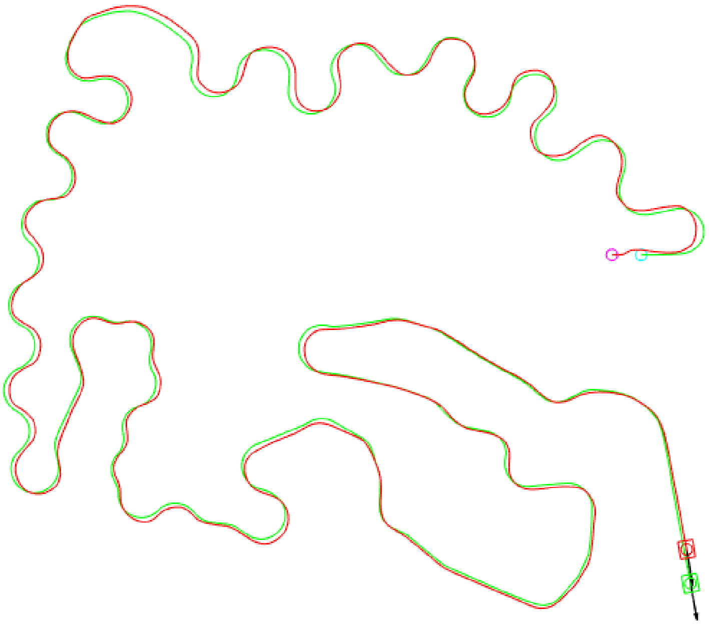
CHALMERS



Ground vehicle platooning using a dynamic and modular GNSS system

Based on GPS Real-Time Kinematics with moving base tower to achieve accurate relative positioning

Master's thesis in Systems, Control and Mechatronics

Michael Vu

Martin Johnsson

Department of Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2019

master thesis 2019:EENX30

Ground vehicle platooning using a dynamic and modular GNSS system

MICHAEL VU
MARTIN JOHNSSON

Department of Electrical Engineering
Division of Signal processing
Chalmers University of Technology
Gothenburg, Sweden 2019

Ground vehicle platooning using a dynamic and modular GNSS system
MICHAEL VU
MARTIN JOHNSSON

Supervisor: Henrik Norström, DevPort
Examiner: Lars Hammarstrand, Department of Electrical Engineering

Master Thesis 2019:EENX30
Department of Electrical Engineering
Division of Signal processing
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 (0)31-772 1000

Cover: An autonomous vehicle (red) tracking the path of the lead vehicle (green) using the implemented LQR controller, EKF and GPS RTK estimation.

Printed by Chalmers Reproservice
Gothenburg, Sweden 2019

Ground vehicle platooning using a dynamic and modular GNSS system
MICHAEL VU
MARTIN JOHNSSON
Department of Electrical Engineering
Chalmers University of Technology

Abstract

In the area of vehicle coordination and autonomous driving it is of interest to know the position relative to other vehicles and objects surrounding the car. Most autonomous vehicles use a combination of camera and lidar to perceive their own surrounding to better make decisions. In this thesis, carried out at DevPort AB, different concepts using GNSS are investigated which would allow a vehicle to get a highly accurate relative position of all nearby vehicles. The specific vehicle coordination scenario that is approached is platooning, when one vehicle leads multiple others. The goal of this thesis is to answer if it is possible to have a functioning platoon using only relative position measurements and Inertial Navigation Systems (INS).

A base platform is designed in which to apply the hypothesis on, as it is a new project for DevPort nothing had been done within the area beforehand. The platoon is made out of two differential steered vehicles each equipped with GPS, IMU, and wheel speed sensors. The vehicles are modeled and a path planning algorithm is developed to emulate a platoon. The platoons performance based on relative positioning is tested assuming both rovers only have an INS as well as with the use of GPS Real-Time Kinematics (RTK) for high accuracy relative position measurements. The results are further improved with the use of the sensor fusion algorithm, Extended Kalman Filter (EKF) for better state estimation. In addition to testing the platoon assuming a working relative position estimation, separate GPS RTK solutions are developed based on the least squares estimation of triple difference and double difference carrier-phase systems.

Simulations of the platoon show that it is possible for a platoon to function with small deviation with the usage of only relative position measurements given that the measurements are accurate enough. Using GPS RTK in simulation to measure relative distance show the possibility of obtaining highly accurate relative position measurements, but with a non-complex least squares estimation the accuracy of the estimations lowers with time. To get a continuous highly accurate relative position measurement more advanced estimations are needed.

