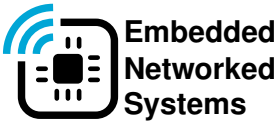
Delft University of Technology
Master's Thesis in Embedded Systems

Wireless Clock Synchronisation for UWB Positioning

Jeroen Overman



TU Delft Delft University of Technology

Wireless Clock Synchronisation for UWB Positioning

Master's Thesis in Embedded Systems

The Embedded and Networked Systems Section
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Mekelweg 4, 2628 CD Delft, The Netherlands

Jeroen Overman
j.overman@student.tudelft.nl

25th August 2019

Author
Jeroen Overman (j.overman@student.tudelft.nl)
Title
Wireless Clock Synchronisation for UWB Positioning
MSc presentation
27th August 2019

Graduation Committee

Prof. dr. K.G. Langendoen	Delft University of Technology
Prof. dr. ir. G.J.T. Leus	Delft University of Technology
ir. D.O. van den Heuvel	TOPIC Embedded Systems

Abstract

An Indoor Positioning System (IPS) is being developed at TOPIC Embedded Systems to track equipment in hospitals. The system should prevent the loss of equipment en make procedures more efficient. The IPS will consist of anchors and tags. Anchors are the radios that form an infrastructure in the building to localise tags that are placed on equipment. Different localisation techniques and methods exist for indoor localisation, of which Ultra-wideband (UWB) is a very promising technology as it is robust to multipath interference. Also the Time Difference of Arrival (TDoA) method has advantages over Two-way Ranging (TWR) in terms of energy consumption of a tag and the rate of supported location measurements.

A fundamental requirement for TDoA-based localisation is that all anchors must be precisely synchronised as radio signals propagate trough air with the speed of light. Existing synchronisation solutions synchronise the anchors either trough wires or wirelessly. To keep the installation costs of the IPS to a minimum synchronisation should work without adding extra infrastructure to a building such as a clock distribution network. Therefore a wireless solution is required. The existing solutions have been evaluated on hardware equipped with precise clock sources that have tight tolerances. These clock sources are not available on the commodity hardware (Decawave DWM1001) that is intended to be used by TOPIC. Also the existing synchronisation algorithms are not designed for large multi-hop networks.

A new synchronisation algorithm based on a 3-state Kalman filter is developed and evaluated with the existing solutions showing that linear interpolation performs the best in terms of the Mean Absolute Error (MAE),Ḣowever the linear-interpolation algorithm comes at the cost of a latency as the timestamps become only available after the next synchronisation message. If the latency (order of seconds) cannot be tolerated, the developed 3-state Kalman filter is the best alternative.

As TOPIC requires a latency of 5 min the linear-interpolation algorithm is integrated in a synchronisation scheme for multi-hop networks. This scheme has the additional ability to measure the propagation delay. The linear-interpolation algorithm is evaluated with practical experiments for single-hop and multi-hop synchronisation. The MAE of the synchronised clock is 229 ps for single hop and 258 ps for multi hop when using a synchronisation period of 1 s. This shows that that the synchronisation algorithm is very suitable for multi-hop networks.

To determine the effect of the clock synchronisation on position accuracy an experiment has been conducted where anchors measure the TDoAs of a message sent by a tag. A standard multilateration algorithm [1] was used to estimate the locations based on these measurements. The accuracy of an

individual estimate was low, but by averaging subsequent measurements a mean error of 51 cm was achieved, meeting the requirement of 1 m accuracy.

Contents

Chapter 1

Introduction

Indoor localisation has many applications such as tracking items, providing assistance for elderly and disabled people and indoor navigation [2], [3]. It can also be used in hospitals for tracking patients and (expensive) equipment. This work has been done on behalf of the company TOPIC Embedded Systems [4]. TOPIC wants to develop an Indoor Positioning System (IPS) that is applicable in hospitals, where the system will track medical equipment. This is important for several reasons.

First of all, hospitals are continuously growing making it easier for equipment to get lost. Being able to localise the equipment will save crucial time when a piece of equipment is urgently needed for a patient. Furthermore, it will prevent the loss of equipment.

A second important application of the localisation system in a hospital is an inventory check of an operating room. Before surgery can start, the inventory of an operating room has to be checked to ensure every piece of equipment is present. This is called the TIMEOUT procedure [5]. This is a time consuming task for the hospital staff. A localisation system could (partly) automate this task, making the operation more efficient.

In order to localise equipment using Ultra-wideband (UWB), the objects must carry a battery-equipped UWB radio, limiting its size to approximately a wrist watch and its energy consumption. The radio node attached to the equipment is called a *tag*. A tag can be localised by other radio nodes that have a fixed location, called *anchors*.

For the applications described above estimating the location of the objects in 2D is sufficient. However the anchors and the tags are not necessarily at the same height.

Anchors will typically be located at the ceiling and tags will be attached to equipment on the floor. Therefore the system must be considered as a 3D IPS, where the accuracy of the height is not considered important.

When tracking assets the Global Positioning System (GPS) is a commonly used technology, but it is not suitable for indoor environments since the weak

radio signals from the satellites cannot penetrate solid walls and obstacles. In the past years indoor positioning has been an emerging research area [6], [7]. Many different technologies have been investigated for IPSs such as Wi-Fi [8], Bluetooth [9], Infrared [10], Ultrasonic [11], Zigbee [12], RFID [13] and Ultra-wideband (UWB) [14]–[20]. The underlying measurement method that these technologies use can be divided into two main groups: measuring the Received Signal Strength (RSS) and measuring the Time of Flight (ToF).

Technologies using RSS to determine a position suffer from multipath interference, signal attenuation when passing through objects and are subject to changes in the environment like opening a door, moving an object in a room [21]–[23]. The advantage of using RSS as a measurement method is that it can be applied with technologies that are already present in many buildings such as WIFI and Bluetooth. However the accuracy of IPSs using RSS measurements is much lower than the accuracy of IPSs that measure the ToF, due to the multipath interference and signal attenuation.

UWB is a technology that uses ToF as a measurement method. It is able to timestamp messages with high precision, making accurate positioning possible [24], [25]. Furthermore UWB is more resilient than other radio technologies to the effects of multipath interference, which occurs a lot indoors. This makes UWB an interesting technology for indoor localisation.

1.1 Requirements and constraints

TOPIC wants to develop an IPS with UWB. From the applications described in the introduction the following requirements can be derived:

R1. The 2D localisation accuracy must be within 1 m
In order to determine if a piece of equipment is inside a operating room, the accuracy must be within 1 m. Providing that the equipment is not to close to an adjacent room.

R2. The localisation system must support 1000+ tags
Hospitals are continuously growing, having thousands of pieces of equipment as a result.

R3. The location of a tag must be updated every 5 min
In order to successfully localise equipment, the location must be frequently updated. Since equipment will not be moved very frequently inside a hospital an update rate of 5 min will be sufficient.

R4. The battery life of a tag must be longer than 12 months
The servicing period for most medical equipment is twelve months.[1] The battery of tag can be replaced when the equipment is serviced.

[1] In an interview with the biomedical engineering department of a local hospital, the maintenance interval of medical equipment was discussed. From a report of their mainte-

R5. The tag must be small
The tag is mounted on equipment that needs to be tracked, therefore the tag must be as small as possible. It must not be larger than an UWB transceiver, antenna and a coin-cell battery. Resulting in the size of approximately a wrist watch.

The requirements should be met within the following constraints:

C1. The system does not require additional infrastructure
Installing additional infrastructure in a building is very expensive. Therefore the system can only use existing infrastructures (besides tags and anchors) such as power outlets and WIFI.

C2. The system does not interfere with other wireless systems
Most hospitals already have existing wireless systems such as WIFI. The localisation system should not interfere with these existing systems. Since UWB sends with a signal strength below the noise floor of Wi-Fi it will not interfere.

1.2 Analysis of localisation methods

Several localisation methods can be used with UWB. Much research has been performed on different methods and on how to improve their accuracy [6]. For the applications explained in the introduction the most suitable localisation method needs to be found considering the requirements stated in Section 1.1.

As stated earlier in this chapter UWB uses ToF instead of RSS to determine a position. Different methods exist to determine a position based on ToF measurements. These methods can be grouped by two principles: trilateration and multilateration. These will be described briefly below.

1.2.1 Trilateration

Trilateration is the process of determining a position based on distances between a tag and multiple anchors. When a circle is drawn around each anchor with a radius equal to the distance between that anchor and the tag, the tag will be located at the intersection of all circles. This is shown in Figure 1.1. For finding a tag's location in 3D, the distances to at least three anchors need to be available.

In order to find the distance between a tag and an anchor Two-way Ranging (TWR) can be used. This process is described below.

nance schedule it could be determined that 80 % of the medical equipment was serviced with an interval of twelve months or less.

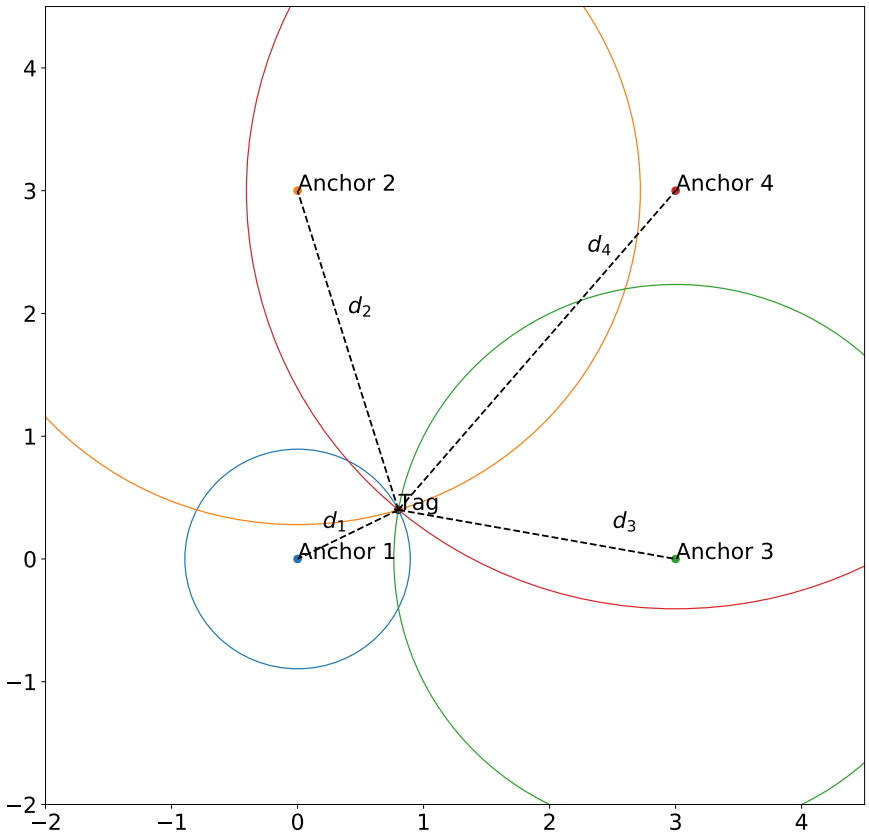


Figure 1.1: *Trilateration*

Two-way Ranging (TWR)

TWR is a method to calculate the distance between a tag and an anchor by determining the ToF of the messages traveling between them. First an anchor sends a message to the tag and records its time of transmission $t1$. The tag receives the message and transmits a response back after a delay t_{reply}. The anchor receives this response and records the time of reception $t2$. This process is shown in Figure 1.2.

The distance can now be calculated as follows:

$$\text{distance} = c \times \frac{t_2 - t_1 - t_{reply}}{2}$$

Where c is the propagation speed of the messages (i.e. speed of light).

Ranging needs to be obtained by multiple anchors, so it needs to be coordinated which anchors perform TWR with a tag. This coordination adds extra complexity to the system. The number of messages when performing TWR with four anchors is in the order of $\mathcal{O}(4 \times 2 \times n) = \mathcal{O}(8n)$ where n is the number of tags.

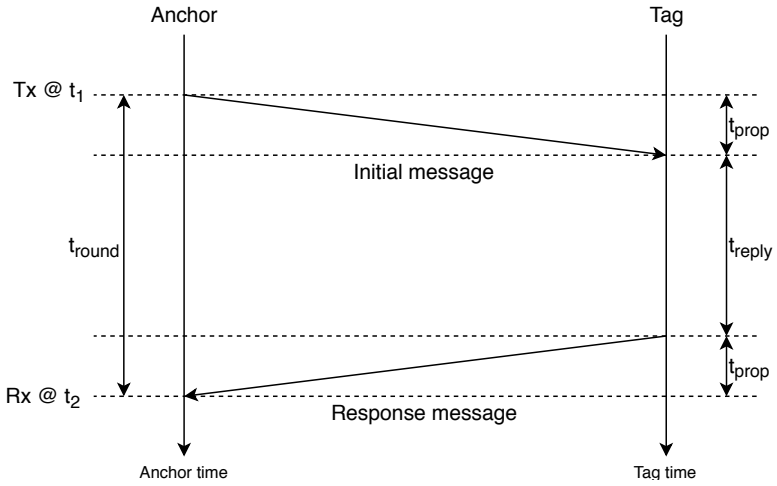


Figure 1.2: *Two-way ranging*

1.2.2 Multilateration

Multilateration is the process of determining a tag's position based on multiple Time Of Arrivals (TOAs) of a message sent by a tag measured at the anchors. The anchors can be grouped in pairs. By taking the time difference from the TOAs of the pair (A and B), the Time Difference of Arrival (TDoA) is determined. With the TDoA it can be determined how much closer the tag was to *anchor A* than it was to *anchor B*. Based on this range difference and knowing the exact location of the anchors a hyperbola can be constructed.

When creating multiple hyperbolas from different anchor pairs, the location of the tag can be determined by finding the intersection of all hyperbolas. This is shown in Figure 1.3. To find a tag's location in 3D, the differences in distance between at least four anchor pairs need to be available.

Time Difference of Arrival (TDoA)

In the process of determining the TDoA, a tag sends a message that is received by all anchors in range. All anchors measure the Time Of Arrival (TOA) of that message. Pairs of anchors can be created with the anchor that received the message first and all other anchors. The distance difference between anchor i and anchor 1 can be expressed as:

$$\text{distance difference} = c \times (t_i - t_1)$$

where c is the propagation speed of the message.

In order for the formula above to work the clocks of the anchors need to be synchronised. Otherwise the reception times cannot be subtracted form

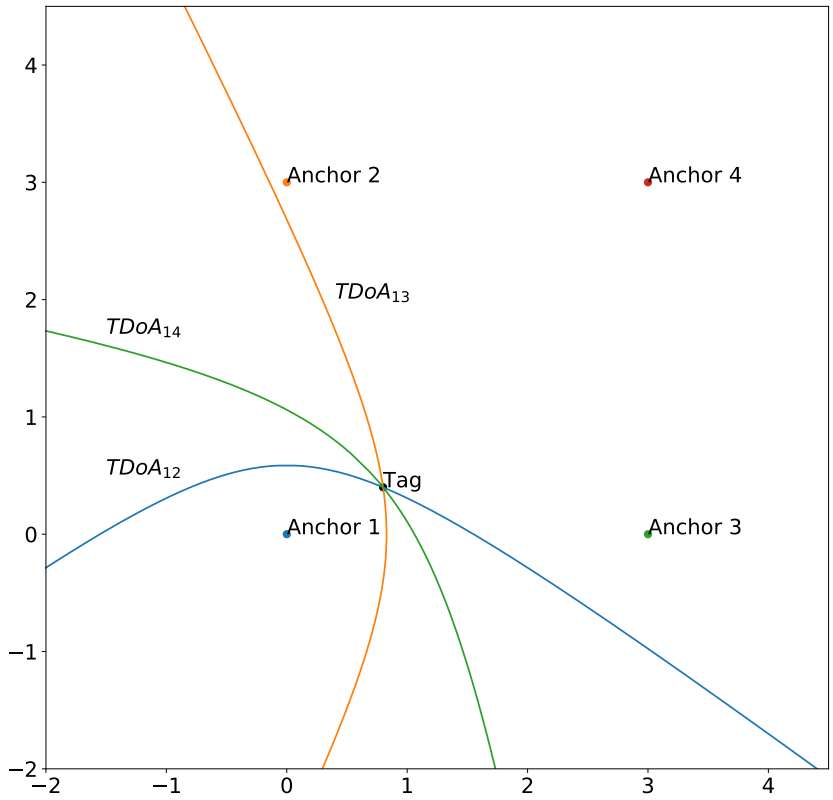


Figure 1.3: *Multilateration*

each other to retrieve the TDoA between the anchor pair. TDoA will be explained in more detail in Section 2.1.

In contrast to TWR this method does not require bidirectional communication and coordination of which anchors should be used for ranging. One message from the tag will allow the anchors to estimate the location of the tag. The number of messages when using TDoA as a localisation method is of the order $\mathcal{O}(n)$ where n is the number of tags.

1.2.3 Deciding on the localisation method

The two methods described above could both be used to create an IPS. In order to compare them a mathematical framework was created that considers both the physical layer of the system and the Medium Access Control (MAC) layer of the system. With the results of this framework an estimation of the number of supported tags and the energy consumption of a tag can be determined for the different localisation methods. This framework is described in detail in Appendix A. The results of this framework help determine if the methods can be used to meet requirements R2, R3, R4 and R5.

