

# **Implementation and Evaluation of mobile RSSI-based LoRa Localization**

Bachelor Thesis  
for  
*B.sc. Informatik*

*Faculty of Computer Science and Media*

*submitted by*  
**Mose Schmiedel**

**Prof. Jens Wagner**  
*First reviewer*

**Marian Ulbricht, M.Eng.**  
*Second reviewer*

## *Abstract*

*For this thesis a LoRa™ localization system for outdoor use was implemented and evaluated. The system uses RSSI measurements to estimate the distances between three anchor nodes and one end node. A trilateration algorithm is employed to estimate the position of the end node based on these distances. The implementation decisions of the localization system are explained and the performance of the system is evaluated experimentally. The evaluation highlights the limitations of RSSI-based localization and provides clues for future optimizations.*

**Keywords:** LoRa • RSSI • localization • distance estimation

## **EIDESSTATTLICHE VERSICHERUNG**

Ich erkläre hiermit, dass ich diese Bachelorarbeit selbstständig ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel verfasst habe. Alle den benutzten Quellen wörtlich oder sinngemäß entnommenen Stellen sind als solche einzeln kenntlich gemacht.

Diese Arbeit ist bislang keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht worden.

Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

---

Ort, Datum, Unterschrift

## **CONTENTS**

<b>1 Introduction .....</b>	<b>5</b>
<b>2 State of the Art .....</b>	<b>6</b>
2.1 Localization methods .....	6
2.1.1 Angle of Arrival .....	6
2.1.2 Time of Arrival .....	7
2.1.3 Time Difference of Arrival .....	7
2.1.4 Received Signal Strength Indicator .....	8
2.2 Low Power .....	8
2.3 Similar Work .....	10
2.4 Contribution of this Thesis .....	11
<b>3 Principles .....</b>	<b>12</b>
3.1 Path-Loss Model .....	12
3.2 Multilateration .....	12
<b>4 Implementation .....</b>	<b>14</b>
4.1 System Overview .....	14
4.2 Hardware .....	14
4.3 Firmware .....	15
4.3.1 Conditional compilation .....	16
4.3.2 Application state machine .....	16
4.3.3 Hardware Drivers .....	18
4.3.4 Utilities .....	19
4.4 Distance Estimation and Localization .....	20
<b>5 Evaluation .....</b>	<b>23</b>
5.1 Distance Estimation .....	23
5.1.1 Data Analysis .....	24
5.1.2 Results .....	26
5.2 Localization .....	26
5.2.1 Data Analysis .....	26
5.2.2 Results .....	29
5.3 Low Power .....	29
<b>6 Future Works .....</b>	<b>30</b>
<b>Bibliography .....</b>	<b>32</b>
<b>7 Appendix .....</b>	<b>35</b>

## 1 INTRODUCTION

With radio technology becoming smaller and smaller, a continuously increasing amount of devices feature some form of localization capability. One popular example for this is the Apple AirTag™ [1]. This device gives the broader public access to low power continuous tracking technology.

On the other hand with the rise of the internet of things (IoT) in industry 4.0 [2], tracking and localization technology is needed for tasks like asset tracking [3], [4], smart campus orientation [5] or collision avoidance [6]. But localization techniques are also essential for other applications like emergency rescue services [7], [8] or elderly care [9]. With or without IoT and industry 4.0 the need for precise localization and tracking technology is clear.

Parallel to the advancements of IoT and industry 4.0, a new radio communication technology LoRa™ was developed [10]. This technology consists of a new modulation technique, also called LoRa, and a networking layer, called LoRaWAN™. It promises long-range communication with minimal power consumption.

Using this new technology for localization systems seems like a good fit because the area of coverage can be maximized, while minimizing the power consumption.

To evaluate the feasibility and performance of LoRa™ based localization, a localization system based on the LoRa™ modulation technique is implemented in this thesis. The localization system uses the received signal strength indicator (RSSI) to estimate distances for the position estimation.

## 2 STATE OF THE ART

LoRa™ and LoRaWAN™ were released as a radio communication technology to the public in 2015 by Semtech. Very early on, it already received coverage in scientific media as it was introduced as a promising long-range radio communication technology for IoT devices [10].

This section presents relevant literature correlating with mobile RSSI-based LoRa™ Localization.

### 2.1 Localization methods

There are multiple approaches for localization in RF based communication networks. The most commonly used all depend on one or more of four physical metrics of the received radio signal. These properties are the angle of arrival (AoA), time of arrival (ToA), Time delay of arrival (TDoA) and received signal strength (RSSI) [11, p. 3].

The presented localization algorithms use fixed reference points with known positions to estimate the position of a device. The reference points will be called anchor nodes and the device will be called end node from now on.

#### 2.1.1 Angle of Arrival

Localization systems which are based on the angle of arrival use a directed antenna array to measure the angle at which the radio signal was received. The AoA information of the signal received at the anchor nodes and the distance between the anchor nodes is used to estimate the position of the end node via triangulation [12, p. 2]. This is done using sine and cosine of the measured angles in combination of the distance between the anchor nodes.

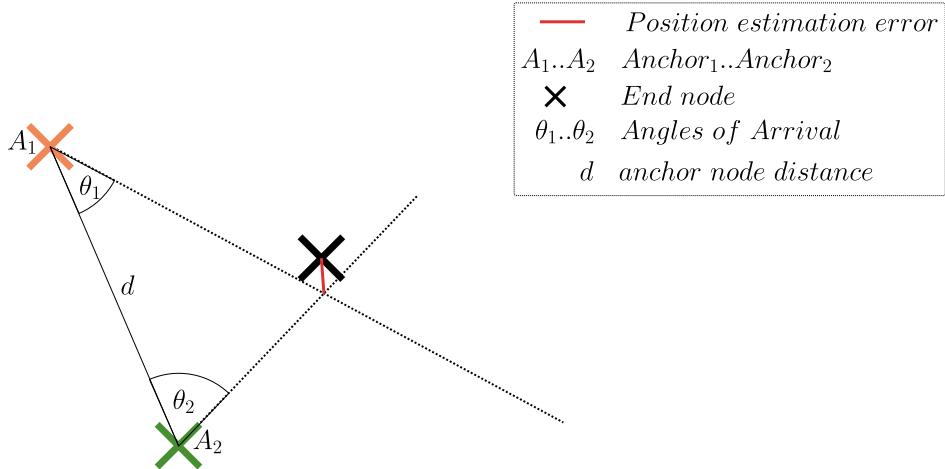
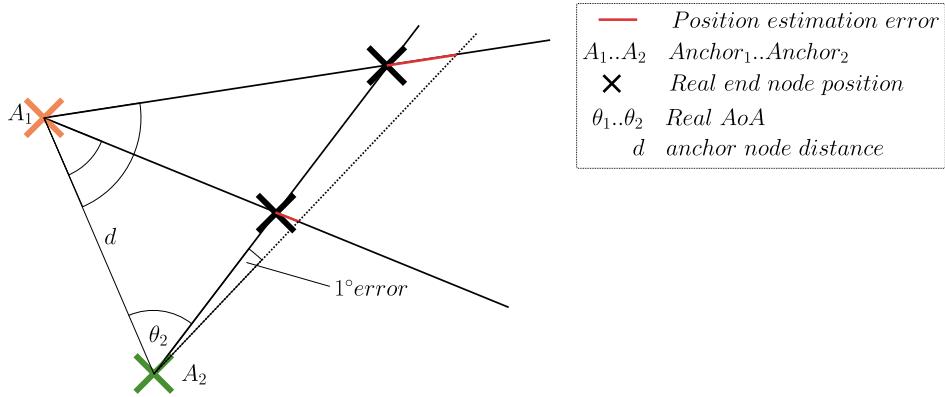


Figure 1: Triangulation of end node with two anchor nodes

AoA-based localization can achieve high position accuracy in short range applications. The accuracy is negatively correlated with increasing distance between anchor node and end node. This can be simply explained with the following graphic. As can be seen, the resulting change of the position when changing the angle by one degree is greater when the end node is farther away from the anchor node [13, p. 3].



*Figure 2: Triangulation error at different distances*

Another drawback of AoA-based localization methods is that they require additional hardware, not yet commonly implemented in all RF receivers. Also, the need of a directed antenna array makes this localization method unattractive for low-cost applications [13, p. 3].

### 2.1.2 Time of Arrival

Another localization method presented in the media uses the time of arrival (ToA) of the radio signal. The ToA is used to calculate the time of flight (ToF) which is the time the radio signal needs to travel from the transmitter to the receiver. From this time and the speed of light, the distance between transmitter and receiver can be calculated [14].

The position of the end node can be estimated when at least three distances between different anchors and the end node can be measured. From these distances, the position can be derived by using trilateration. Trilateration calculates the area of intersection between circles centered at the anchor nodes, each with the measured distance between the corresponding anchor node and the end node. An improved variant of this localization algorithm called multilateration is explained in Section 3.2 in more detail.

ToA-based localization systems need high precision timing capabilities and strict time synchronization between anchor and end node. The following calculation demonstrates that to measure distances up to 1 m the time measurement must provide an accuracy of  $\pm 3.3$  ns.

$$\text{ToF}_{1\text{m}} = \frac{1\text{ m}}{c} \approx \frac{1\text{ m}}{3 \cdot 10^8 \text{ m s}^{-1}} = 3.3 \cdot 10^{-9} \text{ s} \triangleq 3.3 \text{ ns} \quad (1)$$

Due to this requirement, ToA-based localization is more suited for long range applications. As with AoA-based localization, this technique also requires specialized hardware not found in every receiver circuit. Clock systems with high resolution and low drift over time and temperature are required to achieve the before mentioned requirements for precision and synchronization of the time measurement [11, p. 4].

### 2.1.3 Time Difference of Arrival

The strict time synchronization of anchor and end node is a requirement which is sometimes very difficult or even impossible to achieve in real world implementations. In such circumstances, it could be required that end nodes can be deployed dynamically such that they are not always powered on and not physically close to each other. These requirements add much complexity to the time synchronization process. One way to deal with this complexity is to avoid it all together by improving the ToA-based

distance measurement. Instead of using the ToF to estimate the distance between anchor and end node, the time difference between the ToA of different anchors is used to calculate the differences of the distances between the anchors and the end node. Through this optimization, time synchronization between anchor and end node is not required anymore, which decreases the complexity of the localization system. The anchor nodes still need to be synchronized so that the differences between the different ToA can be used effectively [15].

Despite solving the time synchronization challenge of ToA-based localization, TDoA-based localization inherits many of the drawbacks of ToA-based localization. The hardware used for TDoA-based localization still has to provide excellent resolution and drift properties.

#### *2.1.4 Received Signal Strength Indicator*

The last localization method presented in this section uses the Received Signal Strength Indicator (RSSI) of a received radio signal. Like the T(D)oA based method this localization technique also uses trilateration or multilateration to estimate the position of the end node. It differs in the distance estimation technique. Instead of relying on time of flight or a derived measurement, this method uses the decrease in signal strength to estimate the distance between anchor and end node.

