ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institut für Integrierte Systeme
Integrated Systems Laboratory

Department of Information Technology and
Electrical Engineering

Autumn Semester 2020

Towards Autonomous Drones with Ultra-wideband

Master Project

Ilian Jaeger
jaegeri@student.ethz.com

March 2021

Supervisors: Vlad Niculescu, vladn@iis.ee.ethz.ch
Tommaso Polonelli, tommaso.polonelli@pbl.ee.ethz.ch
Dr. Michele Magno, michele.magno@pbl.ee.etzh.ch

Professor: Prof. Dr. Luca Benini, lbenini@iis.ee.ethz.ch

Acknowledgements

I would like to sincerely thank my supervisor Prof. Dr. Luca Benini as well as the Integrated Systems Laboratory at ETH Zürich. They enabled me to realize this thesis by providing me all the means and assistance necessary for achieving my goals.

Furthermore, I would like to thank my advisors Dr. Michele Magno, Tommaso Polonelli and Vlad Niculescu. They supported me with their knowledge and their experience in this field. I appreciate your support, it was great learning from you.

And finally, last but by no means least, I would like to thank the Coredump Hackerspace in Rapperswil for their untiring help in setting up openOCD and everything else I needed for debugging INAV. You guys are great!

Abstract

In this thesis, the localization of wireless sensor nodes using unmanned aerial vehicles (UAVs) is investigated. The vehicle, a custom quadrotor built in the scope of this thesis and specially for its requirements, is acting as a mobile gateway in a wireless sensor network (WSN) consisting of Ultra-wideband (UWB) beacons. Its task is to autonomously localize a remote sensor node with unknown position. UWB technology is used to gather range data in order to localize the beacon by performing a non-linear trilateration algorithm. The localization and navigation happens fully autonomously, without the support of GPS, and without the communication to any external controller.

The evaluation of the accuracy is carried out using real measurement data. UWB range data as well as position estimation data is collected for six different geometrical configurations in order to counteract deviations caused by sensor orientation. With the proposed trilateration algorithm, sub-meter localization accuracy is achieved using the minimum amount of only three range measurements. It is further investigated if the localization accuracy can be increased by calculating the ranging distances from multiple ranging measurements. The results show that increasing the number of measurements does not lead to a significant improvement of the localization accuracy, due to the predominant influence of the drone's erroneous position estimates.

The UWB localization is implemented in a new flight mode in the open-source INAV firmware, which can be openly accessed and used for further development and examination.

Declaration of Originality

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor. For a detailed version of the declaration of originality, please refer to Appendix B

Ilian Jaeger,
Zurich, March 2021

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Unmanned Aerial Vehicles (UAVs), especially drones, grow more and more important for a wide range of applications such as data collection in inaccessible environments, power delivery agents [1], routing, surveillance and monitoring [2], IoT [3], inventory tracking for example in warehouses or cargo ships, military applications [4], transport and logistics [5].

For outdoor applications, UAVs are easily able to locate themselves using the Global Positioning System (GPS), which provides the user with an accurate position estimate. However, in environments where line of sight (LoS) conditions to GPS satellites cannot be granted, for example in forests, urban areas or specially indoors, GPS comes to its limits due to signal loss [4]. Alternative localization systems based on other promising technologies such as ultra-wideband (UWB) radio are thus on the rise [6]. Using UWB beacons, a wireless sensor network (WSN) can be easily deployed in which the nodes are able to localize themselves with respect to neiboring nodes, or possibly so called anchor nodes with known location. Inside this network, the UAV now acts as a mobile wireless sensor node interacting with the remote nodes [7],[8].

In this thesis the problem of localizing a UWB node using a drone is investigated. So to speak, the WSN consists of only two nodes, one being the UAV itself acting as a mobile wireless sensor, and the other being a remote, fixed UWB beacon. The drone then has to localize the remote anchor fully autonomously, meaning without the help of GPS data, and without the regulation of any external controller. The localization as well as the navigation towards the UWB node must be calculated on-board, and only with the available data from its own sensors, as well as with the UWB ranging data. In order to achieve this, the drone gathers three UWB range measurements from different locations. Therewith it performs a trilateration algorithm in order to find the location reference of the target.

Chapter 2

Related Work

Localization in WSNs is a hot topic in recent research [2]. There are many works exploring UAVs as mobile gateways in WSNs, commissioned to localize remote beacons with unknown position [9]. Different approaches try to address this problem, among which are many based on range measurements [10]. The ranging itself can be done using different technologies out of which UWB is a novel and promising choice, thus being in the scope of numerous works using this technology for range-based multi-lateral measurement localization (RB MML) algorithms [11],[12].

This work is strongly based on [12], using exactly the localization algorithm which is proposed therein: A RB MML approach with UWB, trilateration, extended by a cost function in order to handle white noise by penalizing measurement errors. The difference to this work is, however, that the whole algorithm runs on a new platform, and that it is not supported by GPS. This introduces the challenge of self-localization using only on-board sensor data on the one hand, and localization of remote sensors with respect to the own location on the other hand.

There are many works evaluating the localization accuracy of RB MML algorithms based on UWB using synthetic data [4]. However, in this work the evaluation of the trilateration algorithm used for localizing wireless sensor nodes is done using real measurement data. This provides accurate results without discrepancies due to model errors, especially in the GPS free setting afflicted with complex position errors.

Chapter 3

Theory / Algorithms

In this chapter the reader will get a quick overview of the technologies as well as algorithms used in this work. First, the UWB radio technology is introduced. Second, the algorithms which are used in order to localize an UWB node are briefly described. The first algorithm is *Double-sided Two Way Ranging* which is used by the drone's UWB transceiver in order to measure distances to other UWB nodes. A quick overview over the ranging algorithm using UWB will be given based on [13]. The second algorithm is a non-linear *Trilateration* algorithm, implemented on the flight controller enabling it to localize UWB nodes using the range measurements.

3.1. Ultra-Wideband Radio Technology

Ultra-wideband is a radio technology based on high-bandwidth communication, using frequency bands that are over 500 MHz wide. The highly populated frequency range is equivalent to extremely short signal pulses and thus a high temporal resolution in the time domain [14]. This characteristic offers the possibility for multipath resolution which makes UWB technology highly convenient for short-range communication in cluttered multipath environments [15]. Further, due to the different frequency components within a UWB signal, UWB communication ensures a high reliability also in non-line of sight (NLOS) environments. A further exploit of the UWB signals's high temporal resolution are the accurate estimates of the signal's time of flight (ToF). Those estimates of the signal's propagation time serve to measure distances between two UWB transceivers [6]. This way of measuring distances is known as *ranging*. Due to its short signals in the sub-nanosecond range, UWB technology is highly suitable for ranging [16]. In the following section a specific, ToF-based ranging method is introduced.

3.2. Two Way Ranging

Double-sided Two Way Ranging (DS TWR) is a ranging algorithm used to determine the distance between two communicating nodes. The idea is to measure the propagation time a signal uses to reach the target node, denoted as time-of-flight or T_{tof} and from it calculate the distance. As the propagation speed of the UWB signal is the speed of light, the distance d can easily be calculated from the measured time of flight using the following formula

$$d = T_{tof} \cdot c$$

with c denoting the speed of light. The approach of TWR in order to measure the UWB signal's time-of-flight is depicted in figure 3.1. In order to understand the process of two way ranging, let us first consider *Single-sided Two Way Ranging* (SS TWR), and then go to DS TWR which is an extension of the single-sided variant in order to increase the accuracy.

3.2.1. Single-sided Two Way Ranging

Single-sided Two Way Ranging (SS TWR) is used to calculate the signal's time of flight from one single *round trip time*. A round trip time is the time that passes for one device between transmission of a first message and reception of an according response. In order to measure the first round trip time T_{round1}, the following procedure is executed:

- **Polling** The initiating UWB node (device 1 in figure 3.1) starts the ranging process by sending a *poll message* to the remote UWB node (device 2). The time it takes the signal to reach the responding node is denoted as T_{tof}.
- **Response** Device 2 replies by sending a *response message*. The time between reception of the poll message and transmission of the response message is called the reply time denoted as T_{reply2}. This time can be set by the user and is sent along with the response message back to the initalizing node.

With the received response message the initiating device is now able to calculate the first round trip time T_{round1}. Knowing this time, the signal's time-of-flight can be estimated using the following formula

$$\hat{T}_{tof} = \frac{1}{2}(T_{round1} - T_{reply2}) \tag{3.1}$$

3.2.2. Increasing the Estimation Accuracy: Double-sided Two Way Ranging

In order to increase the accuracy of the time of flight estimate, SS TWR can be extended such that a second round trip time is acquired, which is then included into the calcualtion

of T_{tof}. Therefore, device 1 sends a *final message* after the reception of the response message. Contained in the final message is the device's corresponding reply time T_{reply1}, as well as the first round trip time. Now, the responding node can as well calculate it's round trip time T_{round2} and combine it with the first one in order to estimate T_{tof} with the following expression:

$$\hat{T}_{tof} = \frac{T_{round1} \cdot T_{round2} - T_{reply1} \cdot T_{reply2}}{T_{round1} + T_{round2} + T_{reply1} + T_{reply2}} \tag{3.2}$$

Finally device 2 reports the calculated distance to the initiating device 1 by sending it the *report message*.

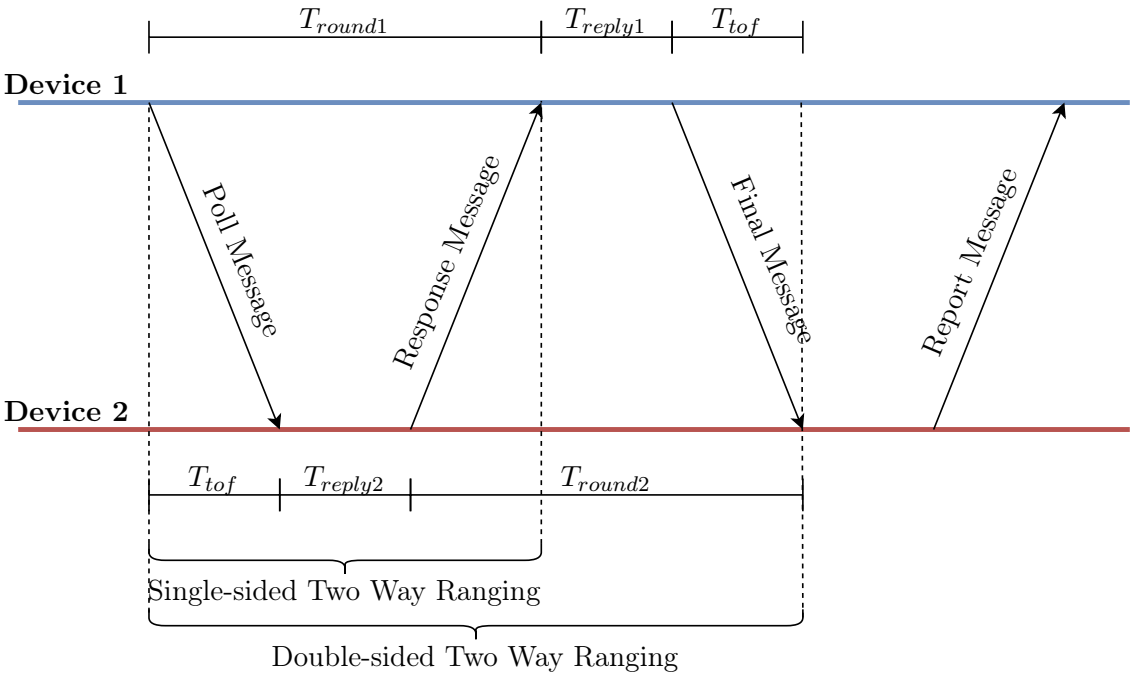


Figure 3.1.: Flow Chart of Double-sided Two Way Ranging.