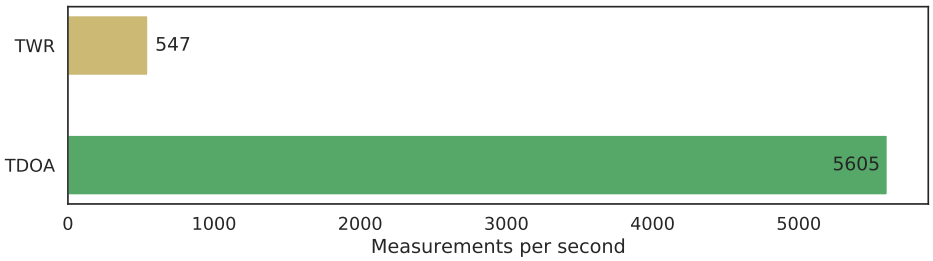


Figure 1.4: *The maximum number of location measurements an anchor can take in a second.*

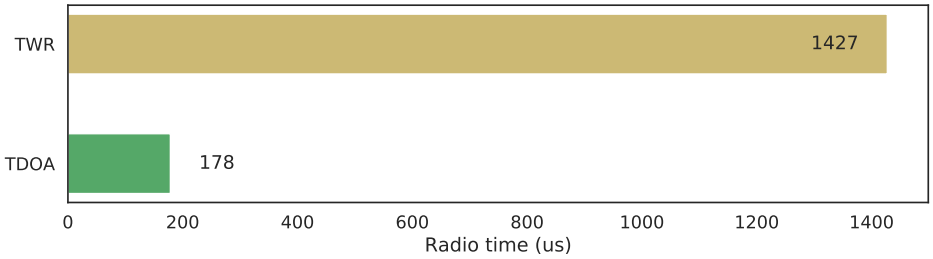


Figure 1.5: *The minimal amount of radio time required at the* TAG *to send and receive all messages required for a location measurement.*

Results of the model

The model shows the scalability of each method in terms of measurements per second in Figure 1.4. It can be seen that many more measurements can be taken with TDoA than with TWR. This can be explained by the fact that multiple messages (and response delay) between the tag and the anchors are necessary with multiple anchors for TWR, whereas TDoA only requires one message.

Requirement	TWR	TDoA
Number of supported tags (R2 & R3)	Yes	Yes
Battery life (R4 & R5)	−−	++

Table 1.1: *Ability to meet requirements*

As stated in Section 1.1 the system should be able to localise 1000+ tags (R2.) and the location should be updated every five minutes (R3.). This means the system should be able to take four measurements per second. Figure 1.4 shows that both methods meet this requirement, however TDoA has much more capacity overhead.

Furthermore the model calculates the time a tag uses its radio. Figure 1.5

shows that for TDoA this time is about $\frac{1}{16}$ of the radio time needed for TWR. This results in a much longer battery life and/or a smaller battery if TDoA is used.

Besides the ability to meet the requirements the complexity to implement a method should also be taken into account. As stated in Section 1.2.1 TWR requires coordination of all anchors for each location measurement, introducing complexity. Whereas TDoA does not require this coordination and can serve as many anchors as there are in range of a tag provided that the clocks of the anchors are synchronised.

Due to the lower complexity in coordination and the better prospect of the battery life, TDoA is chosen as a localisation method in the development of the IPS. With the extra benefit that the system could be expanded in the future to a bigger capacity and/or to a faster update rate.

1.3 Problem statement

As stated in Section 1.2.2 it is necessary that the anchors in the IPS are synchronised. This is needed because the times of reception are recorded at each anchor. When the clocks of these anchors are not synchronised, these times are not relative to each other. The UWB radio signals propagate through air at a speed of about 300,000 km/s. This means that an offset in the clocks of 1 ns will result in a distance error of about 30 cm. Therefore the anchors need to be synchronised accurately.

This can be achieved by running all anchors from the same clock source [26], [27]. However an infrastructure of clock lines must then be installed in the building. One of the constraints (C1) is that the system does not require additional infrastructures, making this solution not feasible.

Another option is to synchronise the anchors wirelessly via their UWB channel. Solutions exist that are able to synchronise anchors via UWB with a synchronisation error of approximately 1 ns [27]–[30]. However those solutions do not use off-the-shelf hardware, but work with a high precision crystal as clock source. This leads to the following problem statement:

Can a wireless synchronisation protocol be developed that minimizes the clock offsets between anchors equipped with simple clock sources?

1.3.1 Methodology

In order to address the problem statement a practical approach is chosen where the properties of the clocks will be determined on hardware that has simple clock sources.

The hardware available for this thesis is a module made by Decawave, an Irish company that makes UWB transceivers, called DWM1001. The

module contains the DW1000 UWB transceiver and a Nordic nRF52832 microprocessor as a host processor. The DW1000 chip has a simple clock source with a tolerance of ±10 Parts per million (PPM) and a PCB antenna optimized for UWB channel 5. The host processor is a 64 MHz ARM Cortex-M4 microprocessor with 64 kB RAM and 512 kB flash [31], [32].

For this hardware custom firmware needs to be developed, since the factory firmware is closed source and does not allow to take the necessary measurements to determine the clock parameters. With the firmware a dataset will be recorded where one device will broadcast synchronisation messages and multiple devices will record the TOA of this message. With this dataset different algorithms will be evaluated in order to develop the most suitable clock-synchronisation algorithm for these devices.

In order to evaluate the developed algorithm an experimental setup will be made in order to evaluate the synchronisation error. In this setup the developed algorithm will also be used to take TDoA measurements. The resulting localisation error will be evaluated.

1.4 Thesis outline

The remaining part of the thesis will be as follows:
Chapter 2 will give an overview of the state-of-the-art for UWB IPS and synchronisation. In Chapter 3 the design for the wireless synchronised indoor positioning system will be presented and implementation details are shown. The evaluation of the system will be presented in Chapter 4. Finally the conclusions and future work will be presented in Chapter 5.

Chapter 2

Related work

This chapter provides an overview of related work that has been done on synchronising clocks and localisation systems. This will help to place this work in context. Section 2.1 will provide an overview of existing methods to process TDoA measurements and Section 2.2 will describe the state of the art for synchronising (UWB) radio nodes.

2.1 Time difference of arrival

TDoA is a commonly used localisation technique, which is also applied in GPS. TDoA uses the fact that electromagnetic waves travel with a constant velocity through air. Because of this the distance between a *tag* and *anchor* is directly proportional to the propagation time of the signal. TDoA uses the difference in time at which the signal is received by multiple *anchors*. From each pair of time differences a hyperboloid can be constructed with a constant range difference between two *anchors*.

An example of one *tag* and four *anchors* can be seen in Figure 1.3. At least four receivers are required for 3D localisation. The intersection of the hyperboloids is the location of the *tag*.

In order to solve the TDoA measurements consider that there are M *anchors* and *anchor* 1 is located at the origin and the other at (x_i, y_i, z_i). The *tag* position is (x, y, z) and the distance between the *tag* and *anchor* i is denoted as r_i. Then

$$r_i^2 = (x_i - x)^2 + (y_i - y)^2 + (z_i - z)^2 \tag{2.1}$$

$$r_{i1} = cd_{i1} = r_i - r_1 \quad , \quad i = 2, 3, \ldots, M \tag{2.2}$$

Defines the set of hyperboloids on which the *tag* must lie. Where d_{i1} is the TDoA of a signal received by *anchor* pair i and 1, c is the signal propagation speed.

Several solutions exist to solve the TDoA equations, they can be placed in two categories: the iterative methods and the closed-form methods. Both will be described below.

2.1.1 Iterative methods for solving TDoA equations

In iterative methods to solve the equations, the TDoA equations are linearised by Taylor-series. These equations are then solved iteratively [33].

Using the Taylor-series expansion has two major drawbacks. First, a good initial guess of the *tags* location is required. Second, there is no guarantee that the solution will converge [33].

2.1.2 Closed-form methods for solving TDoA equations

Another way to solve the TDoA equation is a closed-form solution. Several solutions have been proposed [34]–[36], however they are not optimal in the least squares sense and perform worse than the Taylor-series method.

Chan and Ho [1] proposed a closed-form solution that is optimal in the sense that the Cramèr-Rao lower bound can be reached near a small error region. Chan's method was made for 2D localisation, but can easily be expanded to 3D localisation, for example as has been done by Ni et al. [37].

Chan's algorithm will be briefly described below. Equation (2.2) can be rewritten as follows: $r_i = r_{i1} + r_1$. Now Equation (2.1) can be expressed as:

$$\begin{aligned} r_{i1}^2 + 2r_{i1}r_1 &= K_i - 2x_i x - 2y_i y - 2z_i z \\ K_i &= x_i^2 + y_i^2 + z_i^2 \quad , \quad i = 2, 3, \ldots, M \end{aligned} \tag{2.3}$$

where equality

$$r_1^2 = x^2 + y^2 + z^2 \tag{2.4}$$

has been used for simplification. Equation (2.3) is linear with respect to x, y, z and r_1. But it is still a set of nonlinear equations in x, y and z, because r_1 is nonlinearly related to the *tags* location according to Equation (2.4).

To solve this, first assume x, y, z and r_1 as independent variables, which can be solved by the weighted linear least-squares method. A weighted linear least-squares is used because the TDoA measurements will contain measurement errors. In a second stage the known relationship for Equation (2.4) is used via another weighted linear least-squares in order to minimize the error further. This method does not need an initial guess of the *tags* location.

2.1.3 Measurement error vs. location error

The impact of the TDoA measurement error on the error of the location estimation is dependent on many factors. For example the constellation of the *anchors* and the *tag* has influence on the location error. Furthermore,

the position error on the known *anchor* position influences the error of the location estimate. Because of this behaviour, it is unknown how accurate the synchronisation of the *anchors* must be. Therefore different scenarios with different measurement noises have been simulated for the TDoA algorithm. The results can be found in Section 4.2. This is an extra motivation to evaluate the synchronisation algorithm with TDoA localisation.

2.2 Synchronisation

Synchronisation of nodes is extensively covered in literature [38]. The need for nodes to have the same understanding of time can have various applications: all with their own requirements on the accuracy of the synchronisation. For example nodes with sensors need synchronisation such that their measurements have the same time base. For this application a synchronisation accuracy in the order of milliseconds is often sufficient. Another application is nodes with wireless radios that need to adhere to a Time Division Multiple Access (TDMA) schedule. For the application in this thesis, measuring reception times of UWB messages, synchronisation with very high accuracy is required. Since the UWB messages propagate with a speed of about 300,000 km/s, synchronisation with a sub nanosecond level accuracy is required. An offset of 1 ns will introduce a ranging error of approximately 30 cm.

2.2.1 Common clock source

One way to achieve synchronised clocks at the *anchors*, is to supply all *anchors* with a common clock source [26]. A clock distribution network has to be created in order to supply the same clock to all *anchors*. A typical clock distribution network is shown in Figure 2.1.

Leugner et al. have shown that using a common clock an accuracy of about 133ps can be achieved [27]. Converting this error into a distance using the propagation speed of electromagnetic waves results in a measurement error of about 4 cm. However installing a clock distribution network would violate the constraint that not other infrastructures can be installed.

2.2.2 Clock model

In order to synchronise the local clocks of *anchors* a clock model can be created. The local clocks are driven by an oscillator and can be described by a model that relates the local time of an *anchor* to the global time, i.e. a reference *anchor*. A model based on the physical characteristics of local clocks is described in [39], [40]. The model of a local clock of an *anchor* i

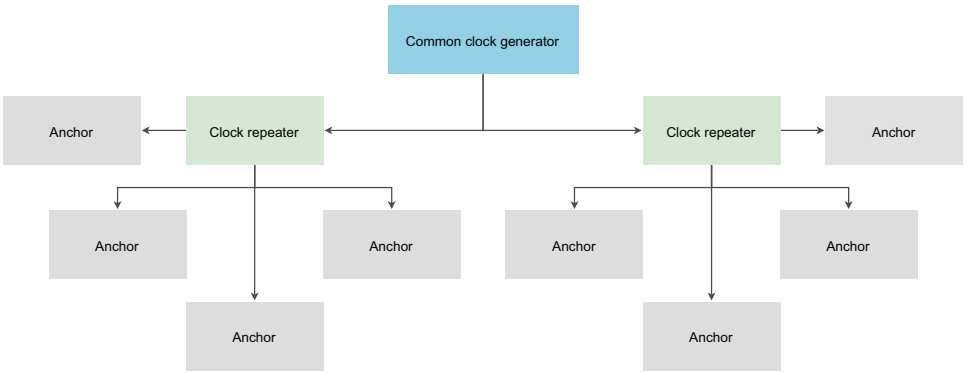


Figure 2.1: *Common clock source for anchor infrastructure*

can be described as:

$$t_i = \phi_i + \omega_i t + \frac{1}{2} D_i t^2 + \epsilon_i(t) \tag{2.5}$$

where t is global time, t_i is local time for *anchor* i, ϕ_i is the clock offset, ω_i is the clock skew, D_i is the frequency drift, and $\epsilon_i(t)$ is the time jitter due to random noise within the clock.

2.2.3 One-Way and Two-Way Time Transfer

Two types of synchronisation protocols exist: one-way synchronisation and two-way synchronisation. In one-way synchronisation the reference *anchor* is the only *anchor* sending synchronisation messages. Receiving *anchors* can synchronise using the reception times of the synchronisation messages.

The one-way synchronisation protocol is shown in Figure 2.2, *anchor* i is taken as a reference and *anchor* j is the *anchor* to be synchronised. *Anchor* i initialises the synchronisation protocol by sending a synchronisation message with his current clock time as time-stamp T_j to *anchor* j (and all other *anchors* in range). When the message is received by *anchor* j it records the reception time-stamp as R_j. τ_{ij} is the propagation time between *anchor* i and *anchor* j. ϕ_j is the offset between the clock of *anchor* i and *anchor* j e.g. due to different turn on times.

The reception time at *anchor* j can be expressed as follows:

$$\begin{aligned} T_i &= t^{(k)} \\ R_j &= \phi_j + \omega_j(t^{(k)} + \tau_{ij}) + \frac{1}{2} D_j (t^{(k)} + \tau_{ij})^2 \end{aligned} \tag{2.6}$$

where $t^{(k)}$ is the global time at the time *anchor* i sends a synchronisation message.

A classical approach to obtain an increased accuracy over one-way protocols is called the Two Way Time Transfer (TWTT) communication protocol

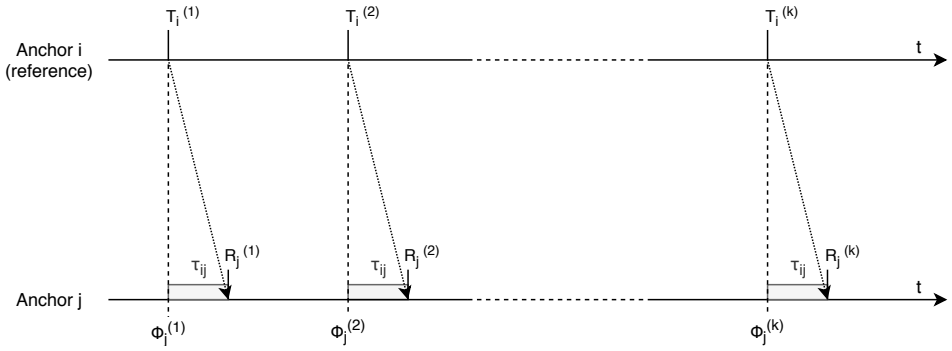


Figure 2.2: *One-way time transfer protocol*

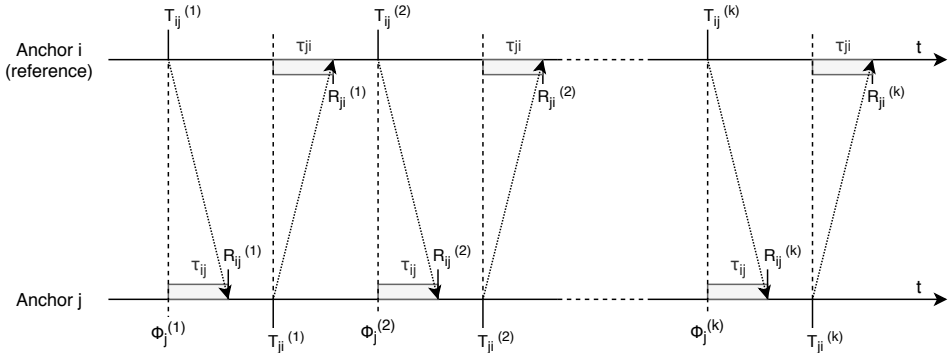


Figure 2.3: *Two-way time transfer protocol*

[41]. It can achieve an increased accuracy because it is possible to measure the propagation delay between the anchors. The TWTT starts with the same procedure as the one-way protocol, but when *anchor* j receives the synchronisation message it will respond by sending its current local time as time-stamp T_{ji}. This is shown in Figure 2.3.

Since a reference *anchor* will synchronise many other *anchors*, the TWTT is not preferred. If the reference *anchor* would need to establish bidirectional communication with all other *anchors*, the overhead of synchronisation would become too big. However the propagation delay between anchors cannot be neglected since this will be in the order of tens of nanoseconds.

2.2.4 Wireless synchronisation

Recently published papers have proposed methods to synchronise *anchors* through one-way protocols using UWB for a TDoA application [27]–[30]. All methods for wireless synchronisation that have been proposed use periodic synchronisation messages sent by a reference *anchor* to synchronise the other *anchors* (one-way synchronisation).

McElroy et al. compared different algorithms for various synchronisation

periods from 150ms to 900ms [28]. The algorithms that have been compared by McElroy et al. are linear interpolation, Proportional-Integral (PI) control, Proportional-Integral-Differential (PID) control, Proportional-Integral-Integral (PII) control and a Kalman filter. In order to compare the different algorithms McElroy et al. developed special hardware based on the Decawave DW1000 transceiver with a Temperature-compensated crystal oscillator (TCXO) that has a tolerance of ±1 PPM. McElroy et al. evaluated all their algorithms by conducting a localisation experiment with four *anchors* and one *tag*.

The different types of proposed algorithms are explained below.