Keywords: Platooning, Autonomous Cars, Sensor Fusion, Extended Kalman Filter, Path Planning, Real-Time Kinematics, Global Navigation Satellite System, Global Positioning System, Linear-Quadratic Regulator.

Acknowledgements

We would like to thank DevPort for giving us the opportunity to do this master thesis at their company. Special acknowledgments go to three people that has helped us throughout the project. Firstly our supervisor at DevPort, Henrik Norström, for his guidance within the project. Secondly Erik Stenborg at department of Electrical Engineering, Chalmers University of Technology, for his assistance with the GNSS theory. Finally our examiner Lars Hammarstrand at department of Electrical Engineering, Chalmers University of Technology, for helping us throughout the project.

Gothenburg, June 2019
Michael Vu, Martin Johnsson

Nomenclature

DARE - Discrete-time Algebraic Riccati Equation

DD - Double Difference

DOP - Dilution Of Precision

ECEF - Earth Centered and Earth Fixed

EKF - Extended Kalman Filter

GA - Ground Antennas

GNSS - Global Navigation Satellite Systems

GPS - Global Positioning System

INS - Inertial Navigation System

ICR - Instantaneous Center of Rotation

IMU - Inertial Measurement Unit

LAMBDA - Least Squares AMbiguity Decorrelation Adjustment

LMMSE - Linear Minimum Mean Square Error

LQR - Linear-Quadratic Regulator

MCS - Master Control Station

MS - Monitor Stations

RTK - Real-Time Kinematics

SD - Single Difference

TD - Triple Difference

Contents

1
Introduction

The interests of autonomous ground vehicles have increased substantially in the last few years [1]. Automotive companies are therefore moving to integrate a more and more complex autonomous system to aid the driver. This trend is also seen in other areas, for example in the creating of the autonomous refuse collection vehicle by Volvo [2]. For DevPort, a technical consulting company that mainly works within the automotive industry, it is of interest to gain more knowledge within the area. Therefore this thesis is performed to create a platform that could be used in educational purposes for DevPort's consultants, as well as allow further research within the area.

In this project, the concept of platooning is used in creating an autonomous platform. Vehicle platooning is the idea of having multiple vehicles moving together in a platoon at high speeds to lower fuel consumption of the vehicles [3]. Moving in a platoon increases the aerodynamic efficiency as the vehicles move close enough together to reduce drag, which is of great interest in larger vehicles such as trucks [4]. Platooning is therefore mostly focused on trucks, also called truck platooning [5, 6, 7], as that is an area where fuel consumption can be reduced the most. In the current industry, the cooperation between the vehicles requires both complex driving support systems and the communication between vehicles [8]. In the future, platooning could lead to safer and more efficient highway travel for all vehicles connected to the platooning system.

As mentioned above, in order to enable a vehicle platoon, complex driving support systems need to be implemented. To create the driving support system, all vehicles within the platoon need to both be able to communicate, as well as have a good estimation of their vehicle state. Specifically, speed, position, and heading are crucial as the vehicles need to stay close to each other while keeping the same speed to increase aerodynamic efficiency [3, 4, 5, 6, 7]. This project therefore focuses on two different areas. Firstly, the implementation of sensor fusion algorithms to correctly estimate both the speed and position of the vehicle. Secondly, the implementation of a controller and coordinator that allow the vehicles to drive in a platoon. Another important area that needs to be implemented is the measurement of the relative distance between vehicles in the platoon. Autonomous cars are usually equipped with lidar and camera sensors to perceive the surroundings. In this thesis the idea is to find out if it is possible to have a working platoon using relative position instead of perceiving the surrounding and too which degree it is possible to measure the relative distance.

Estimating the vehicle state is of utmost importance in the implementation of platooning, and autonomous systems in general, and is usually done by a combination of different methods. An example of vehicle state estimation is the combination of Inertial Measurement Unit sensors (IMUs), a Global Navigation Satellite System (GNSS), and cameras or lidar or both. In this project, the idea is to have 'blind' vehicles that only relies on a set of few sensors to estimate position, velocity, and heading. 'Blind' vehicles, in this project, means no camera or other light-based sensors, nor any ultrasonic sensors. The plan is to have no direct distance measurement between the vehicles to allow platooning, instead use other sensors to calculate the relative distance between the vehicles. The solution that will be investigated is the usage of specific methods based on the GNSS Global Positioning System (GPS), Real-Time Kinematics (RTK), which is a method to combine different GPS receivers data to improve the relative measurement between the used receivers.