3.3. Multilateration

Multilateration is a localization algorithm which can be used to estimate the position $\mathbf{x}_n$ of a UWB node when multiple (at least three) anchor positions $\mathbf{x}_i = (x_i, y_i, z_i)$ along with the respecting distances r_i between those anchors and the UWB node are known. In the case of this thesis, the anchor positions correspond to waypoints visited by the UAV that is trying to localize the UWB node. At each one of those waypoints the UAV acquires a range measurement of the node. Let us represent the i-th range measurement as the tuple $(\mathbf{x}_i, r_i)$, with $\mathbf{x}_i$ denoting the anchor position and r_i being the range distance between

the UWB node and the UAV at this position. With the acquired range measurements and the corresponding anchor coordinates, the following cost function can be formed:

$$L(\mathbf{x}_n) = \sum_{i=0}^{N} \left(\|\mathbf{x}_n - \mathbf{x}_i\|_2 - r_i\right)^2 \tag{3.3}$$

Finding the global minimum of this cost function corresponds to minimizing the measurement error by solving the following non-linear, non-convex optimization problem:

$$\mathbf{x}_n^* = \text{argmin}_{\mathbf{x}_n} L(\mathbf{x}_n) \tag{3.4}$$

In [17] it is shown how this optimization problem can be reformulated and solved as a linear least-squares problem.

3.3.1. Non-linear Trilateration

In this thesis, a specific form of multilateration, *Trilateration*, is implemented. As the name indicates, trilateration is a form of multilateration using only three anchor positions with their respective distances [18]. Hence, the input variables of the algorithm are the tuples $(\mathbf{x}_i, r_i)$ for $i = 1, 2, 3$. In order to further simplify the calculations, the trilateration problem is projected to a two-dimensional plane, the ground. This can be done with the following mathematical mappings.

Projecting the drone's position at the i-th waypoint to the ground plane can easily be done by omitting the z coordinate:

$$\begin{aligned} f :&\mathbb{R}^3 \longrightarrow \mathbb{R}^2 \\ &\mathbf{x}_i \longmapsto \mathbf{x}_i' = (x_i, y_i) \end{aligned} \tag{3.5}$$

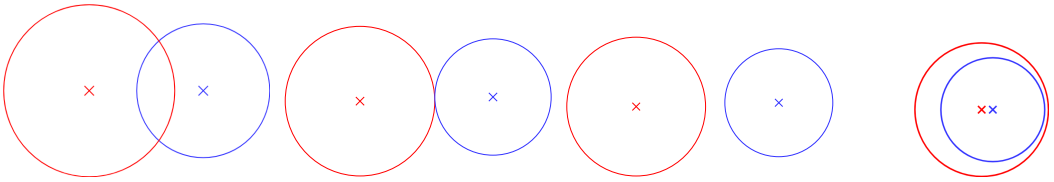
In order to project the measured ranging distance to the ground, the mapping

$$\begin{aligned} &g : \mathbb{R}^2 \longrightarrow \mathbb{R} \\ &(r_i, z_i) \longmapsto g(r_i, z_i) = d_i = \sqrt{r_i^2 - z_i^2} \end{aligned} \tag{3.6}$$

with z_i being the drone's altitude above the ground plane. The resulting d_i denotes the distance between the i-th waypoint and the UWB node, projected to the ground.

Combining these two mappings, the resulting function used to project the problem to two dimensions looks as follows:

$$\begin{aligned} h : \mathbb{R}^4 &\longrightarrow \mathbb{R}^3 \\ (\mathbf{x}_i, r_i) = (x_i, y_i, z_i, r_i) &\longmapsto (f(\mathbf{x}_i), g(r_i, z_i)) = (\mathbf{x}_i', d_i) \end{aligned} \tag{3.7}$$



(a) Two intersections **(b)** One intersection **(c)** No intersection, non-overlapping **(d)** No intersection, overlapping

Figure 3.2.: Possible node positions if two waypoints are considered.

Restating expression 3.3 of the cost function for the two-dimensional case results in the follwing cost function:

$$L(\mathbf{x}'_n) = \sum_{i=0}^{N} \left(\|\mathbf{x}'_n - \mathbf{x}'_i\|_2 - d_i \right)^2 \tag{3.8}$$

Now, in order to estimate the UWB node's position (x_n^*, y_n^*) one could do a grid search over the whole 2D plane and find the global minimum of 3.8. However, in order to speed up the calculations the grid must first be constrained to a smaller area. For this, a candidate point $(\tilde{x}, \tilde{y})$ is to be found around which it is reasonable to do a smaller grid search. In order to find this candidate point, only the first two waypoints are considered. Knowing the distance of the UWB node to a specific waypoint, the possible location of the node could be anywhere on the circumference of the circle around the waypoint with radius length corresponding to the distance. If now two waypoints are considered, the two circles formed with the respecting distances can either intersect in two points, overlap in one point or, due to measurement errors, not have any intersection point at all. All of the possible cases are depicted in figure 3.2

Let us now assume the easiest case where the two circles intersect in two points as shown in figure 3.2a. The cost function 3.8 is now evaluated at both of these points, and the one which yields a lower cost is chosen as candidate point. Around this point the global minimum of 3.8 is now searched within a 9.5×9.5 meter area. The grid's verteces are placed 10 cm apart from each other. Finally the vertex yielding the lowest cost is chosen as position estimate (x_n^*, y_n^*) of the UWB node position (x_n, y_n). For the cases illustrated in subfigures 3.2b to 3.2d the procedure is the same, only the way of finding the candidate point is slightly different. The implementation of these three cases can be seen in the implementation [1].

[1] `https://gitlab.ethz.ch/jaegeri/parhamversion`

Chapter 4

Hardware

This chapter gives an overview over all the hardware components that the UAV is built of. A list of all the components as well as a wiring diagram is shown in the following sections.

4.1. Hardware Components and Wiring Diagram

The following parts were used to build the drone:

- MATEKSYS Flight Controller F722-SE [19]
- MATEKSYS Optical Flow and Lidar Sensor 3901-L0X [20]
- Hobbywing X-Rotor 4-in-1 ESC [21]
- FrSky XM Radio Receiver [22]
- EMAX RS3206 2400kV Motors [23]
- Master Airscrew Electric Only - 6x3 Propellers [24]
- Mainframe: Outlaw 270 [25]

Diagram 4.1 shows how these components are wired together to make up the final drone.

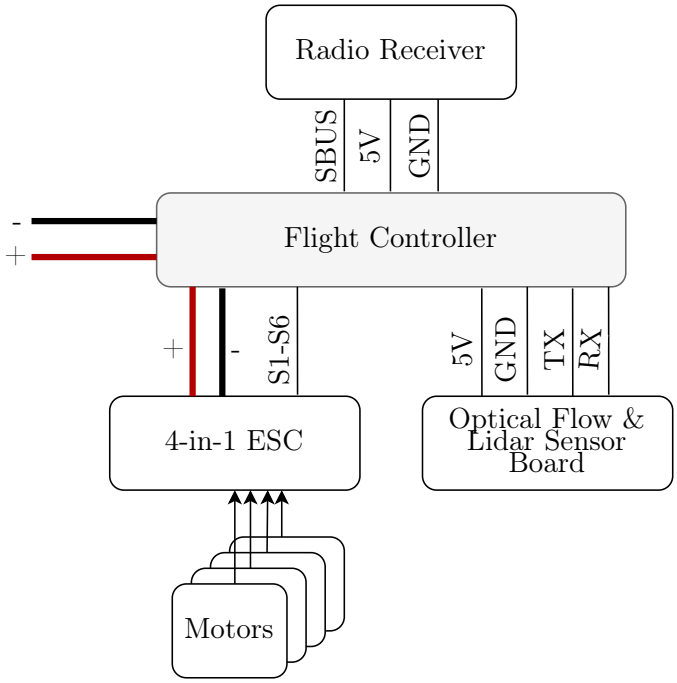


Figure 4.1.: Drone wiring diagram.

4.2. Sensors

The sensors used by the drone in it's final state to estimate its orientation and position are listed below. The drone's flight controller is equipped with:

- **Gyroscope and accelerometer**
  The flight controller's inertial sensors comprise a *dual gyroscope* as well as an *accelerometer*. The gyroscope is used for measuring the sensor's angular velocity, and the accelerometer measures it's external specific force. Together, these two sensors make up the so called *inertial measurement uint* (IMU), and their measurements are combined in order to get a pose estimation, which is the estimation of both the drone's position and orientation. Those estimates are obtained by a process called *dead-reckoning* [26].
- **Barometer**
  With the barometer the UAV's altitude can be estimated.
- **Current & battery voltage sensor**
  For the sake of completion, these two sensors are mentioned here as well. However, they are used for monitoring the battery and not for orientation pruposes.

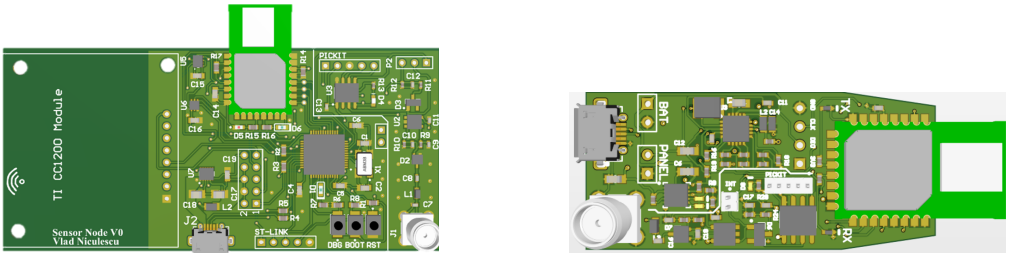
Pose estimations based on IMU measurement data are strongly influenced by sensor noise. Especially over longer time periods they are corrupted by integration drift [26].

For this reason, oftentimes IMUs are combined with other sensors such as cameras [27]. Likewise in this thesis, the UAV is equipped with a further sensor board enclosing the following sensors:

- **Optical flow sensor**
  The optical flow sensor's purpose is to improve the estimation of the UAV's movement in the x-y-plane parallel to the ground.
- **Lidar**
  The lidar sensor is used as a so called *rangefinder*, acquiring precise altitude measurements in the range of 8 to 200 cm [20].

