Elaboration of Methods and Algorithms for Passive Aircraft and Vehicle Detection over Time of Signal Arrival Differences

Master's Thesis

realised by

Ulrich Feichter

at the

Institute for Broadband Communications (IBK)
at Graz University of Technology

Head of Institute: Prof. Dr. Gernot Kubin

in cooperation with

AviBit data processing GmbH

Adviser: Prof. Dr. Erich Leitgeb
Supervisor: Dr. Konrad Köck (AviBit)

Graz, February 10, 2010

Abstract

This thesis presents a methodology to determine the positions of transmitter equipped aircraft and vehicles, solely over time of signal arrival (TOA) measurements. For this purpose, the target must send out a signal which is received at a set of three or more base stations, distributed over the area of interest. Since the developed system measures TOA values with high accuracy and base station positions are known, the location of signal emission can be determined over a so called hyperbolic navigation, or Multilateration (MLAT) calculation.

The work presents a calculation method that firstly calculates algebraic solutions, followed by an iterative accuracy enhancement, which takes advantage of redundant measurements. Furthermore, optional methods for solution ambiguity resolution, measurement outlier detection and solution error estimation are explained. In addition to the presented algorithms and techniques, the work is extended by results derived from real field test data.

Keywords: Multilateration, MLAT, Hyperbolic Navigation, Secondary Surveillance Radar, SSR, Robust Adjustment, Outlier Detection

Kurzfassung

Diese Arbeit präsentiert eine Methode zur Ortung von Flug- und Fahrzeugen, welche mit einem Sender oder Transponder ausgestattet sind. Dazu sind mehrere Empfänger notwendig, die über die zu überwachende Fläche verteilt aufgestellt werden und deren Position bekannt ist. Die Empfänger messen ein ausgesendetes Signal und bestimmen dessen Ankunftszeit mit hoher Genauigkeit. Anschließend kann die Position des Senders an einer zentralen Stelle, über eine so genannte Multilaterationsberechnung, bestimmt werden. Dabei werden die Ankunftszeitmessungen aller Empfänger verarbeitet.

Die hier präsentierte Vorgehensweise für diese Berechnung ermittelt zuerst mathematisch exakte Lösungen. In einem zweiten Schritt werden diese Primärlösungen, unter Zuhilfenahme von redundanten Messungen, verfeinert. Außerdem werden Verfahren zur Ausreißerdetektion, Positionsfehlerbestimmung und zur Auflösung von Lösungsmehrdeutigkeiten präsentiert. Am Ende der Arbeit wird die Theorie mit Berechnungsergebnissen vervollständigt, welche aus Daten ermittelt wurden, die von einem Feldversuch stammen.

Schlüsselwörter: Multilateration, MLAT, Hyperbelnavigation, Sekundärradar, SSR, Ausgleichung nach Parametern, robuste Ausgleichung, Ausreißerdetektion

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

....	....
(date / Datum)	Ulrich Feichter

Acknowledgements

First of all I would like to thank my parents and my sister for the motivation and support throughout my whole life, which helped me a lot to take my own course, to progress my studies and to finally complete them with this thesis.

Thanks also to the company AviBit for providing me equipment and time during process development and the creation of this work. Special thanks I dedicate to my workmates, especially to Stephan Bernhart and Alex Randeu, for supporting me with feedback and for giving me inspirations in some endless discussions and meetings. Furthermore, I want to thank all the people who were assisting me during the field tests and helped me carrying around the heavy measurement equipment. I also want to thank Erich Leitgeb for supervising this work and for inspiring me in his courses.

Finally: Thanks to my friends for all the fun and great time we had in Graz, Stockholm, Südtirol and the rest of the world! It was you providing me the energy, motivation and necessary distraction to get ahead with my studies that well. Special thanks also to my longstanding flatmate Alex, for the awesome years of study we have spent together and for proofreading this work.

Last but not least I would like to thank all the programmers of the software projects GNU Octave, Gnuplot, Inkskape, TEXlive, GNU Emacs, Kate and xdvi for enabling me to create this work primarily with free and open source software.

Ulrich Feichter

Graz, Austria, February 10, 2010

Contents

List of Figures

List of Tables

1 Introduction and Motivation

This thesis presents a technique for aircraft and vehicle detection, which emerged out of Radar technology during the last decades. This new position finding method, called Multilateration (MLAT), follows a different approach than Radar, as it uses several distributed receiving units. In order to facilitate the understanding of the principle and the advantages of this new method, the following sub-chapter gives a short overview of Radar history. Subsequently, the basic principle of MLAT is explained, followed by a motivation for this topic.

1.1 Conventional Radar

Since the early 1930s Radio Detection and Ranging (RADAR) systems were developed by several countries to determine aircraft, ship and vehicle positions. Huge research in this field was performed before and during World War II, since the position of friendly and enemy aircraft was a valuable information [1]. Whereas Radar systems were just used for military purposes in the beginning, after the end of World War II Radar technologies found their way into civil air traffic control [2].

The basic principle of Radar is simple: A transceiver sends out an electromagnetic high frequency signal pulse, which propagates to an object. As soon as the pulse reaches the object's surface, a fraction of the signal power is reflected. Depending on the surface's geometry and material, the reflected signal, also called echo, is scattered to different directions. If a part of this reflection propagates back to the point of emission, the Radar's receiving unit measures it there. Electromagnetic waves propagate at constant speed (approximately speed of light) and so, when knowing the time the signal took to propagate to the object and back, it is possible to calculate the distance between transceiver and object. The calculation of the transceiver to object distance, also called range r, is simple:

$$r = \frac{c \cdot t}{2} \tag{1.1}$$

Here c denotes the speed of light(1) and t the round trip time, i.e. the time between pulse emission and echo reception. However, in order to get the full position information, also the angular direction in which the object is located has to be known. This so-called azimuth φ is determined by knowing the direction the interrogation signal was sent to. For this purpose, highly directional antennas are used to send the interrogation pulses and also to receive just echoes from the expected direction [4]. In order to cover the area of interest the main lobe of the antenna needs to sweep over the whole area while constantly sending interrogation pulses.

These so-called Primary Surveillance Radar (PSR) systems which work with passive reflections are able to detect objects which reflect the detection signal well enough in order to be measured at the receiver. As a big part of the signal power is absorbed during the reflection, the power of the measured echoes is generally much lower than the power of the transmitted pulse.

(1)Speed of light: $c = 299\,792\,458\,ms^{-1}$ from [3]

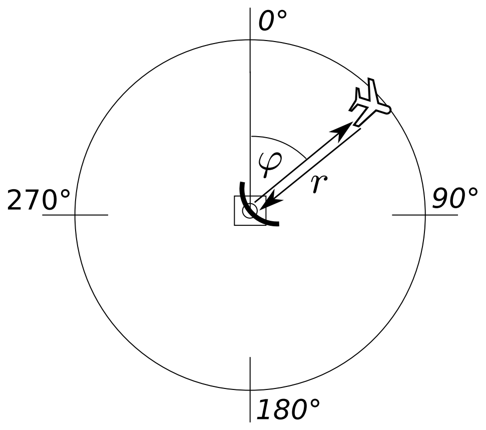


Figure 1.1: Radar principle

During the Second World War it became important to be able to differentiate between friendly and enemy aircraft, and so the system Identification Friend or Foe (IFF) was developed by the Allies. In IFF, friendly aircraft were supplied with equipment which was detecting the Allies radar transmissions and answering to them, in order to indicate that the aircraft was a friend [1]. Out of the IFF technology the civil Secondary Surveillance Radar (SSR) technology has emerged in the 1940s and is still in use today. In SSR, aircraft are equipped with active transponders which receive the radar's interrogation signal and answer to it after a delay time of $3\mu s$ [1, p. 176].

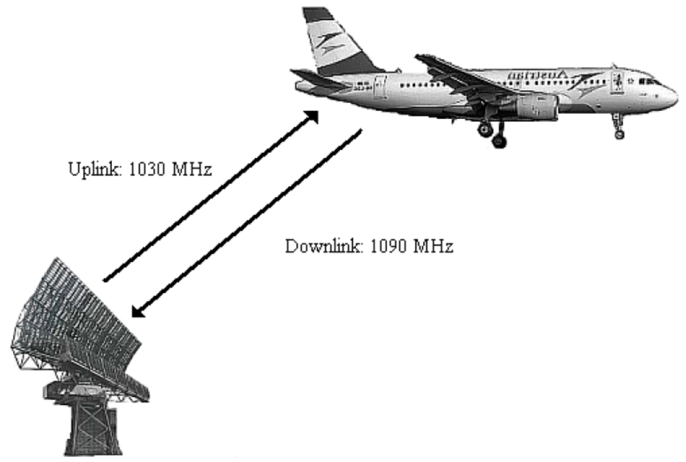


Figure 1.2: SSR principle taken from [5]

SSR radar technology made it possible to use much less transmitting power at the radar site: The detection signal does not have to travel to the object to be detected and back, but just one way to the transponder. The transponder then emits a new signal that propagates the same distance. Furthermore instead of weak signal reflections active transponder replies are detected at the site [4]. SSR technology enabled aircraft also to transmit information like identification or barometric height back to the interrogating station (see Chapter 2.6). The disadvantage of such systems is, however, that just cooperative targets, i.e. objects equipped with transponders, can be detected. However, nowadays almost every aircraft has to be equipped with a SSR transponder (see [6]). In aviation, En-Route and Airport Surveillance Radar (ASR) systems mostly use a combination of PSR and SSR in order to detect aircraft and to identify them over SSR [7, p. 173].

This conventional Radar technology, however, faces some problems: Firstly, directional antennas are needed which either are turned around or can change their directionality. In both cases the costs for these antennas are high and update rates are limited to 4 to 12 seconds, as antennas typically rotate at this speed [8]. Additionally, Radar depends on the availability of Line of Sight (LOS) propagation, and so parts of the area to be observed can be shadowed by terrain obstacles (see [8]).

1.2 Multilateration

Nowadays, thanks to the available computational power, a new type of radio detection systems is in use: This technology is known under the terminologies Hyperbolic Navigation or Multilateration (MLAT). In contrast to Radar, such systems consist of several ground stations which are distributed over the area of interest. These ground stations, also called Remote Units (RUs), listen for SSR replies which are emitted by aircraft or vehicle transponders in the area around them. If a transponder emits a reply, it is received at each Remote Unit at a different time, depending on the distance between transponder and RU. Out of this time of signal arrival (TOA) information from three or more stations, it is possible to calculate back to the point of signal emission. The calculation task is performed by the so-called Target Processor (TP) which receives the TOA measurements from all Remote Units over a communication interface. Figure 1.3 shows such a scenario:

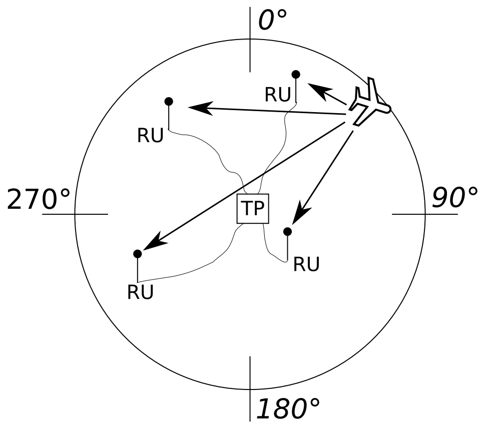


Figure 1.3: MLAT principle

The big difference between MLAT and Radar systems is the distributed system architecture. There is no more need for a big directional antenna, on a tower: The Remote Units which can be mounted on masts, are equipped with omnidirectional bar antennas and basically just need a communication interface and a power supply. The TP fits into a conventional server rack that can be located on an arbitrary place, if necessary also outside the system's area of operation. According to [8], Multilateration systems have the following advantages in comparison to conventional Radars:

- **Higher update rate:** As Radars need turning antennas, their update rate is restricted to 4 to 12 seconds. This is not the case in MLAT as the whole area is monitored at once.
- **In system redundancy:** Multilateration systems normally own more than the minimum amount of base stations. So the failure of a single base station can be recovered by other stations.
- **High system scalability:** The principle can be used for airport area aircraft and vehicle monitoring, or for wide area, with a coverage that exceeds conventional SSR.
- **Lower acquisition and maintenance costs** (see also [9])

MLAT systems can be realised in many ways, so it is important to categorise them. As accurate time measurement is essential in MLAT systems, two types of systems can be defined (see [10]):

Common Clock Systems measure time with high accuracy at one central position, normally at the Target Processor. The Remote Units may be constructed simple, as the system's complexity lies at one central point. However, it is necessary that the propagation delay from each RU's receiving antenna to the point of time measurement is constant and well known. This is achieved by installing expensive communication systems between RUs and TP, such as fiver optic or microwave links.

Distributed Clock Systems decode the replies and timestamp them directly at every Remote Unit. As the communication latency does not matter, this information is then sent to the target-processing unit over a conventional communication link like Ethernet. The disadvantage of such systems is that every Remote Unit has its own timebase, and so the units have to be synchronised.

Since laying of dedicated fiver on airports or microwave links are expensive and difficult to implement, the prototype system, developed in the course of this thesis, forms a distributed clock system. The title of this thesis already states that the system shall operate passively further. The difference between active and passive MLAT systems is the following:

Active systems are equipped with an interrogator unit which is capable to send SSR interrogations. Such systems are able to detect all transponders in their reach as they can interrogate them.

Passive systems just evaluate SSR signals which are available anyway in the area of interest. Therefore, they are dependent on other SSR systems which perform the interrogation work. However, no sending license is required to operate such passive systems.

1.3 Motivation

This master's thesis concerns the core element of a Multilateration system: The Target Processor or calculation unit. This unit receives time of signal arrival (TOA) measurements from Remote Units and performs the calculations in order to determine the position of signal emission. The thesis was written in the course of a Multilateration system prototype development, performed by the company AviBit Gmbh in Graz, Austria, in cooperation with the Institute for Broadband Communication (IBK) at Graz University of Technology. The system is planned to be a passive Multilateration system with distributed time measurement, capable of calculating 2D and 3D position fixes. As long-term objective of the Multilateration project, AviBit aims to collect the necessary expertise, in order to equip airports with fully operational MLAT systems. The Target Processor, implemented by the author along with this thesis, represents one step in this progress.

This work explains the mathematical basics of the Multilateration calculation first: Chapter 2 gives formulae and algorithms which are necessary to solve the hyperbolic positioning problem. Furthermore a small overview over the SSR radar technology is given, explaining some features which are essential for the developed system. Chapter 3 discusses design issues of a MLAT calculation unit and explains how the theory of the previous chapters can be applied in order to build a flexible and real time capable MLAT Target Processor. The actual prototype implementation is presented in Chapter 4. Furthermore, that chapter contains calculation results in order to prove and explain some implementation details and also some principles which were firstly just handled theoretically. Those results were obtained from data, which was accumulated in a real field test in the ORF Park in Graz, Austria. Finally, Chapter 5 gives an evaluation of single measurements performed in the field test, investigates errors and discusses possible system improvements.

2 Theoretical Background

This chapter tries to explain the theoretical principles which build the base for the Multilateration calculation. It starts with a chapter which explains the problem of hyperbolic positioning itself and presents algorithms to solve it in the subsequent chapters. Additionally, some fundamental information about SSR is provided, as the prototype developed in course of this work is working with Secondary Surveillance Radar reply signals.

The theory about the MLAT calculation was mainly accumulated over papers, publications and Internet sources, as it was not possible to find any books about Multilateration theory. The reason for this may lie in the topic's similarity to Global Navigation Satellite Systems (GNSSs). Y. E. Yang states the following in [11]:

> *"The working principle of a Multilateration system is similar to that of the Global Positioning System (GPS), except that instead of broadcasting timing signals, the sensors match up reply messages from target aircraft or mobile transponders, determine their time difference of arrival (TDOA) to derive the target positions."*

Due to this similarity, two books about GPS ([12] and [3]), written by B. Hofmann-Wellenhof, were used to elaborate the theory in the subsequent chapters. Furthermore, some books and scripts from the fields of avionics and geodesy were consulted.

2.1 The Problem of Hyperbolic Positioning

Hyperbolic positioning systems are used to determine the geometrical position of a high frequency signal transmitter. In order to be able to calculate this position the signal has to be received at three or more geometrically diverse located base stations. This chapter explains the principle that lies behind such a positioning calculation in order to motivate the theoretical basics explained in the following chapters.

To keep things easy we make the following assumptions in the subsequent explanations: Firstly, the signal emitted by a transmitter propagates to all directions in the same way; i.e. an omni-directional antenna is used. Secondly, we assume that the signal can propagate to the receivers without being blocked by obstacles, i.e. we have Line of Sight (LOS) communication.

The scenario is the following: A signal is generated by the transmitter, leaves the transmitting antenna, propagates to the base stations and reaches them at different times. The receiving time is denoted as time of signal arrival (TOA) and the time the signal was sent out as time of signal emission (TOE) in the future.

Such a scenario is illustrated in Figure 2.1: The mobile located at $(-20, 30)$ emits the signal which is received at the base stations S_1 to S_3. The distances between target and the i-th base station d_i result out of the geometry and are $d_1 = 42.43m$, $d_2 = 76.16m$ and $d_3 = 28.29m$. Assuming the signal is being emitted at the time $t_e = 0ns$, it will be received by each station at $t_i = \frac{d_i}{c}$, where c denotes the speed of light. This leads to the following receiving times: $t_1 = 141.52ns$, $t_2 = 54.03ns$ and $t_3 = 94.35ns$.

If now both, the TOE and TOA, were known to the system, the positioning calculation would result in a normal triangulation calculation: The distance d_i between transmitter and

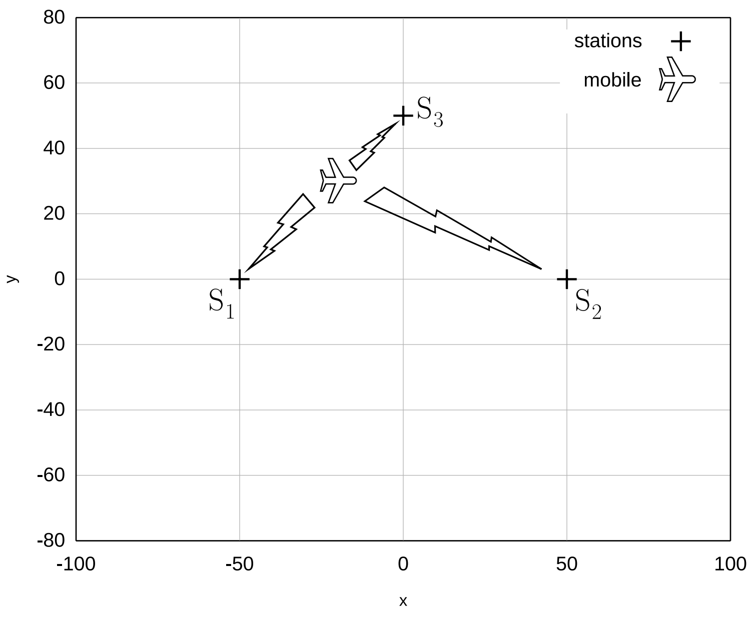


Figure 2.1: MLAT scenario for mobile at $(-20,\ 30)$ and receiving stations at $S_1 = (-50,\ 0)$, $S_2 = (50,\ 0)$ and $S_3 = (0,\ 50)$. Units are given in metres.

receiver i could be calculated simply via the equation

$$d_i = c\,(t_i - t_e) \tag{2.1}$$

where c is the speed of light, t_i the TOA at the station i and t_e the TOE. In the two dimensional case we can set up the following formula for each station, where the position of station i is denoted by x_i and y_i and the transmitter position is given by the coordinates x and y:

$$d_i = c\,(t_i - t_e) = \sqrt{(x_i - x)^2 + (y_i - y)^2} \tag{2.2}$$

As equation (2.2) gives a circle with centre $(x_i,\ y_i)$ and radius d_i, the problem can be solved by intersecting two of such circles, thus by knowing d_i for two different stations. Figure 2.2 shows such circles for the stations S_1 and S_2 in the previously discussed scenario. However, this intersection leads to an ambiguous solution: the correct solution $(-20,\ 30)$ and the wrong solution $(-20,\ -30)$. The wrong solution has to be eliminated in a second step, e.g. by respecting the measurement of the third station, which gives an additional circle.

Passive Multilateration systems, however, do not know at which time the signal was emitted. The terminology passive means here that the system is just evaluating signals that are anyway available in the area of interest, without communicating with the targets to be detected. The time of signal emission, which was essential in the previous calculation, can, however, only be determined if communication between system and target is possible e.g. by interrogating a SSR transponder. Therefore, another approach for the calculation has to be found which solely considers the TOA at different stations.

Useful information which can be retrieved from TOA measurements is the time difference of signal arrival (TDOA) for each pair of base stations. Starting from this TDOA, the

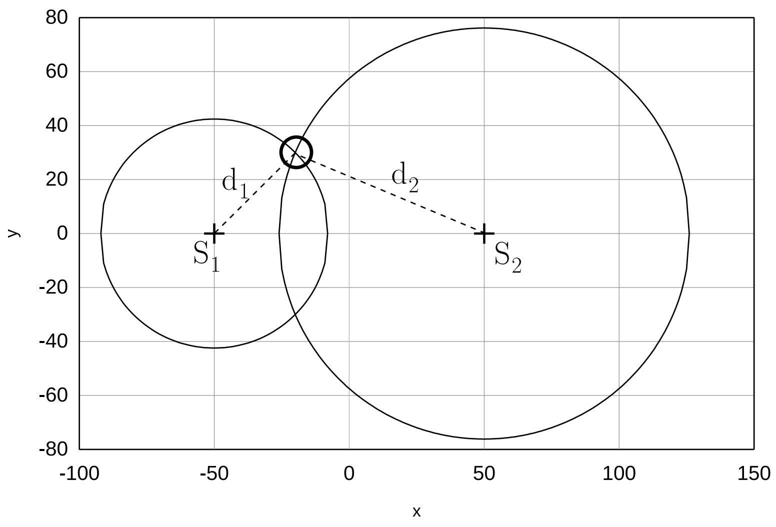


Figure 2.2: Intersecting circles with centres in $S_1 = (-50, 0)$ and $S_2 = (50, 0)$ and radii $d_1 = 42.43$ and $d_2 = 76.16$. Units are given in metres.

difference in target-station distance (Δd_{ij}) for the stations i and j can be calculated as

$$\Delta d_{ij} = d_i - d_j = c\,(t_i - t_j) \tag{2.3}$$

By inserting equation (2.2) into the above formula we come to the following equation, which describes a hyperbola with the focal points in the stations' positions $(x_i,\, y_i)$ and $(x_j,\, y_j)$:

$$\Delta d_{ij} = d_i - d_j = \sqrt{(x_i - x)^2 + (y_i - y)^2} - \sqrt{(x_j - x)^2 + (y_j - y)^2} \tag{2.4}$$

Figure 2.3 shows such a hyperbola for the stations S_1 and S_2 in the scenario discussed before. For every point P on the hyperbola the equation $|d_1 - d_2| = \Delta d_{ij}$ holds.

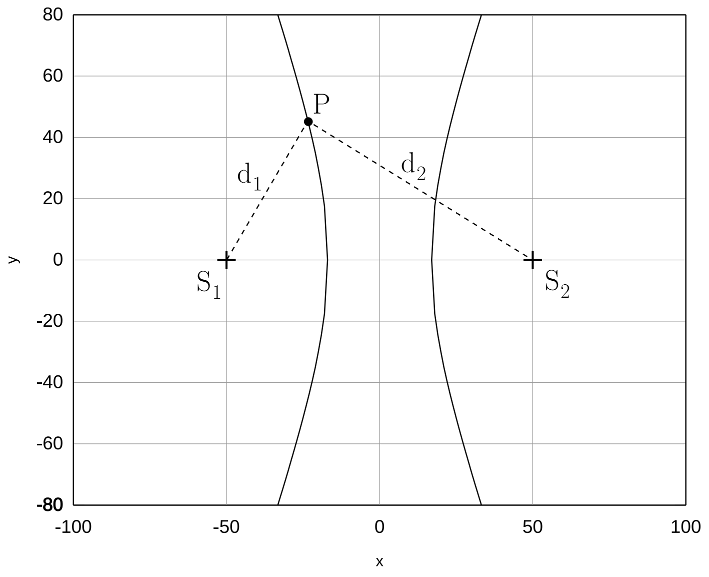


Figure 2.3: Hyperbola with focuses in $S_1 = (-50, 0)$ and $S_2 = (50, 0)$ and $|d_1 - d_2| = 33.73$. Units are given in metres.

This problem can newly be solved by intersecting more of these hyperbolas. If Figure 2.3 is supplemented by the hyperbolas that result out of base station S_3, the scenario looks like in Figure 2.4.

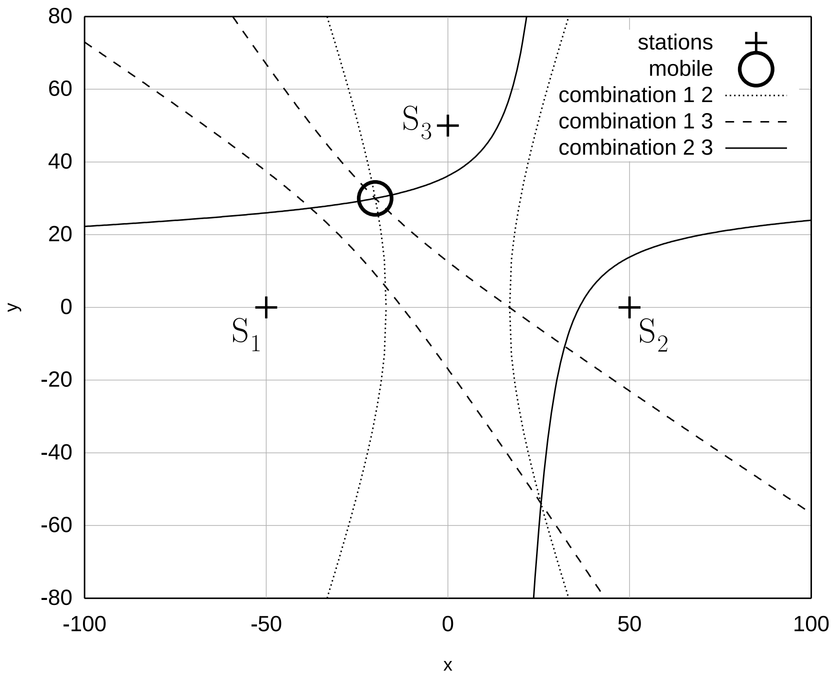


Figure 2.4: Intersecting hyperbolas for base stations $S_1 = (-50, 0)$ and $S_2 = (50, 0)$ and $S_3 = (0, 50)$ target at $(-20, 30)$. Units are given in metres.

Here the solution is ambiguous as well: The three hyperbolas intersect in the correct position $(-20, 30)$ but also in $(25.65, -53.92)$. This ambiguity has to be resolved in a later step; some convenient ways to do so are presented in Chapter 2.2.2. Of course, the intersection points can also be determined mathematically by arranging more hyperbola equations in a system of equations. Chapter 2.2 explains how such a solution can be determined.

As seen in this example, it is necessary to have three TOA measurements in order to calculate a two dimensional solution. If the problem is expanded to three dimensions, the hyperbolas turn into hyperboloids which are three-dimensionally analogical to the hyperbola. The calculation again intersects the hyperboloids; however, since a third position coordinate is determined also an additional measurement is needed. So a three dimensional calculation can be performed if at least four measurements from different Remote Units are available.

These three or respectively four measurements are the minimum amount of measurements which are needed to come to a solution. Unfortunately, TOA measurements are never exact in reality: In fact, they suffer from measurement errors like noise, distortions during wave propagation, multipath propagation or shadowing. This leads to measurement inaccuracies and to missing measurements. Of course, a position calculated with such error-affected measurements leads to a solution with less accuracy. Because of this, it makes sense to design hyperbolic positioning systems in a way, so that more than just the minimum number of base stations receive the signals from targets in the area of interest. Furthermore, this increases the in-system redundancy and gives more freedom in the positioning calculation. Due to the additional measurements, the system of equations, which describes the intersecting hyperbolas, becomes overdetermined and this redundancy can be used to get more accurate solutions, as shown in Chapter 2.3. Additionally overdetermination can also be used to resolve solution ambiguities (see Chap. 3.4.3) and to estimate the solution accuracy (see Chap. 2.5).

The subsequent chapters will explain the calculation for the overdetermined and non-overdetermined case. As it is easier to explain and understand the theory in two dimensions, all algorithms are explained for the two dimensional case for the rest of this theoretical introduction. The step into the third dimension is mostly quite straightforward and is shown in Chapter 2.4.

