints
NLOS	Non Line of Sight
OPS	Outdoor Positioning System
PPWP	Project Proposal and Working Plan
RADAR	Radio Detection and Ranging
RMSE	Root Mean Square Error
RP	Relative Positioning

RTK	Real Time Kinematic
STD	Standard deviation
STR	Snap to Road
TC	Tightly Coupled
TWR	Two Way Ranging
UWB	Ultra Wide Band
VAR	Variance
WP	Working Package