Synchronisation using linear interpolation

This algorithm compared by McElroy et al. is a very simple algorithm. An *anchor* will buffer all messages between two successive synchronisation messages. When a synchronisation message is received, the TOA of the buffered messages are corrected by linearly interpolating between the TOAs of the synchronisation messages. A downside of this algorithm is that the latency of retrieving the corrected TOA is at least as long as the synchronisation period. This makes this algorithm unsuitable for real-time applications, but it is suitable for the application in this thesis since the update rate of the system is only once every five minutes.

The authors do not provide further details on how the algorithm compensates for the propagation delay and the offset between an *anchor* and the reference *anchor*. The offset can be measured with the one-way synchronisation protocol that is used. However the propagation delay cannot be measured with this algorithm.

McElroy et al. do not state the synchronisation accuracy achieved by their linear interpolation algorithm, they state their 2D location error was 14.6 cm for the 95% best measurements using a 150 ms synchronisation interval [28]

Synchronisation using linear extrapolation

Leugner et al. proposed a synchronisation algorithm based on a simplified linear version of the clock model from Equation (2.6):

$$t_j = \omega_j t_i + \phi_j \tag{2.7}$$

where ω_j is the clock skew of *anchor* j with respect to the clock skew of reference *anchor* i. ϕ_j is the offset between anchor i and j for example due to different turn on times [27].

Both the clock skew and the clock offset are determined by the one-way

synchronisation messages. The parameters are measured by:

$$\begin{aligned} \omega_j &= \frac{T_i^{(k)} - T_i^{(k-1)}}{R_j^{(k)} - R_j^{(k-1)}} \\ \phi_j &= T_i^{(k)} - R_j^{(k)} \end{aligned} \tag{2.8}$$

The TOA of messages can now be corrected by the following formula:

$$\text{TOA}^* = (\text{TOA} - R_j^{(k)})\omega_j + T_i^{(k)} \tag{2.9}$$

Leugner et al. achieved wireless synchronisation with a standard deviation of 400 ps when sending a synchronisation message every 150 ms. This precision was measured in their experimental setup where two *anchors* were placed at equal distance from a reference *anchor*. Because the *anchors* are at equal distance the propagation delay is the same for both *anchors* and is neglected in their experiment. The algorithm that is proposed does not compensate for the propagation delay. Leugner et al. developed special hardware for their experiment based on the Decawave DW1000 chip. The chip is driven by a Voltage-controlled temperature-compensated crystal oscillator (VCTCXO) with a tolerance of 1.5 PPM [42].

Another similar solution is proposed by Tiemann et al. [30]. In their solution the offset between *anchor* j and reference *anchor* j is calculated as:

$$\epsilon_j^{(k)} = R_j^{(k)} - T_i^{(k)} \tag{2.10}$$

Also the clock skew is calculated from two successive determined offsets:

$$\dot{\epsilon_j^{(k)}} = (R_j^{(k)} - R_j^{(k-1)}) f_s \tag{2.11}$$

where f_s is the synchronisation frequency. Now the error for the TOA of a message can be calculated as:

$$\epsilon_j = \epsilon_j^{(k)} + \dot{\epsilon_j^{(k)}}(\text{TOA} - R_j^{(k)}) \tag{2.12}$$

Tiemann et al. have conducted an experiment with their proposed algorithm for different synchronisation rates. They have presented their results as a position error and have not measured the synchronisation error. The 95% best location estimates in 2D had an error of approximately 12.5 cm [30]. Furthermore they concluded that a synchronisation period from 10 ms up to 1000 ms leads to approximately the same positioning error. When using a longer synchronisation period the error increases significantly.

Synchronisation using PI, PID, PII control

McElroy et al. also implemented algorithms based on the classical PI control loop. The difference between the actual time of arrival and the expected time of arrival was used as error input. Variants were created for a PII control loop and a PID control loop. The results were inferior to the linear interpolation and the Kalman filter algorithm [28] and are therefore not further discussed here.

Synchronisation using Kalman filters

Another algorithm developed for the synchronisation of UWB *anchors* uses a Kalman filter to synchronise the local clock [28], [29]. The kalman filter is used to predict the TOA between an *anchor* j and the reference *anchor* i. The TOA is approximated using the following model:

$$\mathrm{TOA}(t) = x_0 + x_1 t \tag{2.13}$$

where t is the local time that has past since the reception time of the last synchronisation message.

The state prediction is as follows:

$$\boldsymbol{x}[k] = \boldsymbol{F}\boldsymbol{x}[k-1] + \boldsymbol{Q}[k-1] \tag{2.14}$$

where $\boldsymbol{x}$ is the the state vector, $\boldsymbol{F}$ is the state transition matrix and $\boldsymbol{Q}$ is the process noise covariance matrix. These are defined as:

$$\boldsymbol{x} = \begin{bmatrix} \mathrm{TOA} \\ \mathrm{Skew} \end{bmatrix} \tag{2.15}$$

$$\boldsymbol{F} = \begin{bmatrix} 1 & dt \\ 0 & 1 \end{bmatrix} \tag{2.16}$$

where dt in the time interval between the synchronisation messages.

Measurement y is used to update the filter, which is the TOA of a synchronisation message. The measurement can be determined form the state vector:

$$y[k] = \boldsymbol{H}\boldsymbol{x}[k] + R[k] \tag{2.17}$$

where $\boldsymbol{H}$ is the measurement model matrix and $R[k]$ is the measurement noise. The measurement model matrix is defined as:

$$H = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{2.18}$$

Also for this algorithm McElroy et al. do not state the synchronisation accuracy they achieved. They state that their 2D location error was 11.3 cm for the 95% best measurements using a synchronisation period of 150 ms [28].

They have not stated if or how they compensate for the propagation delay when sending a synchronisation message.

You et al. have proposed a similar algorithm to McElroy et al. They achieved a synchronisation error of 1.02 ns at a synchronisation period of 600 ms [29]. The synchronisation error was measured by placing a tag in the middle of two *anchors*. You et al. do not state if or how they compensate for the propagation delay either.

2.2.5 Applicability of proposed wireless synchronisation algorithms for this thesis

For every proposed method the respective authors of the solution used customised hardware with a very accurate clock source. Such a clock source is not available on the hardware used in this work (factor 10 less accurate). Therefore it must be investigated whether the proposed methods will also lead to satisfactory results on the available hardware.

Another aspect that proposed solutions do not address is multi hop synchronisation. In large *anchor* networks synchronisation messages of a reference *anchor* will not be received by all the *anchors* as they are out of range.

Furthermore the proposed methods do not compensate for the propagation delay or it is not stated how propagation delay is compensated in their experiments. This has to be resolved in the synchronisation algorithm as well.

Chapter 3

Design and Implementation

As covered in Chapter 2 current solutions to synchronise UWB anchors are created for more expensive hardware and do not state if and how they compensate for the offset due to propagation delays. This could be determined based on the positions of the anchors. However if there is no line-of-sight between the anchors the calculated propagation delay will be wrong.

Furthermore the proposed solutions are not designed for multi hop networks. This chapter covers design choices and implementation details of synchronisation algorithms and TDoA solver.

In this chapter two contributions are described. First a synchronisation algorithm that uses a Kalman filter to track a second order clock model of the local clock, described in Section 3.3.2. Second a multi-hop synchronisation scheme which has not yet been proposed for UWB networks, described in Section 3.5.2.

3.1 System overview

The system architecture, as presented in Figure 3.1, consists of one or more floors that are equipped with multiple UWB anchors and a server that communicates with the anchors via a existing Wi-Fi infrastructure. The server is responsible for assigning and maintaining transmission timeslots making sure that no collisions occur due to anchors transmitting at the same time. Also the server will calculate the location estimate of the tags based on the measurements taken by the anchors. These tasks are performed at a central server because both tasks require information from multiple anchors and can therefore not be performed locally.

The anchors will report the TOAs of messages sent by the tags to the server. As stated in Chapter 1 it is required that the anchors are synchronised accurately. In order to achieve synchronisation over all anchors different roles for anchors are introduced. The synchronisation algorithms run on the firmware of the anchors.

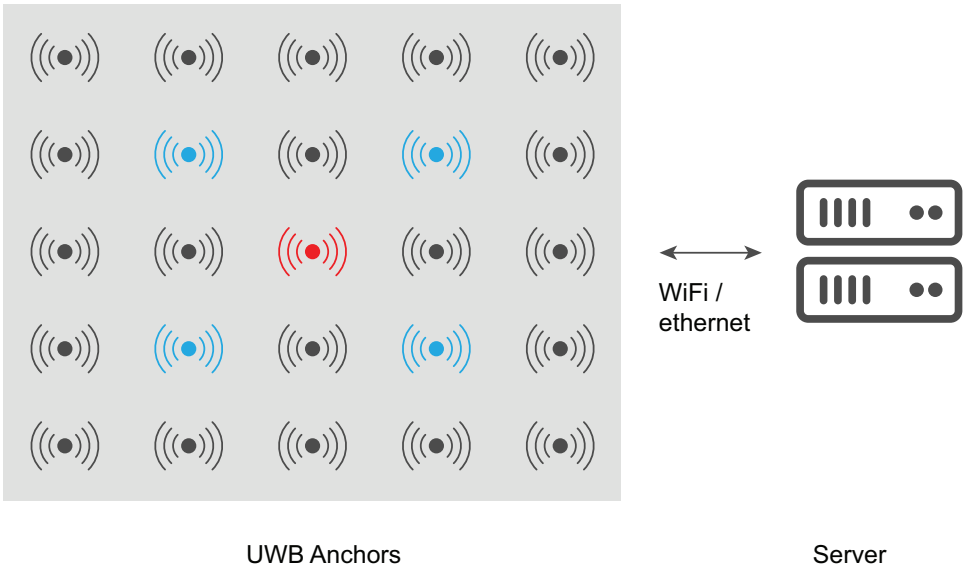


Figure 3.1: *System overview; on the left a floor equipped with UWB anchors is shown; the UWB anchors communicate with a server via WiFi or ethernet*

First one anchor is assigned the *reference anchor* role. This type is depicted in red in Figure 3.1. The local clock of this anchor will be used as global time for the set of anchors to be synchronised. This anchor will initialise the synchronisation protocol. Just as all other anchors, the reference anchor will report TOAs of tags to the server.

Furthermore there are *normal anchors* depicted in black in Figure 3.1. These anchors will listen for synchronisation messages and synchronise their local clocks accordingly. Also these anchors will report TOAs of tags to the server.

Because the UWB radios have a limited range and the radio signal of the reference anchor is not received by all anchors in most cases a third role for anchors is introduced. This anchor is called a *relay* anchor and is depicted in blue in Figure 3.1. This type of anchor will distribute the synchronisation message further in the set of anchors. Of course this anchor will also report TOAs of tags to the server.

Note that the three roles can be assigned to any anchor in the set of anchors. No special hardware is needed to perform a certain role. The role of relay anchor must be assigned to the anchors carefully. In order to have the least overhead the number of relay anchors must be kept minimal, while the reference anchor and the relay anchors must cover all anchors in the set.

This thesis focus on the synchronisation of the anchors. Since the hard-

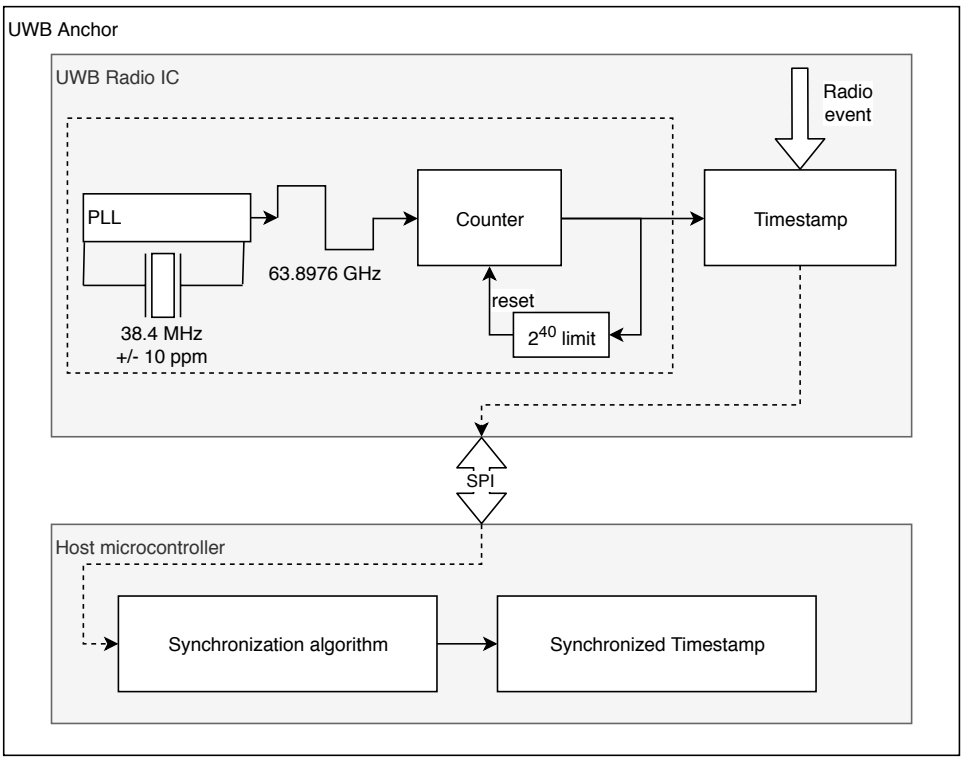


Figure 3.2: *Hardware architecture of UWB module*

ware modules are not connected to a Wi-Fi interface this part of the system has not been implemented. Instead the anchors communicate through the reference anchor via UWB that is connected to the server via a Universal Serial Bus (USB) connection.

3.1.1 Anchor hardware

The UWB modules used in the anchors are equipped with an UWB transceiver and a host processor as shown in Figure 3.2. The transceiver and the host processor communicate via a Serial Peripheral Interface (SPI). The transceiver has an internal hardware counter that is used to timestamp events that happen on the transceiver such as transmitting or receiving a message. This counter is incremented at a frequency of 63.8976 GHz which makes timestamping with a precision of 15.6 ps possible.

The clock that increments this hardware counter is generated by a Phase Locked Loop (PLL) that uses an external crystal to generate it. This external crystal has a frequency of 38.4 MHz and a tolerance of ±10 PPM. The transceiver has the ability to tune the frequency of the external crystal. During fabrication of the module the tune values are determined in a calibration process. Tuning of the crystal happens at startup of the transceiver

based on the programmed values. It is not possible to adjust the tuning of the crystal while the transceiver is running.

The hardware counter used to timestamp events is a 40 bits counter that overflows approximately every 17.2 s. This counter cannot be adjusted by the host processor nor can the frequency of the clock, that increments the counter, be adjusted by the host processor. Therefore it is necessary to create a model of the clock on the host processor and correct the timestamps retrieved from the transceiver afterwards.

When measuring and evaluating clocks of electronic devices a common method is to measure a clock signal or General Purpose Input/Output (GPIO) triggered by the clock using an external measurement device. Since it is not possible to adjust the clock in the UWB transceiver it has no purpose to measure the clock signal. It is also not possible to trigger GPIO by the hardware counter. This means that the clock offset has to be measured by reading the receive and transmit timestamps of the anchors.

3.2 Measure clock parameters

In order to measure clock parameters of the available anchors the following setup has been created. One anchor is assigned the role of reference anchor. This anchor periodically sends precisely timed UWB messages that are received by all other anchors in the setup. The reference anchor is connected to a computer via USB through which all measured data is communicated and saved.

The other anchors in the setup will record the receive timestamps of the UWB messages sent by the reference anchor. Each recorded timestamp will be reported back to the reference anchor via a UWB message. This way the reference anchor has knowledge of the reception timestamps of all anchors for each periodically sent message. This data is sent to the computer where it is saved for analysis. This method is more convenient than connecting every anchor in the setup to the computer and merging the data afterwards.

As mentioned in Section 1.3.1 the firmware loaded onto the UWB modules is not suitable for the experiments and algorithms in mind. Therefore new firmware has been developed including a driver for the UWB transceiver. The firmware has been developed using FreeRTOS [43]. The use of this operating system on the host processor ensured time critical tasks are executed on time.

3.2.1 Avoiding collisions

Since every anchor in the setup will respond to the message sent periodically by the reference anchor, it has to be made sure anchors will not send at the same time. If two anchors will send at the same time collisions will occur,

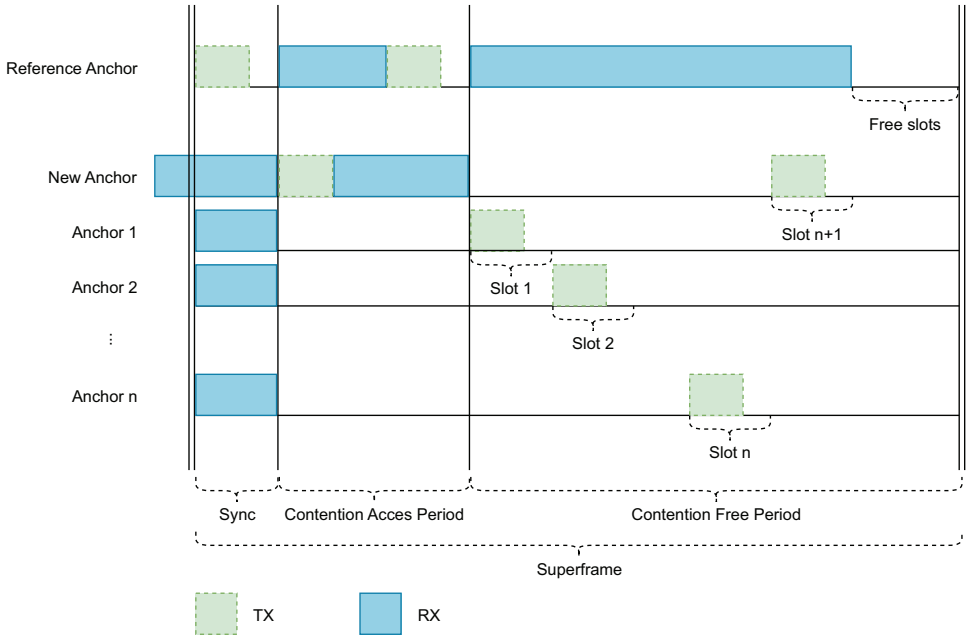


Figure 3.3: *Structure of the TDMA frame*

resulting in message loss. Unfortunately the UWB transceivers are not able to sense if other anchor is sending.