The RSSI measured by the receiver depends on many influences like distance between sender and receiver, number of reflections or attenuation by obstacles (buildings, trees, hills) [16, p. 1]. In the literature, several models for radio propagation with focus on different environments or applications are presented. Examples include log-distance model [17, p. 4], Okumura-Hata model [18, p. 4] or Cost 231 model [19, p. 3]. Although there exists no common classification of propagation models, they can be grouped into different categories. One such grouping differentiates the models by the way, they are fitted [16, p. 2].

**Empirical models** rely on intensive measurement of the real behavior of the used radio system.

**Site-specific models** include factors which are derived from a detailed understanding of the environment where the system is later deployed.

**Theoretical models** use the underlying theory of electromagnetic propagation to calculate the path loss in an ideal environment.

In contrast to the other presented localization methods, RSSI-based localization does not need specialized hardware due to RSSI measurement circuit being available in most LoRa™ receiver chips. This makes this method ideal for low-cost applications. This benefit comes with a cost because this localization method cannot achieve accuracies as high as the other presented localization techniques [11, p. 11].

## **2.2 Low Power**

LoRa™ is radio communication technology promising low power consumption compared to conventional long range radio communication technologies [10, p. 7]. This is an active field of research because the power budget is an important design factor for mobile applications due to them being limited to battery-based power supplies.

One goal of this thesis is to evaluate RSSI-based localization in a mobile application. This implies, for the above mentioned reasons, that the evaluation of the resulting localization system must include some form of power consumption characterization. To find and evaluate the real power characteristics of devices using LoRa™ communication technology, several studies were performed already. Some of these works are presented in the following section.

Fargas et al. evaluate a LoRa™ localization system in [15] and compare its current consumption with a GPS + GSM based device. They found that their device had significant lower current consumption and therefore could drastically decrease the power requirements for devices with localization systems.

In [20] El Agroudy et al. present an outdoor localization system based on RSSI with ZigBee. They evaluate the power consumption of their implementation and find that it draws around 30 mA in “Active mode” and 6.7  $\mu$ A in “Sleep mode”. “Active mode” is the system state in which communication and localization takes place while “Sleep mode” describes the state in which all system activity is lowered to minimum.

El Agroudy et al. propose, based on the significant difference in current draw between Active and Sleep mode, to reduce the amount of time spent in Active mode by implementing a periodic wake-up event which changes the system state from Sleep mode to Active mode. With this approach the total power consumption of the device can be configured by adjusting the time interval between the wake-up events.

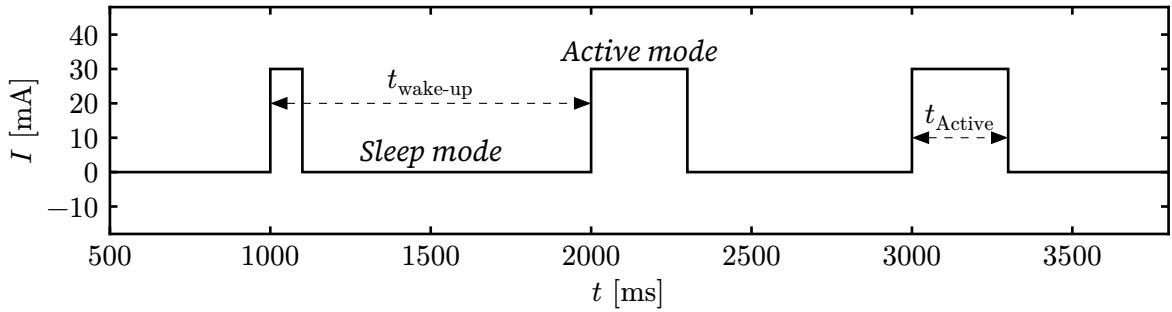


Figure 3: Current consumption in Active and Sleep mode

The total current consumption of this approach can be calculated when the current draw and the time spent in the individual modes is known. Following equation shows the final formula. The resulting current consumption is measured in Milliamperes hours (mAh).

$$\begin{aligned}
 t_{\text{wake-up}} &:= \text{Time interval between Wake-Up events} \\
 t_{\text{Active}} &:= \text{Time per interval spent in Active mode} \\
 I_{\text{total}} &= \frac{I_{\text{Active}} \cdot t_{\text{Active}} + I_{\text{Sleep}} \cdot (t_{\text{wake-up}} - t_{\text{Active}})}{t_{\text{wake-up}}} \quad (2)
 \end{aligned}$$

They used a 620 mAh battery and configured their system to be in the Active mode for 5% of the time. They found that through this method they could improve the life time of their battery from 14.46 hours to 288 hours.

A challenge for this approach is that a device cannot receive any signals while in Sleep mode. This challenge must influence the design of the localization algorithm to ensure that communication between the devices in the localization system is possible. El Agroudy et al. resolve this issue by keeping the anchor node of their localization system always in Active mode so that it can coordinate the communication.

In [8] Mackey et al. evaluate the power consumption of a LoRa™ localization system based on the Dragino LoRa™ v1.3 Shield and an Arduino Uno microcontroller. Like El Agroudy et al. they also implement an localization algorithm working with fixed communication intervals to reduce the amount of time the system is in active mode. They evaluated the power consumption of the receiver node (end node) and the transmitter node (anchor node) separately. They also varied the transmission power and transmission rate and found that they could configure their whole localization system, which consisted of three anchor nodes and one end node, so that it consumed

around 350 mW. They compared this with the power consumption of the GPS module integrated in the Dragino Shield and found that their system outperforms GPS-based localization when at least four end nodes use the deployed anchor nodes.

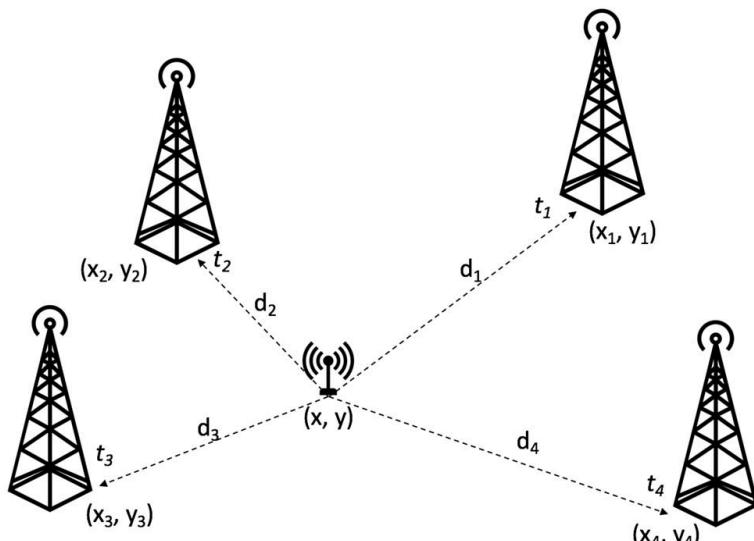
Despite these promising findings there is still room for improvement. Mackey et al. estimated the life time of a 5000 mAh battery when used as a power source for the anchor or the end node. They found that their anchor node could run up to 200 hours with one battery charge, but their end node would only last for 90 hours. These values would make this solution practical in ad-hoc short-term scenarios where a localization system is needed quick and only for a couple of days, but for remote long-term deployments improvements need to be made especially for the power consumption of the anchor node.

Gotthard et al. implement a LoRa™ RSSI-based system in [4], specialized for asset localization in large carparks. They found that with their localization technique a single node of their system could run for about 5 years powered by an 800 mAh CR2 battery. This result looks rather promising compared to the other findings presented in this section. But the system Gotthard et al. builds on fundamental assumptions taken about the environment and scenario the localization is performed in. This includes the assumption that many nodes are deployed in a relatively dense space which simplifies the problem with communication coordination explained earlier.

The assumptions make it difficult to adopt the localization system proposed by Gotthard et al. for general use but they investigate promising ideas which hopefully can be integrated into localization systems in the future to reduce their power consumption.

### 2.3 Similar Work

Fargas et al. evaluate LoRa™ for use in an alternative GPS-free geolocation system [15]. Their proposed approach for localization with LoRa™ signals is based on precise measurements of the Time of Arrival (ToA) of one packet at multiple LoRa™ gateways (anchor node). They then calculate the time difference of the different ToA timestamps and estimate the distance between the end node and each gateway by using the propagation velocity of radio waves, which is the speed of light. These distances are then combined by a multilateration algorithm to estimate the position of the end node.



*Figure 4: Position estimation with multilateration algorithm © 2017 IEEE*

Fargas et al. also showed in their work, that by using the Generalized ESD test for detecting and eliminating outliers in the distance measurement data, they could improve the accuracy of the position estimation.

The idea of GPS-less localization is further advanced by Mackey et al. in [8] where they evaluate LoRa™ as alternative localization system for Emergency Services. Instead of using the TDoA method for localizing the end node, they employ RSSI-based localization. For this they use an Arduino Uno microcontroller together with the Dragino LoRa™ v1.3 Shield, which both are readily available off-the-shelf hardware components. They evaluate their implementation on a soccer field of 200 m × 120 m where they achieve positioning accuracies up to 9 m.

In [21] Dieng et al. use RSSI-based LoRa™ Localization to track individual animals of a cattle herd to better deal with livestock theft. Their solution is based on a hybrid localization approach where they use both GPS and LoRa™ RSSI measurements to estimate the positions of the individual animals. They deploy hybrid nodes, nodes equipped with both LoRa™ and GPS, and LoRa-only nodes. The hybrid nodes are used to continuously improve the RSSI-distance model over time by correlating their GPS position with the current distance estimated by the RSSI-distance model.

In [4] Gotthard et al. evaluate RSSI-based LoRa™ localization as asset tracking system for large used car dealerships. They present a novel variant of RSSI-based localization where end nodes only send “ping” messages between one another. The RSSI acquired from these “pings” are transmitted to a central server, where they can be combined to approximate the position of the individual end nodes. Through this mechanism their proposed system does not require any anchor devices.

## 2.4 Contribution of this Thesis

As shown in Section 2.3 multiple implementations of LoRa™ localization exist already. This thesis focuses on the implementation of a LoRa™ localization system implemented with already available off-the-shelf components. The implemented localization system uses a range-based algorithm for position estimation. The distances needed for that are estimated with the RSSI. This path is chosen because almost every receiver hardware has RSSI measurement capability and is therefore widely accessible.

The goal of this thesis is to evaluate the feasibility of RSSI-based LoRa™ localization with the implemented localization system. For that the performance of the distance and the position estimation are evaluated and the power consumption of the system is measured.

### 3 PRINCIPALS

This section presents the fundamental theory needed for implementing and evaluating RSSI-based localization systems with LoRa.

