Master of Science Thesis in Electrical Engineering
Department of Electrical Engineering, Linköping University, 2023

# Estimating Relative Position and Orientation Based on UWB-IMU Fusion for Fixed Wing UAVs

**Daniel Sandvall and Eric Sevonius**

LINKÖPING
UNIVERSITY

Master of Science Thesis in Electrical Engineering

**Estimating Relative Position and Orientation Based on UWB-IMU Fusion for Fixed Wing UAVs**

Daniel Sandvall and Eric Sevonius

LiTH-ISY-EX–23/5583–SE

Supervisor: **Anja Hellander**
ISY, Linköping University
**Linus Wiik**
Saab Dynamics
**Joel Wikner**
Saab Dynamics

Examiner: **Gustaf Hendeby**
ISY, Linköping University

*Division of Automatic Control*
*Department of Electrical Engineering*
*Linköping University*
*SE-581 83 Linköping, Sweden*

# Abstract

In recent years, the interest in flying multiple Unmanned Aerial Vehicles (UAVs) in formation has increased. One challenging aspect of achieving this is the relative positioning within the swarm. This thesis evaluates two different methods for estimating the relative position and orientation between two fixed wing UAVs by fusing range measurements from Ultra-wideband (UWB) sensors and orientation estimates from Inertial Measurement Units (IMUs).

To investigate the problem of estimating the relative position and orientation using range measurements, the performance of the UWB nodes regarding the accuracy of the measurements is evaluated. The resulting information is then used to develop a simulation environment where two fixed wing UAVs fly in formation. In this environment, the two estimation solutions are developed. The first solution to the estimation problem is based on the Extended Kalman Filter (EKF) and the second solution is based on Factor Graph Optimization (FGO). In addition to evaluating these methods, two additional areas of interest are investigated: the impact of varying the placement and number of UWB sensors, and if using additional sensors can lead to an increased accuracy of the estimates. To evaluate the EKF and the FGO solutions, multiple scenarios are simulated at different distances, with different amounts of changes in the relative position, and with different accuracies of the range measurements.

The results from the simulations show that both solutions successfully estimate the relative position and orientation. The FGO-based solution performs better at estimating the relative position, while both algorithms perform similarly when estimating the relative orientation. However, both algorithms perform worse when exposed to more realistic range measurements.

The thesis concludes that both solutions work well in simulation, where the Root Mean Square Error (RMSE) of the position estimates are 0.428 m and 0.275 m for the EKF and FGO solutions, respectively, and the RMSE of the orientation estimates are 0.016 radians and 0.013 radians respectively. However, to perform well on hardware, the accuracy of the UWB measurements must be increased. It is also concluded that by adding more sensors and by placing multiple UWB sensors on each UAV, the accuracy of the estimates can be improved. In simulation, the lowest RMSE is achieved by fusing barometer data from both UAVs in the FGO algorithm, resulting in an RMSE of 0.229 m for the estimated relative position.

# Acknowledgments

Firstly, we would like to thank our supervisors Anja Hellander, Linus Wiik and Joel Wikner for their support throughout the thesis. Thank you for all the interesting discussions and for guiding the thesis into what it is today. We would also like to thank our examiner Gustaf Hendeby for your advice along the way.

We are also thankful to Torbjörn Crona for giving us the opportunity to do our master's thesis at Saab Dynamics. It has been a great experience. Additionally, we would like to thank our colleagues for all the coffee breaks.

Lastly, we would like to direct a huge thanks to our friends and families for their support throughout our education and throughout this thesis.

*Linköping, Juni 2023*
*Daniel Sandvall och Eric Sevonius*

# Contents

# Notation

## Mathematical Notation

| Notation | Meaning |
|---|---|
| $\boldsymbol{a}$ | Vector |
| $\boldsymbol{A}$ | Matrix |
| $\hat{\boldsymbol{x}}_{k\|k-1}$ | Estimate of the state $\boldsymbol{x}$ at time step $k$ using measurements up to time step $k-1$. |

## Abbreviations

| Abbreviation | Meaning |
|---|---|
| A1 | Anchor number one |
| EKF | Extended Kalman Filter |
| FGO | Factor Graph Optimization |
| FRD | Front, Right, Down (coordinate frame) |
| GNSS | Global Navigation Satellite System |
| IMU | Inertial Measurement Unit |
| INS | Inertial Navigation System |
| LOS | Line of Sight |
| NED | North, East, Down (coordinate frame) |
| NLOS | Non-line of Sight |
| RTK | Real Time Kinematic (GPS) |
| T1 | Tag number one |
| UAV | Unmanned Aerial Vehicle |
| UWB | Ultra-wideband |

# 1 Introduction

In this chapter, the background and aim of the thesis is introduced, as well as the contributions of each author. Also, the available hardware is listed.

## 1.1 Background

Over the last couple of years, advances in the field of Unmanned Aerial Vehicles (UAVs) have led to an interest in using them for more complex tasks. To accomplish these tasks, it is necessary to use teams of UAVs, leveraging the power of cooperating intelligent agents. Teams of UAVs in formation rely on accurate relative positional data, usually provided through an Inertial Navigation System (INS) in combination with sharing accurate positional data from Global Navigation Satellite Systems (GNSS) in the formation. However, since GNSS is not reliable indoors and since there is an interest in using UAVs in GNSS-denied locations outdoors, there is a need for other methods to reliably estimate the relative position between UAVs.

Ultra-Wideband (UWB) is an emerging technology that is being used especially for positioning in GNSS-denied locations and indoors, with Apple's airtag product being a recent success story [1]. Beyond being applicable in environments where GNSS traditionally is not, UWB sensors are inexpensive, energy efficient and operate at a high rate [2]. UWB technology has been successfully used for localization indoors through the use of triangulation of data from UWB anchors placed throughout the environment, such as in [3]. However, requiring placement of sensors in the environment in advance is unsuitable for robots exploring an unknown environment. Furthermore, as the UWB sensors have difficulties communicating over larger distances, using anchors at fixed positions is unsuitable for UAVs operating outside over large areas and at high altitudes.

An alternative to fixed anchors in the environment is placing anchors on one or more of the vehicles in a formation, designated as *leaders*. Thus, the relative distance to the leaders is received directly. Only using the relative distance with a single anchor and tag leaves ambiguity as to the orientation and position of the members of the formation. One way to resolve this ambiguity is by letting each member of the formation be equipped with multiple UWB sensors, thus enabling the possibility of triangulating the position and orientation using the UWB data. This approach requires a sufficient amount of UWB nodes mounted on the members of the formation in order for the relative positioning problem to be observable, especially for aerial vehicles operating in a 3D environment, which is discussed in [4]. Alternatively, fewer UWB sensors can be mounted and fusion of the UWB data with data from Inertial Measurement Units (IMU) mounted on each member of the formation can be utilized, such as in [5].

To fuse measurements from a UWB and an IMU, an Extended Kalman Filter (EKF), or a variation of it, can be used. In [6] the EKF is compared to a variation of the Iterative EKF (IEKF), where the IEKF performs slightly better at relative positioning than the EKF in Line of Sight (LOS) scenarios, and significantly better in Non-line of Sight (NLOS) scenarios. Because of interference in the NLOS scenario, disturbances can occur, causing errors in the estimated range, which could make the linearization more sensitive and thus favor the IEKF. Another solution to this problem is an adaptive EKF, as studied in [7], where the process noise was estimated online using maximum likelihood estimation. This resulted in better performance than an EKF in NLOS situations.

An alternative to the class of Kalman estimation techniques which are usually employed is Factor Graph Optimization (FGO), which has gained traction over the last couple of years in some areas. The technique was first used in a SLAM setting in 1997 [8], and now it has become one of the most popular ways to implement SLAM in research [9]. As opposed to classical Kalman techniques which are typically configured as filters (they only take into account one time step at a time), FGO solvers usually operate as smoothers or fixed-lag smoothers, which means they account for the history of the system during estimation. This is especially helpful in nonlinear estimation where a single poor linearization can have a big effect on the estimation, and hence smoothing estimators which relinearize past states during solving are helpful. A number of authors have recently seen good success with combining UWB positioning with FGO, such as [10] and [11].

## 1.2 Purpose and goal

The aim of this master's thesis is to evaluate the performance of UWB sensors for use in relative position estimation of UAVs in formation flight. The thesis deals with a simple formation consisting of two UAVs, denoted the *leader* and *follower*, in which the follower follows the leader at a close distance. The goal is to deploy a relative positioning estimation algorithm on the follower, utilizing UWB data and INS data to estimate the relative position and orientation of the follower with respect to the leader. Two separate estimation algorithms are developed and evaluated, one EKF-based solution and one FGO-based.

Two Make Fly Easy MFE Believer - Kits equipped with Cube Orange Flight Controller flight computers, and eight Makerfabs DW1000 UWB nodes are available for the thesis. The thesis focuses on the evaluation of the relative positioning algorithms based on UWB sensors with this hardware in mind. Consequently, models of the planes are used in the simulation environment, the choice of placement of the UWB sensors, and the use of additional sensors is motivated by the hardware. The hardware has been purchased and assembled by Saab Dynamics.

The thesis will attempt to fulfill the following goals

1. Evaluate the performance of the hardware UWB nodes.
2. Construct a simulation environment based on the results of the evaluation of the hardware.
3. Implement an EKF-based and an FGO-based algorithm to estimate the relative position and orientation using UWB and INS data from the flight computers in the simulation environment.
4. Evaluate the effect of the placement of UWB nodes on the performance of the developed relative positioning algorithms.
5. Investigate the effect of adding additional sensors to the overall best performing relative positioning algorithm, as well as changing some of the design choices, such as the states which are estimated.
6. Evaluate the best performing algorithm on the hardware.

The estimation scenario in Goal 3, that is using UWB sensors and INS data from the flight computers, is referred to as the base case.

## 1.3 Hardware platform

In this thesis, two Make Fly Easy Believers [12] equipped with Cube Orange Flight Controller flight computers [13] and Makerfab's ESP32 UWB Pro sensors [14] are used for experiments.

The Believers are fixed-wing propeller-planes with a wingspan of 2 meters. One of the Believers shown in Figure 1.1. The flight controllers use the PX4-stack for autonomous vehicles [15] that handles navigation, control and communication with a manual controller. The Cube Orange Flight controllers are equipped with IMU, barometer and a Here+ RTK GPS [16].

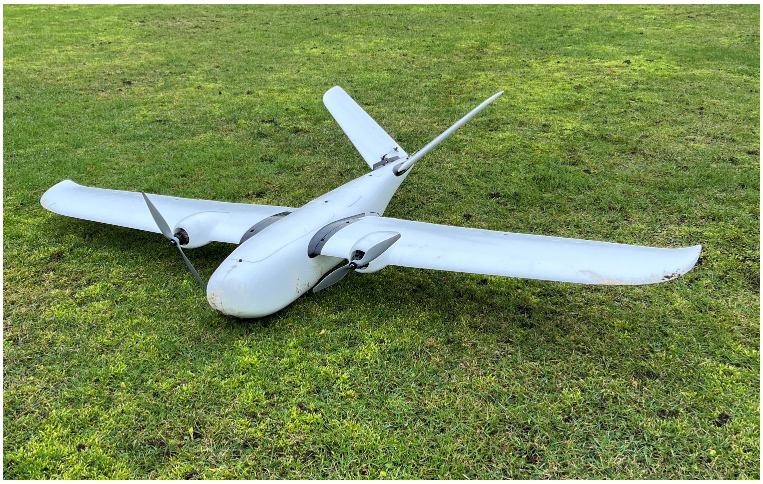


***Figure 1.1:*** *One of the Believers used in the thesis.*

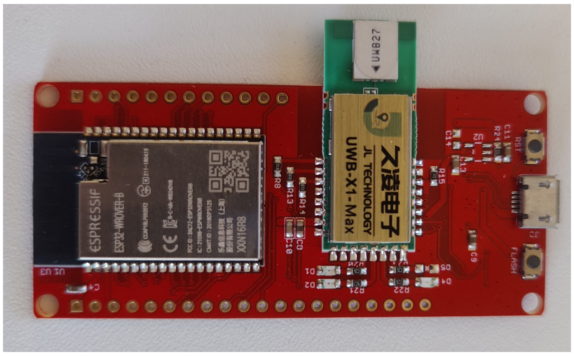


***Figure 1.2:*** *One of the Makerfabs UWB Pro sensors used in the thesis. The white rectangle on the green extension is the antenna of the UWB sensor.*

The Makerfab UWB Pro sensor is a Decawave DW1000 sensor with a more powerful antenna and is embedded on an ESP32 Wrover chip Figure 1.2. The leader plane is equipped with two UWB sensors and the follower is equipped with four.

## 1.4 Individual contributions

Daniel has developed the EKF-based solution and investigated different node constellations, while Eric has developed the FGO-based solution and investigated additional sensors that were added to the FGO. Both authors have contributed to the parts of the thesis regarding the UWB nodes and the experiments done with the hardware.

Throughout the thesis, the authors have written the parts related to their responsibilities and the part related to the UWB together. The general results in Section 7.2 and the corresponding discussion in Section 8.2, as well as the conclusions in Chapter 9, were also written by both authors.

## 1.5 Outline

The required theory for the thesis is presented in Chapter 2. In Chapter 3, the implementation of the UWB ranging protocol and the different measurement models are described. Chapter 4 and Chapter 5 contain the implementation of the relative positioning with EKF and FGO, including the motion model used for the estimation. Chapter 6 covers the method used for evaluating the implementations and the experiment setups. In Chapter 7, the results from the thesis are presented. Chapter 8 contains a discussion of the obtained results, and in Chapter 9 a conclusion is presented along with future work.

# 2 Theory

This chapter introduces the theory behind UWB sensors, the Extended Kalman filter and factor graph optimization. The theory regarding UWB focuses on the ranging aspect of estimating range and using multiple devices.

## 2.1 Ultra-wideband sensor

UWB sensors are sensors that utilize UWB signals to transmit data. The technology has seen most success as a ranging sensor in GNSS-denied environments, such as indoors. This type of ranging is typically carried out by an initiating radar, referred to as a *tag*, initiating communication with another tag, referred to as an *anchor*, to calculate its distance relative to the anchor. The names tag and anchor are slightly misleading in that the anchors do not necessarily need to be stationary, but the terminology has been adopted in this thesis since most of the literature concerns localization of mobile tags in environments with stationary anchors [2].

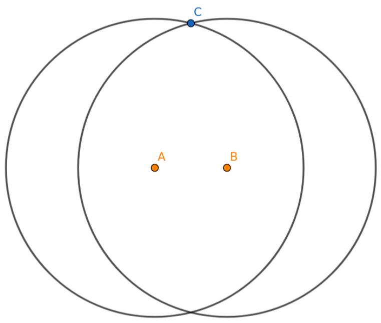
### 2.1.1 Ultra-wideband signal

As the name implies, UWB signals refers to signals with either large relative, or absolute, bandwidth [17]. These signals are transmitted as very short pulses, typically utilizing frequency bands of  500 MHz. It is the fact that the pulses are so short (leading to a high symbol-rate) and that they travel at the speed of light that make UWB signals so useful for ranging.

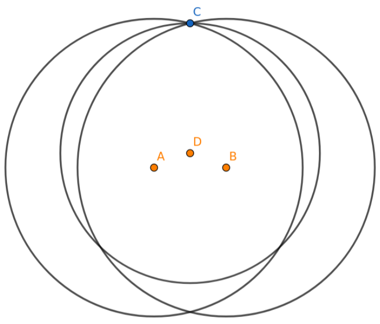
The large frequency bands of UWB signals imply that only a few frequency bands can be employed by local devices without causing interference and whilst adhering to the guidelines regarding the available frequency band for UWB signals by the Federal Communications Commission (FCC) and the European Union (EU) [17].

### 2.1.2 Localization using range measurements

To localize a target, multiple range measurements are required to determine the position. In the two-dimensional scenario, each range measurement results in a circle, with corresponding uncertainty, where the target can be located. By combining these measurements, the intersection points between the circles indicate possible locations of the target. If there are two measurements available, there will be two intersecting points between the circles, assuming that the sensors are placed at different locations. This results in two possible locations, where only one is the true position. If a third sensor is available and placed such that the three sensors are not placed on a single line, the solution is unambiguous, since the three circles only intersect at a single point. This is illustrated in Figure 2.1. In general, the number of measurements needed to localize a target is $n = d + 1$, where $d$ is the number of dimensions of the space where the target is located, and the sensor locations must form a shape with the same number of dimensions $d$. Thus, the minimum of nodes needed for relative positioning in three dimensions is four anchor nodes.



*(a) Two range measurements.*



*(b) Three range measurements.*

***Figure 2.1:*** *Visualization of the localization possibilities from range measurements in two dimensions. With two measurements, there are two possible locations of the target, but with three measurements, there is only one possible location.*

If a UWB node is placed in the origin of the local coordinate frame of the agent, the range measurement between two agents only depends on the distance between them. However, by placing a node away from the origin, the range measurement becomes coupled with the orientation of the agent. This means that the distance measured between two UWB nodes does not only depend on the distance between the agents, but also the orientation of each agent. By increasing the distance between the node and the origin, the correlation between the orientation and the range measurements is increased. By placing more than one tag on one agent, in this thesis the leader, as in [18], [4], [6], and [11], the relative orientation can be estimated from the measurements.

### 2.1.3 Time of arrival

One way of estimating distance from a pulse is by measuring the Time of Arrival (ToA). If the ToA can be measured with enough accuracy and the time the pulse was sent is known, the Time of Flight (ToF) can be calculated, which in turn can be used to calculate the distance if the speed of the pulse is known. There are a few different methods to estimate the transmission time, for example synchronizing the clocks of the devices [19]. Another way to estimate the transmission time is to have the sending device send the initiation message itself, and by knowing the processing time of the other device before it sends a reply, the distance can be calculated. This is called two-way ranging.

### 2.1.4 Two-way ranging

Two-Way Ranging (TWR) is a technique for recovering ToF measurements from messages passed between two independent devices with unsynchronized clocks. This text will refer to the device that initiates communication as the tag, and the other device as the anchor.

The *single-sided TWR* protocol is the simplest TWR protocol, consisting of two messages passed between the devices, and which yields a ToF measurement for the tag only. This is illustrated in Figure 2.2. The protocol starts with the tag sending a message to the anchor, denoted Poll. When a pre-defined point in the Poll message has left the antenna of the tag, called the *R-marker*, it starts measuring the round-trip time, $T_r$. Due to the fact that there is a limited bit-rate, packages take time to transmit and receive, and there will be a reply delay time, $T_d$, which needs to be accounted for. The anchor will measure $T_d$ from the time it receives the R-marker in the Poll message, until it has transmitted the R-marker of its reply message, denoted Range. The measured reply delay time is then appended to the Range message. Once the tag has received the R-marker in the Range message, it will stop measuring $T_r$. Using these quantities, the ToF, $T_{ToF}$, can be calculated as

$$T_{ToF} = \frac{1}{2}(T_r - T_d). \tag{2.1}$$

Since electromagnetic waves travel at the speed of light, $c$, the distance measurement $y$ is trivially obtained as $y = c \cdot T_{ToF}$.

Due to the fact that TWR protocols measure the distance travelled by an electromagnetic pulse over a short time period rather than the distance itself, they are sensitive to delays. A delay of one nanosecond roughly translates to a range

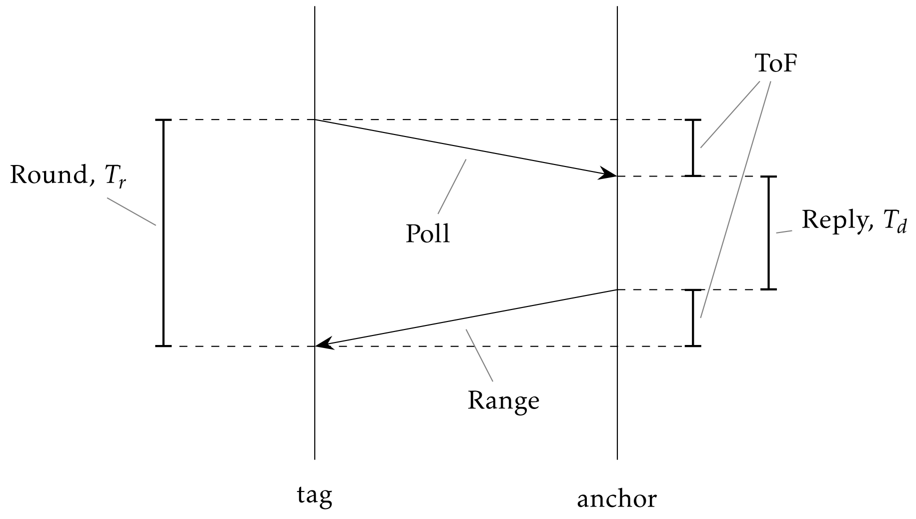


***Figure 2.2:*** *Illustration of the single-sided TWR protocol. The arrows represent messages which are passed between the tag device and anchor device. Time moves down along the vertical axis, and the thick vertical bars are time periods. The protocol starts with the tag sending the Poll message. Once received, the anchor will respond with the Range message. Practically, the Poll message may contain an instructed reply delay time timer period $T_d$, which is the time between the anchor receiving the Poll message and when the Range message is transmitted. It is also possible to keep a fixed $T_d$, thus bypassing the need to communicate it to the anchor, or for the anchor to decide the reply delay time, but which will require the anchor to transmit this reply delay time in the Range message. Once the Range message has been received by the tag, it will have a measurement of the round trip time, $T_r$, which is the time from the Poll being broadcast, to the time when the Range was received. Using the $T_r$ and $T_d$ the time of flight can be calculated.*

error of 30 cm. Therefore, the influence of imperfect independent clocks can have a large impact on the time-of-flight measurement using TWR protocols. The influence of the clock drift on a measured time is typically modelled as linearly dependent on the elapsed time. Using the linear model, the measured times in the single-sided TWR would then be

$$\begin{aligned} \hat{T}_r &= (1+\epsilon^t)T_r \\ \hat{T}_d &= (1+\epsilon^a)T_d, \end{aligned} \tag{2.2}$$

where $\hat{T}_r$ and $\hat{T}_d$ are the perturbed round-trip time and reply delay time respectively, and $\epsilon^t$ and $\epsilon^a$ are the deviations from the nominal frequency for the tag and anchor respectively. The error in the measurement is then

$$\begin{aligned}
&\frac{1}{2}(\hat{T}_r - \hat{T}_d) - \frac{1}{2}(T_r - T_r) = \frac{1}{2}(\epsilon^t T_r - \epsilon^a T_d) \\
&\overset{(2.1)}{=} \epsilon^t T_{ToF} + \frac{T_d}{2}(\epsilon^t - \epsilon^a) \approx \frac{T_d}{2}(\epsilon^t - \epsilon^a) ,
\end{aligned} \tag{2.3}$$

where in the third step the fact that $T_{ToF}$ is several magnitudes smaller than $T_d$ was used (order of nanoseconds versus order of milliseconds). As seen from (2.3) there are two ways that the magnitude of error can be kept low: through minimizing the reply delay time and by reducing the clock drift. The simplest way to reduce clock drift is to use a more accurate crystal, however to reach clock errors below 10 parts per million (ppm) can be expensive. The largest contribution to the reply delay time is the package size and the data rate. To adjust the package size, whilst adhering to the 802.15.4 IEEE standard [20], involves tuning the preamble length or the data length. The preamble is a code of repeated symbols which is the first part of the message and is used to estimate the channel impulse response, and in turn allow for better long-range performance [21]. To change the preamble length is to change the amount of repetitions of the predefined code, and a longer preamble length is generally necessary to achieve longer ranges since it affects the resolution of the impulse response, however, the effect will diminish as the preamble length grows larger in proportion to the data rate [21]. The data length will depend on the implementation of the protocol and whether there is a need to send additional data as part of the protocol. Adjusting the data rate will affect the power consumption and maximum range. A lower data rate leads to a longer maximum range, but an increase in power consumption [21]. Table 2.1 shows the estimated error based on the clock drift and reply delay time.

***Table 2.1:*** *Effect of clock offset and reply delay time on the error in single sided TWR. Adapted from [21]*

| $T_d$ \ clock error | 2 ppm | 5 ppm | 10 ppm | 20 ppm | 40 ppm |
|---|---|---|---|---|---|
| 100µs | 0.1 ns | 0.25 ns | 0.5 ns | 1 ns | 2 ns |
| 200µs | 0.2 ns | 0.5 ns | 1 ns | 2 ns | 4 ns |
| 500µs | 0.5 ns | 1.25 ns | 2.5 ns | 5 ns | 10 ns |
| 1 ms | 1 ns | 2.5 ns | 5 ns | 10 ns | 20 ns |
| 2 ms | 2 ns | 5 ns | 10 ns | 20 ns | 40 ns |
| 5 ms | 5 ns | 12.5 ns | 25 ns | 50 ns | 100 ns |

An alternative to the single-sided protocol are the double-sided protocols. Double-sided TWR (DS-TWR) protocols typically involve three to four messages passed between the devices and use a similar method as the single-sided protocol to estimate the ToF. The start of a double-sided TWR protocol is a full single-sided

TWR protocol, but involves the tag responding with an additional message. This adds another reply delay time and round-trip time, as seen in Figure 2.3. In the double-sided protocol, the second message is renamed Poll Ack so that the final message remains the Range message. A fourth message can be added at the end, denoted Range Report, in which the anchor reports the estimated range to the tag.

There are different double-sided TWR protocols which estimate the ToF using the messages in Figure 2.3. Typically, they arise from efforts to minimize the error caused by clock drift. One such protocol is the *Symmetric Double-Sided Two-Way Ranging protocol* [20] which is a part of the IEEE 802.15.4a standard. It minimizes the error by enforcing a fixed length reply delay time, hence the name symmetric. While it has good performance, with errors due to clock drift below a second [22], the fixed length reply delay time can be problematic. The *Alternative double-sided two-way ranging protocol* [22] is, as the name suggests, an alternative to the classic symmetric double-sided two-way ranging protocol. It is formed by considering the product between the round-trip times of the tag and anchor

$$T_r^t T_r^a = (2T_{ToF} + T_d^t)(2T_{ToF} + T_d^a), \tag{2.4}$$

which can be rearranged as

$$T_{ToF} = \frac{1}{2}\frac{T_r^t T_r^a - T_d^t T_d^a}{T_r^t + T_r^a + T_d^t + T_d^a}. \tag{2.5}$$

In [22] it is shown that the error resulting from clock drift is on the order $\epsilon T_{ToF}$, which is small enough to be ignored. In practice, the UWB signals are usually transmitted and received by antennas. As a consequence, the time from the message being sent until it is received will include the time for the signal to pass through the transmitting and receiving antennae. While such small delays may be ignored in most use cases, when it is added to the propagation time of a signal travelling at the speed of light it can cause large static errors in the TWR calculation. The antenna delay of the sensors can be accounted for by subtracting the propagation time through the transmitting and receiving antenna from every message. The propagation time through the antenna varies from chip to chip, and is therefore something that must be calibrated for each sensor. In [23] a method of calibrating antenna delays is presented, and the result is a bias error less than one centimeter.

The antenna delay is often modelled as linear, either as a single linear parameter accounting for the added propagation time for sending and receiving messages via the antenna [24], or two different parameters for sending and receiving messages via the antenna [23].

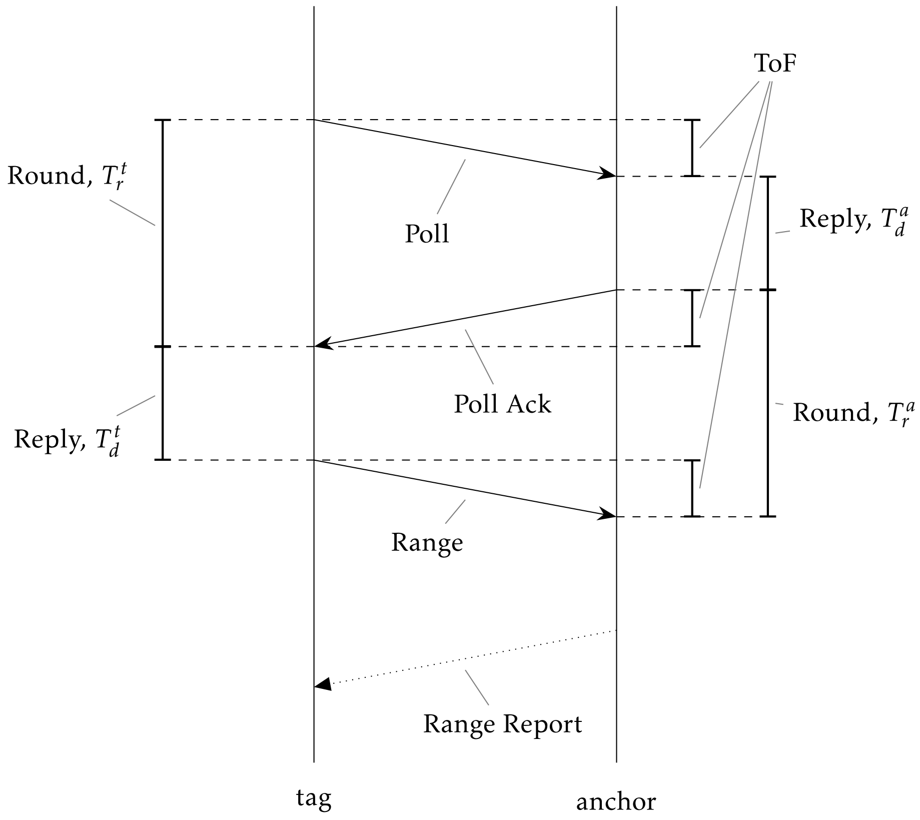


***Figure 2.3:*** *Illustration of the double-sided TWR protocol. The arrows represent messages which are passed between the tag device and anchor device, where the dashed arrow represents an optional message. Time moves down along the vertical axis, and the thick vertical bars are time periods. There are two types of time periods involved in the protocol, round trip times $T_r^*$ , and reply delay times $T_d^*$. A round trip time is the time it takes for a device to send a message to another device and receive a reply, and the round trip time is the time from when the other device receives the message until it responds. The protocol starts with the tag sending the Poll message. Once received, the anchor will respond with the Poll Ack message. When the tag has received the Poll Ack it will respond with the Range message. Optionally, the anchor can respond to the Range message with a Range Report, which simply contains the estimated range from the protocol. Practically, each message may contain an instructed reply delay time timer period for the other device. However, it is also possible to keep a fixed $T_d$, thus bypassing the need to communicate it to the other device. Or alternatively, for the other device to decide the reply delay time, but this will require the other device to transmit this reply delay time in the reply message. In order for the anchor to calculate the estimated range from the message exchanges, the tag must add the round-trip time of the first exchange, $T_r^t$, and the reply delay time of the second exchange, $T_d^t$, to the Range message. Thus, after the Range message the anchor can compute the time of flight message, and optionally transmit it to the tag with a Range Report message.*

### 2.1.5 Time division multiple access

Time Division Multiple Access (TDMA) is a technique for allowing multiple users to communicate over the same channel [2]. The exact details vary by implementation, but in general the scheme allocates the user slots over the time dimension rather than over the frequency dimension like in Frequency Division Multiple Access (FDMA) [2]. Of course, both techniques can be used together [2] as the time and frequency axes are independent. However, since UWB messages occupy large parts of the frequency spectrum, they can only be split into a few non-interferring radio frequency bands given current standards.