To avoid this situation a TDMA scheme has been implemented where each anchor is assigned a timeslot in which it may send its message. The scheme is defined by a super frame that is repeated, presented in Figure 3.3. The super frame consists of three phases. It starts with the sync phase. The purpose of this phase is twofold. It indicates the beginning of a new super frame and it is the precisely timed UWB message that must be received by all other anchors.

The next phase is the Contention Access Period (CAP). During this phase anchors that do not have been allocated a timeslot yet can send a request, e.g. anchors that have recently been turned on. At the end of this phase the reference anchor will send a confirmation if a slot has been allocated. This phase is called contention access period since during this period anchors that do not have an agreement on a timeslot try to get to an agreement on a timeslot. If an unallocated anchor does not receive a confirmation on his request it will send a new request during the next super frame.

The last phase is called the Contention Free Period (CFP). This is the phase where all anchors are in agreement on when each anchor can send its message. If a timeslot has been allocated to an anchor it can send its message and no other anchor will send a message during this time so that no collisions will occur. In the remaining time after the last allocated timeslot and before the next super frame starts no messages are sent.

The designed TDMA scheme has been implemented in the firmware as a table-based finite state machine. This ensures that messages are always correctly handled according to the design.

3.2.2 Sending a precisely-timed UWB message

As mentioned in Section 3.2 the message sent by the reference anchor has to be precisely timed. Besides being timed precisely its transmission time must be predictable so that the transmission time can be sent as payload of the message. The UWB transceiver has the capability to schedule a transmission. The scheduled transmission time is the time when the marker symbol of the message is sent by the antenna. It compensates for delays introduced by the path to the antenna. This mechanism allows to send the transmit time of a message in the payload of the message itself.

3.2.3 Transfer timestamps to the computer

A common way to exchange data over a serial communication channel is to send it as ASCII text. However, this is not very efficient as it requires more data to be sent over the USB connection than necessary and the processor has to parse the recorded data to ASCII. Therefore a communication protocol was developed to transfer the recorded data as raw bytes that can be parsed by the computer and saved in a human readable file format.

To achieve this a simple point-to-point protocol was implemented with a start byte and a stop byte so the computer knows when a data message starts and when it ends. The data structure of the recorded data was sent over this protocol as raw bytes. A Python program on the computer was able to cast these raw bytes to the original data structure and place them in a Comma Separated File (CSV) file.

3.2.4 Measurement setup

In order to get insight in the clock parameters of the local clocks from anchors, the offset between the anchors and reference anchor has been measured periodically. The setup for this static experiment consisted of seven anchors from which one anchors was assigned the role of reference anchor. All anchors were placed at equal distance from the reference anchor.

Measurements were taken for approximately two minutes. As synchronisation algorithms proposed by others have synchronisation periods from 100 ms up to 10 s this dataset allows analysis over multiple synchronisation periods for every already proposed algorithm. The measurements were taken at a frequency of 20 Hz, twice the frequency of the fastest proposed algorithm to provide a reasonable resolution.

The results of the experiment are shown in Figure 3.4. Almost linear behaviour is observed where offset is between 50 µs and 230 µs after 120 s. This

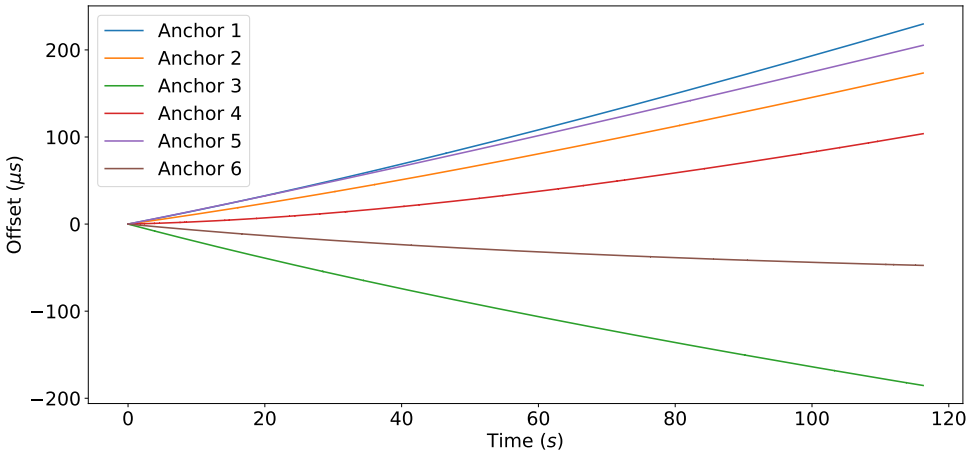


Figure 3.4: *Experimental evaluation of the clock offset for six anchors*

experiment shows the need for clock synchronisation as 230 µs corresponds with 69 km at the speed of electromagnetic waves in air.

It can be seen in Figure 3.5, where the derivative of the offset is shown, that the skew is not as linear as it appeared to be. Only the derivative of anchor 1 is shown as the other derivatives are comparable. Because the skew is not linear the parameters of the clock must be determined periodically.

3.3 Skew and drift compensation algorithms

In order to address the clock skew and drift of the local clocks several already-proposed algorithms are evaluated on the dataset collected by the measurement described in Section 3.2. Also some possible improvements of the algorithms have been implemented and tested.

All different algorithms have been evaluated for different synchronisation periods. As shown in Figure 3.5 the skew is not linear as the crystals of the anchors are subject to changes in their environment and these changes are different for every measurement. So to make a fair comparison the same dataset was used to evaluate the algorithms for different synchronisation periods. For synchronisation periods longer than 50 ms the measurements were decimated. This ensured all algorithms were using the same data.

The implementation details of the different algorithms and the optional improvements are described in this section. All algorithms have been implemented in Python to evaluate them on the dataset. As will be presented in Chapter 4 linear interpolation gives the best results. This algorithm was also implemented on the anchors in C.

As explained in Section 3.1.1 all receive and transmit timestamps are saved values from the hardware counter. These values from the counter are 40-bit unsigned integers. The counter overflows every 17.2 s, which means

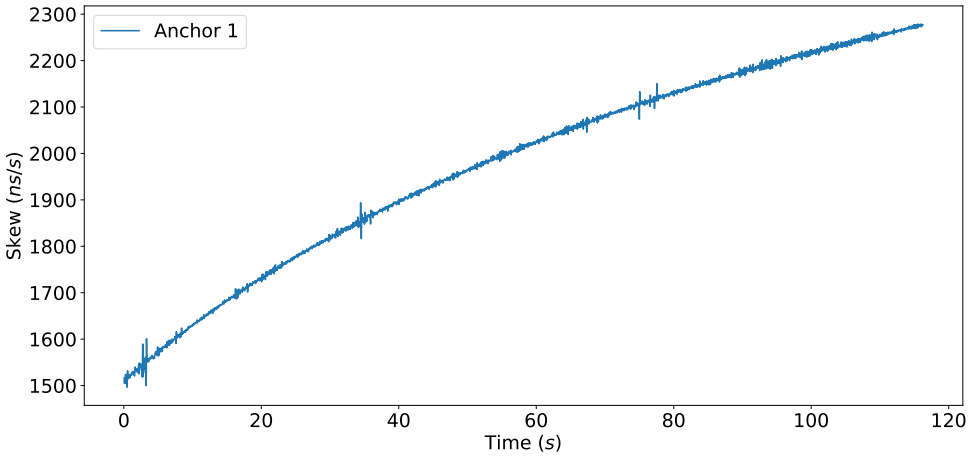


Figure 3.5: *Experimental evaluation of the clock skew of* anchor 1 *(first derivative of the offset)*

that a lower value of the timestamp might actually be further in time. If the timestamps are subtracted a negative time difference would be calculated. The algorithm should take this into account. As 17.2 s is much longer than a synchronisation period will be the algorithms should be able to handle only a single overflow in the timestamps.

To address this problem all arithmetic executed on the timestamps is followed by an AND operation with a 40-bit bitmask. As a result subtracting a bigger timestamp from a smaller will result in the actual time difference instead of a negative time difference. Furthermore the overflow that would happen on the hardware counter is also applied to the corrected timestamp.

3.3.1 Linear extrapolation

The linear extrapolation algorithm was implemented in a similar way as Leugner et al. have implemented it [27]. Timestamps are referenced as in the explanation of the one-way time transfer protocol discussed in Section 2.2.3. The clock of anchor j is modelled according to the following equation

$$t_j(t) = \omega t + \phi \tag{3.1}$$

where ω is the clock skew and ϕ is the clock offset. The algorithm will determine the clock skew and the reference time.

The algorithm holds the following variables:

$T^{(k)}$	Transmit timestamp of last received synchronisation message
$R^{(k)}$	Receive timestamp of last received synchronisation message
Skew	The current skew of the clock

Upon the arrival of a synchronisation message $k+1$ the UWB transceiver will determine the receive timestamp, $R^{(k+1)}$. The payload of the message contains the transmit timestamp of the message, $T^{(k+1)}$. The variables of the algorithm are update as followed:

Algorithm 1: Update clock model

Input: $T^{(k+1)}$, $R^{(k+1)}$

1 Skew $= \frac{R^{(k+1)} - R^{(k)}}{T^{(k+1)} - T^{(k)}}$

2 $T^{(k)} = T^{(k+1)}$

3 $R^{(k)} = R^{(k+1)}$

For all messages that arrive that are not synchronisation messages a synchronised timestamp can be calculated with the determined clock parameters. For these messages it is assumed that the TOA is larger than $R^{(k)}$. The synchronised timestamp will then be calculated as follows:

Algorithm 2: Correct time of arrival of a message

Input: TOA

Output: TOA^*

1 delta $= TOA - R^{(k)}$

2 $TOA^* = T^{(k)} +$ delta/Skew

3 **return** TOA^*

Moving average

In order to address potential measurement errors a moving average over the clock skew was added to the linear extrapolation algorithm. The moving average acts as a low pass filter over the determined skew. The filter removes abrupt changes in the determined skew that might be the result of a measurement error.

The moving average was implemented in the extrapolation algorithm by replacing the skew variable for an array of determined skews. The size of this array corresponds with the size of the moving average filter.

The moving average is applied when calculating the synchronised TOA of a message. The synchronised timestamp is calculated using the following formula:

$$TOA^* = T^{(k)} + \text{delta/Mean(Skew)} \tag{3.2}$$

3.3.2 Kalman filter

A synchronisation algorithm based on a Kalman filter was also implemented. The clock parameters can be tracked by the Kalman filter in two ways. McElroy et al. [28] filter the TOA of the the synchronisation messages. The state vector then contains the TOA and the skew. Another way is to filter the offset between the local clock and the reference clock which You et

al. [29] have proposed. The state vector then contains the offset defined as $R^{(k)} - T^{(k)}$ and its derivative.

If the TOA would be used as an input the input would overflow every 17.2 s as this time is determined by the hardware timer. Because using timestamps that overflow as an input for a Kalman filter will result in complications and wrong behaviour of the filter it was chosen to implement a filter that filters the offset.

Two versions of a Kalman filter algorithm have been implemented. The first version is comparable with the algorithm proposed by You et al. [29]. It is a two-state model of the local clock. This model is based on a first order clock model, just like the linear extrapolation algorithm. The other filter is a 3-state filter that to the best of our knowledge has not been used to synchronise UWB clocks. This algorithm is described in Section 3.3.2 and extends the filter described in this section.

The filter keeps track of the clock offset of the local clock with respect to the reference clock and its derivative, depicted by δ. This relates to the clock skew in the following way: $\omega = 1 + \delta$.

$$\boldsymbol{x} = \begin{bmatrix} \text{offset} \\ \delta \end{bmatrix} \tag{3.3}$$

The state prediction from the epoch k is as follows:

$$\boldsymbol{x}^{-}[k+1] = \boldsymbol{F}[k]\boldsymbol{x}^{+}[k] \tag{3.4}$$

$$\boldsymbol{P}^{-}[k+1] = \boldsymbol{F}[k]\boldsymbol{P}^{+}[k]\boldsymbol{F}[k]^{T} + \boldsymbol{Q}[k] \tag{3.5}$$

where $\boldsymbol{x}$ and $\boldsymbol{P}$ denote the state vector and its covariance matrix. The $-$ and $+$ superfix indicate the priori and posteriori state. Furthermore the state transition matrix is defined as:

$$\boldsymbol{F}[k] = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \tag{3.6}$$

where T is the synchronisation period. The process noise covariance matrix $\boldsymbol{Q}$ is dependent on the interval duration.

$$\boldsymbol{Q}[k] = T\boldsymbol{Q}_{n}orm \tag{3.7}$$

where $\boldsymbol{Q}_{n}orm$ is the time normalized state covariance matrix, a diagonal 2-by-2 matrix. The filter is updated with a measurement of the offset of the local clock. An innovation vector $\boldsymbol{y}$ and its covariance matrix $\boldsymbol{S}$ are computed

$$\boldsymbol{y}[k] = z[k] - \boldsymbol{H}\boldsymbol{x}^{+}[k-1] \tag{3.8}$$

$$\boldsymbol{S}[k] = \boldsymbol{H}\boldsymbol{P}^{-}[k]\boldsymbol{H}^{T} + R[k] \tag{3.9}$$

where z is the measured clock offset and R is the covariance of the measurement. According to Decawave the variance of the measurements taken by the UWB chip are $\sigma^2 = 1.5 \cdot 10^{-20} s^2$ [28]. So $R = \sigma^2$. The measurements are equal to the first element of the state vector, therefore the measurement model matrix is $\boldsymbol{H} = [1, 0]$. Now the Kalman gain $\boldsymbol{K}$ and the posteriori state vector are computed by applying:

$$\boldsymbol{K}[k] = \boldsymbol{P}^{-}[k]\boldsymbol{H}^{T}\boldsymbol{S}[k]^{-1} \tag{3.10}$$

$$\boldsymbol{x}^{+}[k] = \boldsymbol{x}[k] + \boldsymbol{K}[k]\boldsymbol{y}[k] \tag{3.11}$$

$$\boldsymbol{P}^{+}[k] = (\boldsymbol{I} - \boldsymbol{K}[k]\boldsymbol{H})\boldsymbol{P}^{-}[k] \tag{3.12}$$

where $\boldsymbol{I}$ is the identity matrix.

To correct the TOA of a message sent by a tag it is again assumed that the TOA is larger than $R^{(k)}$, the reception timestamp of the last synchronisation message. A method similar to the prediction step of the Kalman filter is used. The offset at the TOA of a message sent by a tag is calculated as

$$\phi = \boldsymbol{F}_1 \boldsymbol{x}^{+} \tag{3.13}$$

where $\boldsymbol{F}_1$ is a matrix that predicts the measurement at the TOA of the message using the posteriori state of the Kalman filter. $\boldsymbol{F}_1$ is dependent on the interval between the reception of the last synchronisation message and the reception of the message from the tag and defined by $\boldsymbol{F}_1 = [1, T_m]$. T_m denotes the interval. The TOA can now be corrected by subtracting the calculated offset from the TOA.

Three state model

In all proposed methods using a Kalman filter [28], [29] a 2-state Kalman filter was used. As was observed in Figure 3.5 the skew was not linear. Therefore it makes sense to implement a 3-state Kalman filter based on a 2nd order clock model as described in Section 2.2.2.

To the best of our knowledge this type of filter has not yet been proposed for synchronisation in UWB.

This is implemented by adjusting the 2-state Kalman filter in the following way. State vector $\boldsymbol{x}$ now also contains the frequency drift. The state transition matrix is extended

$$F[k] = \begin{bmatrix} 1 & T & \frac{1}{2}T^2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix} \tag{3.14}$$

Also the matrix $\boldsymbol{F_1}$ to predict the offset at the TOA of a message from a tag is extended

$$\boldsymbol{F_1} = [1, T_m, \frac{1}{2}T_m^2] \tag{3.15}$$

3.3.3 Linear interpolation

The last algorithm that was implemented is linear interpolation. In this algorithm the TOAs of messages from tags are buffered and corrected when a synchronisation message arrives. As a consequence it might take up to one synchronisation period before the corrected TOA becomes available. Fortunately this is not a problem for the system in this project because an update rate of 5 min is sufficient and the synchronisation period will be much smaller than 5 min.

The algorithm holds the following variables:

$T^{(k)}$	Transmit timestamp of the last received synchronisation message
$R^{(k)}$	Receive timestamp of the last received synchronisation message
$TOA[]$	Array with the receive timestamps of all messages received from tags

When the anchor receives a synchronisation message the variables are updated and the buffered receive timestamps are corrected by interpolating between the synchronisation messages.

Algorithm 3: Process synchronisation message

Input: $T^{(k+1)}$, $R^{(k+1)}$
Output: $TOA^*[]$

```
Skew = (R^(k+1) − R^(k)) / (T^(k+1) − T^(k))
for TOA in TOA[] do
    delta = TOA − R^(k)
    TOA*[] = T^(k) + delta/Skew
end
T^(k) = T^(k+1)
R^(k) = R^(k+1)
return TOA*
```

Message loss

Since the synchronisation messages are sent over a wireless medium, message loss should be considered. If the algorithm does not receive a synchronisation message it will proceed buffering the TOAs of messages from tags. When it receives the next synchronisation message it will correct the buffered TOAs eventually. Only ΔT and ΔR will be twice as large as normal. This has the same effect as increasing the synchronisation period by a factor 2.