2.2 Algorithms for Closed Solutions

The solutions presented in this section are mathematically exact, which means that they use algebraic approaches to solve the problem of hyperbolic positioning. Basically, the problem can be defined by the following system of equations which is set up by writing down equation (2.2) for different stations:

$$\begin{aligned} c(t_1 - t_e) &= \sqrt{(x_1 - x)^2 + (y_1 - y)^2} \\ c(t_2 - t_e) &= \sqrt{(x_2 - x)^2 + (y_2 - y)^2} \\ c(t_3 - t_e) &= \sqrt{(x_3 - x)^2 + (y_3 - y)^2} \end{aligned} \tag{2.5}$$

Since we have three unknowns in the two dimensional case (the coordinates x and y and the TOE t_e) we need three equations to solve the problem. Respectively, we require three TOA measurements (t_i) of three different base stations to come to a solution. If we expand the problem to the third dimension, we need one more measurement from another base station.

As the system is non-linear, it cannot be solved simply over e.g. a Gaussian elimination, but other strategies have to be used. The straightforward way, as presented in [13], would be to solve the system by using conventional mathematical strategies like insertion or substitution. However, there already exists a solution for this problem which is not that confusing, but clearly arranged in matrix and vector calculations. This solution which was initially developed for the Global Positioning System (GPS) is presented in the next chapter.

2.2.1 Bancroft's Algorithm

The algorithm presented here was initially published 1985 by Stephen Bancroft in [14] as an attempt to solve the system equations for the GPS in a closed form. The algorithm "[...] is algebraic and noniterative in nature, computationally efficient and numerically stable [...]", as Bancroft states in [14]. A good explanation of the algorithm's application to hyperbolic positioning systems is found in [15]. The latter paper builds also the base for the illustrations in this chapter.

The calculation starts with the system of equations shown in formula (2.5). First, we square each equation to get them into form (2.6) and the subsequent simplification leads us to the form shown in (2.7).

$$c^2(t_i - t_e)^2 = (x_i - x)^2 + (y_i - y)^2 \tag{2.6}$$

$$2(x_i x + y_i y - c^2 t_i t_e) = x^2 + y^2 - c^2 t_e^2 + x_i^2 + y_i^2 - c^2 t_i^2 \tag{2.7}$$

By arranging three equations of the form (2.7) to a system of equations, it is possible to write it down by using matrix and vector notation. In order to do this, we describe each station by vector $\vec{s_i}$ and the target to be located by vector $\vec{x}$. Each station vector consists of two Cartesian coordinates for the station's location (x_i, y_i) and the time of signal arrival (TOA) at the station (t_i). The target vector contains the unknown target coordinates (x, y) and the also unknown time of signal emission (t_e). In order to get all variables into the same scale and dimension, the time variables are multiplied by the speed of light (c) which leads to the so-called pseudo ranges r_i and r_e respectively.

$$\vec{s_i} = \begin{bmatrix} x_i \\ y_i \\ c\,t_i \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \\ r_i \end{bmatrix} \qquad \vec{x} = \begin{bmatrix} x \\ y \\ c\,t_e \end{bmatrix} = \begin{bmatrix} x \\ y \\ r_e \end{bmatrix} \tag{2.8}$$

In the following deviations, the Lorentzian inner product, denoted by $\langle \vec{u}, \vec{v} \rangle$, is used in order to simplify the mathematical representation.

$$\langle \vec{u}, \vec{v} \rangle = u_1 v_1 + u_2 v_2 - u_3 v_3 \qquad \vec{u} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}, \ \vec{v} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \tag{2.9}$$

By using these notations, we can write down the system of equations (2.7) as follows:

$$\boxed{2\mathbf{A}\vec{x} = \lambda\vec{1} + \vec{b}} \tag{2.10}$$

where

$$\mathbf{A} = \begin{bmatrix} x_1 & y_1 & -r_1 \\ x_2 & y_2 & -r_2 \\ x_3 & y_3 & -r_3 \end{bmatrix} \tag{2.11a}$$

$$\lambda = \langle \vec{x}, \vec{x} \rangle = x^2 + y^2 - r_e^2 \tag{2.11b}$$

$$\vec{1} = [1 \quad 1 \quad 1]^T \tag{2.11c}$$

$$\vec{b} = \begin{bmatrix} \langle \vec{s_1}, \vec{s_1} \rangle \\ \langle \vec{s_2}, \vec{s_2} \rangle \\ \langle \vec{s_3}, \vec{s_3} \rangle \end{bmatrix} = \begin{bmatrix} x_1^2 + y_1^2 - r_1^2 \\ x_2^2 + y_2^2 - r_2^2 \\ x_3^2 + y_3^2 - r_3^2 \end{bmatrix} \tag{2.11d}$$

Since our goal is to calculate the position $\vec{x}$ of the target, we multiply equation (2.10) on both sides by the inverse of $\mathbf{A}$:

$$\vec{x} = \frac{1}{2}\lambda\mathbf{A}^{-1}\,\vec{1} + \frac{1}{2}\mathbf{A}^{-1}\,\vec{b} \tag{2.12}$$

With some substitutions, we get to the simpler form:

$$\begin{aligned} \vec{x} &= \lambda\vec{d} + \vec{e} \qquad (2.13) \\ \vec{d} &= \frac{1}{2}\mathbf{A}^{-1}\vec{1} \\ \vec{e} &= \frac{1}{2}\mathbf{A}^{-1}\vec{b} \end{aligned}$$

By applying the Lorentzian inner product onto both sides of equation (2.13), and by following the derivation shown in Appendix A.1, we get to this quadratic equation in λ:

$$\lambda = \langle\vec{x},\vec{x}\rangle = \lambda^2\langle\vec{d},\vec{d}\rangle + 2\lambda\langle\vec{d},\vec{e}\rangle + \langle\vec{e},\vec{e}\rangle \qquad (2.14)$$

This equation is then brought into the normal form

$$\begin{aligned} \alpha\lambda^2 + \beta\lambda + \gamma &= 0 \qquad (2.15) \\ \alpha &= \langle\vec{d},\vec{d}\rangle \\ \beta &= 2\langle\vec{d},\vec{e}\rangle - 1 \\ \gamma &= \langle\vec{e},\vec{e}\rangle \end{aligned}$$

and finally the quadratic equation is solved, which leads to the ambiguous solution:

$$\lambda_{1/2} = \frac{1}{2\alpha}\left(-\beta \pm \sqrt{\beta^2 - 4\alpha\gamma}\right) \qquad (2.16)$$

The ambiguous target position $\vec{x}_{1/2}$ is then finally calculated by inserting both values of $\lambda_{1/2}$ into (2.13). As the solution received by the algorithm is mostly ambiguous, one of the two solutions is wrong and has to be identified in a later step (see next chapter). An example for an ambiguous solution can be seen in Figure 2.4: The true solution is the intersection of three hyperbolas marked with the circle, the wrong one is formed by the intersection at the point (25.65, −53.92).

2.2.2 Solution Ambiguity Resolution

As discussed in the previous chapters, closed solutions are mostly ambiguous because the problem is of second order. Fortunately, in many cases the wrong solution can be eliminated by applying some reasonable rules. Thus, it is possible to resolve the ambiguity without considering additional measurements, if at least one of the following conditions holds:

1. It is impossible that the signal's TOE lies after one measured TOA. So, if $r_e > r_i$ for at least one station i, the solution is wrong.

2. The chronological order of signal arrival for the determined position must be the same as the order in the measurements. In other words: The nearest station to the calculated solution must have received the signal first and so on.

3. Solutions in unreasonable locations can be neglected e.g. positions outside the area of interest, or, in case of three dimensions, height values below ground level.

These criteria mainly cover the so called “Indicators” listed in [15] and hold for two- and three-dimensional results. The first two conditions just have mathematical background and can be applied without making assumptions about the target’s location. They result of the fact that one branch of each hyperbola can always be neglected, as, due to the problem’s geometry, the signal must arrive first at the station which is nearer to the target position. This can be seen in Figure 2.3 on page 8, where d_1 is smaller than d_2, thus just the left branch of the hyperbola can form the true solution. If now both solutions are built by different hyperbola branches, as shown in Figure 2.4 on page 9, the correct solution can be identified by the rules 1 and 2 listed above.

However, there are also cases where both solutions are built by the same hyperbola branches. Such a case is shown in Figure 2.5 where the wrong solution at (90.7, −30.8) results of the intersection of the same hyperbolas as the correct solution in the circle does. Such cases appear mainly in areas referred to as “unfavourable regions” in [15], where Remote Units lie approximately on one line with the mobile, but the mobile is not located within the remote units (see Figure 2.6). In such areas conditions 1 and 2 will fail.

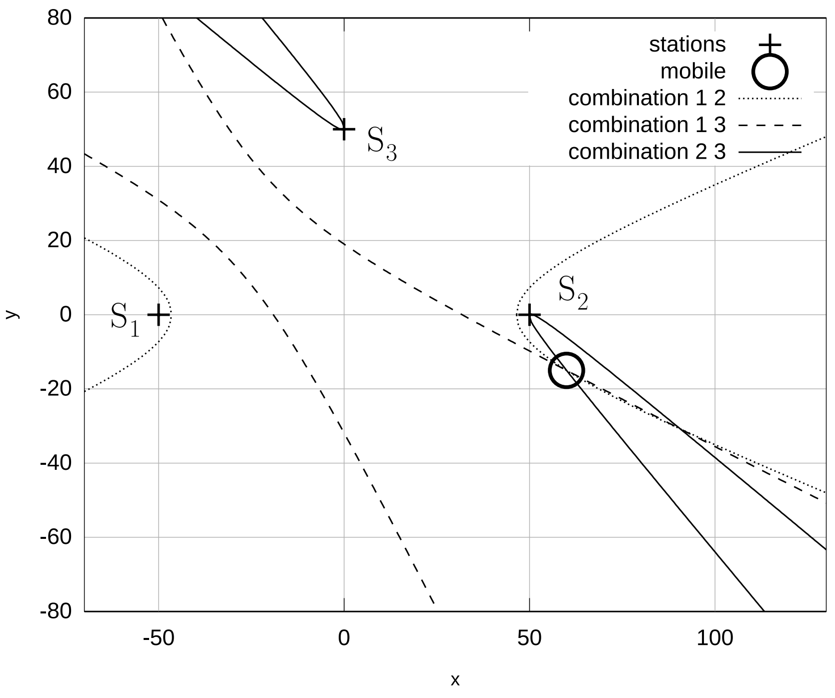


Figure 2.5: Same branches solution for mobile at (60, −15) and wrong ambiguous solution at (90.7, −30.8)

Condition 3 needs a preliminary assumption where the target is allowed to be located. This is normally the area of interest of the MLAT system, as targets outside this area are neglected anyway. In case of three-dimensional calculations also a minimum altitude level can be specified, as no target can be located below ground level.

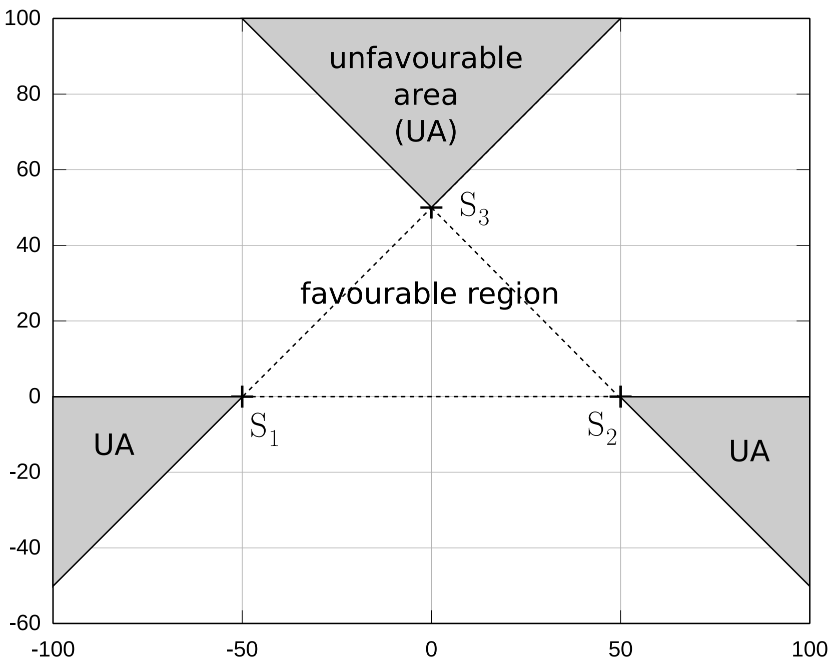


Figure 2.6: Unfavourable areas as stated in [15]

In general criteria 1 and 2 perform well in two dimensional scenarios, as there the most solutions are built by different hyperbola branches (see [15]). In the three-dimensional case ambiguous solutions often just differ in the height value, so the third condition is essential by specifying a lowest allowed height value. However, it has to be taken into account that the height value of the solution is mostly affected by big error scatter: Just strictly defining ground level as minimum height is dangerous, as, due to measurement inaccuracies, correct solutions can also necessarily lie some metres below ground level. The same also applies when specifying a tight area of interest. For further information about geometry dependence of the conditions, please refer to [15].

In some cases, solution ambiguity cannot be resolved by the conditions listed above , so the information of an additional measurement is needed. A method to handle this is shown in chapter 3.4.3 “Solution Quality”.

2.3 Iterative Solution

The previous section presented closed, mathematically exact solutions for the basic equation system (2.5). Exact solutions, however, have the disadvantage of always taking exactly as many input arguments as unknowns determined in the calculation. In our case this is three input arguments in the two dimensional case and four arguments if the calculation is done in three dimensions. Mostly, however, hyperbolic positioning systems are designed in a way that targets are “seen” by more than the minimum amount of base stations in the area of interest. This leads then to additional measurements and these can obviously be used to improve the solution accuracy. This section copes with the problem of involving these measurements into the calculation, i.e. to increase solution accuracy by taking advantage of the equation system’s overdetermination.

A first trivial attempt is to calculate one closed solution for all possible combinations of base stations and to average the results afterwards in some way. This attempt is however not effective because of two reasons: Firstly, the number of calculations grow heavily with the number of base stations. This is due to the fact that there are n choose k[(1)] possible input combinations, where n is the number of base-stations and k the number of unknowns. So if e.g. five measurements are available in a two dimensional scenario, already $\binom{5}{3} = 10$ closed solutions have to be calculated. Secondly, simulation results have shown that closed results from different input combinations are often affected by excessive scatter and averaging these does not lead to good results.

Consequently, the next chapter presents a calculation method which is known under the terminology “Adjustment by Parameters” and which is normally used in geodesy and Global Navigation Satellite System (GNSS) to improve calculation accuracy (see [17] and [3]). Furthermore, Chapter 2.3.2 presents how this calculation method can be used to perform outlier detection. All the theory in the following chapters is firstly presented in general and finally applied to the MLAT equations in Chapter 2.3.3.

2.3.1 Adjustment Theory

The theory presented in this chapter is one possibility to solve overdetermined systems of linear equations and is also known under the terminology “Least Squares Adjustment by Parameters”. As in most applications the system equations are not linear, also the possibility to operate with linearised system equations is shown. The content of this chapter was taken mainly from [12] and [3] and was fused with the theory in [17].

We start with the system equations in matrix notation:

$$\vec{m} = \mathbf{A}\vec{x} \tag{2.17}$$

$$\vec{m} = \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_n \end{bmatrix} \qquad \vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}$$

Vector $\vec{m}$ describes the n uncorrelated measurements which will be used to calculate the d unknown parameters $\vec{x}$. The n x d matrix $\mathbf{A}$ is the so-called system matrix, which describes the linear dependence of $\vec{m}$ on $\vec{x}$. Since we have more measurements than unknowns ($n > d$), the matrix $\mathbf{A}$ is not square and the problem cannot be solved over a simple matrix inversion.

In case all measurements were totally accurate, the optimum solution could already be determined by just solving the problem with a subset of d out of n available measurements, thus neglecting unnecessary measurements and working with a square system matrix. It would also not matter which measurements were taken to derive the solution. In real life, however, measurements are always afflicted by the measurement errors $\vec{\varepsilon}$ which make the

(1)Binomial coefficient $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ from [16] system inconsistent. So we can define the vector of correct measurements $\hat{\vec{m}}$ and the correct solution $\hat{\vec{x}}$ as follows:

$$\hat{\vec{m}} = \vec{m} - \vec{\varepsilon} = \mathbf{A}\,\hat{\vec{x}} \tag{2.18}$$

However, in reality the measurement errors are unknown, as well as the correct solution. Therefore, we define the following equation as the base of our calculation:

$$\vec{m} + \vec{v} = \mathbf{A}\,\vec{x} \tag{2.19}$$

Here $\vec{v}$ is called vector of residuals and $\vec{x}$ is the solution of the calculation. Our goal is now to get the term $\vec{m} + \vec{v}$ close to the correct measurements $\hat{\vec{m}}$, as we want to minimise the measurement error. This goal is fulfilled if residuals are found which are close to $\vec{v} = -\vec{\varepsilon}$.

In the following we assume the measurement error being uncorrelated and zero mean Gaussian distributed, so also the residuals should fit this distribution, as we want them to be close to $-\vec{\varepsilon}$. Hence we can write down the residuals' Probability Density Function (PDF) $f(\vec{v})$ as a multivariate Gaussian distribution like follows:

$$f(\vec{v}) = \frac{1}{\sqrt{(2\pi)^n \prod_{i=1}^{n} \sigma_i}} e^{-\frac{1}{2}\sum_{i=1}^{n} v_i^2/\sigma_i^2} = \frac{1}{\sqrt{(2\pi)^n \, det(\mathbf{\Sigma}(\vec{v}))}} e^{-\frac{1}{2}\vec{v}^T \mathbf{\Sigma}(\vec{v})^{-1}\vec{v}} \tag{2.20}$$

The values for σ_i^2 describe the variance of the i-th measurement error and since the measurement errors are uncorrelated, they form the diagonal elements of the covariance matrix

$$\mathbf{\Sigma}(\vec{v}) = \begin{bmatrix} \sigma_1^2 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \sigma_n^2 \end{bmatrix} \tag{2.21}$$

As mentioned before, the residuals should best fit this distribution and so we try to find an optimisation criterion for equation (2.19) which fulfils this. A good choice in this case is to maximise the residuals' likelihood function $L(\vec{v})$ which is equivalent to getting the exponent of the PDF in (2.20) close to zero. So we can write down:

$$L(\vec{v}) = \prod_{i=0}^{n} f(v_i) \rightarrow Max\,|_{\vec{v}} \quad \Leftrightarrow \quad \frac{1}{2}\sum_{i=1}^{n} \frac{v_i^2}{\sigma_i^2} = \frac{1}{2}\,\vec{v}^T \mathbf{\Sigma}(\vec{v})^{-1}\vec{v} \rightarrow Min\,|_{\vec{v}} \tag{2.22}$$

The measurements $\vec{m}$ do not necessarily have the same accuracy and so the variance of the measurement errors may differ from each other. That is why it makes sense to prioritise measurements with small measurement error and give bad measurements a lower priority in the calculation. To do so we define the priority matrix $\mathbf{P}$ as

$$\mathbf{P} = \begin{bmatrix} \frac{\sigma_m^2}{\sigma_1^2} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \frac{\sigma_m^2}{\sigma_n^2} \end{bmatrix} = \sigma_m^2\,\mathbf{\Sigma}(\vec{v})^{-1} \tag{2.23}$$

where each diagonal element p_i denotes the priority of the i-th measurement. σ_m^2 is the mean of the measurement errors σ_i^2 and is used to keep the priority values around unity. In case all measurements have the same accuracy, all σ_i^2 have the same value and the priorities will become one. By combining equations (2.22) and (2.23) we can finally define our optimisation criterion as

$$\boxed{\vec{v}^T \mathbf{P} \vec{v} = \sum_{i=1}^{n} p_i \, v_i^2 \rightarrow Min\,|_{\vec{v}} \qquad (2.24)}$$

This criterion basically describes the minimisation of the sum of squared residuals, prioritised by $\mathbf{P}$. Such an optimisation is often called Least Square Error Optimisation. If this criterion is applied to equation (2.19) we come to the following (see Appendix A.2):

$$\mathbf{A}^T \mathbf{P} \mathbf{A} \, \vec{x} - \mathbf{A}^T \mathbf{P} \, \vec{m} = 0 \tag{2.25}$$

By using the same naming conventions as in [12], we substitute the normal equation matrix $\mathbf{N} = \mathbf{A}^T \mathbf{P} \mathbf{A}$ and call its inverse $\mathbf{Q}$. Finally, by combining (2.19) and (2.25), we get the following set of equations for the solution:

$$\vec{x} = \mathbf{Q} \, \mathbf{A}^T \, \mathbf{P} \, \vec{m} \tag{2.26a}$$

$$\vec{v} = \mathbf{A} \, \vec{x} - \vec{m} \tag{2.26b}$$

$$\mathbf{Q} = \mathbf{N}^{-1} = (\mathbf{A}^T \, \mathbf{P} \, \mathbf{A})^{-1} \tag{2.26c}$$

The above listed formulae work with linear systems which are expressed by the system matrix $\mathbf{A}$. In order to be able to calculate an adjustment by parameters also on non-linear systems, the non-linear system equations $m_i(\vec{x})$ have to be linearised. We do so by calculating a Taylor expansion, neglecting all terms after the linear one (see [12] and [16]).

$$m_i(\vec{x}) = m_i(\vec{x}_a) + \frac{\partial m_i}{\partial \vec{x}}(\vec{x}_a) \cdot (\vec{x} - \vec{x}_a) \tag{2.27}$$

The linearisation is based in the point $\vec{x}_a$ which is given by an initial approximate solution. The expansion's constant part $m_i(\vec{x}_a)$ is derived from $\vec{x}_a$ over the non-linear system equation. $\frac{\partial m_i}{\partial \vec{x}}(\vec{x}_a)$ denotes the total derivation of the system equation with respect to $\vec{x}$ at the point $\vec{x_a}$. In order to simplify the subsequent calculations, we arrange the constant parts of the linearisation into the vector $\vec{m}(\vec{x}_a)$ and define:

$$\begin{aligned} d\vec{x} &= \vec{x} - \vec{x}_a \\ d\vec{m} &= \vec{m} - \vec{m}(\vec{x}_a) \end{aligned} \tag{2.28}$$

and we come to the following linearised analogy of (2.17):

$$d\vec{m} = \mathbf{A}\, d\vec{x} \tag{2.29}$$

$$\mathbf{A} = \begin{bmatrix} \frac{\partial m_1}{\partial x_1}(\vec{x}_a) & \frac{\partial m_1}{\partial x_2}(\vec{x}_a) & \cdots \\ \frac{\partial m_2}{\partial x_1}(\vec{x}_a) & \ddots & \\ \vdots & & \end{bmatrix}$$

The equations (2.26) finally become:

$$\boxed{\begin{aligned} d\vec{x} &= \mathbf{Q}\,\mathbf{A}^T\,\mathbf{P}\,d\vec{m} && \text{(2.30a)} \\ \vec{v} &= \mathbf{A}\,d\vec{x} - d\vec{m} && \text{(2.30b)} \\ \vec{x} &= \vec{x}_a + d\vec{x} && \text{(2.30c)} \end{aligned}}$$

This linearised system, however, is just a first approximation of the real system, thus the determined solution is not an optimum solution. In order to increase the accuracy of the outcome, linearisation and adjustment can be repeated iteratively: First, the system matrix is built up by using the initial $\vec{x}_a$. Then equations (2.30) are solved and the new calculated vector of unknowns $\vec{x}$ is taken as new approximate solution $\vec{x}_a$ for the next iteration step. It is also important that the initial approximate solution $\vec{x}_a$ is already quite close to the final solution, because otherwise the optimisation might get stuck in a local extremum.

It is interesting to know that the normal equation matrix **N** is symmetric: This is because the multiplication of a matrix with its transposed analogy results in a symmetric matrix. [17, p.17]. This enables **N** being inverted computationally more efficient, as e.g. a Cholesky decomposition can be performed (see [18, p. 60]).

2.3.2 Robust Adjustment

Least Square adjustment, as discussed in the previous chapter, gives an optimum solution if the measurement error follows a Gaussian distribution (see [17]). In fact, we can assume the measurement error to consist mainly of a Gaussian part, but also outliers, often also referred to as gross errors, may appear. These outliers may have different roots: no Line of Sight (LOS) propagation, synchronisation problems, inclusion of measurements which do not belong to the actual signal emission, etc. If measurements with such gross errors are Least Square adjusted, as shown in the previous chapter, the error in the result will be substantial, as outliers have the same influence on the outcome as every other measurement. Therefore, outliers need to be detected in order to exclude them from the calculation, or at least to give them less influence onto the result. This can be achieved by calculating a so-called robust adjustment as discussed in [17] and [19]. The book [17] provides a good and complete elaboration of this topic and the subsequent explanations were mainly taken from it; however, they were adapted to fit into the nomenclature used herein.

An effective possibility to make an adjustment robust against outliers is to use a Maximum

Likelihood estimator, also called M-Estimator. Such an estimator tries to choose the residuals $\vec{v}$ in a way that they best fit a given probability distribution by maximising a likelihood function. For a multivariate zero mean Gaussian distribution this was already done in the equations (2.22), found in the previous chapter.

In order to generalise M-Estimators we define the so called loss-function $\sigma(v_i)$ which describes the content of the sum which is intended to be minimised in the estimation. For the least square error case in the last chapter, and when neglecting the weighting of the measurements, this function becomes (compare to (2.22)):

$$\sigma(v_i) = \frac{1}{2} v_i^2 \tag{2.31}$$

If the the loss-function is changed, obviously also the probability distribution, for which the likelihood is maximised in the estimation, will vary. In our case we will model the measurement error as Gaussian distributed with possible outliers. This means that the Probability Density Function (PDF) has to follow mainly a normal distribution, but a few values far outside this distribution should also be allowed. A common M-estimator which fulfils these requirements is the Huber-k-Estimator with the loss-function $\sigma_H(v_i)$ and its derivative $\Psi_H(v_i)$ [17, p. 107]

$$\sigma_H(v_i) = \begin{cases} \frac{1}{2} v_i^2 & \text{if } |v_i| < k \\ k \cdot |v| - \frac{1}{2}k^2 & \text{if } |v_i| \geq k \end{cases} \tag{2.32}$$

$$\Psi_H(v_i) = \frac{\partial \sigma_H(v_i)}{\partial v_i} = \begin{cases} v_i & \text{if } |v_i| < k \\ k \cdot \frac{v_i}{|v_i|} & \text{if } |v_i| \geq k \end{cases} \tag{2.33}$$

For values of v in the interval $(-k, k)$ the Huber-k estimator behaves like an estimator for a Gaussian PDF with the square loss-function like in (2.31). Outside the interval but the loss-function is linear (see Figure 2.7). The value for k is normally chosen between $1.5\,\sigma_m$ and $2\,\sigma_m$, where σ_m is the expected a-priori measurement error standard deviation without outliers [19].

According to [17], a robust M-estimator adjustment can be calculated like a Least Square adjustment where the weight matrix, now called $\mathbf{W}$, is adapted iteratively in every adjustment step according to the formula (2.34c). At the beginning of every adjustment iteration all weights are initialised to unity. So the formulae for the robust adjustment become:

$$\boxed{\begin{aligned} d\vec{x} &= (\mathbf{A}^T\,\mathbf{W}\,\mathbf{A})^{-1}\,\mathbf{A}^T\,\mathbf{W}\,d\vec{m} && \text{(2.34a)} \\ \vec{v} &= \mathbf{A}\,d\vec{x} - d\vec{m} && \text{(2.34b)} \\ \mathbf{W} &= diag(w_i) \qquad w_i = \frac{\Psi(v_i)}{v_i} && \text{(2.34c)} \end{aligned}}$$

If a robust adjustment is used for non-linear problems, there will appear two iteration loops in the adjustment: The outer loop is needed for the linearisation, where the linearisation's new working point $\vec{x}_a$ is determined and $\mathbf{A}$ and $d\vec{m}$ are adapted as explained on page 18.

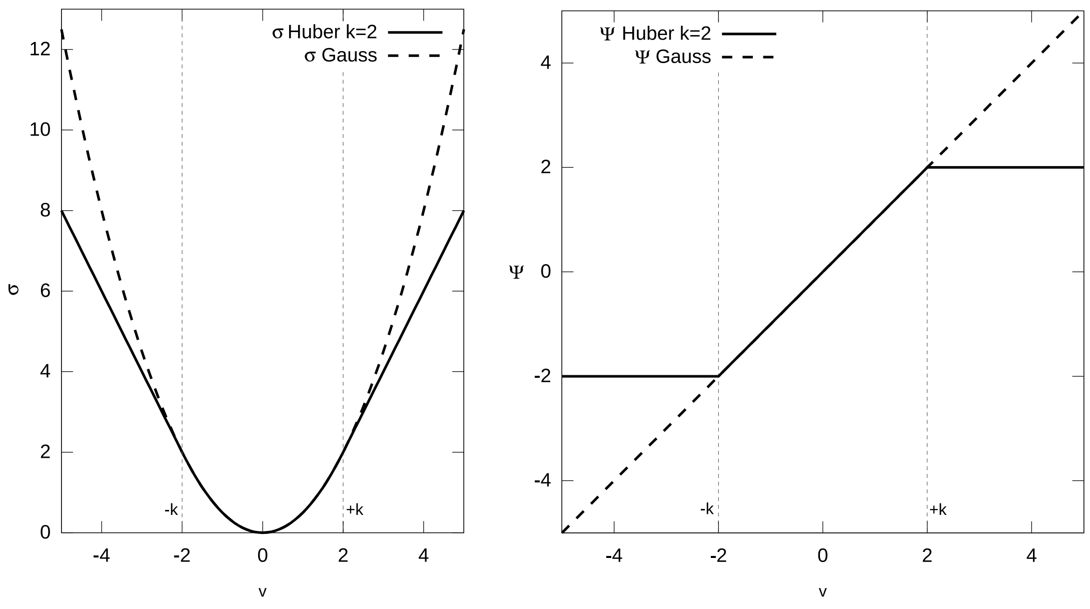


Figure 2.7: Left: $\sigma(v)$, right: $\Psi(v)$ of a M-Estimator for a Gaussian PDF N(0, 1) and for the Huber-k estimator with N(0, 1) and k=$2\sigma_0$

The nested loop calculates the weight adoption for every linearisation by iterating over the equations (2.34).