To use TDMA some form of scheduling is necessary in order to allocate the channel for one user at a time. When choosing a scheduling scheme, factors such as the number of users, average use of the channel by the users and channel usage efficiency vs. energy consumption needs to be considered. The simplest scheme involves splitting some time period equally between the users, without taking into account the needs of the users during the time period. This is a *static scheduling scheme*, which may lead to inefficiency in larger networks with a lot of users who often have varying needs for channel usage.

Alternatively, the scheduling may depend on external events, in which case it is called *dynamic scheduling*. Dynamic scheduling can be used to adjust the size of the time slots depending on the needs of the users, adjusting which users are allowed to access the channel based on some criterion, changing the scheduling scheme based on external events and more.

### 2.1.6 Data transmission

Apart from measuring distance, UWB sensors can be used to transmit data over shorter distances. This is used in the DS-TWR protocol to transmit the timestamps from the tag needed to calculate the time of flight in (2.5), but can also be used to share data between agents, as done in [18].

## 2.2 Motion models

In estimation theory, a central part of the problem is the system model that describes how the states evolves over time. When the model describes the motion of a vehicle of some sort, it is called a motion model. In some cases, it may be enough with a model that roughly describes the system, but sometimes a more accurate model is required. For example, the motion of a car can be described with a bicycle model with only a few parameters affecting the motion, or with a model including friction, wheel slip and other parameters affecting a real car.

If the motion model is unknown, or the existing model is more complicated than necessary for the task, a model describing a general motion, such as a constant velocity model or a coordinated turn model, can be used. For a constant velocity model, the vehicle is assumed to move at a constant velocity with an unknown acceleration affecting it, and for a coordinated turn model, the vehicle is assumed to maintain velocity and turn with a constant angular velocity affected by an unknown angular acceleration [19].

When the goal is relative positioning, one way to formulate a motion model is to derive it from the motion models of the agents, as described for two front steered cars in [25], but this, as stated in [25], requires the state of the leader to be available to the follower. To do this with an aircraft model, the relative motion model would be complex, and a lot of data needs to be transferred between the agents.

To simplify the problem, it is possible to consider how the relative state should change in a formation flight. The following agent should maintain its relative position to the leader, but the orientation of the two agents could change faster. One way to model this is to assume constant relative velocity, which should be close to zero when flying in formation, and constant angular velocity for each agent. The model error is then modelled by adding a noise corresponding to changes in velocity and angular velocity, affecting the corresponding states.

## 2.3 Kalman filters

For estimation of states where both the system and measurement models are linear, the Kalman filter has been the obvious choice since it was developed. For a system with a known model affected by white Gaussian noise with known covariance, the Kalman filter estimate is optimal [26].

The filter operates with two main steps, a state prediction and a measurement update. The prediction step uses a model of the system to predict the state in the next time step $k$, and updates the covariance of the state estimate accordingly, and the measurement step updates the state by comparing the measurement with the expected value calculated from the predicted states. The general state space model used to describe the system is formulated as

$$\boldsymbol{x}_k = \boldsymbol{F}\boldsymbol{x}_{k-1} + \boldsymbol{G}_u\boldsymbol{u}_{k-1} + \boldsymbol{G}_v\boldsymbol{w}_{k-1}, \qquad \operatorname{cov}(\boldsymbol{w}) = \boldsymbol{Q} \tag{2.6}$$

$$\boldsymbol{y}_k = \boldsymbol{H}\boldsymbol{x}_k + \boldsymbol{D}\boldsymbol{u}_k + \boldsymbol{e}_k, \qquad \operatorname{cov}(\boldsymbol{e}) = \boldsymbol{R} \tag{2.7}$$

where $\boldsymbol{x}$ are the states, $\boldsymbol{u}$ are the input signals, $\boldsymbol{y}$ are the measurements, and the vectors $\boldsymbol{w}$ and $\boldsymbol{e}$ are noises. With this system model, the prediction step of the Kalman filter is defined as

$$\hat{\boldsymbol{x}}_{k|k-1} = \boldsymbol{F}\hat{\boldsymbol{x}}_{k-1|k-1} + \boldsymbol{G}_u\boldsymbol{u}_{k-1} \tag{2.8}$$

$$\boldsymbol{P}_{k|k-1} = \boldsymbol{F}\boldsymbol{P}_{k|k}\boldsymbol{F}^T + \boldsymbol{G}_v\boldsymbol{Q}\boldsymbol{G}_v^T \tag{2.9}$$

and the measurement step is defined as

$$S = HP_{k|k-1}H^T + R \tag{2.10}$$
$$K = P_{k|k-1}H^T S^{-1} \tag{2.11}$$
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - H\hat{x}_{k|k-1} - Du_k) \tag{2.12}$$
$$P_{k|k} = P_{k|k-1} - K_k H P_{k|k-1} \tag{2.13}$$

where $\hat{x}$ is the state estimate, $P$ is the covariance of the estimate, and $K$ is called the Kalman gain.

### 2.3.1 Extended Kalman filter

An extension of the Kalman Filter (KF) for non-linear models is the Extended Kalman Filter (EKF). A general non-linear model can be described as

$$x_k = f(x_{k-1}, u_{k-1}, w_{k-1}), \qquad \text{cov}(w) = Q \tag{2.14}$$
$$y_k = h(x_k) + e_k, \qquad \text{cov}(e) = R \tag{2.15}$$

where $f$ and $h$ are possibly non-linear functions. The EKF applies the KF algorithm to a linearization of the model, with either the first or second order Taylor Transformation (TT1, TT2). This linearization is then updated with each time step around the current state estimate $\hat{x}$. Unlike the KF, the EKF has no guarantee that the estimate will converge from an arbitrary initial guess [26].

The algorithm for the EKF, using TT1 and the non-linear model described in (2.14), is presented below, with the prediction step defined as

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_{k-1}, 0) \tag{2.16}$$
$$P_{k|k-1} = F_{k-1}P_{k-1|k-1}F_{k-1}^T + W_{k-1}QW_{k-1}^T \tag{2.17}$$

and the measurement update defined as

$$S_k = H_k P_{k|k-1} H_k^T + R_k \tag{2.18}$$
$$K_k = P_{k|k-1} H_k^T S_k^{-1} \tag{2.19}$$
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - h(\hat{x}_{k|k-1})) \tag{2.20}$$
$$P_{k|k} = P_{k|k-1} - P_{k|k-1} H_k^T S_k^{-1} H_k P_{k|k-1} \tag{2.21}$$

where $F_k$ and $H_k$ are the Jacobian matrices with respect to the state defined as

$$F_k = \left.\frac{\partial f(x, u, 0)}{\partial x}\right|_{\hat{x}_{k|k}, u_k} \tag{2.22}$$

$$H_k = \left.\frac{\partial h(x)}{\partial x}\right|_{\hat{x}_{k|k-1}} \tag{2.23}$$

where $\boldsymbol{x}$ is the state, and $\boldsymbol{W}$ is the Jacobian matrix with respect to the noise, defined as

$$\boldsymbol{W}_k = \left.\frac{\partial f(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{w})}{\partial \boldsymbol{w}}\right|_{\hat{\boldsymbol{x}}_{k|k}, \boldsymbol{u}_k} \tag{2.24}$$

where $\boldsymbol{w}$ is the noise vector in (2.14). The filter is initialized with an initial guess of the state $\hat{\boldsymbol{x}}_{0|0}$ and the covariance $\boldsymbol{P}_{0|0}$ [19].

## 2.4 Factor graph optimization

Factor graphs represent the factorization of a multi-variate function as a bipartite graph. The recent upswing in use of graphical models, such as Bayesian Networks [27] has sparked interest in applying graphical models to a wider set of problems. One of these graphical models which has gained attention in the estimation and navigation community is the factor graph. Factor graphs intuitively represent complex problems and offers efficient computations through a sparse representation of the problems and through the use of algorithms such as the sum-product algorithm.

### 2.4.1 Definition of factor graphs

A factor graph is a bipartite graph $\mathcal{G} = (\mathcal{X}, \mathcal{F}, \mathcal{E})$, where $x_i \in \mathcal{X}$ is a *state*, $f_j \in \mathcal{F}$ is a *factor* and $e_{i,j} \in \mathcal{E}$ is an *edge*. Thus, factor graphs have two types of nodes; the state node $x_i$ and factor node $f_j$. The edges $e_{i,j}$ can only connect between a state node and factor node. In the Forney style of factor graphs, which this thesis adopts, states are graphically represented as larger circles and factors as dots, see Figure 2.4 for an example. The *neighbourhood* $\mathcal{N}(f_j)$ is the set of variables connected to the factor $f_j$ (the set of variables that the factor $f_j$ depends on), and for which the shorthand $\mathcal{X}_j$ will be used. A factor graph $\mathcal{G}$ is a factorization of a global function $f(\mathcal{X})$ and can be expressed using the above notation as

$$f(\mathcal{X}) = \prod_j f_j(\mathcal{X}_j). \tag{2.25}$$

For example, in Figure 2.4, the joint density of a simple hidden Markov model with three time-steps, states $x_k$ and measurements $z_k$, $k = 0, 1, 2$ is represented as a factor graph.

Notice that the expression in (2.25) matches the example in Figure 2.4, since the joint density of the Hidden Markov Model is

$$P(\mathcal{X}, \mathcal{Z}) = P(x_0)P(x_1 \mid x_0)P(x_2 \mid x_1)P(z_0 \mid x_0)P(z_1 \mid x_1)P(z_2 \mid x_2)\,,$$

where $\mathcal{X} = \{x_0, x_1, x_2\}$ and $\mathcal{Z} = \{z_0, z_1, z_2\}$, which is exactly the product of the marginal densities represented as factors.

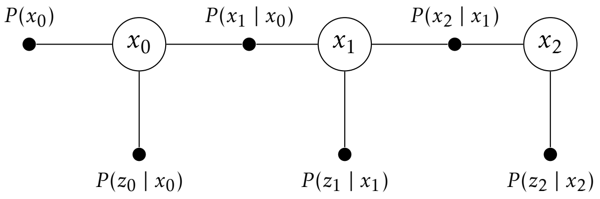


***Figure 2.4:*** *A factor graph encoding the joint density of a hidden Markov model over three time steps. The large circles are nodes in the factor graph and represent the hidden variables, and the smaller dots are the factors and encode the probability densities.*

### 2.4.2 Factor graph optimization for state estimation

Typically, factor graphs for state estimation are expressed in terms of Maximum A Posteriori estimate (MAP) inference, though of course most estimators can be expressed in the form of a factor graph, including Bayesian Filters [28]. The MAP estimator is the value for which the posterior distribution of an unknown state evolution $\mathcal{X} = \{x_1, x_2, ..., x_n\}$, given measurements $\mathcal{Z} = \{z_1, z_2, ..., z_m\}$ of the states, is maximized. To arrive at an expression for the MAP estimator, consider the posterior density of the states and measurements $P(\mathcal{X}, \mathcal{Z})$, which can be expressed using Bayes' law as

$$P(\mathcal{X}|\mathcal{Z}) = \frac{P(\mathcal{Z}|\mathcal{X})P(\mathcal{X})}{P(\mathcal{Z})}\,, \tag{2.26}$$

where $P(\mathcal{Z}|\mathcal{X})$ is the measurement density, $P(\mathcal{X})$ is the prior distribution, and $P(\mathcal{Z})$ is a normalization constant. Using (2.26), the MAP estimator can be expressed as

$$\mathcal{X}_{MAP} = \underset{\mathcal{X}}{\arg\max}\ P(\mathcal{X}|\mathcal{Z}) = \underset{\mathcal{X}}{\arg\max}\ P(\mathcal{Z}|\mathcal{X})P(\mathcal{X})\,, \tag{2.27}$$

where the normalization constant does not appear in the expression since it is independent of $\mathcal{X}$. Assuming that the measurements are conditionally independent, the measurement density can be factorized as

$$P(\mathcal{Z}|\mathcal{X}) = \prod_{j=1}^{m} P(z_j|\mathcal{X}_j)\,, \tag{2.28}$$

where $\mathcal{X}_j$ is the subset of states which the measurement $z_j$ depends on. Furthermore, if the measurements are assumed to be affected by Gaussian noise, then the individual factors in (2.28) can be rewritten as

$$P(z_j|\mathcal{X}_j) = \frac{1}{\sqrt{2\pi\Sigma_j}} \exp(-\frac{1}{2}\|h(\mathcal{X}_j) - z_j)\|_{\Sigma_j})\,, \tag{2.29}$$

where $h(\mathcal{X}_j)$ is the measurement model of $z_j$, and $\Sigma_j$ is the covariance of the Gaussian noise affecting $z_j$. Since (2.27) is phrased in terms of a maximization, the

MAP estimate is typically rewritten as

$$\mathcal{X}_{MAP} = \underset{\mathcal{X}}{\arg\max}\ l(\mathcal{X};\mathcal{Z})P(\mathcal{X})\,, \tag{2.30}$$

where $l(\mathcal{X};\mathcal{Z})$ is the likelihood of the states $\mathcal{X}$ given the measurements $\mathcal{Z}$, and is defined as any function proportional to $P(\mathcal{Z}|\mathcal{X})$. Using (2.29), the likelihood can be expressed as the product of likelihood functions:

$$\begin{aligned} l(\mathcal{X};\mathcal{Z}) &= \prod_{j=1}^{m} l(\mathcal{X}_j; z_j)\,, \\ l(\mathcal{X}_j; z_j) &= \exp(-\frac{1}{2}\|h(\mathcal{X}_j) - z_j)\|_{\Sigma_j})\,. \end{aligned} \tag{2.31}$$

Using (2.30) and (2.31), the final expression of the MAP estimate is

$$\mathcal{X}_{MAP} = \underset{\mathcal{X}}{\arg\max}\, P(\mathcal{X}) \prod_{j=1}^{m} \exp(-\frac{1}{2}\|h(\mathcal{X}_j) - z_j\|_{\Sigma_j}). \tag{2.32}$$

Since (2.32) is expressed as the product of the prior and the likelihood functions, the MAP estimate can be expressed as a factor graph with the prior and likelihood functions as factors. To compute the MAP estimate is therefore equivalent to maximizing its corresponding factor graph. Given the problem posed as a factor graph, the estimate can be obtained by reformulating the graph in terms of a nonlinear weighted least-squares problem and solving it using a typical optimization algorithm such as Levenberg-Marquardt [29]. The problem can also be solved using an optimizer that utilizes the graph structure, such as ISAM2 [30].

In practice, the MAP estimate is formulated using the same equations as a standard Bayesian filter, with the equations formulated as likelihood functions. The biggest difference lies in the role of the state transition function; in a factor graph, there is no difference between measurements and state transition functions. It is treated as a measurement that depends on the elapsed time between two consecutive states. Given enough measurements from the sensors, it is not even necessary to use a state transition function at all. This is in contrast to Bayesian filters, in which the state transition function is a part of a prediction step in each state update, as seen in (2.16).

For the typical state estimation task using the MAP estimate in which the goal is to estimate an unknown quantity over time, the factor graph will consist of nodes representing the quantity over time. New nodes are added for every incoming measurement, so that the factor corresponding to the measurement can be connected to the nodes. Therefore, if there are measurements that only affect a subset of the unknown quantities, then only the nodes which are a part of the factor will be added and connected to the new factor. Thus, the framework naturally handles irregular measurements. Depending on the optimizer, the graph will either have to be solved or iteratively updated every time a new estimate of the unknown quantities is required, that is, unlike a Bayesian filter in which adding a measurement updates the estimate automatically, for the factor graph framework it simply updates the formulation of the optimization problem.

In order for the nonlinear optimizer to solve the problem formulation, it is necessary to specify the Jacobians of the factors. The Jacobian can represent a TT1 or TT2 expansion. This thesis will exclusively deal with TT1 expansions due to the relatively well-behaved nonlinear functions treated. For a factor connected to multiple nodes, it is necessary to specify the Jacobian with respect to every connected node. Since the factors are implemented as likelihood functions, it is the Jacobians of the difference between the model and measured valued which are specified.

## 2.5 Lie groups

In this chapter, the basic theory of manifolds and Lie groups is established. The theory is then applied to the state estimation problem, which will be used by the FGO algorithm. It is by no means a thorough, or particularly mathematically rigorous introduction to Lie theory, since the theory of Lie groups is rich, but it explains the basics. For a more rigorous treatment, which still approaches the theory of Lie groups from a state estimation perspective, see [31].

### 2.5.1 Introduction

Manifolds, $\mathcal{M}$, are topological spaces that locally reassemble linear spaces. The subset of manifolds called *smooth manifolds* are the manifolds which are continually differentiable. A smooth manifold can be visualized as a smooth surface embedded in a higher dimensional space. The smoothness of these manifolds, combined with the existence of a local linear vector space which they reassemble at every point, implies that there exists a unique local tangent space at every point [31]. Since the local tangent space is a finite vector space it is isometric to $\mathbb{R}^n$, where $n$ is the dimensionality of the manifold, and thus smooth manifolds can be said to represent quantities which can locally be parameterized by $\mathbb{R}^n$ [31]. An important quantity for state estimation which has a manifold-structure is the set of orientations in three-dimensional space; locally parameterized by the triplet roll, pitch and yaw, but which is globally either represented by quaternions, or equivalently rotation matrices.

A Lie Group is a smooth manifold endowed with a group structure. In short, a group $(\mathcal{G}, \circ)$ is composed of a set $\mathcal{G}$ and a composition operator $\circ$. Furthermore, elements of the set $\mathcal{G}$ fulfill the following axioms with respect to the operator $\circ$: *Closure*, *Existence of an Identity element*, *Existence of an Inverse element*, and *Associativity*. The group structure comes with many benefits in the form of strong properties associated with the restrictions imposed on it. A few important consequences of the group structure for Lie Groups are: composition between elements on the manifold remain on the manifold, that each element on the manifold has an inverse element on the manifold, and that there exists a special identity element on the manifold with an associated special tangent space, called the *Lie alebra*, $\mathfrak{g}$. Another property of the Lie groups is that they can act on other manifolds through a *group action*, denoted $\cdot$. A group action, $a$, of a Lie group $(\mathcal{G}, \circ)$ acting on a manifold $\mathcal{M}$ is a group homomorphism between $(\mathcal{G}, \circ)$ and the group of diffeomorphisms of $\mathcal{M}$. That is, it is a structure-preserving map between elements of $(\mathcal{G}, \circ)$ and the group of all invertible functions applied to $\mathcal{M}$. For example, the Lie group $S^1$, which is the group of complex numbers with absolute value 1 endowed with the composition operator multiplication, acts on the manifold $\mathbb{R}^2$ through rotations $x \cdot p = xp, x \in S^1, p \in \mathbb{R}^2$.

As mentioned above, the set of orientations in three-dimensional space is represented by two equivalent smooth manifold-structures, the set of rotation matrices and the set of quaternions. Both of these sets are associated with Lie groups, known as the Special Orthogonal group, SO(3), and the Symmetric group, $\mathrm{S}^4$, respectively. Since these groups are equivalent, only SO(3) will be discussed for the rest of the text. The group action of SO(3) on $R^3$ represents the rotation of a point in three-dimensional space, that is: $x \cdot p = xp \in \mathbb{R}^3, x \in \mathrm{SO}(3), p \in \mathbb{R}^3$.

It is of interest to study local perturbations of quantities with a manifold-structure, for example in the context of a noise model or to perform non-linear optimization over the manifold. However, adding perturbations directly onto the manifold will almost certainly lead to an element outside the manifold. For example, adding a small perturbation matrix to a rotation matrix will most certainly lead to a matrix which no longer is a rotation matrix. Hence, it would be preferable to add increments to the local tangent vector space instead. To do so, a relationship must be established between these local changes on the tangent space and the underlying manifold. It is the Lie algebra which is the key to the relationship. The *exponential map*, $exp : \mathfrak{g} \mapsto \mathcal{M}$, and its inverse, the *logarithm map* $log : \mathcal{M} \mapsto \mathfrak{g}$, are defined as the operation that converts elements from the lie algebra to the manifold, and the operation that converts elements of the manifold to the lie algebra respectively.

Since the Lie algebra is a finite vector space, it is isometric to $\mathbb{R}^n$, and therefore there exists an isomorphism which maps elements from the Lie algebra to $\mathbb{R}^n$, which is denoted Vee: $(.)^\vee$, and another isomorphism which maps from $\mathbb{R}^n$ to the Lie algebra, which is denoted Wedge: $(.)^\wedge$. These isomorphisms are formed through:

$$\begin{aligned} \text{Wedge}: \quad x \mapsto x^\wedge \quad &= \sum_{i=1}^{n} x_i e_i^\wedge \\ \text{Vee}: \quad x^\wedge \mapsto (x^\wedge)^\vee \quad &= \sum_{i=1}^{n} x_i e_i, \end{aligned} \tag{2.33}$$

where $e_i$ is the $i$:th basis vector of $\mathbb{R}^n$, and $e_i^\wedge$ is called the $i$:th *generator* of $\mathfrak{g}$.

For SO(3), the Lie algebra, $\mathfrak{so}(3)$, is the space of skew-symmetric matrices [31]. A skew-symmetric matrix is a matrix which is the negative of its transpose, and is on the form:

$$[\omega]_\times = \begin{pmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix}. \tag{2.34}$$

(2.34) can be rewritten to the following:

$$[\omega]_\times = \omega_x \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} + \omega_y \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix} + \omega_z \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}. \tag{2.35}$$

Using the form in (2.35) and identifying it with the expression in (2.33), the generators of $\mathfrak{so}(3)$ are found to be on the following form:

$$e_1^\wedge = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}, \; e_2^\wedge = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix}, \; e_3^\wedge = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}. \tag{2.36}$$

In state estimation there is generally little to gain in working in the Lie algebra of the Lie group, therefore usually the capitalized version of the exp- and log-operators are introduced:

$$\begin{aligned} \mathbf{X} &= \mathrm{Exp}(x) = \exp(x^\wedge) \\ x &= \mathrm{Log}(\mathbf{X}) = \log(\mathbf{X})^\vee, \end{aligned} \tag{2.37}$$

where $\mathbf{X} \in \mathcal{M}$, and $x \in \mathbb{R}^n$. By using the Exp- and Log-operator it is possible to circumvent the lie-algebra entirely. For closed-form expressions of the capitalized exp- and log-operators for SO(3), see [31].

### 2.5.2 Lie groups for state estimation

When estimating quantities that are a part of a Lie group, the underlying manifold structure needs to be considered. For example, for a nonlinear optimizer it is important to be able to iterate over the solution space without encountering singularities, such as in the roll, pitch, yaw formulation of orientation. Secondly, to correctly propagate noise in the estimation framework it is important to take into account the underlying structure, otherwise the covariance will be deformed.

To use a nonlinear optimization scheme on a manifold that is part of a Lie group is the same in principle, but differ in that some concepts, such as incremental addition, need to be handled differently. Standard addition is not necessarily well-defined on a manifold, as it is not clear how to, e.g, add an incremental rotation to a rotation matrix. Therefore, it is common to introduce the $\oplus$ and $\ominus$ operators, which are defined as:

$$\begin{aligned} \mathbf{Y} &= \mathbf{X} \oplus x = \mathbf{X} \circ \mathrm{Exp}(x) \\ x &= \mathbf{Y} \ominus \mathbf{X} = \mathrm{Log}(\mathbf{X}^{-1}\mathbf{Y}). \end{aligned} \tag{2.38}$$

Using the operators introduced in (2.38), adding an angular increment $x$ to a rotation $\mathbf{X}$ is simply expressed as $\mathbf{X} \oplus x$, which is analogous to how addition would be used to incrementally iterate over roll, pitch, yaw in $R^3$ using a nonlinear optimization scheme. Additionally, these operators can be used to first define covariance on the tangent space on the manifold, and then define how to perturb a variable belonging to a manifold by a Gaussian variable. Covariance is defined in the same way as in the Cartesian case, through the expectation operator:

$$\Sigma = \mathrm{E}[(\mathbf{X} \ominus \bar{\mathbf{X}})(\mathbf{X} \ominus \bar{\mathbf{X}})^T] \,. \tag{2.39}$$

Furthermore, an element $\mathbf{X} \in \mathcal{M}$ is perturbed by a Gaussian variable $\nu \sim (0_{nxn}, \Sigma)$ through: $\mathbf{X} \oplus \nu$. Thus, the noise is specified in terms of the tangent plane rather than the manifold directly, the difference lies in that the noise is mapped unto the manifold.

# 3 System overview

This chapter covers the coordinate system, and how the states are defined in this system, the UWB implementation, and the measurement models used. The implementation of the relative positioning algorithms are presented in Chapter 4 and Chapter 5.

## 3.1 Coordinate frames

Two different coordinate frames are used in this thesis: the North, East, Down (NED) coordinate frame, and the Front, Right, Down (FRD) coordinate frame. The NED frame is defined as a local tangent plane with axes pointing north, east and down. Since the axes have a fixed direction, the coordinate system has a fixed global orientation. The second frame, the FRD frame, is fixed to the aircraft body, with axes in the front, right and down directions.

The relative position is in this thesis defined as the position of the follower in the NED frame with the origin in the leader. Hence, the relative position does not depend on the orientations of the agents. The relative velocity is defined as the difference in velocity between the agents in the NED frame. The orientation of each agent is also defined in the NED frame. In other words, the agent is pointing along the north axis if the rotation is set to zero. The angular velocity of each agent is defined in the FRD frame.

## 3.2 Ultra-wideband

In this section, the implementation of the UWB protocol and the motivation behind the parameters chosen is presented. Furthermore, the antenna delay calibration methods chosen are presented.

### 3.2.1 Two-way ranging with multiple anchors

Due to the numerous benefits of the alternative DS-TWR discussed in Section 2.1.4, it was used for implementing the DS-TWR in this thesis. In the standard alternative DS-TWR protocol, as described in Section 2.1.4, three messages are sent between the devices. If more anchors are added to the system, the number of messages required to estimate the ranges between the tag and all anchors can be calculated as $n = 3x$, where $x$ is the number of anchors. If four anchors are present, a total of twelve messages are required. To decrease the number of messages, the POLL message is broadcast to multiple anchors simultaneously. Each anchor then receives a unique delay for responding to the tag, and when the tag has received a response from all devices, a RANGE message is broadcasted. With this method, the number of messages sent is calculated as $n = x + 2$, and with $x = 4$, only six messages need to be sent. Thus, this protocol is faster than the previous one in the case of multiple anchors.

### 3.2.2 Scheduling

Each tag has a timer that schedules the initiation of the ranging protocol. With a single tag, this timer must be long enough for the previous cycle to finish. Thus, the duration of each cycle must be known.

### 3.2.3 Tag synchronization

The DS-TWR protocol available in the DW1000 library does not support multiple tags in the network. To solve this problem, TDMA was implemented. By having the tags synchronize each POLL message, the possibility of errors due to interruptions is reduced. The way the tags are synchronized is by resetting the scheduling timer each time a tag receives a POLL message from another tag. The timer is then set to the time it takes for the second tag to finish its cycle. However, if the cycle time depends on the number of anchors that are known to that tag, it is not guaranteed that the timer is the correct length. To eliminate this problem, the number of devices in the system is predefined and known to all devices. By using this number of devices when scheduling the tags, instead of the current number of devices that are known, both tags will always schedule such that the other will finish its cycle before starting a new cycle. As a result of this, the system performs measurements at a fixed frequency.

When a tag initiates its ranging protocol and sends a POLL message, the timer is set to the time it takes for two tags to communicate with four anchors. Thus, if the tag does not receive the POLL message from the second tag, it is scheduled to not start again until the second tag has performed a ranging cycle. This adds robustness to the system because single misses in the synchronization can be handled.

### 3.2.4 Choice of data rate and preamble length

When two or more devices receive time of flight measurements from distances that are relatively much larger than the mutual distance between the devices, it grows increasingly difficult to localize the source of the time of flight measurement. Since the UWB devices on the UAVs differ at most two meters, the 60-meter range reported in [21] while using the maximum data rate of 6.8 Mbps was deemed sufficient, given that measurements beyond that range would have little value. The preamble length was chosen to be 256 bytes, which is the maximum recommended size of preamble length given a data rate of 6.8 Mbps, and where a longer preamble length improves the maximum range of the UWB devices, as described in Section 2.1.4.

The number of bytes in the transferred message is set to 90 bytes to be able to carry all the necessary data for the DS-TWR protocol and the additional data that needs to be shared between the agents. This length of the data in combination with the other chosen parameters results in a message duration of approximately 400 µs according to [32]. Using these parameters, the frequency at which the range can be measured between one tag and four anchors is 80 Hz. With two tags present, all eight range measurements are received at 40 Hz.

### 3.2.5 Antenna calibration

To calibrate the antenna delays, two different methods are used. The first method calibrates the antenna delay of three devices based on the method in [24], and the second method calibrates one device using an already calibrated device as reference [33]. The measured time of flight is approximated to

$$t_{meas} = t_{TOF} + t_{d1} + t_{d2} \tag{3.1}$$

where $t_{TOF}$ is the theoretical time of flight for the set distance, and $t_{d1}$, $t_{d2}$ are the antenna delays of the two devices. In reality, the antenna delay is different for transmitting and receiving, and the estimate depends on multiple transmissions, as described in Section 2.1.4, and thus depends on the four different delays.

In the first method, the distance between two devices is measured, with the antenna delay set to zero, by measuring about 5000 times and calculating the mean distance. This is done between all three devices, such that six measurements are generated by measuring the distance between two devices twice, once where device one is the tag and device two is the anchor and vice versa. The least squares method is then used to calculate antenna delays for the devices. Because the delay affecting the estimate is different depending on which one of the two devices that initiates the ranging, measuring both ways between two devices, instead of only one way as in [24], should result in a more accurate estimation.

The second method uses an already calibrated device as reference and the antenna delay set to zero on the device that is to be calibrated. 5000 measurements are performed at a known distance with the reference device set as tag, and the antenna delay is calculated with the least squares method.

## 3.3 Measurement models

This section describes the measurements models that are used in the estimation algorithms.

### 3.3.1 Angle measurement

The global orientation of the agents are measured from the flight computer of each agent, and the data from the leader is transferred over UWB to the follower. As a result, these measurements can only be received at the same rate as the UWB measurements. The angular measurements are described as

$$\boldsymbol{q}_e = \boldsymbol{q} + \boldsymbol{e}_q \tag{3.2}$$

where $\boldsymbol{q}_e$ is the orientation estimate from the flight computer, $\boldsymbol{q}$ is the true orientation of the agent, and $\boldsymbol{e}_q$ is the error of the estimate.

### 3.3.2 Range measurement

As presented in [18] and [11], the distance measurements between two UWB nodes, a tag and an anchor, are described as

$$d_{ij} = \|\boldsymbol{p}_f + \boldsymbol{R}_f \boldsymbol{p}_{f_j} - \boldsymbol{R}_l \boldsymbol{p}_{l_i}\|_2 + e_{d_{ij}} \tag{3.3}$$

where $d_{ij}$ is the distance between tag $i$ and anchor $j$, $\boldsymbol{p}_f$ is the relative position of the follower and $\boldsymbol{R}_f$ is the rotation matrix of the follower, $\boldsymbol{R}_l$ is the rotation matrix of the leader, $\boldsymbol{p}_{f_j}$ and $\boldsymbol{p}_{l_i}$ are the local positions of the UWB nodes in relation to the agents, and $e_{d_{ij}}$ is the measurement noise. The relative position $\boldsymbol{p}_f$ is expressed in the NED frame, with the origin placed in the origin of the leader. The node positions $\boldsymbol{p}_{f_j}$ and $\boldsymbol{p}_{l_i}$ are expressed in the FRD frame of each agent. In Figure 3.1, a graphical representation of the range measurement is presented.