4.3. UWB Transceiver

The UWB Transceiver that is used in this work is the Decawave module DWM1000 [28]. It is integrated on a custom PCB which was designed in the scope of the Master Thesis of Vlad Niculescu. The first version of the board is the one attached to the drone. It has a USART connection over which it communicates to the flight controller. The the considerably smaller second version of the board serves as UWB node. Both boards can be seen in figure 4.2.



(a) UWB Board version 1. **(b)** UWB Board version 2

Figure 4.2.: PCBs with integrated DWM1000 UWB module

Chapter 5

Design Implementation

5.1. Drone Firmware: INAV

The firmware that runs on the flight controller is the opensource flight controller firmware *INAV* [29]. It is a wide-spread software used not only to navigate drones, but all sorts of RC vehicles such as fixed-wing airplanes or helicopters, and even cars and boats. It is the firmware recommended by MATEKSYS to use on the F722-SE flight controller. It provides many functionalities especially for autonomous navigation.

The version of INAV used in this project is 2.6.0 RC3. The functionalities that are used are the communication with the radio controller, as well as integrating all the sensor data mentioned in chapter 4.

5.1.1. Firmware Modification: Autonomous UWB Node Localization

This work's main contributions to INAV consist in the extension of two new features:

- Communication with the UWB transceiver
- Implementation of the nonlinear trilateration algorithm for UWB node localization

In order for the drone to perform the UWB node localization, a new separate flight mode has been added to the already existing flight modes of INAV. Entering this new *localization* flight mode, the drone autonomously starts localizing the UWB node by performing the trilateration algorithm, and then it finally flies to the node's estimated position. This flight mode has to be enabled along with the already existing modes *position hold* and *altitude hold* (and / or the *surface* mode enabling terrain following) such that the drone holds it's 3D position and does not drift. Once the localization mode is enabled, the INAV scheduler creates a new periodic task which is repeated with the

task rate of 50 Hz. In this task, the program flow is implemented as a state machine, depicted in figure 5.1.

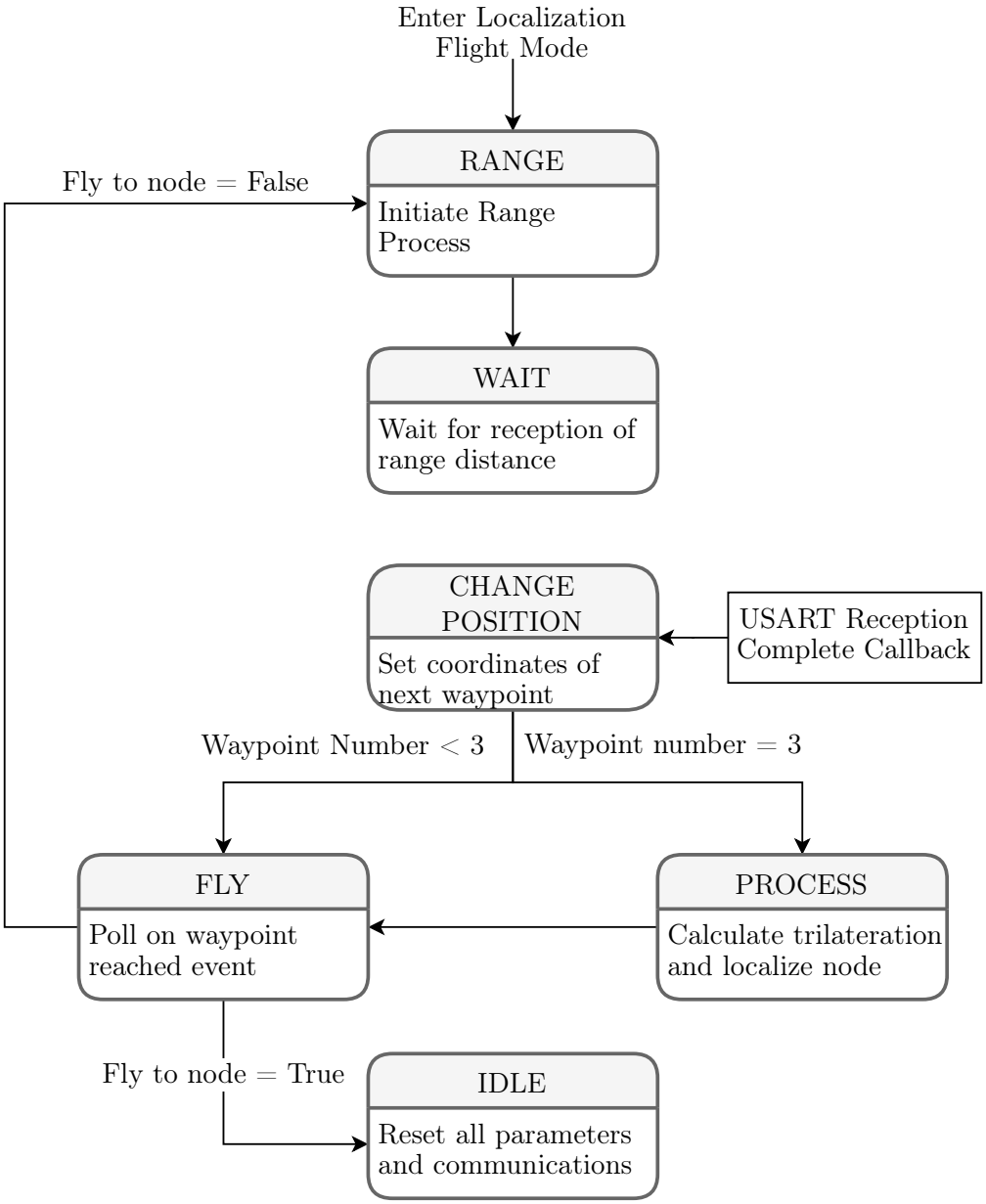


Figure 5.1.: State machine of the new Localization flight mode in INAV.

5.2. UWB Board Firmware: Parham Version

The firmware running on the UWB boards also had to be modified for the purposes of this thesis. There are two different state machines implemented, one for the board attached to the drone, in the following called initiator, and one for the UWB nodes used for ranging, in the following called responder.

5.2.1. Initiator

The UWB board connected to the drone, the initiator board, runs the state machine depicted in figure 5.2. The firmware has to be manually adjusted such that the board starts up in initiator mode. Then, the board waits for two types of messages over USART: The initiation request for a ranging process or a log message. If the first kind of message is received, the board initiates a ranging process, after which it reports the calculated distance over USART to the flight controller. If a log message is received, it is just forwarded over UWB to the logger node, which prints the message over the serial port connected to the computer.

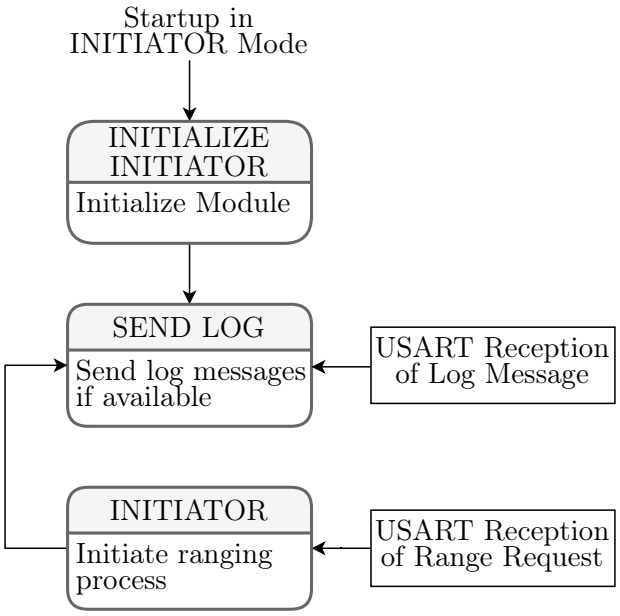


Figure 5.2.: State machine of the initiator node.

5.2.2. Responder and Logger

To start the board in responder mode, this also has to be manually set in firmware by initializing `tag_FSM_state_t state = INITIALIZE_RESPONDER`. When the board starts

up this way, it acts as responder, so as that UWB node that does ranging and that has to be localized. After the initialization it waits for poll messages of the initiator. Once a poll message is received, the ranging process is executed, and the responder enters the process state. There it acknowledges a successful procedure if more ranging measurements are expected. If all the measurements have been completed, instead of an acknowledgement it sends a report message back to the initiator, containing the calculated (mean) distance.

If the node is supposed to act as logger, it also has to start up as responder. However, then the debug button must be pressed in order to change to logger mode. The node is assigned new callback functions such that it only reacts to log messages. If such a message is received, the node does nothing but print it out to the serial port.

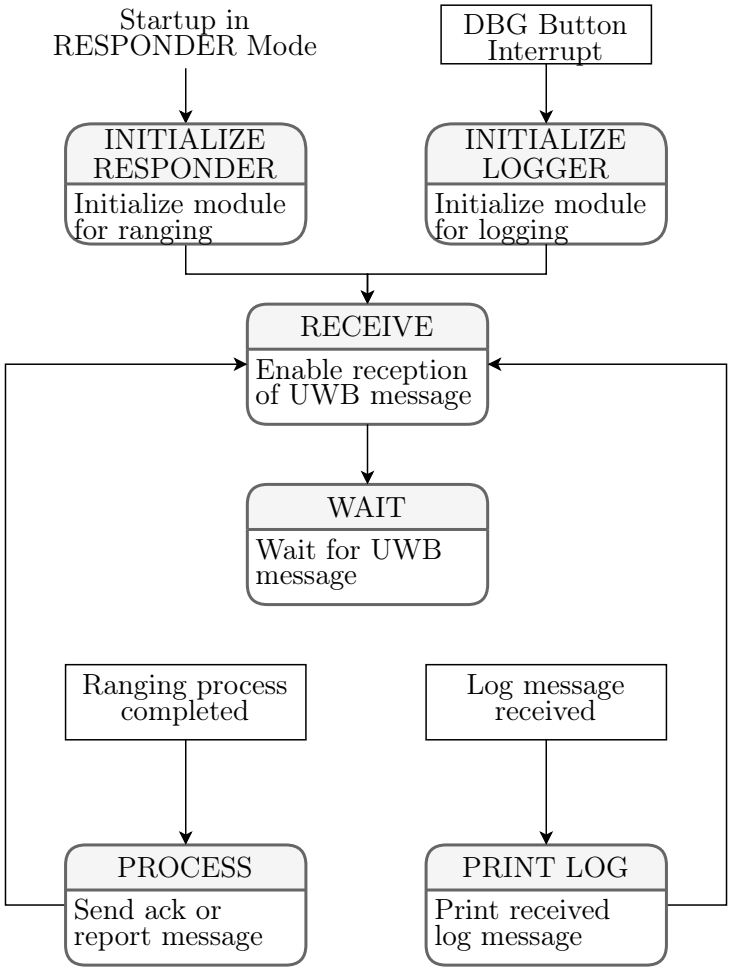


Figure 5.3.: State machine of the new responder and logger node.