#### 3.1 Path-Loss Model

To estimate distances based on the RSSI measured at the receiver a relation between RSSI and distance needs to be established. This relation is commonly modeled as the power loss experienced by the transmitted signal over distance. This is based on the assumption that the average power of a radio signal decays with the distance from the transmitter according to some deterministic attenuation law [22, p. 2]. This model is called Path-Loss Model (PLM).

Multiple Path-Loss Models exist in the literature [18], [17], [19]. For this thesis the most basic model is used. This model is derived from the equation for received power (in [dBm]).

$$P_{\text{rx}}(t, s) = D(d) + \Psi(s) + a(t) \quad (3)$$

In this equation  $D(d)$  is the deterministic component of the received power at distance  $d$ , while  $\Psi(s)$  and  $a(t)$  model the random variation in space and time. The relation  $D(d)$  can be modeled further as:

$$D(d) = P_0 - 10n \cdot \log_{10}\left(\frac{d}{d_0}\right) \quad \forall d \geq d_0 \quad (4)$$

As can be seen  $D(d)$  is only defined for distances  $d$  greater than  $d_0$ .  $d_0$  is the reference distance at which the initial received power  $P_0$  can be measured. The factor  $n$  is called the path-loss coefficient. It mainly influences the slope with which the model predicts the attenuation of the received power over distance [22, pp. 5-7].

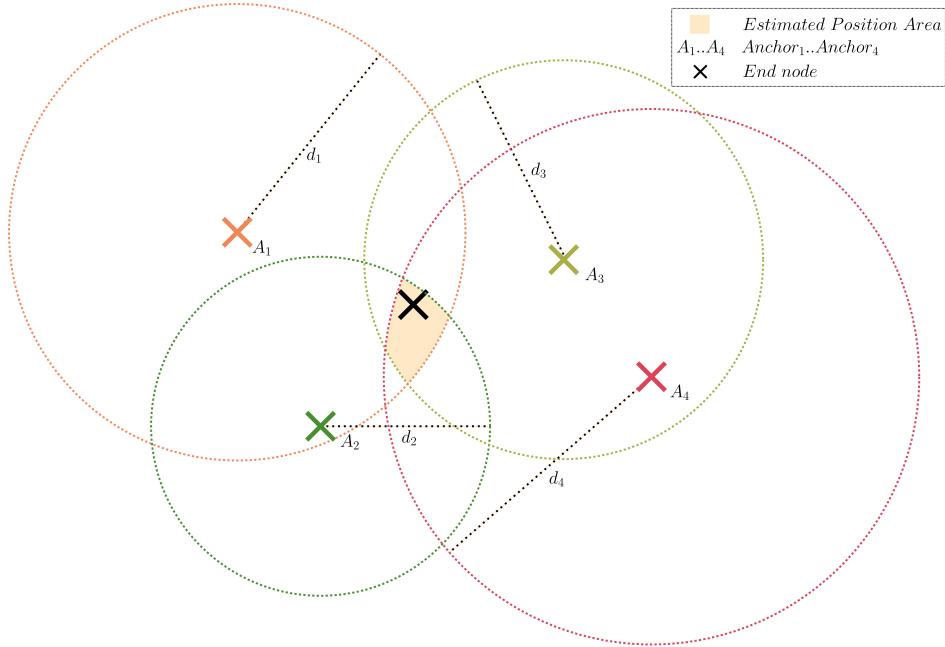
For this thesis  $\Psi(s)$  and  $a(t)$  are assumed as negligible and therefore the received power or RSSI is only modeled dependent on  $D(d)$ :

$$\text{RSSI}(d) = P_0 - 10n \cdot \log_{10}\left(\frac{d}{d_0}\right) \quad \forall d \geq d_0 \quad (5)$$

#### 3.2 Multilateration

Multilateration is a range-based position estimation algorithm to determine an unknown position. For this the distances between the node with unknown position (end node) and nodes with known positions (anchor nodes) are measured. At least three of these distances must be known to localize the end node.

The algorithm can be visualized by drawing circles, each with the measured distance at the anchor as radius, around the positions of the anchors. The position of the end node is estimated by the point of intersection of all the circles. In a perfect scenario this single point of intersection would exist, but in a real-world scenario the distance measurement always includes an error. Due to this error the circles all intersect at different points. These points describe an area in which the real position of the end node must be located.



**Figure 5:** Position estimation error with multilateration

To solve the multilateration problem algebraically the formulas for the circles are combined to form a system of equations. For the special case of three points  $P_1(x_1, y_1), P_2(x_2, y_2), P_3(x_3, y_3)$  with corresponding distances  $d_1, d_2, d_3$  the resulting system would look like this:

$$\begin{aligned} (x - x_1)^2 + (y - y_1)^2 &= d_1^2 \\ (x - x_2)^2 + (y - y_2)^2 &= d_2^2 \\ (x - x_3)^2 + (y - y_3)^2 &= d_3^2 \end{aligned} \quad (6)$$

This system can also be represented with matrices:

$$\begin{bmatrix} 1 & -2x_1 & -2y_1 \\ 1 & -2x_2 & -2y_2 \\ 1 & -2x_3 & -2y_3 \end{bmatrix} \begin{bmatrix} x^2 + y^2 \\ x \\ y \end{bmatrix} = \begin{bmatrix} d_1^2 - x_1^2 - y_1^2 \\ d_2^2 - x_2^2 - y_2^2 \\ d_3^2 - x_3^2 - y_3^2 \end{bmatrix} \quad (7)$$

In this form the general equation for an arbitrary amount  $n$  of anchor nodes  $a_n$  and distances  $d_n$  can be expressed as follows:

$$\begin{bmatrix} 1 & -2x_1 & -2y_1 \\ 1 & -2x_2 & -2y_2 \\ 1 & -2x_3 & -2y_3 \\ \vdots & \vdots & \vdots \\ 1 & -2x_n & -2y_n \end{bmatrix} \begin{bmatrix} x^2 + y^2 \\ x \\ y \end{bmatrix} = \begin{bmatrix} d_1^2 - x_1^2 - y_1^2 \\ d_2^2 - x_2^2 - y_2^2 \\ d_3^2 - x_3^2 - y_3^2 \\ \vdots \\ d_n^2 - x_n^2 - y_n^2 \end{bmatrix} \quad (8)$$

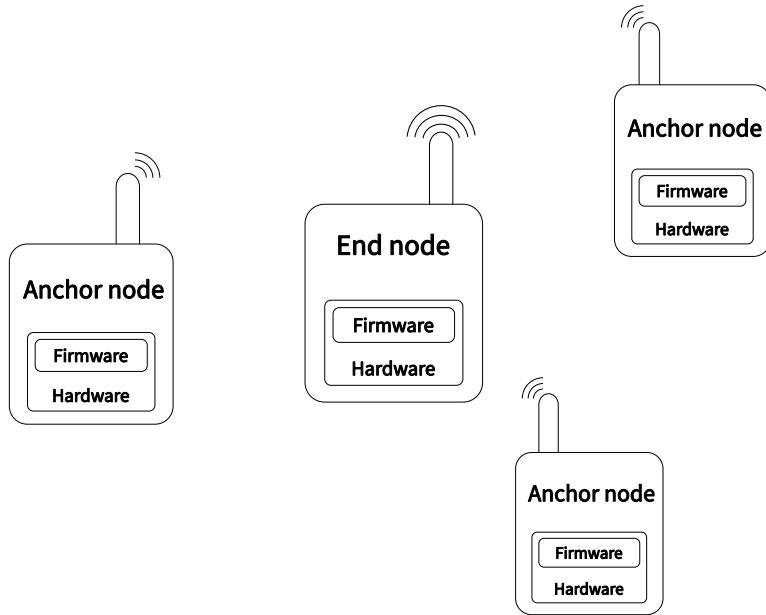
The derivation and solution of this system of equations is explained in more detail in [23]

## 4 IMPLEMENTATION

In the scope of this thesis a simple RSSI-based LoRa™ localization system was implemented. This system and the decisions which led to the final design are presented in this section. For the sake of conciseness from now on the term “localization system” references, where the context does not say otherwise, the localization system that was implemented for this thesis.

### 4.1 System Overview

The localization system implemented for this thesis consists of multiple components. Some of them have physical other have only logical boundaries. The following graphic illustrates the localization system as a whole and gives a quick overview.



*Figure 6: LoraLocator system overview*

The graphic shows that there are two different types of devices Anchor nodes and End nodes. These serve different purposes in the localization system. The End node is the device which position should be localized. The Anchor nodes serve as fixpoints for the measurements taken to estimate the position of the End node. The system which was implemented and evaluated in this thesis consists of three Anchor nodes and one End node. In-depth explanations of the individual components that make up one Anchor or one End node follow in the subsequent sections of this chapter.

### 4.2 Hardware

Both device types of the localization system are built with the same hardware and differ only in firmware. Every device consist of a NUCLEO-WL55JC [24] this is a evaluation board for the the STM32WL55JC microcontroller [25] developed and manufactured by STMicroelectronics (ST) [26]. This board features LoRa™ transceiver, an integrated programmer and debugger, connectors for the GPIO pins of the microcontroller and easy power supply over Micro-USB. This solution was chosen for several reasons. Most important is of course that the hardware satisfies the basic requirements imposed by the goal of this thesis to implement and evaluate a mobile RSSI-based LoRa™ localization system. This criterium is fulfilled by the NUCLEO-WL55JC evaluation board because it provides an integrated LoRa™ transceiver with built-

in RSSI measurement and promises comparatively low power consumption in the datasheet [27]. Another reason for which the NUCLEO-WL55JC was chosen is the benefit of using prebuilt hardware with a big ecosystem like that of ST. This allows for rapid prototyping by focusing development work on the parts where this localization system differs from previous implementations and reusing boilerplate code where possible. One such part is *SubGHz\_Phy* [28, pp. 25-27] which is a driver for the LoRa™ transceiver of the STM32WL55JC.

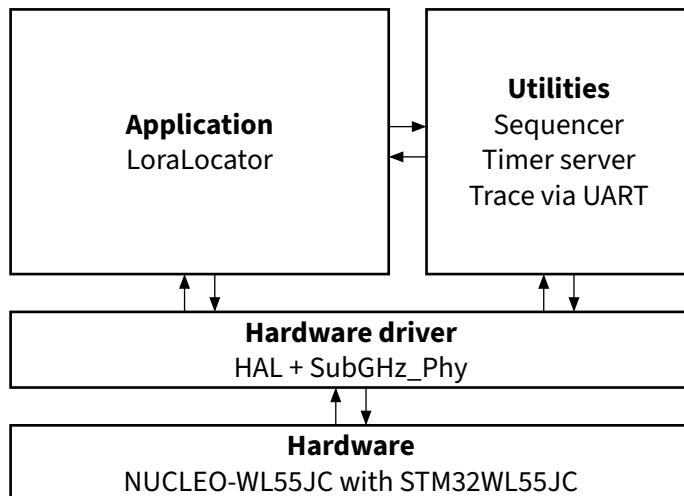


*Figure 7: NUCLEO-WL55JC*

### 4.3 Firmware

The firmware for the NUCLEO-WL55JC board was written in the programming language C and utilizes prebuilt drivers and libraries provided by STMicroelectronics. The source code was managed with the version control system *git* and uploaded to *GitHub* [29].