2.3.3 Applied Adjustment Theory in Multilateration

In Multilateration the basic equations are built by the well known equation system (2.5) shown in Chapter 2.2. These equations can be transformed into the following form and, for the sake of simplicity, we newly replace the time variables by the so-called pseudo ranges $r_i = c\,t_i$:

$$c\,t_i = \sqrt{(x_i - x)^2 + (y_i - y)^2} + c\,t_e \tag{2.35}$$

$$m_i = r_i = \sqrt{(x_i - x)^2 + (y_i - y)^2} + r_e \tag{2.36}$$

In the initial chapters the measured pseudo ranges were denoted by r_i. Since they correspond to the measurements at the i-th station, they will be called m_i in this chapter, in analogy to the chapters about adjustment theory. The variables x, y and t_e are the unknowns and we can build a linearised equation system like described in equation (2.29). Therefore, we have to calculate the following derivatives to perform the tailor expansion:

$$\begin{aligned}\frac{\partial m_i}{\partial x} &= \frac{x - x_i}{d_i}\\ \frac{\partial m_i}{\partial y} &= \frac{y - y_i}{d_i}\\ \frac{\partial m_i}{\partial r_e} &= 1\end{aligned} \tag{2.37}$$

$$d_i = \sqrt{(x_i - x)^2 + (y_i - y)^2}$$

where d_i is the distance between the i-th base station and the target. The linearisation of equation (2.36) then looks like this:

$$dm_i = \frac{\partial m_i}{\partial x}\, dx + \frac{\partial m_i}{\partial y}\, dy + \frac{\partial m_i}{\partial r_e}\, dr_e \tag{2.38}$$

By setting up the system of equations and writing it down in matrix notation like in equation (2.17) we get

$$d\vec{m} = \mathbf{A}\, d\vec{x} \tag{2.39}$$

where

$$\mathbf{A} = \begin{bmatrix} \frac{\partial m_1}{\partial x}(\vec{x}_a) & \frac{\partial m_1}{\partial y}(\vec{x}_a) & \frac{\partial m_1}{\partial r_e}(\vec{x}_a) \\ \frac{\partial m_2}{\partial x}(\vec{x}_a) & \frac{\partial m_2}{\partial y}(\vec{x}_a) & \frac{\partial m_2}{\partial r_e}(\vec{x}_a) \\ \vdots & \vdots & \vdots \\ \frac{\partial m_n}{\partial x}(\vec{x}_a) & \frac{\partial m_n}{\partial y}(\vec{x}_a) & \frac{\partial m_n}{\partial r_e}(\vec{x}_a) \end{bmatrix} \tag{2.40}$$

$$d\vec{m} = \vec{m} - \vec{m}(\vec{x}_a) \qquad d\vec{x} = \vec{x} - \vec{x}_a \tag{2.41}$$

With this linear system, we are now able to calculate adjustments like described in the previous chapters. The linearisation is performed in the point $\vec{x}_a$, an approximate solution, which has to be known before the adjustment is calculated. $\vec{x}_a$ is normally obtained by a closed formula e.g. Bancroft's Algorithm (see Chap. 2.2.1). In practise, the adjustment is calculated in the following steps:

1. Putting up the system matrix $\mathbf{A}$ in the point $\vec{x}_a$. This is done by calculating the matrix elements via the formulae for the derivatives (2.37). Therefore, the elements of the approximate solution $\vec{x}_a$ need to be filled in for x and y in these equations.

2. Calculating $d\vec{m}$ by subtracting the real measurements $\vec{m}$ by the measurements derived from the approximate solution $\vec{m}(\vec{x}_a)$. The derived measurements are calculated of $\vec{x}_a$ via the basic equation (2.36):

$$\vec{m}(\vec{x}_a) = \begin{bmatrix} m_{a1} \\ \vdots \\ m_{an} \end{bmatrix} \qquad m_{ai} = \sqrt{(x_i - x_a)^2 + (y_i - y_a)^2} + r_{ea}$$

3. Calculating the adjusted solution $\vec{x}$ via the equations (2.30).

4. Replacing $\vec{x}_a$ by the new calculated $\vec{x}$ and start the next iteration at step 1.

2.4 Equations for Three Dimensions

If the calculations are performed in three dimensions, one more unknown value, here denoted by z, is added to the basic equation system (2.5). Due to an additional unknown, also an additional equation is needed, in order to be able to solve the system. The system equations in three dimensions are then given by:

$$\begin{aligned}
r_1 - r_e &= \sqrt{(x_1 - x)^2 + (y_1 - y)^2 + (z_1 - z)^2} \\
r_2 - r_e &= \sqrt{(x_2 - x)^2 + (y_2 - y)^2 + (z_2 - z)^2} \\
r_3 - r_e &= \sqrt{(x_3 - x)^2 + (y_3 - y)^2 + (z_3 - z)^2} \\
r_4 - r_e &= \sqrt{(x_4 - x)^2 + (y_4 - y)^2 + (z_4 - z)^2}
\end{aligned} \tag{2.42}$$

As these equations form the base for Bancroft's algorithm and the applied adjustment equations, both are adapted in the following two chapters.

2.4.1 Bancroft's Algorithm in 3D

When calculating in three dimensions the measurement and solution vectors change to

$$\vec{s_i} = \begin{bmatrix} x_i \\ y_i \\ z_i \\ r_i \end{bmatrix} \qquad \vec{x} = \begin{bmatrix} x \\ y \\ z \\ r_e \end{bmatrix} \tag{2.43}$$

The Lorentzian inner product for four variables is given by

$$\langle \vec{u}, \vec{v} \rangle = u_1 v_1 + u_2 v_2 + u_3 v_3 - u_4 v_4 \qquad for \qquad \vec{u} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix}, \vec{v} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} \tag{2.44}$$

While the initial equation (2.10) does not change, the matrix **A** and the vectors have to be enlarged as follows:

$$\mathbf{A} = \begin{bmatrix} x_1 & y_1 & z_1 & -r_1 \\ x_2 & y_2 & z_2 & -r_2 \\ x_3 & y_3 & z_3 & -r_3 \\ x_4 & y_4 & z_4 & -r_4 \end{bmatrix} \tag{2.45a}$$

$$\lambda = \langle \vec{x}, \vec{x} \rangle = x^2 + y^2 + z^2 - r_e^2 \tag{2.45b}$$

$$\vec{1} = [1 \quad 1 \quad 1 \quad 1]^T \tag{2.45c}$$

$$\vec{b} = \begin{bmatrix} \langle \vec{s_1}, \vec{s_1} \rangle \\ \langle \vec{s_2}, \vec{s_2} \rangle \\ \langle \vec{s_3}, \vec{s_3} \rangle \\ \langle \vec{s_4}, \vec{s_4} \rangle \end{bmatrix} \tag{2.45d}$$

All other vectors change implicitly, and the calculation can be performed as described in Chapter 2.2.1 already.

2.4.2 Applied Adjustment Theory in 3D

As vector size in adjustment theory is not fixed, it is easy to adapt it for the three dimensional case. Also here the base station and solution vectors are enlarged by the z variable, like already shown in equation (2.43). The linearisation (2.37) is also extended by an additional derivation:

$$\begin{aligned} \frac{\partial m_i}{\partial x} &= \frac{x - x_i}{d_i} \\ \frac{\partial m_i}{\partial y} &= \frac{y - y_i}{d_i} \\ \frac{\partial m_i}{\partial z} &= \frac{z - z_i}{d_i} \\ \frac{\partial m_i}{\partial r_e} &= 1 \end{aligned} \tag{2.46}$$

$$d_i = \sqrt{(x_i - x)^2 + (y_i - y)^2 + (z_i - z)^2} \tag{2.47}$$

The matrix $\mathbf{A}$ then turns into

$$\mathbf{A} = \begin{bmatrix} \frac{\partial m_1}{\partial x}(\vec{x}_a) & \frac{\partial m_1}{\partial y}(\vec{x}_a) & \frac{\partial m_1}{\partial z}(\vec{x}_a) & \frac{\partial m_1}{\partial r_e}(\vec{x}_a) \\ \frac{\partial m_2}{\partial x}(\vec{x}_a) & \frac{\partial m_2}{\partial y}(\vec{x}_a) & \frac{\partial m_2}{\partial z}(\vec{x}_a) & \frac{\partial m_2}{\partial r_e}(\vec{x}_a) \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial m_n}{\partial x}(\vec{x}_a) & \frac{\partial m_n}{\partial y}(\vec{x}_a) & \frac{\partial m_n}{\partial z}(\vec{x}_a) & \frac{\partial m_n}{\partial r_e}(\vec{x}_a) \end{bmatrix} \tag{2.48}$$

All other symbols in the adjustment equations change implicitly.

2.5 Error Estimation

For every measurement, regardless of its area of application, the knowledge about its accuracy is essential in order to be able to work with it. The same applies to the solutions determined by a hyperbolic positioning system, especially in aviation.

This chapter presents a methodology to estimate two accuracy values: Section 2.5.1 discusses the estimation of the input data accuracy and Section 2.5.2 shows how this estimation can be used in order to calculate the accuracy of the solution. The methodology bases on the principles of adjustment theory, as they were explained in Chapter 2.3. Whereas the most important output of an adjustment is of course the Least Square optimised solution $\vec{x}$, there are much more variables available which carry useful information:

- The residuals $\vec{v}$ give the error for each measurement, in order to come to the solution $\vec{x}$.
- The inverse of the normal equation matrix, called $\mathbf{Q}$, contains information about the system's geometry and can be used to calculate an error propagation.
- The measurement weight matrix $\mathbf{P}$ describes how the input measurements $\vec{m}$ were weighted in the calculation

From these parameters, it is possible to determine two major error indicators: the error σ_0 of the input measurements $\vec{m}$ and the error σ_x of the solution $\vec{x}$. It is also important to distinguish between two types of error measures: A-priori errors are derived from an initial assumption about the measurement error and provide information how great the error will be in general. A-posteriori errors are determined from the measurements used in the calculation and provide information about the error in exactly this set of measurements.

2.5.1 Input Measurement Error

Statistical information about the measurement error ε_i is often known a-priori, in form of the expected measurement error σ_i and the mean measurement error σ_m. However, for each n-element sample of measurements $\vec{m}$, also the a-posteriori error σ_0 can be calculated by consulting the residuals $\vec{v}$. The formulae explained in this chapter are stated in [17] and [12], however, no detailed explanation is given in these books. This chapter tries to explain the principles of input error estimation in a reasonable and easy way, the mathematical deviations can be found in [18].

The vector of true errors $\vec{\varepsilon} = \vec{m} - \hat{\vec{m}}$ contains the necessary values to be added to the measurements $\vec{m}$ in order to come to $\hat{\vec{x}}$ over the well-known formula $\vec{m} - \vec{\varepsilon} = \mathbf{A}\,\hat{\vec{x}}$ (see (2.18)). Here $\hat{\vec{x}}$ denotes the correct solution, which describes the real position of signal emission and $\hat{\vec{m}}$ are the error-free measurements. With this knowledge, we can calculate the a-posteriori variance of the measurement errors, denoted by σ_0^2, via the formula:

$$\sigma_0^2 = \frac{1}{n}\sum_{i=1}^{n} p_i \cdot (m_i - \hat{m}_i)^2 = \frac{1}{n}\sum_{i=1}^{n} p_i\,\varepsilon_i^2 = \frac{\vec{\varepsilon}^T\mathbf{P}\vec{\varepsilon}}{n} \qquad (2.49)$$

However, as already discussed in Chapter 2.3.1, the real position of signal emission is unknown, and with it also the real error $\vec{\varepsilon}$. The error-free measurements $\hat{m}_i$ in the formula above can be calculated of the real point of signal emission $\hat{\vec{x}}$ over $\hat{\vec{m}} = \mathbf{A}\hat{\vec{x}}$. Since $\hat{\vec{x}}$ is unknown, we replace it by its best approximation, which is the Least Square adjusted result $\vec{x}$. Instead of $\hat{\vec{m}}$ in the previous formula then $\mathbf{A}\,\vec{x}$ will appear. This then causes the error $\vec{\varepsilon}$ being replaced by the residuals $\vec{v}$ in formula (2.50), as we have defined the residuals to be $\vec{v} = \vec{m} - \mathbf{A}\,\vec{x}$ (see equation (2.19)).

Because $\vec{x}$ was determined over the error-affected measurements $\vec{m}$ it is afflicted by a certain error, which in return influences the calculation of the measurement error variance in (2.49). As there are at least d measurements necessary to calculate $\vec{x}$, the best approximation of this influence can be defined as $d \cdot \sigma_0^2$, where σ_0^2 is the measurement error variance itself. So we come to the following formula, for the mathematical derivation please refer to [18].

$$s_0^2 = \frac{1}{n-d}\sum_{i=1}^{n} p_i\,v_i^2 = \frac{\vec{v}^T\mathbf{P}\vec{v}}{n-d} \qquad (2.50)$$

Here the variance is denoted by s_0^2 as it is determined from the finite vector $\vec{v}$, which is calculated of a sample of n measurements. Hence s_0^2 is just an approximation of the real measurement variance σ_0^2, it is better to call it the a-posteriori sample variance.

We see in equation (2.50) that the denominator is given by $n - d$, which describes the overdetermination of the system, i.e. the number of redundant measurements. In fact, redundancy makes it possible in the first place to calculate a sample variance: By having just d measurements, it would indeed be feasible to calculate the result $\vec{x}$, but then the residuals $\vec{v} = \vec{m} - \mathbf{A}\vec{x}$ would all be zero. If more measurements are available, the additional information is used to improve the solution and the measurements are corrected by $\vec{v}$ in order to find a compromise, which leads to an optimum solution $\vec{x}$. In this case $\vec{v}$ contains the information how well the measurements fit together, which is expressed statistically in the error variance. That is the reason why a sample variance can only be calculated if the system is overdetermined.

2.5.2 Error of the Solution

In the previous chapter the measurement error was determined, but normally it is more important to know the error of the calculated solution $\vec{x}$. The simplest way to do this is to calculate a Gaussian error propagation via a Pythagorean sum as shown in [16, p. 562]:

$$x = f(\vec{m}) \tag{2.51}$$

$$\sigma_x = \sqrt{\sum \left(\frac{\partial f(\vec{m})}{\partial m_i} \right)^2 \cdot \sigma_i^2} \tag{2.52}$$

$f(\vec{m})$ denotes a general system where the solution x is determined out of the vector of measurements $\vec{m}$. $\frac{\partial f(\vec{m})}{\partial m_i}$ is the first derivation of $f(\vec{m})$ with respect to the i-th measurement m_i, σ_i is the error of the i-th measurement and σ_x the resulting error for x. In case $f(\vec{m})$ is a linear system of equations we can write it down as a matrix multiplication:

$$\vec{x} = \mathbf{F}\,\vec{m} = \begin{bmatrix} a_1 & a_2 & \dots \\ b_1 & b_2 & \\ \vdots & & \ddots \end{bmatrix} \cdot \begin{bmatrix} m_1 \\ m_2 \\ \vdots \end{bmatrix} \tag{2.53}$$

As the system is linear, the differentiations are trivial and the error propagation (2.52) for x_1, the first element of the solution $\vec{x}$, becomes

$$\sigma_{x_1} = \sqrt{a_1^2\sigma_1^2 + a_2^2\sigma_2^2 + \cdots} = \sqrt{\sum a_i^2\sigma_i^2} \tag{2.54}$$

However, as $\vec{x}$ is multidimensional, it makes more sense to calculate a covariance propagation as proposed in [17]:

$$\mathbf{\Sigma}(\vec{x}) = \mathbf{F}\,\mathbf{\Sigma}(\vec{m})\,\mathbf{F}^T \tag{2.55}$$

$$\mathbf{\Sigma}(\vec{x}) = \mathbf{F}\begin{bmatrix} \sigma_1^2 & 0 & \cdots \\ 0 & \sigma_2^2 & \cdots \\ \vdots & & \ddots \end{bmatrix}\mathbf{F}^T \tag{2.56}$$

$$\mathbf{\Sigma}(\vec{x}) = \begin{bmatrix} \sum a_i^2\sigma_i^2 & \sum a_i b_i \sigma_i^2 & \cdots \\ \sum b_i a_i \sigma_i^2 & \sum b_i^2\sigma_i^2 & \\ \vdots & & \ddots \end{bmatrix} = \begin{bmatrix} \sigma_{x_1}^2 & \sigma_{x12} & \cdots \\ \sigma_{x21} & \sigma_{x_2}^2 & \cdots \\ \vdots & & \ddots \end{bmatrix} \tag{2.57}$$

We see that the diagonal elements of the covariance matrix $\mathbf{\Sigma}(\vec{x})$ contain the propagated errors $\sigma_{x_i}^2$ for each element x_i in $\vec{x}$. The other errors, denoted by $\sigma_{x_{ij}}$, result from the correlation between x_i with x_j and form the covariances. In the deviation it was assumed that the measurements are uncorrelated, which leads to a diagonal measurement covariance matrix $\mathbf{\Sigma}(\vec{m})$.

Now we apply this covariance propagation to the adjustment equations (2.26) and keep in mind that $\mathbf{\Sigma}(\vec{m}) = \mathbf{P}^{-1}\,\sigma_m^2$ (see formula (2.23)). The result will look like this; the entire deviation is shown in Appendix A.3.

$$\vec{x} = \mathbf{F}\,\vec{m} = \mathbf{Q}\,\mathbf{A}^T\mathbf{P}\,\vec{m} \tag{2.58}$$

$$\mathbf{\Sigma}(\vec{x}) = \mathbf{F}\,\mathbf{\Sigma}(\vec{m})\,\mathbf{F}^T = \mathbf{F}\,\mathbf{P}^{-1}\sigma_m^2\,\mathbf{F}^T \tag{2.59}$$

$$\vdots \qquad \text{(see Appendix A.3)}$$

$$\mathbf{\Sigma}(\vec{x}) = (\mathbf{A}^T\mathbf{P}\mathbf{A})^{-1}\,\sigma_m^2 = \mathbf{Q}\,\sigma_m^2 \tag{2.60}$$

As we see, the inverse of the normal equation matrix, called $\mathbf{Q}$, can be used to determine the error of the result $\vec{x}$. The big advantage is that $\mathbf{Q}$ is already available at the end of the adjustment calculation, so the result's error propagation is received together with the result quasi "for free". Furthermore, depending on the value which is inserted for σ_m^2, the a-priori or the a-posteriori error for $\vec{x}$ can be calculated:

The a-priori error is received when σ_m^2 is set to the mean of the a-priori measurement variances σ_i^2, as it was previously used to calculate the weight matrix $\mathbf{P}$ in formula (2.23). This will lead $\mathbf{\Sigma}(\vec{m}) = \mathbf{P}^{-1}\,\sigma_m^2$ to become a diagonal matrix with the measurement errors σ_i^2 in its diagonal (see definition of $\mathbf{P}$ in equation (2.23)). If this a-priori measurement error is propagated, the result of the propagation $\mathbf{\Sigma}(\vec{x})$ will describe the a-priori solution error. Alternatively, if the a-posteriori measurement error value s_0^2, like calculated in equation (2.50), is inserted as σ_m^2 into formula (2.60), the resulting covariance matrix $\mathbf{\Sigma}(\vec{x})$ will contain the a-posteriori error.

Figure 2.8 shows the result of an a-priori error propagation as discussed before. In the figure each ellipse illustrates the 1σ error ellipse (see Appendix A.4) for the solution located

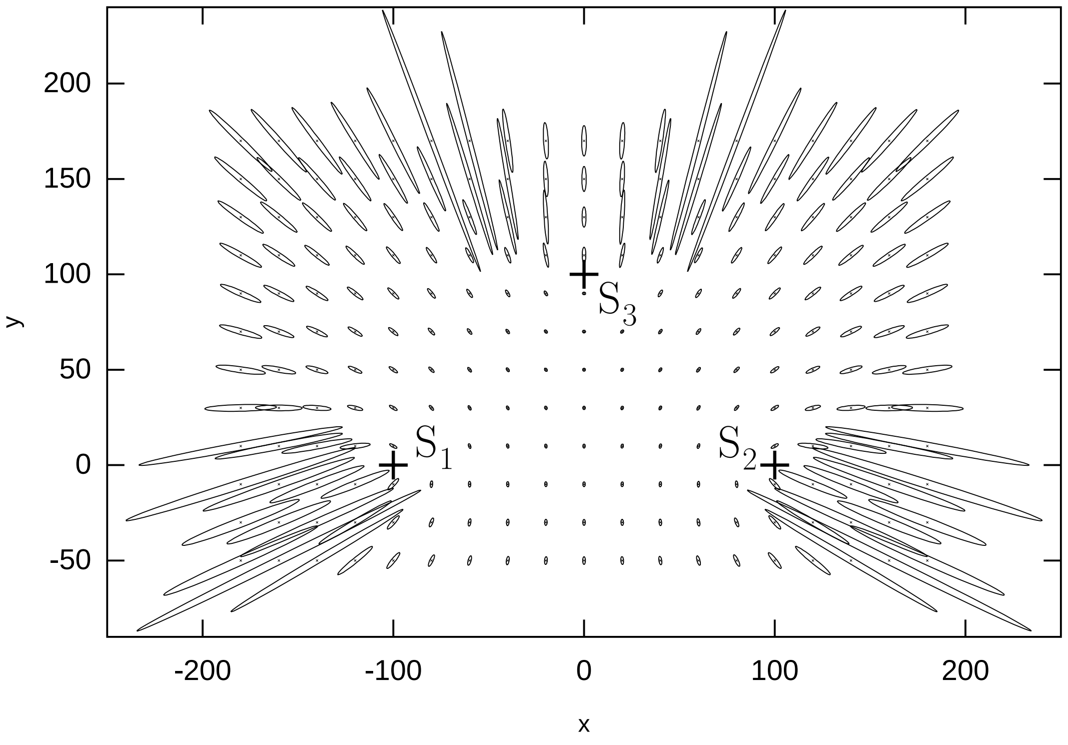
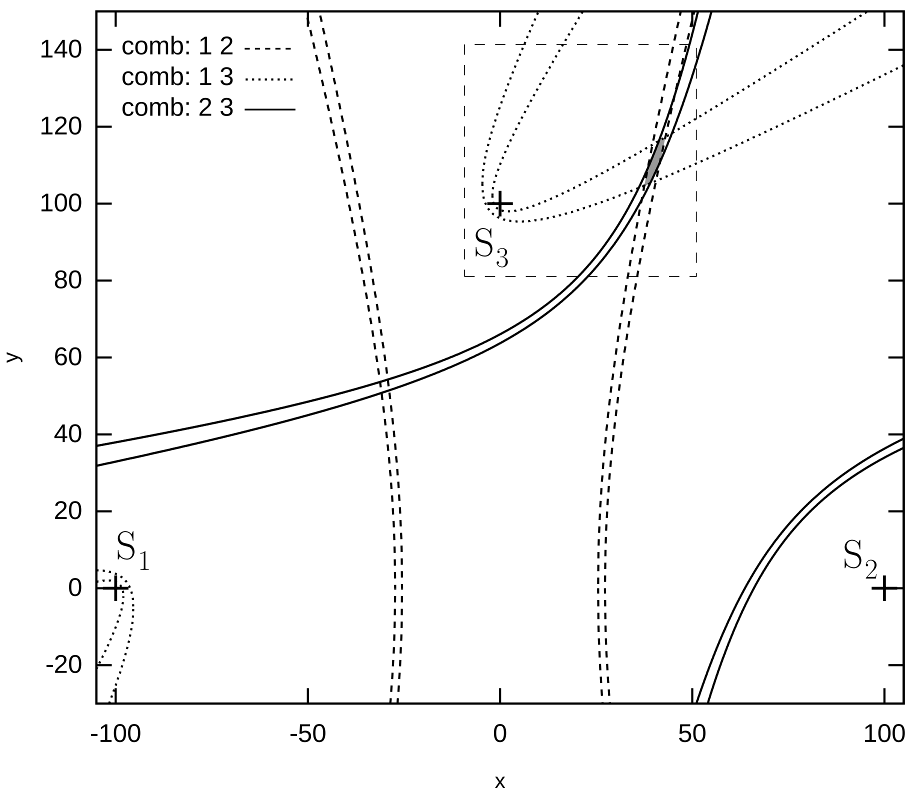


Figure 2.8: 2D error ellipses for a grid with mesh-width of $20m$ and a measurement error of $\sigma_i = 3 \cdot 10^{-9}s$. Coordinates are given in metres.

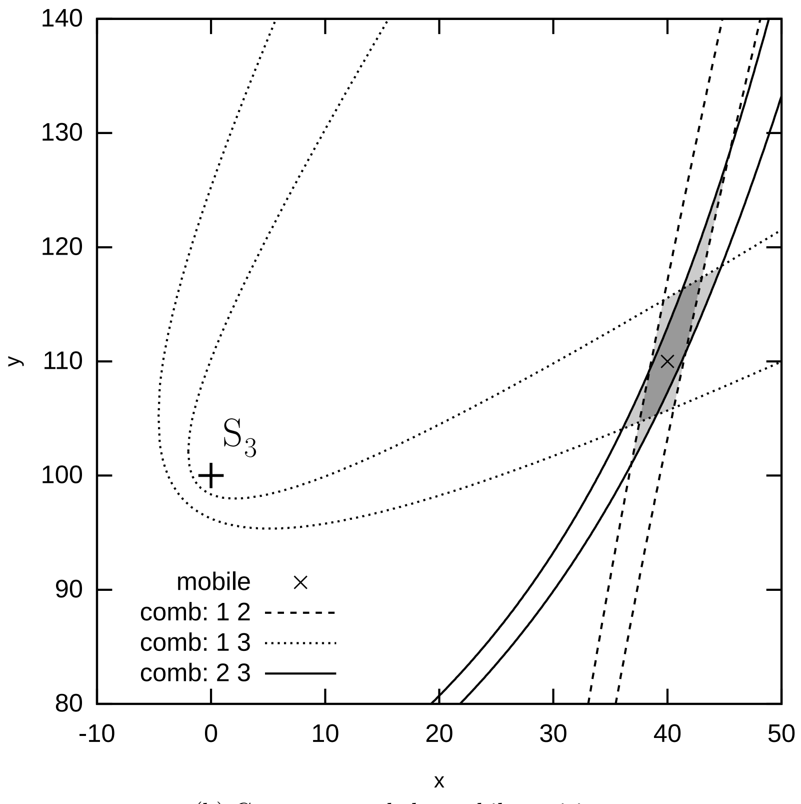
in its centre. The propagation was calculated in two dimensions with a measurement error of $\sigma_i = 3ns$ and it was assumed to be additive white Gaussian noise (AWGN). In the area between the Remote Units S_1 to S_3 we see that the ellipses are quite small, hence also the solution error is small there. Generally, we can say that the further we get from the centre the more substantial the error becomes. The worst solution accuracy is received "behind" the RUs which mostly coincides with the "unfavourable regions" shown in Figure 2.6.

It is noticeable that the error ellipses are oriented towards to the centre. The reason for that lies in the hyperbola intersections which become glancing with the distance from the centre. This becomes clearer in Figure 2.9 which shows the cuts between dispersed hyperbolas for the mobile position (40, 110) in the same scenario. Each hyperbola is illustrated by two lines, describing the 1σ dispersion caused by a measurement inaccuracy of again $3ns$. The ideal hyperbola would be located in the middle of the two lines.

As the left figure shows, the dispersion is smallest on the connection line between the Remote Units, which are used to put up the hyperbola. The dispersion grows then with the distance from the RUs, how fast the dispersion grows depends on the geometry (see difference between combination S1-S3 and S2-S3). The right figure contains a more detailed view of the hyperbola cuts at the mobile's position. The bright grey areas symbolise cuts between two hyperbolas, the dark grey area between three hyperbolas.