Chapter 6

Experiments

In order to validate the implemented localization algorithm as well as to evaluate its accuracy, several experients have been carried out. In this section it is described how the experiments have been set up, what data have been gathered and how the localization accuracy has beed evaluated. In section 7, the according results are then presented.

6.1. Experimental Setup

In the experiments, the drone performed the trilateration algorithm as explained in chapter 3 in order to estimate the position of one single UWB node. The first experiment was set up according to figure 6.1.

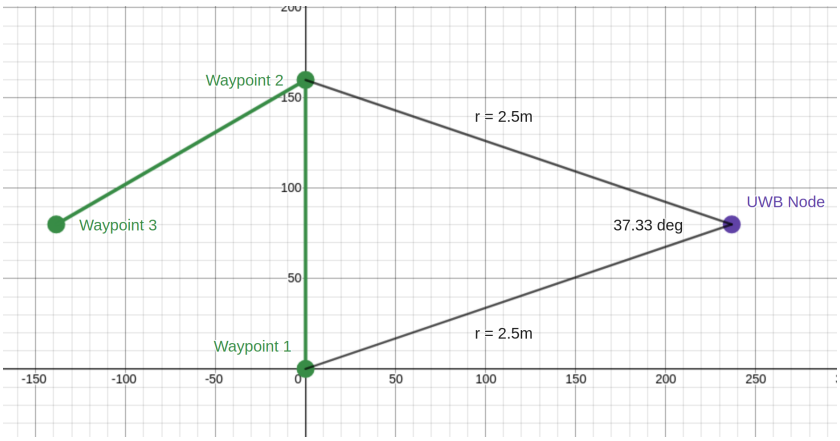


Figure 6.1.: Experimental Setup of first Scenario.

The drone flew an equilateral triangle with 1.6 meters side length, marked by the green line in the image. The node that had to be localized, marked with a purple dot, was placed 2.5 meters away from the first two waypoints. Accordingly, the angle formed by the trajectory waypoint 1 - UWB node - waypoint 2, in the following called *alpha*, amounts to 37.33 degrees. Previous works have shown that the localization error depends on this alpha angle - the biggest angle between two waypoints and the node. For a value of alpha bigger than 20 degrees, it was shown that the localization error is stable with respect to the error related to the range measurements [12].

In [30] it is shown that the localization error strongly correlates with the relative pose of the two ranging UWB modules. This means that the range error depends on the direction from which the ranging messages come from [4]. In order to better generalize and to grant results which are independent of the relative pose of the modules, the above described experiment has been repeated five additional times. Each time the UWB node's position has been moved along on a circle around the middle point of the triangle. The composition of the resulting six scenarios is depicted in figure 6.2.

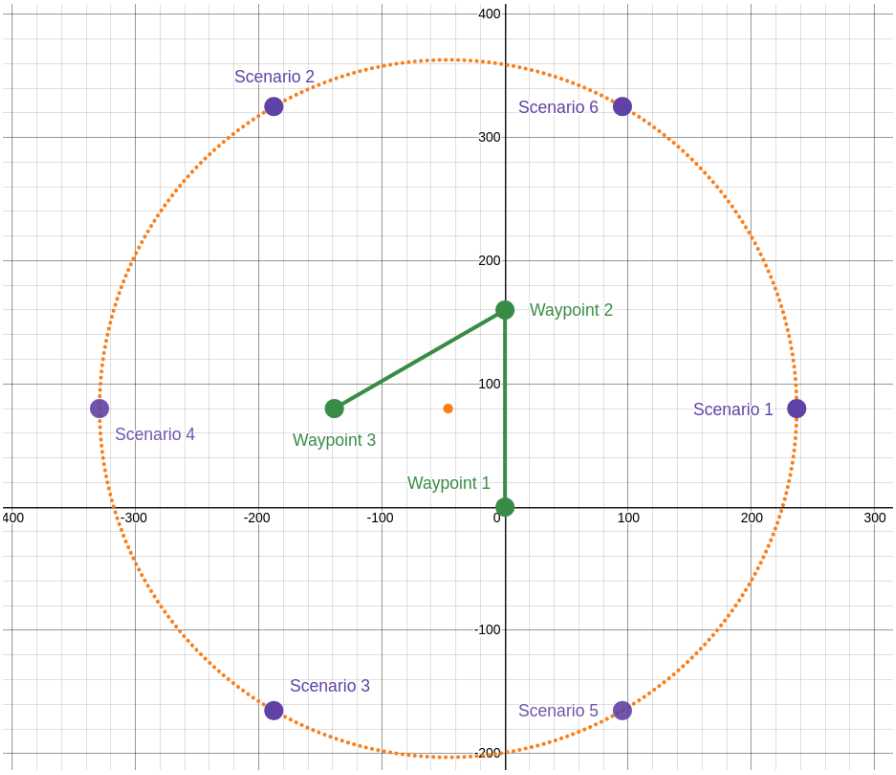


Figure 6.2.: Experimental Setup of all the Six Scenarios.

6.2. Simulating the Trilateration Algorithm

For the sake of statistical significance of the experimental results, each experiment has to be carried out multiple times. As this is an extremely time consuming procedure, the trilateration algorithm was re-implemented in python on a computer and then simulated repeatedly. Figure 6.3 shows schematically what input variables are needed in order to simulate trilateration and obtain the node's position estimate. The first kind of input variables are the measured ranging distances d_1 to d_3 for each waypoint, all projected to the ground as seen in chapter 3.3.1. The second set of input variables are the waypoint's coordinates (x_i, y_i) again for each waypoint $i = 1, 2, 3$, and again projected to the ground. In conventional *Monte Carlo Simulations* (MCS), the input random variables are modelled, and then the input data acquired by sampling from these modelled input distributions [31]. As opposed to this approach, in the case of this thesis real measurements were used as input data. The reason for this is that it is not trivial to model the range error, as it depends on different factors such as the nodes' distance, their relative orientation and other [30]. Similarly, the drone's position estimation error is not easy to model as own investigations have shown. Thus, real measured input data was used for the simulation in order to decrease the model error.

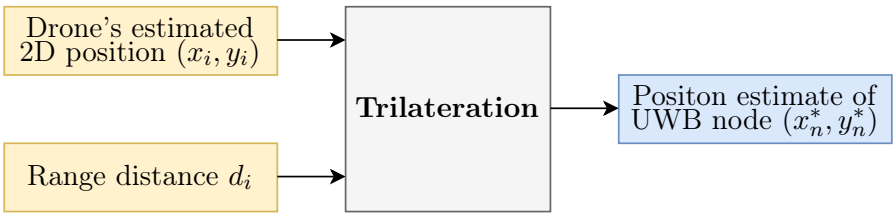


Figure 6.3.: Input and output variables of the trilateration algorithm.

The experiments were set up such that all the necessary data could be gathered in one single run per scenario. This means that for each scenario, the drone flew the triangle one single time, and during this run following data were acquired:

- **Ground truth position data**
  The *VICON* system is a camera system that is able to measure the position of specially marked objects using infrared radiation. It achieves an accuracy in the range of micrometers. Thus, the position measurements aquired with VICON were treated as ground truth position of the drone as well as the UWB node.
- **Estimated position data**
  Further, the drone's estimates of it's own position were logged over UWB with a rate of 50 Hz. One measurement of the drone's position consists of the three spatial coordinates x, y and z, with (x, y) indicating the drone's position on the 2D plane parallel to the ground, and z being the altitude. As the drone was only equipped with one single UWB transceiver, no position data were logged during the rangin processes.

- **Measured range data**
  At each of the three waypoints, the drone acquired 500 range measurements r. The resulting ranging distances were captured by the UWB transceiver that served as the node which was to be localized.

Using these gathered data, the trilateration could finally be simulated 500 times for each scenario. The resulting position estimate was then compared to the UWB node's ground truth position, and the estimation error was calculated.

6.3. Trilateration and Measurement Errors

Further simulations were made in order to better understand where the localization error comes from. Both of the simulation's input variables, the distances and the drone's positon estimates are corrupted by noise. Both of these types of error contribute to the overall localization error. Figure 6.4 depicts this circumstance.

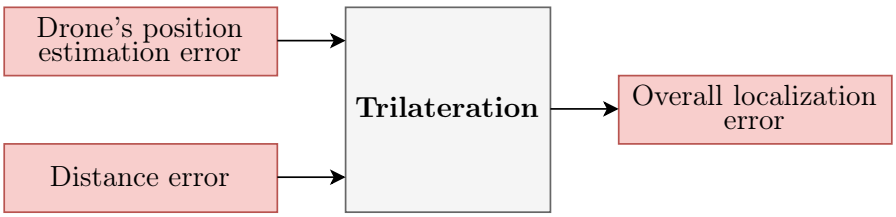


Figure 6.4.: Errors of input and output variables.

As mentioned in section 3.3.1 the trilateration problem is projected to the ground. There are two variables per waypoint which are used as input to the simulation: the 2D position of the waypoint as well as the distance from the waypoint to the node. Those two variables are combined in the tuple $(\mathbf{x}'_i, d_i)$. The first kind of error, the drone's *position estimation error*, arises out of the drone's defected measurement of its x and y coordinates. The second kind of error, the *distance error*, marked red in figure 6.5, is composed by two different measurement errors: the range measurement error as well as the drone's altitude estimation error, or simply its z coordinate, which are also depicted in the same figure.

In order to examine the influence of both of these input errors individually, two further simulations have been carried out. In the first one, ground truth range data as well as ground truth drone altitude were used for the calcualtion of the distances. Hence, the calculations were rid of the distance error. Equivalently, in the second simulation, the drone's ground truth 2D position was used and so the drone's position estimation error was eliminated. For both simulations, the prediction errors were calculated in the same manner as in section 6.1 and then compared.

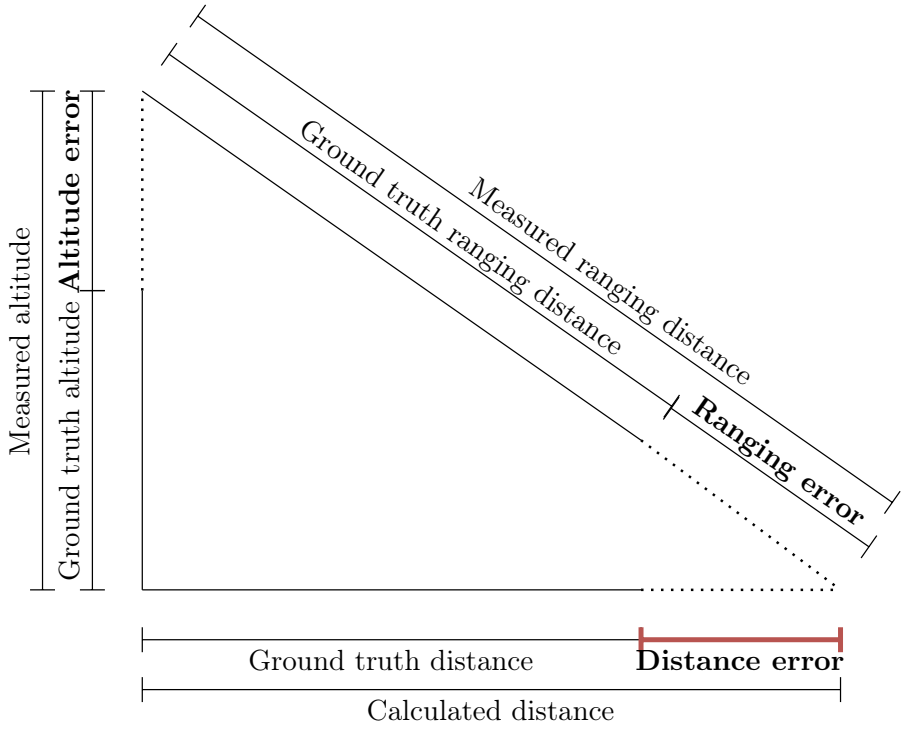


Figure 6.5.: Distance error.