Implementation on the host processor

As will be presented in Chapter 4 the linear interpolation resulted in the smallest clock offset between the anchors. Therefore this algorithm has been

implemented in the firmware of the anchors, which is written in C. As the anchors have a processor with limited computing power the algorithm was implemented using only integer arithmetic.

To efficiently calculate the corrected timestamps on the micro controller the algorithm keeps track of the skew as $\Delta T = T^{(k+1)} - T^{(k)}$ and $\Delta R = R^{(k+1)} - R^{(k)}$. With these values a timestamp can be corrected in the following way

$$\delta = TOA - R^{(k)}$$
$$TOA^* = T^{(k)} + \frac{\delta * \Delta T}{\Delta R}$$

As explained in Section 3.1.1 timestamps are the values of the 40-bit hardware counter that is incremented with a frequency of 63.8976 GHz. To use these timestamps on the host processor they are stored in 64-bit unsigned integers, being the smallest datatype that will fit the timestamp. If the synchronisation period is for example 1 s the value of ΔT will be approximately $63.9 \cdot 10^9$. The value of δ can also reach this value just before a new synchronisation message arrives. The product of these two number will then be to big to fit in in 64-bit unsigned integer.

The programming language C does not support bigger integer types natively. If the division would be performed before this multiplication the result will fit in the 64-bit integer. However doing so will greatly reduce the precision of the result. Therefore an alternative representation of the large integers was implemented. In this implementation the value was stored in an array of multiple smaller integers. In this case storing the timestamps in three 32-bit integers was sufficient to allow the multiplication. For these data representation a multiplication and division algorithm were implemented that performed operation on 64-bit integers such that the overflow could be saved in another 32-bit integer.

3.4 Determining the propagation delay

In Section 2.2.3 it was explained that the propagation delay between the reference anchor and other anchors is significant and cannot be neglected. Therefore the delay should be determined. One option is to use a Two Way Time Transfer (TWTT) to measure the propagation delay, however bidirectional communication is established with each anchor every synchronisation period. This would make the overhead of the synchronisation algorithm very big.

Another method is to calculate the propagation delay based on the location of the anchors. The delay between *anchor* i and *anchor* j can be calculated using the following formula:

$$\tau_{ij} = \frac{d_{ij}}{c} \tag{3.16}$$

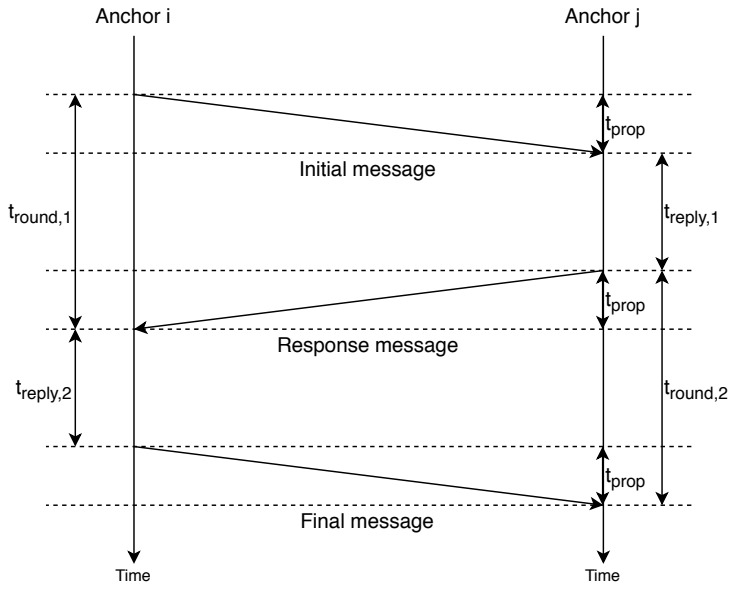


Figure 3.6: *Symmetrical Double Sided Two-way ranging scheme*

where d_{ij} is the euclidean distance between between *anchor i* and *anchor j* and c is the propagation speed of electromagnetic waves in air.

The euclidean distance between *anchor i* and *anchor j* can be calculated as the euclidean norm of the subtracted position vectors from *anchor i* and *anchor j*, $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$.

$$d_{ij} = \|\boldsymbol{p}_i - \boldsymbol{p}_j\| \tag{3.17}$$

The advantage of calculating the propagation delay is that it does not require any communication between the anchors. And the exact positions of the anchors are also needed by the TDoA solver. The downside is that it is a theoretical determined delay. In reality the delay might be different. For example due to an object between the anchors that obstructs the UWB signal. The receiving anchor then receives an reflection of the signal which travelled a longer distance. Furthermore measuring the propagation delay can compensate for errors in the calibrated antenna delays, discussed in Section 3.2.2.

For the reasons mentioned above it is preferred to determine the propagation delay based on a measurement instead of calculating the delay. The propagation delay is considered constant as the anchors will not move and objects are nog likely to be moved near the ceiling. Therefore it is not necessary to measure the delay every synchronisation period. A hybrid solution is proposed with a measured propagation delay, but not the overhead of a TWTT.

An accurate way of measuring the propagation delay is called Symmetrical Double Sided Two-way Ranging (SDS-TWR). For this method three

messages need to be sent between the anchors. The process is shown in Figure 3.6 and is proposed by Decawave [44]. When the process is finished the propagation delay can be calculated as:

$$\tau_{ij} = \frac{t_{round,1} \cdot t_{round,2} - t_{reply,1} \cdot t_{reply,2}}{t_{round,1} + t_{round,2} + t_{reply,1} + t_{reply,2}} \tag{3.18}$$

The SDS-TWR method will be used to measure the propagation delay. In Section 3.5.2 a synchronisation scheme will be proposed containing this method but does not perform this with each anchor every synchronisation period.

The determined propagation delay should be subtracted from every $T^{(k)}$ in order to compensate for the propagation delay between the reference anchor and the anchor.

3.5 Applying the synchronisation algorithm in multi hop networks

In most cases the range of the reference anchor will not be large enough to reach all anchors in the system. As a solution the relay anchor is introduced. This anchor will distribute the synchronisation message further into the set of anchors.

3.5.1 Determine transmit timestamp

Just like the synchronisation messages sent by the reference anchor the synchronisation messages sent by the relay anchor have to contain the transmit timestamp. However at the relay anchor it should not contain the transmit timestamp of its local clock but the synchronised transmit timestamp based on its synchronisation algorithm.

The difference with correcting the transmit timestamp with respect to correcting the receive timestamps is that the transmit timestamp has to be extrapolated using Equation (3.2).

3.5.2 Synchronisation scheme

In order to coordinate the distribution of the synchronisation messages in the set of anchors a synchronisation scheme has been designed, shown in Figure 3.7. The scheme starts by the reference anchor sending a synchronisation message. Then the relay anchors that received the message will send a synchronisation message. This stage is called a hop. If a set of nodes covers a large area there will be multiple hops that will happen subsequently.

During a hop multiple relay anchors will transmit their synchronisation message. The range of their UWB signals might overlap, so they cannot

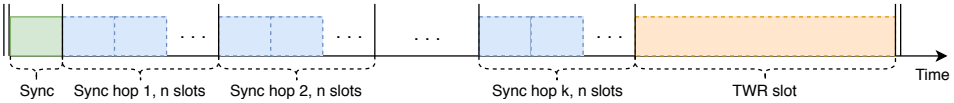


Figure 3.7: *Synchronisation scheme*

send at the same time. To address this every hop has multiple slots that can be assigned to individual relay anchors.

Lastly there is also a slot for SDS-TWR in order to measure the propagation delay between two anchors. This slot will be assigned to an anchor pair if a new anchor is added to the set. If the propagation delay of all necessary anchor pairs have been measured the slot can be used to redo a measurement of an anchor that shows large synchronisation errors. The slot allocation is communicated to the anchors by the server over Wi-Fi An anchor can detect synchronisation errors by correcting the receive timestamp of a synchronisation message using Equation (3.2) and compare this with the $T^{(k)}$. If the anchor detects a bias it can request a SDS-TWR slot to update the propagation delay.

3.6 Estimate position from TDoA measurements

Now that the synchronisation algorithm is implemented on the anchor it is able to take TDoA measurements. To estimate the position of the tag based on these measurements a solver has to be designed. As discussed in Section 2.1 a closed-form solver is preferred as no prior information about the tag location is necessary. Therefore Chan's algorithm [1], [37] was implemented for 3D localisation. This implementation is described in detail in Appendix B

3.6.1 Collect measurements form the anchors

In order to collect measurements form the anchors the system would use the Wi-Fi connection between the anchors and the server. As explained in Section 3.1 the setup for experiments does not have this Wi-Fi interface.

To collect the measurements in the current setup the TDMA scheme presented in Section 3.2.1 is extended with slots for tags. The anchors are running the synchronisation algorithm and report the synchronised TOAs to the reference anchor that is connect via USB.

As the TOAs are synchronised at the anchors it is sufficient to only send a part of the timestamps to the server. Only the lowest 16 bits of the timestamp are sent to the server. Allowing to report a time difference up to approximately 1025 ns as the frequency of the counter is 63.8976 GHz. Being able to report a time difference of 307 m is more than sufficient for an IPS.

Chapter 4

Evaluation

The performance of the synchronisation algorithms and the multilateration algorithm described in Chapter 3 are evaluated in this chapter. The skew and drift compensation of the algorithms is evaluated in Section 4.1 and the multilateration performance in combination with the synchronisation algorithm is evaluated in Section 4.2.

4.1 Clock skew and drift compensation

In this section the different synchronisation algorithms described in Chapter 3 are evaluated. To evaluate the algorithms the error between a local clock and the clock of the reference anchor is measured.

4.1.1 Experimental setup

In order to measure the clock error the dataset from the experiment described in Section 3.2 is used. In this experiment the reference anchor was sending synchronisation messages to six anchors. Those anchors were all placed at an equal distance from the reference anchor so the propagation delay is equal for all anchors. The anchors report the TOA from the synchronisation messages.

The error between an anchor's clock and the reference anchor's clock can be determined by comparing the corrected TOA of a synchronisation message with the transmit timestamp of the synchronisation message. All synchronisation messages were used to evaluate the errors, but only part of the messages were given as an input for the algorithm depending on the synchronisation period which had to be a multiple of 50 ms.

4.1.2 Synchronisation period

As already mentioned in Section 3.2 and as can be seen in Figure 3.5 the clock skew is not linear. Therefore the clock parameters must be updated

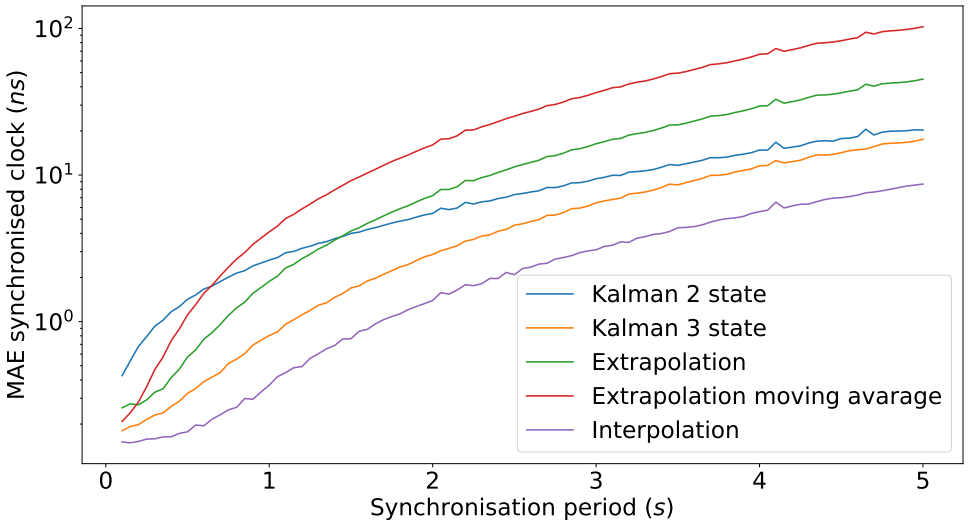


Figure 4.1: *MAE of the clock vs. the synchronisation period for the implemented algorithms*

periodically. The influence of the synchronisation period is evaluated for all the implemented algorithms.

Every anchor has its own local clock with different parameters so the performance of an algorithm will be different on every anchor. In order to compare the performance of the different algorithms the Mean Absolute Error (MAE) was calculated over the synchronised clocks of all the different anchors. The result of this test is visualised in Figure 4.1.

The MAE varies from 151 ps up to 102 ns corresponding to a ranging error from 4.5 cm up to 30.57 m. Where the standard deviation of clock error can be up to 75 ns. Note the error is greatly dependent on the synchronisation period. It is clear that the interpolation algorithm is superior to all the other algorithms in terms of the MAE. Therefore it was chosen to implement this algorithm in the firmware and perform the multi hop and multilateration experiment. The algorithm is further investigated in this section.

To evaluate the effect of changing the synchronisation period on the clock error for the linear interpolation algorithm the Cumulative Distribution Function (CDF) for different synchronisation periods has been created from the absolute clock errors of all *anchors*. The CDFs are depicted in Figure 4.2. In this plot it can be seen how many of the measured clock errors samples are within a certain clock error relative to all samples. The faster the line reaches 1 the smaller the clock error for this synchronisation period.

It can be observed that the CDFs for synchronisation periods under 1 s are close to each other and above are further apart. Nearly all errors are below 1 ns for a synchronisation period of 1 s corresponding to a range error below 30 cm. So it seems like a synchronisation period of 1 s is the best trade

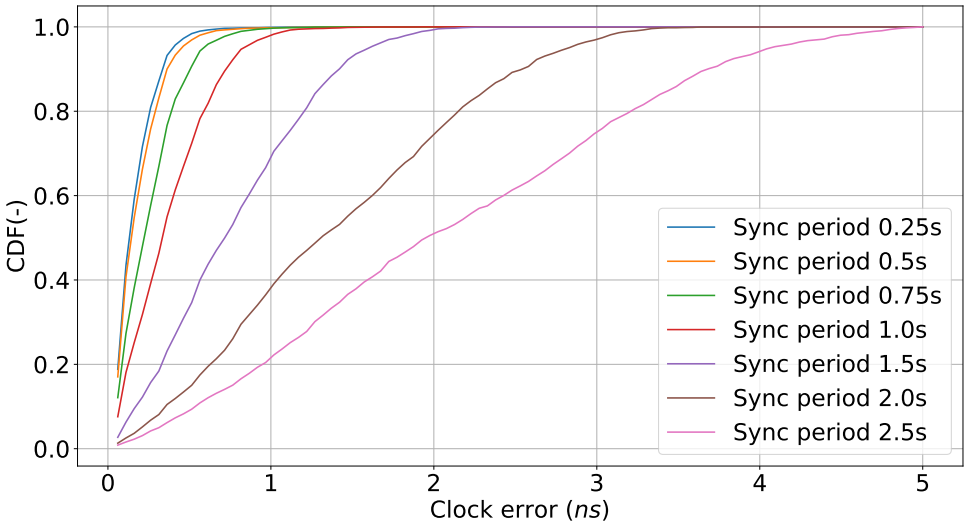


Figure 4.2: *Cumulative Distribution Function of the clock error for different synchronisation periods when using the linear interpolation algorithm*

off between accuracy and synchronisation overhead.

The histogram of the clock error of the clocks synchronised by linear interpolation at a synchronisation period of 1 s is shown in Figure 4.3. As can be seen in the histogram the clock error has a mean of −330 ps and a standard deviation of 308 ps. Ideally the mean would be 0 s. The mean of the interpolation algorithm is very close to this ideal value with a small standard deviation. The bias of the mean could be caused by the floating point representation used to express the skew. As can be seen in Section 4.1.5 the bias is not present anymore when the algorithm was implemented with integer arithmetic.

4.1.3 Moving average linear extrapolation

As mentioned in Chapter 3 an attempt was made to improve the linear extrapolation algorithm by using a moving average over the determined clock parameters. It was already shown in Figure 4.1 that the moving average algorithm performs worse than the normal linear interpolation algorithm.

The moving average algorithm has one tune parameter namely the size of the moving average window (i.e. how many samples it averages). The larger the filter the more robust it will be to high frequent errors but that also makes the filter slower. This parameter is evaluated in Figure 4.4. This figure shows the MAE of the clock for different window sizes evaluated with a synchronisation period of 1 s.