(a) All hyperbolas



(b) Cutout around the mobile position

Figure 2.9: 1σ dispersion of the hyperbolas leading to the mobile position (40, 110) in the scenario of Figure 2.8. Measurement errors are $\sigma_i = 3 \cdot 10^{-9} s$, distances are given in metres.

2.5.3 Dilution of Precision (DoP)

Another error indicator which is often provided is the so called Dilution of Precision (DoP). The DoP describes the influence of the systems geometry onto the solution accuracy. According to [3, p. 250] it can be calculated from the co-factor matrix (here given in three dimensions).

$$\mathbf{Q} = (\mathbf{A}^T\mathbf{P}\mathbf{A})^{-1} = \begin{bmatrix} q_x^2 & q_{xy} & q_{xz} & q_{xr} \\ q_{xy} & q_y^2 & q_{yz} & q_{yr} \\ q_{xz} & q_{yz} & q_z^2 & q_{zr} \\ q_{xr} & q_{yr} & q_{zr} & q_r^2 \end{bmatrix} \tag{2.61}$$

The following DoP measures are commonly used:

$$GDOP = \sqrt{q_x^2 + q_y^2 + q_z^2 + q_r^2} \qquad \textit{geometrical DoP} \tag{2.62a}$$

$$HDOP = \sqrt{q_x^2 + q_y^2} \qquad \textit{horizontal DoP} \tag{2.62b}$$

$$VDOP = \sqrt{q_z^2} \qquad \textit{vertical DoP} \tag{2.62c}$$

$$TDOP = \sqrt{q_r^2} \qquad \textit{time (range) DoP} \tag{2.62d}$$

If the calculation was performed in two dimensions the factor q_z^2 is not available. So it is missing in the $GDOP$ calculation and no $VDOP$ information can be given obviously. At this place, it is important to mention that the DoP gives a solution error measure, which is independent from the input data accuracy. Figure 2.10 shows a contour plot of the $HDOP$ for the same scenario as in the error propagation explanations in the last chapters and where to all measurements equal weights were assigned.

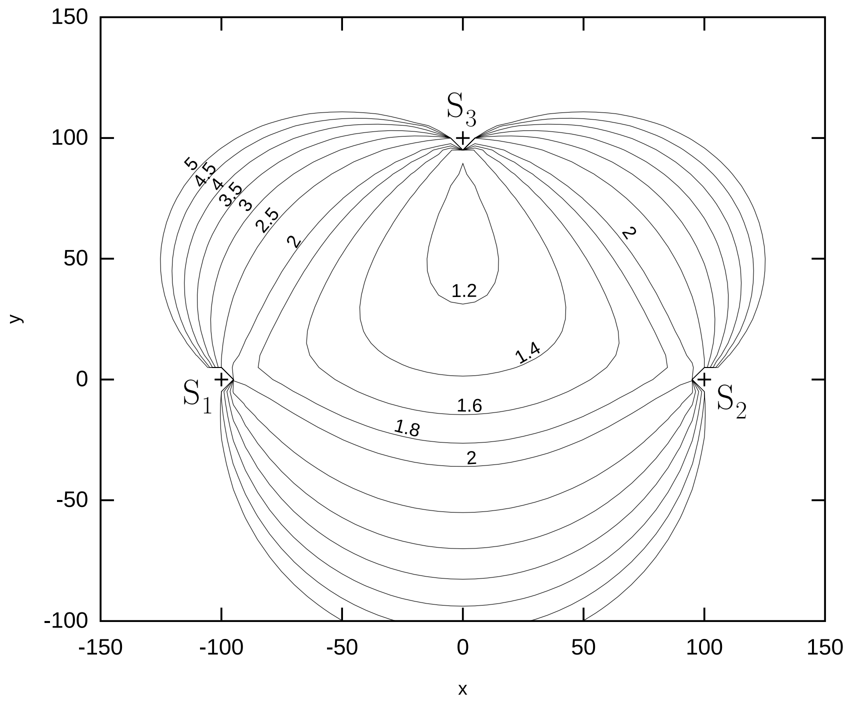


Figure 2.10: HDOP contour plot. Units are given in metres.

2.6 Secondary Surveillance Radar Modes

Secondary Surveillance Radar (SSR) technology uses transponders in order to localise vehicles and aircraft and also to provide information about the latter. The technology is used for military and civil purposes and is standardised by ICAO mainly in [20]. The basic principle works like follows:

A SSR radar site sends a pulse sequence, modulated onto a $1030MHz$ carrier, in order to interrogate the transponders of aircraft and vehicles in its reach. Depending on this sequence, transponders will answer with a reply pulse sequence on a frequency of $1090MHz$. The information content of the reply depends on the mode which was requested by the Radar in the interrogation. The commonly used SSR modes in civil aviation are called Mode-A, C and S. These three modes are the only few treated in this chapter, as the other modes are either used for military purposes or not operationally used at all [4]. The explanations in the subsequent chapters are limited to the encoding and content of the SSR replies, as mainly this is interesting for Multilateration systems. For more detailed information about SSR please refer to [4], which gives a well and clearly structured tutorial about SSR modes and the working principle. The information in the following chapters was merged from [4], [1] and [7].

2.6.1 Mode-A and Mode-C Reply

SSR Mode-A/C is widely used in civil aviation but it is quite an old standard which became operational in the mid 1960s [1, p.13]. The Replies of both modes have the following pulse framing:

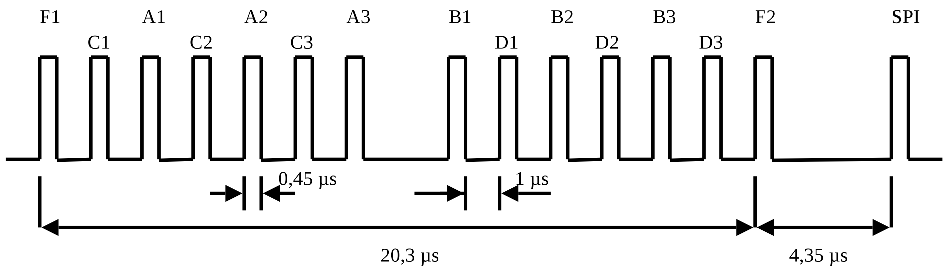


Figure 2.11: SSR Mode-A/C reply taken from [5]

The pulses $F1$ and $F2$ are called the framing pair and are always available. The twelve pulses denoted by A to D contain the payload information. The Special Purpose Identification (SPI) pulse can be enabled on request and is activated by the pilot if an air traffic controller requests a so called “squawk ident” which is needed if an aircraft needs to be specially identified [4].

Depending on the interrogation mode, the replies’ pulse triples A to D have two different meanings: In Mode-A replies these pulses encode the so-called Mode-A identity or squawk code which is simply a number assigned to an aircraft in order to identify it. As all in all there exist only 4096 different squawk codes, these codes are dynamically assigned to aircraft by air controllers [1, p. 149]. If the reply is of Mode-C, eleven of the twelve bits contain the Gillham encoded barometric height of the aircraft, with a resolution of $100ft$ (see [20]). Normally Mode-A and Mode-C are interrogated in the sequence “ACACAC...” [7, p.123] and so both reply types are received constantly. From the reply itself there is no trivial way to identify whether the reply is of Mode-A, or Mode-C. [4] states to this fact:

“Note that the reply information itself does not contain any information to indicate which mode it is a reply to. The interrogator will assume that the replies received are in answer to it[s] *latest mode of interrogation.”*

If SSR is used in the conventional way, where just the interrogating site evaluates the replies, this does not lead to severe problems. If, however, a third-party system, e.g. a passive Multilateration system, works with the replies, statistical ways and/or a transponder position history have to be used in order to distinguish between Mode-A and Mode-C replies.

2.6.2 Mode-S Reply

Mode-S is a newer, more flexible SSR standard which enables the report of more detailed and diverse information. A Mode-S reply has the following format:

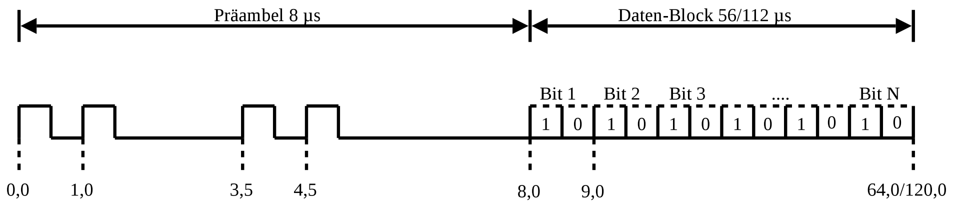


Figure 2.12: SSR Mode S reply taken from [5]

The preamble consists of four $0.5\mu s$ long pulses. The data block is pulse position modulated (PPM) and can be 56 or 112 bits long, depending on the transmitted data format. Each reply contains a 24-bit long Mode-S address which is uniquely assigned to every aircraft by ICAO, and also a parity information (see [20, p. 3-86]).

According to [4], Mode-S has some big advantages in comparison to Mode-A/C: Firstly, it is possible to interrogate single transponders over their Mode-S address and to poll specific information this way. Secondly, the altitude reports are provided in a more accurate 25ft altitude coding. Finally, Mode-S also provides check-sums in its replies in order to achieve higher data integrity via parity checking.

The standard supports different data formats and builds the transport layer for several services like ADS-B, ACAS, etc. Over these services, useful information can be exchanged between aircraft themselves and between aircraft and ground stations. Some services also support so-called broadcasts, which means that the transponder sends replies also without being interrogated. So a Mode-S transponder can, for instance, emit an Extended Squitter reply once per second which contains the aircraft’s address, GPS position and other useful information. For further reading about Mode-S and ADS-B please refer to [20], [21] and [22].

3 Design

This chapter discusses design issues of a Multilateration system's calculation unit, often also called Target Processor (TP). The unit should be a prototype which is able to investigate different scenarios and algorithms with the aim to elaborate the necessary know-how to implement a fully functional Multilateration system for civil aviation. The European Organization for Civil Aviation Equipment (EUROCAE) has standardised requirements for Multilateration systems in [23, p. 24]. The document states:

> *"The system shall calculate the horizontal positions of targets detected on the runways, taxiways and centerlines of the apron areas to within 7.5 m with a confidence level of 95% and to within 12 m with a confidence level of 99%."*

The intention of the MLAT system in development is to fulfil these accuracy constraints and, as the TP is the central point of such a system, it is best to perform the accuracy investigation there. Therefore, the Target Processor, presented in this chapter and was implemented by the author in the course of this work, is designed to be modular, highly configurable and extensible.

Sub-chapter 3.4 "Calculation" deals with the application of Multilateration theory, and combines the calculation methods presented in Chapter 2 "Theoretical Background". However in order to work with data from real measurements, more problems need to be solved: measurements have to be collected and correlated, Remote Units have to be synchronised, etc. Solutions for these and other problems are provided in the following sub-chapters in order to give a complete design structure of a MLAT Target Processor. The main tasks which have to be solved by the TP can be defined as follows:

1. **Data collection:** Data of all Remote Units needs to be received, decoded and brought together.

2. **Synchronisation:** As every remote unit measures time on its own, a common time base is needed in order to use the measurements. This is done in the synchronisation step, when synchronisation messages are evaluated in order to determine parameters which are then used to synchronise TOA measurements.

3. **Measurement correlation:** Measurements received at different units, but belonging to one and the same signal emission, have to be identified and be tagged as a tuple.

4. **Calculation:** Here for each measurement tuple the actual calculation of the target position is performed. Furthermore, also parameters like Dilution of Precision (DoP), measurement or solution error are determined.

5. **Result emission:** Information about the detected target needs to be emitted in several data formats depending on the actual process configuration.

It makes sense to partition the Target Processor implementation into modular blocks, according to this logical task separation. Figure 3.1 shows such a block diagram with data

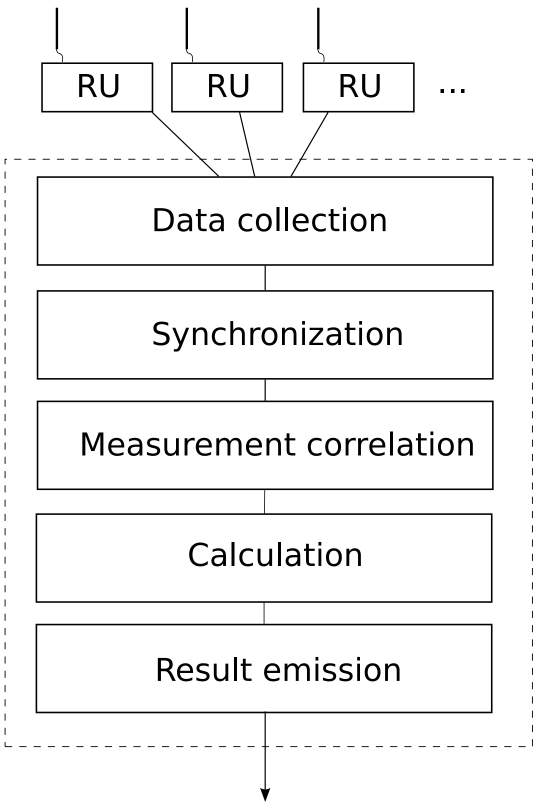


Figure 3.1: Block diagram of the Target Processor unit

flow. The tasks of these blocks are well separable and so each block can be implemented more or less independently. Furthermore communication between these blocks is quite simple to implement, as the data flow is linear. Design details to each of the blocks are discussed in the following sub-chapters.

An important feature of the Target Processor is its modularity: Different input formats, synchronisation methods and output formats have to be supported, as well as calculation in two or three dimensions. That is why for each block different implementations will be necessary, which must be easily replaceable over configuration parameters.

3.1 Remote Unit

This chapter gives a very short overview of the design of the Remote Units which is used in the developed distributed time Multilateration system prototype. In fact, this topic could fill a whole book, however, as this work focuses on the Target Processor, at this point just some basic but relevant features of the Remote Unit are discussed.

The task of a Remote Unit for such a system is to collect measurement data and to deliver it timestamped to the TP. It basically consists of an antenna, a receiver unit, data processing facilities and a communication interface to communicate with the TP. In general, all radio signals and frequencies can be used for Multilateration. As the developed Multilateration system prototype is meant to be used in civil aviation, the RU works with SSR reply signals. Figure 3.2 shows the hardware layout of the Remote Unit.

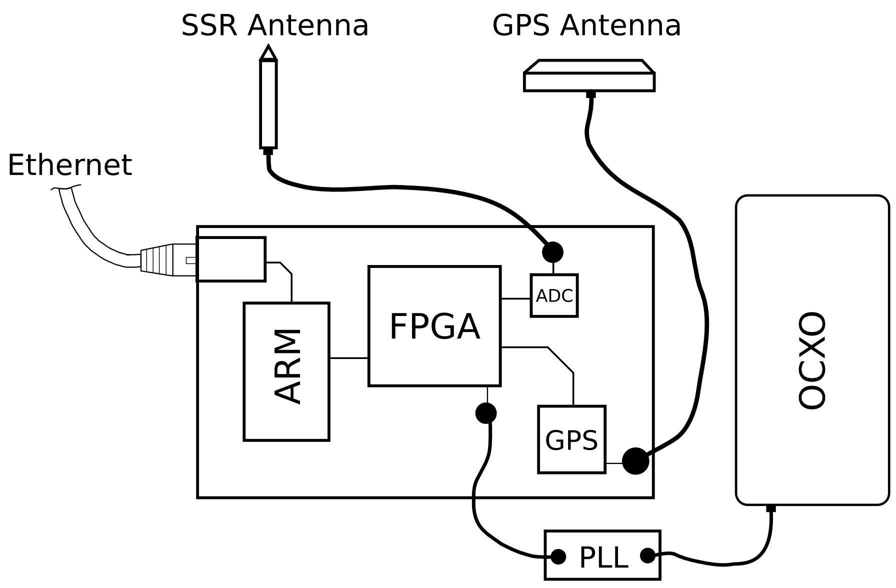


Figure 3.2: Block diagram of the Remote Unit used in the MLAT system prototype

After a signal was received over the omnidirectional SSR antenna (Xerxes CX-91050V-1090), the signal is mixed, digitalised with an analog to digital converter and demodulated in the Xilinx Spartan3e FPGA. Then the most important task is performed: the timestamping, where the TOA timestamps are being determined. After that, the ARM STR912FA micro-processor decodes the signal's message content because it contains useful information about the aircraft or vehicle, which is among other things needed for measurement correlation (see Chap. 3.3). The message content, accompanied by the receiving timestamp and information like signal power, is then sent over Ethernet to the TP. In the following the two most important features of the Remote Unit are explained: the timestamping and the generation of GPS synchronisation messages.

Timestamping: The time base at each unit is given by a highly precise Oven-Controlled Crystal Oscillator (OCXO) of type 2920136 and produced by Piezo Crystal Company. The oscillator has a frequency of $10MHz$ which is up-sampled to $80MHz$ by the Phase-locked loop (PLL) ADF4360-8, located on a separate board. A digital counter on the FPGA is incremented in every cycle of this clock. This counter builds the base value from which the timestamps are derived afterwards. Due to the clock frequency of $80MHz$ the time measurement resolution is $12.5ns$ which corresponds to a range resolution of $3.75m$. However, as discussed in [24], for Multilateration systems a range measurement accuracy of around $1m$ should be aspired, hence also the resolution should be in this scale. That is why the Remote Unit uses the same clock four times 90° shifted, in order to provide also fractional information to the timestamps. This technique increases the measurement resolution to $0.94m$.

For the timestamping, a point in the received SSR reply signals has to be defined, at which the counter value is taken as TOA timestamp. The current implementation of the RU provides timestamps for the rising edges of all pluses in SSR Mode-A/C replies and for the preamble pulses in Mode-S replies. The timestamps are taken at 50% of the pulse's mean peak power, measured relative to the noise level.

In order to achieve a good measurement accuracy, the oscillators stability is essential: Oscillator frequencies are sensitive to external influences, primarily to temperature changes which cause their frequencies to drift. That is why the RU is equipped with an oven heated oscillator, that keeps the temperature of the crystal constant via an electronic oven. According to the datasheet [25] the installed oscillators have a short time stability of $\sigma_{1s} < 1 \cdot 10^{-10}$ (Allan deviation with $\tau = 1s$). The accuracy of the PLL is assumed to be negligible at this point, since it follows the oscillator frequency with high accuracy as soon as it is locked.

GPS synchronisation messages: Every Remote Unit has its own oscillator to measure time by incrementing the FPGA counter. Therefore, the Target Processor needs to create a common time base for all RUs in the synchronisation step (see Chap. 3.2). For this purpose the RU is equipped with the GPS timing module u-blox Lea4T which receives the satellite signals over a Tecsys RV-76 antenna. This module provides one time pulse per second (PPS) with a maximal achievable accuracy of $\sigma = 15ns$ to UTC [26]. This pulse is registered at the FPGA and timestamped the same way the SSR replies are. The timestamp and the corresponding UTC time which is fetched from the GPS module, are then sent together to the TP in a synchronisation message.

3.2 Synchronisation

As all Remote Units measure time on their own via their local oscillators, it is necessary to find a common time base which is valid for all RUs. Just with this common time it is possible to calculate the time of signal arrival differences (TDOA) which are essential in the localisation calculation. The subsequent chapter gives a general approach for the synchronisation calculation, and explains the thereby necessary parameters. Chapter 3.2.2 finally shows how these synchronisation parameters can be determined from Global Navigation Satellite System (GNSS) timing data.

3.2.1 Synchronisation Model

The time of signal arrival (TOA) measurements are determined by a counter on each RU, which is incremented constantly by a precise oscillator. This counter value then forms the measurement timestamps. If these timestamps are multiplied by the Remote Unit's oscillator frequency, the outcome builds the time which has passed by since the counter was zero. In order to determine a time which is valid for all RUs, the calculation from the timestamp z to the synchronised receiving time t_s can be modelled by the formula

$$t_s = \check{t} + (z - \check{z}) \cdot T \tag{3.1}$$

where we call $\check{t}$ the synchronisation base time and $\check{z}$ synchronisation base counter. T $(=\frac{1}{f})$ is the cycle duration of the oscillator. These synchronisation parameters have to be derived from synchronisation events which are received at all RUs.

The oven-controlled crystal oscillators in the RUs have an adequate short term stability ($\sigma = 1 \cdot 10^{-10}$ Allan deviation with $\tau = 1s$ [25]) but can drift more over longer time intervals. The synchronisation situation is even worse if, like in our case, more oscillators are used which possibly drift into different directions. So it is advantageous to synchronise the oscillators in intervals of one to ten seconds in order to keep the influence of long time drift small. Because of this, the synchronisation base time at the i-th synchronisation event is denoted by $\check{t}_i$ and the corresponding base counter by $\check{z}_i$. As the oscillator frequency may vary over time as well, the cycle duration at the i-th synchronisation event is denoted by T_i. Finally the synchronisation formula becomes

$$t_{s,i} = \check{t}_i + (z - \check{z}_i) \cdot T_i \tag{3.2}$$

where $t_{s,i}$ denotes the synchronised time, synchronised to the i-th synchronisation event. This model is basically a linear extrapolation, but it is expected to be precise enough: The oscillator frequency can be assumed to be constant in the period between two subsequent synchronisation events. The quality of the synchronisation depends on the way the parameters are determined.

According to [11] there are two practical ways to produce synchronisation events: One possibility is to use a time reference from another accurate time source like a Global Navigation Satellite System (GNSS). Special receiver modules for such systems offer the possibility to get an accurate time pulse each second (see Chap. 3.2.2). The second synchronisation possibility is to use reference transponders with accurately known positions. If such transponders constantly emit a signal which is received at all RUs, the synchronisation can be calculated over the signal's TOA at each unit. When this thesis was being created, the MLAT system prototype was supporting only GNSS synchronisation. Also this chapter concerns just this way of synchronisation.

3.2.2 Synchronisation over GNSS

Global Navigation Satellite Systems (GNSSs) measure time with atomic clocks in their satellites and broadcast accurate timestamps in order to enable receiver modules to determine their own position (see [12]). Receivers which are specially designed for timing purposes, are able to use the broadcasted timestamps to provide accurate time pulses. For this purpose, however, they need to know their actual position in 3D which can either be set manually or be determined by the module itself [26]. The provided pulses have a 1σ accuracy to UTC of approximately 50 nanoseconds, but it is possible to increase the accuracy to 15ns, by fetching a quantisation error information from the receiver [26][27]. Additionally, such modules provide the possibility to poll the accurate Coordinated Universal Time (UTC) the pulse was triggered.

As already discussed in Chapter 3.1, the RUs used in the prototype are equipped with such a GPS module which provides one pulse per second (PPS). The synchronisation messages, the RU sends to the Target Processor after the pulse was detected, contain the pulse's (RU–clock) timestamp and its UTC-time. With this information it is possible to synchronise every unit to one common time source: the time given by GPS.

In a first attempt the provided UTC time can build the synchronisation base time $\check{t}_i$, the corresponding synchronisation base counter $\check{z}_i$ would be given by the timestamp. The only missing information in the synchronisation formula is the oscillators cycle duration T_i. This value can be determined by subtracting two subsequent base counters, or by the averaging some of these base counter increments.

However, the GPS time-pulse is affected by an error of $\sigma = 15ns$ [26] in the best case which corresponds to a distance of $4.5m$. If the timestamp–UTC-time pair would simply be used as base counter and base time, this error would have direct influence onto the synchronised time $t_{s,i}$ (see equation (3.2)). Since a measurement accuracy of $1m$ to $3m$ is intended, a better method, taking more synchronisation events into account, has to be developed. This means that a compromise of more synchronisation events has to be found for synchronisation parameter determination, e.g. by calculating an average of more events.

After several attempts to average frequency and counter–time pairs separately, it has turned out that calculating a linear regression (see Appendix A.5) over the pairs is a good attempt to overcome the problem: A linear regression basically fits a line trough the counter–time pairs by reducing the resulting squared error. Also the synchronisation equation (3.2) describes a line: T_i is the slope, $\check{t}_i$ the y-intercept and $\check{z}_i$ is an offset on the x-axis. Consequently the line parameters, provided by the outcome of the regression, can be used directly as synchronisation parameters.

In order to apply linear regression to the GNSS synchronisation problem, the counter values form the x-axis and the UTC time values the y-axis. The resulting regression parameters can then be used as follows: The y-intercept (a_0) builds the synchronisation base time $\check{t}_i$, and the slope (a_1) the cycle duration T_i.

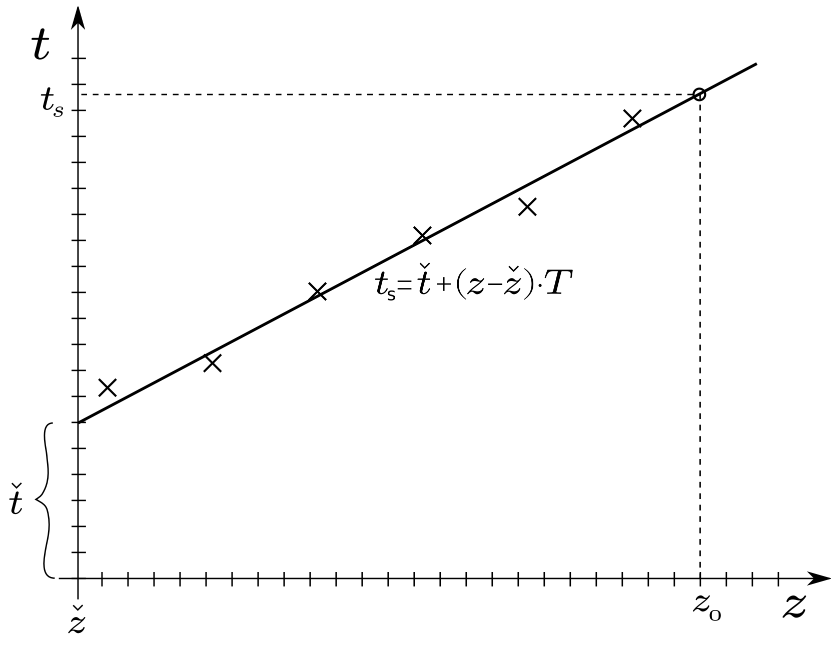


Figure 3.3: GNSS synchronisation over linear regression

Figure 3.3 shows a diagrammatic illustration of this regression: The crosses illustrate six measured counter–time pairs, the line is the the regression line which results out of the calculated regression parameters. The dashed lines illustrate a synchronisation: the synchronisation time t_s is determined from a received counter value z_0 over the synchronisation parameters which were determined in the regression. In this case the synchronisation was calculated with a history length of six elements, which means that the six lastly received synchronisation messages are used in the synchronisation calculation.

The synchronisation base counter $\check{z}_i$ would be zero if the counter values were used directly in the regression like explained before. However, in order to avoid precision problems, it makes sense to subtract an offset, e.g. the smallest counter value in the respected history, from each counter when the regression is calculated. This offset then forms the parameter $\check{z}_i$ in equation (3.2).

3.3 Measurement Correlation

Multilateration systems are designed in a way that one and the same signal is received at multiple base stations. So for every signal there is a couple of time of signal arrival (TOA) measurements available, which belong to the same signal emission. As these measurements belong together, they have to be identified in the measurement correlation step and be marked as a measurement tuple. In a later step the positioning calculation is performed for each tuple. There are two main features over which the correlation can be performed: the signal content and the TOA. The following two chapters describe possibilities and problems of these two strategies. An algorithm which solves the correlation problem is finally given in the implementation chapter in Section 4.5.

3.3.1 Correlation over Signal Content

The correlation task over signal content can be solved quite trivially by matching the message content data bit by bit. However, there may appear matching ambiguities, i.e. matches of measurements that do not belong to the same signal emission.

These ambiguities are caused by wrong match of replies emitted by either the same transponder, or by different transponders. The first case appears if one target emits the same message more than one time. However, in SSR the minimum time between two such emissions is restricted to $60\mu s$ to $125\mu s$ [1, p.27], and so a correlation over TOA timestamps may solve this problem. The second case appears if two targets emit the same message. It is much harder to resolve such an ambiguity, as targets behave independently.

The system prototype developed during the creation of this work can cope with SSR radar signals for civil aviation and so the three SSR modes A, C and S (see Chap. 2.6) are used. As Mode-S and Mode-A/C differ in terms of message length and content, there appear different correlation problems in both standards. The rest of this chapter discusses these difficulties.

Mode-S content correlation: Mode-S replies have a data block containing a payload of 56 or 112 data bits as explained in Chapter 2.6.2. In each reply, 24 of these bits contain the sender's Mode-S address, be it encoded directly in plain-text or xor-ed by the message's parity information (see [20]). Mode-S addresses are assigned globally by ICAO, and so it is improbable that two different transponders send the same reply at the same time. Still, wrong correlation of replies coming from the same transponder may appear, as the transponder can send out the same message more than one time. In this case the correlation ambiguity has to be solved over TOA correlation like explained in the next chapter.