The implementation of the firmware is based on the *SubGhz\_Phy\_PingPong* example application provided by STMicroelectronics [28, pp. 44-46]. The following graphic illustrates the individual logical components the firmware is made of.



*Figure 8: Firmware architecture*

The main component of the firmware is the *LoraLocator* application. It controls all the other components and implements the logic that is needed for estimating the position of an End node.

#### 4.3.1 Conditional compilation

As described earlier, the firmware of the Anchor node and the End node differ because they both fulfill different tasks in the localization system. Despite these differences these also share some similarities especially in the way they both process incoming and outgoing communication data via LoRa. To avoid unnecessary code duplication both the firmware for the Anchor node and for the End node share the same code-base and node type specific code sections are conditionally included upon compilation depending on which device type was selected via preprocessor macros. For this purpose there are three macro definitions which must be configured when compiling the firmware.

```
1 // 1=is anchor node; 0=is not anchor node
2 #define IS_ANCHOR_NODE 1
3
4 // 1=is end node; 0=device is not end node
5 #define IS_END_NODE 0
6
7 // unique device identifier
8 // - end node: numeric as uint8_t
9 // - anchor node: alphabetical as char
10 #define DEVICE_ID 'A'
```

*Listing 1: Device type and ID configuration*

These three macro definitions are then used inside macro conditionals which control which part of the source code is included in the firmware build. Following code snippet demonstrates a use of a macro conditional.

```
1 #if IS_ANCHOR_NODE==1 && IS_END_NODE==0
2 // Anchor node specific code goes here...
3 #elif IS_ANCHOR_NODE==0 && IS_END_NODE==1
4 // End node specific code goes here...
5 #else
6 #error "Set atleast/only one of IS_ANCHOR_NODE and IS_END_NODE to 1."
7 #endif
```

*Listing 2: Device type specific code compilation*

#### 4.3.2 Application state machine

The driving design concept of the LoraLocator application is the finite state machine. The current state is tracked by the `node_state` variable, state changes are performed by the event handlers and the logic executed in a specific state is handled by the core application function `LoraLocator_Process()`.

All application specific state is centralized and managed with a fixed amount of memory. All variables needed can be found at the top of the file `lora_locator_app.c` [29].

```
1 typedef enum {
2     NODE_STATE_INIT,
3     NODE_STATE_INTERVAL_START,
4     NODE_STATE_RX_END,
5     NODE_STATE_TX_END,
6 } NodeState_t;
7
8 static NodeState_t node_state = NODE_STATE_INIT;
```

*Listing 3: Track last performed action*

The variable `node_state` is used in the core application function `LoraLocator_Process()` to determine the last action that was performed before the core function was called. The possible node states are:

**NODE\_STATE\_INIT** State the application is in after initialization. Start of the application lifecycle.

**NODE\_STATE\_INTERVAL\_START** State after the main application interval was triggered. `Ping_t` is transmitted or received depending on device type.

**NODE\_STATE\_RX\_END** State after node was receive mode. The received packet is decoded and the next action is triggered depending on device type, received packet type and if reception was successful.

**NODE\_STATE\_TX\_END** State after node was transmission mode. Next action is triggered depending on device type, transmitted packet type and if transmission was successful.

The following listing shows how the behavior of the core function is dependent on the state recorded in the `node_state` variable.

```

1 void LoraLocator_Process(void) {
2     switch(node_state) {
3         case NODE_STATE_INIT: {
4             // execute NODE_STATE_INIT logic and trigger next action
5             }break;
6         case NODE_STATE_INTERVAL_START: {
7             // execute NODE_STATE_INTERVAL_START logic and trigger next action
8             }break;
9         case NODE_STATE_RX_END: {
10            // execute NODE_STATE_RX_END logic and trigger next action
11            }break;
12        case NODE_STATE_TX_END: {
13            // execute NODE_STATE_TX_END logic and trigger next action
14            }break;
15        }
16    }
```

*Listing 4: Execute logic depending on last performed action*

As mentioned before, all state changes occur in event handler functions. These are subroutines that are called when some kind of event occurs, for example when a LoRa™ transmission is successfully terminated. The functions are passed as function pointers to a hardware driver or some other event-based component. The component receiving the event can then delegate the handling of the event to the function defined by the programmer by calling it via the provided function pointer.

```

1 static RadioEvents_t RadioEvents;
2 ...
3 RadioEvents.TxDone = &OnTxDone;
4 RadioEvents.RxDone = &OnRxDone;
5 RadioEvents.TxTimeout = &OnTxTimeout;
6 RadioEvents.RxTimeout = &OnRxTimeout;
7 RadioEvents.RxError = &OnRxError;
8
9 Radio.Init(&RadioEvents);
```

*Listing 5: Pass function pointer to radio driver for event handling*

In the next listing the definitions of all event handlers used in the *LoraLocator* app are listed. Note the usage of the prefix `On`, followed by a description of an event, to signal that the function is called when the specified event occurs.

```

1 // interval_timer elapsed
2 static void OnIntervalEvent(void *context);
3
4 // listen_timer elapsed
5 static void OnListenEndEvent(void* context);
6
7 // Transmission done
8 static void OnTxDone(void);
9
10 // Timeout triggered while transmitting
11 static void OnTxTimeout(void);
12
13 // Timeout triggered while receiving
14 static void OnRxTimeout(void);
15
16 // Error occurred during reception
17 static void OnRxError(void);

```

*Listing 6: List of event handlers used in LoraLocator*

To complete the implementation of the application state machine the state stored in the `node_state` variable must be changed in order to accomplish a transition from one state to another. This is done in the event handler functions by setting the state via the helper function `SetState(NodeState_t next_state)` and calling `QueueLoraLocatorTask()` to signal the scheduler to run the `LoraLocator_Process()` function again. The corresponding code can be found in the listing below on lines 5 and 6.

```

1 static void OnTxDone(void) {
2     State = TX;
3     switch (tx_result.packet_type) {...} // store and log transmitted packet
4     tx_result.state = RESULT_OK;
5     SetState(NODE_STATE_TX_END);
6     QueueLoraLocatorTask();
7 }

```

*Listing 7: Example state change in transmission-done event handler*

#### 4.3.3 Hardware Drivers

To develop firmware for the STM32WL55JC microcontroller the hardware drivers developed by STMicroelectronics were used. They provide an abstraction layer over the hardware so that the programmer can access hardware functionality via high level functions instead of configuring the peripheral registers themselves. This abstraction layer has the very creative name **HAL**, which stands for **H**ardware **A**bstraction **L**ayer. Its documentation can be found here [30].

For controlling the integrated LoRa™ transceiver of the STM32WL55JC microcontroller the *SubGHz\_Phy* driver was used additionally. It already implements the most common used functionality like sending and receiving packets via LoRa™ and provides an easy to use API with a single object `Radio` which is used to configure and control the radio transceiver. An example of how the event handlers are configured was demonstrated earlier in Listing 5. Transmission and reception of LoRa™ packets can be triggered by calling methods of the global `Radio` object, demonstrated in the following listing.

```

1 #define PAYLOAD_LEN 10
2 uint8_t tx_buffer[PAYLOAD_LEN];
3 Radio.Send(tx_buffer, PAYLOAD_LEN);
4
5 #define RX_TIMEOUT_MS 200
6 uint8_t rx_buffer[PAYLOAD_LEN];
7 Radio.Rx(rx_buffer, RX_TIMEOUT_MS);

```

*Listing 8: Switch LoRa™ transceiver to transmission-/reception-mode*

The code above transmits a payload of 10 bytes via a LoRa™ packet, as demonstrated in lines 1 to 3. Lines 4 to 6 show how the transceiver can be configured to receive the transmitted packet. Reception will fail after the time specified by RX\_TIMEOUT\_MS which in this case would be 200ms. Of course, sending and receiving should be done by separate devices at the same time for them to communicate successfully.

A more detailed explanation of how to build LoRa™ applications with the *SubGHz\_Phy* driver and the documentation of the all available methods can be found in here [28].

#### 4.3.4 Utilities

Several utility modules distributed by STMicroelectronics were used to simplify the firmware development process. They helped reduce the amount of boilerplate code, i.e. source code required for basic project setup that has little variation between different projects. Figure 8 lists all the utility modules used for the final firmware. This section briefly introduces the two most influential modules.

The firmware uses the **Sequencer** module to schedule the execution of the *LoraLocator\_Process()* function. This allows the event handler functions, which are often executed in an interrupt context, to indirectly call the core function and leave the interrupt context quickly. The documentation of the Sequencer module can be found here [31].

```

1 UTIL_SEQ_RegTask(
2     (1 << CFG_SEQ_Task_SubGHz_Phy_App_Process),
3     UTIL_SEQ_RFU,
4     LoraLocator_Process);

```

*Listing 9: Sequencer task registration*

```

1 static void QueueLoraLocatorTask() {
2     UTIL_SEQ_SetTask(
3         (1 << CFG_SEQ_Task_SubGHz_Phy_App_Process),
4         CFG_SEQ_Prio_0);
5 }

```

*Listing 10: Helper function to schedule task execution*

Basic timing functionality is implemented in the *LoraLocator* application with the **Timer server** module. This module allows to create an arbitrary amount of independent timers. A timer generates a timeout event and calls a provided event handler function after a fixed amount of time provided upon creation. The time is specified in Milliseconds. Unfortunately no official documentation could be found for this module but most important functionality can be understood by directly reading the source code [32].

```

1  /* Timer that triggers `LoraLocator_Process` periodically to either transmit a
2   `Ping_t` (end node) or listen for a `Ping_t` (anchor node). */
3  static UTIL_TIMER_Object_t interval_timer;
4  ...
5  UTIL_TIMER_Status_t timer_result = UTIL_TIMER_Create(
6      &interval_timer,           // timer object
7      INTERVAL_PERIOD_MS,       // timeout value
8      UTIL_TIMER_PERIODIC,      // timer mode (ONESHOT=run once and stop,
9      // PERIODIC=run and restart until stopped)
10     &OnIntervalEvent,         // callback function to call when timer elapses
11     NULL);                  // argument passed to callback function
12
13 if (timer_result != UTIL_TIMER_OK) {
14     // handle timer creation error
15     APP_LOG(TS_ON, VLEVEL_M, "Could not create interval timer.\n\r");
16 }

```

*Listing 11: Timer creation*

```

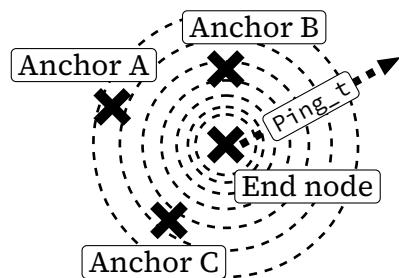
1 UTIL_TIMER_StartWithPeriod(&interval_timer, INTERVAL_PERIOD_MS);

```

*Listing 12: Start timer with specified timeout value*

#### 4.4 Distance Estimation and Localization

In order to estimate the position of an end node by trilateration, it is necessary to determine the distances between the end node and at least three anchor nodes. As stated in the thesis title and in preceding chapters, the implemented localization system utilizes RSSI measurements to estimate these distances. For this to work the end node periodically transmits LoRa™ packets of type `Ping_t`.