6.4. The Optimal Number of Range Measurements

A further aspect of interest was the question whether, and by how much, the localizaion accuracy can be improved by increasing the number of acquired range measurements per experiment. Leaving the number of waypoints at three and thereby sticking to the trilateration algorithm, the number of acquired range measurements **per waypoint** was increased. Then, the mean over these measurements was calculated and used for further calculations. By doing this it was aimed at reducing the ranging error and thus also the localization error. Further simulations have been done with different numbers of measurements per waypoint in order to evaluate this assumption. The results are again shown in the following chapter in section 7.3.

Chapter 7

Results

The following chapter is structured in the same way as chapter 6. In the first section the general results on the localization accuracy are shown and the error distribution is analyzed. In the second section, the impact of the two different types of input errors, position error and distance error is presented. Finally, in the third section, the results concern the question of the optimal number of range measurements per waypoint, as posed in section 6.4.

7.1. Localization Accuracy

Figure 7.1 shows in a box plot how the localization error is distributed. The red line marks the *median* error of all the 500 simulations, which amounts to 23.2 cm. Half of the errors lie inside the *inter-quartile range* which contains half of the errors. It ranges from 17.5 cm to 33.6 cm which is an interval of 16.1 cm. The two notches at the bottom and the top of the boxplot mark the *minimum* and the *maximum* error. The best experiment yielded an estimation error of only 0.2 cm. In the worst case experiment, the error amounted to 69.5 cm. All of those numbers are also displayed in table 7.1.

In figure 7.2 the localization error is displayed for each scenario separately. It stands out that for scenario six, the resulting errors were much bigger than for the other five. Its median error lies above the maximum error of scenarios two, three and five.

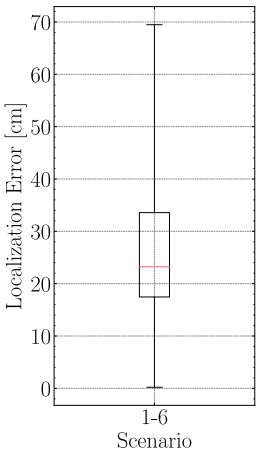


Figure 7.1.: Localization error statistics of trilateration algorithm.

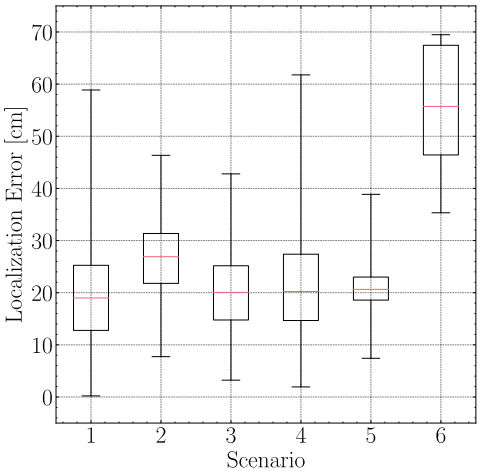


Figure 7.2.: Localization error for each scenario separately

7.2. Separate Inspection of Position and Distance Error

In this subsection the impact of the two different kinds of input error stated in section 6.3 are individually analyzed and compared. Figure 7.3 shows the error statistics for three different simulations: the first one with measured position and distance data as also seen in figure 7.1, the second one with ground truth position data, and the third one with ground truth distance data. It can be seen that the median error of the simulation that used ground truth position data became significantly smaller, namely 14.5 cm. For the ground truth distance simulation, the median error increased by 1.2 cm. The inter-quartile range became smaller for the ground truth position case, but it increased for the ground truth distance case. The best- and worstcase errors stayed nearly the same.

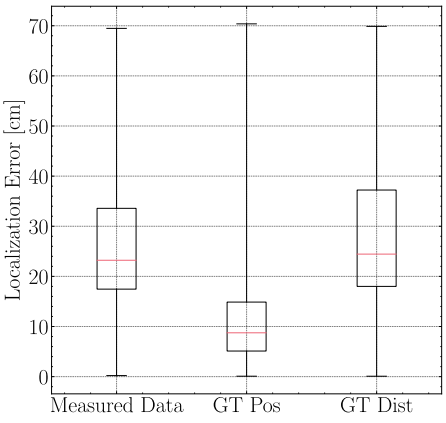


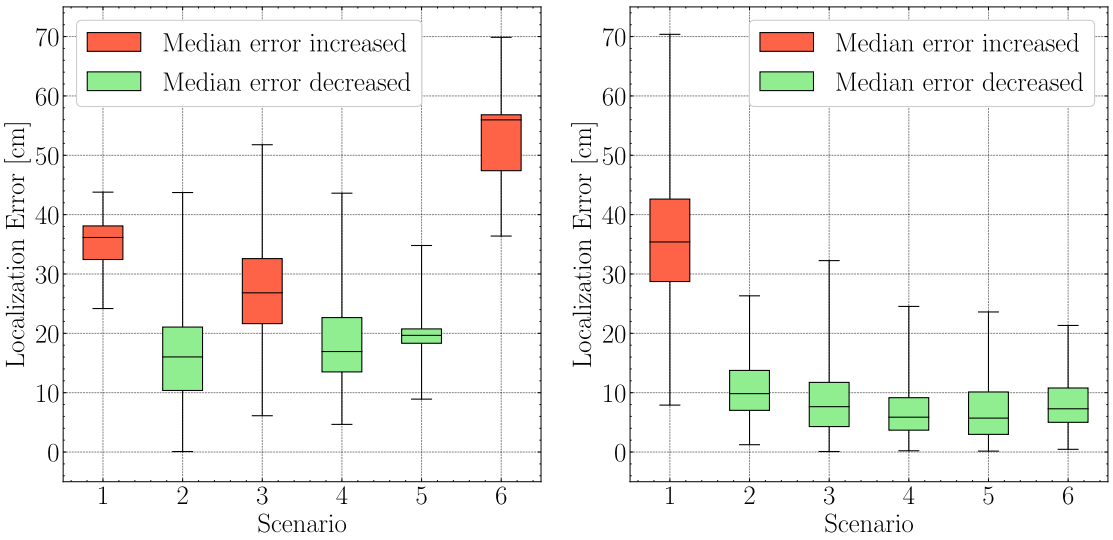
Figure 7.3.: Localization error compared for different ground truth data.

Figures 7.4a and 7.4b show the same results for each of the six scenarios separately. How the accuracy evolved using the respective ground truth data is indicated with colors: The scenarios that yielded a lower median error compared to the case where only measured input data was use are marked green, and the ones where the median error increased are marked red. Using ground truth distance data, half of the six scenarios achieved a better localization accuracy. In contrast, using

[cm]	Measured Data	Ground Truth Position	Ground Truth Distance
Max	69.5	70.4	69.9
75 percentile	33.6	14.9	37.2
Median	23.2	8.7	24.4
25 percentile	17.5	5.1	18.0
Min	0.2	0.1	0.1
Mean	27.9	12.8	28.7

Table 7.1.: Localization error statistics for different ground truth data

ground truth position data, five of six scenarios yielded a significantly lower localization accuracy. Only for scenario 1, the error increased. How wide-spread the error is, or in other words the error precision, can also better be analyzed in these two figures. They show that the seeming lack of improvement in error precision is in both cases due to only one single scenario. For the ground truth distance case, scenario six yielded significantly worse results, and for the ground truth position case the same is true for scenario one. Especially in this case the error precision of the single scenarios increased significantly. All the errors of scenarios two to five lie beneath 33 cm. The according numbers of the three boxplots can be seen in appendix C.



(a) Localization error with ground truth distance data.

(b) Localization error with ground truth position data

Figure 7.4.: Localization error compared to simulation without ground truth data, see figure 7.2.

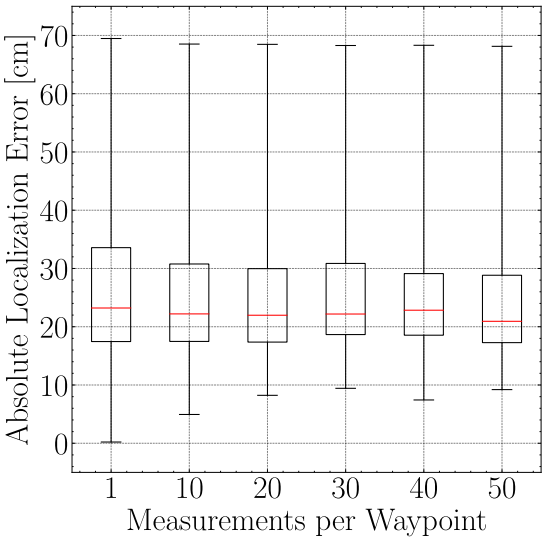


Figure 7.5.: Absolute error for different numbers of measurements.

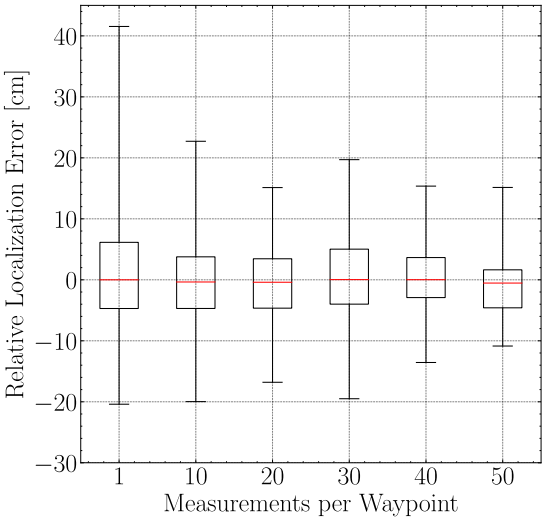


Figure 7.6.: Relative error for different numbers of measurements.

7.3. Accuracy improvement by Increased Number of Measurements

In the following, the results of the simulation described in the previous chapter in section 6.4 are shown. The localization error is evaluated for different numbers of measurements per waypoint. Figure 7.5 shows how the errors are distributed if up to fifty range measurements are taken at each waypoint. In order to better visualize the improvement on a smaller scale, Figure 7.6 shows the relative improvement compared to taking only one measurement per waypoint. The median error of the case where only one measurement is taken is subtracted from all the errors. Thus, the errors can be compared to zero in order to see how the accuracy improves or worsens compared to that particular case.