The measurement noise is assumed to be Gaussian with zero mean, and the standard deviation is estimated with experiments on the hardware.

### 3.3.3 Barometric measurement

A barometer is a sensor that measures temperature and air pressure, and in this thesis, it is used to measure the altitude. To calculate the altitude from barometric

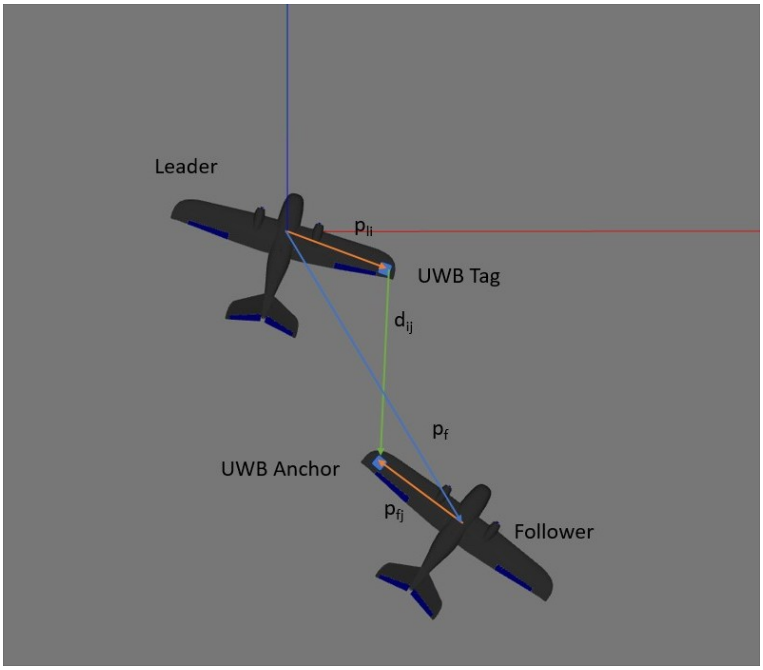


***Figure 3.1:*** *Illustration of how the measured distance between two UWB nodes depends on the node positions and the orientations of the agents. The dark blue and red lines represent the $x$-axis and $y$-axis of the NED frame sharing origin with the leader. The UWB nodes are illustrated with blue rectangles and the measured distance, the relative position of the follower, and the local position of the UWB nodes are illustrated with arrows. All notation in the figure is according to* (3.3).

measurements, the following expressions are used, which hold approximately in the troposphere (the atmosphere below 11,000 meters) [34]

$$\begin{aligned} \tau &= 15.04 - 0.00649 \cdot a \\ p &= 101.29 \cdot [(\tau + 273.1)/288.08]^{5.256}\,, \end{aligned} \tag{3.4}$$

where $\tau$ is the temperature in Celsius, $p$ is the air pressure in atmospheres, and $a$ is the altitude in meters. The expressions in (3.4) can be rewritten to the following expression of the altitude as a function of the pressure

$$\bar{h}(p) = a = \frac{1}{-0.00649}[288.08 \cdot (\frac{p}{101.29})^{\frac{1}{5.256}} - 273.1 - 15.04]. \tag{3.5}$$

Typically, barometers suffer from static biases which need to be calibrated or estimated online. This thesis considers the simplified case in which it is assumed that the sensor is well-calibrated.

### 3.3.4 IMU measurements

This thesis employs the combined IMU factor graph from [35], which is freely available in the GTSAM Toolbox [36]. The factor employs a measurement model for the angular velocity $\tilde{\boldsymbol{\omega}}(t)$ and acceleration $\tilde{\boldsymbol{a}}(t)$ originating from a 3-axis gyroscope and accelerometer on the form

$$\begin{aligned} \tilde{\boldsymbol{\omega}}(t) &= \boldsymbol{\omega}(t) + \boldsymbol{b}^g(t) + \boldsymbol{e}^g(t) \\ \tilde{\boldsymbol{a}}(t) &= \boldsymbol{R}^\top(t)(\boldsymbol{a}(t) - \boldsymbol{g}) + \boldsymbol{b}^a(t) + \boldsymbol{e}^a(t), \end{aligned} \tag{3.6}$$

where $\boldsymbol{\omega}(t)$ is the true angular velocity, $\boldsymbol{a}(t)$ is the true acceleration in the world frame, $\boldsymbol{b}^g(t), \boldsymbol{b}^a(t)$ are the biases of the gyroscope and accelerometer respectively, $\boldsymbol{e}^g(t), \boldsymbol{e}^a(t)$ is additive zero-mean Gaussian noise affecting the gyroscope and accelerometer respectively, $\boldsymbol{R}^\top(t)$ is the rotation matrix from the world frame to the body frame (NED to FRD using the notation in this thesis), and $\boldsymbol{g}$ is the gravitational constant in the world frame.

# 4 Extended Kalman filter

In this chapter, the implementation of the EKF is presented. The motion model is defined along with the Jacobians required by the EKF.

## 4.1 Filter implementation

The Extended Kalman filter is implemented using a C++ library [37] and runs offline with data collected through simulation or hardware experiments.

## 4.2 Motion model

Based on Section 2.2, a model with constant relative velocity and constant angular velocity is chosen for the EKF. As in [18], a frame with fixed orientation is used as reference, and in this case, the NED frame is chosen.

The angles of an aircraft are often denoted as roll, pitch and yaw, which are easy to interpret. However, as the aircraft rotates around its axes, there are discontinuities as the angles are defined on the interval $[-\pi, \pi]$. To prevent this behavior, quaternions are used to describe the orientation. In [19], the discrete time evolution of the quaternion and angular velocity is approximated to

$$\begin{pmatrix} \boldsymbol{q}_{k+1} \\ \boldsymbol{\omega}_{k+1} \end{pmatrix} \approx \begin{pmatrix} \boldsymbol{I}_4 + \frac{T}{2}\boldsymbol{S}(\boldsymbol{\omega}_k) & \frac{T^2}{2}\bar{\boldsymbol{S}}(\boldsymbol{q}_k) \\ \boldsymbol{0}_{3\times 4} & \boldsymbol{I}_3 \end{pmatrix} \begin{pmatrix} \boldsymbol{q}_k \\ \boldsymbol{\omega}_k \end{pmatrix} + \begin{pmatrix} \frac{T^3}{4}\bar{\boldsymbol{S}}(\boldsymbol{q}_k) \\ T\boldsymbol{I}_3 \end{pmatrix} \boldsymbol{w}_{\omega_k} \tag{4.1}$$

with $T$ defined as the time step between $k$ and $k+1$, the unknown angular acceleration $\boldsymbol{w}$, and $\boldsymbol{S}(\boldsymbol{\omega}_k)$ defined as the skew-symmetric matrix

$$\boldsymbol{S}(\boldsymbol{\omega}_k) = \begin{pmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & -\omega_z & \omega_y \\ \omega_y & \omega_z & 0 & -\omega_x \\ \omega_z & -\omega_y & \omega_x & 0 \end{pmatrix} \tag{4.2}$$

where $\omega$ is the angular velocity around the specified axis, and $\bar{\boldsymbol{S}}(\boldsymbol{q})$ is defined as

$$\bar{\boldsymbol{S}}(\boldsymbol{q}) = \begin{pmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & q_3 & -q_2 \\ -q_3 & q_0 & q_1 \\ q_2 & -q_1 & q_0 \end{pmatrix}. \tag{4.3}$$

with $\boldsymbol{q} = q_0 + q_1\boldsymbol{i} + q_2\boldsymbol{j} + q_3\boldsymbol{k}$ where $\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$ are the basic quaternion vectors.
With (4.1), the model, with constant relative velocity and constant angular velocity, can be written as

$$\boldsymbol{x}_{k+1} = \begin{pmatrix} \boldsymbol{I}_3 & T\boldsymbol{I}_3 & 0 & 0 & 0 & 0 \\ 0 & \boldsymbol{I}_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & \boldsymbol{I}_4 + \frac{T}{2}\boldsymbol{S}(\boldsymbol{\omega}_{l_k}) & \frac{T^2}{2}\bar{\boldsymbol{S}}(\boldsymbol{q}_{l_k}) & 0 & 0 \\ 0 & 0 & 0 & \boldsymbol{I}_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & \boldsymbol{I}_4 + \frac{T}{2}\boldsymbol{S}(\boldsymbol{\omega}_{f_k}) & \frac{T^2}{2}\bar{\boldsymbol{S}}(\boldsymbol{q}_{f_k}) \\ 0 & 0 & 0 & 0 & 0 & \boldsymbol{I}_3 \end{pmatrix} \boldsymbol{x}_k + \begin{pmatrix} \frac{T^2}{2}\boldsymbol{I}_3 & 0 & 0 \\ T\boldsymbol{I}_3 & 0 & 0 \\ 0 & \frac{T^3}{4}\bar{\boldsymbol{S}}(\boldsymbol{q}_{l_k}) & 0 \\ 0 & T\boldsymbol{I}_3 & 0 \\ 0 & 0 & \frac{T^3}{4}\bar{\boldsymbol{S}}(\boldsymbol{q}_{f_k}) \\ 0 & 0 & T\boldsymbol{I}_3 \end{pmatrix} \boldsymbol{w}_k, \tag{4.4}$$

$$\boldsymbol{x} = \begin{pmatrix} \boldsymbol{p} \\ \boldsymbol{v} \\ \boldsymbol{q}_l \\ \boldsymbol{\omega}_l \\ \boldsymbol{q}_f \\ \boldsymbol{\omega}_f \end{pmatrix} \tag{4.5}$$

where $\boldsymbol{x}$ is the state, $\boldsymbol{p}$ is the relative position, $\boldsymbol{v}$ is the relative velocity, $\boldsymbol{q}_l$ and $\boldsymbol{q}_f$ are the quaternions describing the orientation of the leader and the follower, and $\boldsymbol{\omega}_l$ and $\boldsymbol{\omega}_f$ are the angular velocities of the agents. The position, velocity and orientation are described in the NED frame, and the angular velocities are described in the FRD frame. The process noise $\boldsymbol{w}_k$ represents the unknown relative acceleration and the unknown angular acceleration of the leader and follower, respectively. The process noise vector is defined as

$$\boldsymbol{w} = \begin{pmatrix} w_{v_x} & w_{v_y} & w_{v_z} & w_{\omega_{l_x}} & w_{\omega_{l_y}} & w_{\omega_{l_z}} & w_{\omega_{f_x}} & w_{\omega_{f_y}} & w_{\omega_{f_z}} \end{pmatrix}^T. \tag{4.6}$$

## 4.3 Filter for the proposed model

Since the system model presented in Section 4.2 matches the nonlinear model defined in (2.14), the prediction equations (2.16) and (2.17) are used, and the matrix $\boldsymbol{W}$, defined in (2.24), is calculated as

$$\boldsymbol{W} = \begin{pmatrix} \frac{T^2}{2}\boldsymbol{I}_3 & 0 & 0 \\ T\boldsymbol{I}_3 & 0 & 0 \\ 0 & \frac{T^3}{4}\bar{S}(\boldsymbol{q}_{l_k}) & 0 \\ 0 & T\boldsymbol{I}_3 & 0 \\ 0 & 0 & \frac{T^3}{4}\bar{S}(\boldsymbol{q}_{f_k}) \\ 0 & 0 & T\boldsymbol{I}_3 \end{pmatrix} \tag{4.7}$$

with $\bar{S}(\boldsymbol{q})$ defined as in (4.3). The matrix $\boldsymbol{F}$, defined in (2.22), is for the chosen model defined as

$$\boldsymbol{F} = \begin{pmatrix} \boldsymbol{I}_3 & T\boldsymbol{I}_3 & 0 & 0 & 0 & 0 \\ 0 & \boldsymbol{I}_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & \boldsymbol{I}_4 + \frac{T^2+T}{2}S(\omega_{l_k}) & \frac{T^2+T}{2}\bar{S}(\boldsymbol{q}_{l_k}) & 0 & 0 \\ 0 & 0 & 0 & \boldsymbol{I}_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & \boldsymbol{I}_4 + \frac{T^2+T}{2}S(\omega_{f_k}) & \frac{T^2+T}{2}\bar{S}(\boldsymbol{q}_{f_k}) \\ 0 & 0 & 0 & 0 & 0 & \boldsymbol{I}_3 \end{pmatrix}. \tag{4.8}$$

With the UWB communication implemented as described in Section 3.2, all distance measurements do not arrive simultaneously, and thus there must be separate measurement updates for each new batch of measurements. The different measurement functions $\boldsymbol{h}_{d_i}(\boldsymbol{x}_k)$ can then be defined as

$$\boldsymbol{h}_{d_1}(\boldsymbol{x}_k) = \begin{pmatrix} d_{11} \\ d_{12} \\ d_{13} \\ d_{14} \\ \boldsymbol{q}_l \\ \boldsymbol{q}_f \end{pmatrix}, \quad \boldsymbol{h}_{d_2}(\boldsymbol{x}_k) = \begin{pmatrix} d_{21} \\ d_{22} \\ d_{23} \\ d_{24} \\ \boldsymbol{q}_l \\ \boldsymbol{q}_f \end{pmatrix} \tag{4.9}$$

where $d_{ij}$ is defined as in (3.3), and both $\boldsymbol{q}_l$ and $\boldsymbol{q}_f$ are defined as in (3.2). The Jacobian for these is, according to (2.22), defined as

$$\boldsymbol{H}_{d_i} = \begin{pmatrix} \frac{\partial d_{i1}}{\partial \boldsymbol{p}} & 0 & \frac{\partial d_{i1}}{\partial \boldsymbol{q}_l} & 0 & \frac{\partial d_{i1}}{\partial \boldsymbol{q}_f} & 0 \\ \frac{\partial d_{i2}}{\partial \boldsymbol{p}} & 0 & \frac{\partial d_{i2}}{\partial \boldsymbol{q}_l} & 0 & \frac{\partial d_{i2}}{\partial \boldsymbol{q}_f} & 0 \\ \frac{\partial d_{i3}}{\partial \boldsymbol{p}} & 0 & \frac{\partial d_{i3}}{\partial \boldsymbol{q}_l} & 0 & \frac{\partial d_{i3}}{\partial \boldsymbol{q}_f} & 0 \\ \frac{\partial d_{i4}}{\partial \boldsymbol{p}} & 0 & \frac{\partial d_{i4}}{\partial \boldsymbol{q}_l} & 0 & \frac{\partial d_{i4}}{\partial \boldsymbol{q}_f} & 0 \\ 0 & 0 & \boldsymbol{I}_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \boldsymbol{I}_4 & 0 \end{pmatrix} \tag{4.10}$$

where $i = 1, 2$.

## 4.4 Covariance matrices

The EKF utilizes two different known covariance matrices, the covariance of the process noise $Q$ and the covariance of the measurements $R$. The covariance matrices are chosen as

$$\boldsymbol{Q} = \begin{pmatrix} 4^2 I_3 & 0 \\ 0 & 0.1^2 I_6 \end{pmatrix} \tag{4.11}$$

$$\boldsymbol{R} = \begin{pmatrix} \sigma_{UWB}^2 \boldsymbol{I}_4 & 0 \\ 0 & 0.01^2 \boldsymbol{I}_8 \end{pmatrix} \tag{4.12}$$

where the values of $Q$ and the variance of the angle measurements are tuned from simulations and $\sigma_{UWB}$ from hardware experiments, as described in Section 6.1.1.

# 5 Factor graph optimization

This chapter summarizes the factor graph estimation framework used. It includes a description of the problem formulation, the library used to implement the framework, the factors and the optimizer.

## 5.1 Problem formulation

The problem is formulated in two ways: one way that only estimates the relative translation, which is the one used in the base case, and one way that estimates the translation of the vehicles in relation to a fixed NED frame where the leader started, and that will be denoted *the local position case*. Thus, the first problem formulation matches the one used by the EKF developed in this thesis, see Chapter 4. The reason that two different approaches are studied is that it is easier to join an IMU integration scheme, such as the preintegrated IMU factor in [35], to the local position case in which the states involved in the IMU integration are separate. This is in contrast to the base case, where the translation and velocity of the leader and follower are joined into a single relative term. Furthermore, if the respective states of the leader and follower can be estimated without too much drift, then it would be useful in applications such as flying to known destinations, exploring unknown areas, and more.

The first formulation of the estimation problem considers the relative translation of the follower with respect to the leader, the rate of change of the relative translation. The individual orientations and angular velocities of the leader and follower are to be estimated as well. The relative translation and the orientation of the follower is represented as a pose, $\boldsymbol{x} \in \mathrm{SE}(3)$, the orientation of the leader is represented as a rotation matrix, $\boldsymbol{r} \in \mathrm{SO}(3)$, the rate of change in relative position and the angular velocity of the follower is represented as a 6d-vector, $\boldsymbol{t} \in \mathbb{R}^6$, and finally, the angular velocity of the leader is represented using a 3d-vector, $\boldsymbol{o} \in \mathbb{R}^3$. Thus, the problem can be formulated as an estimation problem of the pose of the follower and the orientation of the leader, given that the position of the mutual coordinate system is fixed to the position of the leader.

The local position formulation is in terms of a pose estimation problem, where the goal is to estimate the pose of the leader and the follower, $\boldsymbol{x}_k^l, \boldsymbol{x}_k^f \in \mathrm{SE}(3)$ respectively. The velocities of the leader and the follower, $\boldsymbol{v}_k^l, \boldsymbol{v}_k^f \in \mathbb{R}^3$ respectively, are also estimated in this formulation. The goal of the formulation is to integrate raw gyro and accelerometer data, as described in Section 3.3.4, instead of the processed gyro and accelerometer data in the form of orientations from the flight computers, which is used in the rest of the thesis. This has the advantage of creating an estimation algorithm which only depends on sensor data, and not an external estimator. To this end, the biases $\boldsymbol{b}_k = [\boldsymbol{b}_k^g, \boldsymbol{b}_k^a] \in \mathbb{R}^6$, where $\boldsymbol{b}_k^g, \boldsymbol{b}_k^a \in \mathbb{R}^3$ are the gyroscope and accelerometer biases respectively, will also need to be estimated as part of the problem formulation.

## 5.2 Factor graph software library

The Georgia Tech Smoothing and Mapping (GTSAM) is an actively maintained C++ library [36] which was used to implement the factor graph framework. It is a BSD-licensed project which was started 2010 at the BORG lab at Georgia Tech by Frank Dallaert. It has been used successfully for problems in areas such as SLAM [9], Structure from Motion [38], and Visual Odometry [39].

## 5.3 Factors

For the base case, 4 different types of factors are used for the estimation: a UWB factor, two different types of motion model factors, and an angular update factor. In the local position formulation, 3 types of factors are used for the estimation: a modified UWB factor, a barometer factor, and an IMU factor. To make the comparison between the base case and the local position case fair, a version of the barometer factor is also implemented for use in the base case, but it is only used when comparing the base case to the local position case.

### 5.3.1 Ultra-wideband factor - base case

The UWB factor for the base case implements the range measurement detailed in (3.3). It is a binary factor, connecting the nodes representing the pose of the follower, $\boldsymbol{x}$, and the orientation of the leader, $\boldsymbol{r}$, at a time step. Each UWB factor represents a single range measurement between a tag on the leader and an anchor on the follower. Formulating the factor as a single range measurement rather than all incoming range measurements at a timestamp has the benefit of allowing the factor to be used even when there are lost individual measurements.

The measurement equation (3.3) is reformulated in terms of the SE(3) Lie group as:

$$h = \| \boldsymbol{x}_k \cdot \boldsymbol{p}^a - \boldsymbol{r}_k \cdot \boldsymbol{p}^t \|_2 + e, \tag{5.1}$$

where $\boldsymbol{p}^a$ and $\boldsymbol{p}^t$, are the anchor positions in the frame of the follower and the tag positions in the frame of the leader, respectively. Since the follower state $x_k$ and the leader orientation $r_k$ are elements of Lie groups (SE(3) and SO(3)) the quantities $\boldsymbol{x}_k \cdot \boldsymbol{p}^a$ and $\boldsymbol{r}_k \cdot \boldsymbol{p}^t$ are the actions of $x_k$, and $r_k$ on the anchor position $p^a$, and the tag position $p^t$, respectively. The measurement is modelled as being perturbed by a one-dimensional Gaussian noise $e \sim \mathcal{N}(0, \sigma^2_{UWB})$.

By using the chain rule, the Jacobians for the difference between the measurement and model in (5.1) with respect to $\boldsymbol{x}_k$ and $\boldsymbol{r}_k$ can be calculated as:

$$\begin{aligned} \frac{\partial h}{\partial \boldsymbol{x}_k} &= \boldsymbol{J}^{norm} \boldsymbol{J}^{Action,\ SE(3)} \\ \frac{\partial h}{\partial \boldsymbol{r}_k} &= -\boldsymbol{J}^{norm} \boldsymbol{J}^{Action,\ SO(3)} , \end{aligned} \tag{5.2}$$

where $\boldsymbol{J}^{norm}$ is the Jacobian of the Euclidean norm, $\boldsymbol{J}^{action,\ SE(3)}$ is the Jacobian of the SE(3) action with respect to $\boldsymbol{x}_k$, and $\boldsymbol{J}^{action,\ SO(3)}$ is the Jacobian of the SO(3) action with respect to $\boldsymbol{r}_k$. For closed form expressions of these Jacobians, see [31].

### 5.3.2 Ultra-wideband factor - local position case

The local position version of the UWB factor is nearly identical to the first presented in Section 5.3.1, but instead of connecting between a pose and an orientation, it connects between the pose of the leader, $\boldsymbol{x}^l_k$ and the pose of the follower, $\boldsymbol{x}^f_k$. Thus, (5.1) is rewritten as

$$h = \|\boldsymbol{x}^f_k \cdot \boldsymbol{p}^a - \boldsymbol{x}^l_k \cdot \boldsymbol{p}^t\|_2 + e\,. \tag{5.3}$$

which corresponds to the norm of the difference between the anchor and tag position in the world frame. The Jacobian of the local position case is very similar to (5.2), with the difference being that the leader state is defined in SE(3) instead of SO(3):

$$\begin{aligned} \frac{\partial h}{\partial \boldsymbol{x}^f_k} &= \boldsymbol{J}^{norm} \boldsymbol{J}^{Action,\ SE(3)} \\ \frac{\partial h}{\partial \boldsymbol{x}^l_k} &= -\boldsymbol{J}^{norm} \boldsymbol{J}^{Action,\ SE(3)} . \end{aligned} \tag{5.4}$$

### 5.3.3 Motion model factors

The motion model needs to be formulated using the theory of Lie groups to be used in a factor graph. Using a first order Euler integration for twist integration, a constant twist model can be expressed as:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{x}_k \oplus t_k T \\ \boldsymbol{t}_{k+1} &= \boldsymbol{t}_k, \end{aligned} \tag{5.5}$$

where $\boldsymbol{x}_k \in \mathrm{SE}(3)$ is the pose at time index $k$, $\boldsymbol{t}_k \in \mathbb{R}^6$ is the twist at time index $k$, and $T$ is the time interval between $k+1$ and $k$. A constant angular velocity model can be similarly be expressed as:

$$\begin{aligned} \boldsymbol{r}_{k+1} &= \boldsymbol{r}_k \oplus \boldsymbol{o}_k T \\ \boldsymbol{o}_{k+1} &= \boldsymbol{o}_k, \end{aligned} \tag{5.6}$$

where $\boldsymbol{r}_k \in \mathrm{SO}(3)$ is the orientation at time index $k$, and $\boldsymbol{o}_k \in \mathbb{R}^3$ is the angular velocity at time index $k$. All the variables are defined as in Section 5.1.

Two different types of motion models are employed in the base case factor graph. The first is of the constant twist type, as expressed in (5.5), and is used to propagate the pose $\boldsymbol{x}_k$, and the twist $\boldsymbol{t}_k$. The second one is of the constant angular velocity type, as expressed in (5.6), and is used to propagate the orientation $\boldsymbol{r}_k$, and the angular velocity $\boldsymbol{o}_k$. Thus, both types of factor will be connected to four nodes: $\boldsymbol{x}_k, \boldsymbol{x}_{k+1}, \boldsymbol{t}_k, \boldsymbol{t}_{k+1}$ for the first type, and $\boldsymbol{r}_k, \boldsymbol{r}_{k+1}, \boldsymbol{o}_k, \boldsymbol{o}_{k+1}$ for the second type. Both factors use the elapsed time $T$ to form the measurement. As mentioned in Section 2.4.2, the treatment of the motion model as a factor in a MAP estimate means that it will not be used to predict in the same way as in a Bayesian filter. Instead, its importance lies in helping to shape the covariance to fit the model.

In [40] the continous time model of the noise $\bar{\boldsymbol{Q}}$ of a constant velocity model is found to be:

$$\bar{\boldsymbol{Q}} = \begin{pmatrix} \frac{1}{3}T^3\Sigma_Q & \frac{1}{2}T^2\Sigma_Q \\ \frac{1}{2}T^2\Sigma_Q & T\Sigma_Q \end{pmatrix}, \tag{5.7}$$

where $\Sigma_Q$ is the variance of the acceleration noise. In general, an ad hoc approximation of a discrete noise model, $\boldsymbol{Q}$, can be attained by assuming constant noise over the interval and integrating a continuous time noise model $\bar{\boldsymbol{Q}}$:

$$\bar{\boldsymbol{Q}} = T\boldsymbol{Q}. \tag{5.8}$$

By assuming that the angular acceleration noise will affect the angular velocity and angles in the same way as the acceleration noise affects the velocity and position, and using the ad hoc constant noise integration, the constant twist noise model and the constant angular velocity noise model can be written as:

$$\bar{\boldsymbol{Q}} = \begin{pmatrix} \frac{1}{3}T^4\Sigma_Q & \frac{1}{2}T^3\Sigma_Q \\ \frac{1}{2}T^3\Sigma_Q & T^2\Sigma_Q \end{pmatrix}. \tag{5.9}$$

The variance $\Sigma_Q$ is chosen as:

$$\begin{aligned} &\text{twist:} && \Sigma_Q = \boldsymbol{I}_{6\times6}\begin{pmatrix}64 & 64 & 64 & 1 & 1 & 0.64\end{pmatrix}^T \\ &\text{angular velocity:} && \Sigma_Q = \boldsymbol{I}_{6\times6}\begin{pmatrix}64 & 64 & 64\end{pmatrix}^T. \end{aligned} \tag{5.10}$$

The values were arrived at from tuning in simulation. The choice of a lower variance of the acceleration along the $z$-axis in the constant twist noise model is motivated by the fact that while maintaining a formation, the members of the formation usually try to maintain a fixed altitude with respect to each other.

Using the chain rule, the Jacobians with respect to $\boldsymbol{x}_k$, $\boldsymbol{x}_{k+1}$, $\boldsymbol{t}_k$, and $\boldsymbol{t}_{k+1}$ of the likelihood factor for the constant twist model, can be calculated as:

$$\begin{aligned}
\frac{\partial h}{\partial \boldsymbol{x}_k} &= \begin{pmatrix} \boldsymbol{J}^{I-Comp,\ arg1} \cdot \boldsymbol{J}^{Comp,\ arg1} \\ \boldsymbol{Z}_{6\times 6} \end{pmatrix} \\
\frac{\partial h}{\partial \boldsymbol{x}_{k+1}} &= \begin{pmatrix} \boldsymbol{J}^{I-Comp,\ arg2} \\ \boldsymbol{Z}_{6\times 6} \end{pmatrix} \\
\frac{\partial h}{\partial \boldsymbol{t}_k} &= \begin{pmatrix} T \cdot \boldsymbol{J}^{I-Comp,\ arg1} \cdot \boldsymbol{J}^{Comp,\ arg2} \cdot \boldsymbol{J}^{Exp} \\ -\boldsymbol{I}_{6\times 6} \end{pmatrix} \\
\frac{\partial h}{\partial \boldsymbol{t}_{k+1}} &= \begin{pmatrix} \boldsymbol{Z}_{6\times 6} \\ \boldsymbol{I}_{6\times 6} \end{pmatrix},
\end{aligned} \tag{5.11}$$

where $\boldsymbol{J}^{I-Comp,\ arg1}$ and $\boldsymbol{J}^{I-Comp,\ arg2}$ are the first and second arguments of an inverse composition $\boldsymbol{X}_{arg1}^{-1} \cdot \boldsymbol{X}_{arg2}$ between SE(3) arguments, and $\boldsymbol{J}^{Comp,\ arg1}$ and $\boldsymbol{J}^{Comp,\ arg2}$ are the first and second arguments of a composition $\boldsymbol{X}_{arg1} \cdot \boldsymbol{X}_{arg2}$ between SE(3) arguments, and $\boldsymbol{J}^{Exp}$ is the Jacobian of the capitalized exponential operator. See [31] for expressions for these Jacobians. The Jacobians for the constant angular velocity model are the same as (5.11), but with the Jacobians for the inverse composition and composition being defined for SO(3), and the zero and identity matrices being of size $3 \times 3$.

### 5.3.4 Angular update factor

The angular update factor is a simple unary measurement factor employed in the base case. It takes a measured orientation in the form of a rotation matrix as input and treats it as a measurement of the orientation at the time-step:

$$\begin{aligned}
&\text{Follower:} \quad h(\boldsymbol{x}_k) = \mathrm{Rot}(\boldsymbol{x}_k) + e \\
&\text{Leader:} \quad h(\boldsymbol{r}_k) = \boldsymbol{r}_k + e,
\end{aligned} \tag{5.12}$$

where Rot() is the operator which extracts the orientation from an SE(3) object. It has the Jacobian $(\boldsymbol{I}_{3\times 3},\ \boldsymbol{Z}_{3\times 3})$ if using the rotation first and orientation after notation. The noise is modelled as a zero-mean Gaussian with covariance $0.01 \cdot \boldsymbol{I}_{3\times 3}$. In practice, this more or less instructs the optimizer to use the measured rotation from the flight computer as its source of orientation.

For the factor using $\boldsymbol{r}_k \in \mathrm{SO}(3)$ the Jacobian of the difference between the estimated orientation, $\boldsymbol{r}_k$, and the measured orientation is simply $\boldsymbol{I}_{3\times3}$. Similarly, through the use of the chain rule, and since Jacobian of the Rot() action is $(\boldsymbol{I}_{3\times3}, \boldsymbol{Z}_{3\times3})$, the resulting Jacobian is $(\boldsymbol{I}_{3\times3}, \boldsymbol{Z}_{3\times3})$.

### 5.3.5 Barometer factor - base case

The barometer factor for the base case implements a measurement of the relative height between the leader and follower using pressure data from barometers on the vehicles.

Using (3.5), the measurement of $z$-component of the state $\boldsymbol{x}$ can be expressed as

$$h(p_l, p_f) = \bar{h}(p_l) - \bar{h}(p_f), \tag{5.13}$$

where $p_l$ and $p_f$ are the measured pressure in atmospheres of the leader and follower, respectively.