*Figure 9: 'Ping\_t' transmission*

```

1 typedef struct {
2     // attribute used to discriminate
3     // between packet types
4     PacketType_t packet_type;
5     // ID of the device
6     // sending the `Ping_t`
7     uint8_t device_id;
8     // ID of the `Ping_t`, `device_id`
9     // combined with this should be unique
10    uint8_t packet_id;
11 } Ping_t;

```

*Listing 13: Packet type 'Ping\_t'*

An anchor node receiving a `Ping_t` responds to it by transmitting an `AnchorResponse_t` which includes the RSSI measured by the anchor node while receiving the `Ping_t`.

```

1 // Note that this packet does not need a `packet_type` discriminator because
2 // it is the only type that is 4 bytes long.
3 typedef struct {
4     // ID of the anchor sending the `AnchorResponse_t`
5     Device_t anchor_id;
6     // ID of the `Ping_t` that triggered this `AnchorResponse_t`
7     uint8_t packet_id;
8     // RSSI of `Ping_t` measured by the anchor node
9     int16_t recv_rssi;
10 } AnchorResponse_t;

```

*Listing 14: Packet type 'AnchorResponse\_t'*

During the transmission of the `AnchorResponse_t` a collision could occur which hinders the end node from decoding the packet. This happens when multiple anchor nodes start transmitting its response almost at the same time. To detect when this is happening another packet type `Ack_t` is introduced. An anchor nodes always expects an end node to respond to an successful `AnchorResponse_t` with an `Ack_t`. If the anchor node does not receive this `Ack_t` in a configurable amount of time it retries sending the `AnchorResponse_t`. The maximum of retries can also be configured via the macro `MAX_ANCHOR_RESPONSE_RETRIES`, which defaults to 3. When this number of retries is reached the anchor node gives up and goes to sleep until it receives the next `Ping_t`.

```

1 typedef struct {
2     // attribute used to discriminate between packet types
3     PacketType_t packet_type;
4     // ID of the anchor this `Ack_t` is addressed to
5     Device_t receiver_id;
6     // ID of the `Ping_t` that triggered the communication
7     uint8_t packet_id;
8 } Ack_t;

```

*Listing 15: Packet type 'Ack\_t'*

The end uses the decoded RSSI value `recv_rssi` from an successful `AnchorResponse_t` to calculate the distance between itself and the anchor node with a path-loss model. The chosen path-loss model is the log-normal model which is explained in more detail during the evaluation of the distance estimation in Section 5.1. The estimated distances can then be combined to calculate a position relative to the positions of the anchor nodes by employing the trilateration algorithm or its generalization the multilateration algorithm. These algorithms are explained in Section 3.2. To obtain a absolute position the relative position can be added to the absolute position of an anchor node.

The rest of this section demonstrates the architectural design decisions of the localization system by highlighting some of the key features of the chosen implementation.

A significant feature of this localization system is the amount of control the end node can exert. Because the whole system relies on the periodic `Ping_t` packets of the end node, it can effectively regulate the frequency with which measurements are taken. This frequency is referred to as tracking rate. This equips the end node with a mechanism to increase or reduce the localization precision on demand, which is useful in a situation where the velocity of the end node changes. In this scenario the end node could increase the tracking rate while traveling at a high velocity and subsequently reduce the tracking rate, thereby lowering power consumption, when traveling at a low velocity.

Another benefit of the end node transmitting the `Ping_t` packets, is that multiple anchor nodes can receive the same `Ping_t`. This mitigates the impact of variations in the transmitter circuit or other undesired environmental influences on the localization process.

The main drawback of this approach is the amount of transmitted packets. In comparison to the distance estimation approach implemented by Bluetooth Low Energy beacons [33], [34], this implementation needs twice as much packets for a single localization cycle. This is because with the BLE approach the anchor nodes periodically send short packets which are received by the end node and are used to estimate the distances between itself and the anchor nodes without the need for response to the ping message.

Despite this drawback, the benefits would enable some interesting advantages over existing solutions. For this reason this localization method was chosen for the evaluated system.

## 5 EVALUATION

In this chapter two experiments, conducted for evaluating the performance of the implemented localization system, are presented and the resulting data is discussed.

### 5.1 Distance Estimation

The distance estimation was evaluated with two devices, an anchor and an end node. As previously described, the end node periodically sent `Ping_t` packets which the anchor node responded with `AnchorResponse_t` packets which included the RSSI values it measured. These measurements were taken at different distances. At each distance multiple measurements were performed so that the average RSSI value per distance can be calculated to reduce the impact of RSSI variations. The measured distances reached from 5m to 50m and were measured with a tape measure. The end node and the anchor node were mounted on two wooden poles at approximately 1.5 m above the ground with the antennas parallel to each other.

To also measure larger distances, the experiment setup was slightly modified. Instead of using a tape measure as reference for the distance, the iOS-App “GPS Tracks” [35] was used. With it the position of the anchor node was recorded as a waypoint. All distances of the end node were then measured relative to this waypoint. The collection of the RSSI measurements stayed the same. This setup was used to measure distances in a range from 10m to 160m.

The recorded data is openly accessible and can be found in the GitHub repository of this thesis [36].



*Figure 10: Distance estimation setup at multiple distances*

The devices that were used for the experiments were described previously in Section 4. For all experiments the LoRa™ transceiver was configured with following parameters.

Parameter	Value
Frequency	868 MHz
Bandwidth	125 kHz
Spreading Factor	7
Coding Rate	4/5
Preamble Length	8
Output Power	+14 dBm

*Table 1: LoRa™ transceiver configuration*

### 5.1.1 Data Analysis

Multiple experiment runs were performed to evaluate the distance estimation method presented in this thesis. Based on Equation 5 empirical models to estimate the distance from a measured RSSI value were built with the data of each run. The following table shows the maximum and relative error for each experiment run. The relative error was obtained by calculating the average of the differences between actual and estimated distance divided by the actual distance.

$$e_r(d_{\text{actual}}) = \frac{d_{\text{estimated}} - d_{\text{actual}}}{d_{\text{actual}}} \quad (9)$$

The maximum error is simply the maximum value of the absolute distance differences.

$$e_{\max} = \max(\{|d_{\text{estimated}} - d_{\text{actual}}| \mid \forall d_{\text{estimated}}, d_{\text{actual}}\}) \quad (10)$$

ID	Range	n	P <sub>0</sub>	$\mu(e_r)$	$\sigma(e_r)$	e <sub>max</sub>
01_2	5 m – 50 m	2.0873	-15.8050	0.0183	$\pm 0.1967$	+11.0087 m
02_1	5 m – 50 m	2.1331	-12.8029	0.0052	$\pm 0.1051$	+6.5532 m
02_2	5 m – 50 m	1.7558	-17.6398	0.0323	$\pm 0.2710$	+19.4399 m
07	10 m – 160 m	2.6444	-10.1465	0.0088	$\pm 0.1454$	+60.4273 m

Table 2: Experiments for distance estimation evaluation

To evaluate the data recorded during the experiments, the Log-Normal path-loss model of each dataset was calculated. This model was previously explained in Section 3.1. To apply this model for distance estimation the values for n and P<sub>0</sub> had to be determined. The RSSI equation of the PLM can be expressed as a linear function depending on the logarithm of the distance.

$$\text{RSSI}(d) = P_0 - 10n \cdot \log_{10}\left(\frac{d}{d_0}\right) \quad (11)$$

$$y = ax + b \quad (12)$$

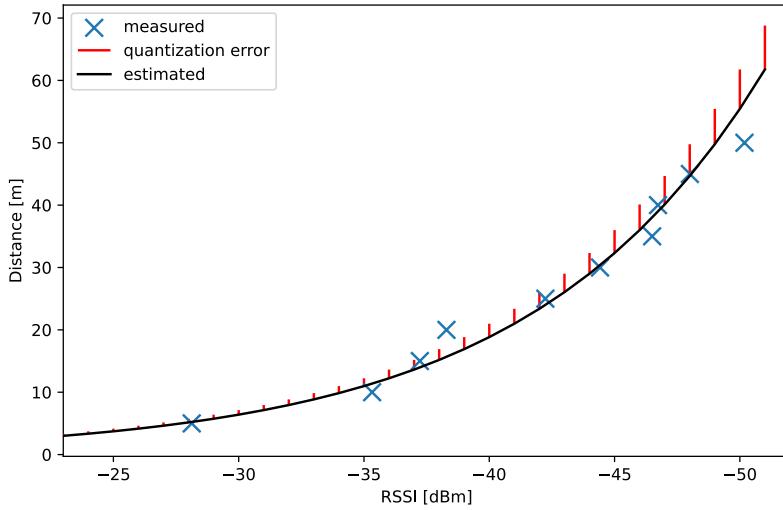
$$y = \text{RSSI}(d), \quad a = -10n, \quad x = \log_{10}\left(\frac{d}{d_0}\right), \quad b = P_0 \quad (13)$$

Because of this expression of the Log-Normal model equation as a linear function, the coefficients could be determined by using linear regression to find a and b. These are directly proportional to the unknown coefficients of the model. With the determined coefficients the Log-Normal model equation could be rewritten to obtain a function from RSSI to the distance d. Assuming that the distance d<sub>0</sub> at which P<sub>0</sub> is the measured power is 1 m following statement is true.