Chapter 8

Discussion

8.1. Localization Accuracy

First, the achieved localization accuracy as seen in section 7.1 will be put into context. With a median error of 23.2 cm, the error lies in the magnitude of the drone's size itself. Further, it has to be considered that the accuracy of the UWB transceiver itself is in the range of 10 cm.

In figure 7.2 where the localization errors are show for each scenario separately it is clearly visible how the experiments of scenario 6 yielded significantly higher localization errors. The median error of the 500 experiments lies above three maximum errors of other experiments. With its nearly sixty centimeters it is twice as high as the median errors of the other five scenarios. Figure 8.1 depicts the trajectory flown by the drone as well as the range measurements at each waypoint for one of the 500 experiments of scenario 6. What is striking is that the position error at the third waypoit is quite big compared to the other scenarios (the same illustration for all scenarios can be found on the gitlab repository[1]). That the large localization error is indeed due to the drone's poor position estimate is supported by the fact that the localization error vanishes if ground truth position data is used in order to calculate the trilateration, as

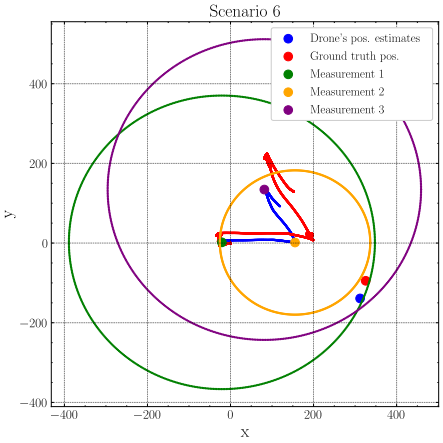


Figure 8.1.: Trilateration of Scenario 6.

[1] https://gitlab.ethz.ch/jaegeri/towards-autonomous-drones-with-ultra-wideband/-/wikis/Results

seen in figure 7.4b. However, the question that remains is why the position error has such a big impact on the localization accuracy for this specific geometric configuration of waypoints and node.

8.2. Position Error and Distance Error

The results shown in figure 7.4 clearly show that the drone's position estimation error has a much bigger influence on the localization accuracy than the measurement error of the range distances. The simulations with ground truth position data show that for five out of six scenarios the localization errors lie under 33 cm (see tables in appendix C). A possible explanation why scenario one yielded worse results when ground truth positions were used for the simulation could be found at the second waypoint: It can be seen in figures 8.2 and 8.3 that the position error and the distance error mutually cancel each other. The measured distance is too short, but the waypoint coordinate is shifted in negative x direction, which accomodates the distance error. In order to make a general statement however, further investigations of the measurement errors of that specific geometric contellation would be necessary.

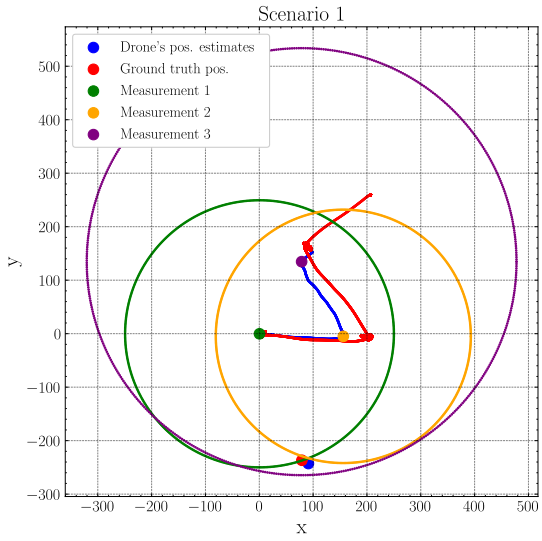


Figure 8.2.: Scenario 1 with measrued data.

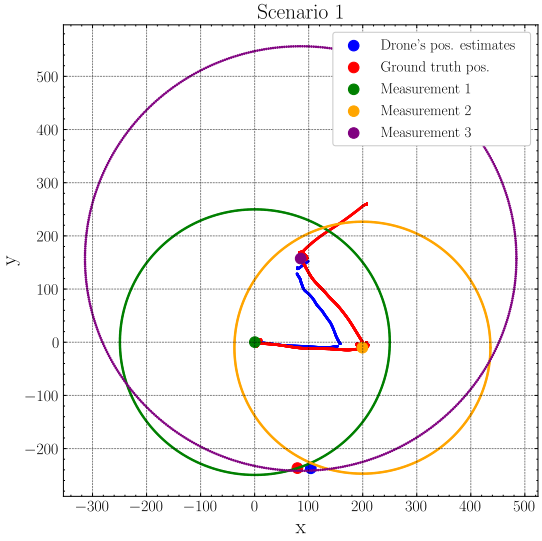


Figure 8.3.: Scenario 1 with ground truth position data.

8.3. Multiple Range Measurements per Waypoint

Figure 7.6 shows the relative improvement when taking more than one measurement per waypoint into account. It can be seen that the accuracy does not improve by increasing

the number of measurements. The median stays around zero, for all of the cases. Only the precision increases a bit, so the errors are less scattered around the median. This is true however not only for the worst cases, but also for the cases with a smaller error. The errors seem to converge against the median error when only one measurement is used. So, the accuracy stays the same, and the precision increases. Now the same

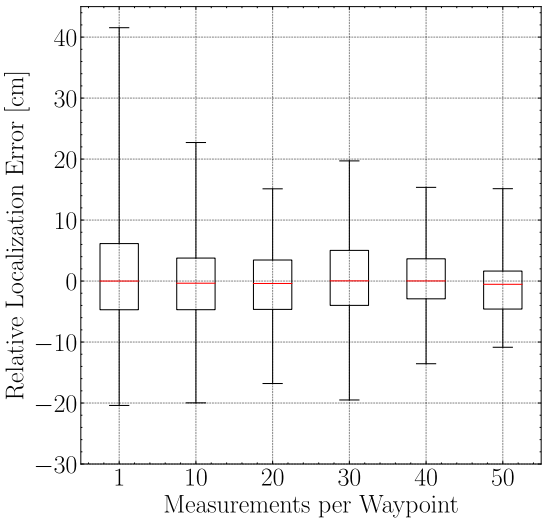


Figure 8.4.: Relative localization error with measured data.

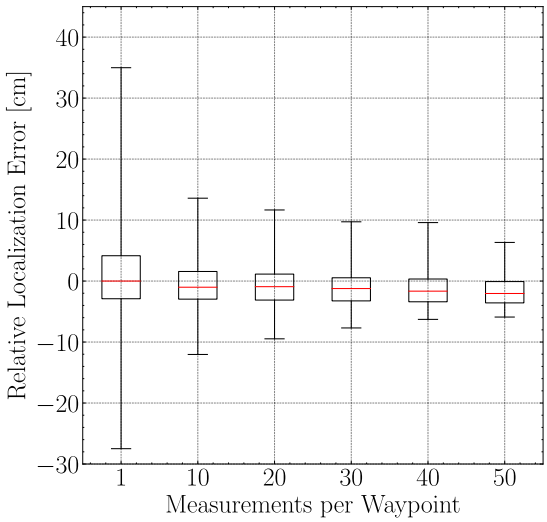


Figure 8.5.: Relative localization error with ground truth position data.

can be examined when the position error is ignored: Does increasing the number of measurements per waypoint increase the localization accuracy? Figure 8.5 shows that indeed, there is a slight improvement of the localization accuracy. The median error is about two centimeters lower if fifty measurements are acquired at each stop. The precision increases even more for that case. One could conclude that increasing the number of measurements only pays off if the position error can be eliminated. Otherwise, the improvement is so small that it can be interpreted as noise.

Chapter 9

Conclusion and Future Work

The goal of this thesis was to build a drone which is able to autonomously localize ultra-wideband nodes using trilateration. The final product, the drone built in this work, is able to stably fly indoors without the help of an external controller and without using GPS. It is able to autonomously localize UWB nodes with sub-meter localization accuracy using only three range measurements. In view of this, it can be said that the goal of this thesis was achieved. Further the localization accuracy was investigated and analyzed. It was shown that the localization error is mainly caused by the drone's erroneous position estmates. However, in order to better understand the localization error, both the ranging error as well as the position error must be further investigated.

The main contributions of my work include:

- The assembly of a custom drone
- The configuration of the drone using the open-source firmware INAV in order to ensure a stable and secure flight behaviour given only the onboard sensor data, without GPS support.
- The extension of INAV by the following features:
  - Communication between flight controller and UWB transceiver over UART
  - Trilateration algorithm for localizing UWB nodes
  - Realization of both in a new INAV flight mode
- The extension of the UWB ranging process by acknowledgement and report messages as well as further modifications in order to increase reliability.
- UWB communication between flight controller and UWB node for data logging.

- The acquisition of experimental data in order to evaluate the localization accuracy for different sensor orientations.

A possible next step could be to implement a recursive estimation algorithm such that the drone could also follow moving UWB nodes. Another interesting improvement would be to increase the number of UWB nodes and include them in the drone's localization process.

Appendix A

Task Description

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Center for Project-Based
Learning
Integrated Systems
Laboratory

Master Project at the
Department of Information Technology and
Electrical

Engineering

Towards Autonomous Drones with Sensor Fusion and UWB

Student: Ilian Jäger

Advisors: Vlad Niculescu
Michele Magno
Tommaso Polonelli

Professor: Prof. Luca Benini

Handout Date: 01.10.2020
Due Date: 01.04.2021

1 Project Goals

In the last decade, unmanned aerial vehicles (UAVs) have been used in many applications, such as aerial inspection, surveillance, and ambient awareness. A recent trend is their adoption as intelligent and ubiquitous coordinators of wireless sensor networks (WSNs), reliving from the otherwise necessary, expensive infrastructure, not always possible in all the application scenarios . WSNs are widely used for many practical applications, such as remote monitoring. Usually, WSNs contain a large number of simple, cheap, and resource-constrained (e.g., battery-powered) wireless sensor nodes deployed in the environment. Within a WSN, data are first collected in some local-host, which forwards the relevant information to a central infrastructure that can be geographically extremely decoupled from the deployment area. UAVs are envisioned as an alternative to traditional gateways to overcome their limits, such as limited bandwidth, range, energy consumption, and needs of infrastructure.

A UAV can quickly fly at any point in the monitored area and reach all the WSN’s nodes, reducing the transmission range/power, acting as an “intelligent and ubiquitous” localhost. However, in many real application scenarios, this vision is challenged by the unknown position of the WSN’s nodes. This is the case, for example, in dangerous or inaccessible areas, where the precise deployment of nodes is not feasible, as well as in all cases of dynamically changing environments (e.g., floods, landslides). Therefore, precise and fast localization of imprecisely positioned sensor nodes is essential in this challenging scenario.