It can be observed in Figure 4.4 that increasing the size of the filter has a negative effect since the MAE of the clock increases. Apparently the

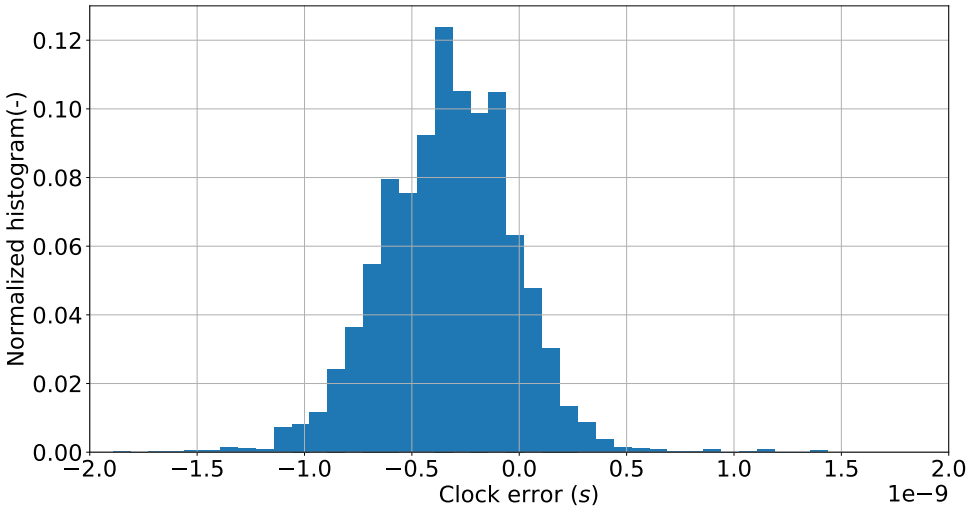


Figure 4.3: *Clock error with a synchronisation period of* 1 s *($\mu = -3.30 \cdot 10^{-10}s, \quad \sigma = 3.08 \cdot 10^{-10}s \quad n = 3796$)*

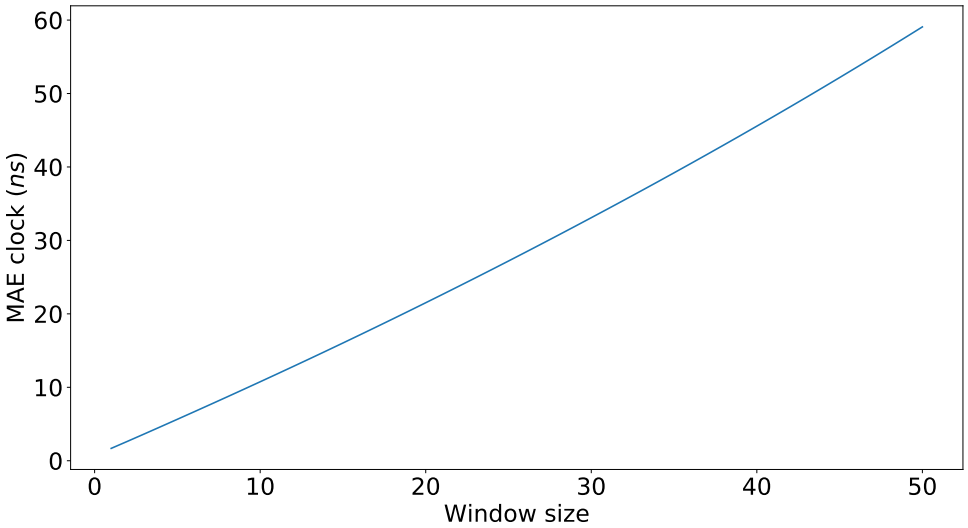


Figure 4.4: *Clock error vs. size moving average*

clock parameters are changing faster than the filter can keep up with. As increasing the window size does not reach a limit the skew must also have low frequency components.

4.1.4 2-state vs 3-state Kalman filter

Also for the Kalman synchronisation algorithm an improvement has been implemented. The existing 2-state Kalman filter solution was extended to a 3-state Kalman filter. As a result the algorithm can estimate the clock

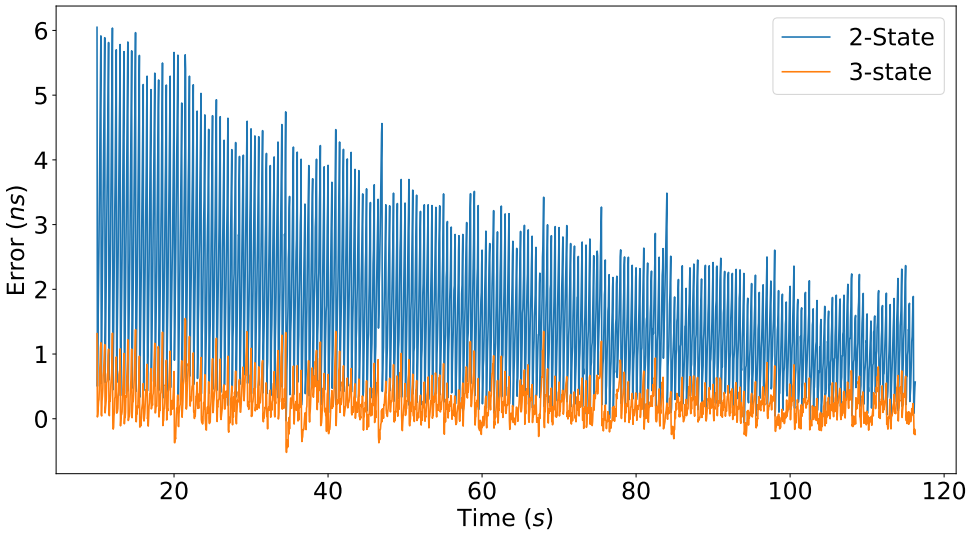


Figure 4.5: *Clock error over time*

parameters much better. This can be observed in Figure 4.5 where the error over time is plotted. The error of the clock synchronised by the 3-state Kalman filter is approximately a factor 3 smaller than the clock synchronised by the 2-state Kalman filter.

Furthermore the settling time of the 3-state Kalman filter to account for drift is much shorter as the influence of the clock drift is modelled as well. This has a lot of influence during the warm up time op the crystal.

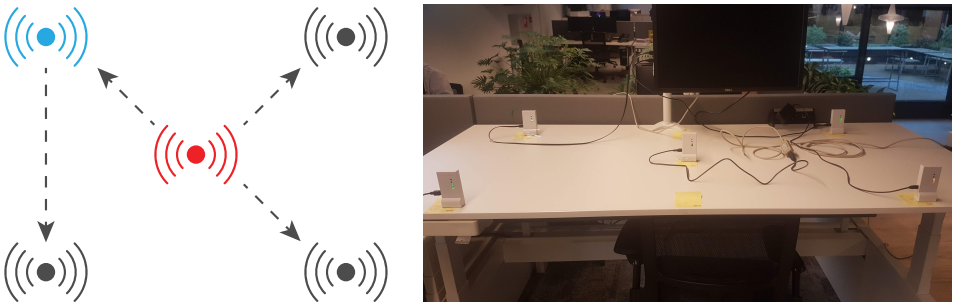
If the latency introduced by the linear interpolation algorithm would have been a problem for the IPS then the 3-state Kalman filter algorithm would have been the preferred method. For a similar performance as the linear interpolation algorithm the synchronisation period of the Kalman filter should be shortened by a factor two.

4.1.5 Multi hop synchronisation

The synchronisation algorithm is also able to synchronise in a multi hop network. In this scenario the reference anchor synchronises the anchors that are in its range. Some of those anchors have the role of relay anchor that will repeat the synchronisation message to all anchors in its range that are not in the range of the reference anchor. This is called a hop. There can be multiple hops in a network.

It is important that the clock error of the anchor does not increase between the hops. A tag might send a message that is received by anchors that have been synchronised by different relay anchors. If the clock error between those hops becomes too big it is not possible to measure the TDoA.

During this experiment only one hop is evaluated. This is reasonable as



(a) Overview of the experimental setup.

(b) Picture of the experimental setup

Figure 4.6: *Experimental setup for evaluating multi hop synchronisation*

a message sent by a tag will not cross more than one hop.

Experimental setup

In order to measure the clock error of *anchors* that are one hop apart the interpolation algorithm was implemented in the firmware of the *anchors* including compensation for the measured propagation delay. The role of *reference anchor*, *relay anchor* and normal *anchor* were implemented. The *anchors* were placed as presented in Figure 4.6a. A picture of the experimental setup can be seen in Figure 4.6b.

The *reference anchor* is placed in the middle (depicted in red) and at equal distance are four *anchors* surrounding the *reference anchor*. One of those *anchors* has the role of *relay anchor* (depicted in blue). The *relay anchor* will relay the synchronisation message to the *anchor* below it. That *anchor* will ignore the synchronisation messages form the *reference anchor*.

During the experiment the *reference anchor* will send synchronisation messages with a synchronisation period of 1 s. In between the synchronisation messages the *reference anchor* will send blink messages that are used to evaluate the clock error of the *anchors*. Both the transmit timestamp of the blink message at the reference anchor and the synchronised receive timestamps at the other anchors are sent to the computer. The clock error can be evaluated

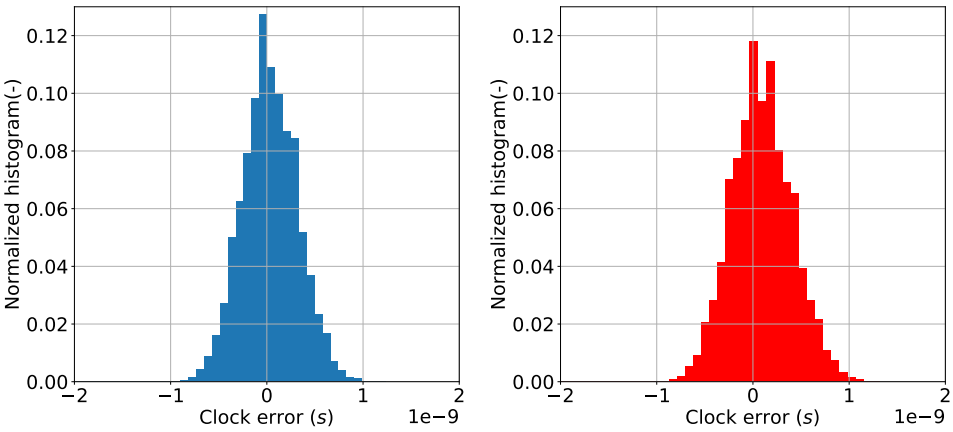
The reception timestamp at anchor i of a blink message can be expressed as

$$R_i^{(k)} = T_{ref}^{(k)} + \tau_i$$

where τ_i is the propagation delay between the reference anchor and anchor i which is known in this setup (because the distance is known). By using this equation the clock error between the anchors and the reference anchor can be determined.

Clock error

The results of the experiment are shown in Figure 4.7a and Figure 4.7b. Figure 4.7b shows the clock error in the multi hop experiment. For comparison also the histogram of the same setup with a single hop is presented in Figure 4.7a. The histograms show the difference between a single hop network and a multi hop network with a hop distance of 1. As expected the result is similar to the result from the interpolation algorithm based the dataset.



(a) Single hop clock error with a synchronisation period of 1 s($\mu = 2.93 \cdot 10^{-11} s$, $\sigma = 2.85 \cdot 10^{-10} s$)

(b) Multi hop clock error with a synchronisation period of 1 s($\mu = 9.31 \cdot 10^{-11} s$, $\sigma = 3.12 \cdot 10^{-10} s$)

Figure 4.7: *Clock error in multi hop synchronisation*

Furthermore it can be seen that the clock error is hardly affected by the fact that the synchronisation message was relayed by the relay anchor. The mean is almost identical only the standard deviation became slightly bigger. From this result it can be concluded that TDoA measurements are possible in a multi hop anchor network.

4.2 Multilateration

In this section the position estimation based on measurements corrected by the linear interpolation algorithm is evaluated to determine if the clock synchronisation is good enough to determine a position within 1 m (Requirement 1). Anchors are used to measure the TDoAs of a message sent by a tag. These measurements are used to estimate the position of the tag. The error between the estimated position and the real position is estimated.

4.2.1 Experimental setup

In order to measure the TDoAs of a tag five anchors were placed in a room, one reference anchor and four normal anchors. The tag is placed at 15 different known positions in the area bounded by the anchors.

The anchors were placed on a square area of 2 m by 5 m. Although this is not a very large area the results should still be representable as the standard deviation of the TOA estimation is unaffected by the distance [28].

The anchors were synchronised using the linear interpolation algorithm with a synchronisation period of 1 s. During the experiment one tag was used that was placed on each position consecutively. Per position measurements were taken for at least 60 seconds at 15 Hz. This was repeated three times. The last 200 measurements of each run per position were evaluated.

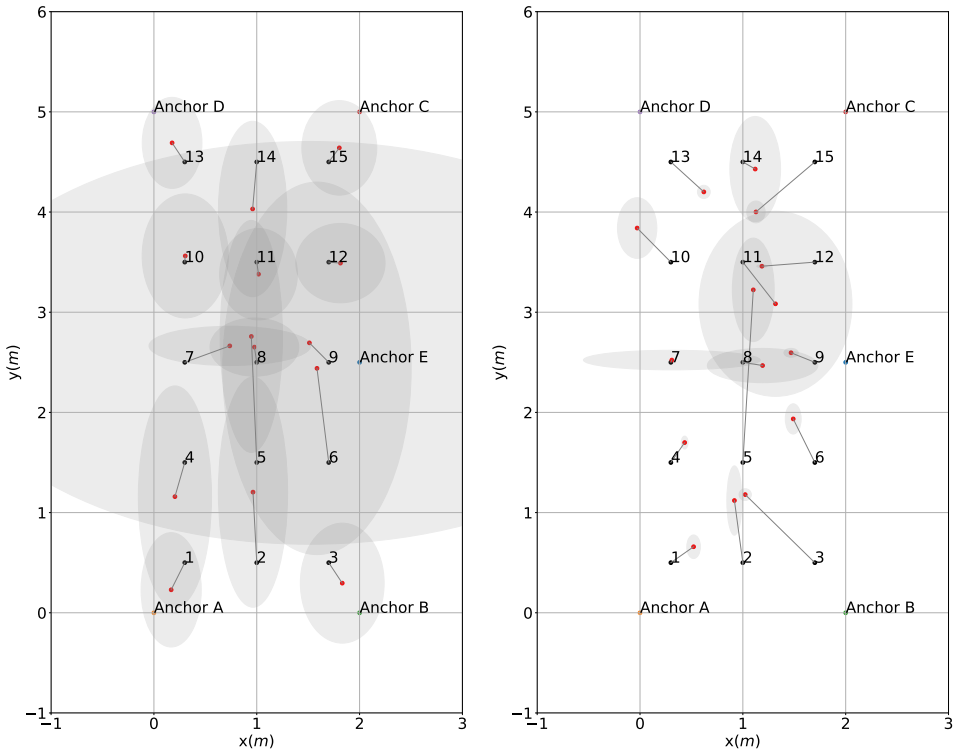
The position estimations will be evaluated using two metrics. The first metric is the error between te real position and the geometric mean of all estimations. This metric relates to the accuracy of the estimates. The second metric is the standard deviation of the location estimates which relates to the precision of the estimates.

4.2.2 Position error

In order to compare the results of the experiment the same setup of tags and anchors has been simulated with an expected measurement error with a standard deviation of $2 \times \sigma_{synchronisation}$ ($\sigma_{synchronisation} = 2.85 \cdot 10^{-10}$, evaluated in Section 4.1.5) as the measurement consist of two measured TOAs on different anchors. The simulation added random samples of a normal (gaussian) distribution to each TOA at an anchor. For every position 3,000 simulations have been run. The results of the simulation are shown in Figure 4.8a and in Table 4.1. In Figure 4.8a the five anchors are depicted as A through E, where E is the reference anchor. The positions where the tag was placed are depicted as 1 through 15 in black. An ellipse with the size of the standard deviation is plotted around the mean of the estimated positions.

The results of the practical experiment are shown in Figure 4.8b and Table 4.1. During the experiment the standard deviation of the location estimates is smaller for most positions, which means that the precision of the estimates in the experiment was higher than the precision in the simulations. This could be explained by the clock error tending to be positive or negative at the same time for all anchors. That would decrease the standard deviation of the measurement error since it was assumed to be twice the standard deviation of the synchronisation experiment.

The error of the mean position estimate is for some positions comparable to the simulations such as position 5, which is estimated too high in both the simulation and the experiment. Furthermore some positions are opposite to



(a) Simulation of experimental setup

(b) Results of experimental setup

Figure 4.8: *Experimental setup to evaluate multilateration. Five anchors are setup in a room depicted by A..D. Multilateration has been evaluated for 15 tag positions depicted by 1..15. The black dots indicate the real position, the red dots indicate the mean of the estimates and the ellipse around the estimate indicates the standard deviation.*

the positions of the simulation.

Figure 4.9 shows the location estimates of position 15, all estimations are concentrated on a line. This line might be one of the hyperbolas that defines the distance difference between to anchors. The location estimates might be pushed along the hyperbola because of a faulty measurement.

In order to evaluate where the position error originates from the theoretical TDoA between anchors for a tag's position has been compared to what was measured by the anchors. The combination of all errors form all measurements of all positions leads to the histogram shown in Figure 4.10. It can be seen that all measurement errors are normally distributed between $-2\,\text{ns}$ and $2\,\text{ns}$ with a standard deviation of $586\,\text{ps}$. This is smaller than $2 \times \sigma_{synchronisation} = 570\,\text{ps}$, which is the error that can be expected from the linear interpolation algorithm.

Furthermore positions where the estimation was accurate stayed accurate

Sim: $\|\|\boldsymbol{p}-\mu_{\hat{\boldsymbol{p}}}(m)\|\|$	Sim: $\sigma_x(m)$	Sim: $\sigma_y(m)$	Exp: $\|\|\boldsymbol{p}-\mu_{\hat{\boldsymbol{p}}}(m)\|\|$	Exp: $\sigma_x(m)$	Exp: $\sigma_y(m)$
0.32	0.59	1.14	0.27	0.13	0.24
0.58	0.67	2.30	0.63	0.15	0.69
0.28	0.81	1.20	0.96	0.12	0.12
0.50	0.71	2.21	0.24	0.07	0.13
1.30	0.60	2.32	1.77	0.41	1.03
1.05	1.83	3.72	0.48	0.15	0.30
0.44	1.57	0.39	0.02	1.71	0.20
0.16	0.86	0.58	0.19	1.07	0.34
0.14	7.08	4.02	0.25	0.14	0.09
0.05	0.84	1.24	0.47	0.38	0.61
0.12	0.76	0.91	0.52	1.49	1.85
0.08	0.87	0.79	0.52	0.10	0.03
0.18	0.58	0.90	0.44	0.12	0.13
0.52	0.66	1.75	0.14	0.49	1.05
0.23	0.73	0.94	0.76	0.18	0.22

Table 4.1: *Position estimates multilateration experiment for positions 1 trough 15*

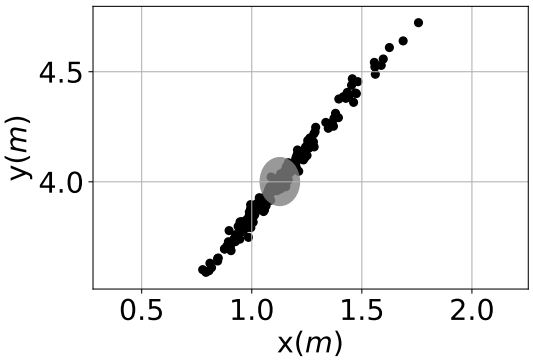


Figure 4.9: *Scatter plot of all estimations of position 15. A circle with the standard deviation as radius is plotted in grey*

over a long time and positions where the estimation was not accurate stayed not accurate over a long time. From this observation it is not expected that the synchronisation algorithm malfunctioned. The same experiment was also conducted with a synchronisation period of 0.5 s which should give more accurate synchronisation. Still the same errors were encountered.