The control of a platoon can be done in different ways depending on the coordinator and the controller. The controller's task is to steer the platoon so that the platoon is following a trajectory created by the coordinator. The trajectory is created by using, for instance, one or multiple path planning algorithms such as straight lines between two points or Bezier's curve [9]. The goal is to create a trajectory based on the current and previous location of the lead vehicle such that the following vehicles are mimicking the trajectory of the lead vehicle.

1.1 Purpose

The main purposes of this thesis are to investigate whether it is possible to have a working platoon comprised of 'blind' autonomous vehicles and whether it is possible to use GPS RTK for relative position measurement between the vehicles in the platoon. In addition to investigating this hypothesis, it is also needed to create a hardware platform which can be used to implement the found solutions. DevPort plans on using this project as a base for further development within the area of autonomous vehicles as well as for educational purposes internally, hence a hardware platform needs to be developed.

The hardware platform is planned to be made out of two vehicles (rovers) that are capable of performing all needed measurements, control, and computations to physically test the proposed solutions. It is important to note that the hardware platform is merely a deliverable for DevPort and not evaluated in this thesis. It mainly restricts the theoretical parts of the report as explained in Section 1.2.

1.2 Project delimitation

As the main interest of the thesis is in the overall platoon and the developed GPS solution, the result will be theoretical. Therefore the thesis will be limited to present the theoretical result of the developed solutions and the hardware platform will only be made to allow for future work within the area. The hardware platform mainly restricts the theoretical solutions as the solutions have to be compatible with the developed hardware. Therefore, the theory produced in this thesis will use the developed hardware platform as its foundation.

Besides being restricted to the limitations of the hardware platform, the theory will not use complex models for the basis (e.g. vehicle model and motion model) of the implementation. This project delimitation is made to allow the thesis to focus on the relevant theoretical parts instead of making a more realistic base.

1.3 Thesis outline

The content of this thesis is presented as follows:

- Chapter 2 presents the dynamic vehicle model of the rovers and the resulting discrete state space representation of the system.
- Chapter 3 provides the theory about the GNSS and how it can be used to accurately determine the position of a receiver.
- Chapter 4 describes the theory behind sensor fusion and how it is used to obtain state estimations of the system.
- Chapter 5 describes how the trajectories are created and how the reference points are fed into the controller.
- Chapter 6 describes the controller and how it is used to control the states to follow a trajectory.
- Chapter 7 describes the simulation tests of the platooning and presents the obtained results.
- Chapter 8 presents a discussion of the results and methods as well as summarize and conclude the thesis.

2
Vehicle modeling

In this chapter the vehicle model of the system is defined in Section 2.1 and 2.2. The model's frames are defined in Section 2.3 where a local and global frame is derived. Lastly in Section 2.4 the discrete state space representation of the local frame is derived which will be used for the controller in Chapter 6.

2.1 Overview of the model

The vehicle model presented in this chapter is based on the rover of the hardware platform described in Appendix A.1 (schematic shown in Figure 2.1). The rover is a differentially steered vehicle which means that the turn rate of the vehicle is dependant on the wheels speeds on each side of the vehicle [10], unlike regular modern cars which use conventional steering actuators to turn the wheels towards the desired driving direction.

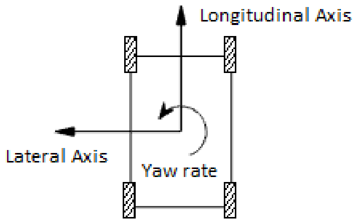


Figure 2.1: Schematic of the used vehicle. The schematic and model is only defined in 2D (no vertical axis).

There are multiple ways to model the differential steering [10, 11, 12] but a very simplified model is chosen where the input to the vehicle model is velocity rather than acceleration. This is because the motors of the used rover are very powerful for the rover's weight and could almost instantly reach the desired velocity. Other factors such as rolling resistance, air resistance, and slip are neglected to simplify the modeling. The simplifications made to the vehicle modeling reduces its depiction of reality but simplifies its use for future implementations requiring a model, such as the sensor fusion filter and the controller.