The Jacobian representing changes of the translation of $\boldsymbol{x} \in SE(3)$ is on the following form

$$\boldsymbol{J}^{trans} = \begin{pmatrix} \boldsymbol{Z}_{3x3} & \boldsymbol{R}(x) \end{pmatrix}, \tag{5.14}$$

where $\boldsymbol{R}(x)$ is the orientation of the pose $\boldsymbol{x}$. Furthermore, extracting the z-coordinate leads to the following Jacobian

$$\boldsymbol{J}^{z} = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix} \cdot \boldsymbol{J}^{trans}, \tag{5.15}$$

which is also the Jacobian of the factor with respect to the state $\boldsymbol{x}$.

### 5.3.6 Barometer factor - local position case

The barometer factor defined for the local position case measures the height of the leader and follower, rather than the relative height. While it would be possible to use a relative height measurement in the local position case as well, the problem is that the height estimate would be able to drift in a way that is not possible when considering two separate height measurements of the individual agents. Therefore, the measurement function is taken to be (3.5). Since it is only the measurement that differs between the first and local position case, the Jacobian in the local position case is also (5.15).

### 5.3.7 Preintegrated IMU factor

The Preintegrated IMU factor from GTSAM, which is an implementation of the IMU integration scheme presented in [35], is used to integrate the gyro and accelerometer data in the local position case. The novelty of the preintegration IMU factor lies in the fact that it combines IMU measurements over a period of time into a single factor. For sensors such as gyros and accelerometers with high frequency, creating a single factor instead of one for every measurement drastically improves the performance of the estimation algorithm. Furthermore, the preintegrated IMU factor performs its integration on-manifold, thus avoiding singularities and maintaining the proper covariance as discussed in Section 2.5.2.

The preintegration IMU factor connects the poses $\boldsymbol{x}_k^* \in \mathrm{SE}(3)$, velocities $\boldsymbol{v}_k^* \in \mathbb{R}^3$, and biases $\boldsymbol{b}_k^*$ between two time steps. For more information about the specifics of the preintegrated IMU factor, refer to [35].

## 5.4 The factor graphs

In Figure 5.1 the factor graph of the base case is presented, and in Figure 5.2 the factor graph of the local position case is presented. Both figures demonstrate the ideal case in which no data is lost, and every factor is added. In practice, if some data needed to form a factor is not available, then the factor will simply not be added to the graph. Notice that the graph in Figure 5.2 contains preintegrated IMU factors for both the leader and follower, and therefore all the accelerometer and the gyro data has to be available to add to the preintegrated measurement. Alternatively, the leader measurements could be preintegrated first and then sent to the follower, however, for the sake of simplicity all measurements are presumed to be sent to the follower in this thesis.

## 5.5 Optimizer

The by now classic factor graph optimizer ISAM2 (Incremental smoothing and mapping 2) [30] was used for the estimation. It is well-proven, including for real-time inference [41], and is available freely as a part of GTSAM. The ISAM2 version provided in the GTSAM 4.2 version was used in the thesis.

ISAM2 is an incremental nonlinear solver, utilizing a Bayes net representation to efficiently update the estimate. There are a few core ideas which makes the optimizer so efficient: a Bayes tree representation of the problem which allows for efficient algorithms for adding new measurements through editing of the tree, only relinearizing when necessary, and partial state updates which only propagate changes throughout a subset of the nodes when measurements only have a limited effect on previous states. The biggest disadvantage to the algorithm is due to it being an iterative algorithm, which makes it more sensitive to an improper problem statement. Specifically, an optimizer such as Levenberg-Marquardt can handle a slightly indeterminate system, however, ISAM2 cannot handle it at all. In practice, this means that the states need to be observable, or partially observable.

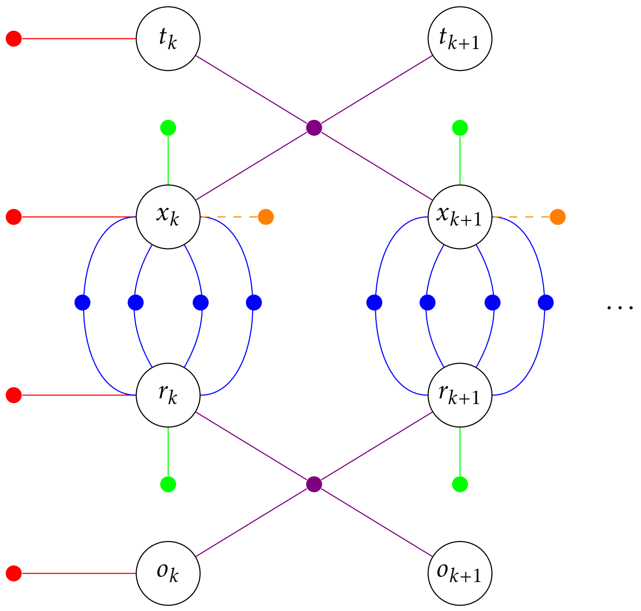


***Figure 5.1:*** *The base case factor graph. The states consist of the pose $\boldsymbol{x}_k \in SE(3)$ representing the orientation of the follower and its relative translation to the leader in the NED frame, the twist $\boldsymbol{t}_k \in \mathbb{R}^6$ of $\boldsymbol{x}_k$, the orientation $\boldsymbol{r}_k \in SO(3)$ of the leader, and finally the angular velocity $\boldsymbol{o}_k \in \mathbb{R}^3$ of $\boldsymbol{r}_k$. The red factors are priors on the states. The blue factors are the UWB factors as described in Section 5.3.1. The green factors are the angular update factors, which are described in Section 5.3.4. The violet factors are the two different kinds of motion models defined in Section 5.3.3. Finally, the orange dashed factor is the barometer factor described in Section 5.3.5 and only used when comparing to the local position case.*

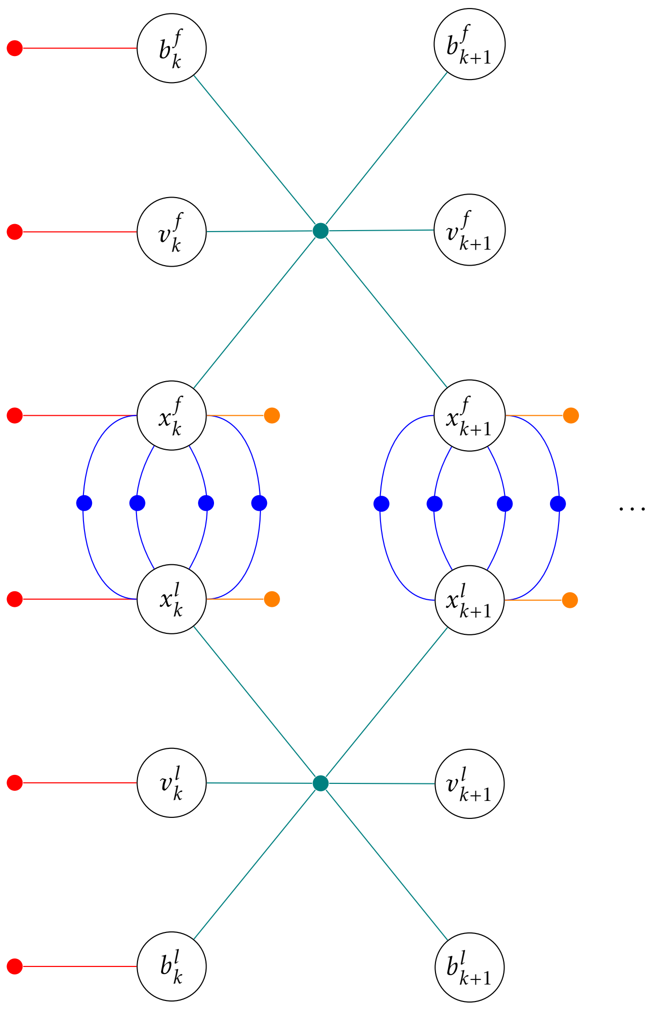


***Figure 5.2:*** *The local position case factor graph. The states consist of the pose* $\boldsymbol{x}_k^f, \boldsymbol{x}_k^l \in SE(3)$ *representing the pose of the leader and follower in the NED frame defined at the starting point of the leader, the velocities* $\boldsymbol{v}_k^f, \boldsymbol{v}_k^l \in \mathbb{R}^3$ *of the follower and leader, and the IMU biases* $\boldsymbol{b}_k^f, \boldsymbol{b}_k^l \in \mathbb{R}^6$ *of the follower and leader IMUs. The red factors are priors on the states. The blue factors are the UWB factors as described in Section 5.3.2. The orange factors are the barometer factors, which are detailed in Section 5.3.6. The teal factors are the preintegrated IMU factors from [35], and which are introduced in Section 5.3.7.*

# 6 Simulation studies and experiments

This chapter describes the methods used to obtain the results. The simulation environment, the formation flight scenarios used and the implementation of the range measurements are covered. The experimental setups are also described, as well as the methods for the antenna calibration and validation. Further, the method for investigating different node constellations and the method for comparing results are presented.

## 6.1 Experiments

In this section, the hardware experiments will be presented. All experiments are conducted outside, on the ground, in an environment with line of sight between all UWB nodes, except for the experiment when nodes are placed on the UAV, where the UAV itself may be between two nodes. When measuring the actual distance between two UWB devices, the position of the antenna, as seen in Figure 1.2, is used as the origin for each node.

### 6.1.1 Antenna calibration

All calibration are performed with the nodes placed seven meters apart on two tripods, which is close to the recommended distance for calibration [21]. Each distance is measured 5000 times and this data is split into training and validation data. The training data is then used to estimate the antenna delay with one of the methods described in Section 3.2.5. To validate the calibration, the time of flight is calculated using (3.1) by subtracting the delay from the measured time. This calculated time of flight is then compared to the ground truth. Validation is also carried out by measuring the distance between two calibrated nodes at a known distance, with the estimated antenna delay specified in the firmware. The measured distance is then directly compared to the ground truth.

Throughout the calibration experiments, the standard deviation $\sigma_{UWB}$ of the range measurement is calculated. This value is used when generating simulated range measurements.

### 6.1.2 Validating ranging implementation

To confirm that the ranging protocol behaves as expected, measurements are performed with all six nodes, two tags and four anchors, present. In the first experiment, only one tag is used, and in the second, both tags are used.

## 6.2 Simulation environment

The simulation engine Gazebo Classic [42] is used since it is compatible with ROS2 [43] and the software in the loop capabilities of PX4 Autopilot [15]. This allows the user to simulate a flight controller and its internal sensors that produce similar sensor data as the actual hardware would. Thus, all sensor data and state estimates are available to use in the positioning algorithm. To facilitate the development of the algorithms, all relevant data from the simulation is stored to be able to run the algorithms offline. To further increase the similarities between simulation and experiments, a simulation model of the hardware platform is used to achieve similar flight characteristics.

The general setup of the simulation is that two agents fly in different formations, depending on the scenario, following a predefined path using the flight controller to navigate to each point. When reaching the next point, both agents receive the next destination simultaneously and corrects its course. To prevent the agents from colliding, the two paths of the agents have a slight difference in height.

### 6.2.1 Simulation scenarios

To test the chosen methods, three scenarios with different characteristics are simulated. The first scenario is a close formation flight where the agents fly within four meters of each other and the relative velocity is close to constant, which is one of the assumption made for the system model in Section 4.2.

The second scenario is a long distance formation flight where the agents are approximately 20 meters apart. The goal for this scenario is to test the localization capabilities of the UWB network at a distance much greater than the largest distance between two UWB nodes located on the same agent.

In the third scenario, a uniformly distributed noise, in the range [-1,1] m, is added to each waypoint along the path of each agent. This results in sharper turns, where the state deviates more from the constant relative velocity and constant angular velocity model.

### 6.2.2 Simulated range measurements

In simulation, the UWB measurements are generated by calculating the distance between each anchor-tag pair and adding Gaussian noise with zero mean. To calculate the distance, the following data is required:

- Ground truth global position for both agents.
- Ground truth attitude for both agents.
- Anchor positions on the follower with respect to the FRD frame.
- Tag positions on the leader with respect to the FRD frame.

All ground truth data is published by the simulated flight controllers to ROS2 topics. The position published is expressed in relation to the starting position of the agent, and the difference between the global origin and the start position needs to be added to get the global position of the agent. To calculate the global position of the nodes, the node positions need to be rotated to the same orientation as the agent. The global position of a UWB node is calculated as

$$\boldsymbol{p}_{global} = \boldsymbol{p}_{local} + \boldsymbol{p}_{start} + \boldsymbol{R}\boldsymbol{p}_{node} \tag{6.1}$$

where $\boldsymbol{p}_{local}$ is the position of the agent relative to its starting position on the ground expressed in the NED frame, $\boldsymbol{p}_{start}$ is the initial position of the agent, $\boldsymbol{p}_{node}$ is the position of the UWB node in the FRD frame and $\boldsymbol{R}$ is the rotation matrix describing the orientation of the agent. The expression $\boldsymbol{R}\boldsymbol{p}_{node}$ gives the rotated position of the node that can be added to the position of the agent to get the global position. The distance is then calculated with the norm and added noise as

$$d_{ij} = \|\boldsymbol{p}_i - \boldsymbol{p}_j\|_2 + e_{d_{ij}} \tag{6.2}$$

where $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$ are the calculated global positions of tag $i$ and anchor $j$.

In Table 6.1, the positions of all UWB nodes used in the simulations are presented. These positions are chosen by placing UWB nodes on the hardware in the desired locations. On the follower, A1 and A2 are placed on the wing tips to maximize the difference along the $y$-axis. A3 is placed in the front of the follower as high up as possible without elevating the node above the body of the UAV, and A4 is placed in the back as low down as possible. This results in the largest possible difference along the $z$-axis, and an almost maximized distance between the nodes on the $x$-axis. On the leader, T1 and T2 are placed on the wing tips. These are the tags mainly used in the simulation. T3 is placed in the same position as A3, and T4 is placed as close to the origin of the agent as possible. These tags are used when examining node constellations.

***Table 6.1:*** *The placement of the UWB nodes in simulation. All positions are expressed in the FRD frame.*

| UWB node | x [m] | y [m] | z [m] |
|---|---|---|---|
| A1 | 0.13 | 0.92 | -0.04 |
| A2 | 0.13 | -0.92 | -0.04 |
| A3 | 0.50 | 0 | -0.04 |
| A4 | -0.15 | 0 | 0.18 |
| T1 | 0.12 | -0.91 | 0 |
| T2 | 0.12 | 0.92 | 0 |
| T3 | 0.50 | 0 | -0.04 |
| T4 | 0.1 | 0 | -0.04 |

### 6.2.3 Calibration error and missed measurements

The measurements on the hardware does not only depend on the distance and zero mean Gaussian noise, but also a static noise as a result of inaccurate calibrations of the antenna delay. The measurements can be described as

$$d = ct_{TOF} + ct_{error} + e, \tag{6.3}$$

where $d$ is the measured distance, $c$ is the speed of light, $t_{TOF}$ is the time of flight, and $e$ is zero mean Gaussian noise. The constant $t_{error}$ is the total antenna calibration error for both devices, and the distance $ct_{error}$ is the static error caused by the calibration error. This static noise is simulated with a different error added to each range measurement, since each pair of nodes has a different total antenna calibration error. The static errors are chosen from the results of the calibration experiments. With this added static noise, the assumptions made about the range measurements in both the EKF and the FGO are incorrect. For the EKF, the expected standard deviation for the range measurements is specified in (4.12), and for the FGO, the standard deviation is a part of the UWB factor described in Section 5.3.1. Because of the incorrect assumptions, two different choices of the expected standard deviation of the range measurements are tested, one that is the chosen $\sigma_{UWB}$, as described in Section 6.1.1, and one that is larger.

When measuring on hardware, some measurements are lost. By counting these, a probability of losing a measurement can be estimated. This is added to the simulation, where single measurements between nodes can be lost with this predefined probability.

## 6.3 Investigating node constellations

To answer the question regarding how the estimate is affected by number of nodes and their placement on the agent, the simulation environment is used. Since the minimal setup for achieving localization from range measurements is four anchors and one tag, no less than this number of nodes is examined.

## 6.4 Investigation additional sensors

Barometer and IMU data is investigated in this thesis as a part of the local position case formulation of the FGO, which is discussed in Chapter 5. Whilst the barometer and IMU data technically is fused with the orientation factors used in the base case formulation, the local position formulation investigates using the unfiltered data. Both the IMU model and barometer model used in simulation to generate the data are the same ones which generate the data for the simulated flight computer in simulation, and which are a part of the PX4 Software In The Loop stack [15].

Practically, the IMU model generates combined accelerometer and gyro data at 250 Hz and consists of a 4 Byte time stamp, three 4 Byte gyro data, and three 4 Byte accelerometer data, totaling 28 Bytes of data. As mentioned in Section 5.4, the follower will need to use the gyro and accelerometer data generated by the leader in the combined IMU factor. The Barometer generates data at a rate of 40 Hz, and only the pressure data is used, which is 4 Bytes of data. In order to keep it simple, simulated losses in data or simulated biases, such as those discussed in Section 6.2.3 will not be considered when evaluating the estimation algorithm based on the new sensors. Therefore, since UWB measurements are published at 80 Hz, each ranging protocol will need to transmit $\lceil 250/80 \rceil = \lceil 3.125 \rceil = 4$ IMU messages containing 28 bytes of data, and a single barometer message at 4 Bytes, totaling 116 Bytes. Every message in the protocol has room for 90 Bytes, and the Poll message in the DS-TWR protocol only needs to transmit a single Byte identifying it as a poll message. Furthermore, the Range message only needs to transmit 4 round trip time stamps (16 Bytes), 4 timer delay time stamps (16 Bytes), and a byte identifying it as a Range message, totaling 33 bytes, leaving 57 Bytes, which is well enough to send the remaining data that cannot fit in the Poll message. Therefore, it would be possible to send all the collected data as part of the TWR protocol. Another strategy would be to mount an additional UWB node on the leader and on the follower, connect them to another channel, and to continuously stream the IMU and barometer data from the leader (which would more than cover the need). Since the investigation into the extra sensors will be carried out in simulation, it is sufficient to conclude that it is possible to receive the required data for the estimation algorithm.

## 6.5 Comparing results

To compare the results, the root-mean-square error (RMSE) is calculated as

$$RMSE = \sqrt{\frac{\sum_{k=1}^{N}(\hat{\boldsymbol{x}}_k - \boldsymbol{x}_k)^2}{N}} \tag{6.4}$$

where $\boldsymbol{x}$ is the ground truth of the state, $\hat{\boldsymbol{x}}$ is the state estimate, and $N$ is the number of samples. As the first seconds of the datasets include the takeoff of the agents, and thus not flying in formation flight, the first 1000 samples are excluded from the RMSE.

# 7 Results

This chapter presents the results in this thesis. Firstly, the results from the UWB calibrations are presented. Secondly, the results from the different simulated scenarios described in Section 6.2.1 are presented along with the results from the simulations with added noise, described in Section 6.2.3. Finally, the results from both the evaluations of the different constellations of UWB nodes, and the evaluation of additional sensors are presented.

## 7.1 Calibration of Ultra-wideband nodes

In this section, the results of the calibration of the UWB nodes are presented. Firstly, the results from the three-way method are presented. Secondly, the results from the second method are presented. Both methods are described in Section 3.2.5. In both sections, two different types of validation data is used, as is described in Section 6.1.1. Throughout this section, the different UWB nodes are referred to with the abbreviation *T1* and *A1*, where T1 is tag number one, and A1 is anchor number one.

### 7.1.1 Calibration with three devices

In Figure 7.1, the estimated antenna delay is applied to the validation data using the approximate model in (3.1). The resulting mean values are presented in Table 7.1 and are within three centimeters of the true distance.

To further evaluate the calibration, the estimated antenna delay is set in the firmware of the UWB nodes, as previously described in Section 6.1.2, and the results are presented in Figure 7.2 where the mean distance is 6.972 m. The

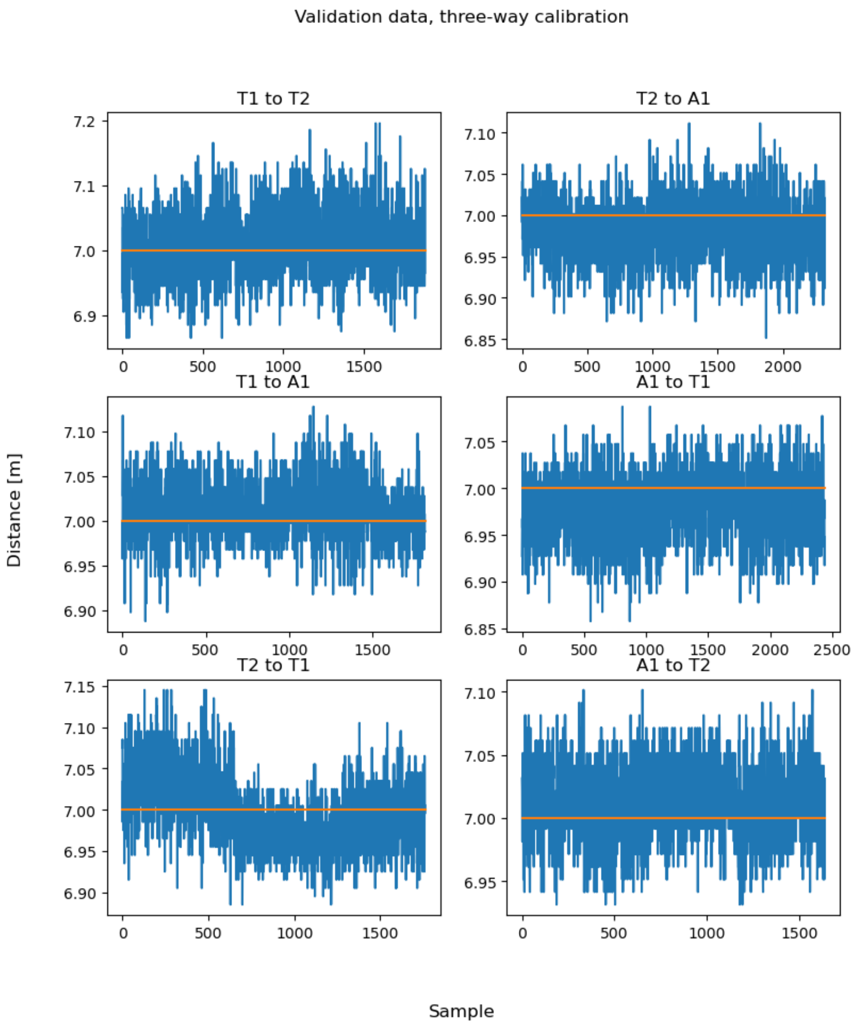


***Figure 7.1:*** *Validation data for three-way calibration. In the subplot showing the measured distance between T2 and T1, there is a shift in the measured range around sample 600. This is the result of some kind of disturbance, internal or external, that affects the measurements.*

***Table 7.1:*** *The mean distance for validation data.*

| Measurement | T1 to T2 | T1 to A1 | T2 to T1 | T2 to A1 | A1 to T1 | A1 to T2 |
|---|---|---|---|---|---|---|
| Mean [m] | 7.008 | 7.009 | 6.997 | 6.979 | 6.977 | 7.012 |
| Standard deviation [m] | 0.053 | 0.034 | 0.046 | 0.036 | 0.035 | 0.033 |

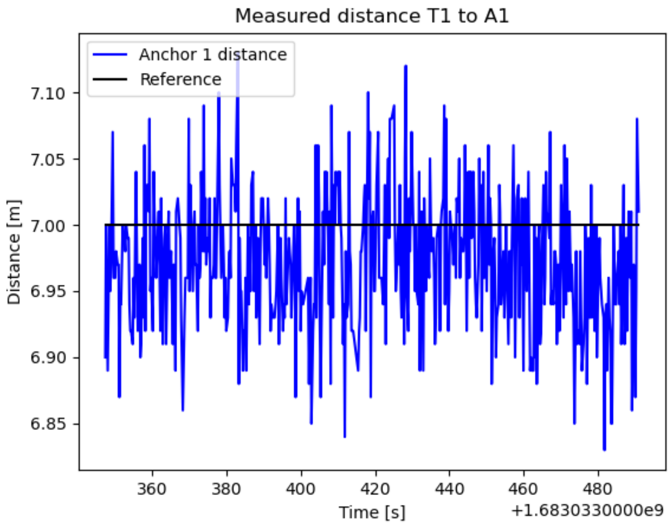


***Figure 7.2:*** *Validation of three-way calibration at 7 m, with estimated antenna delay set in the firmware of T1 and A1. The mean distance is 6.972 m and the standard deviation is 0.050 m.*

difference between the theoretical and experimental result for the measured distance between T1 and A1 is then 3.7 cm. The measurements are also evaluated at the distance 4.85 m shown in Figure 7.3, where the measured mean distance is 4.823 m.

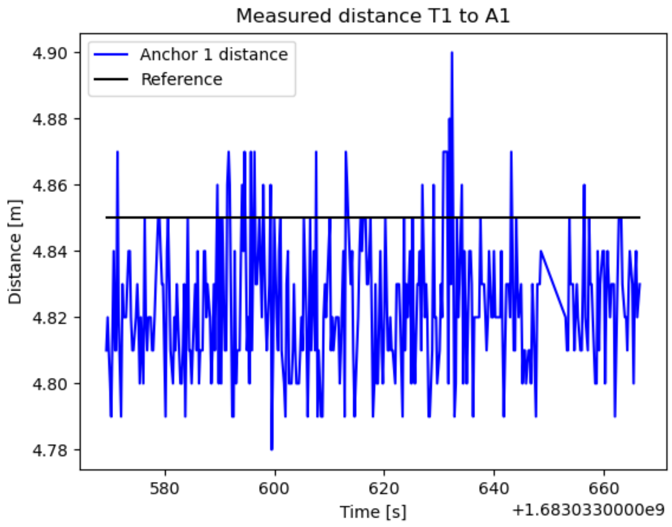


***Figure 7.3:*** *Validation of three-way calibration at 4.85 m, with estimated antenna delay set in the firmware of T1 and A1. The mean distance is 4.823 m and the standard deviation is 0.020 m.*

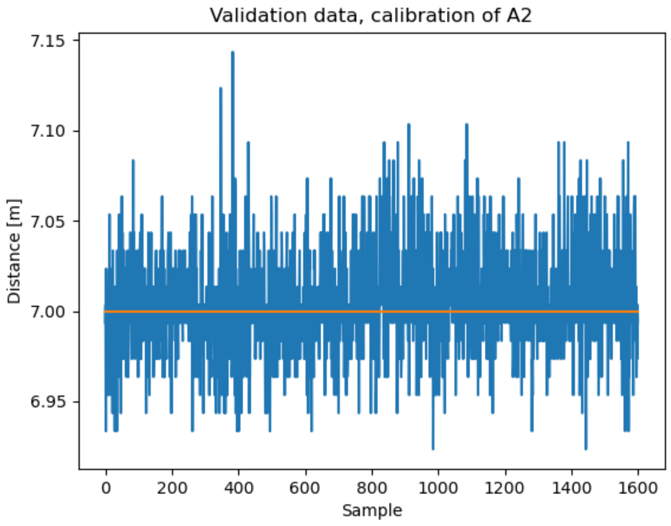


***Figure 7.4:*** *Validation data for calibration of A2 at 7 m. The mean distance is 7.006 m and the standard deviation is 0.030 m.*

### 7.1.2 Calibration with reference device

By using T1 as reference device, A2 is calibrated using the second method described in Section 3.2.5. The estimated antenna delay is applied to the validation data and the result is presented in Figure 7.4. The measured distance between the calibrated devices is shown in Figure 7.5. Here, the difference in mean distance is 3.9 cm. With this method, A2 is calibrated assuming that the set antenna delay of T1 is the true value, and it is therefore affected by previous calibration errors. In Figure 7.6, the distance between T2 and A2 is measured, using the estimated antenna delay, and a mean distance of 7.073 m is observed. This is a larger error than the error observed between T1 and A2 in Figure 7.5.

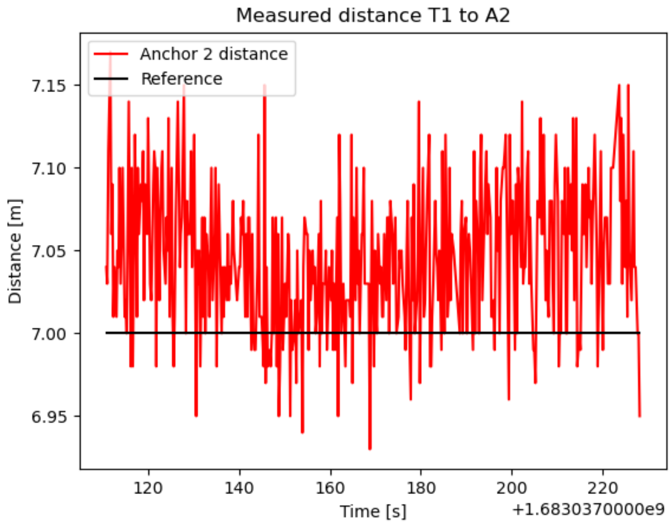


***Figure 7.5:*** *Measured distance between T1 and A2 with calibrated antenna delays. The mean distance is 7.045 m and the standard deviation is 0.044 m.*

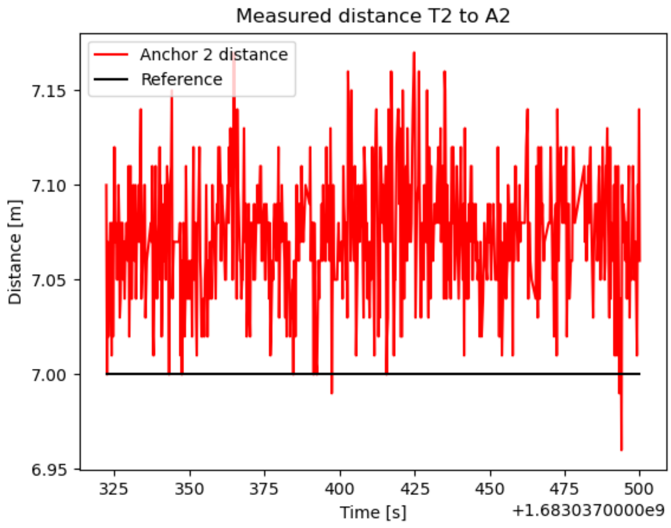


***Figure 7.6:*** *Measured distance between T2 and A2 with calibrated antenna delays. The mean distance is 7.073 m and the standard deviation is 0.033 m.*

### 7.1.3 Verification with all devices

In Appendix A, the results from experiments with all UWB devices are presented. It is concluded that the system works as expected with two tags and four anchors. However, when the antennas of the UWB nodes were rotated, the measured range changed drastically and the maximal error observed was 0.42 m.

### 7.1.4 Measurement properties

Throughout the experiments, the standard deviation of the measurements is between two and six centimeters. Thus, the standard deviation of the measurements in simulation is chosen as $\sigma_{UWB} = 0.06$ m. From the calibration results, the static noise for each measurement is chosen, as presented in Table 7.2.

The amount of lost range measurements is also measured, and the resulting probability of loosing a measurement between a tag and an anchor is 5%. This value is used in the simulation scenario described in Section 6.2.3.

***Table 7.2:*** *The static noises added to the range measurements. The values are chosen to mimic the results from Section 7.1.2.*

| Range measurement | Static noise [cm] |
|---|---|
| T1 to A1 | -2.7 |
| T1 to A2 | -5.4 |
| T1 to A3 | 8.2 |
| T1 to A4 | -8.4 |
| T2 to A1 | 2.6 |
| T2 to A2 | 4.8 |
| T2 to A3 | 1.4 |
| T2 to A4 | 6.9 |

## 7.2 Simulation

In this section, the results from simulation are presented, comparing the EKF algorithm to the FGO algorithm.