Unfortunately the local clock timestamps of the anchors were not recorded so it cannot be said with 100 percent certainty that the errors are not due to bad clock synchronisation. In order to evaluate this the experiment should

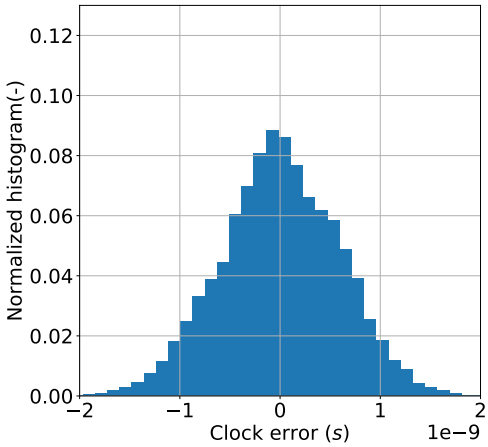


Figure 4.10: *Measurement errors during experiment*($\mu = 1.19 \cdot 10^{-12}s, \quad \sigma = 5.86 \cdot 10^{-10}s$)

be conducted again.

The mean position error of all locations was 51 cm and are all (except one outlier of 1.77 m) within 1 m. So by averaging measurements and by filtering the result the requirement of 1 m accuracy can be met.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

At Topic Embedded Systems various healthcare solutions are being developed. Recently the ability to localise equipment inside hospitals using Ultra-wideband has been in demand. The requirements and constraints of the Indoor Positioning System in mind were composed in consultation with Topic. The location accuracy must be within 1 m, the system must support 1000+ tags and the position must be updated every 5 min.

A literature survey showed that UWB with Time Difference of Arrival is the most suitable localisation method. Provided that all anchors can be synchronised at a nano second level accuracy.

Existing synchronisation algorithms for UWB either require a wired clock infrastructure or require precision clock sources that have very tight tolerances. To keep cost down existing wireless synchronisation algorithms were evaluated and improved to work with simple clock sources instead. It was concluded that a linear interpolation algorithm will give the most accurate synchronisation. The interpolation algorithm comes at the cost of a latency as the timestamps become available after the next synchronisation message, which is acceptable for the IPS of TOPIC as the position must be updated only once every 5 s. If the latency cannot be allowed a 3-state Kalman filter is the best alternative.

The linear interpolation synchronisation algorithm was implemented on the available hardware from Decawave. Evaluation of the algorithm showed that the Mean Absolute Error of the clock error was 229 ps with $\mu = 29.3 \cdot 10^{-12}s, \quad \sigma = 285 \cdot 10^{-12}s$. The developed algorithm can be expanded over large anchor networks using the proposed multi-hop synchronisation scheme. It was shown that relaying synchronisation messages hardly influences the synchronisation accuracy.

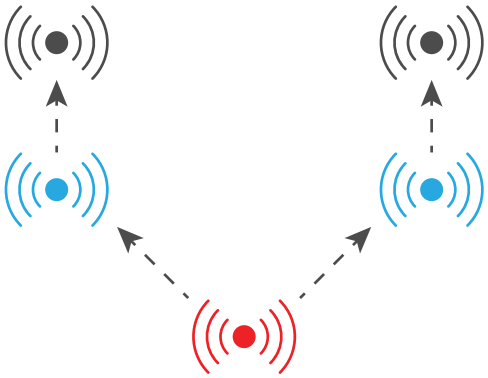


Figure 5.1: *Multihop limitation*

The synchronisation algorithm was further evaluated by implementing a multilateration positioning algorithm to estimate a position based on the TDoA measurements taken by the anchors. It was observed through a series of real-world experiments that accuracy of a single position estimate falls quite often outside the required 1 m range.By averaging over multiple measurements, however, the position error was reduced to 51 cm.

5.2 Limitations

Multihop synchronisation

When synchronising a multi hop *anchor* network it can happen that two *anchors*, although they are close to each other, have a completely different path back to the *reference anchor*. An example of this situation is shown in Figure 5.1.In this work only the clock error has been evaluated for anchors that are one hop apart. The clock error will probably be a lot bigger if the *anchors* are more hops apart. This situation has not been tested and therefore it cannot be said of localisation would still be possible in this situation.

One way to overcome this situation is to maintain multiple instances of the synchronisation algorithm, one for each *reference/relay anchor* it can receive. It will communicate the TOAs of messages in both timebases to the server. The server can then select the timebase that is mutual for all TOAs.

Multilateration

As was shown in Section 4.2 the accuracy and precision are hardly sufficient to meet the requirement of localisation with 1 m accuracy. It can only be achieved when averaging over many estimations.

New experiments should be conducted to further exclude the synchronisation accuracy as the source for the low performance of localisation. Fur-

thermore it must be investigated what is the source of this error and maybe evaluate different multilateration algorithms.

Another way to overcome the threshold performance might be to develop a filter that discards measurement outliers.

5.3 Future Work

The currently open problems and some proposals for improvement are given below.

Synchronisation slot allocation

The synchronisation slots in the proposed synchronisation scheme are not very efficient or flexible. In the current scheme the network can only go to the next hop if every *anchor* in the first hop is synchronised. However part of the network that is synchronisation as first could already synchronise the next hop. To enable this more efficient situation smart slot allocation must be developed.

Ensure *tags* do not interfere with the synchronisation scheme

This work has focused on synchronising the *anchors*. However the IPS also has *tags* in the network that send UWB messages. As the synchronisation scheme is a vital part of the IPS collisions with synchronisation messages should be avoided. One way to achieve this is by implementing a TDMA scheme where *tags* are allocated a slot outside the transmit time of synchronisation messages.

Another solution would be to synchronise at another UWB channel than the *tags* will send their messages. This way the *tags* do not have to keep track of time so they will use less energy. If the *tags* do not have to request a slot or keep track of time they do not need to be able to receive UWB messages. As the receiving part of an UWB transceivers is by far more complex than the transmitting part of a transceiver this makes the *tags* very low cost and energy efficient.

Investigate the performance in non line-of-sight conditions

The ranging capabilities of UWB radios are dependent on whether there is line of sight between the radios. It will probably also influence the synchronisation capabilities of the radios. To mitigate these errors the scheme measures the propagation delay instead of calculating them based on the positions of the *anchors*. The efficiency of this mechanism must still be evaluated.

Bibliography

[1] Y. T. Chan and K. C. Ho, "An efficient closed-form localization solution from time difference of arrival measurements", in *Proceedings of ICASSP '94. IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. ii, Apr. 1994, II/393–II/396 vol.2. DOI: 10.1109/ICASSP.1994.389638.

[2] S. Ram and J. Sharf, "The people sensor: A mobility aid for the visually impaired", in *Digest of Papers. Second International Symposium on Wearable Computers (Cat. No.98EX215)*, Oct. 1998, pp. 166–167. DOI: 10.1109/ISWC.1998.729548.

[3] H. Koyuncu and S. H. Yang, "A survey of indoor positioning and object locating systems", p. 8, 2010.

[4] (). Embedded system development — TOPIC embedded systems, [Online]. Available: https://topic.nl/en (visited on 05/07/2019).

[5] C. T. Stock and T. Sundt, "Timeout for checklists?:" *Annals of Surgery*, vol. 261, no. 5, pp. 841–842, May 2015, ISSN: 0003-4932. DOI: 10.1097/SLA.0000000000001141.

[6] A. Alarifi, A. Al-Salman, M. Alsaleh, A. Alnafessah, S. Al-Hadhrami, M. A. Al-Ammar and H. S. Al-Khalifa, "Ultra wideband indoor positioning technologies: Analysis and recent advances", *Sensors (Basel, Switzerland)*, vol. 16, no. 5, 16th May 2016, ISSN: 1424-8220. DOI: 10.3390/s16050707.

[7] M. A. Al-Ammar, S. Alhadhrami, A. Al-Salman, A. Alarifi, H. S. Al-Khalifa, A. Alnafessah and M. Alsaleh, "Comparative survey of indoor positioning technologies, techniques, and algorithms", in *2014 International Conference on Cyberworlds*, Oct. 2014, pp. 245–252. DOI: 10.1109/CW.2014.41.

[8] F. Evennou and F. Marx, "Advanced integration of WiFi and inertial navigation systems for indoor mobile positioning", *EURASIP J. Adv. Signal Process*, vol. 2006, pp. 164–164, Jan. 2006, ISSN: 1110-8657. DOI: 10.1155/ASP/2006/86706.

[9] Y. Wang, Xu Yang, Yutian Zhao, Yue Liu and L. Cuthbert, "Bluetooth positioning using RSSI and triangulation methods", in *2013 IEEE 10th Consumer Communications and Networking Conference (CCNC)*, Jan. 2013, pp. 837–842. DOI: 10.1109/CCNC.2013.6488558.

[10] R. Want, A. Hopper, V. Falcão and J. Gibbons, "The active badge location system", *ACM Trans. Inf. Syst.*, vol. 10, no. 1, pp. 91–102, Jan. 1992, ISSN: 1046-8188. DOI: 10.1145/128756.128759.

[11] M. Hazas and A. Hopper, "Broadband ultrasonic location systems for improved indoor positioning", *IEEE Transactions on Mobile Computing*, vol. 5, no. 5, pp. 536–547, May 2006, ISSN: 1536-1233. DOI: 10.1109/TMC.2006.57.

[12] S. Fang, C. Wang, T. Huang, C. Yang and Y. Chen, "An enhanced ZigBee indoor positioning system with an ensemble approach", *IEEE Communications Letters*, vol. 16, no. 4, pp. 564–567, Apr. 2012, ISSN: 1089-7798. DOI: 10.1109/LCOMM.2012.022112.120131.

[13] H. D. Chon, S. Jun, H. Jung and W. An, "Using RFID for accurate positioning", *Journal of Global Positioning Systems*, vol. 3, no. 1, pp. 32–39, 31st Dec. 2004, ISSN: 14463156, 14463156. DOI: 10.5081/jgps.3.1.32.

[14] J. Tiemann, F. Eckermann and C. Wietfeld, "ATLAS - an open-source TDOA-based ultra-wideband localization system", in *2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Alcala de Henares, Spain: IEEE, Oct. 2016, pp. 1–6, ISBN: 978-1-5090-2425-4. DOI: 10.1109/IPIN.2016.7743707.

[15] G. Krukar, M. Wenzel, P. Karbownik, N. Franke and T. v. d. Grün, "Proof-of-concept real time localization system based on the UWB and the WSN technologies", in *2014 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Oct. 2014, pp. 756–757. DOI: 10.1109/IPIN.2014.7275559.

[16] A. Lo, A. Yarovoy, T. Bauge, M. Russell, D. Harmer and B. Kull, "An ultra-wideband (UWB) ad hoc sensor network for real-time indoor localization of emergency responders", in *Advances in Vehicular Networking Technologies*, M. Almeida, Ed., InTech, 11th Apr. 2011, ISBN: 978-953-307-241-8. DOI: 10.5772/14232.

[17] J. Tiemann, F. Schweikowski and C. Wietfeld, "Design of an UWB indoor-positioning system for UAV navigation in GNSS-denied environments", in *2015 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Oct. 2015, pp. 1–7. DOI: 10.1109/IPIN.2015.7346960.

[18] L. Zwirello, T. Schipper, M. Harter and T. Zwick, "UWB localization system for indoor applications: Concept, realization and analysis", *Journal of Electrical and Computer Engineering*, vol. 2012, pp. 1–11, Feb. 2012, ISSN: 2090-0147. DOI: 10.1155/2012/849638.

[19] A. Ledergerber, M. Hamer and R. D'Andrea, "A robot self-localization system using one-way ultra-wideband communication", in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2015, pp. 3131–3137. DOI: 10.1109/IROS.2015.7353810.

[20] M. Hamer and R. D'Andrea, "Self-calibrating ultra-wideband network supporting multi-robot localization", *IEEE Access*, vol. 6, pp. 22 292–22 304, 2018, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2018.2829020.

[21] P. Ali-Rantala, L. Ukkonen, L. Sydanheimo, M. Keskilammi and M. Kivikoski, "Different kinds of walls and their effect on the attenuation of radiowaves indoors", in *IEEE Antennas and Propagation Society International Symposium. Digest. Held in conjunction with: USNC/CNC/URSI North American Radio Sci. Meeting (Cat. No.03CH37450)*, vol. 3, Jun. 2003, 1020–1023 vol.3. DOI: 10.1109/APS.2003.1220085.

[22] D. Madigan, E. Einahrawy, R. P. Martin, W.-H. Ju, P. Krishnan and A. S. Krishnakumar, "Bayesian indoor positioning systems", in *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, vol. 2, Mar. 2005, pp. 1217–1227. DOI: 10.1109/INFCOM.2005.1498348.

[23] J. Hallberg, M. Nilsson and K. Synnes, "Positioning with bluetooth", presented at the International Conference on Telecommunications : Special Session on IoT Emerging Technologies: Design and Security (ITEMS'16) 16/05/2016 - 18/05/2016, IEEE Communications Society, 2003, pp. 954–958.

[24] B. Silva, Z. Pang, J. Åkerberg, J. Neander and G. Hancke, "Experimental study of UWB-based high precision localization for industrial applications", in *2014 IEEE International Conference on Ultra-WideBand (ICUWB)*, Sep. 2014, pp. 280–285. DOI: 10.1109/ICUWB.2014.6958993.

[25] N. C. Rowe, A. E. Fathy, M. J. Kuhn and M. R. Mahfouz, "A UWB transmit-only based scheme for multi-tag support in a millimeter accuracy localization system", in *2013 IEEE Topical Conference on Wireless Sensors and Sensor Networks (WiSNet)*, Jan. 2013, pp. 7–9. DOI: 10.1109/WiSNet.2013.6488616.

[26] Decawave. (2014). Wired synchronization of anchor nodes in a TDOA real-time location system, [Online]. Available: https://www.decawave.com/wp-content/uploads/2018/10/APS007_Wired-Sync-RTLS-With-The-DW1000_v1.10.pdf.

[27] S. Leugner, M. Pelka and H. Hellbrück, "Comparison of wired and wireless synchronization with clock drift compensation suited for u-TDoA localization", in *2016 13th Workshop on Positioning, Navigation and Communications (WPNC)*, Oct. 2016, pp. 1–4. DOI: 10.1109/WPNC.2016.7822846.

[28] C. McElroy, D. Neirynck and M. McLaughlin, "Comparison of wireless clock synchronization algorithms for indoor location systems", in *2014 IEEE International Conference on Communications Workshops (ICC)*, Australia: IEEE, Jun. 2014, pp. 157–162, ISBN: 978-1-4799-4640-2. DOI: 10.1109/ICCW.2014.6881189.

[29] B. You, X. Li and W. Liu, "A kalman-filter-based wireless clock synchronization method in indoor localization", in *Eighth International Conference on Digital Image Processing (ICDIP 2016)*, vol. 10033, International Society for Optics and Photonics, 29th Aug. 2016, 100335Y. DOI: 10.1117/12.2243983.

[30] J. Tiemann, F. Eckermann and C. Wietfeld, "Multi-user interference and wireless clock synchronization in TDOA-based UWB localization", in *2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Alcala de Henares, Spain: IEEE, Oct. 2016, pp. 1–6, ISBN: 978-1-5090-2425-4. DOI: 10.1109/IPIN.2016.7743696.

[31] Decawave. (). Dwm1001-dev_datasheet.pdf, [Online]. Available: https://www.decawave.com/sites/default/files/dwm1001-dev_datasheet.pdf (visited on 24/06/2019).

[32] ——, (). Dwm1001_datasheet.pdf, [Online]. Available: https://www.decawave.com/sites/default/files/dwm1001_datasheet.pdf (visited on 24/06/2019).

[33] W. H. FOY, "Position-location solutions by taylor-series estimation", *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-12, no. 2, pp. 187–194, Mar. 1976, ISSN: 0018-9251. DOI: 10.1109/TAES.1976.308294.

[34] B. Friedlander, "A passive localization algorithm and its accuracy analysis", *IEEE Journal of Oceanic Engineering*, vol. 12, no. 1, pp. 234–245, Jan. 1987, ISSN: 0364-9059. DOI: 10.1109/JOE.1987.1145216.

[35] H. Schau and A. Robinson, "Passive source localization employing intersecting spherical surfaces from time-of-arrival differences", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 35, no. 8, pp. 1223–1225, Aug. 1987, ISSN: 0096-3518. DOI: 10.1109/TASSP.1987.1165266.

[36] J. Smith and J. Abel, "Closed-form least-squares source location estimation from range-difference measurements", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 35, no. 12, pp. 1661–1669, Dec. 1987, ISSN: 0096-3518. DOI: 10.1109/TASSP.1987.1165089.

[37] J. Ni, D. Arndt, P. Ngo, C. Phan, K. Dekome and J. Dusl, "Ultra-wideband time-difference-of-arrival high resolution 3d proximity tracking system", in *IEEE/ION Position, Location and Navigation Symposium*, May 2010, pp. 37–43. DOI: 10.1109/PLANS.2010.5507198.