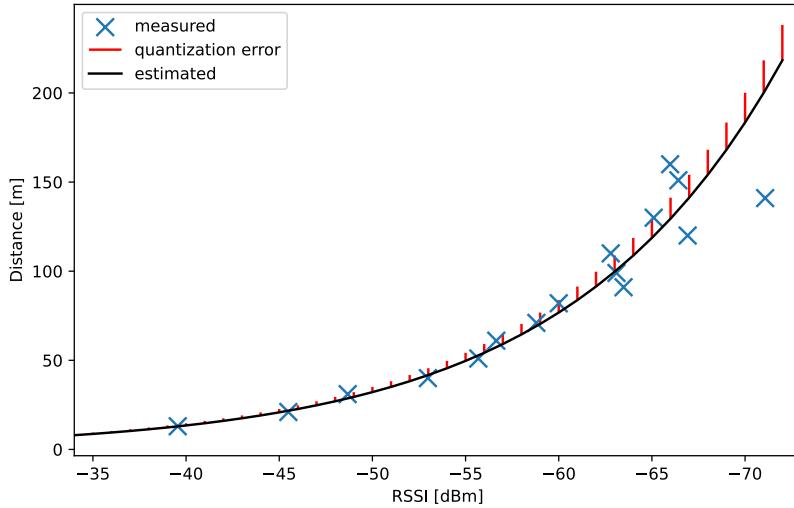
$$d(\text{RSSI}) = 10^{(\text{RSSI} + 10n) \cdot \frac{1}{P_0}} \quad (14)$$

This formula was then used as an distance estimator. Note that although the estimator function would take any real number as RSSI value, the measurement circuit of the LoRa™ transceiver only provides integer RSSI values. This means that the same RSSI value can be measured at different distances. In the following figures the results

from two different experiments is plotted as a function from RSSI to distance. Both the measured and the estimated distance are included in the graph



*Figure 11: RSSI vs. Distance data from experiment 02\_1*



*Figure 12: RSSI vs. Distance data from experiment 07*

This visualization indicates, that the sensitivity of this distance estimation approach decreases the further away the end node and the anchor node are. For example the distance between the RSSI of  $-50$  dBm and  $-51$  dBm are  $6.32$  m. The distance delta in which the same RSSI value is measured can be calculated with the following equation.

$$\Delta d(\text{RSSI}) = d(\text{RSSI} - 1) - d(\text{RSSI}) \quad (15)$$

Despite this quantization effect arising from the hardware capabilities of the used LoRa™ transceiver, the distance estimation function is not modified for further evaluations because the all RSSI values used are the average result of multiple measurements.

### 5.1.2 Results

The results of the distance estimation evaluation may look promising at first glance as the plotted curves of the models fit the measured data quite well. But all models show high error deviation. Also despite using the same experiment setup, the differences of the derived coefficients of experiments 01\_1 through 02\_2 are also big. These observations both imply low repeatability.

At last the data of experiment 07 shows an interesting pattern above 80 m. The origin of this pattern could not be determined as part of this thesis. The pattern can also be found in the data shown in [37, pp. 7-8].

## 5.2 Localization

For the evaluation of the localization implementation the test setup was modified. On a large grass field (Rosental in Leipzig) three anchor nodes were setup on wooden poles in three corners of the field. Their positions were chosen to maximize the area where communication with all three anchors is possible. With the iOS-App “GPS Data Smart” [38] the absolute positions of the three anchors were measured.

After installing all three anchors, the end node was used to perform RSSI measurements at three different positions. Again the actual position of the end node was measured with the previously mentioned iOS-App. The RSSI readings were recorded for later evaluation.

### 5.2.1 Data Analysis

The actual positions of the anchor nodes and the end node were all measured with GPS. This data was in latitude/longitude format. To further process the positions their cartesian representation was needed. This conversion was performed based on the WGS-84 ellipsoid which is used for GPS [39]. After that for better readability all positions were expressed relative to the position of Anchor A.

Anchor ID	GPS Position	Cartesian Position (based on WGS 84)		Cartesian Position (rel. to Anchor A)	
A	latitude: 51°20'54.213"N	x:	1 376 568.54	x:	0.00
	longitude: 12°21'57.332"E	y:	6 649 762.65	y:	0.00
B	latitude: 51°21'0.174"N	x:	1 376 485.20	x:	-83.34
	longitude: 12°21'54.637"E	y:	6 650 057.00	y:	294.35
C	latitude: 51°21'0.72"N	x:	1 376 154.67	x:	-413.86
	longitude: 12°21'43.948"E	y:	6 650 083.97	y:	321.31

Table 3: Anchor positions

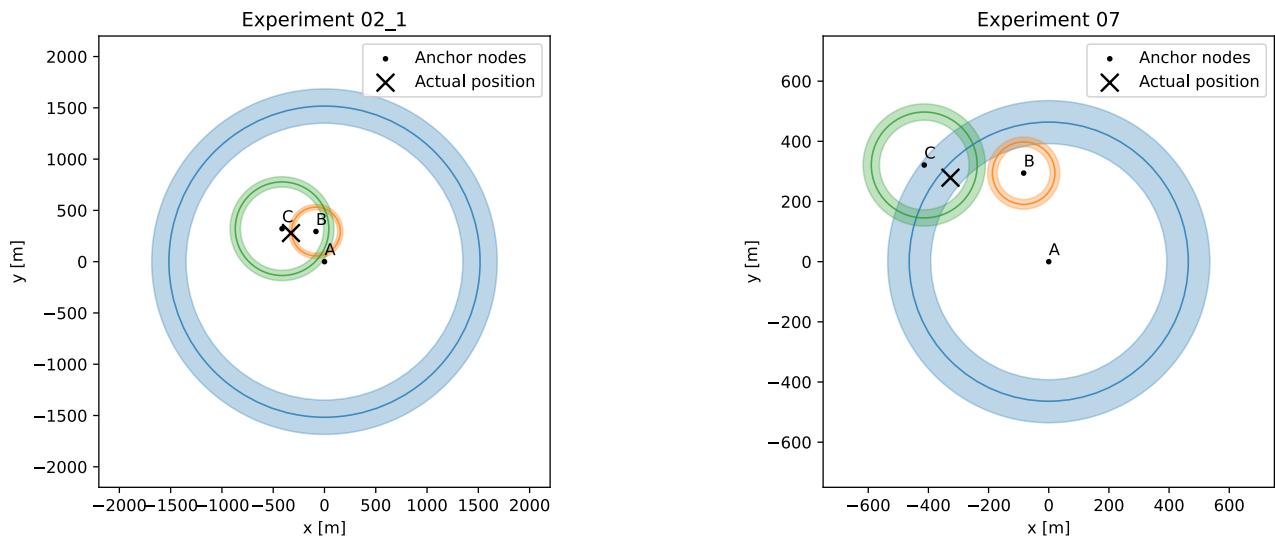
The localization data was evaluated with the models of the two best distance estimations. For that the average RSSI value for every anchor at each position was determined. This was then used with the distance estimator to obtain the distances from the end node to each anchor node. With these distances the trilateration (multilateration with three distances) algorithm was used to estimate the position of the end node. Following table shows the data collected in the localization experiments and the associated estimated positions. Most distance estimations led to not enough circles intersecting which makes a position estimation impossible.

To accommodate for the uncertainty of the distance estimation, the multilateration algorithm was also slightly modified. Instead of calculating the point where all circles intersect it now just calculates the pairwise intersection points. Each pairwise

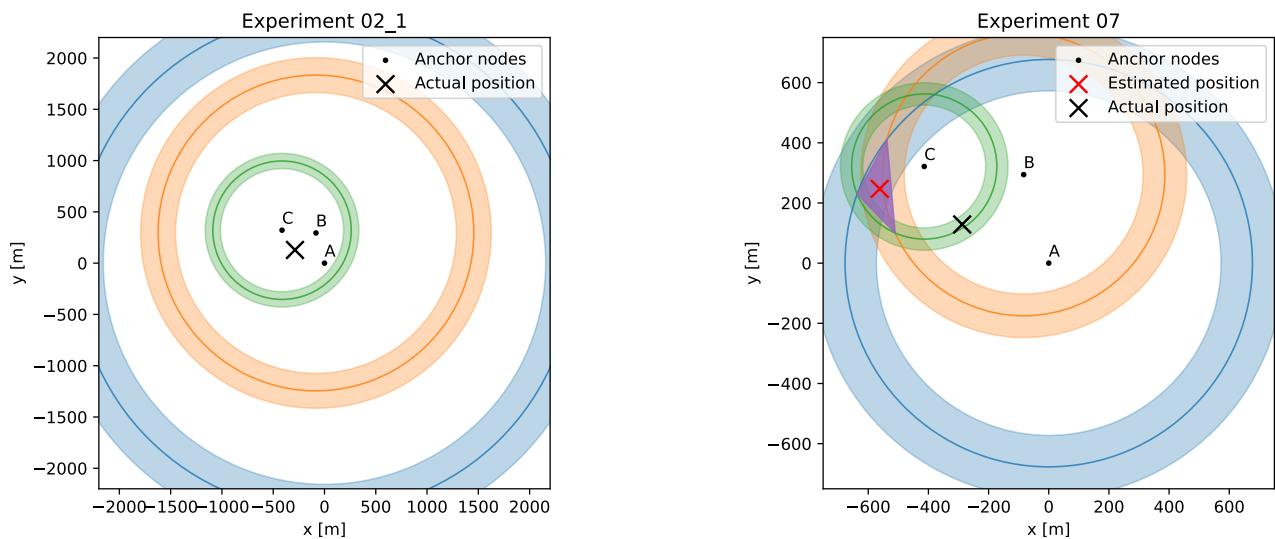
intersection could results in two points, so a strategy to choose one of them must be provided. The first point is chosen by calculating the distance of both points to the third anchor node that was not involved in the intersection. The point with the lower distance is chosen as starting point. After that always the point with lower distance to all already chosen points is included. The result can be seen in Figure 14.

<b>Distance model</b>	<b>Position</b>	<b>Anchor ID</b>	<b>RSSI [dBm]</b>	<b>Distance [m]</b>	<b>Estimated Position</b>	<b>Actual Position</b>
02_1	pos1	A	-80.66	1516.98 ± 167.34	unknown	x: -326.73 y: 278.99
		B	-63.48	237.63 ± 26.21		
		C	-69.53	456.15 ± 50.32		
	pos2	A	-85.00	2424.21 ± 267.42	unknown	x: -287.79 y: 128.70
		B	-80.79	1539.06 ± 169.77		
		C	-73.15	674.67 ± 74.42		
	pos3	A	-80.17	1438.75 ± 158.71	unknown	x: -218.91 y: 194.24
		B	-61.88	199.91 ± 22.05		
		C	-73.73	717.80 ± 79.18		
07	pos1	A	-80.66	463.87 ± 71.55	unknown	x: -326.73 y: 278.99
		B	-63.48	103.99 ± 16.04		
		C	-69.53	175.97 ± 27.14		
	pos2	A	-85.00	677.06 ± 104.43	x: -561.83 y: 246.99	x: -287.79 y: 128.70
		B	-80.79	469.31 ± 72.39		
		C	-73.15	241.30 ± 37.22		
	pos3	A	-80.17	444.48 ± 68.56	unknown	x: -218.91 y: 194.24
		B	-61.88	90.45 ± 13.95		
		C	-73.73	253.66 ± 39.12		