### 7.2.1 Close formation flight

In Figure 7.7-7.9 the plots of the estimated relative position, relative velocity, and relative orientation for the close formation flight scenario are presented, and in Table 7.3 the RMSE of the estimates is presented. The distance between the agents is calculated from the estimates and is shown in Figure 7.10. Position-wise, both estimates are more or less equivalent. However, the FGO seems to have a problem with estimating the relative velocity. The relative angles seem quite close, the difference being that the estimates from the FGO have some small bumps where the estimate is worse compared to the EKF.

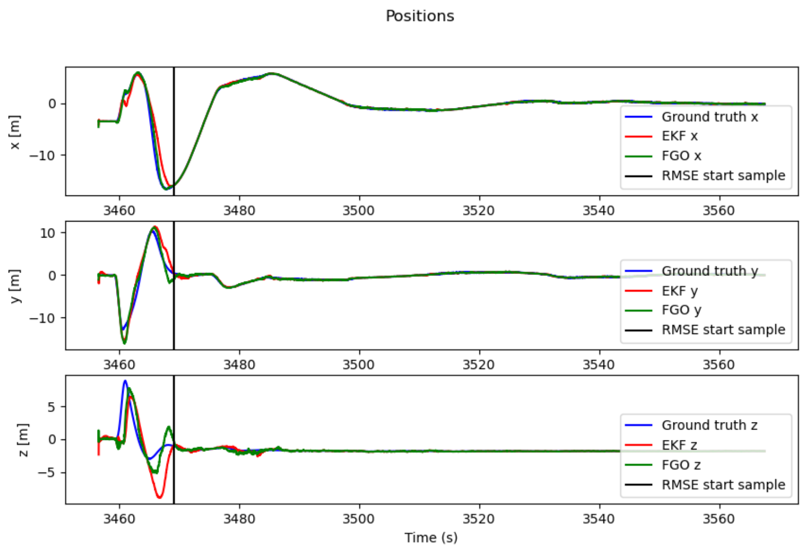


*Figure 7.7: Relative position estimates for the close formation flight. The vertical black line denotes the sample from which the RMSE calculations start.*

*Table 7.3: RMSE for close formation flight.*

| Estimate | Position [m] | Velocity [m/s] | Roll, Pitch, Yaw [rad] |
|---|---|---|---|
| EKF | 0.165 | 0.444 | 0.0048 |
| FGO | 0.170 | 1.0499 | 0.13 |

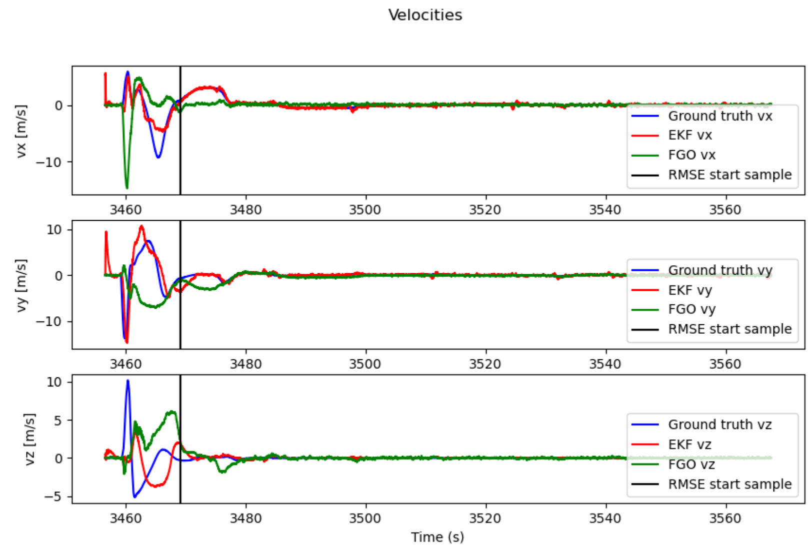


***Figure 7.8:*** *Relative velocity estimates for the close formation flight. The vertical black line denotes the sample from which the RMSE calculations start.*

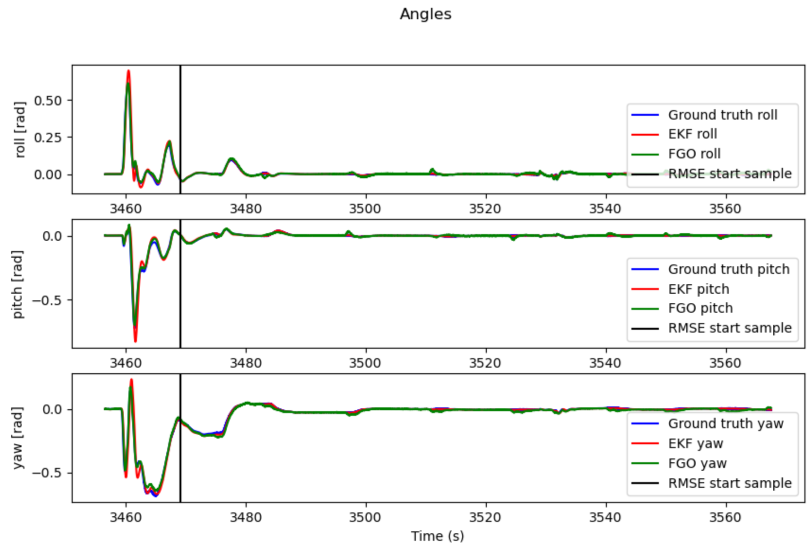


***Figure 7.9:*** *Relative orientation estimates for the close formation flight. The vertical black line denotes the sample from which the RMSE calculations start.*

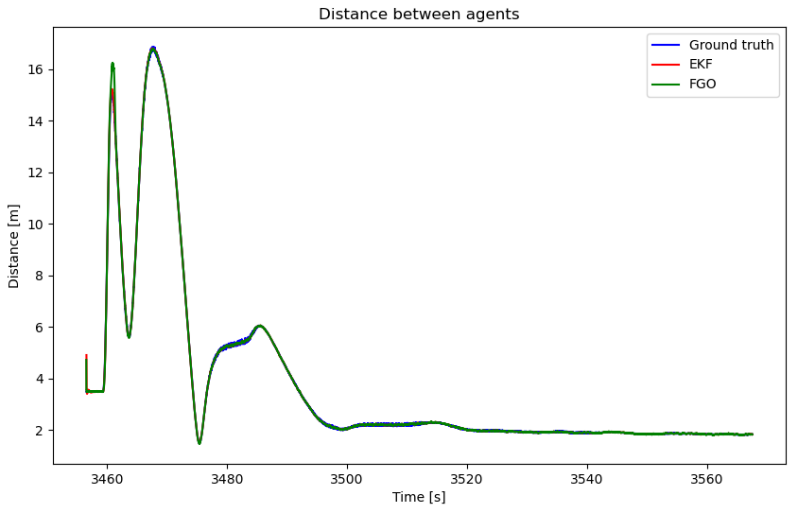


***Figure 7.10:*** *Estimated distance between the agents for the close formation flight. The distance is calculated from the relative position estimates.*

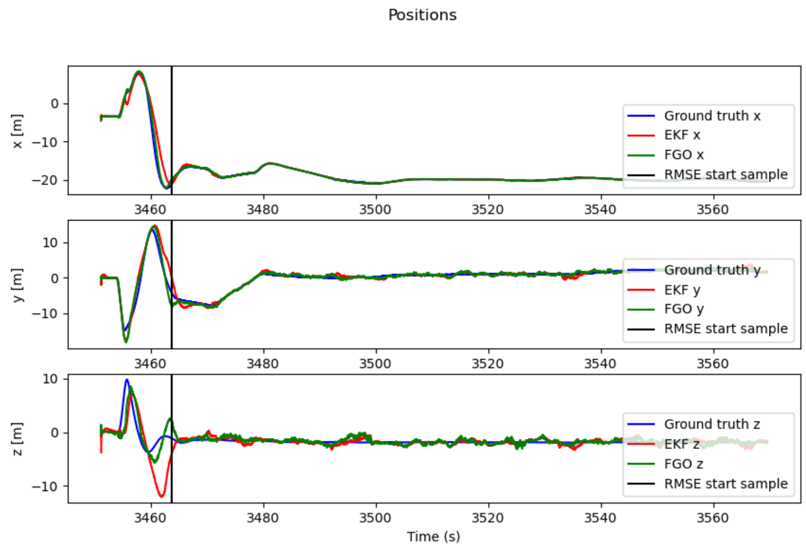


***Figure 7.11:*** *Relative position estimates for the distant formation flight. The vertical black line denotes the sample from which the RMSE calculations start.*

### 7.2.2 Distant formation flight

In Figure 7.11-7.13 the plots of the estimated relative position, relative velocity, and relative orientation for the distant formation flight scenario are presented, and in Table 7.4 the RMSE of the estimates is presented. The distances between the agents are calculated from the estimates and is shown in Figure 7.14. The estimates of the relative position and the relative velocity are comparably similar. The FGO estimate displays some small bumps along the relative orientation estimate.

***Table 7.4:*** *RMSE for distant formation flight.*

| Estimate | Position [m] | Velocity [m/s] | Roll, Pitch, Yaw [rad] |
|---|---|---|---|
| EKF | 0.755 | 0.758 | 0.0052 |
| FGO | 0.669 | 0.859 | 0.011 |

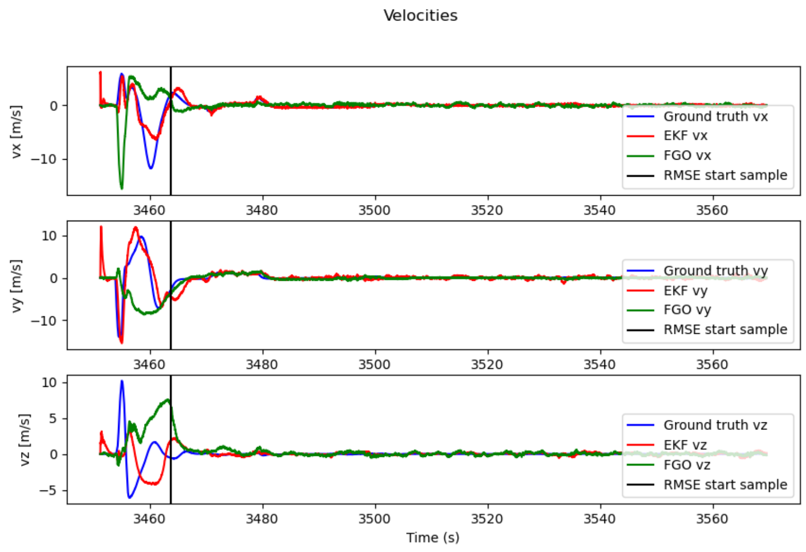


***Figure 7.12:*** *Relative velocity estimates for the distant formation flight. The vertical black line denotes the sample from which the RMSE calculations start.*

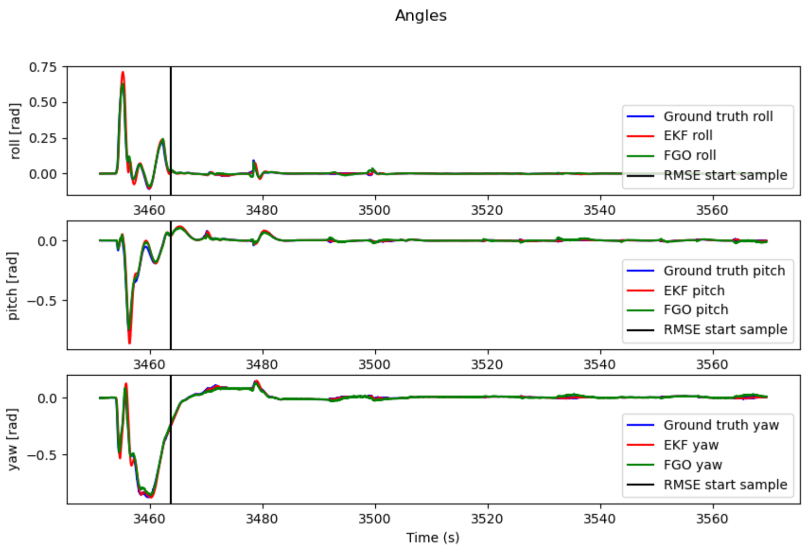


***Figure 7.13:*** *Relative orientation estimates for the distant formation flight. The vertical black line denotes the sample from which the RMSE calculations start.*

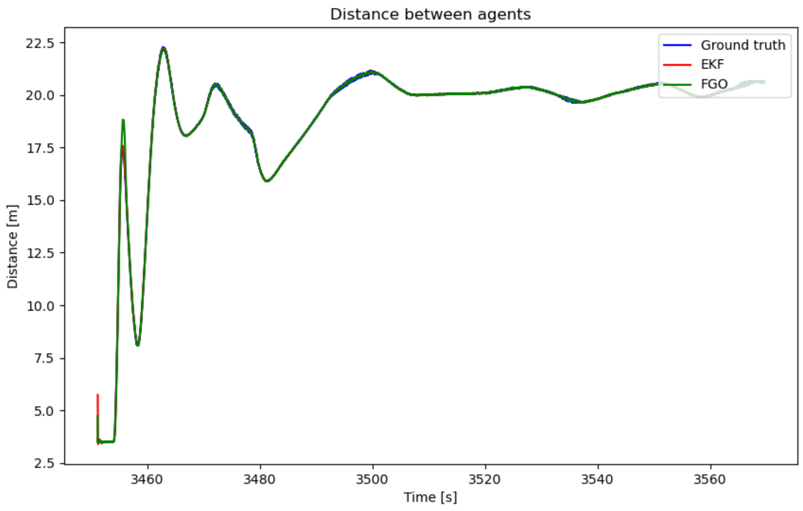


***Figure 7.14:*** *Estimated distance between the agents for the distant formation flight. The distance is calculated from the relative position estimates.*

## 7.2.3 Noisy formation flight

In Figure 7.15-7.17 the plots of the estimated relative position, relative velocity, and relative orientation for the noisy formation flight scenario is presented, and in Table 7.4 the RMSE of the estimates is presented. The distance between the agents is calculated from the estimates and is shown in Figure 7.18. The FGO estimates the position slightly better than the EKF in this scenario, but has a poor estimate of the relative velocity in comparison to the EKF as seen in Table 7.5 and Figure 7.15. The estimates of the relative orientation are near-identical.

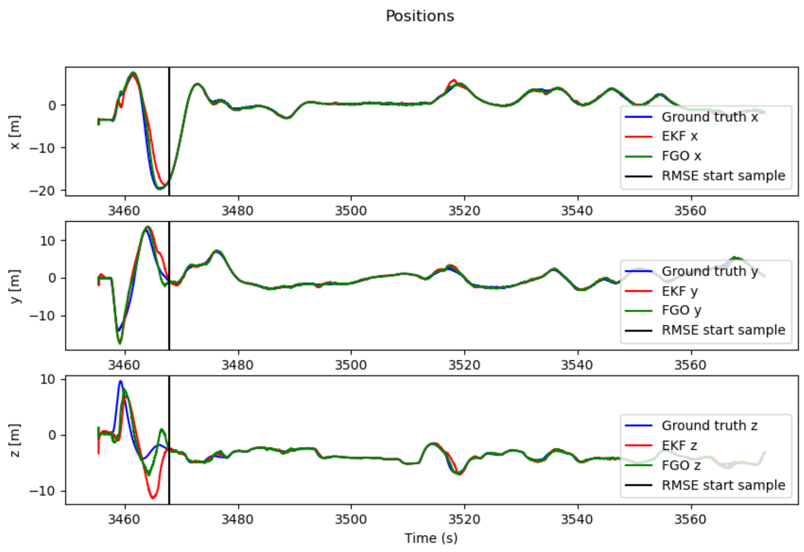


***Figure 7.15:*** *Relative position estimates for the noisy formation flight. The vertical black line denotes the sample from which the RMSE calculations start.*

***Table 7.5:*** *RMSE for noisy formation flight*

| Estimate | Position [m] | Velocity [m/s] | Roll, Pitch, Yaw [rad] |
|---|---|---|---|
| EKF | 0.428 | 0.845 | 0.016 |
| FGO | 0.275 | 2.010 | 0.013 |

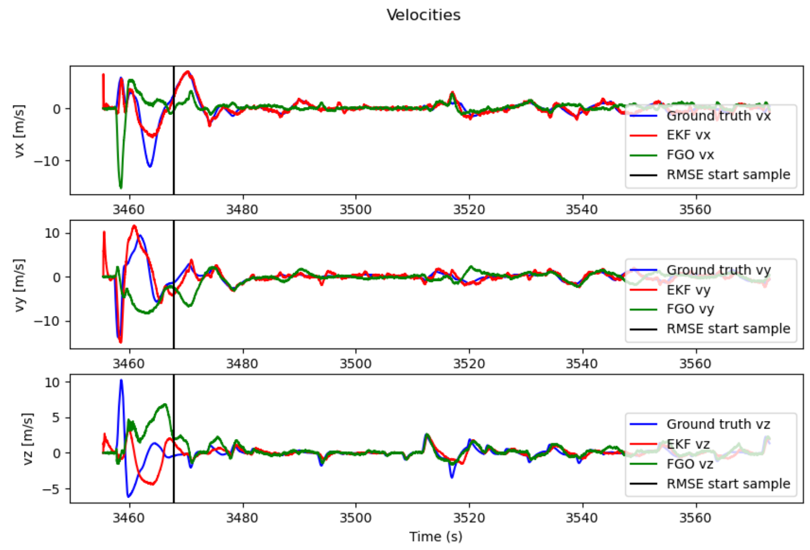


***Figure 7.16:*** *Relative velocity estimates for the noisy formation flight. The vertical black line denotes the sample from which the RMSE calculations start.*

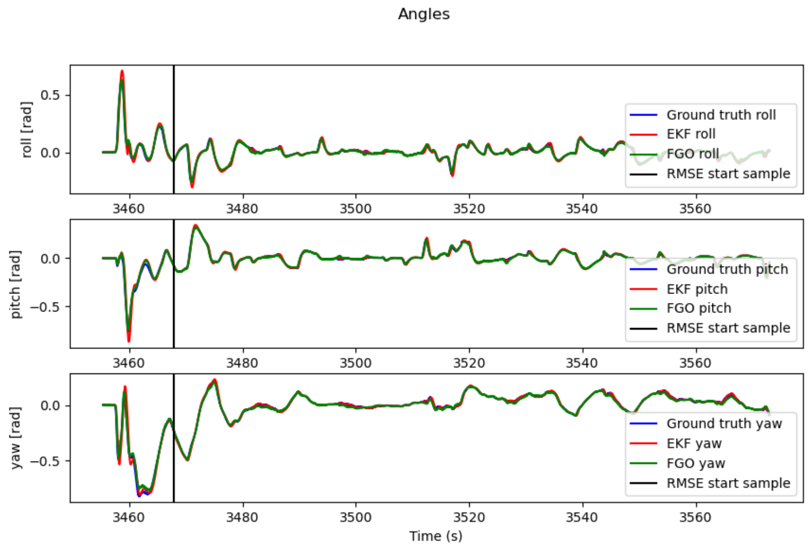


***Figure 7.17:*** *Relative orientation estimates for the noisy formation flight. The vertical black line denotes the sample from which the RMSE calculations start.*

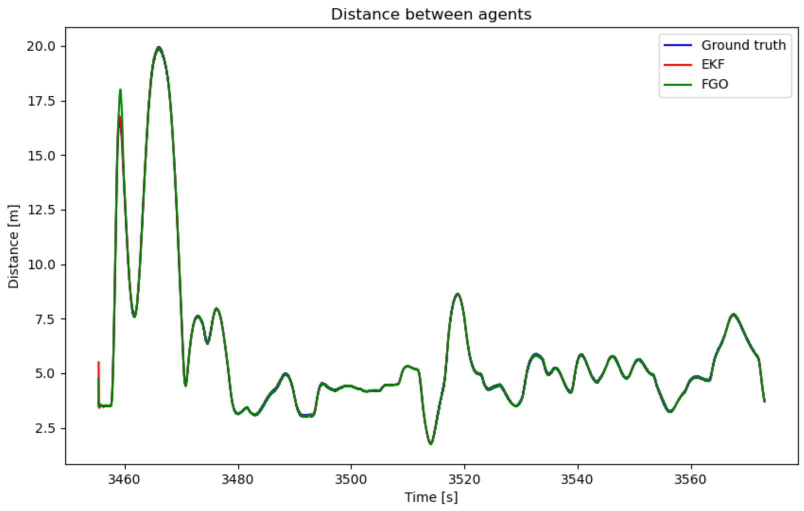


***Figure 7.18:*** *Estimated distance between the agents for the noisy formation flight. The distance is calculated from the relative position estimates.*

### 7.2.4 Calibration errors and missed measurements

From the results in Section 7.1.2, static noises are added to each range measurement in the noisy formation flight according to Table 7.2, as described in Section 6.2.3. In Figure 7.19-7.21 the estimated relative states are presented, and in Table 7.6 the RMSE for all estimates are presented. These estimations are generated assuming that the standard deviation of the range measurements is $\sigma_{UWB} = 0.06$ m. In Figure 7.22-7.24, and Table 7.7, the results from the filters are presented, but in this case, the standard deviation used in the EKF and the FGO is increased to $\sigma = 0.15$ m.

With $\sigma_{UWB} = 0.06$ m, the FGO estimate is significantly worse than the EKF estimate. When increasing the used standard deviation, the RMSE for the FGO position estimate decreases to approximately the same as the EKF position estimate in Table 7.6. When increasing the standard deviation, the RMSE for the EKF position and velocity estimates increases slightly. Overall, the estimates are worse than for the standard scenario presented in Section 7.2.3, which is expected since the measurements deviate more from the model.

***Table 7.6:*** *RMSE for noisy formation flight with added calibration error and missed measurements to range measurements. In the filters, the standard deviation of the measurements is assumed to be 0.06 m.*

| Estimate | Position [m] | Velocity [m/s] | Roll, Pitch, Yaw [rad] |
|---|---|---|---|
| EKF | 0.860 | 0.862 | 0.016 |
| FGO | 2.893 | 2.353 | 0.014 |

***Table 7.7:*** *RMSE for noisy formation flight with added calibration error and missed measurements to range measurements. In the filters, the standard deviation of the measurements is assumed to be 0.15 m.*

| Estimate | Position [m] | Velocity [m/s] | Roll, Pitch, Yaw [rad] |
|---|---|---|---|
| EKF | 1.171 | 0.971 | 0.016 |
| FGO | 0.839 | 2.030 | 0.014 |

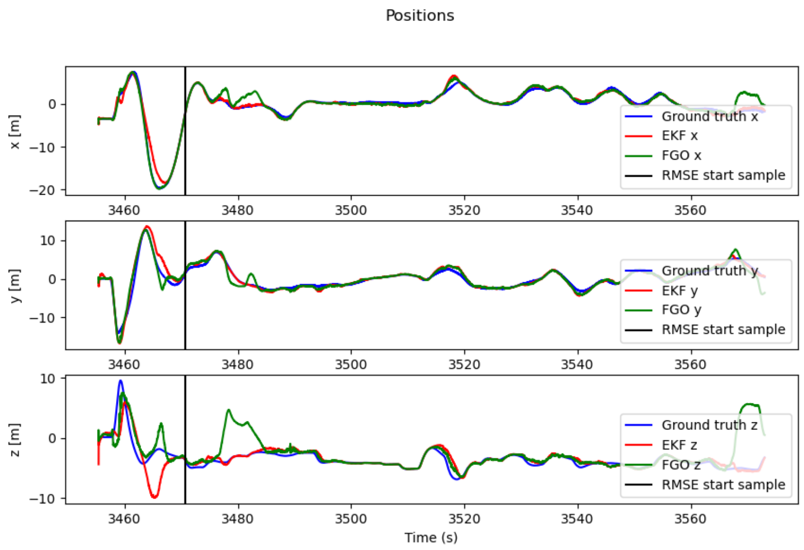


***Figure 7.19:*** *Relative position estimates for scenario with added static noise and missed measurements. The standard deviation used in the EKF and the FGO is $\sigma_{UWB} = 0.06$. The vertical black line denotes the sample from which the RMSE calculations start.*

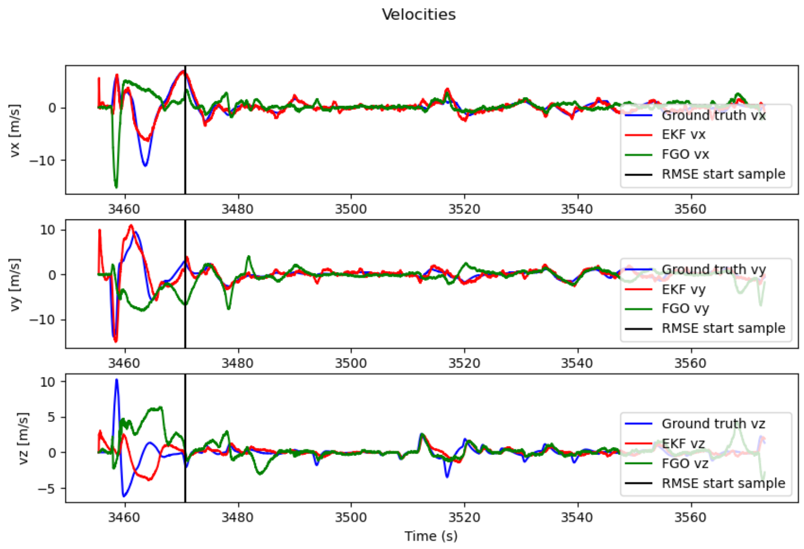


***Figure 7.20:*** *Relative velocity estimates for scenario with added static noise and missed measurements. The standard deviation used in the EKF and the FGO is $\sigma_{UWB} = 0.06$. The vertical black line denotes the sample from which the RMSE calculations start.*

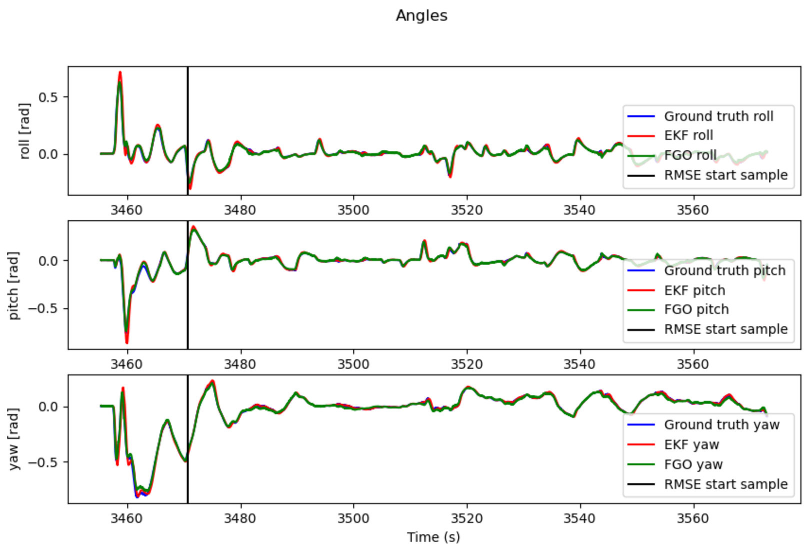


***Figure 7.21:*** *Relative orientation estimates for scenario with added static noise and missed measurements. The standard deviation used in the EKF and the FGO is $\sigma_{UWB} = 0.06$. The vertical black line denotes the sample from which the RMSE calculations start.*

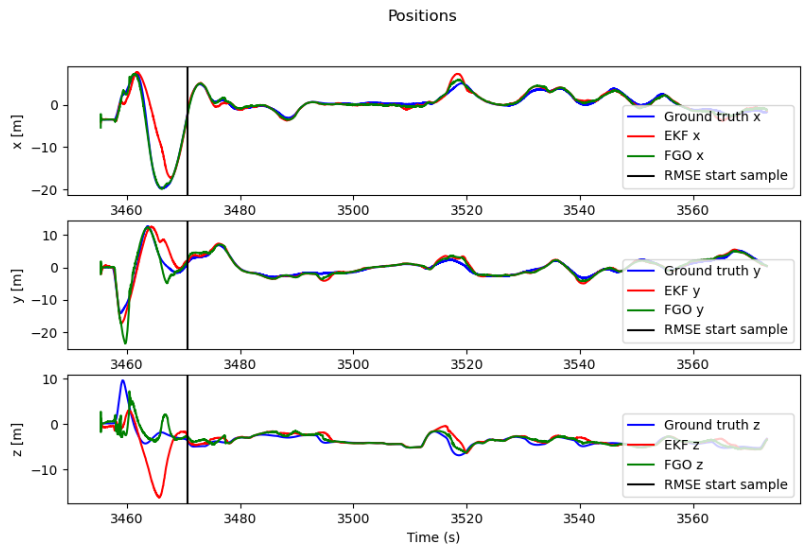


***Figure 7.22:*** *Relative position estimates for scenario with added static noise and missed measurements. The standard deviation used in the EKF and the FGO is $\sigma = 0.15$. The vertical black line denotes the sample from which the RMSE calculations start.*

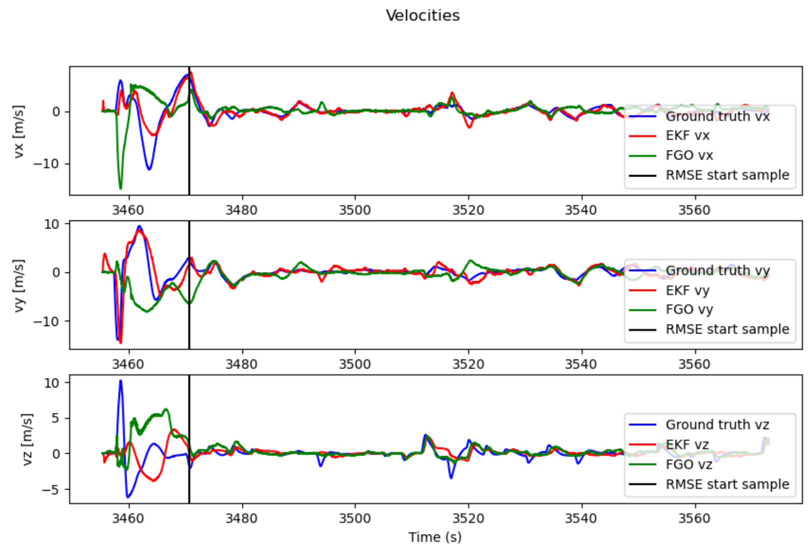


***Figure 7.23:*** *Relative velocity estimates for scenario with added static noise and missed measurements. The standard deviation used in the EKF and the FGO is $\sigma = 0.15$. The vertical black line denotes the sample from which the RMSE calculations start.*

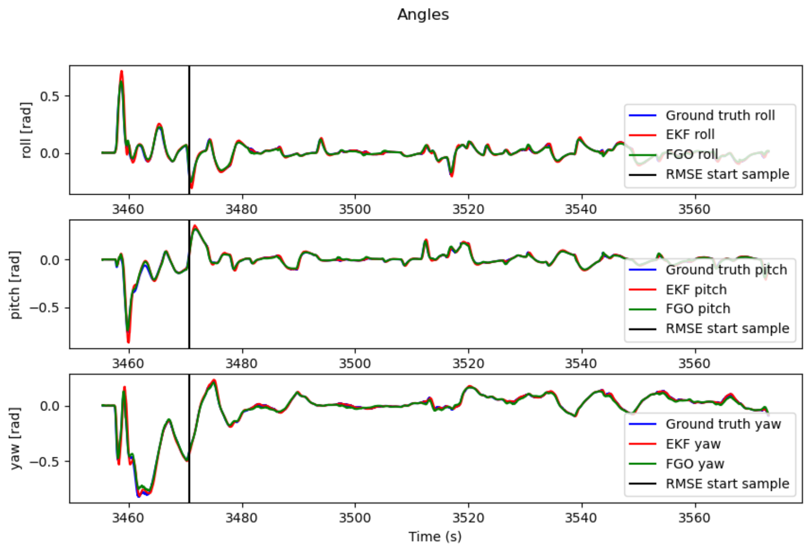


***Figure 7.24:*** *Relative orientation estimates for scenario with added static noise and missed measurements. The standard deviation used in the EKF and the FGO is $\sigma = 0.15$. The vertical black line denotes the sample from which the RMSE calculations start.*

## 7.3 Node constellations

In this section, the results from different constellations of tags are presented. In Section 7.3.1, both filters are examined with different number of tag on the leader, and in Section 7.3.2, different placements of the tags are examined. All results in this section are generated from the noisy formation flight.