Mode-A/C content correlation: Content correlation is much more difficult when working with SSR Mode-A/C replies. The first problem is that there is no non-statistical way to identify whether the reply is a Mode-A squitter or a Mode-C altitude reply. Normally, aircraft are alternately interrogated for Mode-A and Mode-C. Hence, replies for both modes are received all the time (see Chap. 2.6.1). This leads to correlation collisions of e.g. one aircraft's Mode-A message and another target's Mode-C altitude reply. Furthermore, Mode-A codes are dynamically assigned to aircraft and multiple targets may emit the same Mode-A code within the area of interest. So also collisions between two target's Mode-A codes are possible.

These two problems form matching ambiguities of measurements belonging to different targets. However, in Mode-A/C also correlation ambiguities of replies, coming from the same transponder, appear: It has also been observed in measurements, performed in the aerospace of Graz, that one and the same transponder may emit up to ten equal Mode-A/C replies within one millisecond. As normally all these replies have the same content, they will be falsely mapped together.

3.3.2 Correlation over Time of Signal Arrival

As TOA information is essential for the Multilateration calculation it is measured with high resolution (see Chap. 3.1). Therefore, it is possible to define small time windows in which two or more measurements of a tuple must or must not lie.

Firstly, one time interval is limited by the systems geometry: The system's base stations are located at known distances from each other. We can say that the maximum receiving time difference between two Remote Units, is equal to the time the signal needs to travel from one unit to the other. We call this timeout the detection timeout Δt_{det} which tells us that measurements received at different Remote Units, but not within this interval, cannot belong to the same signal emission. This time interval increases approximately by one microsecond for each $300m$ base station separation, as electromagnetic waves propagate at speed of light. However, reflections and multipath effects can cause signals to arrive later.

Secondly, SSR transponders are not able to answer to more than one interrogation at the same time and they also need a recovery time after each reply sent. In Mode-A/C this dead time lies within $60\mu s$ and $125\mu s$ [1, p.27]. In Mode-S the time is even longer, as, due the Mode-S replies bit-length, each message has already a duration of $64\mu s$ or $120\mu s$ (see Chap. 2.6.2). With this knowledge we can define the reflection timeout Δt_{refl} which tells us that, if two replies from the same emitter are received within this interval on the same receiver unit, the one that has secondly arrived will be most probable a reflected copy of the signal.

With these two timeouts it is possible to develop an algorithm which solves the correlation problem. The algorithm that fuses content and TOA correlation is presented in Section 4.5, in the implementation chapter of this work.

3.4 Calculation

The core element of a Multilateration TP is the calculation unit. This unit uses tuples of synchronised TOA measurements which belong to the same signal emission, and calculates the position where the signal was sent out from. This chapter describes how such a calculation unit can be set up by applying the theoretical principles explained in Chapter 2. The clue lies in the way the theory is combined at the end, thus the calculation unit has to complete the following tasks:

1. **Calculation of closed solutions:** Closed solutions form the base for the later calculation steps. As their solution accuracy strongly depends on the geometry, the measurement sets, from which they are calculated, need to be well selected.

2. **Solving solution ambiguities:** Closed solutions are mostly ambiguous, so the correct solution has to be identified.

3. **Increase of solution accuracy by adjustment:** In order to increase solution accuracy, more measurements can be taken into account to calculate a Least Squares adjustment.

4. **Estimation of the calculation error:** It is important to provide the accuracy of a calculated target position and so an a-priori and an a-posteriori solution error estimation need to be determined.

5. **Outlier detection:** Measurements can be affected by outliers or gross errors, coming from different sources. These errors need to be detected and handled accordingly.

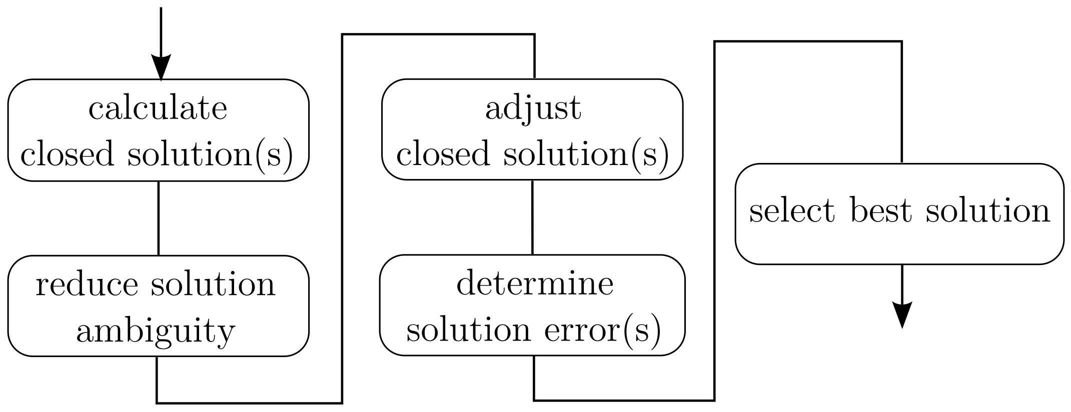

calculate closed solution(s)
reduce solution ambiguity
adjust closed solution(s)
determine solution error(s)
select best solution

Figure 3.4: Simplified flowchart of the Multilateration calculation

A simplified flowchart of a Multilateration calculation is given in Figure 3.4. The following sub-chapters elaborate the different blocks, shown in the figure. The final combination of the calculation tasks and the implementation of the calculation unit are finally found in Chapter 4.6, in the implementation section of this work.

3.4.1 Calculation of Closed Solutions

Closed solutions form the first step in the calculation chain and give an initial solution for the positioning calculation. Basically they are the mathematical exact solution of a system of equations which consists of as many equations as unknowns are to be determined. This is three equations for a two dimensional solution and four for a 3D solution. As every equation takes exactly one measurement as argument (see equation (2.2)), also just this number of input measurements can be considered. The theoretical basics for closed solution determination are found in Chapter 2.2.

As already discussed in Chapter 2.2.2, closed solutions may be affected by non-trivially resolvable ambiguities. Such cases mainly appear in so called “unfavourable areas” [15] which can generally be mostly avoided by a good positioning of the Remote Units. If ambiguities still appear, both solutions, the correct and the wrong one, need to be passed on to the next calculation steps. The ambiguity resolution has then to be performed in a later step, by taking additional measurements into account (see Chap. 3.4.3). It is, however, important to mark ambiguous solutions, as later calculation steps need to be aware of that fact.

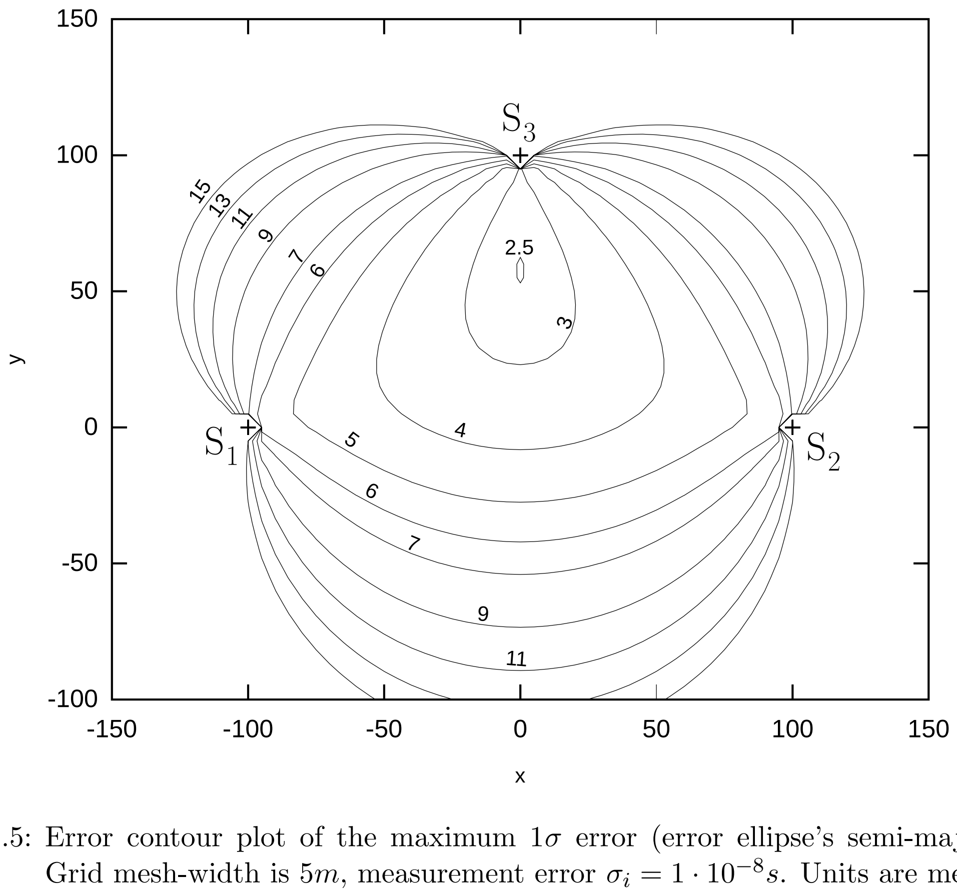


Figure 3.5: Error contour plot of the maximum 1σ error (error ellipse's semi-major axis). Grid mesh-width is $5m$, measurement error $\sigma_i = 1 \cdot 10^{-8}s$. Units are metres.

Whereas it is trivial to calculate a closed solution, it is more important to carefully choose the measurements which are used as input for the calculation. Figure 3.5 shows the geometrical dependence of the closed solution accuracy, which was determined by calculating an error propagation as explained in Chapter 2.5.2. The contours form the boarder of the maximum solution error σ_{max}, which is the 1σ error in the direction of the error ellipse's main diagonal. Outside the contours the error is bigger than 15 metres.

In between the three remote units the accuracy is quite good, which means that a closed solution, calculated for such a location, is not affected by big scatter. Outside the triangle, built up by the three units, the solution error grows rapidly with the distance. This is due to the glancing intersections between the hyperbolas, which form the solution (see Chap. 2.5.2). The worst accuracy appears in outer areas where solution and Remote Units lie more or less on one line. These areas are newly the "unfavourable areas" in [15]. Simulations have shown that also in the three dimensional case the best solutions are determined in the area in between the remote units.

In Multilateration systems normally a signal is not just measured at the minimum amount of Remote Units, and so there are n choose k possibilities to group the RUs in order to calculate closed solutions. n here denotes the number of available measurements and k the number of measurements which are necessary to calculate a closed solution ($k = 3$ in 2D, $k = 4$ in 3D). As each input data combination builds up a different geometrical scenario, there are combinations which lead to better measurement accuracies than others for a given location. So it is important to guess the input data combinations which lead to good results, to only calculate closed solutions for this subset afterwards.

In order to choose good combinations we can define two main criteria: Figures 2.8 and 3.5 show that it is preferable to have the target surrounded by the base stations, as the error of the result is smallest in this area. Additionally, the probability of multipath propagation and reflections is smaller if measurements are used which come from base stations that lie close to the point of signal emission. Accordingly, our goal is to find input station combinations where the mobile is most probably surrounded by Remote Units which in addition are close to it.

The only problem is that before the actual calculation the target position is not known in most cases. Often, however, some vague a-priori knowledge of the target position is available that can be used instead. Depending on the availability of this knowledge, two major cases have to be taken into account when guessing good measurement combinations:

- The target position is approximately known from an auxiliary source, as e.g. Mode-S position information or tracking of previous calculations.
- The target position is completely unknown.

Approximately known mobile position: In this case the problem is trivial: The best combinations are those where Remote Units surround the approximate mobile position and are close to it. If the potential mobile position is not surrounded by any combination of base stations, a combination can be chosen where the mobile does not lie on one line with two base stations, i.e. is not situated "behind" any Remote Unit (see Figure 2.8).

Totally unknown mobile position: In case of completely unknown target position, the best combination guessing can be performed in the following steps: First, closed solutions for the station combinations which enclose the biggest area are calculated in order to increase the probability to surround the mobile by these stations. In a second step these solutions can be used to guess other combinations until a good combination is found. If non-trivially solvable ambiguities appear, other combinations can be tried out until a resolvable ambiguity is received. This makes sense as non-trivially resolvable ambiguities mainly appear in areas where solution accuracy is quite low anyway (see Chap. 2.2.2). If this also fails, some random ambiguous solutions have to be taken and the ambiguities need to be resolved in a later step as explained in Chapter 3.4.3.

It also makes sense to calculate more than one closed solution from different good measurement combinations and to perform the subsequent calculation steps in parallel for each of these. The reason for that is again the solution error: In closed solution calculation, each measurement error has direct influence onto the result, which then may be affected by substantial error and scatter to one direction. By using other input data combinations the solution error will turn out differently and maybe scatter to other directions. The subsequent calculation steps are basically just refining this primary solutions and use closed solutions as starting points for their optimisations. Depending on this starting point, the refinement might get stuck in a local minimum or maximum. So, the risk of wrong convergence is reduced, if the following steps are performed for more closed solutions and if, finally, the best result of these is picked (see Chap. 3.4.3).

3.4.2 Adjustment

The adjustment which follows the closed solution calculation is needed to improve solution accuracy. This is done by taking redundant measurements into account and calculating a Least Square (LS) optimisation which starts with a closed solution. If no redundant measurements are available, however, it makes no sense to calculate an adjustment, since the optimum solution is already provided by the closed solution in this case.

Adjustment theory itself was already well explained in Chapter 2.3 and the adjustment step is basically just an application of the equations presented in these sections. As the adjustment is calculated in iterations, it is important to define good abort conditions, in order to keep the calculation time small. Experiments have shown that it is a good choice to stop the adjustment iterations if the change of the solution position from one step to the next is below a certain value. This forms e.g. the abortion condition $|d\vec{x}| < 1m$. Additionally it is advantageous to also limit the maximum number of iterations: This is needed in cases with bad measurements or if a bad closed solution is being refined, since in such cases the adjustment may not converge soon, or not converge at all. Ten iterations should be a good choice for this value.

Adjustment theory also provides the possibility to assign weights to each measurement in order to give measurements with high accuracy also a bigger influence and vice versa. So it is useful to know an a-priori measurement accuracy of each Remote Unit and to calculate the measurement weight matrix like explained in formula (2.23). The a-priori measurement error has to be determined in advance for each Remote Unit and depends on the RU's hard- and software, the location of its antenna, wiring etc.

As explained in Chapter 2.3.2, also an outlier detection is possible by calculating a robust adjustment. The principle of a robust adjustment can be seen as follows: All measurements are allowed to be affected by a certain, random measurement error with a known a-priori variance σ_i. If measurements are affected by gross errors, however, their influence is scaled by reducing their weights. The error, according to which the outlier decision is performed, is received by checking how well the measurements fit together in order to lead to the solution. If there are much more non-outlier measurements than outliers available, the probability to find the outlier is much higher, as the non-outlier measurements kind of outvote the outlier. So it makes sense to use outlier detection only if overdetermination is high.

After the convergence of a robust or non-robust adjustment, besides the refined position, additional data about the system and the solution is received. This data can be used to classify the quality of the solution and also to calculate an error propagation. The next two chapters explain how this can be performed.

3.4.3 Solution Quality

As discussed in the previous chapters, it is sometimes helpful to determine more than one closed solution from a measurement tuple, because closed solutions may be affected by excessive scatter. In this case all these solutions are also adjusted and afterwards the best solution has to be determined. This chapter presents a methodology to identify this best solution, which can also be used to resolve non-trivially resolvable closed solution ambiguities.

First we need to figure out what a good solution refers to. Obviously the best measure to identify the quality of a solution $\vec{x}$ is its deviation from the point of signal emission $\hat{\vec{x}}$, calculated like $\vec{\varepsilon_x} = \vec{x} - \hat{\vec{x}}$. As the input measurements $\vec{m}$ are associated with the solution over $\vec{m} = \mathbf{A}\vec{x}$, the measurement error $\vec{\varepsilon} = \vec{m} - \hat{\vec{m}}$ contains the same quality information as the solution error $\vec{\varepsilon_x}$ does. So we can say that a good solution is a solution which was derived from measurements with a small measurement error $\vec{\varepsilon}$. Since the real measurement error $\vec{\varepsilon}$ is normally unknown, we have to use its best approximation which is the vector of residuals $\vec{v}$ (see Chap. 2.5).

In order to be able to classify the quality over one parameter per solution, it makes sense to use the sum of squared measurement errors, calculated like $\vec{\varepsilon}^T\mathbf{P}\vec{\varepsilon}$ or $\vec{v}^T\mathbf{P}\vec{v}$ respectively. However in this case the quality is just comparable if the solutions were determined by the same amount of measurements. To overcome this problem, the squared error can be divided by the factor of overdetermination which finally leads to the a-posteriori measurement variance $s_0 = \frac{\vec{v}^T\mathbf{P}\vec{v}}{n-d}$, like already shown in equation (2.50). Therefore, after all solutions were refined, the solution with the smallest a-posteriori measurement variance s_0 can be chosen as best solution for the measurement tuple.

It has been observed that in most cases, however, the refinement of different closed solutions converges to the same adjusted solution, i.e. to the absolute extremum. Of course in this case also the s_0 values are the same, but it does not matter which solution is picked finally. s_0 just differs significantly if the Least Square optimisation converged into a local extremum, though, just in this case a decision over the solution quality is necessary.

Non-trivially resolvable closed solution ambiguities can also be solved via s_0: If both ambiguous solutions are refined, in the best case the refinements converge to the same solution. If this happens there is no need to identify the wrong solution anymore. Often, however, the wrong closed solution lies far away from the point of signal emission and can cause the LS optimisation to get stuck in a local extremum. Also here s_0 can be taken to identify the wrong solution and to resolve the ambiguity. However, as discussed already in Chapter 2.5.1, s_0 is just defined if the system is overdetermined, i.e. if more than the minimum amount of measurements is available. This means that non-trivially resolvable ambiguities can just be resolved if overdetermination is given.

3.4.4 Solution Accuracy

Especially in aviation it is important to know the accuracy of a determined position. This accuracy can be determined over the error estimation as it was explained in Chapter 2.5 already. It makes sense to provide the following parameters with each solution:

- a-posteriori measurement error (see Chap. 2.5.1)
- a-priori and a-posteriori solution error (see Chap. 2.5.2)
- Dilution of Precision, especially HDOP (see Chap. 2.5.3)

The a-posteriori measurement error s_0 was already needed in the calculation, so it is available with every adjusted solution anyway. The solution error measures are calculated over the error propagation explained in Chapter 2.5.2. Depending on whether the a-priori or a-posteriori measurement error are used in the propagation, also the a-priori or a-posteriori solution accuracy is received. Both solution errors and the Dilution of Precision are determined from the cofactor matrix $\mathbf{Q}$ which was determined in the adjustment already.

All three values can be used to classify the solution quality. This is important as solutions with bad accuracy have to be recognised and eventually be dropped in order to satisfy the accuracy requirements of the system. The accuracy values can be interpreted as follows:

A-posteriori measurement error $\mathbf{s_0}$: This error describes the mean error of the measurements. A small error s_0 is a sign for a good result, as the measurements fit together well for the given solution.

A-priori solution error and DoP: These values depend just on the systems geometry and the position of the solution. Because of this, they can also be determined for closed solutions and form the only available accuracy information for them. A maximum a-priori solution error constraint is useful to avoid the adjustment of closed solutions, where the solution does not lie in a good area for the given Remote Unit constellation (see Figure 3.5). Furthermore they can be used to specify the system's area of interest. The DoP and the a-priori solution error describe basically the same: By comparing equation (2.60) and equation (2.61) it becomes clear that the a-priori solution error can be seen as a DoP value, scaled by the a-priori measurement error.

A-posteriori solution error: This error depends on the geometry and the measurement quality. Because of this it fuses the information of the two error measures explained before; However, it is just available for overdetermined measurement constellations. A maximum a-posteriori solution error constraint can be used to restrict the solution scatter of the emitted calculation results.

4 Implementation

This chapter explains some implementation details of the MLAT Target Processor prototype, which was developed along with this work. Some of the algorithms, especially the calculations, error propagation etc. were firstly implemented in GNU Octave(1) scripts in order to get in touch with them and to test them. The implementation of the prototype itself was written in C++, using the Trolltech Qt 3.3(2) framework and the AviBit proprietary AVCommon library. Matrix and vector calculations, mainly needed in the calculator unit, were implemented with the use of the Eigen 2.0(3) library. This library performs well, supports various compilers, is easy to include into a project and elegant to use (see [28]).

The implementation details in the following sub-chapters are completed by calculation results which were derived from measurements recorded in a real field test. The chapters are divided like follows: Chapter 4.1 shows the overall process structure and data flow. The field test setup and the single measurement scenarios are explained in Chapter 4.2 “Measurements”. Finally Chapters 4.3 to 4.6 explain the implementation of the single process modules, starting from measurement input and heading to the calculation of the sender’s position.

At this place it has to be mentioned that the system presented here is a Target Processor without tracking facility. This means that there is no target history kept which can be used to facilitate the calculation in e.g. ambiguity resolution or closed solution determination. However, this makes sense in the prototype state, as the characteristics and problems of the MLAT calculation can be explored anyway. Furthermore, also an operational system has to be able to work without history information: in the track initiation phase it is essential to determine good position fixes also without auxiliary information about the target position.

4.1 General Process Structure

The MLAT calculation problem is dividable into five main tasks as discussed in Chapter 3 already. It was found reasonable to implement each of these tasks in different threads in order to separate them totally and make the process scalable on multiprocessor or multicore architectures. The Qt 3.3 Toolkit was used for this purpose, as it provides good possibilities to implement threading and concurrent programming over the `QThread` and `QMutex` classes.

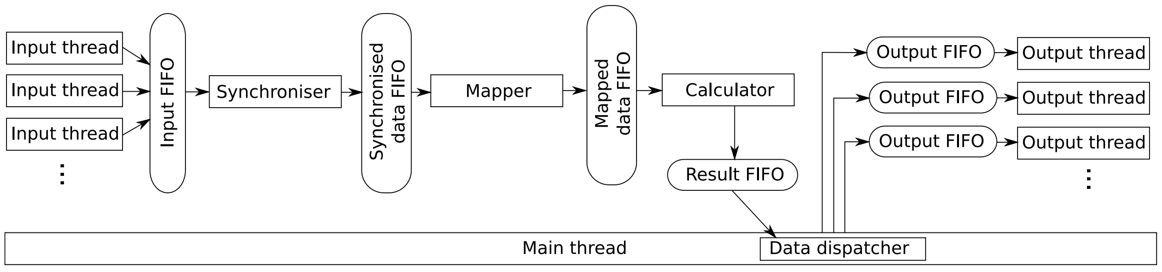


Figure 4.1: Threads and data flow in the TP prototype

(1) `http://www.gnu.org/software/octave/`

(2) `http://doc.trolltech.com/3.3/index.html`

(3) `http://eigen.tuxfamily.org`

Figure 4.1 shows the thread and data flow structure of the process. Input data has to be collected from every Remote Unit (RU) and the process needs to be flexible for different measurement setups. For this reason the amount of input threads, where each serves one RU, is freely configurable over configuration files. Synchroniser, Mapper and Calculator have just single instances in the process. The data output needs to be provided in different formats and so also multiple output threads can be configured. Each output thread gets the result data from the Calculator and emits output data in a different data format e.g. plain-text file or Eurocontrol Asterix.

Data flow between the single threads is regulated over FIFO buffers which are basically derivations of the `QPtrQueue` class, supplemented by locking facilities, using `QMutex`es. So e.g. there is an input FIFO which is filled by all input threads and is read out by the Synchroniser. The Synchroniser itself fills the synchronised data into the synchronised data FIFO which is read out by the subsequent mapper and so on (see Figure 4.1). These connections are realised directly from worker-thread to worker-thread and so the main thread is not bonded with data distribution. The only exception is build by the output FIFOs: As more output threads are possible and each of these needs the whole amount of result data, the main thread dispatches and copies the calculation results, in order to provide it for each thread.

Due to the task separation into different threads and the FIFO buffer interface it is possible to make the process highly configurable and to replace different modules by other implementations. So e.g. input threads can be switched between file input and live data (TCP/IP) input, Calculators between 2D and 3D calculations etc. Additionally each such module/thread has its own configuration file; Hence, the possibility to tune calculation and data flow parameters is provided.

4.2 Measurements

The most recent measurements and system tests, when this thesis was being created, were performed in the ORF Park in Graz, Austria. For this purpose five Remote Units were positioned around the park as shown in Figure 4.2. Each unit was powered by a car battery and the collected data was recorded on a laptop computer directly at each unit. Data evaluation, combined with prototype improvement and tuning, was performed in a later step on the basis of this recording files.

The vehicle locator Sensis VeeLo (see [29]) was used in the measurements as SSR transmitter to be localised. Such transmitters are intended to be mounted on vehicles at airports, in order to localise them over SSR Mode-S and ADS-B. For this purpose they broadcast SSR replies constantly, without being interrogated. Their transmitting power is restricted to $20W$, as they are built for low range detection purposes. The vehicle locator, which was used in the measurements, is equipped with a GPS module and, depending on whether it is in movement or not, it broadcasts its GPS position each 0.5 or 5 seconds in SSR Mode-S data format DF 18[(4)] [29].

Figure 4.2 shows the measurement setup: The Remote Unit positions are labelled with the

[(4)]for explanation of SSR Mode-S DF 18 see [20] and [22]

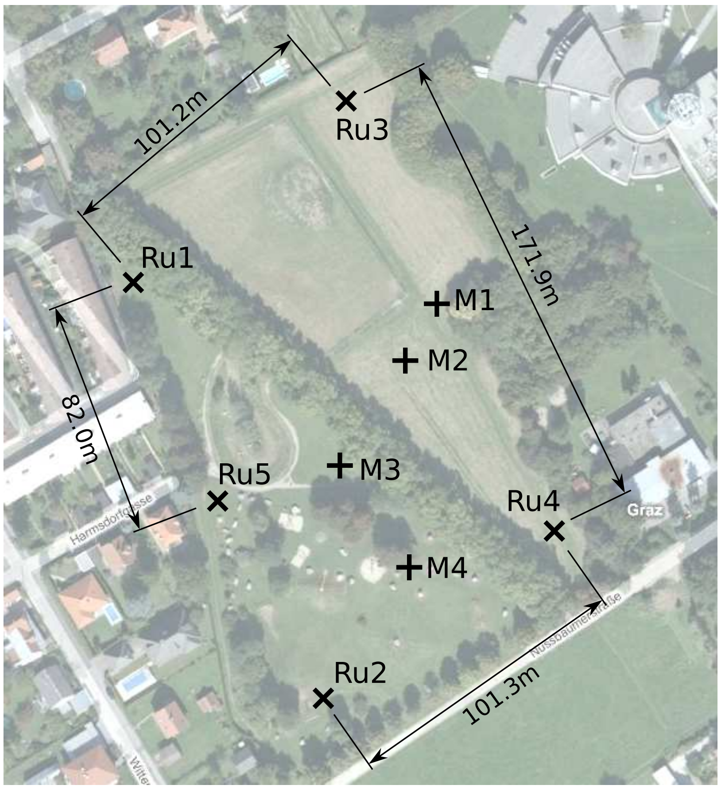


Figure 4.2: Measurement setup in ORF Park, Graz Austria. Background image taken and modified from [30]

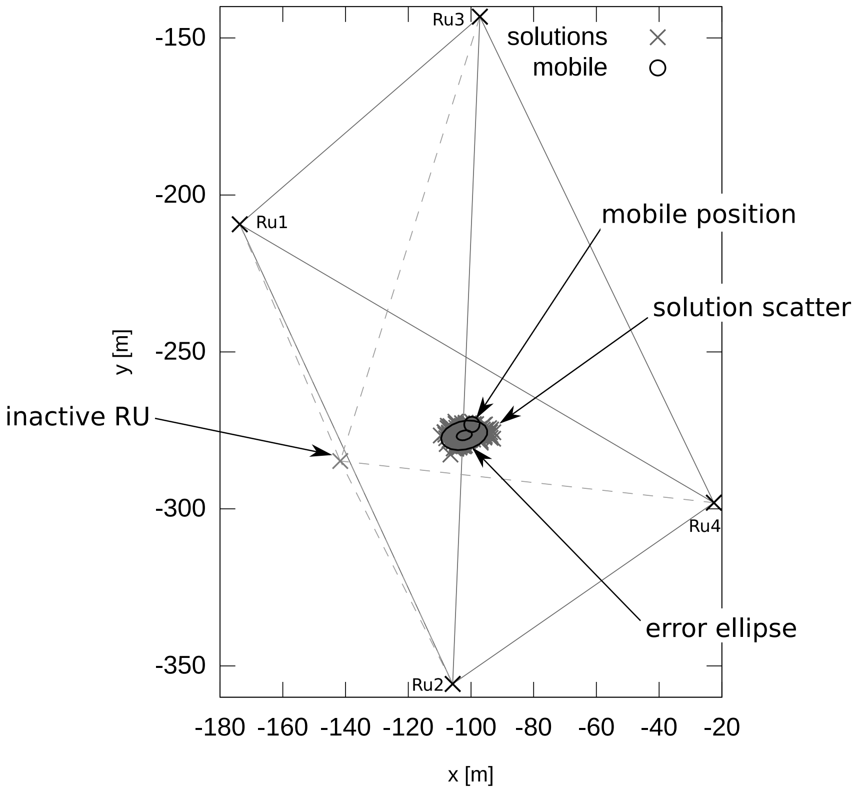


Figure 4.3: Explanation of the calculation result representation for mobile at M3 and inactive Remote Unit Ru5

prefix “Ru” followed by the RU-number. The positions at which the transmitter was located during the measurements have the prefix “M”. The transmitter was situated for around 15 minutes at each measurement point, in order to let it emit approx 1000 replies. The coordinates of measurement points and RU positions were determined by the GPS modules on the Remote Units. For this purpose the GPS position fixes were averaged for around 30 minutes directly on the receiver module, which was set into survey-in mode. Finally the measurement errors lay between $\sigma = 0.1m$ and $\sigma = 0.16m$. Table 1 contains all relevant coordinates of the field test in WGS84 format.