The goal of the project is to build a localization system which uses a sensor fusion algorithm (with UWB) and allows an UAV to estimate the position of imprecisely positioned sensor nodes.

The student will have to build a drone platform and integrate it with the additional necessary hardware. Furthermore, the student will have to develop the algorithm and test the closed loop system by evaluating the localization accuracy with the motion capture system.

The project will be split up in four phases, as described below:

Phase 1 (2-3 months)

1. Familiarization with basic embedded programming notions as well as with the fundamental elements of a drone.
2. Building the experimental drone platform , integrate the UWB and get it operational.
3. Familiarization with how to program the drone.
4. **Porting the UWB localization algorithm on the new platform .**
5. Desing and implementation of eventual new development board for UWB or other sensors .
6. Investigation of novel sensor to use. For example mmWave Radar from Texas Instruments for autonmouns navigation.

Phase 2 (1 month)

1. Integrate the drone with the Vicon framework and evaluate the performance.
2. Investigare on the use of sensor fusion including **mmWave or MEMS sensors** and UAV to imrpvoe the navigation of the drone trought UWB anchors.
3. Investigation and adavanced control (MPC or other) for autonomous navigation.

Phase 3 (2 months)

1. Investigate new sensor options for enhancing the localization performance.
2. Integrate the new hardware within the current system.
3. Develop the sensor fusion algorithm
4. Preliminary filed test on the final version

Phase 4 (1 month)

1. Test the overall system.
2. Cleaning of the source code and software documentation.
3. Write the final document and prepare a presentation.

2 Milestones

By the end of **Phase 1** the following should be completed:

- Consolidation and in-depth understanding of the constitutive elements of a drone.
- An operational drone platform.
- Inestigation of sensors to be use in the following phases.

By the end of **Phase 2** the following should be completed:

- The MCU firmware that enables the drone to localize sensor nodes using UWB.
- P

By the end of **Phase 3** (TBD) the following should be completed:

- The MCU firmware that features the sensor fusion algorithm.

By the end of **Phase 4** (TBD) the following should be completed:

- Final presentation
- Final report, including final results.

3 Project Organization

3.1 Weekly Report

There will be a weekly report sent by the student at the end of every week. The main purpose of this report is to document the project's progress and should be used by the student as a way to communicate any problems that arise during the week. The report, along with all other relevant documents (source code, datasheets, papers, etc), should be uploaded regularly to the assigned shared account.

3.2 Final Report

A pdf copy of the report has to be turned in and remains the property of the Center for Project-Based Learning. A copy of the developed software needs to be handed in on CD or DVD at the end of the project.

3.3 Final Presentation

At the end of the project, the outcome of the thesis will be presented in a 20 minutes task, again during a group meeting of the Center for Project-Based Learning or the Integrated Systems Laboratory.

Appendix B

Declaration of Originality

ETH

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

TOWARDS AUTONOMOUS DRONES WITH ULTRA-WIDEBAND

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
JAEGER	IVAN

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**
Zürich, 30.3.2021	I. Jaeger

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.

Appendix C

Further Results

On the following three pages the results as shown in the result section are displayed again, but this time the boxplots are accompanied by all the important numbers related to them. The values of the notches, the percentiles as well as the medians are listed in the respective tables below the figures.

C.1. Results for measured data only

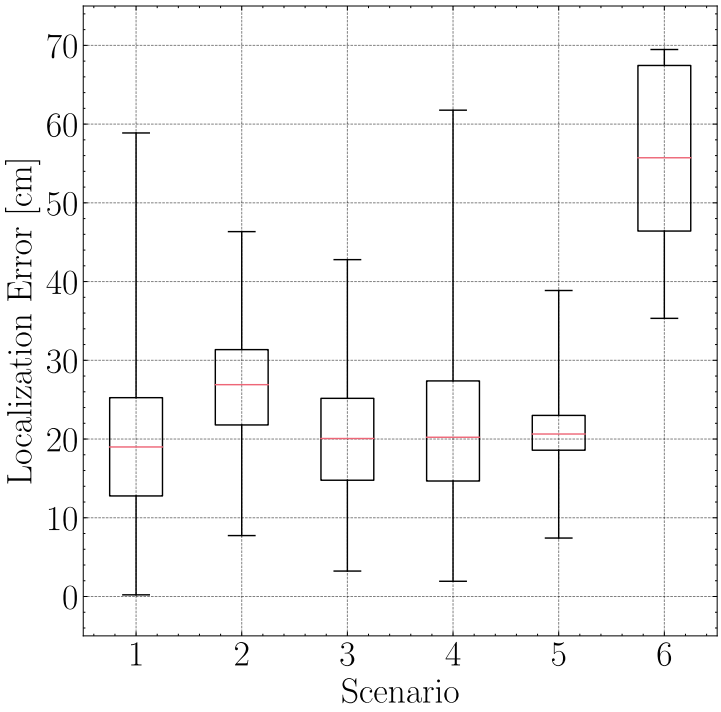


Figure C.1.: Localization errors for all six scenarios separately using only measured data.

Scenario	1	2	3	4	5	6
Max	58.9	46.3	42.8	61.8	38.9	69.5
75%-ile	25.3	31.4	25.2	27.4	23.0	67.4
Median	19.0	26.9	20.0	20.2	20.6	55.7
25%-ile	12.8	21.8	14.8	14.7	18.6	46.4
Min	0.2	7.7	3.2	1.9	7.4	35.3
Mean	19.7	26.7	20.7	21.7	21.2	57.2

Table C.1.: Localization error statistics for different ground truth data

C.2. Results for ground truth position data

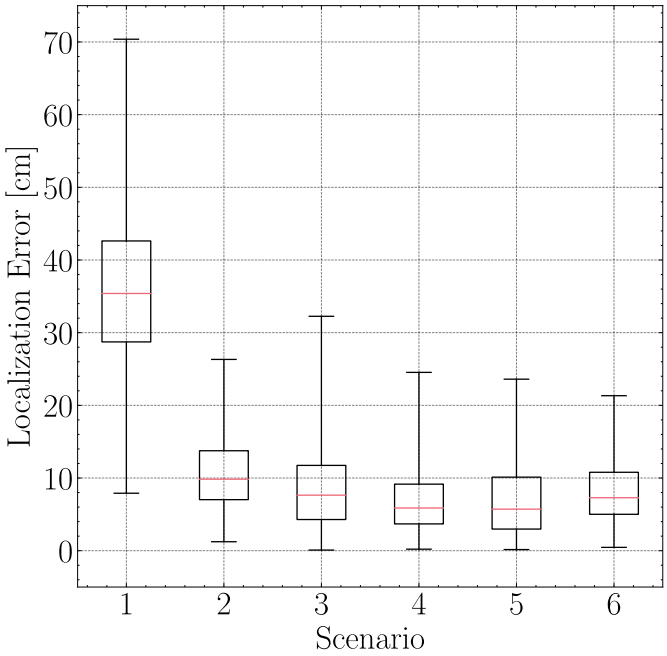


Figure C.2.: Localization errors for all six scenarios separately using ground truth position data.

Scenario	1	2	3	4	5	6
Max	70.4	26.3	32.3	24.5	23.6	21.3
75%-ile	42.6	13.8	11.7	9.2	10.1	10.8
Median	35.4	9.8	7.6	5.9	5.7	7.3
25%-ile	28.7	7.0	4.3	3.7	3.0	5.0
Min	7.9	1.2	0.1	0.2	0.2	0.5
Mean	35.2	10.8	8.5	6.8	6.9	8.2

Table C.2.: Localization error statistics for different ground truth data

C.3. Results for ground truth distance data

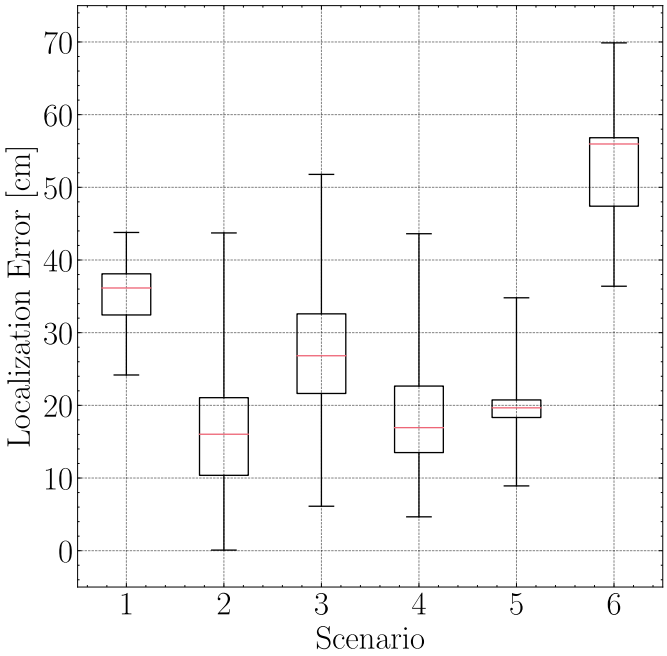


Figure C.3.: Localization errors for all six scenarios separately using ground truth distance data.

Scenario	1	2	3	4	5	6
Max	43.8	43.7	51.8	43.6	34.8	69.9
75%-ile	38.1	21.0	32.6	22.7	20.7	56.8
Median	36.1	16.0	26.8	16.9	19.7	56.0
25%-ile	32.4	10.4	21.6	13.5	18.3	47.4
Min	24.2	0.1	6.1	4.7	8.9	36.4
Mean	35.0	16.3	27.3	18.5	20.0	54.9

Table C.3.: Localization error statistics for different ground truth data

Appendix D

Getting Started

This chapter is about how to use the drone and set everything up from my thesis.

D.1. My platform and what I use

- Ubuntu 20.04 LTS
- STMCubeIDE version 1.4.0
- Debugger - ST-LINK v2
- openOCD version 0.10.0
- Betaflight version 4.2.5
- Betaflight configurator version 10.7.0
- INAV version 2.6.0 RC3
- INAV configurator verson 2.6.0-RC1

D.2. Set the drone up for flying

There are many youtube tutorials that lead you through all of the steps listed in this section. In case of questions, refer to those.

- Install the INAV configurator
- Flash the 2.6.0 release of INAV onto FC (like this)

- Restore Configuration backup. The configuration backup can be found in the backups/INAV folder of this repo.
  - Connect the drone to the configurator. If you have troubles connecting it, see first paragraph here
  - Open the CLI Tab
  - Click "Load from File"
  - Chose "INAV_cli_diff_all.txt" file and hit enter
- Install openTX
- Flash "opentx-x7-en" firmware on RC (if not already done). I used version 2.3.10. See this video for reference.
- Flash backup config on RC. The config file can be found in the "backups/openTX" folder of this repo.
- Connect drone to INAV configurator. Power the flight controller (12V, at least 500mA, or just by battery). Go to "Receiver" tab and see if the RC signal of the different sticks is displayed.
- In the model selection menu of the RC, select "IJ_2020-10"
- Go to the setup page (page 2) of the RC and choose channel 1 for the "Ch. Range" option (Channel 2 is assigned for the "debugger drone") as shown in figure D.1. Otherwise newly bind the RC to the receiver of the drone.
- Go to the "Modes" tab and see which switches enable which flight modes.