### 7.3.1 Number of tags

Figure 7.25-7.27 presents the estimates when only one tag is used, and Figure 7.28-7.30 presents the estimates when three tags are used. In Table 7.8 the RMSE for one, two and three tags are presented for both filters. The results for two tags are taken from Table 7.5.

As the number of tags increase in the system, the frequency that all tags performs measurements is decreased. In the standard case, with two tags, this frequency is 40 Hz. With one and three tags, this frequency becomes 80 Hz and 26.8 Hz, respectively. However, the frequency at which the filter receives measurements from a tag is 80 Hz, no matter how many tags are in the system. In Table 7.8, a decrease in RMSE of the relative position estimate is observed for both filters. The relative velocity estimate becomes better for the EKF when increasing to three tags, but no significant difference can be seen for the FGO. Throughout all of these results, the RMSE for the orientation estimates are unchanged.

***Table 7.8:*** *RMSE for the noisy formation flight with different number of tags. Node positions are specified in Table 6.1.*

| Estimate | Tags | Position [m] | Velocity [m/s] | Roll, Pitch, Yaw [rad] |
|---|---|---|---|---|
| EKF | T1 | 0.430 | 0.845 | 0.016 |
| EKF | T1, T2 | 0.428 | 0.845 | 0.016 |
| EKF | T1, T2, T3 | 0.343 | 0.780 | 0.016 |
| FGO | T1 | 0.338 | 2.028 | 0.014 |
| FGO | T1, T2 | 0.275 | 2.010 | 0.013 |
| FGO | T1, T2, T3 | 0.255 | 2.010 | 0.014 |

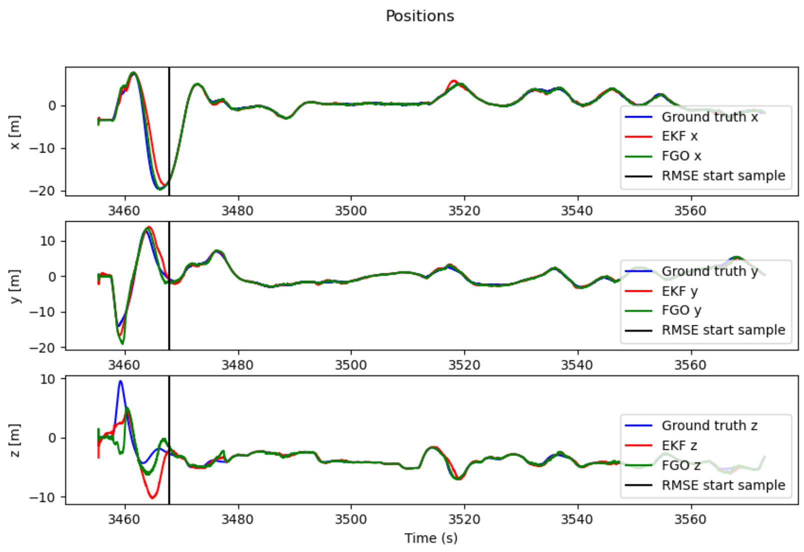


***Figure 7.25:*** *Relative position estimates using T1. The vertical black line denotes the sample from which the RMSE calculations start.*

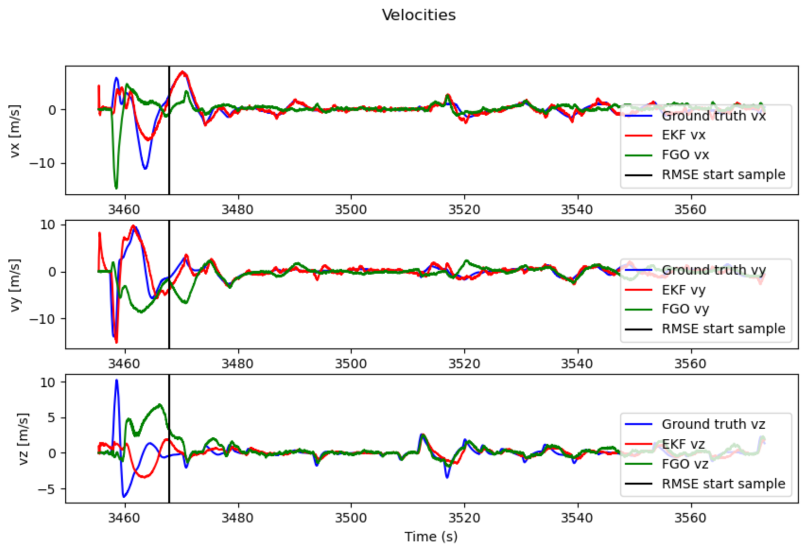


***Figure 7.26:*** *Relative velocity estimates using T1. The vertical black line denotes the sample from which the RMSE calculations start.*

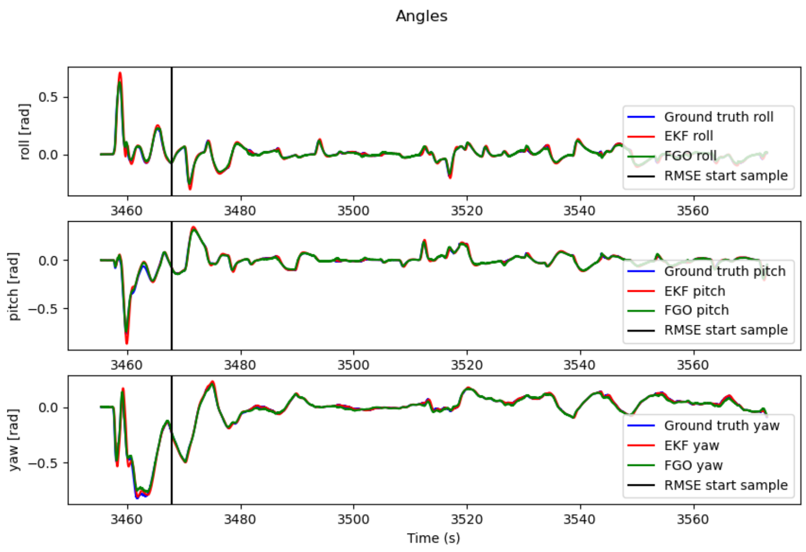


***Figure 7.27:*** *Relative orientation estimates using T1. The vertical black line denotes the sample from which the RMSE calculations start.*

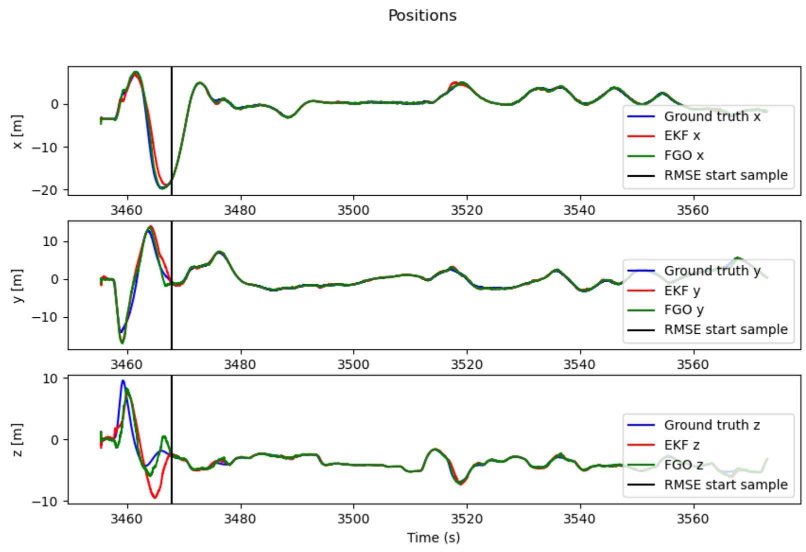


***Figure 7.28:*** *Relative position estimates using T1, T2 and T3. The vertical black line denotes the sample from which the RMSE calculations start.*

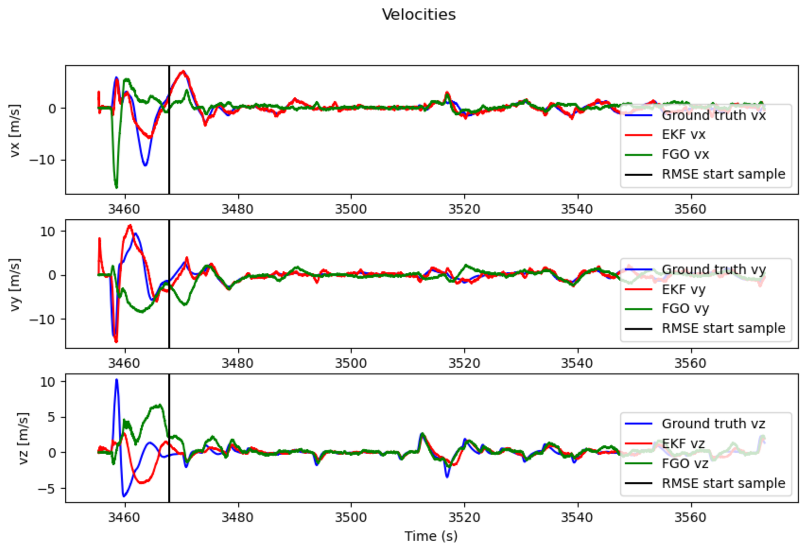


*Figure 7.29: Relative velocity estimates using T1, T2 and T3. The vertical black line denotes the sample from which the RMSE calculations start.*

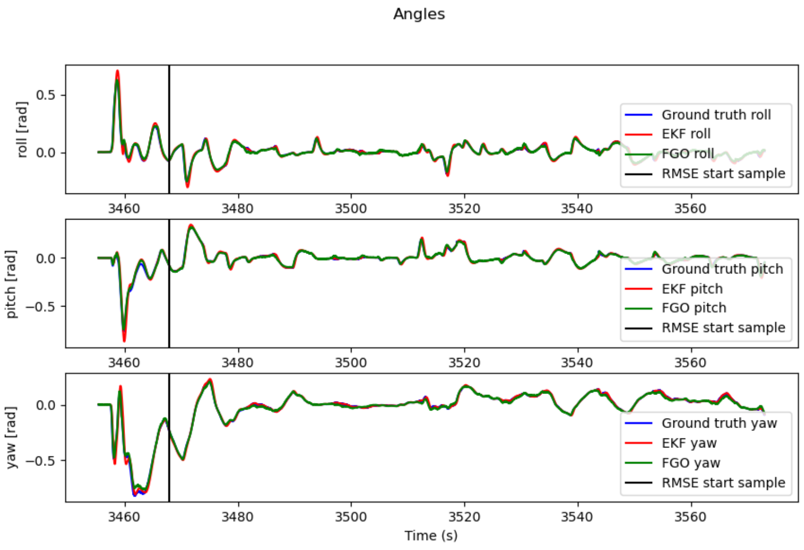


*Figure 7.30: Relative orientation estimates using T1, T2 and T3. The vertical black line denotes the sample from which the RMSE calculations start.*

### 7.3.2 Tag placement

In this section, two cases with two different tag placements are considered. In the first case, one tag is placed on the leader, and in the second, two tags are used. The tags are placed according to Table 6.1.

In the first case, the standard placement for one tag, as presented in Section 7.3.1, is compared to estimates shown in Figure 7.31-7.33 where the tag is placed closer to the origin of the leader. The RMSE for both placements are presented in Table 7.9. For both the EKF and the FGO, the second placement of the tag results in worse position and velocity estimates. The RMSE for the orientations are unchanged.

In the second case, where two tags are used with two different placements, the resulting estimates are presented in Section 7.2.3 and Figure 7.34-7.36, and Table 7.9 contains the resulting RMSE. For the EKF, the RMSE of the position and velocity estimates are lower for the second configuration, but for the FGO, the results are the opposite.

***Table 7.9:*** *RMSE for the noisy formation flight with different tag placements. Node positions are specified in Table 6.1.*

| Estimate | Tags | Position [m] | Velocity [m/s] | Roll, Pitch, Yaw [rad] |
|---|---|---|---|---|
| EKF | T1 | 0.430 | 0.845 | 0.016 |
| EKF | T4 | 0.522 | 0.893 | 0.016 |
| FGO | T1 | 0.338 | 2.028 | 0.014 |
| FGO | T4 | 0.364 | 2.053 | 0.014 |
| EKF | T1, T2 | 0.428 | 0.845 | 0.016 |
| EKF | T1, T3 | 0.381 | 0.795 | 0.016 |
| FGO | T1, T2 | 0.275 | 2.010 | 0.013 |
| FGO | T1, T3 | 0.305 | 2.018 | 0.014 |

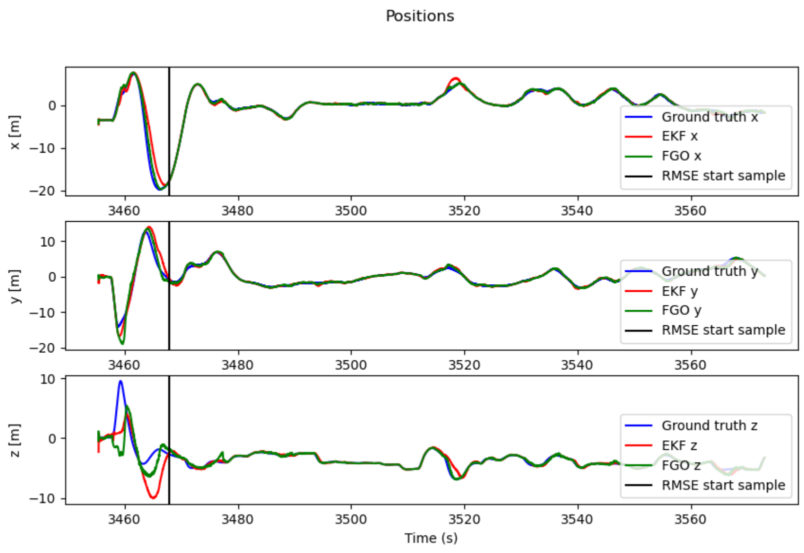


***Figure 7.31:*** *Relative position estimates using T4. The vertical black line denotes the sample from which the RMSE calculations start.*

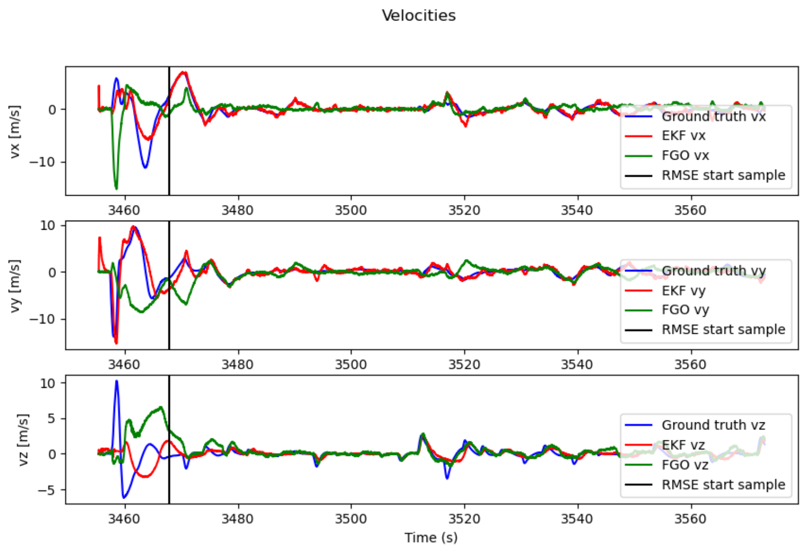


***Figure 7.32:*** *Relative velocity estimates using T4. The vertical black line denotes the sample from which the RMSE calculations start.*

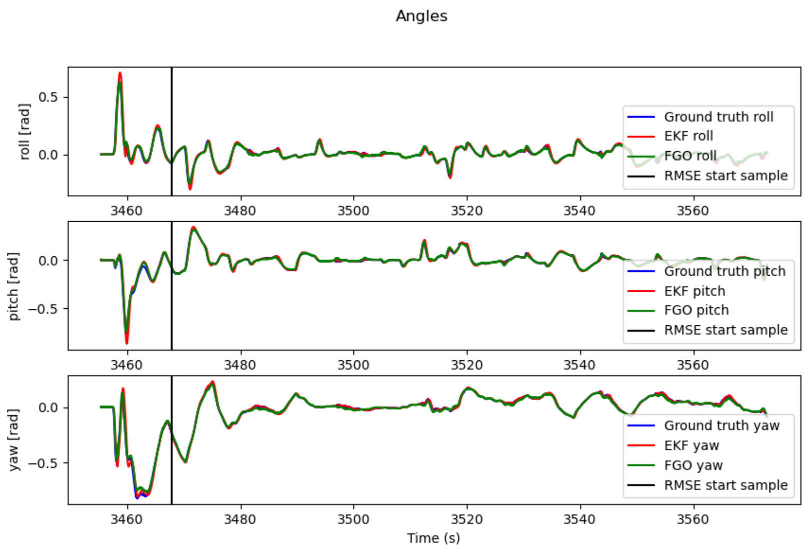


***Figure 7.33:*** *Relative orientation estimates using T4. The vertical black line denotes the sample from which the RMSE calculations start.*

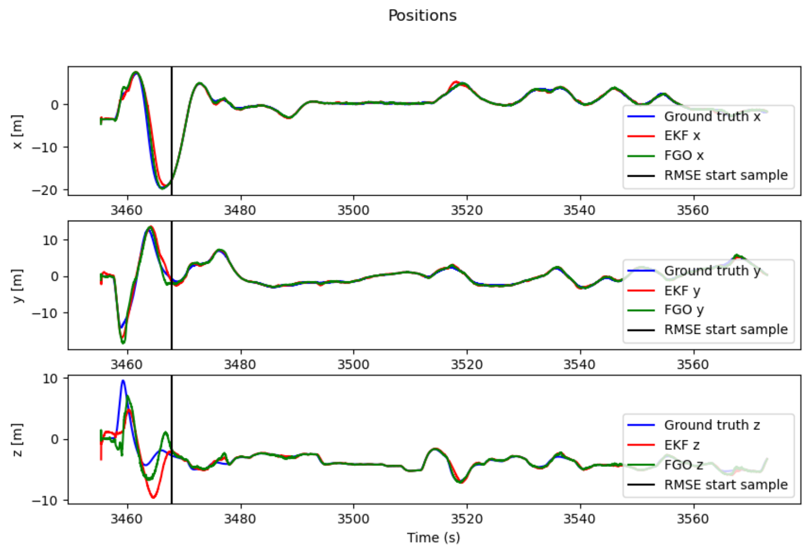


***Figure 7.34:*** *Relative position estimates using T1 and T3. The vertical black line denotes the sample from which the RMSE calculations start.*

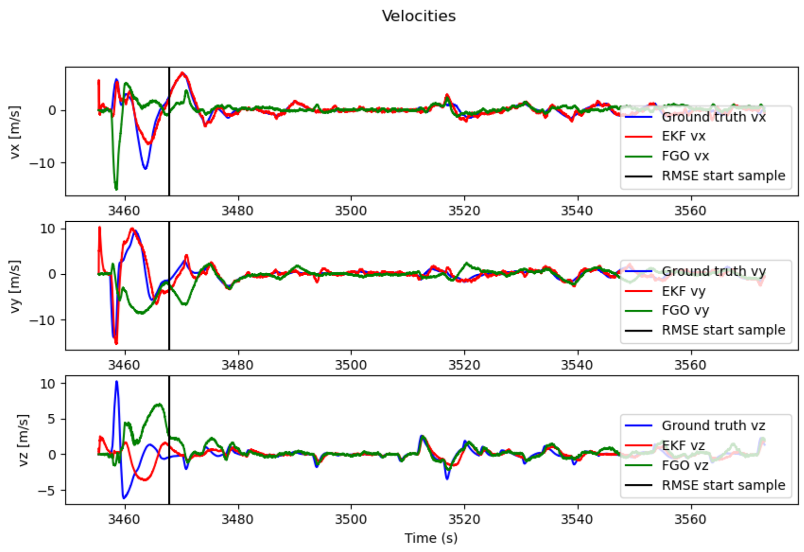


***Figure 7.35:*** *Relative velocity estimates using T1 and T3. The vertical black line denotes the sample from which the RMSE calculations start.*

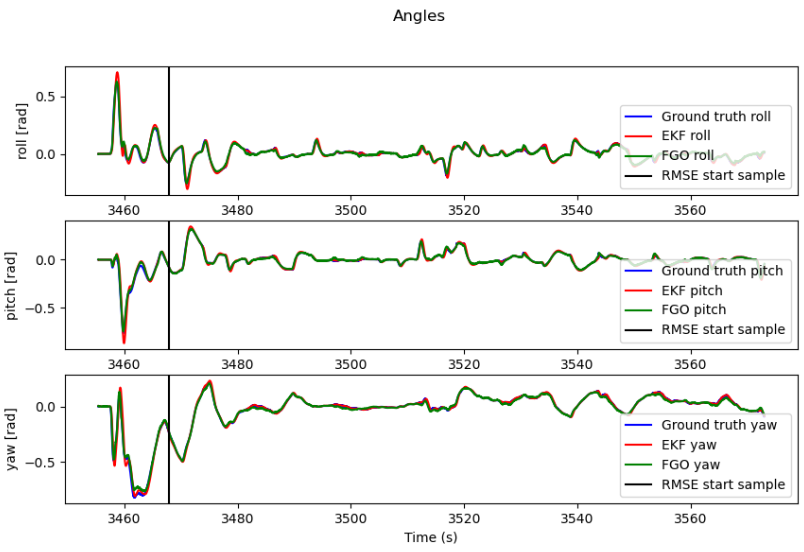


***Figure 7.36:*** *Relative orientation estimates using T1 and T3. The vertical black line denotes the sample from which the RMSE calculations start.*

## 7.4 Investigation of additional sensors

In this section, the results from the investigation into the additional sensors in the FGO algorithm are presented. One difference between the investigated method and the standard case is that in the investigated method, the individual pose and velocity of the leader is estimated, yielding six additional plots. The dataset used was gathered using the same noisy route as in Figure 7.18, but while publishing and saving IMU and barometer data. The RMSE of the comparison between the ground truth and the estimates is presented in Table 7.10. In Figure 7.37-7.39 the modified version of the base case using a barometer factor is compared to the base case. The standard version using the barometer factor is used to evaluate the local position case estimation algorithm.

The results in Figure 7.37-7.39 and Table 7.10 show that the estimate employing the barometer has slightly better position estimation, especially in the $z$-position, but is otherwise similar to the estimate without the barometer.

In Figure 7.40-7.42 the base case using the barometer is compared to the local position case algorithm. The base case using the barometer factor has a slight edge over the local position case algorithm. On the other hand, the difference between the estimated relative velocity is very large, with the algorithm utilizing the unfiltered accelerometer data clearly performing better. The relative orientation estimates are almost exactly the same.

Finally, in Figure 7.43-7.48 the estimate of the leader and follower from the local position case is evaluated against ground truth data. The plots show that the estimated $z$-position, which is supported by the barometer factor, does not drift, but the $x$- and $y$-positions drift during execution of the algorithm.

***Table 7.10:*** *RMSE for flight in which barometer data and IMU data was published. The flight scenario is of the close noisy type as described in Section 6.2.1.*

| Estimate | Position [m] | Velocity [m/s] | Roll, Pitch, Yaw [rad] |
|---|---|---|---|
| FGO no baro | 0.272 | 2.566 | 0.012 |
| FGO with baro | 0.229 | 2.557 | 0.012 |
| FGO local position case | 0.339 | 0.152 | 0.012 |

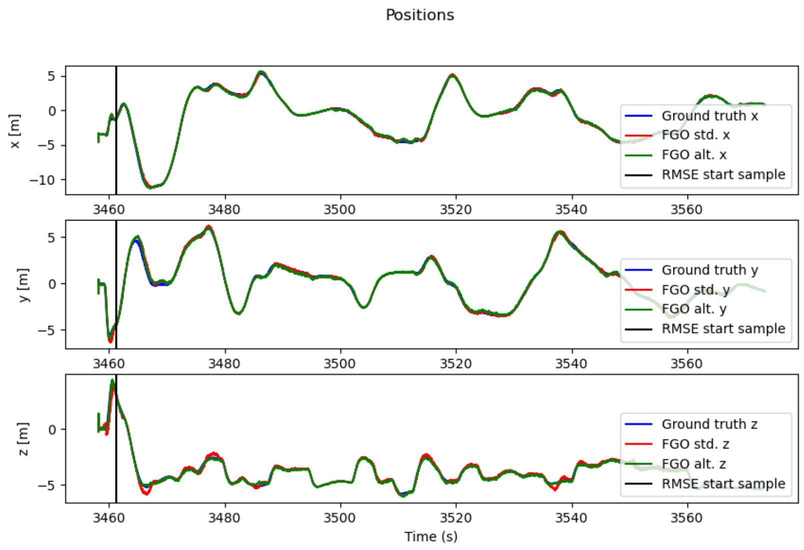


***Figure 7.37:*** *Relative position when using the base case and when using the base case with the barometer factor. The vertical black line denotes the sample from which the RMSE calculations start.*

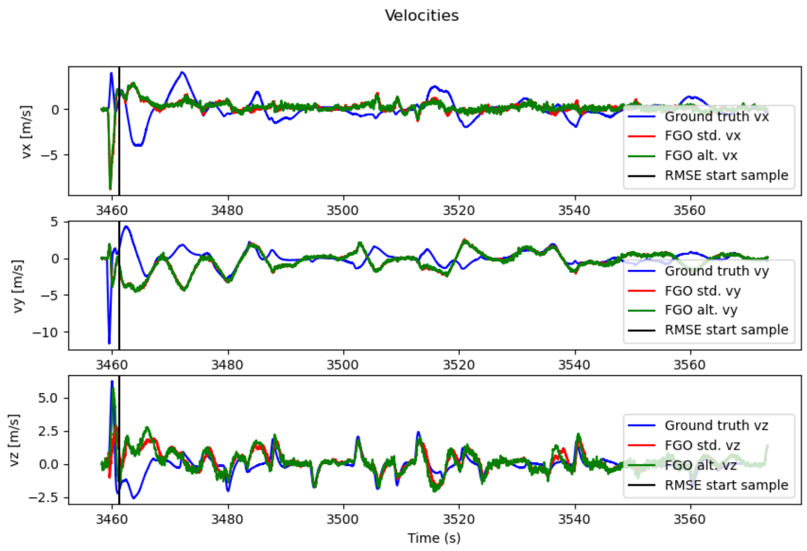


***Figure 7.38:*** *Relative velocity when using the base case and when using the base case with the barometer factor. The vertical black line denotes the sample from which the RMSE calculations start.*

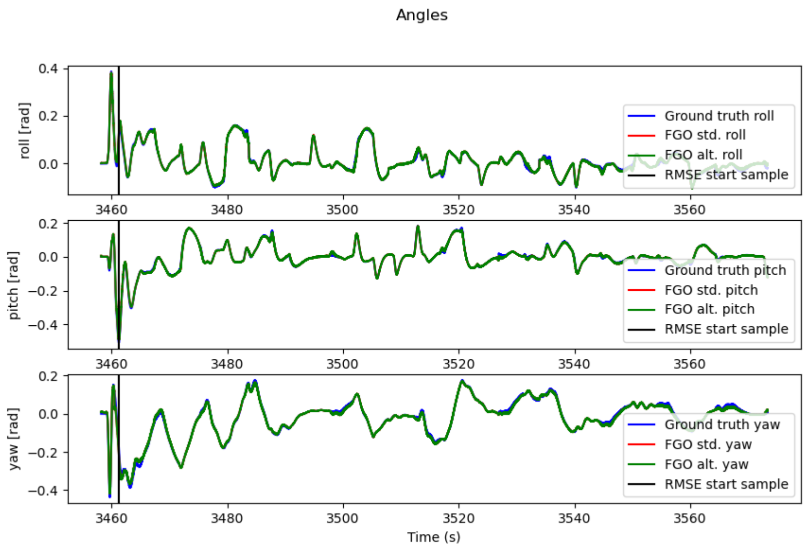


*Figure 7.39:* *Relative orientation when using the base case and when using the base case with the barometer factor. The vertical black line denotes the sample from which the RMSE calculations start.*

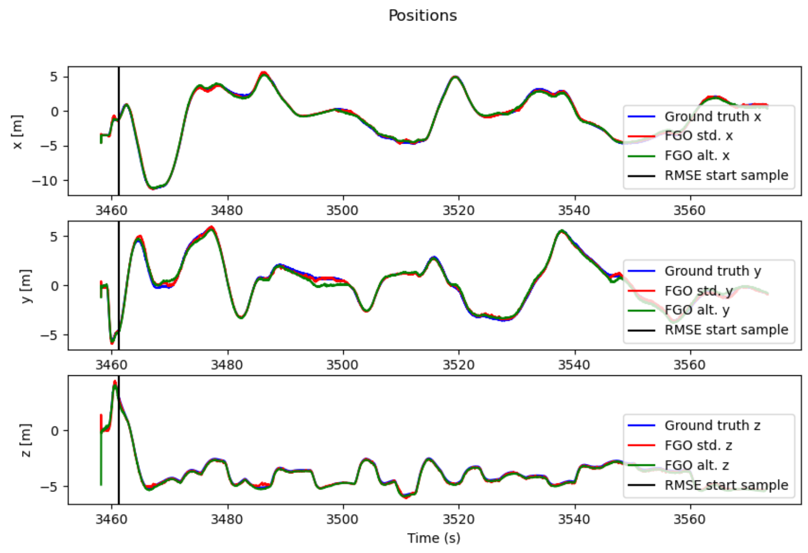


*Figure 7.40:* *Relative position when using the IMU preintegration factor, barometer factor, and UWB factor. The vertical black line denotes the sample from which the RMSE calculations start.*