Point	**Latitude [°]**	**Longitude [°]**	**Geod. Height [m]**	σ **[m]**
Ru1	47.0540656	15.4635945	408.02	0.146
Ru2	47.0527487	15.4644880	409.77	0.166
Ru3	47.0546605	15.4646018	407.29	0.118
Ru4	47.0532679	15.4655837	408.29	0.118
Ru5	47.0533871	15.4640163	406.58	0.137
M1	47.0540173	15.4650109	408.13	0.146
M2	47.0538246	15.4648577	403.21	0.148
M3	47.0534922	15.4645686	408.02	0.174
M4	47.0531646	15.4648857	407.37	0.111

Table 1: RU and measurement point positions in the ORF Park test in WGS84 coordinates

The obtained recording data builds the input for the measurement evaluation by the MLAT Target Processor prototype. In this scenario there is almost no height difference between the Remote Units, so the calculation is limited to two dimensions. This is due a lack of height information in the calculation’s equation system, causes a determined height value being affected by excessive error. Furthermore it is mostly impossible to calculate a three dimensional solution in this scenario as, because of the limited numeric precision, matrices often become singular in the calculation.

In order to visualise the calculation outcome, the results were firstly written to a plain-text file by the TP process and then read in by a GNU Octave script. This script performs some data post-processing, like data selection and coordinate transformation, and finally plots the results. Figure 4.3 shows an example plot for measurement point M3 with results determined from data recorded on Ru1 to Ru4. In such plots the active Remote Units are displayed as black crosses and are interconnected by solid lines in order to display the triangles they span. Inactive RUs and their spanned triangles are displayed also in order facilitate keeping track of the geometry. They are plotted in grey and as dashed lines, though. The result scatter itself is plotted as dark grey crosses, sometimes overlaid by an error ellipse (see Appendix A.4) which shows the 1σ and 3σ bounds of the scatter. The transmitter’s GPS position is displayed as black circle.

The measurements in the park were facing a big problem, however: The obstacles, like small hills, trees or fences were influencing the signal propagation, mostly by shadowing. It was also observed that GPS accuracy suffered from this fact since, depending on the location, certain satellites were not visible.

4.3 Data Collection

As already mentioned in Chapter 4.1 “General Process Structure” the TP’s input interface has to be configurable in order to adopt to different measurement scenarios. For this purpose the communication with the RUs is handled by one input thread per unit, and the amount of threads is configurable. These threads are able to receive data in two modes: live data over TCP/IP or data from recording files. The recordings basically contain the messages created by a Remote Unit in binary format, extended by recording timestamps.

The recording file input mode additionally supports the possibility of time scaling i.e. to read out the file faster than the recording time stamps pretend. This is useful during process development and tuning, as many calculations can be performed within short time and result evaluation times become shorter as well. In the subsequent processing steps, however, one has to be aware of the time scaling possibility: Timeouts, as used in measurement correlation and synchronisation, have to be independent of the current machine time and need to be adopted according to the time scaling factor.

As the process design intends to support different input message formats, the input threads translate the received messages into a process internal measurement format which is then used by the subsequent modules. This data is then fed into the input FIFO for later synchronisation, as shown in Figure 4.1.

4.4 Synchronisation

In the current MLAT system prototype each Remote Unit is synchronised to UTC time, provided by the GPS satellites. Therefore, each RU is equipped with a GPS module and generates synchronisation messages every second, as described in Chapter 3.1. These messages, which contain timestamp–UTC-time pairs, are then evaluated by the Target Processor, that determines the synchronisation parameters for each RU by calculating a linear regression (see Chapter 3.2.2 “Synchronisation over GNSS”).

For this purpose a synchronisation message history is kept per Remote Unit, because the regression is calculated from previously received and the current synchronisation information. The synchronisation parameters are calculated by a factory class which takes the synchronisation event history and the current synchronisation message as input data. Of this information it calculates the synchronisation parameters via the regression formula (A.19). Finally, the currently valid synchronisation parameters for each RU are stored in a hash table in order to provide quick look-up for measurement synchronisation.

During the implementation it has been observed that the regression was suffering from numerical problems. The reason for that lies in the dimension difference of counter and time values: The time value is incremented by one each second, the counter value by approximately 80 millions, as the oscillators with operate at 80MHz. The problem was solved by scaling the input data before the regression is calculated: First, in order to avoid large numbers, the time and counter values are subtracted by their corresponding values taken from the oldest history event. In a second step the counter value is divided by the rated oscillator frequency to bring time and counter into the same scale, i.e. unity. Of course, this scaling must also be respected if the synchronisation parameters are later on used to synchronise measurements.

Figure 4.4 shows a typical developing of the measured oscillator frequency for over almost two hours, recorded on Ru2 during the ORF park test. The frequency was determined by subsequent synchronisation events: The counter values were subtracted and divided by the time increment between the events, giving a counter increment per second Δz_{1s}. The plot contains the relative error of the measured frequency related to the rated oscillator frequency of $f_{rat} = 80MHz$. This error was calculated for each Δz_{1s} like follows:

$$f_{err} = \frac{\Delta z_{1s} - f_{rat}}{f_{rat}} \tag{4.1}$$

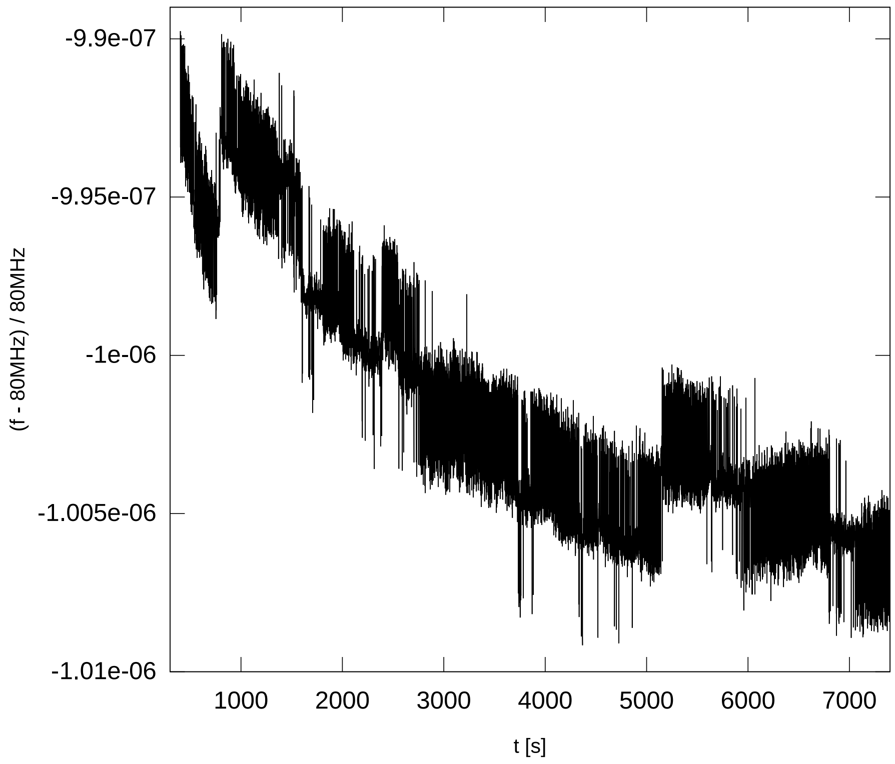
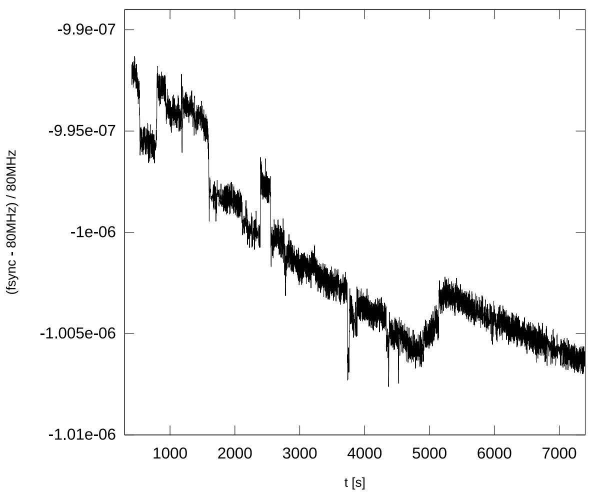


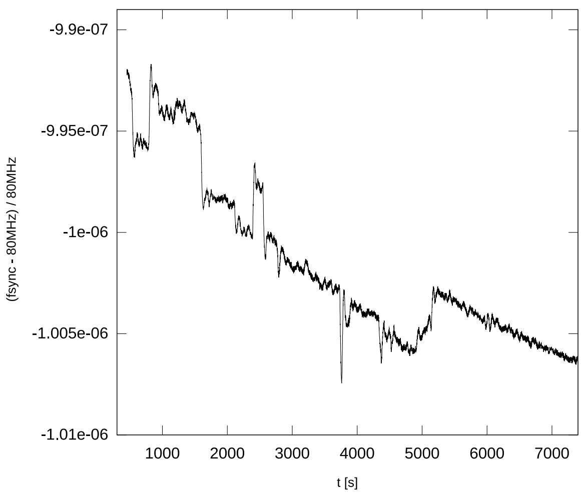
Figure 4.4: Developing of the frequency error relative to $80Mhz$ at Ru2 in the ORF Park test

The frequency error in the figure drifts approximately between $-9.9 \cdot 10^{-7}$ and $-10.1 \cdot 10^{-7}$ which gives a relative drift during this two hour period of $2 \cdot 10^{-8}$. This equals a time drift of $20ns$. The stability of the OCXO is given as $\sigma < 1 \cdot 10^{-10}$ (one second Allan deviation) and the daily aging as $5 \cdot 10^{-10}$ in the datasheet [25]. These values correspond to $0.1ns$ and $0.5ns$ respectively and are factor 100 smaller than the frequency drift. Consequently the observed frequency drift seems to mostly belong to GPS timing inaccuracies. Unfortunately, at the time this work was created, there were no better possibilities available to analyse the oscillator, PLL and GPS errors. Because of this, it was tried out in the synchronisation to at least get rid of the noise in Figure 4.4 that causes the graph to be bold.

Depending on the amount of history synchronisation events which are respected in the regression, the influence of the GPS error can be partly reduced. The plots in Figure 4.5

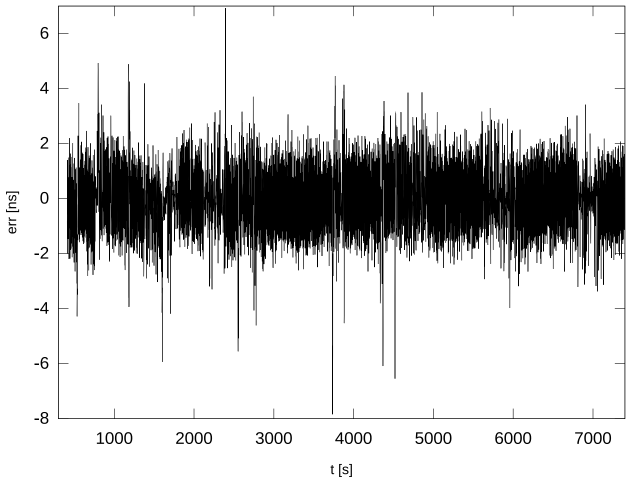


(a) History of 10 elements

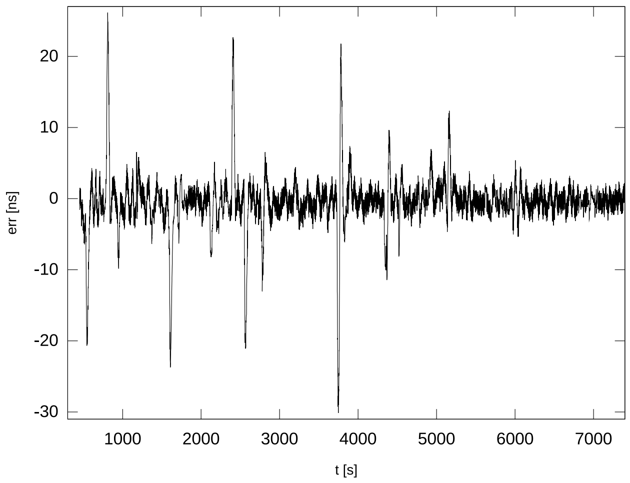


(b) History of 50 elements

Figure 4.5: Influence of synchronisation on the frequency



(a) History of 10 elements: mean=$-0.027ns$ $\sigma_{1s} = 1.23ns$



(b) History of 50 elements: mean=$0.47ns$ $\sigma_{1s} = 3.98ns$

Figure 4.6: Influence of synchronisation on the one second regression error. Note the different scaling of the axis of ordinate!

show the relative frequency error after the synchronisation. This frequency error was newly calculated via equation (4.1), the value Δz_{1s} was determined differently, though: Instead of using the UTC time which is provided by the RU together with the corresponding counter, the synchronised time values $t_{s,i}$ were used to calculate Δz_{1s}. To determine this synchronised time, first the synchronisation parameters were determined by calculating the regression over 10 or 50 history events, including the current event. These parameters were then used in order to calculate the synchronised receive time $t_{s,i}$ for the current counter value, via equation (3.2).

It was also found useful to investigate the resulting error if the synchronisation parameters are used to extrapolate over one second. For this purpose the regression was calculated as explained before, but the synchronised time $t_{s,i}$ was calculated for the synchronisation event coming after the current, i.e. the synchronisation parameters were used to extrapolate over one second. This synchronised time was then subtracted by the UTC time of the corresponding event, provided by the RU, in order to determine the error which is shown in Figure 4.6.

Figure 4.5 makes clear that with increasing history size the noise on the synchronised frequency shrinks. This is because more history elements cause the curve being flattened. The 1s error in Figure 4.6 however increases if the history is chosen too large: A history of 10 leads to a 1s-error standard deviation of $\sigma_{1s} = 1.23ns$. With 50 history points the error increases to $\sigma_{1s} = 3.98ns$; Also if outliers outside $\pm 15ns$ are removed the deviation is still $\sigma_{1s} = 3.15ns$ in this case. The reason for this lies in the jumps the measured frequency performs and the longer the history is chosen, the slower a the synchronisation reacts on them. Evaluations of different history sizes have shown that a history value of 10 is a good trade-off between frequency noise and extrapolation error in the current system configuration. Thus, a synchronisation history of 10 events was used for all calculations presented in this work.

4.5 Measurement Correlation

Experiments with live data have shown that the best way to map all measurements of a signal together is a combination of the two strategies explained in Chapter 3.3. It is advantageous to firstly do a correlation over message content in order to minimise the probability that different transponders disturb each other. A good choice for this purpose is to use two hash tables, one for Mode-A/C and one for Mode-S replies, in order to store replies depending on their content. As SSR replies are quite short, the hash value can be calculated directly from the message content i.e. the 12 bit Mode-A/C data or the 56 or 112 bit Mode-S data block. All messages with the same message content are then stored in a list, in order to be further processed.

The communication between Remote Units and Target Processor is afflicted by a communication delay which depends on the communication interface. For data integrity it is important that every measurement, belonging to one signal emission, was received and stored on the content correlation list before the subsequent time correlation is performed. So at least the estimated RU–TP communication delay has to be waited after the first message for a certain content mapping was received. The TOA correlation can then be performed by defining the two timeouts as proposed in Chapter 3.3.2 and by applying them as follows, one after another:

1. **Detection timeout** Δt_{det}: Replies received at different Remote Units and within Δt_{det} belong to one signal emission.
2. **Reflection timeout** Δt_{refl}: Replies received at the same Remote Unit and within Δt_{refl}, are a reflected copy of the first received reply.

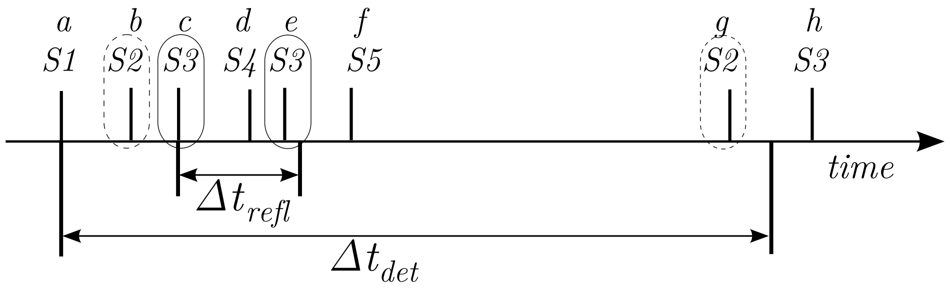


Figure 4.7: Example of signal receiving times and correlation timeouts

Figure 4.7 shows an example of seven measurements (a to h) which have the same message content and were received at the base stations $S1$ to $S5$. The measurements a to g lie within the detection timeout Δt_{det} and are added to the set of measurements which might belong to the same signal emission. Measurements d and g but are a second occurrence of measurements at stations $S2$ and $S3$ and since rule 1 says that the replies have to be received at different units, they cannot belong to the same signal emission. Furthermore measurement d was received within the reflection detection timeout Δt_{refl} regarding measurement c - so according to point 2 it is denoted to be a reflection and is dropped totally. Measurement g is not a reflection, but may belong to a subsequent signal emission. So it is put back into the measurement list and will be retaken into account in the next detection phase.

4.6 Calculation

This chapter presents the implementation of the calculation module, completed with a sample of intermediate and final calculation results in order to explain the single calculation steps. The implementation details are explained in the order the calculation steps are performed; these steps were already shown in a simplified manner in Figure 3.4.

The implementation of the calculator was split up into two main modules: The control logic and the `TPCoreAlgorithms` module. The algorithm module contains implementations of the MLAT algorithms in Chapter 2 and is implemented as a class template which takes the dimensionality, i.e. 2D or 3D, as template argument. This was found advantageous as the dimensionality mostly just influences vector and matrix sizes, and large parts of the algorithm code are the same for both dimensions. All Matrix calculations in this module were implemented utilising the Eigen 2.0 library which itself is solely implemented as C++ templates, thus it facilitates the dimensionality switching over template parameter.

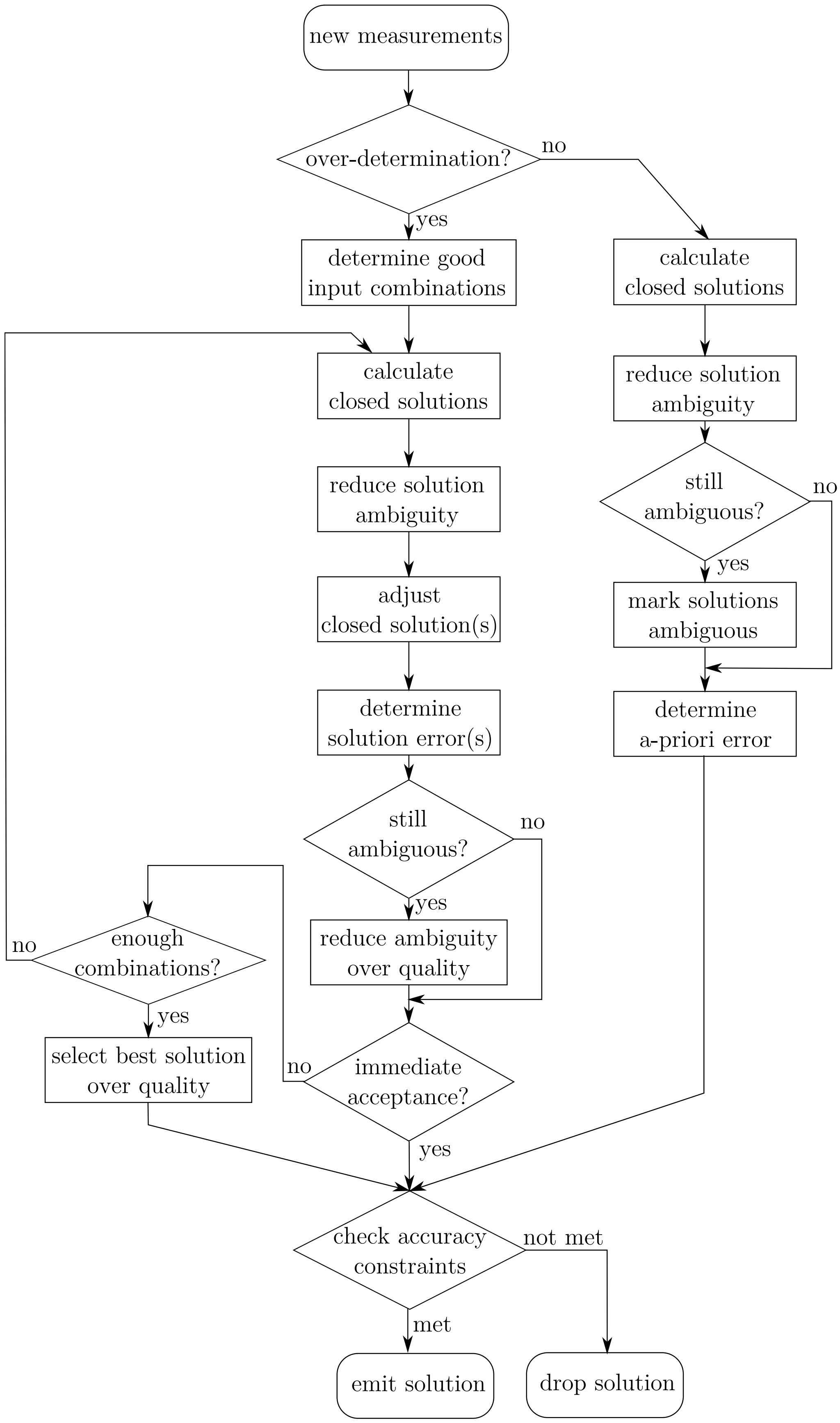

new measurements
over-determination?
no
yes
determine good input combinations
calculate closed solutions
calculate closed solutions
reduce solution ambiguity
reduce solution ambiguity
still ambiguous?
no
yes
mark solutions ambiguous
adjust closed solution(s)
determine solution error(s)
determine a-priori error
still ambiguous?
no
yes
reduce ambiguity over quality
enough combinations?
no
yes
select best solution over quality
immediate acceptance?
no
yes
check accuracy constraints
not met
met
emit solution
drop solution

Figure 4.8: Complete flowchart of the Multilateration calculation

Whereas the algorithms were explained already quite well in the previous parts of this work, this chapter puts more focus onto the control logic. This logic defines how and in which sequence the algorithms are applied, which results are accepted finally, the number of calculated iterations etc. As there are many possibilities to implement this logic, during the work on the prototype different attempts were started to implement it. The presented control logic emerged out of the pool of approaches and gives plenty of freedom for configuration and tuning. The flow-chart in Figure 4.8 illustrates the logic.

The degree of overdetermination splits the chart in Figure 4.8 into two branches: A branch for overdetermined measurement tuple constellations, and one where no overdetermination is given. This division was also taken in order to separate the subsequent two chapters.

4.6.1 Calculation Without Overdetermined Input Data

If no overdetermination is given, calculation is easy: It is possible to calculate only one closed solution since just the minimum amount of measurements is available. Furthermore, adjustment has no sense, as there is no accuracy improvement possible without additional measurements. However, not overdetermined results have to be treated with precaution as they may be affected by excessive scatter, as the end of this section will show. The single steps in the flow chart are explained here.

Ambiguity resolution: According the right branch of Figure 4.8, firstly a closed solution is determined via Bancroft's algorithm (see Chapter 2.2.1). This solution is ambiguous and so the ambiguity is tried to be removed by applying the rules presented in Chapter 2.2.2 “Solution Ambiguity Resolution”. Therefore it is advantageous to firstly apply the simpler rules 1 (TOE check) and 3 (area of interest), as they basically just imply the comparison of the result with pre-configured values. Criterion 2 (TOA order) is more complicated to be applied: The current implementation first calculates the distances between closed solution and Remote Units and sorts these values in a list. Afterwards also the measurements are sorted by their TOA and the order of both sorted lists is compared of being equal.

Figure 4.9 shows a scenario with trivially resolvable solution ambiguities for measurement point M2, where the results were determined from measurements of Ru2, Ru3 and Ru4: The solution accumulation on the right side forms the correct solutions, on the left side we have the wrong solutions. By applying the ambiguity resolution rule 1 (TOE check), in this case all wrong solutions are eliminated.

As discussed in Chapters 2.2.2 and 3.4.3 already, the ambiguity is mostly resolvable by this trivial rules, but unfortunately there are also cases where they fail. Since there are no additional measurements available, a resolution can just be performed over an approximate knowledge of the target's position. This information could be provided e.g. over tracking or Mode-S position decoding, however, the TP prototype does not support such checks currently. So a solution with irresolvable ambiguity can just be tagged in order to warn subsequent steps about this fact.

After the ambiguity resolution, the a-priori solution error is determined for the location

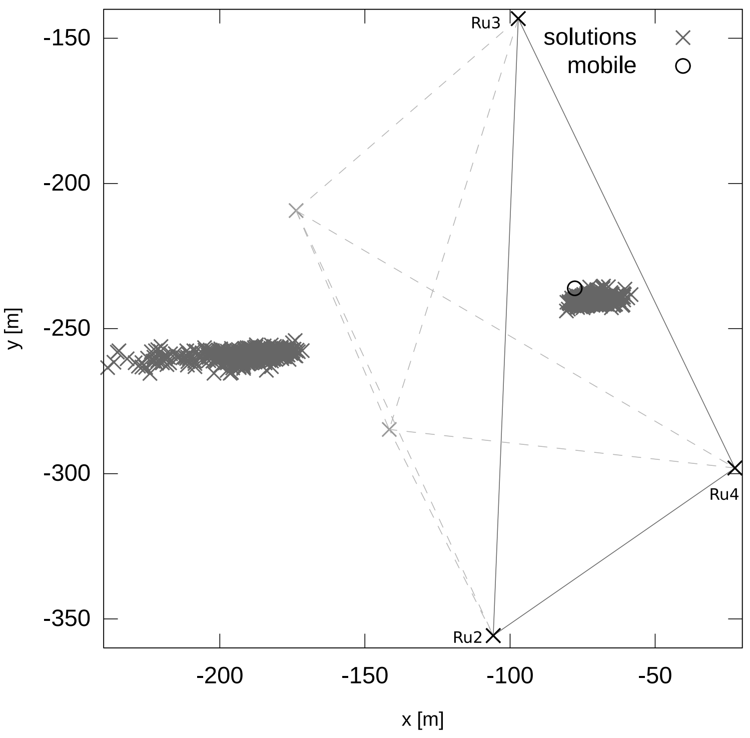


Figure 4.9: Solution with trivially resolvable ambiguity at M2

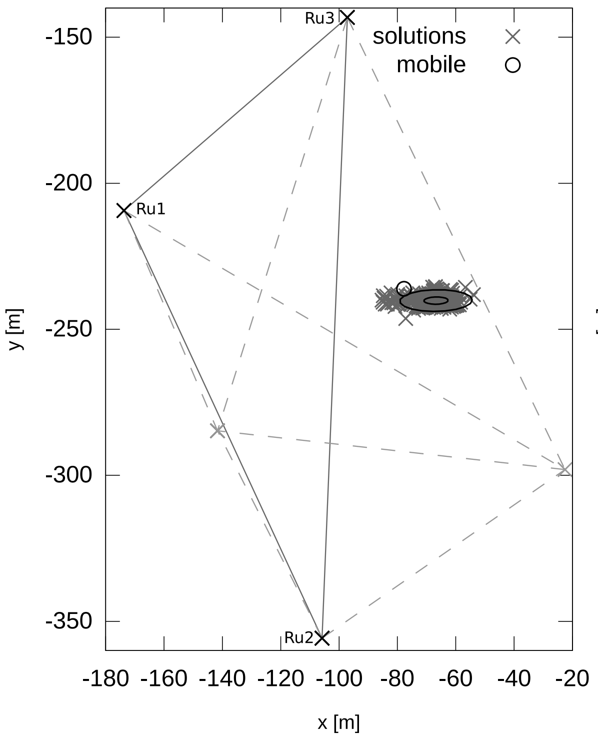
the closed solution points to. This is done over formula (2.60) on page 25. The matrix **Q** which is needed hereby, can easily be determined over adjustment equation (2.26c) on page 17.