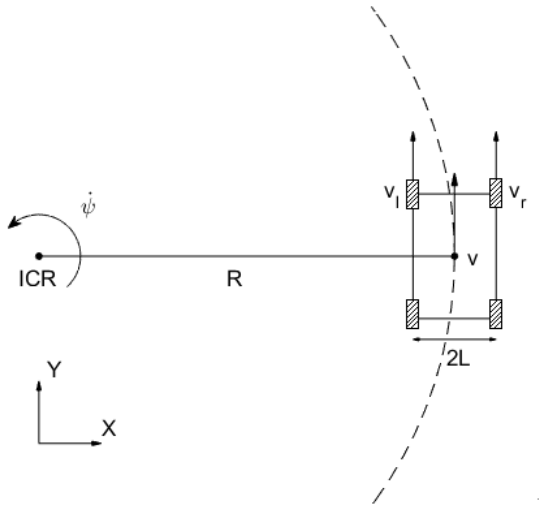


Figure 2.2: Schematic of the kinematics model for a differentially steered vehicle.

2.2 Differentially steered model

A differentially steered vehicle relies on the different speed on each side of the vehicle in order to turn [10]. The wheels on each side maintain the same speed due to the functionality of the motor controller which is further explained in the hardware section in Appendix A.1. The input to the model, $\boldsymbol{u}$, is given by

$$\boldsymbol{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} \omega_l \\ \omega_r \end{bmatrix}, \tag{2.1}$$

where ω_l and ω_r are the angular velocities of the left and right wheels respectively. This gives the following relations

$$v_l = \omega_l r, \tag{2.2}$$

$$v_r = \omega_r r, \tag{2.3}$$

where v_l is the velocity of the left side of the vehicle, v_r is the velocity of the right side of the vehicle and r is the wheel radius. The steering of a differentially steered vehicle is illustrated in Figure 2.2, where ICR is the instantaneous center of rotation, v is the velocity of the vehicle, R is the turning radius of the vehicle, L is the horizontal distance between the wheel and the vehicle's center, and $\dot{\psi}$ is the angular velocity around ICR [10]. The velocity v of the vehicle and the velocity of each side, v_l and v_r, can be expressed as

$$v_l = \dot{\psi}(R - L), \tag{2.4}$$

$$v_r = \dot{\psi}(R + L), \tag{2.5}$$

$$v = \dot{\psi}R. \tag{2.6}$$

It can be derived from (2.4), (2.5) and (2.6) that the velocity v and angular velocity $\dot{\psi}$ are given by

$$v = \frac{v_r + v_l}{2}. \tag{2.7}$$

$$\dot{\psi} = \frac{v_r - v_l}{2L}. \tag{2.8}$$

The expressions for the velocity (2.7) and angular velocity (2.8) derived in this section are used to formulate the vehicle state space model in Section 2.4.

2.3 Transformation between local and global coordinates

There are two defined frames in this model to describe the vehicle's movement. These are illustrated in Figure 2.3 where the local xy frame's origin is centered at the vehicle where the x-axis is paralell to the vehicle's longitudinal axis and where the y-axis is paralell to the lateral axis. The vehicle is placed in a fixed global XY world frame where the origin may be placed arbitrarily.

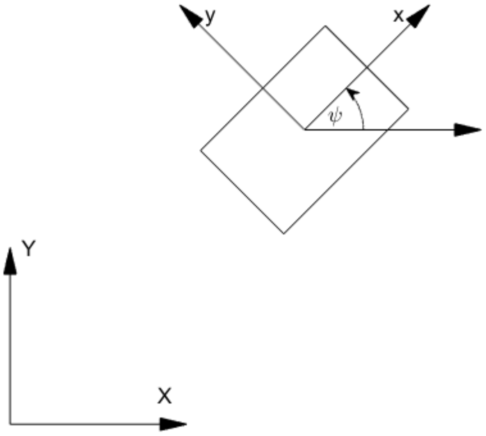


Figure 2.3: Schematic of the vehicle and its local and global coordinate frames.

The velocity v_x and v_y in the vehicle's local frame is defined according to

$$v_x(t) = \frac{v_l(t) + v_r(t)}{2}, \tag{2.9}$$

$$v_y(t) = 0. \tag{2.10}$$

The expressions for the vehicle's velocities V_x and V_y in the global frame can be obtained by using the trigonometric functions $\sin\psi$ and $\cos\psi$ where ψ is the angle between the global X axis to the local x axis. to project the vehicle's local velocities to global velocities according to