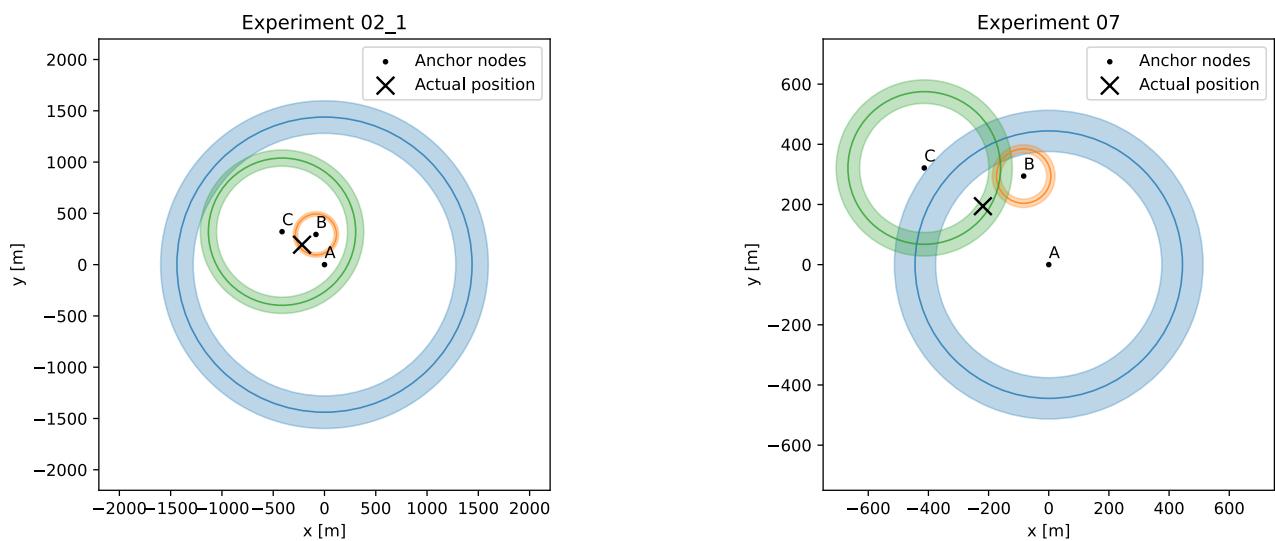
Table 4: Experiments for localization evaluation



**Figure 13: Localization of position 1**



**Figure 14: Localization of position 2**



**Figure 15: Localization of position 3**

### 5.2.2 Results

The evaluation of the localization data shows that the distance estimator is too inaccurate to be useable for position estimation. As concluded in Section 5.1.2, the distances estimated for RSSI values at high distance deviate significantly from the actual distances. This likely originates in the fluctuations of the RSSI signal at high distances and therefore lower fit of the model for these distances. Additional to this conclusion another problem in the repeatability between different devices can be observed. Instead of showing similar deviation from the actual position, the accuracy of the distance estimation varies largely between different anchor nodes. For example in Figure 15 experiment 07 the actual position of the end node is chosen so that every anchor node has similar distance from the end node. This leads to the assumption that all estimated distance show similar error characteristics. But while the distance estimated from Anchor C is very accurate, both Anchor A and B show large deviation from the actual distance.

All these observations lead to the conclusion that the localization system cannot provide reliable positioning data despite being able to estimate a position in Figure 14 experiment 07.

## 5.3 Low Power

To estimate the power consumption of the implemented localization system. The current drawn at a fixed voltage (5 V) of one node was measured in different operating modes. This measurement was conducted by connecting a multimeter in series with lab bench power supply and the device. The power supply was configured to provide a fixed voltage of 5V. Special firmware with hardcoded operating modes was used to eliminate as much other influences as possible.

Mode	Current [mA]	Power consumption [mW]
Transmitting	27	135
Receiving	8.5	42.5
Sleep	2.9	14.5

*Table 5: Power consumption*

It can be seen that the power consumption of one node is quite low. If we assume that the node transmits two packets per second, listens for incoming packets for 400 ms and sleeps for the rest of the time, following average power consumption can be calculated.  $T_{tx}$  is the time one packet needs to be transmitted. For all packets in the implemented system this time is approximately 40 ms.

$$P_{\text{total}} = P_{\text{tx}} + P_{\text{rx}} + P_{\text{sleep}} \quad (16)$$

$$P_{\text{tx}} = \frac{2 \cdot T_{\text{tx}}}{1000 \text{ ms}} \cdot 135 \text{ mW} \quad (17)$$

$$P_{\text{rx}} = \frac{400 \text{ ms}}{1000 \text{ ms}} \cdot 42.5 \text{ mW} \quad (18)$$

$$P_{\text{tx}} = \frac{1000 - (2 \cdot T_{\text{tx}} + 400 \text{ ms})}{1000 \text{ ms}} \cdot 14.5 \text{ mW} \quad (19)$$

$$P_{\text{total}} = 30.7 \text{ mW} \quad (20)$$

With a battery capacity of 40000 mWh (LiPo battery with 10000 mAh) the lifetime of the system would be 1303 h or 54.3 days.

## 6 FUTURE WORKS

As stated in the goal of this thesis a localization system based on RSSI measurements for LoRa™ was implemented and evaluated. The implementation presented the design decision necessary for building a localization system. These included for example the importance of communicating the reception of a packet sent by an anchor node and the benefits of triggering the localization by a packet sent by an end node.

The evaluation of this localization system showed that the possibility to fit models for distance estimation based on RSSI exists but refinement has to be done to use the distance estimation models for position estimation.

These refinements could not be explored as part of this work but some clues from the evaluation allow to draw some assumptions where those refinements would likely be most impactful.

The first refinement that should be considered is the repeatability of the distance estimation. This could possibly be improved by increasing the distance range measured. For this the measuring process could also be improved to increase the amount of gathered data. This could for example be achieved by replacing the absolute position measurement using the mobile phone, with a GPS receiver integrated in the testing devices. This receiver could be sampled in addition to the RSSI measurements.

Following work should also consider increasing the sample size of the measuring devices. Due to manufacturing limitations minor variations in transmitting and receiving circuits exists. Maybe the effect of these variations on the estimated distance could be observed and eliminated.

When refining the RSSI distance estimation, one could also analyze the variations shown at higher distances. Finding the cause and decreasing or even eliminating it, would possibly result in more accurate distance estimations and consequently better position estimation.

Another possible optimization lies in the chosen path-loss model. This thesis only employed a single-slope model, but in the literature models with two or more slopes exists [22, p. 20]. This possibly could improve the accuracy of the distance estimation by employing different curves at different distances.

Regarding the localization algorithm, other methods could be explored. Possibilities include the weighted LMS approach by [20] which counteracts the error at high distances by introducing weights.

## LIST OF FIGURES

Figure 1: Triangulation of end node with two anchor nodes .....	6
Figure 2: Triangulation error at different distances .....	7
Figure 3: Current consumption in Active and Sleep mode .....	9
Figure 4: Position estimation with multilateration algorithm © 2017 IEEE .....	10
<i>Reprinted, with permission, from Bernat Carbonés Fargas, GPS-free geolocation using LoRa in low-power WANs, 2017 Global Internet of Things Summit (GIoTS), June 2017</i>	
Figure 5: Position estimation error with multilateration .....	13
Figure 6: LoraLocator system overview .....	14
Figure 7: NUCLEO-WL55JC .....	15
Figure 8: Firmware architecture .....	15
Figure 9: `Ping_t` transmission .....	20
Figure 10: Distance estimation setup at multiple distances .....	23
Figure 11: RSSI vs. Distance data from experiment 02_1 .....	25
Figure 12: RSSI vs. Distance data from experiment 07 .....	25
Figure 13: Localization of position 1 .....	28
Figure 14: Localization of position 2 .....	28
Figure 15: Localization of position 3 .....	28

## LIST OF TABLES

Table 1: LoRa transceiver configuration .....	23
Table 2: Experiments for distance estimation evaluation .....	24
Table 3: Anchor positions .....	26
Table 4: Experiments for localization evaluation .....	27
Table 5: Power consumption .....	29

## LIST OF LISTINGS

Listing 1: Device type and ID configuration .....	16
Listing 2: Device type specific code compilation .....	16
Listing 3: Track last performed action .....	16
Listing 4: Execute logic depending on last performed action .....	17
Listing 5: Pass function pointer to radio driver for event handling .....	17
Listing 6: List of event handlers used in LoraLocator .....	18
Listing 7: Example state change in transmission-done event handler .....	18
Listing 8: Switch LoRa transceiver to transmission-/reception-mode .....	19
Listing 9: Sequencer task registration .....	19
Listing 10: Helper function to schedule task execution .....	19
Listing 11: Timer creation .....	20
Listing 12: Start timer with specified timeout value .....	20
Listing 13: Packet type `Ping_t` .....	20
Listing 14: Packet type `AnchorResponse_t` .....	21
Listing 15: Packet type `Ack_t` .....	21

## BIBLIOGRAPHY

- [1] “AirTag.” Accessed: Oct. 22, 2024. [Online]. Available: <https://www.apple.com/de/airtag/>
- [2] M. Rizzi, P. Ferrari, A. Flammini, E. Sisinni, and M. Gidlund, “Using LoRa for industrial wireless networks,” in *2017 IEEE 13th International Workshop on Factory Communication Systems (WFCS)*, May 2017, pp. 1-4. doi: [10.1109/WFCS.2017.7991972](https://doi.org/10.1109/WFCS.2017.7991972).
- [3] I. F. Priyanta, F. Golatowski, T. Schulz, and D. Timmermann, “Evaluation of LoRa Technology for Vehicle and Asset Tracking in Smart Harbors,” in *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*, Lisbon, Portugal: IEEE, Oct. 2019, pp. 4221–4228. doi: [10.1109/IECON.2019.8927566](https://doi.org/10.1109/IECON.2019.8927566).
- [4] P. Gotthard and T. Jankech, “Low-Cost Car Park Localization Using RSSI in Supervised LoRa Mesh Networks,” in *2018 15th Workshop on Positioning, Navigation and Communications (WPNC)*, Oct. 2018, pp. 1-6. doi: [10.1109/WPNC.2018.8555792](https://doi.org/10.1109/WPNC.2018.8555792).
- [5] H. B. M. Alves *et al.*, “Introducing a survey methodology for assessing LoRaWAN coverage in Smart Campus scenarios,” in *2020 IEEE International Workshop on Metrology for Industry 4.0 & IoT*, Jun. 2020, pp. 708–712. doi: [10.1109/MetroInd4.0IoT48571.2020.9138300](https://doi.org/10.1109/MetroInd4.0IoT48571.2020.9138300).
- [6] B. Gardiner, W. Ahmad, T. Cooper, M. Haveard, J. Holt, and S. Biaz, “Collision Avoidance Techniques for Unmanned Aerial Vehicles Technical Report # CSSE 1101,” 2011. Accessed: Sep. 30, 2024. [Online]. Available: <https://www.semanticscholar.org/paper/Collision-Avoidance-Techniques-for-Unmanned-Aerial-Gardiner/49d4db3e71beb1795838c0e06fa4a0335a3608d5>
- [7] L. Sciullo, A. Trotta, and M. D. Felice, “Design and performance evaluation of a LoRa-based mobile emergency management system (LOCATE),” *Ad Hoc Networks*, vol. 96, p. 101993–101994, Jan. 2020, doi: [10.1016/j.adhoc.2019.101993](https://doi.org/10.1016/j.adhoc.2019.101993).
- [8] A. Mackey and P. Spachos, “LoRa-based Localization System for Emergency Services in GPS-less Environments,” in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, Paris, France: IEEE, Apr. 2019, pp. 939–944. doi: [10.1109/INFCOMW.2019.8845189](https://doi.org/10.1109/INFCOMW.2019.8845189).
- [9] C. D. Fernandes *et al.*, “Hybrid indoor and outdoor localization for elderly care applications with LoRaWAN,” in *2020 IEEE International Symposium on Medical Measurements and Applications (MeMeA)*, Jun. 2020, pp. 1-6. doi: [10.1109/MeMeA49120.2020.9137286](https://doi.org/10.1109/MeMeA49120.2020.9137286).
- [10] L. Vangelista, A. Zanella, and M. Zorzi, “Long-Range IoT Technologies: The Dawn of LoRa™,” in *Future Access Enablers for Ubiquitous and Intelligent Infrastructures*, V. Atanasovski and A. Leon-Garcia, Eds., Cham: Springer International Publishing, 2015, pp. 51–58. doi: [10.1007/978-3-319-27072-2\\_7](https://doi.org/10.1007/978-3-319-27072-2_7).
- [11] L. E. Marquez and M. Calle, “Understanding LoRa-Based Localization: Foundations and Challenges,” *IEEE Internet of Things Journal*, vol. 10, no. 13, pp. 11185–11198, Jul. 2023, doi: [10.1109/JIOT.2023.3248860](https://doi.org/10.1109/JIOT.2023.3248860).
- [12] R. Peng and M. L. Sichitiu, “Angle of Arrival Localization for Wireless Sensor Networks,” in *2006 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks*, Sep. 2006, pp. 374–382. doi: [10.1109/SAHCN.2006.288442](https://doi.org/10.1109/SAHCN.2006.288442).