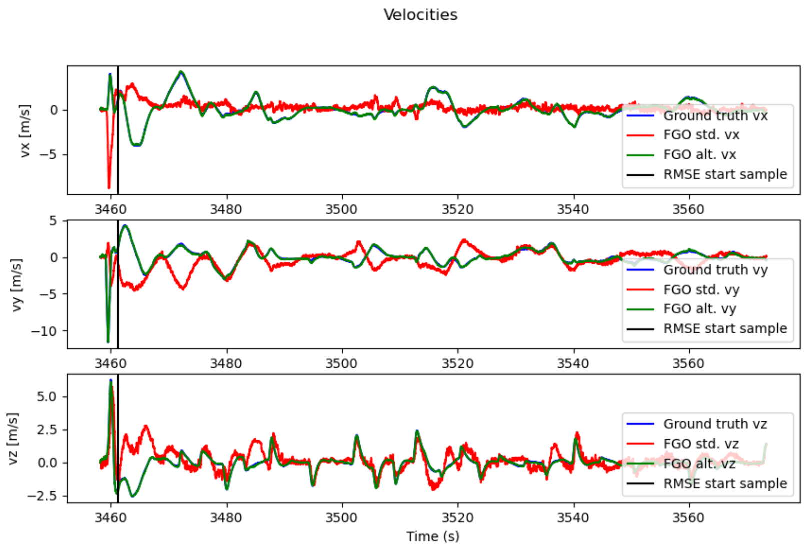


***Figure 7.41:*** *Relative velocity when using the IMU preintegration factor, barometer factor, and UWB factor. The vertical black line denotes the sample from which the RMSE calculations start.*

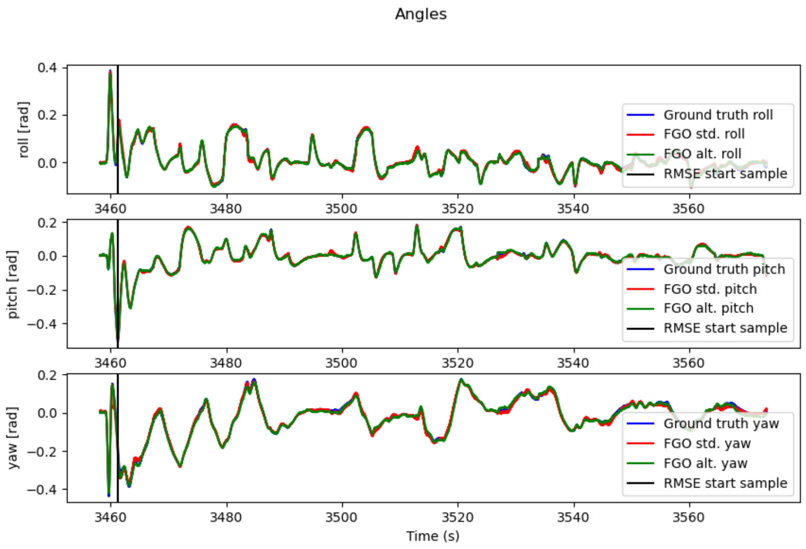


***Figure 7.42:*** *Relative orientation when using the IMU preintegration factor, barometer factor, and UWB factor. The vertical black line denotes the sample from which the RMSE calculations start.*

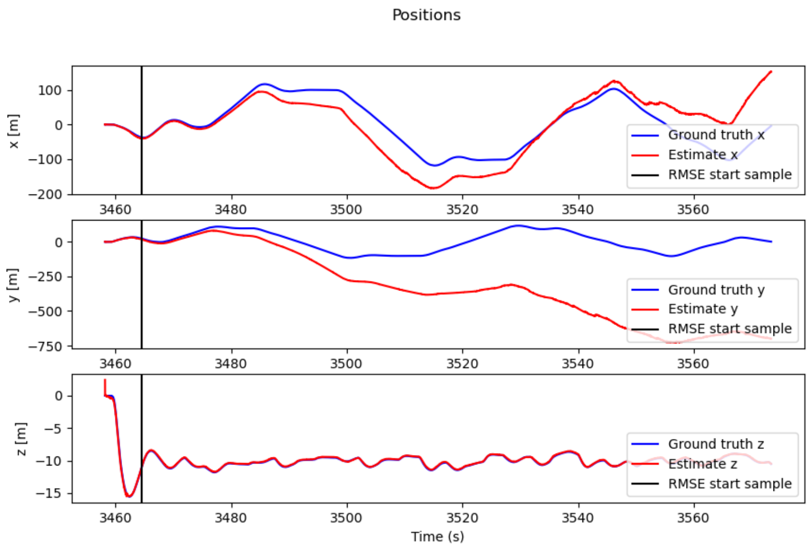


***Figure 7.43:*** *Leader position when using the IMU preintegration factor, barometer factor, and UWB factor. The vertical black line denotes the sample from which the RMSE calculations start.*

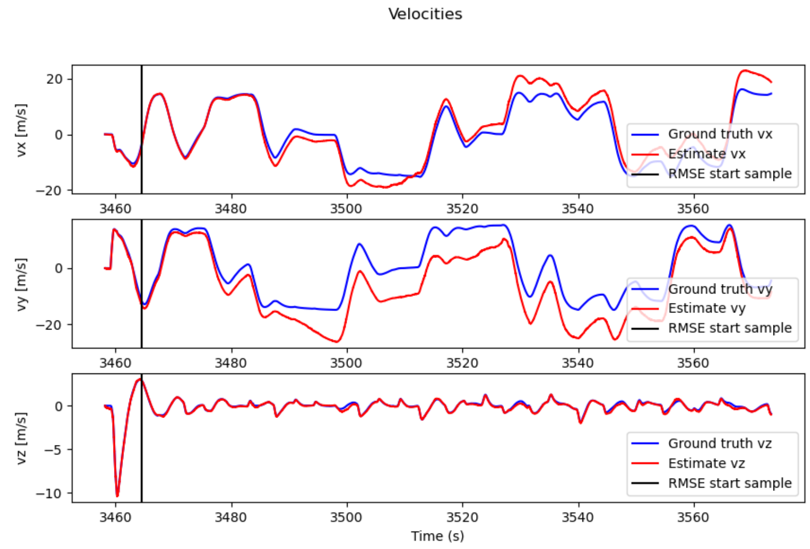


***Figure 7.44:*** *Leader velocity when using the IMU preintegration factor, barometer factor, and UWB factor. The vertical black line denotes the sample from which the RMSE calculations start.*

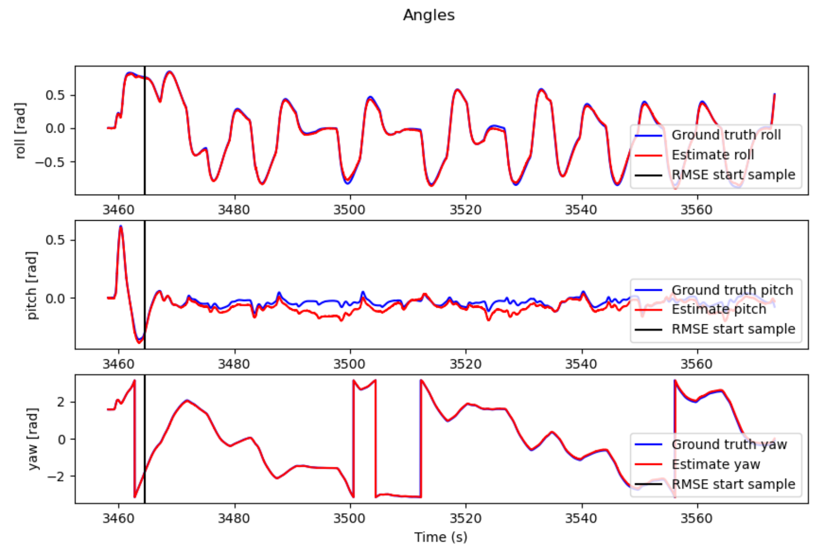


***Figure 7.45:*** *Leader orientation when using the IMU preintegration factor, barometer factor, and UWB factor. The vertical black line denotes the sample from which the RMSE calculations start.*

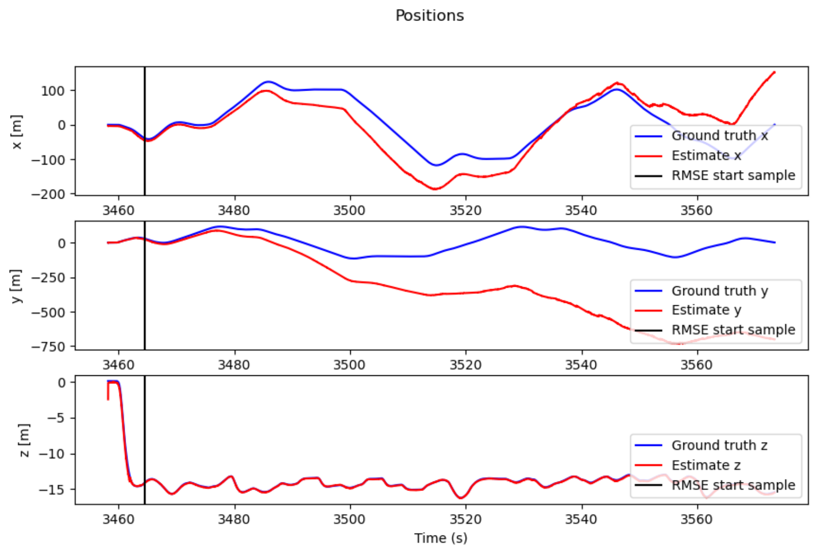


***Figure 7.46:*** *Follower position when using the IMU preintegration factor, barometer factor, and UWB factor. The vertical black line denotes the sample from which the RMSE calculations start.*

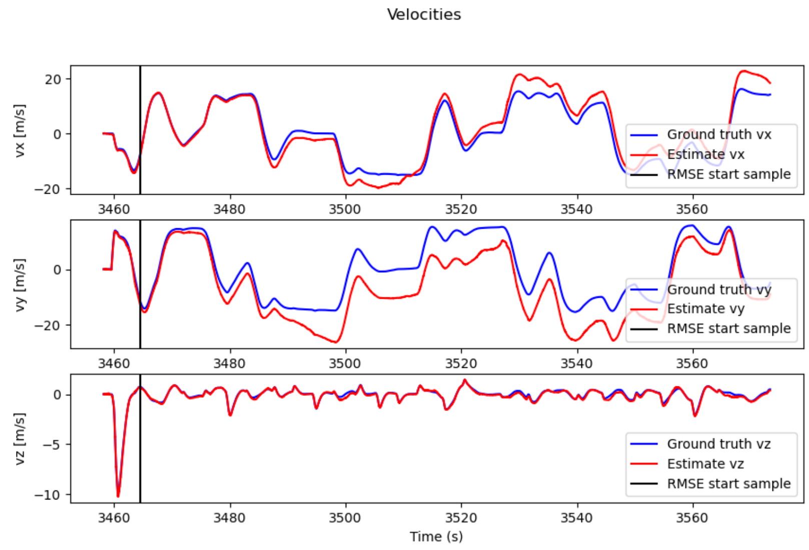


***Figure 7.47:*** *Follower velocity when using the IMU preintegration factor, barometer factor, and UWB factor. The vertical black line denotes the sample from which the RMSE calculations start.*

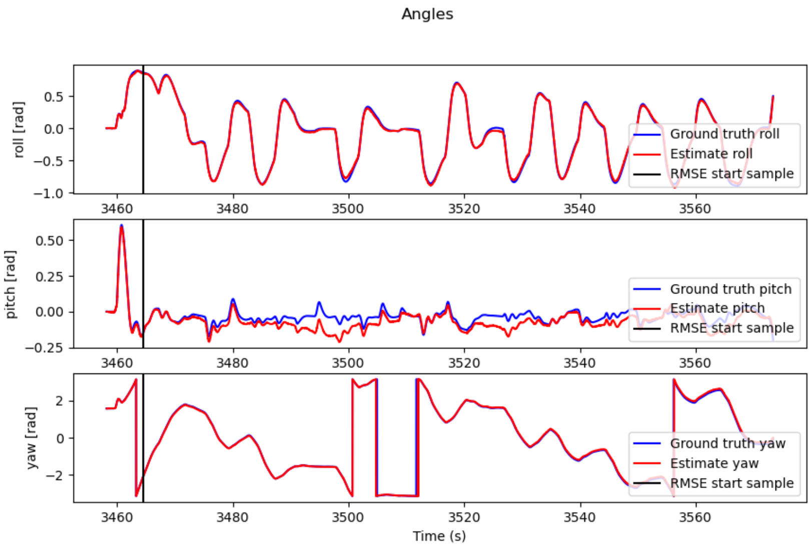


***Figure 7.48:*** *Follower orientation when using the IMU preintegration factor, barometer factor, and UWB factor. The vertical black line denotes the sample from which the RMSE calculations start.*

# 8 Discussion

In this chapter, the results presented in Chapter 7 regarding the hardware and performance in simulation are discussed.

## 8.1 Antenna calibration

From the results in Section 7.1, the expected accuracy from the calibration methods used is a static error of a few centimeters. When the antenna delay is set in the firmware of the UWB nodes, similar results are obtained. This confirms that the model (3.1) is somewhat accurate and that the method can be used to get a static error of a few centimeters. For the second method, the accuracy of the measurement between the calibrated device and the referece device is the same as the first method. However, when measuring the distance to a device other than the reference device, the accuracy decreases. In this case, the calibration error from T1 is compensated for in the calibration of A2. This error is present when measuring the distance between T2 and A2, where the error becomes larger. This can be prevented by calibrating all devices using the first method. However, this requires five sets of measurements for each device, and a total of 30 measurement sets. When combining the two methods, only nine measurement sets are required. However, by using the method that the first method is based on, [24], only one set of measurements between each device is required, resulting in a total of 15 sets of measurements.

Overall, the results are worse than the results in [23] presented in Section 2.1.4, but since the method used in this thesis is more simple, this is expected. Roughly the same performance is achieved as in [24], thus it could be enough to only measure the distance between two devices once. If the calibration was to be automated, there is a point in requiring less measurements to speed up this process, for example if the system would calibrate itself on startup. However, since the accuracy of the measurements heavily rely on the calibration, a more advanced method as in [23] may be the best option in either case.

The results presented in Appendix A show significant measurement errors for the UWB nodes when the orientation of the antenna changes. The errors change from a few centimeters to almost half a meter when the nodes are rotated. This indicates that the UWB nodes used are sensitive to changes in orientation of the antenna, which must be considered during calibration and deployment. Since the measured distance in Figure A.1-A.6 is lower than the ground truth in most cases, it is possible to conclude that the nodes are rotated in a non-optimal way during calibration. A measured distance that is shorter than expected indicates that the estimated antenna delays are too large, resulting in $t_{TOF}$ being too small in (3.1). During calibration, the measured time, $t_{meas}$, is larger than it should be because of the orientation of the nodes, resulting in estimated antenna delays that are too large.

If the measured range depends on the relative orientation between the anchors and the tags, in addition to the measurement noise and the static noise caused by the antenna calibration error, the measurements are harder to model. This could cause problems when estimating the relative position. A solution to this problem could be to examine the performance of other antennas. Another solution could be to find the appropriate orientation of the antennas, and limiting the formation flight to only fly in formations where the range measurements are accurate. However, this is a significant limitation of the system and should be avoided.

## 8.2 Simulation

In this section, the results from the simulations are discussed in the same order as they are presented in Section 7.2.

### 8.2.1 Close, distant and noisy formation flights

For the close formation flight, the RMSE of the EKF and the FGO estimates are calculated as 0.165 m and 0.170 m respectively. These are significantly lower than for the estimates for both the distant and the noisy formation flight. Two different factors play a part in this good result, the short distance between the agents and the close to constant relative velocity. At a short distance, it is possible to achieve a more accurate position estimation than at larger distances. This is a result of the theory described in Section 2.1.2. In the two-dimensional scenario, each range measurement corresponds to a circle with some uncertainty where the tag could be located. At a shorter distance, the intersecting areas from different measurements where the target could be located is smaller than at larger distances. Thus, the estimates are better for the close formation flight than for the distant formation flight. When comparing the close formation flight to the noisy formation flight, the reason the first performs significantly better is that the flight more closely resembles the constant velocity model. In Figure 7.8 small changes in velocity are observed compared to Figure 7.15, where larger changes in velocity are present. Thus, the noisy formation flight deviates more from the model, resulting in less accurate estimations than for the close formation flight. As seen in Figure 7.10, the distance between the agents is approximately two meters during formation flight. This could be deemed as too close since the wingspan of each agent is two meters. Thus, this is not a realistic scenario.

The results for the distant formation flight are good, considering that the distance between the agents is around 20 meters, as seen in Figure 7.14. However, the position estimates are highly relying on the fact that the orientation estimates for each agent are accurate. A lower accuracy in the estimated orientations would result in larger error for the position estimation, since these are highly correlated. In Figure 7.11, larger errors are observed for the estimates of the position along the $z$-axis, than for the other axes. As the UWB anchors are placed such that the maximum distance between two anchors along the $z$-axis is only 0.22 m, compared to 0.65 m and 1.84 m for the $x$-axis and the $y$-axis, the accuracy of the estimations along this axis is decreased, assuming that both agents are flying horizontally. This problem is hard to prevent using only UWB nodes. To increase the accuracy, the distance between the anchors along the $z$-axis must be increased, but that would require a node being placed on a stick above the aircraft or on one of the tail fins. However, this can impact the aerodynamics of the UAV, causing further problems.

As for the noisy formation flight, the estimates are slightly worse than for the close formation flight, as described previously. However, the estimate is still rather good with respect to the distance between the agents. During the flight, the distance is approximately five meters as seen in Figure 7.18, and the RMSE of the relative position estimates are 0.428 m and 0.275 m for the EKF and the FGO, respectively. In Figure 7.15, the position estimates only deviate from the true position when the relative velocity changes rapidly, as seen in Figure 7.16.

One important detail from the plots is the poor performance of the FGO solution in estimating the relative velocity compared to the EKF solution. One possible cause could be that there is some other unobservable variable which is coupled with the velocity, and that the estimation algorithm controls the unobservable variable through the velocity. Alternatively, since the FGO solver used works like a smoother, and since the motion model is not used for prediction explicitly, there might be an issue in the solver trying to enforce a single constant velocity over a large period, which clearly does not match reality. Finally, there might simply be a small error in the implementation.

One oddity is that the angle estimate of all the estimation algorithms is nearly the same, but that is explained by the fact that all the estimators use the angles from the flight computers as measurement updates with a low associated measurement noise. Therefore, the problem is that the angle estimates from the flight computers are unrealistically good, seeing as there is no drift or particularly much noise affecting the measurements.

### 8.2.2 Calibration errors and missed measurements

With the added calibration errors and missed measurements used in the simulation, the performance decreases for both algorithms. For the EKF estimate, the RMSE of the relative position is doubled, and for the FGO estimate it increases from 0.275 m to 2.893 m. As the range measurements are affected by different static noises, the position is more difficult to estimate. As seen in Figure 7.19, the FGO estimates have large errors during a short period in the first half of the simulation, as well as in the end. The EKF estimates are fairly good most of the time, but as seen for the noisy formation flight in Figure 7.15, there are errors when relative velocity changes rapidly, as seen in Figure 7.20.

With the expected standard deviation of the range measurements set to $\sigma = 0.15$ m, the FGO algorithm performs much better in comparison to when it was set to $\sigma_{UWB} = 0.06$ m. The RMSE of the relative position estimates is only 0.839 m, whilst the EKF performs slightly worse than with the standard deviation set to $\sigma_{UWB} = 0.06$ m. In Figure 7.22, the large errors previously observed for the FGO estimate are gone. However, the errors observed for the EKF estimate have increased slightly.

When it comes to missed measurements, the FGO algorithm is more robust than the EKF. If a measurement is missed, the FGO still adds the other measurements to the graph and only excludes the missed measurement. The EKF on the other hand skips the measurement update if not all measurements are received. Because of this, the FGO is expected to perform better than the EKF in this scenario, but this is not the case. The EKF performs approximately 80% of all measurement updates, which at 40 Hz is 32 measurement updates per second. Thus, the problem is probably caused by the antenna calibration error.

One possible explanation as to why the antenna calibration error affects the FGO more than the EKF, and especially why the performance of the FGO improves so much after increasing $\sigma_{UWB}$, is that the FGO may be tuned to be less reactive. Essentially, it could be the case that the estimator is not reactive enough to move the estimate away from what the faulty measurements indicate. By increasing the noise of the estimate, there is added reactiveness, as the covariance is more evenly spread around the states indicated by the faulty measurements. It is, however, unexpected that the EKF should perform worse when the noise is increased. It could be that the increased covariance results in the EKF trusting the prediction more, and since the model is very uncertain, the accuracy is decreased.

As discussed in this section, the accuracy of the estimates decrease if there are calibration errors present in the range measurements. In Section 7.1.3, results indicate that large errors appear when the UWB nodes are rotated, as seen in Figure A.1-A.6. With these results in mind, it is concluded that in order to achieve good position estimates on hardware, better methods for calibration and a better antenna which distributes the effect more evenly are needed.

## 8.3 Node constellations

In this section, the results regarding different node constellations are analyzed.

### 8.3.1 Number of tags

As seen in Table 7.8, the RMSE for the position estimates is reduced when more tags are added to the system. For the EKF estimates, the largest change in RMSE is when a third tag is added, and for the FGO estimates it is when a second tag is added. The increased precision for both algorithms is expected, since adding more tags results in more information available to the system. Furthermore, the algorithms receive range measurements at the same rate for all three constellations, and the increase in accuracy must be a consequence of the additional information.

Each set of four ranges between a tag and all four anchors is enough to estimate a relative position, and with three tags, three positions can be estimated. This should increase the accuracy of the system. However, if the frequency of the range measurements is lower, the relative position could change too much between measurements for the additional tags to help. Following this logic, if more followers are added to the system, which reduced the frequency of the range measurements, then there might not be any benefit in having more than a single tag on each follower. Furthermore, with a lower frequency, the algorithms rely more on the motion model, requiring a more accurate model than the one that is used in this thesis.

Also, with three tags available, the relative orientation can be estimated from the UWB measurements. If the orientations are not measured as accurately, the UWB measurements could increase the accuracy of the relative orientation estimates. A common problem with orientation estimates is that the estimate drifts over time. With these additional measurements from a third tag, drifts can be reduced.

### 8.3.2 Tag placements

In Section 7.3.2, the results for the tag placement evaluation are presented. For the scenarios with only one tag, both algorithms perform worse when the tag is placed close to the origin of the leader compared to when it is placed on one of the wing tips. As the tag is placed closer to the origin of the agent, the orientation of the leader affects the measurements less. This could explain the less accurate estimations when the node is placed closer to the origin, since the orientation estimates are very accurate. If the case was that the orientation of the leader is estimated more poorly, the results should be the opposite. In the equation describing the range measurements, (3.3), the position of the tag depends on the orientation of the leader and its local position on the leader. Thus, if the orientation is less accurate, placing the tag closer to the origin increases the accuracy of the estimations.

Two different constellations with two tags are examined. For the EKF estimate, the RMSE is lower when using T1 and T3 compared to when using T1 and T2. As specified in Table 7.9, T1 and T2 are placed on the wing tips, and T3 at the front of the leader. T1 and T2 are placed along the $y$-axis of the leader. Thus, the local position of both tags, as calculated in (3.3), mainly depends on the roll and yaw angles of the leader. T3 is placed along the $x$-axis of the leader, and the position of the tag mainly depends on the pitch and yaw angles. The fact that the tags depend on different angles can be the reason the estimate improves. For the FGO estimate, the RMSE of the relative position is higher when using T1 and T3. This is the opposite result from the EKF. One possible explanation for this is that the distance between T1 and T3 is shorter than between T1 and T2, and that this benefits the FGO more than the EKF.

## 8.4 Additional sensors

From the results presented in Section 7.4, a few conclusions can be drawn. Firstly, the proposed local position case estimation algorithm had some only partially observable states that drifted over time. This is not particularly surprising given that the scenario has many more states to estimate, and few additional equations to compensate with in comparison to the base case. Therefore, additional sensors, such as an airspeed sensor, or a good motion model for the vehicles would need to be added to improve the performance. However, a part of the appeal of the UWB nodes is their cheap price. To require additional expensive sensors in order to get a good estimate of the position may be in direct opposition to many use cases. If adding additional sensors beyond the UWB sensors, barometer and IMU is not possible, then the results clearly indicate that the base case estimation of the relative states is better than forming the relative states from the estimates of the individual poses of the vehicles.

Secondly, the barometer had a positive impact on the estimated position in the base case. However, the base case already has a good estimate of the relative $z$-position in the scenario, so it would be interesting to apply the barometer factor to a case in which the base case FGO implementation struggled with the relative position estimate, such as with the biases in Section 7.2.4.

An interesting note is that there was very little drift in the orientations, even while dead-reckoning the IMU data, as was done in Appendix B, with the resulting estimates presented in Figure B.6 and Figure B.9. This would indicate that there was very little noise in the gyro measurements, which would otherwise cause cumulative errors to build up into drift errors. This would help explain the observation made in Section 8.2.1 about how the estimated orientations from the flight computer (which uses the same data) are excellent over time. It would be interesting to investigate whether noisier gyro measurements could be kept from causing drifts in the estimate by using the estimated relative orientation from the UWB measurements.

# 9 Conclusions and future work

In this thesis, estimation of relative position and orientation by fusing UWB measurements and IMU data has been studied using two different methods: an extended Kalman filter and factor graph optimization. In this chapter, the findings are summarized.

## 9.1 Conclusions

In this section, the conclusions and the results of the goals presented in Section 1.2 are presented.

In this thesis, the performance of the UWB nodes was evaluated. Using a simple method for the antenna calibration, a static error of a few centimeters is achieved, matching the results from other works using similar methods. The measured standard deviation of the range measurements is also within a few centimeters. With the DS-TWR protocol used along with the TDMA implementation, measurements from two tags with four anchors connected are made at 40 Hz. However, the thesis discovered that the measurements are very sensitive to the orientation of the antennas on the UWB nodes, resulting in errors as large as 0.42 meters. Also, about 5% of all measurements are lost.

Based on the performance of the UWB nodes, the range measurements were implemented into the simulation, where one EKF-based and one FGO-based solution were developed. Using orientation estimates from the flight computers fused with the UWB measurements, both the EKF and the FGO estimates the relative position within half a meter during close formation flight, and within one meter during formation flight further away from each other. Overall, the FGO solution results in a better estimation of the relative position, but with worse estimation of the relative velocity. It was not concluded if using two tags over one tag results in a better orientation measurement, since the simulated gyro data used by the flight computer lead to too accurate orientations. With the added antenna calibration error and the missed measurements to the range measurements, both algorithms have a lower accuracy. The FGO solution is affected more when the measurements deviate from the model, and performs worse than the EKF in this case.

The thesis also evaluated the performance of the algorithms with different tag constellations. It was concluded that the system performs better with three tags than with one or two tags. However, the results from using two tags in two different placements are not clear. The EKF performs better with the tags placed along different axes on the leader, while the FGO performs better when the tags are placed on each wing tip, where the distance between the tags is maximized.

The results in regard to the IMU preintegration version of the FGO indicate that more additional sensors than an IMU and a barometer would have to be used in order to get a better estimate. However, the results do indicate that the barometer helped the base case relative $z$ position estimate.

Since the performance of the estimation algorithms declines when the antenna calibration error is introduced in combination with the large errors observed when rotating the UWB antennas, the thesis did not evaluate the algorithms on the hardware. As the error caused by the rotation is significantly larger than the antenna calibration error, the estimates from the algorithms would be useless.

## 9.2 Future work

The results presented in this thesis indicates that it is possible to estimate relative position and orientation by fusing UWB measurements and IMU data for two fixed wings UAVs. Multiple tags on the leader can increase the accuracy of the estimations, and a simple motion model can be enough to achieve good estimations.

To run the estimation algorithm on hardware, a more accurate method to estimate the antenna delay must be used. Furthermore, the choice of antenna on the UWB nodes should be evaluated to decrease the errors caused by rotations of the antenna.

To improve the estimates, more test regarding node placements can be conducted to determine the best possible positions of the UWB nodes. A fifth anchor can be added to the leader to evaluate if this improves the robustness of the system when range measurements are lost.

To further improve the estimates, additional sensors can be added. For example, if accurate height measurements are available, the estimations of the relative positions could be reduced to two dimensions.

# Appendix

# A
# Experiment with all Ultra-wideband devices

To verify that the system works as expected, measurements with one or two tags and all four anchors are performed. In Figure A.1, the measured distance between T1 and all anchors are shown. In the figure, the black lines above each measurement are the actual distance between the tag and the anchor. In Figure A.2 and Figure A.3, T2 is added to the system. As seen in these figures, the system continues to perform as expected, with measurements received from all four anchors for both the tags. Worth noticing is that in Figure A.1, the frequency is twice as high as in Figure A.2 and Figure A.3, since there is only one tag in the system. Throughout these experiments, there is a static error present for all range measurements. To further examine this, all anchors are rotated such that their antenna points at the tag, and additional measurements are performed with one tag. The results are shown in Figure A.4. With new orientations of the anchors, the tag is rotated to point in the direction of the anchors, and the measurements are shown in Figure A.5. As a last test, the tag is rotated to point away from the anchors, which is shown in Figure A.6. The error of the mean measured ranges for each experiment is summarized in Table A.1. It is clear that the measured range is dependent on the orientation of the antennas of the nodes, and that relatively large errors can occur because of this.