Solution error: Closed solutions are quite problematic as they might be affected by strong scatter. The scatter results out of the fact that every measurement has direct influence onto the result. Furthermore, as already discussed in Chapter 3.4.1, the error strongly depends on the problems geometry: Figure 4.10 shows closed solutions for point M2, calculated of different Remote Unit combinations. It is well noticeable in Figure 4.10c that the solution error increases with its distance from the Remote Units. Unsurprisingly, the best results are achieved in Figure 4.10d, where the transmitter is surrounded by the Remote Units.

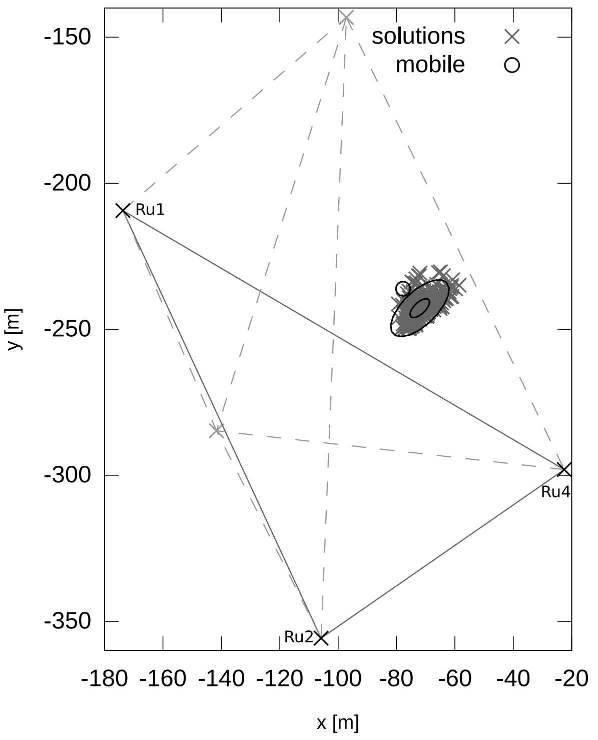
To prevent the emission of solutions which may be affected by strong error, the a-priori error can be used as drop criterion: The result shall just be emitted if its a-priori error lies below a certain threshold. This check is performed in the last calculation step where accuracy constraints are being verified.

4.6.2 Calculation With Given Overdetermination

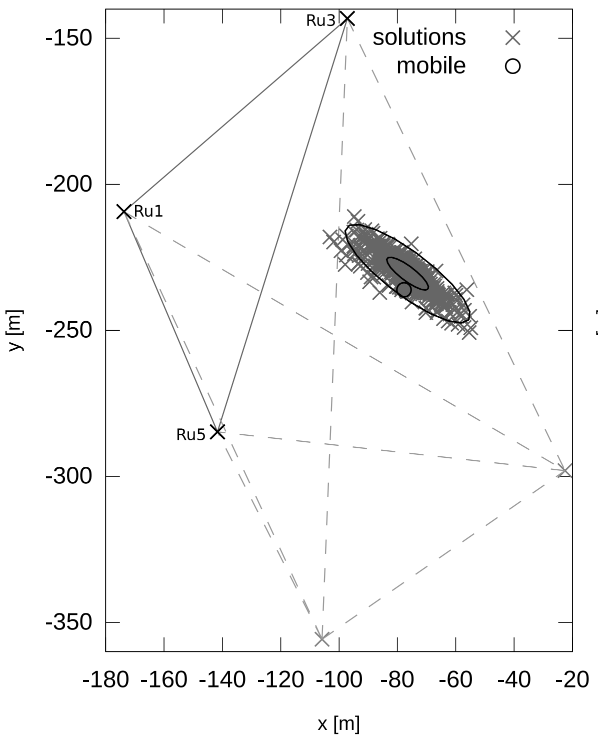
Although overdetermination in general brings the advantage of more accurate calculation outcomes, it also makes the calculation more complicated than in the non-overdetermined case. The central branch in Figure 4.8 shows the flow chart which illustrates how such a calculation is performed in the Target Processor prototype. The single steps are explained here.



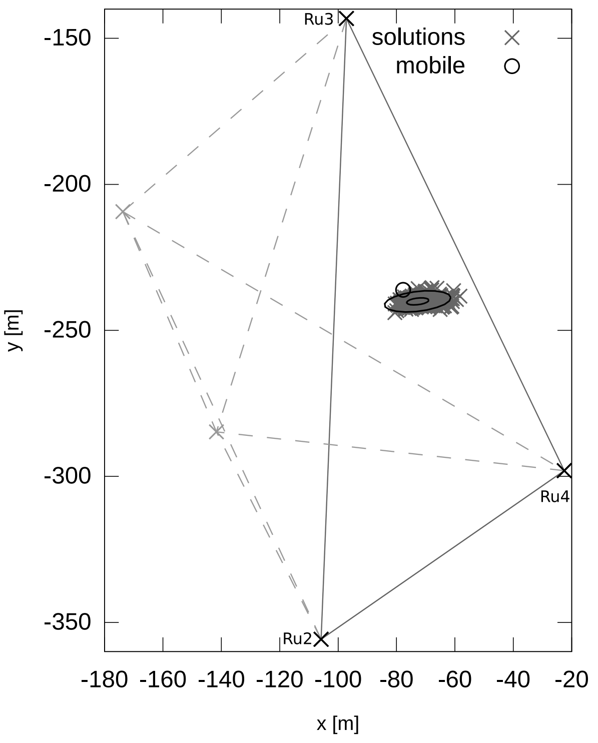
(a) RU combination (1, 2, 3)



(b) RU combination (1, 2, 4)



(c) RU combination (1, 3, 5)



(d) RU combination (2, 3, 4)

Figure 4.10: Closed solutions for measurement point M2

Determination of good input data combinations: Also an overdetermined calculation starts with closed solutions. As discussed in the design chapter already, there are n choose k (n = number of measurements, k = 3 in 2D, 4 in 3D) possible Remote Unit combinations which can be used in order to come to closed solutions. Depending on the geometry, formed by RUs and mobile, the solution quality of these solutions varies. Chapter 3.4.1 was explaining that it is advantageous to prioritise combinations with bigger ground area and so for each combination the enclosing ground area is calculated. Consequently a list of advantageous RU combinations is created by sorting the combinations according their ground area size.

Closed solution determination: At this point the iteration through the list of advantageous RU combinations starts. Therefore, the best found combination is taken from the list and its closed solution is determined. The subsequent trivial ambiguity resolution is performed as described in the previous chapter. If the resolution fails, however, this is no problem: Due to the overdetermination the ambiguity can be resolved in a later step.

Adjustment: In this step the closed solution is adjusted, or in other words refined. If the ambiguity was irresolvable before, an adjustment is calculated for both closed solutions. At this point it is important to mention that the refinement respects all measurements of the tuple, including the ones already used to determine the closed solution. The adjustment iteratively adapts the solution, in order to fit best to all measurements, starting with the closed solution. Figure 4.11 shows an example of the solution quality improvement, achieved by an adjustment: The light grey crosses describe the closed solutions which were determined from different RU combinations. The dark grey targets are the solutions after the adjustment, in which measurements from all five RUs were taken into account.

Depending on the configuration, a conventional or a robust adjustment is calculated in this step. Experiments have shown that a robust adjustment makes just sense if an overdetermination factor of at least two is given. Otherwise it can happen that a wrong measurement is being denoted to be an outlier and the overall calculation outcome becomes worse than without outlier detection.

Non-trivial solution ambiguity resolution: With the parameters determined in the adjustment, the a-priori and a-posteriori errors are determined as shown in Chapter 2.5.2. If the solution ambiguity was not resolvable before, the resolution can be performed now: Chapter 3.4.3 "Solution Quality" has shown that the resolution can be performed over the a-posteriori measurement variance s_0, by dropping the solution with bigger s_0. An example for this is shown in Figure 4.12: The light grey targets in both plots are the ambiguous closed solutions, obtained from the RU combination (1, 2, 5). On these results all three ambiguity resolution rules from Chapter 2.2.2 fail. The dark grey crosses are the solutions after the adjustment which in addition was respecting also Ru3. The left figure shows the solutions that were found wrong in the s_0 check, as they converged into a local extremum. The right figure is a plot of the solutions, which were converging correctly and were found correct also in the s_0 check.

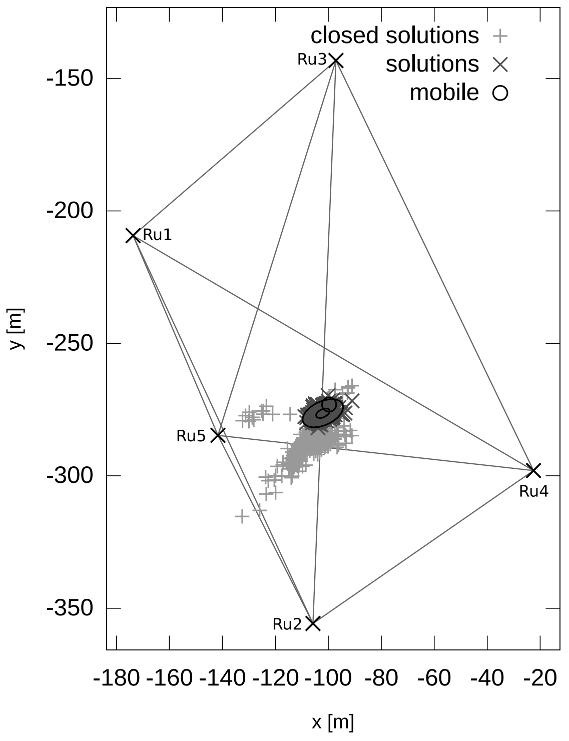
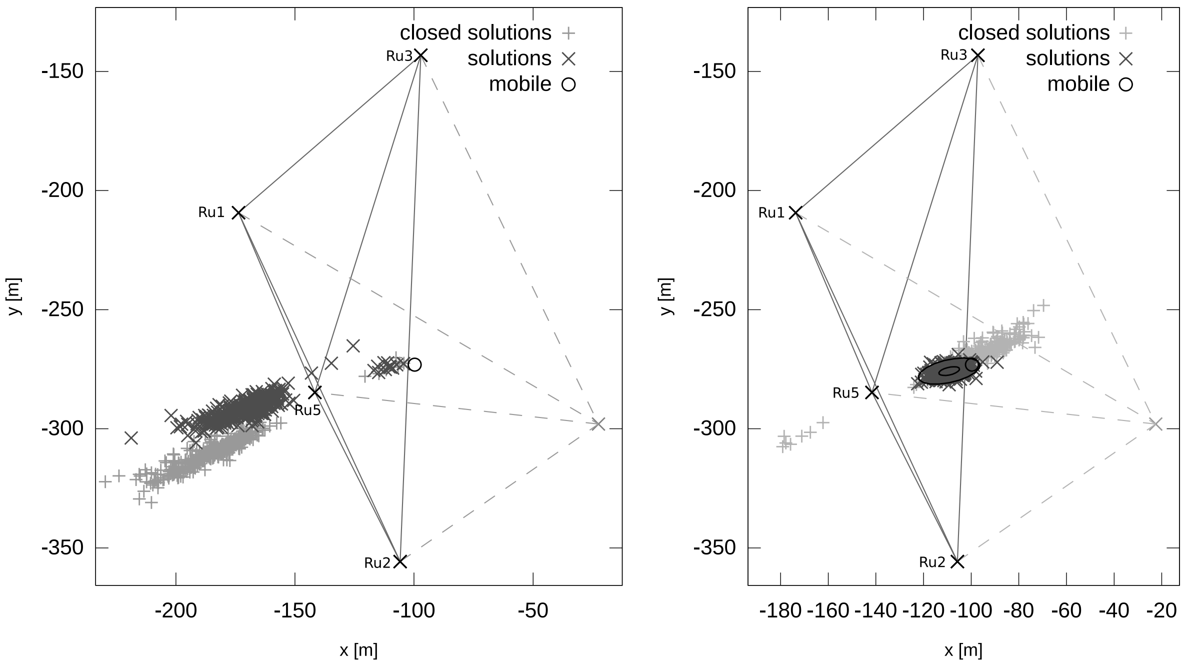


Figure 4.11: Effect of an adjustment with tuples of 5 measurements at measurement point M3. $\sigma_{max} = 2.74$



(a) Solutions found wrong

(b) Solutions found correct ($\sigma_{max} = 4.42$)

Figure 4.12: Non-trivial ambiguities of closed solutions at M3. Closed solutions from combination (1, 2, 5). Resolution was performed by respecting Ru3

It is noticeable that a few solutions in the left figure seem to be correct too, and some closed solutions in the right figure are obviously wrong, even if they were found correct. These are the cases where both, correct and wrong, ambiguous closed solutions converge correctly and point to the same position. Since the refinement is calculated for a restricted amount of iterations, it can happen that the refined, previously wrong solution gets a slightly smaller s_0 value than the refined, previously correct solution. The non-trivial ambiguity resolution over s_0 picks then the previously wrong solution. Anyway, this does not matter, as both solutions are equivalent in this case and, as seen in the right figure, no wrong convergence is noticeable in the outcome.

Best solution determination: At this point of the calculation, we have a refined unambiguous solution. However, in cases where closed solutions are affected by big scatter, the refinement may converge to a local extremum (see Chapters 2.3.1 and 3.4.2). Therefore, if the closed solution was an outlier, its adjusted equivalent might, but must not necessarily be wrong. Therefore, in order to assure a correct final result, the current result is stored on a list and the whole calculation is repeated for the next Remote Unit combinations on the list of good combinations. This step is repeated until a preset number of results, coming from different combinations, are available. In the flow chart in Figure 4.8 this corresponds to the path where "immediate acceptance" is always no.

After the desired number of solutions was determined, the best of these has to be identified. This is newly done over the a-posteriori measurement variance s_0, like described in Chapter 3.4.3 "Solution Quality". Finally the picked result is checked against accuracy constraints like a-priori error and a-posteriori error, in order to avoid the emission of solutions with to low accuracy. If these constraints are not met, the result is dropped, otherwise it is passed on to the output threads.

Immediate Acceptance: It has been observed that most times all adjustment results, derived from different closed solutions, were pointing to the same position. This is no surprise since the adjustment step takes the same measurement data in each calculation, the only parameter that differs is the point the optimisation starts with. That is why the immediate acceptance feature was implemented and checks the solution's a-posteriori measurement error s_0 immediately after it was determined. If this measurement error is very small, the solution is accepted immediately. In order to avoid bad results being accepted this way, the value for the maximum error for immediate acceptance should be chosen within some metres.

Another immediate acceptance criterion could be the following: Results of preceding calculations which were derived from other closed solutions, are compared to the current result. If two or more results point to the same position, most probably both solutions are correct. However, this check was not implemented and tested yet at the time this work was created.

5 Measurement Evaluation and Error Investigation

This chapter presents and discusses the results of the field test, described in Chapter 4.2. Therefore, the best achievable results for each measurement point M1 to M4 are presented and analysed. Sub-chapter 5.5 finally investigates different errors and gives advises for future improvements. At this point it has to be mentioned that the outcome of the calculations was surprisingly good, as it was the first useful test that was conducted with Remote Units in the configuration presented in Chapter 3.1. However, as the next chapters will show, the results are affected by systematic errors. Unfortunately it was not possible to correct them in the calculation performed by the Target Processor prototype. The source of error is assumed to lie in the measurements and the test setup itself; Chapter 5.5 discusses this topic.

The maximum standard deviation σ_{max} of the solution scatter is mostly stated for error description in the following sections. This measure has to be understood as the standard deviation of the solution scatter in the direction of the error ellipses main diagonal (see Chapter A.4). Also the solution accuracy constraints according to which solutions were accepted or dropped, are always related to this standard deviation. In order to determine the a-priori solution accuracy estimations, a measurement accuracy of $\sigma_i = 10ns$ was assumed. This is admittedly a really optimistic assumption, however, the value is easy to handle, as it corresponds to approximately $3m$. A measured value for the measurement accuracy σ_i was not available yet when this thesis was being created.

5.1 Measurement at M1

Measurement point M1 was chosen to lie slightly outside the base lines between the RUs (see Figure 4.2). The best results for this point were received for tuples where five measurements could be mapped together. In this case no accuracy constraints were needed to be defined, as all 725 tuples gave good results, leading to a final distribution with $\sigma_{max} = 3.65m$. Figure 5.1 shows the outcome.

In the calculations where 4- and 5-tuples were respected some constraints were necessary to be defined: Unsurprisingly, the RU combination (1, 2, 4, 5) led to bad results (see light grey targets in Figure 5.2) as its RUs are quite distant from the mobile position. These results were possible to be removed by setting the worst a-priori solution accuracy constraint to 5m, so no solution from this combination was accepted anymore. This had the effect, however, that, all in all, just 67% of the 1132 tuple matches were leading to an accepted result. A final maximum measurement deviation of $\sigma_{max} = 4.02m$ was achievable.

However, the systematic measurement error is quite big: the distance between the mobiles GPS position and the mean of the result scatter is $16.2m$ in the 5-tuple case and $15.9m$ in the case where also 4-tuples were accepted.

5.2 Measurement at M2

The results of the 5-tuple mapping at point M2 are just affected by little scatter ($\sigma_{max} = 3.5m$) as the results in Figure 5.3 show. Accuracy constraints were almost unnecessary: Just two

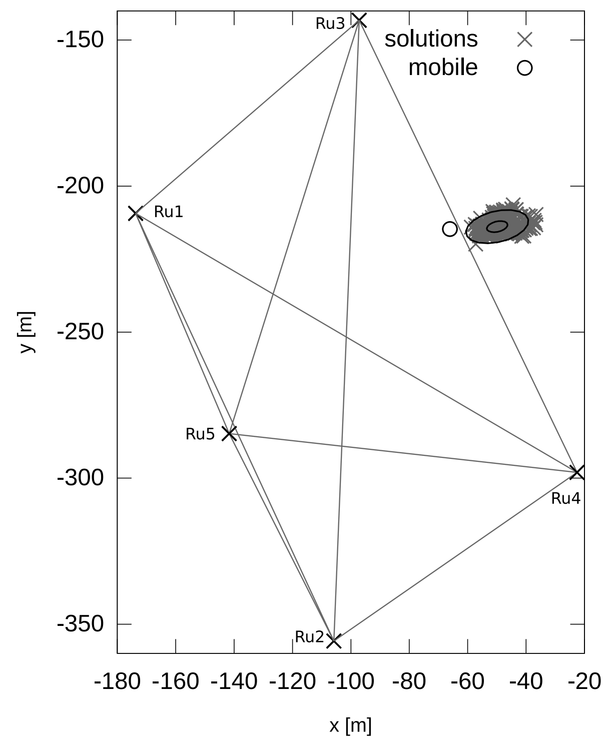


Figure 5.1: Solutions for 725 5-tuple mappings at M1 ($\sigma_{max} = 3.65m$)

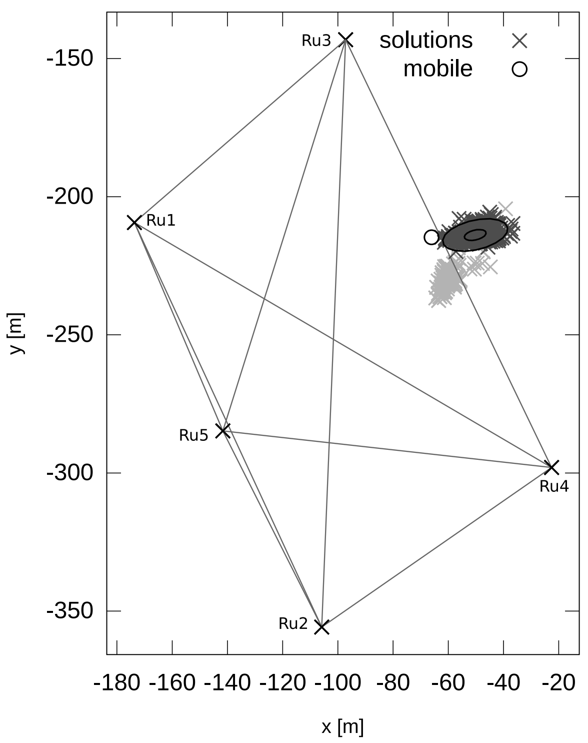


Figure 5.2: Dark grey: Good solutions for 758 4-tuple mappings at M1 ($\sigma_{max} = 4.02m$). Light grey: Solutions for combination (1, 2, 4, 5) which were suppressed by the constraint setting.

of the 504 input data tuples were leading to small outliers and had to be dropped with an a-posteriori error constraint of $15m$. Newly a quite big systematic error of $11.7m$ is noticeable.

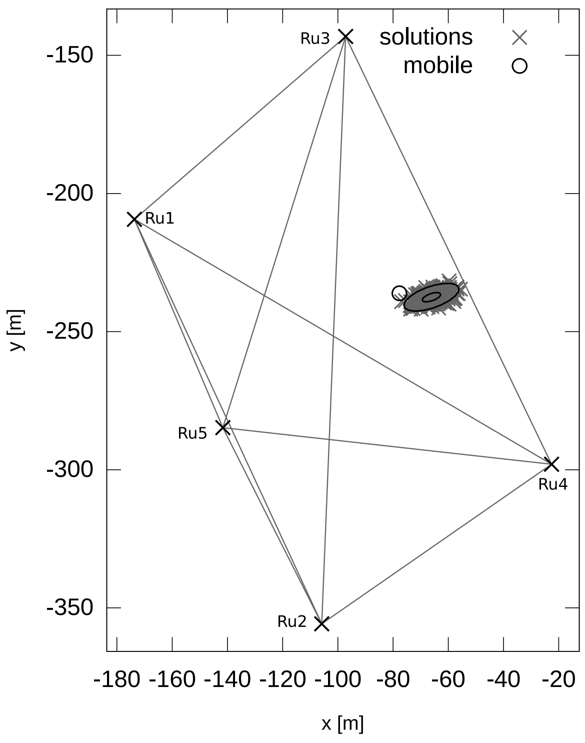


Figure 5.3: Solutions for 502 5-tuple mappings at M2 ($\sigma_{max} = 3.5m$)

The 4-tuple mappings were leading to similar problems as discussed in the previous chapter. Hence, newly an a-priori error constraint of $5.5m$ had to be defined in order to suppress results from combination (1, 2, 3, 5). Finally, 702 out of 843 solutions were fulfilling the accuracy constraints and their scatter had a maximum standard deviation of $\sigma_{max} = 3.8m$. The systematic error in this measurement was $12m$.

5.3 Measurement at M3

Measurement M3 is the only one which led to a small systematic error. Figure 5.4 shows the calculation outcome of the 5-tuple mappings for this measurement point.

RU	**Rcvd. Replies**	**RU-Transmitter Dist.**	**Mean Residuals** v $[m]$	$\sigma_v[m]$
Ru1	1040	$97.7m$	5.3	2.3
Ru2	1383	$82.9m$	-3.3	2.0
Ru3	1370	$129.9m$	-5.9	1.6
Ru4	1236	$81.1m$	4.0	1.5
Ru5	665	$43.6m$	-0.1	2.5

Table 2: Number of replies received per RU, RU-transmitter distances and mean residuals in the 5-tuple calculation at M3

It is noticeable that in this measurement there were significantly more 4-tuple than 5-tuple matches; i.e. 704 4-tuples and 355 5-tuples. The reason for this was Ru5 which had received

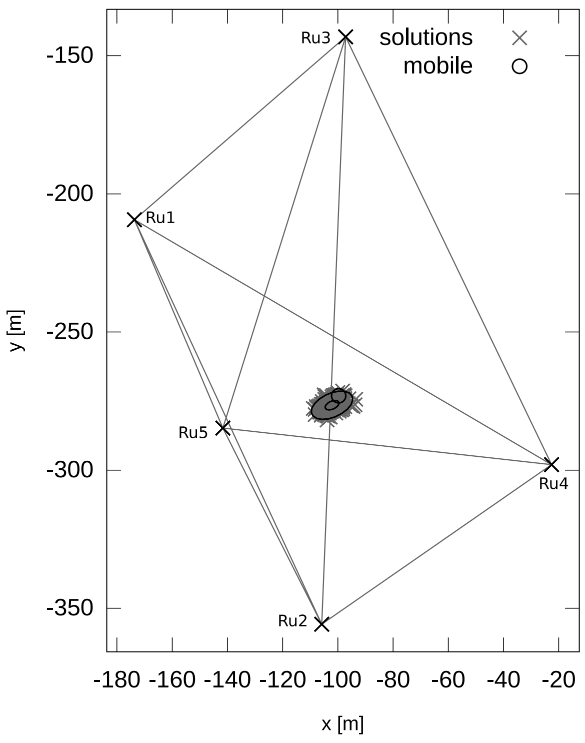


Figure 5.4: Solutions for 355 5-tuple mappings at M3 ($\sigma_{max} = 2.7m$, systematic error is $4.18m$)

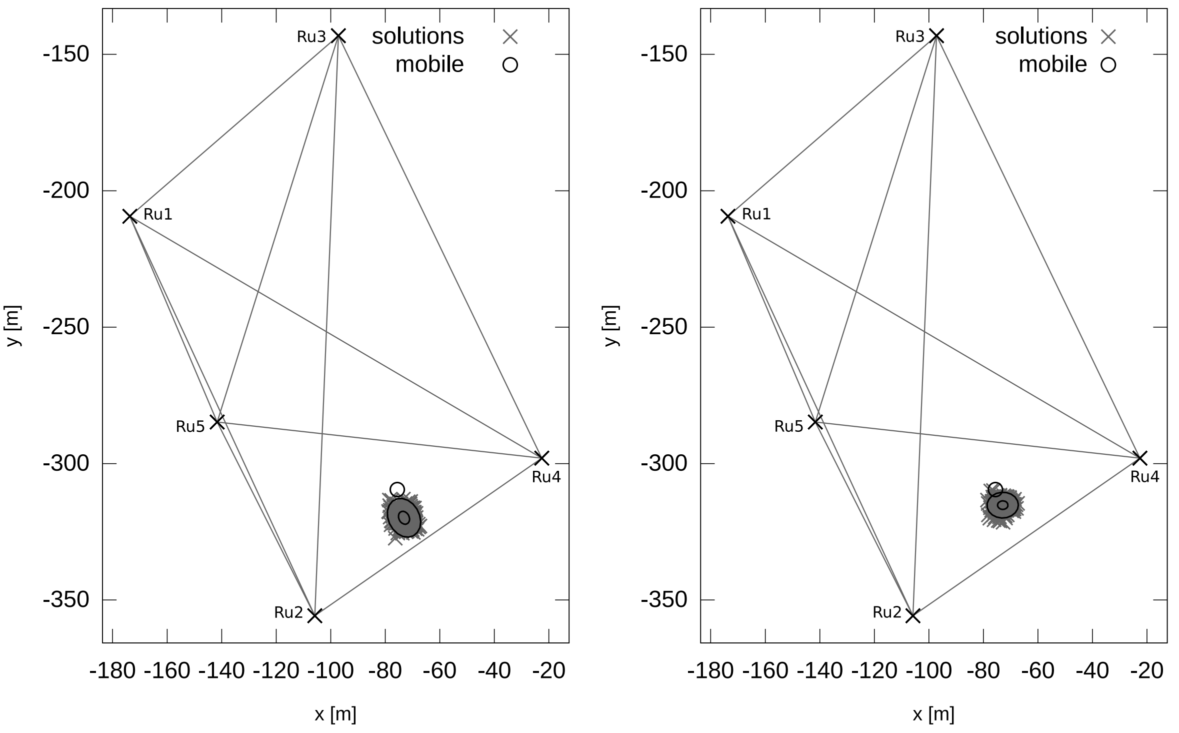
far less replies than the other Remote Units (see Table 2). The small distance between Ru3 and the transmitter is assumed to be the reason for this problem; The RU's receiver unit went most probably into saturation due to the high receiving power. However, the mean residuals of Ru5 are not extraordinary, so the measurement quality of the few received replies seems to be alright.

The 4- and 5-tuple calculation did not lead to interesting results: due to the central position of the measurement point the solution scatter was just increased to $\sigma_{max} = 3.45m$ and the accuracy constraints did not have to be set restrictively.

5.4 Measurement at M4

Measurement M4 has turned out to be a special case: The measurement point was quite distant from Ru3 and a metal fence which had to be crossed twice by the signal, was located in the direct way of propagation between transmitter and Ru3.