- [13] M. Zare, R. Battulwar, J. Seamons, and J. Sattarvand, “Applications of Wireless Indoor Positioning Systems and Technologies in Underground Mining: a Review,” *Mining, Metallurgy & Exploration*, vol. 38, no. 6, pp. 2307–2322, Dec. 2021, doi: [10.1007/s42461-021-00476-x](https://doi.org/10.1007/s42461-021-00476-x).
- [14] J. Kuriakose, S. Joshi, and V. George, “Localization in Wireless Sensor Networks: A Survey,” Oct. 2014.
- [15] B. C. Fargas and M. N. Petersen, “GPS-free geolocation using LoRa in low-power WANs,” in *2017 Global Internet of Things Summit (GIoTS)*, Jun. 2017, pp. 1–6. doi: [10.1109/GIOTS.2017.8016251](https://doi.org/10.1109/GIOTS.2017.8016251).
- [16] J. A. Azevedo and F. Mendonça, “A Critical Review of the Propagation Models Employed in LoRa Systems,” *Sensors*, vol. 24, no. 12, p. 3877–3878, Jan. 2024, doi: [10.3390/s24123877](https://doi.org/10.3390/s24123877).
- [17] G. M. Bianco, R. Giuliano, G. Marrocco, F. Mazzenga, and A. Mejia-Aguilar, “LoRa System for Search and Rescue: Path-Loss Models and Procedures in Mountain Scenarios,” *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1985–1999, Feb. 2021, doi: [10.1109/JIOT.2020.3017044](https://doi.org/10.1109/JIOT.2020.3017044).
- [18] A. I. Griva *et al.*, “LoRa-Based IoT Network Assessment in Rural and Urban Scenarios,” *Sensors*, vol. 23, no. 3, p. 1695–1696, Jan. 2023, doi: [10.3390/s23031695](https://doi.org/10.3390/s23031695).
- [19] M. Stusek *et al.*, “Accuracy Assessment and Cross-Validation of LPWAN Propagation Models in Urban Scenarios,” *IEEE Access*, vol. 8, pp. 154625–154636, 2020, doi: [10.1109/ACCESS.2020.3016042](https://doi.org/10.1109/ACCESS.2020.3016042).
- [20] N. El Agroudy, N. Joram, and F. Ellinger, “Low power RSSI outdoor localization system,” in *2016 12th Conference on Ph.D. Research in Microelectronics and Electronics (PRIME)*, Jun. 2016, pp. 1–4. doi: [10.1109/PRIME.2016.7519456](https://doi.org/10.1109/PRIME.2016.7519456).
- [21] O. Dieng, C. Pham, and O. Thiare, “Outdoor Localization and Distance Estimation Based on Dynamic RSSI Measurements in LoRa Networks: Application to Cattle Rustling Prevention,” in *2019 International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, Barcelona, Spain: IEEE, Oct. 2019, pp. 1–6. doi: [10.1109/WiMOB.2019.8923542](https://doi.org/10.1109/WiMOB.2019.8923542).
- [22] A. Zanella, “Best Practice in RSS Measurements and Ranging,” *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2662–2686, 2016, doi: [10.1109/COMST.2016.2553452](https://doi.org/10.1109/COMST.2016.2553452).
- [23] A. Norrdine, “An Algebraic Solution to the Multilateration Problem,” 2012. doi: [10.13140/RG.2.1.1681.3602](https://doi.org/10.13140/RG.2.1.1681.3602).
- [24] “NUCLEO-WL55JC - STM32 Nucleo-64 development board with STM32WL55JC MCU, SMPS, supports Arduino and morpho connectivity - STMicroelectronics.” Accessed: Oct. 14, 2024. [Online]. Available: <https://www.st.com/en/evaluation-tools/nucleo-wl55jc.html>
- [25] “STM32WL55JC - Sub-GHz Wireless Microcontrollers. Dual-core Arm Cortex-M4/M0+ @48 MHz with 256 Kbytes of Flash memory, 64 Kbytes of SRAM. LoRa, (G)FSK, (G)MSK, BPSK modulations. AES 256-bit. Multiprotocol System-on-Chip. - STMicroelectronics.” Accessed: Oct. 14, 2024. [Online]. Available: <https://www.st.com/en/microcontrollers-microprocessors/stm32wl55jc.html>
- [26] “STMicroelectronics: Our technology starts with you.” Accessed: Oct. 14, 2024. [Online]. Available: [https://www.st.com/content/st\\_com/en.html](https://www.st.com/content/st_com/en.html)

- [27] “STM32WL55xx STM32WL54xx - Multiprotocol LPWAN dual core 32-bit Arm® Cortex® -M4/M0+ LoRa®, (G)FSK, (G)MSK, BPSK, up to 256KB flash, 64KB SRAM.” Accessed: Oct. 14, 2024. [Online]. Available: <https://www.st.com/resource/en/datasheet/stm32wl55jc.pdf>
- [28] “AN5406: How to build a LoRa® application with STM32CubeWL.” Accessed: Oct. 14, 2024. [Online]. Available: [https://www.st.com/resource/en/application\\_note/an5406-how-to-build-a-lora-application-with-stm32cubewl-stmicroelectronics.pdf](https://www.st.com/resource/en/application_note/an5406-how-to-build-a-lora-application-with-stm32cubewl-stmicroelectronics.pdf)
- [29] M. Schmiedel, “moseschmiedel/lora-locator.” Accessed: Oct. 14, 2024. [Online]. Available: <https://github.com/moseschmiedel/lora-locator/tree/submission>
- [30] “UM2642: Description of STM32WL HAL and low-layer drivers.” Accessed: Oct. 15, 2024. [Online]. Available: [https://www.st.com/content/ccc/resource/technical/document/user\\_manual/group1/6f/be/85/55/8c/26/4c/22/DM00660673/files/DM00660673.pdf/jcr:content/translations/en.DM00660673.pdf](https://www.st.com/content/ccc/resource/technical/document/user_manual/group1/6f/be/85/55/8c/26/4c/22/DM00660673/files/DM00660673.pdf/jcr:content/translations/en.DM00660673.pdf)
- [31] “Utility:Sequencer - stm32mcu.” Accessed: Oct. 15, 2024. [Online]. Available: <https://wiki.st.com/stm32mcu/wiki/Utility:Sequencer>
- [32] “lora-locator/Utilities/timer/stm32\_timer.c at submission · moseschmiedel/lora-locator.” Accessed: Oct. 22, 2024. [Online]. Available: [https://github.com/moseschmiedel/lora-locator/blob/a515014227e14d8f465ef4837db8f4efc585e634/Utilities/timer/stm32\\_timer.c](https://github.com/moseschmiedel/lora-locator/blob/a515014227e14d8f465ef4837db8f4efc585e634/Utilities/timer/stm32_timer.c)
- [33] R. Faragher and R. Harle, “Location Fingerprinting With Bluetooth Low Energy Beacons,” *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 11, pp. 2418–2428, Nov. 2015, doi: [10.1109/JSAC.2015.2430281](https://doi.org/10.1109/JSAC.2015.2430281).
- [34] Y. Qamaz, A. Schowering, and J. Bistrón, “Experimental evaluation of using BLE beacon for outdoor positioning in GPS-denied environment,” *AGILE: GIScience Series*, vol. 3, pp. 1–9, Jun. 2022, doi: [10.5194/agile-giss-3-13-2022](https://doi.org/10.5194/agile-giss-3-13-2022).
- [35] “GPS Tracks.” Accessed: Oct. 22, 2024. [Online]. Available: <https://apps.apple.com/us/app/gps-tracks/id425589565>
- [36] M. Schmiedel, “moseschmiedel/bachelor-thesis.” Accessed: Oct. 17, 2024. [Online]. Available: <https://github.com/moseschmiedel/bachelor-thesis>
- [37] H. Kwasme and S. Ekin, “RSSI-Based Localization Using LoRaWAN Technology,” *IEEE Access*, vol. 7, pp. 99856–99866, 2019, doi: [10.1109/ACCESS.2019.2929212](https://doi.org/10.1109/ACCESS.2019.2929212).
- [38] “GPS Data Smart.” Accessed: Oct. 21, 2024. [Online]. Available: <https://apps.apple.com/us/app/gps-data-smart/id1076838348>
- [39] GISGeography, “World Geodetic System (WGS84).” Accessed: Oct. 21, 2024. [Online]. Available: <https://gisgeography.com/wgs84-world-geodetic-system/>

## 7 APPENDIX

All software and data implemented and recorded for this thesis can be found online or on the provided USB drive.

The data and the scripts for evaluation can be found in this repository on GitHub: <https://github.com/moseschmiedel/bachelor-thesis/tree/submission>.

The firmware for the anchor nodes and the end node can be found in this repository on GitHub: <https://github.com/moseschmiedel/lora-locator/tree/submission>.

The contents of the USB drive are:

data	folder with the data and data evaluation repository
lora-locator	folder with the anchor and end node firmware repository
thesis	folder with the source code and PDF of this thesis