***Table A.1:*** *The error of the mean measured distance for different orientations of the UWB nodes.*

| Measurement | Unaligned case | Straight anchors | Straight tag | Backwards tag |
|---|---|---|---|---|
| T1 to A1 [m] | 0.33 | 0.09 | 0.22 | -0.19 |
| T1 to A2 [m] | 0.08 | 0.01 | 0.20 | -0.14 |
| T1 to A3 [m] | 0.42 | 0.07 | 0.27 | 0.00 |
| T1 to A4 [m] | 0.42 | 0.12 | 0.29 | 0.04 |

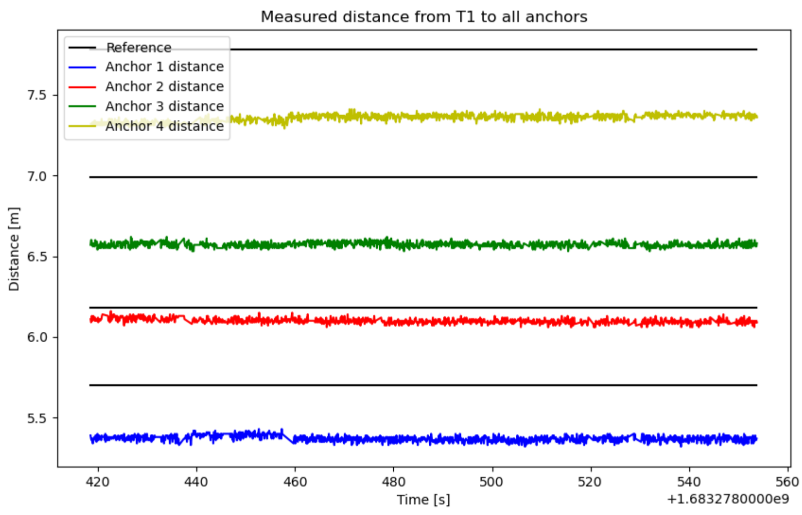


***Figure A.1:*** *Measured distances between T1 and all anchors, with only one tag present in the system.The black lines above each measurement represent the measured ground truths.*

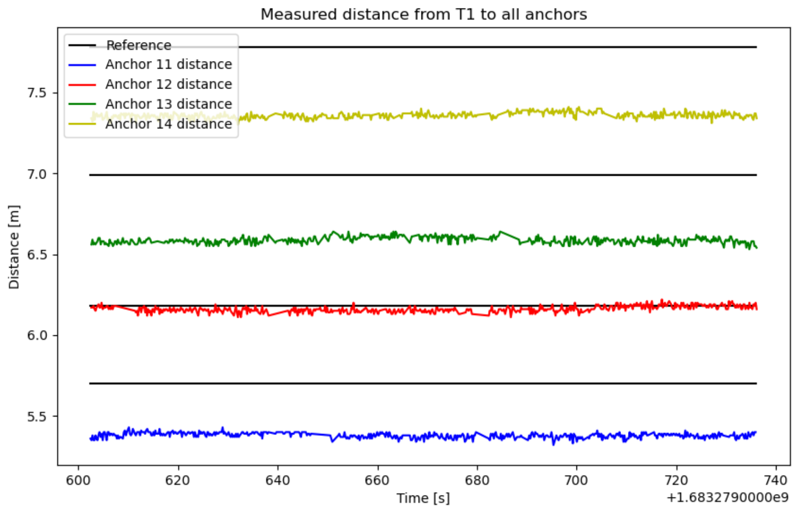


***Figure A.2:*** *Measured distances between T1 and all anchors, with two tags present in the system.The black lines above each measurement represent the measured ground truths.*

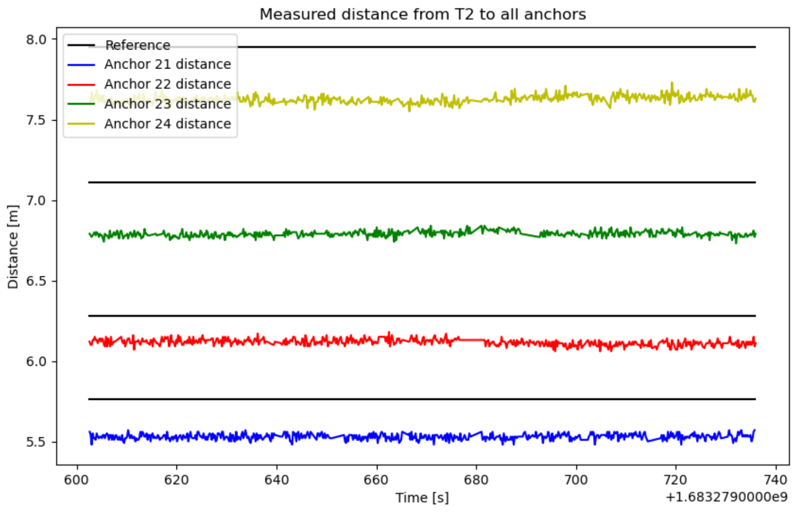


***Figure A.3:*** *Measured distances between T2 and all anchors, with two tags present in the system. The black lines above each measurement represent the measured ground truths.*

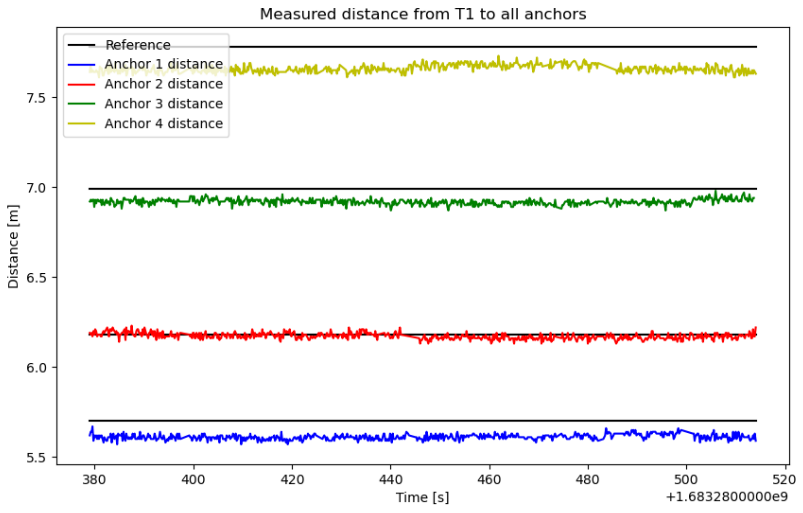


***Figure A.4:*** *Measured distances between T1 and all anchors, with the anchor antennas pointing toward the tag. The black lines above each measurement represent the measured ground truths.*

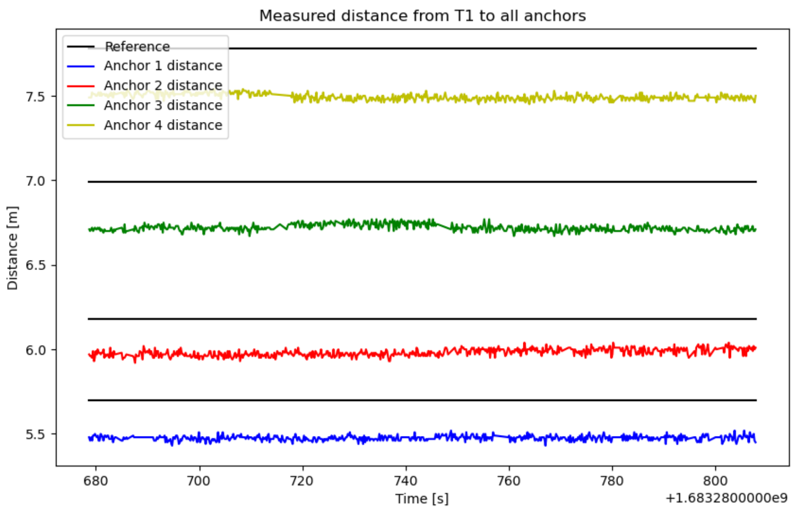


***Figure A.5:*** *Measured distances between T1 and all anchors, with the anchor antennas pointing toward the tag and the antenna of the tag pointing toward the anchors. The black lines above each measurement represent the measured ground truths.*

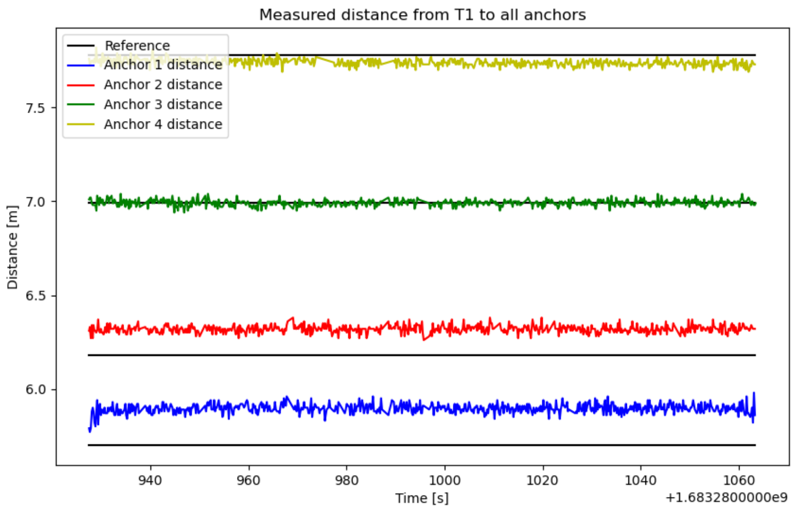


***Figure A.6:*** *Measured distances between T1 and all anchors, with the anchor antennas pointing toward the tag and the antenna of the tag pointing away from the anchors. The black lines close to each measurement represent the measured ground truths.*

# B Estimating UAV poses using only the preintegration factor

In this Appendix the results from estimating the position and orientation of the UAVs using only the preintegration factor from [35] in combination with ISAM2 [30] is presented.

## B.1 Results

In Figure B.1-B.9 the results from running only the IMU preintegration factor from [35] on the IMU data are presented. As the figures show, the position and velocity estimate starts drifting quite quickly, and after a while the ISAM2 optimizer stopped the estimation due to detecting an under determined system. The orientation estimate does not display the same drift tendencies, but there is some error in the estimate.

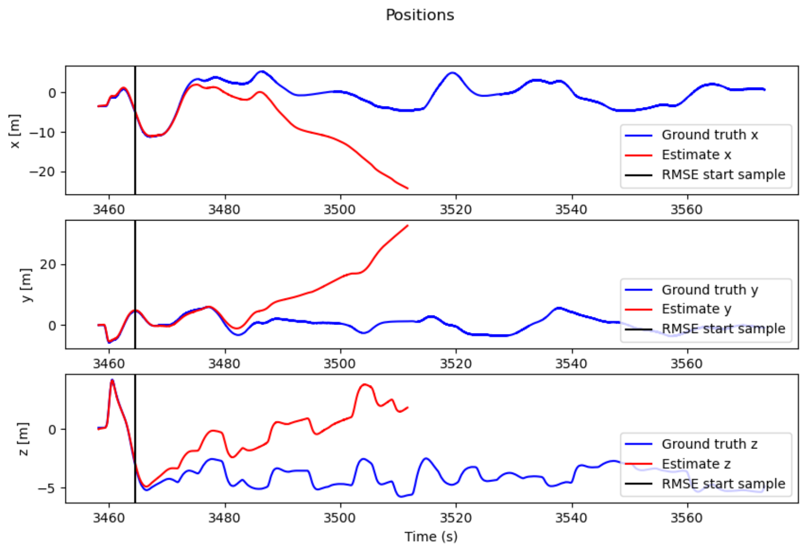


*Figure B.1:* *Relative position when using only the IMU preintegration factor.*

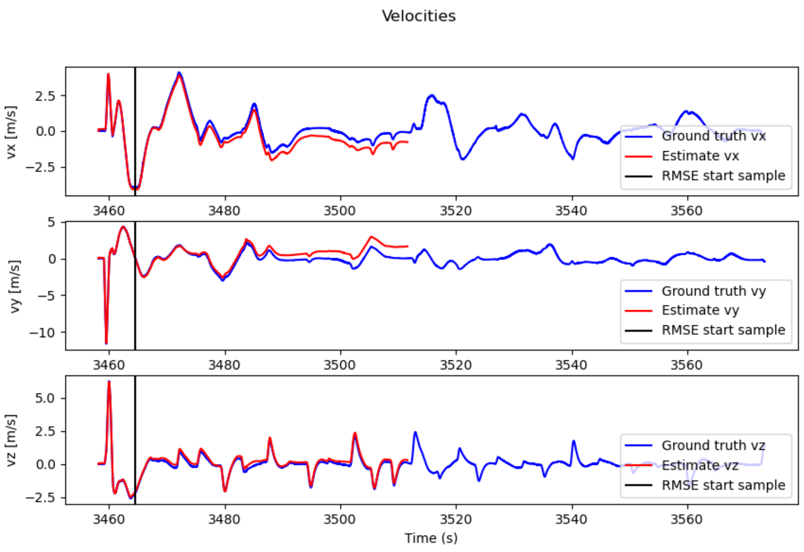


*Figure B.2:* *Relative velocity when using only the IMU preintegration factor.*

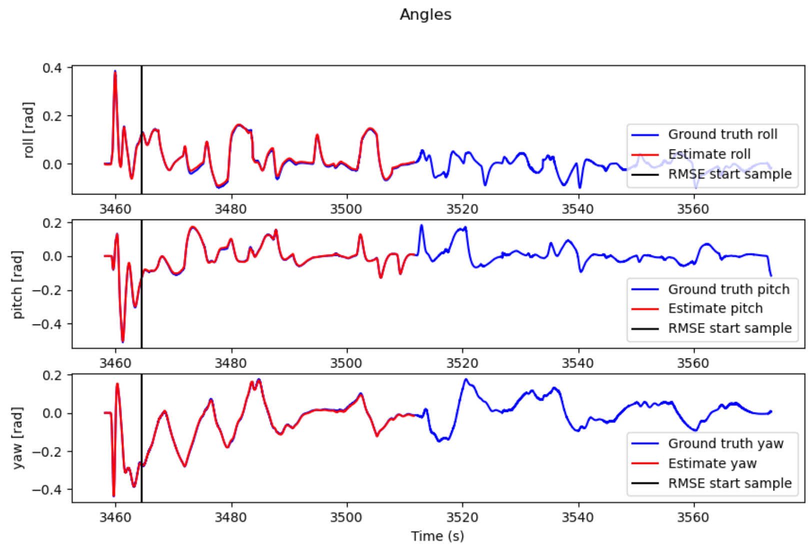


***Figure B.3:*** *Relative orientation when using only the IMU preintegration factor.*

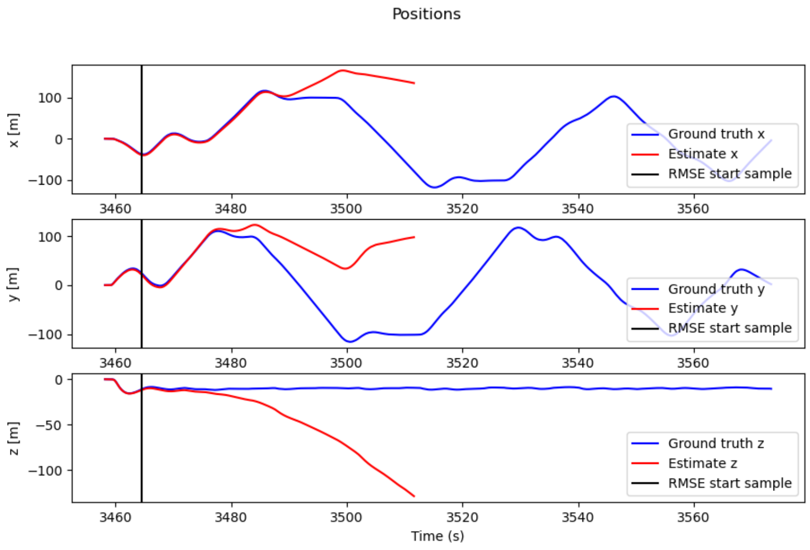


***Figure B.4:*** *Leader position when using only the IMU preintegration factor.*

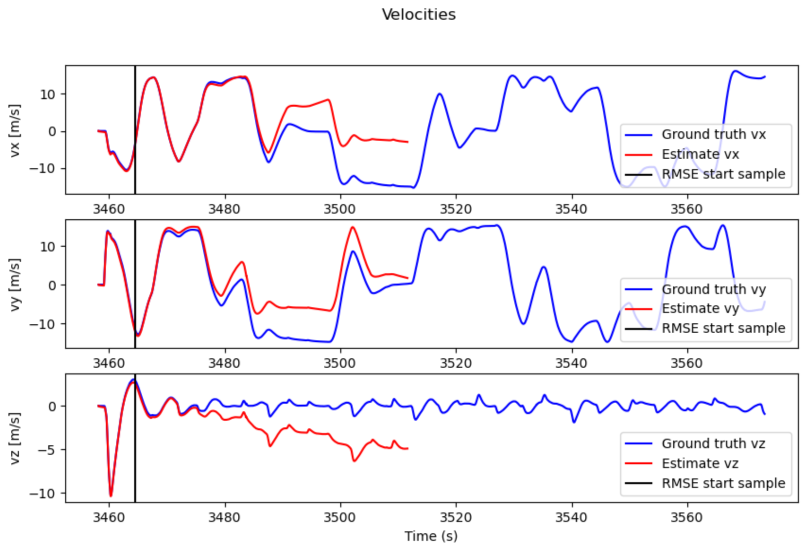


***Figure B.5:*** *Leader velocity when using only the IMU preintegration factor.*

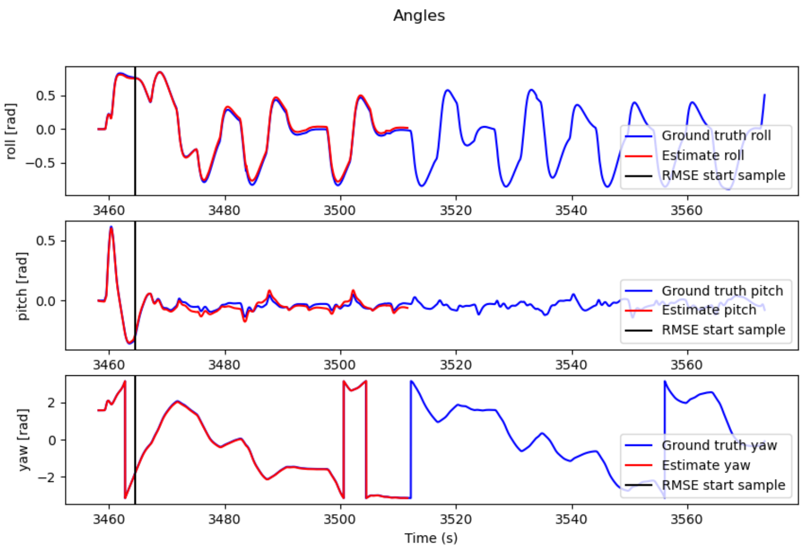


***Figure B.6:*** *Leader orientation when using only the IMU preintegration factor.*

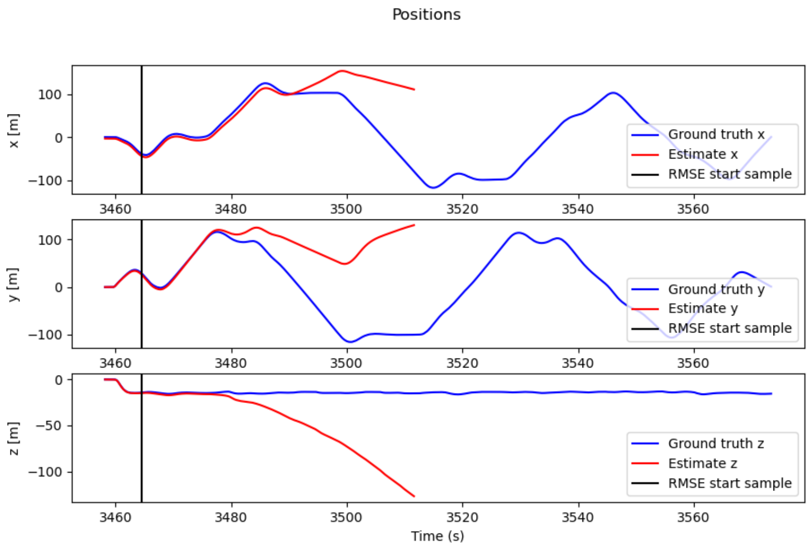


***Figure B.7:*** *Follower position when using only the IMU preintegration factor.*

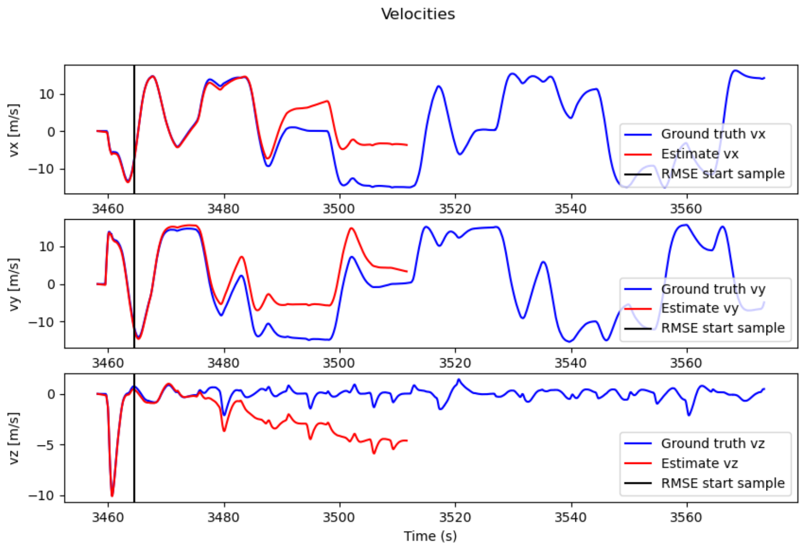


***Figure B.8:*** *Follower velocity when using only the IMU preintegration factor.*

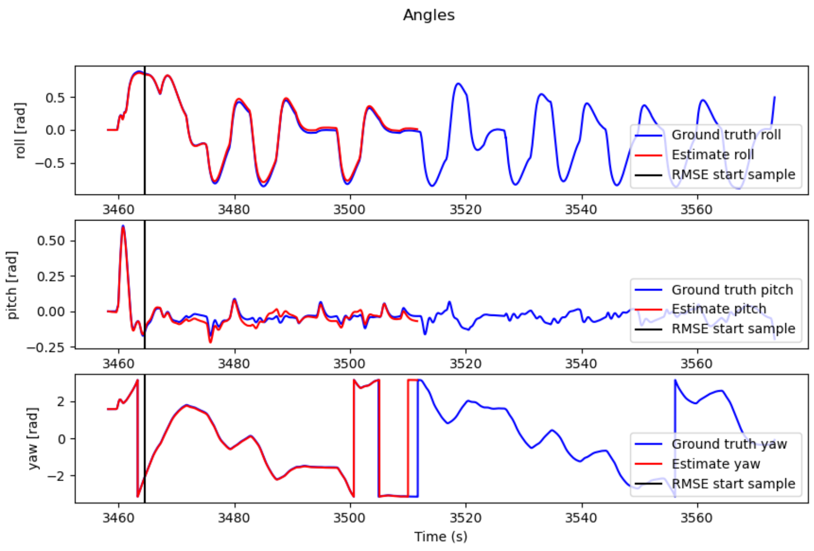


***Figure B.9:*** *Follower orientation when using only the IMU preintegration factor.*

# Bibliography

[1] timesofindia.com. Apple's AirTag is a 'hit' product, analyst claims second-gen may arrive soon. 2022-06-20. *The Times of India*. URL https://timesofindia.indiatimes.com/gadgets-news/apples-airtag-is-a-hit-product-analyst-claims-second-gen-may-arrive-soon/articleshow/92327750.cms.

[2] Matteo Ridolfi, Samuel Van de Velde, Heidi Steendam, and Eli De Poorter. Analysis of the Scalability of UWB Indoor Localization Solutions for High User Densities. *Sensors*, 18(6):1875, Jun 2018. ISSN 1424-8220. doi: 10.3390/s18061875. URL http://dx.doi.org/10.3390/s18061875.

[3] J. González, J.L. Blanco, C. Galindo, A. Ortiz de Galisteo, J.A. Fernández-Madrigal, F.A. Moreno, and J.L. Martínez. Mobile Robot Localization Based on Ultra-Wide-Band Ranging: A Particle Filter Approach. *Robotics and Autonomous Systems*, 57(5):496–507, 2009. ISSN 0921-8890. doi: https://doi.org/10.1016/j.robot.2008.10.022. URL https://www.sciencedirect.com/science/article/pii/S0921889008001747.

[4] Mohammed Shalaby, Charles Champagne Cossette, James Richard Forbes, and Jerome Le Ny. Relative Position Estimation in Multi-Agent Systems Using Attitude-Coupled Range Measurements. *IEEE Robotics and Automation Letters*, 6(3):4955–4961, 2021. doi: 10.1109/LRA.2021.3067253.

[5] Ran Liu, Chau Yuen, Tri-Nhut Do, Dewei Jiao, Xiang Liu, and U-Xuan Tan. Cooperative Relative Positioning of Mobile Users by Fusing IMU Inertial and UWB Ranging Information. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5623–5629, 2017. doi: 10.1109/ICRA.2017.7989660.

[6] H. Zhang, Z. Li, S. Zheng, Y. Liu, P. Zheng, and X. Zou. UWB/INS-based robust anchor-free relative positioning scheme for UGVs. *Measurement Science and Technology*, 33(12), 2022. ISSN 13616501.

[7] Yangyang Liu, Baowang Lian, Chengkai Tang, and Jun Li. Maximum likelihood principle based adaptive extended Kalman filter for tightly coupled

INS/UWB localization system. In *2021 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, pages 1–6, 2021. doi: 10.1109/ICSPCC52875.2021.9565021.

[8] Evangelos Milios Feng Lu. Globally Consistent Range Scan Alignment for Environment Mapping. *Autonomous Robots*, 4, 1997. doi: 10.1023/A:1008854305733.

[9] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J. Leonard. Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age. *IEEE Transactions on Robotics*, 32(6):1309–1332, 2016. doi: 10.1109/TRO.2016.2624754.

[10] H. Zhang, Z. Li, S. Zheng, Y. Liu, P. Zheng, and X. Zou. Leader-follower cooperative localization based on VIO / UWB loose coupling for AGV group. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLVI-3/W1-2022:263–269, 2022. doi: 10.5194/isprs-archives-XLVI-3-W1-2022-263-2022. URL https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XLVI-3-W1-2022/263/2022/.

[11] Shuaikang Zheng, Zhitian Li, Yuanli Yin, Yunfei Liu, Haifeng Zhang, Pengcheng Zheng, and Xudong Zou. Multi-robot Relative Positioning and Orientation System Based on UWB Range and Graph Optimization. *Measurement*, 195:111068, 2022. ISSN 0263-2241. doi: https://doi.org/10.1016/j.measurement.2022.111068. URL https://www.sciencedirect.com/science/article/pii/S0263224122003347.

[12] Make Fly Easy. Make Fly Easy Believer. URL https://en.makeflyeasy.com/index.php/believer/. Accessed: 2023-05-30.

[13] CubePilot. Cube Orange Autopilot. URL https://www.cubepilot.com/#/cube/features. Accessed: 2023-05-30.

[14] Makerfabs. Makerfabs ESP32 UWB Pro. URL https://www.makerfabs.com/esp32-uwb-high-power-120m.html. Accessed: 2023-05-30.

[15] PX4 Autopilot. PX4 Autopilot. URL https://px4.io/. Accessed: 2023-05-30.

[16] 3DXR. Here3+ RTK GPS. URL https://www.3dxr.co.uk/autopilots-c2/the-cube-aka-pixhawk-2-1-c9/here-gnss-and-rtk-c11/cubepilot-bundle-here-3-and-here-base-m8p-p4253. Accessed: 2023-05-30.

[17] Moe Z. Win, Davide Dardari, Andreas F. Molisch, Werner Wiesbeck, and Jinyun Zhang. History and Applications of UWB [scanning the issue]. *Proceedings of the IEEE*, 97(2):198–204, 2009. doi: 10.1109/JPROC.2008.2008762.

[18] Thien-Minh Nguyen, Abdul Hanif Zaini, Chen Wang, Kexin Guo, and Lihua Xie. Robust Target-relative Localization with Ultra-Wideband Ranging and Communication. 2018.

[19] Fredrik Gustafsson. *Statistical Sensor Fusion*. Studentlitteratur AB, 3 edition, 2018.

[20] IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications for Low-Rate Wireless Personal Area Networks (WPANs): Amendment 1: Add Alternate PHYs. *IEEE Std 802.15.4a-2007 (Amendment to IEEE Std 802.15.4-2006)*, pages 1–210, 2007. doi: 10.1109/IEEESTD.2007.4299496.

[21] Decawave. DW1000 User Manual, How To Use, Configure and Program the DW1000 UWB transceiver, 2017. URL `https://www.qorvo.com/products/d/da007967`. Accessed: 2023-05-30.

[22] Dries Neirynck, Eric Luk, and Michael McLaughlin. An alternative double-sided two-way ranging method. In *2016 13th Workshop on Positioning, Navigation and Communications (WPNC)*, pages 1–4, 2016. doi: 10.1109/WPNC.2016.7822844.

[23] S. Shah, K. Chaiwong, L. Kovavisaruch, K. Kaemarungsi, and T. Demeechai. Antenna Delay Calibration of UWB Nodes. *IEEE Access, Access, IEEE*, 9: 63294 – 63305, 2021. ISSN 2169-3536.

[24] Guo Shuli, Qiming Chen, Lina Han, and Xinzhe Gui. A New Calibration Method of UWB Antenna Delay Based on the ADS-TWR. *2018 37th Chinese Control Conference (CCC)*, 2018.

[25] Arturo Gil-Pinto, Philippe Fraisse, and Rene Zapata. Wireless Reception Signal Strength for Relative Positioning in a Vehicle Robot Formation. In *2006 IEEE 3rd Latin American Robotics Symposium*, pages 100–105, 2006. doi: 10.1109/LARS.2006.334330.

[26] Torkel Glad and Lennart Ljung. *Reglerteori : flervariabla och olinjära metoder*. Studentlitteratur, 2003. ISBN 9144030037.

[27] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[28] Roozbeh Dehghannasiri, Mohammad Shahrokh Esfahani, Xiaoning Qian, and Edward R. Dougherty. Optimal bayesian kalman filtering with prior update. *IEEE Transactions on Signal Processing*, 66(8):1982–1996, 2018. doi: 10.1109/TSP.2017.2788419.

[29] Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, 66(8):164–168, 1944. doi: https://doi.org/10.1090/qam/10666.

[30] Michael Kaess, Hordur Johannsson, Richard Roberts, Viorela Ila, John Leonard, and Frank Dellaert. iSAM2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering. In *2011 IEEE International Conference on Robotics and Automation*, pages 3281–3288, 2011. doi: 10.1109/ICRA.2011.5979641.

[31] Joan Solà, Jeremie Deray, and Dinesh Atchuthan. A micro lie theory for state estimation in robotics. *arXiv, cs.O*, 2021.

[32] Decawave. DW1000 Datasheet, 2017. URL `https://www.qorvo.com/products/d/da007946`. Accessed: 2023-05-30.

[33] Decawave. APS014 Application Note - Antenna Delay Calibration of DW1000-based Products and Systems, 2018. URL `https://www.qorvo.com/innovation/ultra-wideband/resources/application-notes`. Accessed: 2023-05-30.

[34] NASA. Glenn Research Center. Earth Atmosphere Model. URL `https://www.grc.nasa.gov/www/k-12/airplane/atmosmet.html`. Accessed: 2023-05-30.

[35] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. On-Manifold Preintegration for Real-Time Visual–Inertial Odometry. *IEEE Transactions on Robotics*, 33(1):1–21, 2017. doi: 10.1109/TRO.2016.2597321.

[36] Frank Dellaert and GTSAM Contributors. BORGlab/GTSAM. URL `https://github.com/borglab/gtsam)`. Accessed: 2023-05-30.

[37] Markus Herb. MHERB/Kalman: Header-only C++11 Kalman filtering library (EKF, UKF) based on Eigen3. URL `https://github.com/mherb/kalman`. Accessed: 2023-05-30.

[38] Kai Ni and Frank Dellaert. HyperSfM. In *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission*, pages 144–151, 2012. doi: 10.1109/3DIMPVT.2012.47.

[39] Vadim Indelman, Stephen Williams, Michael Kaess, and Frank Dellaert. Factor graph based incremental smoothing in inertial navigation systems. In *2012 15th International Conference on Information Fusion*, pages 2154–2161, 2012.

[40] Sean Anderson and Timothy D. Barfoot. Full STEAM ahead: Exactly sparse gaussian process regression for batch continuous-time trajectory estimation on SE(3). In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 157–164, 2015. doi: 10.1109/IROS.2015.7353368.

[41] Ming Hsiao, Eric Westman, and Michael Kaess. Dense Planar-Inertial SLAM with Structural Constraints. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6521–6528, 2018. doi: 10.1109/ICRA.2018.8461094.

[42] Open Source Robotics Foundation. Gazebo Classic. URL `https://classic.gazebosim.org/`. Accessed: 2023-05-30.

[43] Open Robotics. ROS 2 Documentation. URL `https://docs.ros.org/en/humble/index.html`. Accessed: 2023-05-30.