$$V_x(t) = v_x(t)\cos\psi(t), \tag{2.11}$$

$$V_y(t) = v_x(t) \sin \psi(t). \tag{2.12}$$

The expressions (2.11) and (2.12) are nonlinear due to the trigonometric functions $\sin \psi$ and $\cos \psi$. However, the local velocities, v_x and v_y, are linear and can therefore be used in linear state space models and controllers which require a linear model. If the states $v_x(t)$, $v_y(t)$ and $\psi(t)$ are measured at all times, the mapping of global coordinates can be done if the initial global coordinates X_0 and Y_0 are known according to

$$X(t) = X_0 + \int_0^t V_x dt = X_0 + \int_0^t v_x(t) \cos \psi(t) dt, \tag{2.13}$$

$$Y(t) = Y_0 + \int_0^t V_y dt = Y_0 + \int_0^t v_x(t) \sin \psi(t) dt. \tag{2.14}$$

The expressions for the global positions (2.13) and (2.14) are used in the sensor fusion model in Section 4.2.1 to estimate the global positions for the vehicles.

2.4 Vehicle state space model

The vehicle state space will consist of states that are of interest to control, which are chosen as

$$\boldsymbol{\chi} = \begin{bmatrix} x \\ \psi \end{bmatrix}, \tag{2.15}$$

where x is the x-axis position in its local frame and ψ is the yaw orientation of the vehicle. The reason why the global coordinates X and Y are not chosen as states is that they are nonlinear and can therefore not be represented in a linear state space model.

The local velocity $\dot{x}$ is equal to (2.9) and the angular velocity $\dot{\psi}$ is equal to (2.8). Furthermore the position x and orientation ψ are dependant on the velocities according to

$$x(t) = x_0 + \int_0^t \dot{x} dt, \tag{2.16}$$

$$\psi(t) = \psi_0 + \int_0^t \dot{\psi} dt, \tag{2.17}$$

in the continuous domain. However, the discrete state space representation is of interest because the hardware of the rover operates in the discrete domain. Using Euler first order approximation, (2.16) and (2.17) can be approximated as

$$x_k = x_{k-1} + \dot{x} \cdot t_s, \tag{2.18}$$

$$\psi_k = \psi_{k-1} + \dot{\psi} \cdot t_s \tag{2.19}$$

in the discrete domain where t_s is the sampling time of the system. The expressions (2.18) and (2.19) assume $\dot{x}$ and $\dot{\psi}$ are constant during the sampling time t_s which is not always the case and will result in an inaccurate estimation. This inaccuracy can be reduced by lowering the sampling time t_s to the lowest possible value the hardware can handle.

The discrete state space representation can be expressed as

$$\boldsymbol{\chi}_k = \mathbf{A}_d \boldsymbol{\chi}_{k-1} + \mathbf{B}_d \mathbf{u}_{k-1}, \tag{2.20}$$

$$\mathbf{A}_d = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{B}_d = t_s \begin{bmatrix} \frac{r}{2} & \frac{r}{2} \\ \frac{-r}{2L} & \frac{r}{2L} \end{bmatrix}. \tag{2.21}$$

where the matrices $\mathbf{A}_d$ and $\mathbf{B}_d$ are obtained by (2.7), (2.8), (2.18) and (2.19).

3
Global Positioning System

In this chapter, an introduction of GNSS trilateration concept is made (Section 3.1 and 3.2). This is followed by the model describing the trilateration of receiver coordinates as well as the introduction of the estimation method, see Section 3.3 and 3.3.1 respectively. Lastly, the GPS Real-Time Kinematics (RTK) method is presented, Section 3.4, followed by the application of the least squares estimation on the GPS RTK system (Section 3.4.1).

These theoretical parts will be used as the basis for the proposed relative position measurement solution which is combined with the complete system in Section 4.2.5.

3.1 GNSS basic concept

The basics of a GNSS is to use satellites as reference points to calculate the absolute position of a receiver in 3D-space. In 2D-space, assume there exists a reference point p_1 that sends out a signal that is easily identifiable from a wide distance. Also, assume that the reference point sends out this signal at the time T_0 and that the receiver has a perfectly synchronized clock with the reference. If the receiver receives the signal at T_0+t_d, then the signal has traveled for t_d time before arriving at the receiver, as their clocks are synchronized. This means that the distance between the reference and the receiver is equal to

$$l_1 = t_d \cdot v_s, \tag{3.1}$$

where v_s is the signal velocity. The issue with only having one reference point would mean that the receiver could be anywhere on a radius l_1 from the reference (see Figure 3.1).