[38] Y. Wu, Q. Chaudhari and E. Serpedin, "Clock synchronization of wireless sensor networks", *IEEE Signal Processing Magazine*, vol. 28, no. 1, pp. 124–138, Jan. 2011, ISSN: 1053-5888. DOI: 10.1109/MSP.2010.938757.

[39] D. Allan, "Time and frequency (time-domain) characterization, estimation, and prediction of precision clocks and oscillators", *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control*, vol. 34, no. 6, pp. 647–654, Nov. 1987, ISSN: 0885-3010. DOI: 10.1109/T-UFFC.1987.26997.

[40] D. W. Allan, "Tutorial: Clock and clock systems performance measures", ALLAN'S TIME FOUNTAIN GREEN UT, Dec. 1995.

[41] D. Kirchner, "Two-way time transfer via communication satellites", *Proceedings of the IEEE*, vol. 79, no. 7, pp. 983–990, Jul. 1991, ISSN: 0018-9219. DOI: 10.1109/5.84975.

[42] (). GEYER-KXO-84-VCTCXO-HCMOS datasheet, [Online]. Available: https://www.geyer-electronic.de/uploads/tx_userartikelfrequenz/GEYER-KXO-84-VCTCXO-HCMOS.pdf (visited on 03/08/2019).

[43] (). FreeRTOS - market leading RTOS (real time operating system) for embedded systems with internet of things extensions, FreeRTOS, [Online]. Available: /index.html (visited on 11/08/2019).

[44] Decawave. (). Dw1000_user_manual_2.11.pdf, [Online]. Available: https://www.decawave.com/sites/default/files/resources/dw1000_user_manual_2.11.pdf (visited on 24/06/2019).

Acronyms

CAP Contention Access Period. 25

CDF Cumulative Distribution Function. 38, 39

CFP Contention Free Period. 25

CSV Comma Separated File. 26

GPIO General Purpose Input/Output. 24

GPS Global Positioning System. 1, 11

IPS Indoor Positioning System. iii, 1, 2, 6, 8, 9, 36, 41, 49, 51

MAC Medium Access Control. 6

MAE Mean Absolute Error. iii, 38, 39, 49

PCB Printed Circuit Board. 9

PI Proportional-Integral. 16, 18

PID Proportional-Integral-Differential. 16, 18

PII Proportional-Integral-Integral. 16, 18

PLL Phase Locked Loop. 23

PPM Parts per million. 9, 16, 17, 23

RSS Received Signal Strength. 2, 3

SDS-TWR Symmetrical Double Sided Two-way Ranging. 34–36

SPI Serial Peripheral Interface. 23

TCXO Temperature-compensated crystal oscillator. 16

TDMA Time Division Multiple Access. 13, 25, 26, 36, 51

TDoA Time Difference of Arrival. iii, v, vi, 5–9, 11–13, 15, 21, 34, 36, 41, 43–45, 49, 50, 65

TOA Time Of Arrival. 5, 9, 16–18, 21, 22, 29–32, 36, 37, 44, 50, 65

ToF Time of Flight. 2–4

TWR Two-way Ranging. iii, 3, 4, 6–8

TWTT Two Way Time Transfer. 14, 15, 33, 34

USB Universal Serial Bus. 23, 24, 26, 36

UWB Ultra-wideband. iii, v, 1–3, 8, 9, 11, 13, 15, 18, 21–26, 29–31, 34, 35, 49, 51

VCTCXO Voltage-controlled temperature-compensated crystal oscillator. 17

Appendix A

Mathematical framework for UWB localisation

A model of the UWB phy and mac was created to evaluate different localisation methods in different mac protocols. An overview of the model is shown in Figure A.1. In this appendix only the results of the different localisation methods are shown.

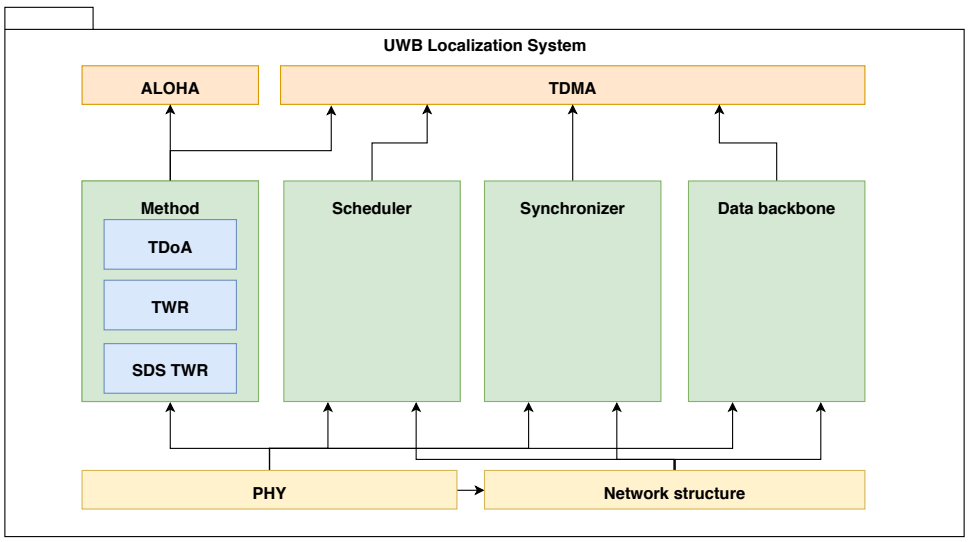


Figure A.1: *Overview of the mathematical model for UWB localization.*

A.1 PHY Layer

First we define a class that represents the PHY layer of UWB. According to the 802.15.4 specification a data frame of the phy is structured as shown below.

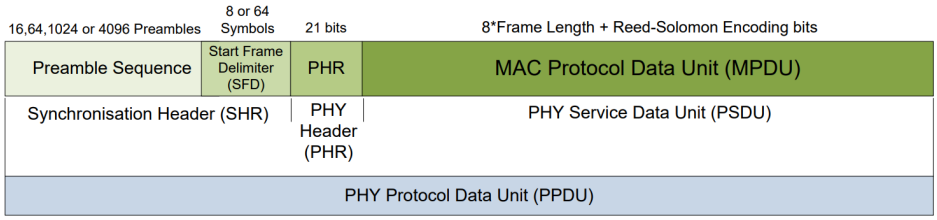


Figure A.2: *IEEE802.15.4-2011 PPDU Structure*

A.1.1 Model

Initialization of model

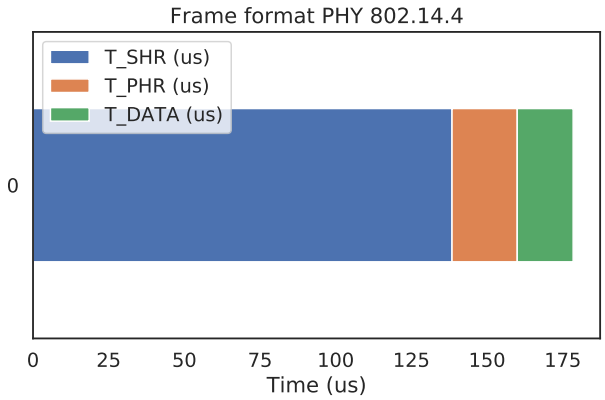
We can initilize the class with the following parameters (these parameter are comparable with the modules used in the development boards): - bitrate: 6.81 Mbps - Preamble 128 bits - PRF: 64 MHz - Payload: 8 bytes (to send its own Extende Unique Identifier(EUI))

In the decawave datasheet we can find the following table:

PRF (MHz)	Data Rate (Mbps)	t_SHR (ns)	t_PHR (ns)	t_DATA (ns)
16	0.11	993.59	8205.13	8205.13
16	0.85	993.59	1025.64	1025.64
16	6.81	993.59	1025.64	128.21
64	0.11	1017.63	8205.13	8205.13
64	0.85	1017.63	1025.64	1025.64
64	6.81	1017.63	1025.64	128.21

Furthermore we know that the start of frame delimiter is length is 64 bits if the bitrate is 110kbps and is 8 bits for every other bit rate.

```
   T_SHR (us)  T_PHR (us)  T_DATA (us)
0   155.69739    21.53844     18.46224
```



Results

The following things can be observed from the simulation

- The data part of the frame is very small compared to the synchronization header, so tags can send their complete EUI. No need to use a shorter id which will be mapped to their actual EUI.

A.1.2 Localization method

with the following classes different localization methods can be simulated. The following methods have been implemented:

- Time Difference Of Arrival
- Two Way Ranging (TWR)
- Symmetrical Double Sided Two Way Ranging (SDS-TWR)
- Optimized version of TWR where a broadcast is used to replace a message to every anchor

The models can simulate the following properties of a method:

- Calculate how long a location measurement will take, based on the number of anchors if it has a dependancy on it
- Calculate how long a tag uses its radio in order to tak a location measurement.

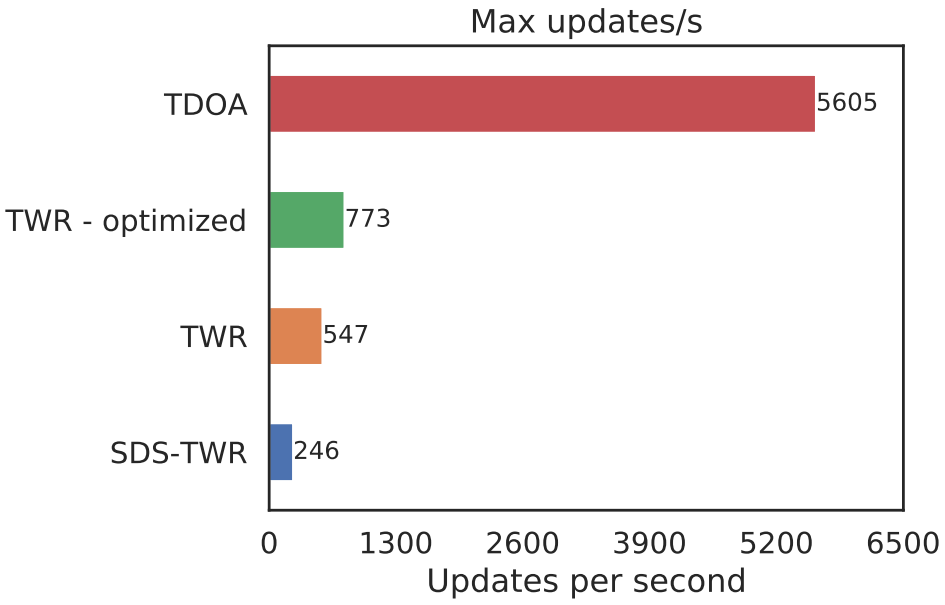
Initialization of localization method

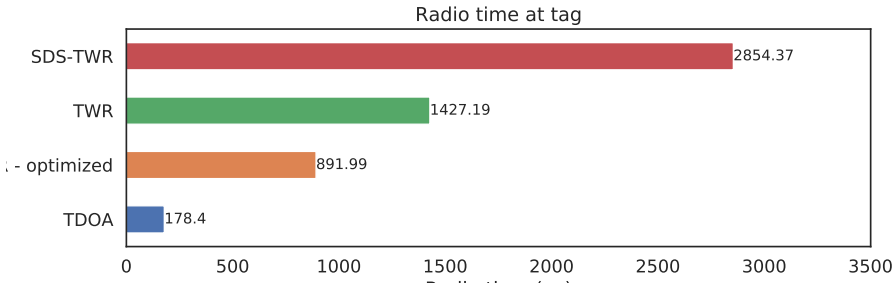
Below all methods are initialized using the phy layer we created before. For this simulation it was chosen that the TWR methods perform ranging with four anchors. And the Anchors need 100 micro seconds to respond to a message.

	Max updates/s	Localization time(s)	Radio time tag (s)
TDOA	5109	0.000196	0.000196
TWR - optimized	725	0.001378	0.000978
TWR	508	0.001966	0.001566
SDS-TWR	230	0.004331	0.003131

Results

From the simulation of the different methods two aspects of the methods can be evaluated. One aspect is the scalability of the method in terms of location measurements per second. The other aspect is the time the radio of the tag is used. This is strongly connected with the energy consumption of the tag, since the radio will be the component of the tag that has the highest energy consumption.


Max updates/s
TDOA
TWR - optimized
TWR
SDS-TWR
5605
773
547
246
0
1300
2600
3900
5200
6500
Updates per second


Radio time at tag
SDS-TWR
TWR
- optimized
TDOA
2854.37
1427.19
891.99
178.4
0
500
1000
1500
2000
2500
3000
3500

Appendix B

Implementation of Chan's algorithm

As explained in Section 2.1.2 Chan's algorithm uses a weighted least squares method to solve the TDoA equations. As input the algorithm requires a matrix containing all the positions of the anchors and the TOA of the message from the tag. This matrix is sorted according to the TOA. The TOA of the first anchor is subtracted form the TOAs and multiplied with the propagation speed in order to get the distance difference with respect to *anchor* 1. For four anchors the matrix looks like:

$$TOA = \begin{bmatrix} x_1 & y_1 & z_1 & c\tau_{11} \\ x_2 & y_2 & z_2 & c\tau_{21} \\ x_3 & y_3 & z_3 & c\tau_{31} \\ x_4 & y_4 & z_4 & c\tau_{41} \end{bmatrix} \tag{B.1}$$

Now the first row is subtracted from all rows so that the positions of the anchors become relative to the position of *anchor* 1. The systems of equation that needs to be satisfied is:

$$\boldsymbol{G}_1 \boldsymbol{P}_1 = \boldsymbol{h}_1 \tag{B.2}$$

where

$$\boldsymbol{G}_1 = 2 \begin{bmatrix} x_2 & y_2 & z_2 & c\tau_{21} \\ x_3 & y_3 & z_3 & c\tau_{31} \\ x_4 & y_4 & z_4 & c\tau_{41} \end{bmatrix}, \quad \boldsymbol{P}_1 = \begin{bmatrix} x \\ y \\ z \\ d_1 \end{bmatrix}, \quad \boldsymbol{h}_1 = \begin{bmatrix} x_2^2 + y_2^2 + z_2^2 - (c\tau_{21})^2 \\ x_3^2 + y_3^2 + z_3^2 - (c\tau_{31})^2 \\ x_4^2 + y_4^2 + z_4^2 - (c\tau_{41})^2 \end{bmatrix}$$

d_1 can be treated as a dummy variable for $d_1^2 = x^2 + y^2 + z^2$

The first weighted least square step is the following:

$$\boldsymbol{P}_{11} = (\boldsymbol{G}_1^T \boldsymbol{Q}^{-1} \boldsymbol{G}_1)^{-1} \boldsymbol{G}_1^T \boldsymbol{Q}^{-1} \boldsymbol{h}_1 \tag{B.3}$$

where $\boldsymbol{Q}$ is the covariance matrix. Then the result is further refined with:

$$\boldsymbol{P}_1 = (\boldsymbol{G}_1^T \boldsymbol{W}_1 \boldsymbol{G}_1)^{-1} \boldsymbol{G}_1^T \boldsymbol{W}_1 \boldsymbol{h}_1 \tag{B.4}$$

where

$$\boldsymbol{B}_1 = \begin{bmatrix} d_2 & 0 & 0 \\ 0 & d_3 & 0 \\ 0 & 0 & d_4 \end{bmatrix}, \quad \boldsymbol{W}_1 = \frac{1}{4c^2} \boldsymbol{B}_1^{-1} \boldsymbol{Q}^{-1} \boldsymbol{B}_1^{-1}$$

for matrix $\boldsymbol{B}_1$ the distance between the tag has to be known, which is to be calculated by the algorithm. So to calculate $d_2..d_4$ the estimation from Equation (B.3) is used.

In stage two of the algorithm the equation $d_1^2 = x^2 + y^2 + z^2$ is satisfied. The following equation must hold:

$$\boldsymbol{G}_2 \boldsymbol{P}_2 = \boldsymbol{h}_2 \tag{B.5}$$

where

$$\boldsymbol{G}_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \quad \boldsymbol{P}_2 = \begin{bmatrix} x^2 \\ y^2 \\ z^2 \end{bmatrix}, \quad \boldsymbol{h}_2 = \begin{bmatrix} P_1(1)^2 \\ P_1(2)^2 \\ P_1(3)^2 \\ P_1(4)^2 \end{bmatrix}$$

The least squares solution of this equation is represented by the following equation:

$$\boldsymbol{P}_2 = (\boldsymbol{G}_2^T \boldsymbol{W}_2 \boldsymbol{G}_2)^{-1} \boldsymbol{G}_2^T \boldsymbol{W}_2 \boldsymbol{h}_2 \tag{B.6}$$

where

$$\boldsymbol{B}_2 = \begin{bmatrix} P_1(1) & 0 & 0 & 0 \\ 0 & P_1(2) & 0 & 0 \\ 0 & 0 & P_1(3) & 0 \\ 0 & 0 & 0 & P_1(4) \end{bmatrix}, \quad \boldsymbol{W}_2 = \frac{1}{4} \boldsymbol{B}_2^{-1} \boldsymbol{G}_1^T \boldsymbol{W}_1 \boldsymbol{G}_1 \boldsymbol{B}_2^{-1}$$

At last the final solution for the position relative to *anchor* 0 is:

$$\boldsymbol{P} = \boldsymbol{S} \sqrt{\boldsymbol{P}_2} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{B.7}$$

where

$$\boldsymbol{S} = \begin{bmatrix} sign[\boldsymbol{P}_1(1)] & 0 & 0 \\ 0 & sign[\boldsymbol{P}_1(2)] & 0 \\ 0 & 0 & sign[\boldsymbol{P}_1(3)] \end{bmatrix}$$