The 5-tuple results for this measurement point are shown in Figure 5.5. The left figure contains the results, which were determined in the conventional way, without outlier detection. There the solution scatter is very small ($\sigma_{max} = 2.5m$), however, the systematic error is quite big with $10.6m$. For this measurement point, the outlier detection feature has turned out to be quite effective: by enabling it, not just the solution scatter became smaller ($\sigma_{max} = 1.9m$), but also the systematic error decreased to $6.3m$ (see Figure 5.5b). The Huber-k estimator with $k = 2 \cdot 10^{-8}s(\approx 6m)$ was mostly reducing the weights of measurements coming from Ru3: Table 3 contains the mean weights and residuals which were received at each RU after all 478 outlier detection calculations. Ru3 is the only Remote Unit with a low weight mean, which means that its measurements were constantly detected to be outliers. Also its mean residuals are big and affected by big scatter; This is also a sign that Ru3 was producing bad measurements.



(a) No outlier detection $\sigma_{max} = 2.5m$ (b) Outlier detection activated ($\sigma_{max} = 1.9m$)

Figure 5.5: Solutions for 478 5-tuple mappings at M4

RU	**Mean Residuals** v $[m]$	$\sigma_v [m]$	**Mean Weights** p	σ_p
Ru1	3.6	1.7	0.990	0.062
Ru2	-2.3	0.4	1.000	0.000
Ru3	-20.4	7.0	0.333	0.137
Ru4	2.5	0.2	1.000	0.000
Ru5	2.4	1.5	0.998	0.019

Table 3: Mean residuals and weights for 5-tuple calculation with outlier detection at M4

In a last calculation Ru3 was totally deactivated and this measure led to the best results at this measurement point: newly a standard deviation of $\sigma_{max} = 1.9m$ was achieved and the systematic error shrunk to $3.5m$.

5.5 Error Investigation

The scatter of the calculation outcome was mostly satisfying in the test results presented in the previous chapters. Especially the 5-tuple matches led to low scatter variances. The systematic errors, however which cause the results to scatter around a wrong point, still

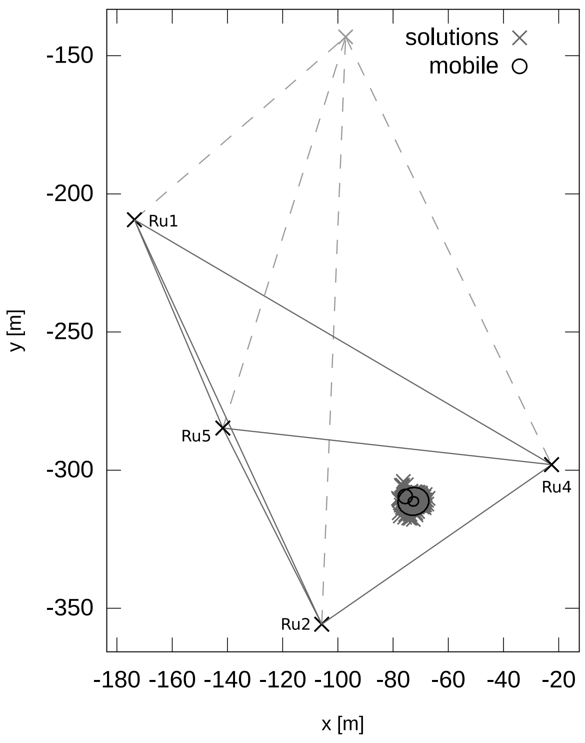


Figure 5.6: Solutions for 496 4-tuple mappings at M4 with Ru3 deactivated ($\sigma_{max} = 1.9m$)

build a severe problem. Three reasons have been figured out which may be responsible for these errors: bad GPS positioning performance, inaccuracies in the timestamping step and problems during synchronisation. The following chapters shortly explain these issues and discuss possibilities to overcome them.

5.5.1 GPS Positioning Inaccuracy

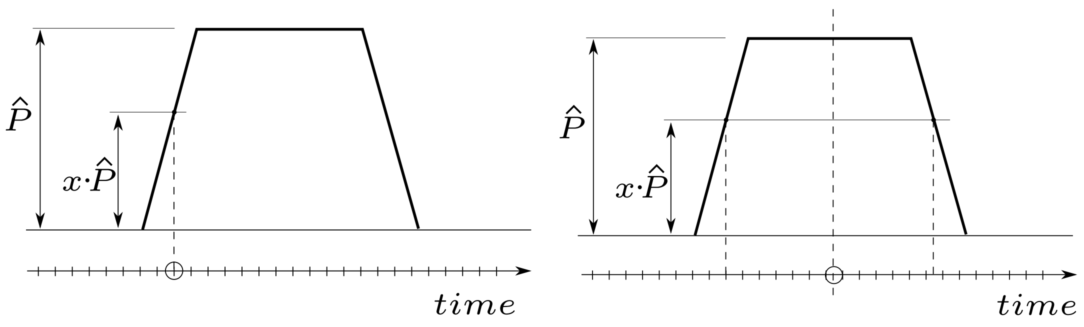
Wide influence to the systematic error of the calculation outcome is caused most probably by the way the Remote Unit and transmitter positions were determined: The u-blox Lea4T GPS timing module, installed at the Remote Units, was used to determine the positions. According to the modules data-sheet [26] its positioning accuracy is $2.5m$ CEP if good satellite visibility is given. This means that 50% of the position fixes lie within a circle with radius $2.5m$ around the correct position. Additionally, all RUs had different satellite view, as the park is surrounded by trees and houses; This even decreases the positioning accuracy. Therefore, despite the position coordinates were determined by averaging the GPS position for around 30 minutes, they still may be affected by errors due to satellite shading, multipath etc.

Remote Units build the focal points of the hyperbolas which are intersected in the calculation. The error which results out of the GPS inaccuracy, causes the focal point being shifted and along with it the hyperbolas. The solution, which is determined by the intersection of more hyperbolas is in follow also moved to another place. Since the intersections are different for each position, also the systematic error can vary between different locations and for different RU combinations.

In order to avoid such errors, the positions of RUs and measurement points, or at least their relative positions, have to be determined with higher accuracy e.g. by using better measurement equipment.

5.5.2 Timestamping Inaccuracies

The timestamping at the Remote Unit in its current configuration is performed like explained in Chapter 3.1: The counter is read out at the point where the rising edge of a pulse crosses 50% of the pulses mean peak power, measured relative to the noise level (see Figure 5.7a). However, if timestamping is performed like this, the height of the noise level and also signal power can influence the location of the sampling point: A higher noise level e.g. shifts the sampling point up, higher signal power shifts it down. Depending on the slope of the edge this causes the timestamp being read out later or earlier.



(a) Edge sampling

(b) Impulse middle sampling

Figure 5.7: Timestamping modes

It is important for the positioning calculation to timestamp a reply at the same point, at each RU. However, replies are attenuated during their propagation and RUs may have different noise levels. As a consequence, additional measurement noise may be introduced to the time of signal arrival differences and in succession also to the calculation result. However, the extend of this noise had not been analysed yet when this thesis was being created. At this place it is important to keep in mind the dimensions we talk about: A Mode-S preamble pulse has a duration of $0.5\mu s$. Therefore, it will be sampled around 40 times at $80MHz$, 160 times if also the oversampling is respected. Since electromagnetic waves propagate at speed of light, the pulse has a width of approximately $150m$ during its propagation.

A better way to perform timestamping would be to take the pulses mid-point as TOA timestamp: If the timestamps of rising and falling edge are determined at a certain power level, the mean of these timestamps describes the impulse middle. In this case the slope would not influence the timestamping that much, assuming that rising and falling edges are affected by more or less the same flattening. The right image in Figure 5.7 shows an example for impulse mid-point sampling.

The Remote Unit provides timestamps for all SSR pulses and therefore it was tried to use different pulses for the TOA measurements and also to average the timestamps of several pulses. However, there were no significant differences noticeable in the calculation outcome.

5.5.3 Synchronisation Problems

The GPS timing module has a maximum achievable timing accuracy of $15ns$ RMS to UTC time, according to the datasheet [26]. This corresponds to approximately $4.5m$, expressed as a pseudo range. If all RUs were affected by the same error, this would not represent a problem, as the calculation just uses time of arrival differences. However, in the measurement scenario every RU was synchronised to UTC separately, and so the synchronisation is totally dependent on the GPS module: Every GPS module synchronises itself to the satellite clocks it receives and emits the pulse per second. Depending on the satellite view or possible multipath propagation this time pulses may be affected by different errors on different Remote Units. The situation is even worse, as GPS modules need to know their 3D position for timing [26] which is basically the position of the RU. However, as discussed in Section 5.5.1, also this position may be affected by a certain error, which consequently influences the timing performance.

Furthermore such a synchronisation has one overhead: All units are synchronised to one, globally valid time, even though there is no need for that. It would be enough if all RUs were synchronised with high accuracy within each other, the offset to UTC time is insignificant. UTC time information is just needed at the point the target is reported by the TP, as every target needs a detection timestamp. The accuracy requirement on this timestamp, however, is much looser and so this time can be provided e.g. by NTP synchronisation of the Target Processor.

A solution for this problem would be to use a common view GPS scenario, where all RUs use the same satellite(s) for synchronisation. In such scenario timing accuracies of $1ns$ to $10ns$ [31] between the RUs should be realisable. Unfortunately, the currently used GPS module cannot be configured to use certain satellites only.

Another possibility would be a reference transponder synchronisation, where the synchronisation is calculated over replies, emitted by a transponder and received at every unit [24]. This replies form common events at all Remote Units and, if the distances between transponder and RUs are known, a synchronisation can be performed over them. Furthermore, the synchronisation events would take the same way trough the Remote Unit's electronics, as normal measurements do. According to [24], synchronisation accuracies up to $1ns$ should be achievable in a reference transponder synchronisation. In the future work on the prototype this synchronisation method will be evaluated.

6 Conclusion and Outlook

The last chapter of this thesis contains a short summary of the work and research which were performed in the course of its creation. AviBit's Multilateration project will be continued during the next years; Therefore, the final sub-chapter gives an outlook on the planned improvements and modifications on the Target Processor.

6.1 Conclusion

This work and especially the field test results have shown that it is possible to solve the hyperbolic positioning problem with the theory and methods elaborated herein. The outcome of the research was this thesis, some GNU Octave scripts for algorithm evaluation and data representation and the C++ implementation of the Target Processor process.

The development of the TP was the most extensive task, however, also the most profit was obtained out of it: During its implementation and testing, the characteristics and main difficulties of the Hyperbolic Navigation calculation were explored and mainly solved. The encountered problems sometimes forced the development of single process parts being restarted from the scratch, with new approaches and plenty of patience. Finally, however, a fully functional and real-time capable Target Processor was created which exceeds the primarily intended requirements: Besides basic features like calculation in two and three dimensions, synchronisation and measurement mapping, the developed process is capable to perform outlier detection and solution error estimation. Furthermore, it is highly configurable over configuration files, which makes the prototype applicable for further system development and test data evaluation. The modular process structure makes the TP easily extensible and so it can build the base for subsequent research.

Since a big part of the testing was performed with measurement data, obtained in field tests, tools to manage and manipulate recording files and were developed. These instruments were put together and form now an useful toolkit, applicable for work in this field.

6.2 Outlook

The development of the AviBit MLAT system will be continued, with the aim to install a system prototype on an airport, which is capable to monitor the whole airport area within the EUROCAE accuracy constraints defined in [23]. This implies that some revision work has to be performed at the Remote Unit and also the Target Processor development will be continued. In order to fulfil the requirements of an operational MLAT system, the TP needs to be extended by some fundamental features.

Firstly the Remote Unit synchronisation has to be improved and be extended by the possibility to use reference transponder synchronisation. This implies that oscillator and RU characteristics must be analysed better in order to be able to reduce the synchronisation error. The need for reference transponder synchronisation has two reasons: Firstly, AviBit expects some synchronisation improvement with this way of synchronisation, as the same signals are used for synchronisation and for TOA measurements. Secondly, the system shall be able to operate independently from third-party systems in order improve reliability and availability.

The TP also needs to be extended by a target tracking feature: Target tracking will facilitate the calculation itself as it offers approximate target positions and so solution ambiguities and wrong convergence can be avoided more easily. Furthermore, a smoothed target output could be provided, completed by data about target movement and statistics.

In a last step the prototype could be extended by an interrogator unit in order to build an active Multilateration system. This would enable the system to detect all transponder equipped objects within in its area of interest, independent of the availability of Mode-S broadcasts and interrogations from other systems. Additionally, when knowing the accurate time of target interrogation, an auxiliary information is available for the calculation.

References

[1] M. C. Stevens. *Secondary Surveillance Radar. I Title.* Artech House Inc., Norwood, MA 02062, 1988.

[2] Wikipedia - Die freie Enzyklopädie. Radar. `http://de.wikipedia.org/wiki/Radar` [Online; accessed 19-January-2010].

[3] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins. *GPS Theory and Practice.* Springer-Verlag, Vienna/New York, 3rd edition, 1994.

[4] Radar Tutorial. `http://www.radartutorial.eu` [Online; accessed 29-January-2010].

[5] S. Bernhart. Design und Implementierung eines FPGA-basierenden SSR Message Decoders. Master's thesis, TU Graz, March 2008.

[6] Austrocontrol GmbH. Transponderpflicht für Kraftangetriebene Luftfahrzeuge im 'Luftraum E'. `http://www.daec.de/aktuell/downfiles/2009/LO_Circ_2009_A_03_en_tcm586-69349.pdf` [Online; accessed 30-January-2010].

[7] J. Göbel. *Radartechnik, Grundlagen und Anwendung.* VDE Verlag, Berlin, Offenbach, 2001.

[8] ERA Coorporation. *Multilateration, Executive Reference Guide.* ERA Coorporation, 2007. also available on `http://www.multilateration.com` [accessed 25-January-2010].

[9] W. Langhans. Multilateration takes off in Europe. *Skyway Magazine Autumn 2005*, pages 32 – 34, 2005.

[10] W. H. L. Neven, T. J. Quilter, R. Weedon, and R. A. Hogendoorn. Wide Area Multilateration. Technical Report NLR-CR-2004-472, National Aerospace Laboratory (NLR), August 2005.

[11] Y. E. Yang, J. Baldwin, and A. Smith. Multilateration tracking and synchronization over wide areas. *2002. Proceedings of the IEEE Radar Conference*, pages 419–424, 2002.

[12] B. Hofmann-Wellenhof, G. Kienast, and H. Lichtenegger. *GPS in der Praxis.* Springer-Verlag, Vienna/New York, 1994.

[13] R. Bucher and D. Misra. A Synthsizable VHDL Model of the Exact Solution for Three-dimensional Hyperbolic Positioning System. In *VLSI Design*, volume 15, pages 507–520. Department of Electrilcal and Computer Engineering, New Jersey Center for Wireless and Telecommunication, New Jersey Institute of Technology, Taylor I& Francis Ltd, 2002.

[14] S. Bancroft. An Algebraic Solution of the GPS Equations. *IEEE Transactions on Aerospace and Electronic Systems*, 21(1):56–59, January 1985.

[15] M. Geyer and A. Daskalakis. Solving passive multilateration equations using Bancroft'salgorithm. In *Digital Avionics Systems Conference, 1998. Proceedings., 17th DASC. The AIAA/IEEE/SAE*, volume 2, Bellevue, WA, USA, October/November 1998.

[16] H. J. Bartsch. *Taschenbuch mathematischer Formeln*, volume 19. Fachbuchverlag Leipzig, 2001.

[17] R. Jäger, T. Müller, H. Saler, and R. Schwäble. *Klassische und robuste Ausgleichungsverfahren - Ein Leitfaden für Ausbildung und Praxis von Geodäten und Geoinformatikern.* Herbert Wichmann Verlag, Heidelberg, 2005.

[18] H. Sünkel. *Ausgleichsrechnung 1, Ein Skriptum nach einer Vorlesung von o.Univ.Prof. Dr. H. Sünkel.* Abteilung für Mathematische Geodäsie und Geoinformatik, Technische Universität Graz, Graz, 2nd edition, 1993.

[19] E. Gökalp, O. Güngör, and Y. Boz. Evaluation of Different Outlier Detection Methods for GPS Networks. *Sensors*, 8(11):7344–7358, 2008.

[20] ICAO. *Annex 10 - Aeronautical Telecommunications*, volume 4 - Surveillance Radar and Collision Avoidance Systems. ICAO, 2007.

[21] ICAO. *Annex 10 - Aeronautical Telecommunications*, volume 3 - Communication Systems. ICAO, 2007.

[22] ICAO. *DOC 9871 - Techinical Provisions for Mode S Services and Extended Squitter.* ICAO, 1 edition, 2009.

[23] EUROCAE. *ED-117 – Minimum Operational Performance Specification for Mode S Multilateration Systems for Use in Advanced Surface Movement Guidance and Control Systems (A-SMGCS).* EUROCAE, April 2003.

[24] G. Galati, M. Gasbarra, and M. Leonardi. Multilateration algorithms for time of arrival estimation and target location in airports. In *Radar Conference, 2004. EURAD. First European*, pages 293–296, 2004.

[25] Piezo Crystal Company. OCXO-2920136 Datasheet.

[26] u-blox AG. LEA-4T - Programmable GPS Module with Precision Timing (Datasheet), 2008.

[27] Trimble Navigation Limited. Resolution T (Datasheet), 2007.

[28] Eigen - a C++ template library for linear algebra. `http://eigen.tuxfamily.org` [Online; accessed 31-January-2010].

[29] Sensis Corporation. *Mode S Bases Vehicle Locator - VeeLo NextGen (Brochure)*, 2008. `http://www.sensis.com/docs/576/` [Online; accessed 2-February-2010].

[30] Microsoft. Bing maps. `http://www.bing.com/maps` [Online; accessed 18-January-2010].

[31] D. W. Allan and M. A. Weiss. Accurate Time and Frequency Transfrer During Common-View of a GPS Satellite. In *Ann. Freq. Control Symposium, USAERADCOM*, volume 34, pages 335 – 346. National Bureau of Standards Boulder, Colorado, May 1980.

A Mathematical derivations

This appendix contains some mathematical derivations and explanations to complete the theory explained in the Chapters 2, 3 and 4. Therefore, the illustrations mostly start with equations which were already stated earlier in this work and explain the derivations step by step, subsequently.

A.1 Tricky Part in Bancroft's Algorithm

Equation (2.14) states that

$$\langle \vec{x}, \vec{x} \rangle = \lambda^2 \langle \vec{d}, \vec{d} \rangle + 2\lambda \langle \vec{d}, \vec{e} \rangle + \langle \vec{e}, \vec{e} \rangle$$

where

$$\vec{x} = \lambda \vec{d} + \vec{e}$$

This result is simply received by applying the Lorentzian inner product (2.9) and doing some derivations. To show this, the vectors $\vec{x}$, $\vec{d}$ and $\vec{e}$ are chosen to be three dimensional.

$$\lambda = \langle \vec{x}, \vec{x} \rangle = \langle \lambda \vec{d} + \vec{e}, \lambda \vec{d} + \vec{e} \rangle \tag{A.1}$$

$$\begin{aligned} \lambda &= (\lambda d_1 + e_1)^2 + (\lambda d_2 + e_2)^2 - (\lambda d_3 + e_3)^2 && \text{(A.2)} \\ &= \lambda^2 d_1^2 + 2\lambda d_1 e_1 + e_1^2 + \lambda^2 d_2^2 + 2\lambda d_2 e_2 + e_2^2 - \lambda^2 d_3^2 - 2\lambda d_3 e_3 - e_3^2 && \text{(A.3)} \\ &= \lambda^2 (d_1^2 + d_2^2 - d_3^2) + 2\lambda (d_1 e_1 + d_2 e_2 - d_3 e_3) + e_1^2 + e_2^2 - e_3^2 && \text{(A.4)} \\ &= \lambda^2 \langle \vec{d}, \vec{d} \rangle + 2\lambda \langle \vec{d}, \vec{e} \rangle + \langle \vec{e}, \vec{e} \rangle && \text{(A.5)} \end{aligned}$$

As seen in the last derivation step, the dimensionality does not matter and so equation (2.14) holds for all dimensions.

A.2 Applying the Least Square Principle

As described in detail in [18], and stated in [12] and [3], starting from equation (2.19)

$$\vec{m} + \vec{v} = \mathbf{A}\, \vec{x}$$

we want to apply the least square principle (2.24)

$$S = \vec{v}^T \mathbf{P} \vec{v} = Min\,|_{\vec{v}}$$

on it. We do so by differentiating S with respect to $\vec{v}$ and setting it zero:

$$\frac{dS}{d\vec{v}} = \vec{v}^T \mathbf{P} = 0 \tag{A.6}$$

By writing (2.19) in its differential form, we come to the following expression:

$$d\vec{v} = \mathbf{A}\, d\vec{x} \tag{A.7}$$

Inserting (A.7) into (A.6) leads to the following:

$$\frac{dS}{d\vec{x}} = \vec{v}^T \mathbf{P}\,\mathbf{A} = 0 \tag{A.8}$$

Transponation of the previous formula and the fact that $\mathbf{P}^T = \mathbf{P}$, as $\mathbf{P}$ is diagonal, leads to:

$$\mathbf{A}^T \mathbf{P}^T \vec{v} = \mathbf{A}^T \mathbf{P}\,\vec{v} = 0 \tag{A.9}$$

Finally we rearrange (2.19) to get the expression $\vec{v} = \mathbf{A}\,\vec{x} - \vec{m}$. Then we insert it into the previous formula and we get to the result which is also stated in (2.25):

$$\mathbf{A}^T \mathbf{P}\,\mathbf{A}\,\vec{x} - \mathbf{A}^T \mathbf{P}\,\vec{m} = 0$$

A.3 Error Propagation for the Adjustment Equations

We start with the formulae (2.58) and (2.59):

$$\begin{aligned} \vec{x} &= \mathbf{F}\,\vec{m} = \mathbf{Q}\,\mathbf{A}^T \mathbf{P}\,\vec{m} = (\mathbf{A}^T \mathbf{P} \mathbf{A})^{-1} \mathbf{A}^T \mathbf{P}\,\vec{m} \\ \mathbf{\Sigma}(\vec{x}) &= \mathbf{F}\,\mathbf{P}^{-1} \sigma_m^2\,\mathbf{F}^T \end{aligned}$$

The first equation gives us the expression $\mathbf{F} = (\mathbf{A}^T \mathbf{P} \mathbf{A})^{-1} \mathbf{A}^T \mathbf{P}$ which is now inserted into the second formula and leads to:

$$\mathbf{\Sigma}(\vec{x}) = (\mathbf{A}^T \mathbf{P} \mathbf{A})^{-1} \mathbf{A}^T \mathbf{P}\,\mathbf{P}^{-1} \sigma_m^2\,\mathbf{P}^T \mathbf{A}((\mathbf{A}^T \mathbf{P} \mathbf{A})^{-1})^T \tag{A.10}$$

as $\mathbf{P}$ is diagonal and $(\mathbf{A}^T \mathbf{P} \mathbf{A})^{-1}$ is symmetric, the transponation of these matrices has no effect. So we get to the result:

$$\mathbf{\Sigma}(\vec{x}) = (\mathbf{A}^T \mathbf{P} \mathbf{A})^{-1} \mathbf{A}^T \mathbf{P}\,\mathbf{P}^{-1} \mathbf{P} \mathbf{A} (\mathbf{A}^T \mathbf{P} \mathbf{A})^{-1} \sigma_m^2 \tag{A.11}$$

$$\mathbf{\Sigma}(\vec{x}) = (\mathbf{A}^T \mathbf{P} \mathbf{A})^{-1} \sigma_m^2 \tag{A.12}$$

A.4 Error Ellipse

A one dimensional zero mean Gaussian distribution is defined by its variance σ^2. A multivariate Gaussian distribution is described by its covariance matrix $\mathbf{\Sigma}$ as shown in equation (2.20). In the two dimensional case this covariance matrix looks like follows:

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{xy} & \sigma_y^2 \end{bmatrix} = \sigma_0^2 \begin{bmatrix} q_x^2 & q_{xy} \\ q_{xy} & q_y^2 \end{bmatrix} \tag{A.13}$$

In some cases it is useful to plot the distribution, e.g. for solution scatter visualisation. This can be done with the error ellipses which can be seen as contours of the distribution. According to [17, p. 78] the error ellipse can be calculated as follows(1):

(1) The square on q_x and q_y is missing in [17, p. 78]. Simulations have shown that it is necessary, therefore it was added here.

$$a = \sigma_0 \sqrt{\frac{1}{2}(q_x^2 + q_y^2 + k)} \tag{A.14}$$

$$b = \sigma_0 \sqrt{\frac{1}{2}(q_x^2 + q_y^2 - k)} \tag{A.15}$$

$$k = \sqrt{(q_x^2 - q_y^2)^2 + 4q_{xy}^2} \tag{A.16}$$

$$\phi = \frac{1}{2} arctan \left(\frac{2q_{xy}}{q_x^2 - q_y^2} \right) \tag{A.17}$$

Here a is the length of the big semi-major axis, b the length of the small semi-major axis. ϕ describes the angular orientation of the ellipse. Figure A.1 shows an error ellipse for $\Sigma = \left(\begin{smallmatrix} 1 & 0.3 \\ 0.3 & 1 \end{smallmatrix}\right)$ and centre in (0, 0).

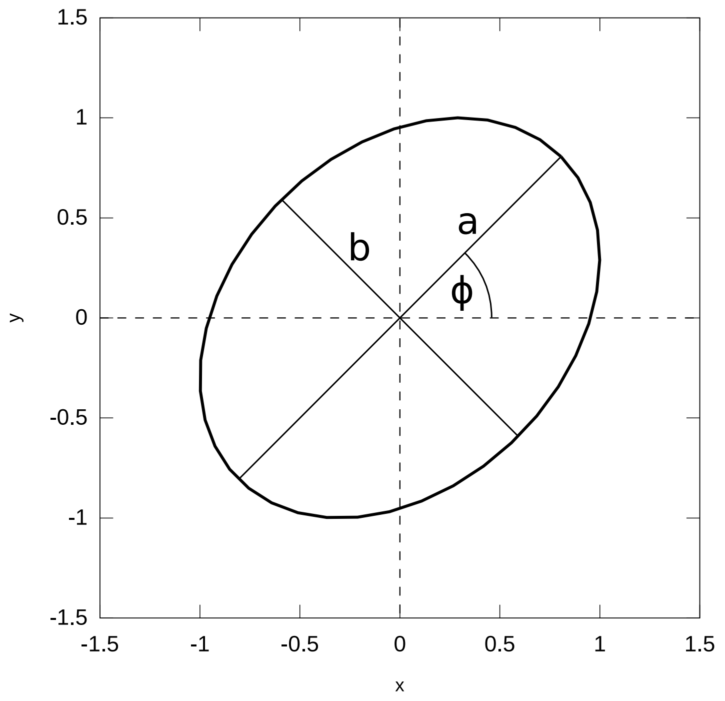


Figure A.1: Error ellipse for $\Sigma = \left(\begin{smallmatrix} 1 & 0.3 \\ 0.3 & 1 \end{smallmatrix}\right)$ and centre in (0, 0)

A.5 Linear Regression

A linear regression between the variables x and y tries to fit a line with the following formula through a cloud of measurements:

$$y = a_1 \cdot x + a_0 \tag{A.18}$$

The parameters a_0 and a_1 are derived from a set of set of measurement pairs (x, y) in order to minimise the squared error of the outcome. [16, pp.565] shows how these parameters can be calculated by solving a system of linear equations. The formula was, however, modified a bit, in order to keep it simpler:

$$\mathbf{A}\,\vec{a} = \vec{b} \tag{A.19}$$

$$\mathbf{A} = \begin{bmatrix} n & \sum x \\ \sum x & \sum x^2 \end{bmatrix} \quad \vec{b} = \begin{bmatrix} \sum y \\ \sum xy \end{bmatrix} \quad \vec{a} = \begin{bmatrix} a_0 \\ a_1 \end{bmatrix}$$

Here for x and y the corresponding values of the measurement pairs are inserted. n is the number of measurements. As $\mathbf{A}$ is a diagonal 2x2 matrix it can be easily inverted to determine $\vec{a}$.

List of Acronyms

ACAS Airborne Collision Avoidance System

ADS-B Automatic Dependent Surveillance Broadcast

ASR Airport Surveillance Radar

AWGN additive white Gaussian noise

DoP Dilution of Precision

EUROCAE European Organization for Civil Aviation Equipment

FIFO First In Fist Out (Buffer)

FPGA Field Programmable Gate Array

GNSS Global Navigation Satellite System

GPS Global Positioning System

IBK Institute for Broadband Communication

ICAO International Civil Aviation Organization

IFF Identification Friend or Foe

LOS Line of Sight

LS Least Square

MLAT Multilateration

NTP Network Time Protocol

OCXO Oven-Controlled Crystal Oscillator

PDF Probability Density Function

PLL Phase-locked loop

PPM Pulse Position Modulation

PPS pulse per second

PSR Primary Surveillance Radar

RADAR Radio Detection and Ranging

RU Remote Unit

SPI Special Purpose Identification

SSR Secondary Surveillance Radar

TDOA time difference of signal arrival

TOA time of signal arrival

TOE time of signal emission

TP Target Processor

UTC Coordinated Universal Time

WGS84 World Geodetic System 1984