Figure D.1.: Channel settings.

D.3. Fly the drone

- Power drone and RC. Wait until the beeping stops.
- Enable HEADFREE and ANGLE mode. This is not mandatory, but it makes flying easier.
- If using HEADFREE, set the drone's head using the HEADADJ switch.
- Fly

D.4. Localize UWB nodes

- Clone modified INAV git repo
- Flash modified INAV onto the FC (after having flashed INAV 2.6.0 first)
  - Open the INAV Configurator, go to the "Firmware Flasher" tab
  - "Load Firmware [Local]"
  - Chose the correct hex file (e.g. git/inav/build/inav_2.6.0_MATEKF722SE.hex)
  - Make sure "full chip erase" is NOT selected. This would reset the configuration.
  - Click "Flash Firmware"
- In ParhamVersion, set NUM_MEASUREMENTS to the number of measurements you want to acquire at each waypoint. The number of waypoints must be the same for both the UWB board attached to the drone, and the UWB node.
- Note: This number has nothing to do with the NUM_MEASUREMENTS variable in INAV, which should be left at three.
- In ParhamVersion, set `tag_FSM_state_t state = INITIALIZE_INITIATOR`. Flash ParhamVersion onto UWB board attached to the drone.
- Connect the UWB board to the drone with the RX, TX and GND pin.
- Set `tag_FSM_state_t state = INITIALIZE_RESPONDER`. Flash ParhamVersion onto UWB responder node.
- Turn on RC. It is important that this happens before the drone is turned on, otherwise it will directly enter the localization flight mode.
- Connect UWB Board to the battery (via attached USB cable). Wait until the orange LED blinks three times
- Connect the FC to the battery. Wait until it stops beeping.

- Enable HEADFREE and ANGLE mode. This is not mandatory, but it makes flying easier.
- Lift off. Enable ALT_HOLD, POS_HOLD and SURFACE flight modes.
- Enter autonomous flight mode. In the configurator under the "modes" tab, this is the flight mode called USER1.

D.5. Acquiring measurements

In order to acquire time-synchronized measurements follow these steps.

- For Vicon: Install this wrapper for the VION Datastream SDK. Follow the steps as described in the repo's readme.
- Download this App in order to manage and synchronize all measurements (Ranging, Logging, Vicon). Use the "ilian" branch if you're working on Ubuntu. This GUI will make your life a lot easier. These are the pip packages I have installed on my virtualenv.
- VICON Ethernet Settings are shown in figure D.2.

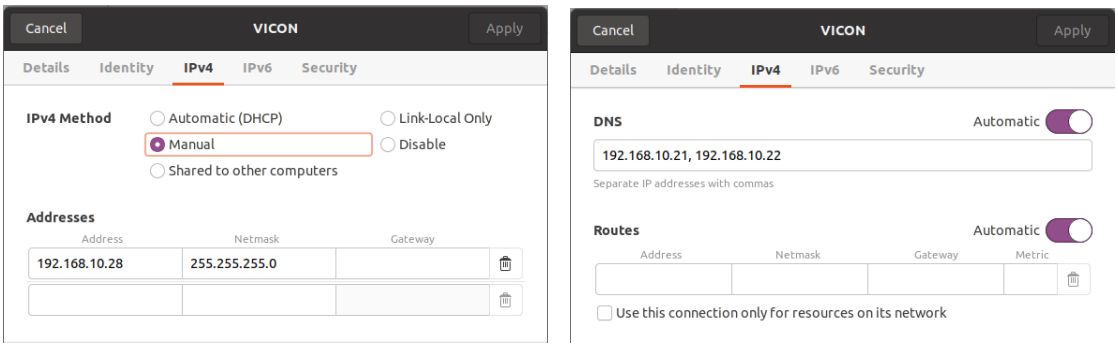


Figure D.2.: VICON Ethernet Settings

D.6. Simulating and Plotting

The simulation scripts as well as the scripts for generating plots can all be found on the official git repository of this thesis [1]. In this repo's wiki further explanations and guides for building a drone, simulating trilateration as well as compiling and debugging INAV can be found.

[1] `https://gitlab.ethz.ch/jaegeri/towards-autonomous-drones-with-ultra-wideband`

Bibliography

[1] L. Bhatia, D. Boyle, and J. A. McCann, "Aerial interactions with wireless sensors," in *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems*, ser. SenSys '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 373–374. [Online]. Available: https://doi.org/10.1145/3274783.3275189

[2] J. Kuriakose, S. Joshi, R. V. Raju, and A. Kilaru, "A review on localization in wireless sensor networks," in *Advances in signal processing and intelligent recognition systems*. Springer, 2014, pp. 599–610.

[3] N. H. Motlagh, M. Bagaa, and T. Taleb, "Uav-based iot platform: A crowd surveillance use case," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 128–134, 2017.

[4] D. B. Jourdan, D. Dardari, and M. Z. Win, "Position error bound for uwb localization in dense cluttered environments," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 44, no. 2, pp. 613–628, 2008.

[5] S. Post, "Drones - a vision has become reality," [Online; accessed 26-March-2021]. [Online]. Available: https://www.post.ch/en/about-us/innovation/innovations-in-development/drones

[6] F. Zafari, A. Gkelias, and K. K. Leung, "A survey of indoor localization systems and technologies," *IEEE Communications Surveys Tutorials*, vol. 21, no. 3, pp. 2568–2599, 2019.

[7] F. B. Sorbelli, S. K. Das, C. M. Pinotti, and S. Silvestri, "Precise localization in sparse sensor networks using a drone with directional antennas," in *Proceedings of the 19th International Conference on Distributed Computing and Networking*, ser. ICDCN '18. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: https://doi.org/10.1145/3154273.3154295

[8] C. M. Pinotti, F. Betti Sorbelli, P. Perazzo, and G. Dini, "Localization with guaranteed bound on the position error using a drone," in *Proceedings of the 14th ACM International Symposium on Mobility Management and Wireless Access*, ser.

MobiWac '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 147–154. [Online]. Available: https://doi.org/10.1145/2989250.2998178

[9] D. Wu, D. Chatzigeorgiou, K. Youcef-Toumi, and R. Ben-Mansour, "Node localization in robotic sensor networks for pipeline inspection," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 2, pp. 809–819, 2016.

[10] F. B. Sorbelli, S. K. Das, C. M. Pinotti, and S. Silvestri, "On the accuracy of localizing terrestrial objects using drones," in *2018 IEEE International Conference on Communications (ICC)*, 2018, pp. 1–7.

[11] M. Larsson, V. Larsson, K. Astrom, and M. Oskarsson, "Optimal trilateration is an eigenvalue problem," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 5586–5590.

[12] V. Niculescu, M. Magno, D. Palossi, and L. Benini, "An energy-efficient localization system for imprecisely positioned sensor nodes with flying uavs," in *18th IEEE International Conference on Industrial Informatics Online Event (INDIN 2020)*, 2020.

[13] D. Ltd, *DW1000 User Manual*, Decawave Ltd, 2017.

[14] S. Gezici, Zhi Tian, G. B. Giannakis, H. Kobayashi, A. F. Molisch, H. V. Poor, and Z. Sahinoglu, "Localization via ultra-wideband radios: a look at positioning aspects for future sensor networks," *IEEE Signal Processing Magazine*, vol. 22, no. 4, pp. 70–84, 2005.

[15] M. Z. Win and R. A. Scholtz, "Impulse radio: how it works," *IEEE Communications Letters*, vol. 2, no. 2, pp. 36–38, 1998.

[16] Z. N. Low, J. H. Cheong, C. L. Law, W. T. Ng, and Y. J. Lee, "Pulse detection algorithm for line-of-sight (los) uwb ranging applications," *IEEE Antennas and Wireless Propagation Letters*, vol. 4, pp. 63–67, 2005.

[17] W. Navidi, W. S. Murphy, and W. Hereman, "Statistical methods in surveying by trilateration," *Computational Statistics and Data Analysis*, vol. 27, no. 2, pp. 209–227, 1998. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167947397000534

[18] F. Thomas and L. Ros, "Revisiting trilateration for robot localization," *IEEE Transactions on Robotics*, vol. 21, no. 1, pp. 93–101, 2005.

[19] Mateksys, "Flight controller f722-se," [Online; accessed 18-March-2021]. [Online]. Available: http://www.mateksys.com/?portfolio=f722-se#tab-id-4

[20] ——, "Optical flow and lidar sensor 3901-l0x," [Online; accessed 18-March-2021]. [Online]. Available: http://www.mateksys.com/?portfolio=3901-l0x#tab-id-2

[21] Hobbywing, "X-rotor micro 60a 4in1 blheli-32 dshot1200," [Online; accessed 18-March-2021]. [Online]. Available: http://www.hobbywing.com/goods.php?id=653&filter_attr=

[22] FrSky, "Frsky," [Online; accessed 18-March-2021]. [Online]. Available: https://www.frsky-rc.com/product/xm-1g-sbus-non-telemetry/

[23] EMAX, "Emax rs2306-2750kv brushless motor (black)," [Online; accessed 19-March-2021]. [Online]. Available: https://hobbyking.com/en_us/emax-rs2306a-2750kv-black-version-motor.html?___store=en_us

[24] master airscrew, "Electric only - 6x3 propeller," [Online; accessed 29-March-2021]. [Online]. Available: https://www.masterairscrew.com/collections/electric-only/products/electric-only-6x3-propeller

[25] Quantum, "Quanum outlaw 270 racing drone frame kit," [Online; accessed 29-March-2021]. [Online]. Available: https://hobbyking.com/en_us/quanum-outlaw-270-racer-quadcopter-frame-kit.html

[26] M. Kok, J. D. Hol, and T. B. Schön, "Using inertial sensors for position and orientation estimation," *Foundations and Trends® in Signal Processing*, vol. 11, no. 1-2, p. 1–153, 2017. [Online]. Available: http://dx.doi.org/10.1561/2000000094

[27] A. Martinelli, "Vision and imu data fusion: Closed-form solutions for attitude, speed, absolute scale, and bias determination," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 44–60, 2012.

[28] Decawave, "Dwm1000 module," [Online; accessed 30-March-2021]. [Online]. Available: https://www.decawave.com/product/dwm1000-module/

[29] INAV, "Inav - navigation capable flight controller," [Online; accessed 19-March-2021]. [Online]. Available: https://github.com/iNavFlight/inav

[30] A. Ledergerber and R. D'Andrea, "Ultra-wideband range measurement model with gaussian processes," in *2017 IEEE Conference on Control Technology and Applications (CCTA)*, 2017, pp. 1929–1934.

[31] C. Mooney and i. Sage Publications, *Monte Carlo Simulation*, ser. Monte Carlo Simulation. SAGE Publications, 1997, no. Nr. 116. [Online]. Available: https://books.google.ch/books?id=xQRgh4z_5acC