By adding an additional reference point p_2, the uncertainty lowers drastically to the intersections of the circles (see o_1 and o_2 in Figure 3.1). This is the principle called trilateration, to use multiple reference points to to obtain a receiver position. The true solution could be found by either adding an additional reference, which would leave only one solution, or by knowing other information about the receivers. An example of additional information that could be used would be for a car which is driving on a road. If only one of the possible solutions (o_1 and o_2) is on the driven road and the other is above water it is clear that one can disregard the solution above water. Four satellites are required to be able to trilaterate the receiver coordinates in 3D.

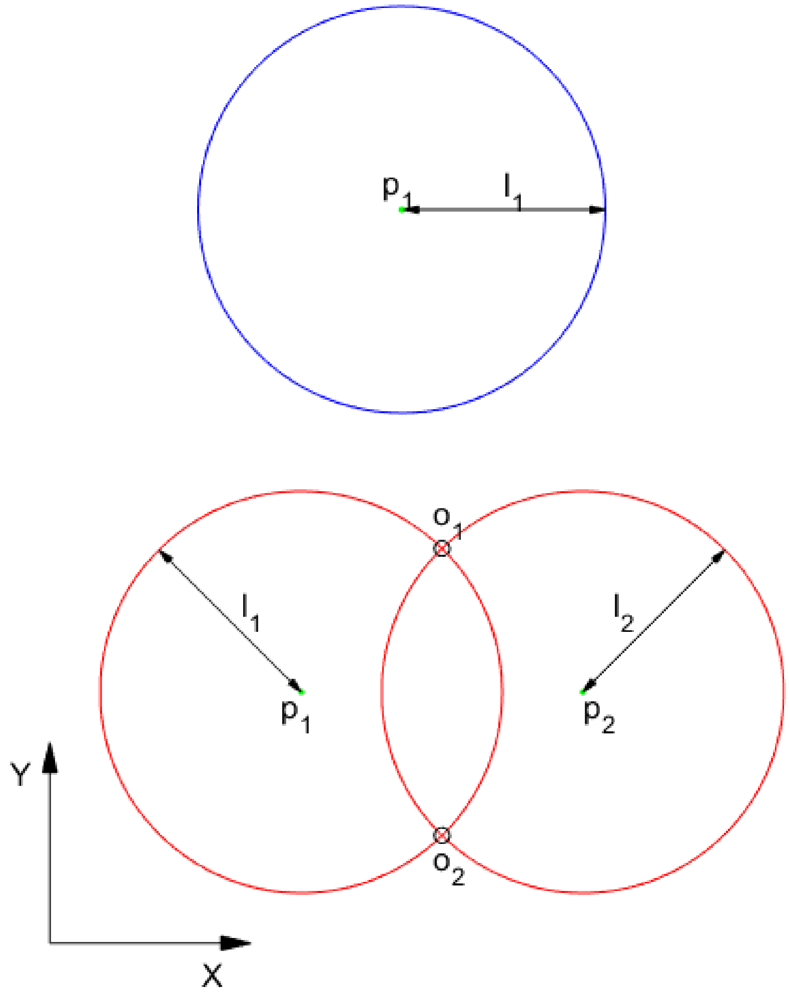


Figure 3.1: For one reference point p_1, the receiver can be anywhere on the blue circle. For two reference points p_1 and p_2, the receiver can be either at o_1 or o_2.

3.2 GNSS in practice

A GNSS is split into three different segments which define how the systems are set up and maintained [13]. The first segment is the space segment which is made up of the satellites in space. These satellites need to keep their orbits to both allow global coverage (four satellites visible from any point on Earth's surface) as well as maintain the accuracy of the system. The second segment, the control segment, is responsible for the maintenance of the satellites. This maintenance is crucial to be able to use the system, which includes e.g. regulating the satellites in case of deviation from an orbit. Lastly, the user segment, includes all the GNSS receivers that are used to calculate position.

The GPS space segments consist of 31 active satellites in orbit to allow for global coverage, which is the GNSS with most active satellites in the world [14]. The GPS control segment is made up of a Master Control Station (MCS) located in Colorado Springs, USA, as well as multiple Monitoring Stations (MS) and Ground Antennas

(GA) which are spread throughout Earth to enable the fine adjustment to be made to the space segment [13]. These fine adjustments are required to keep track of the satellite navigation data that are used in the trilateration of the receiver coordinates.

The calculation of the GNSS receiver's position is usually based on a measurement called pseudorange which is an inaccurate measurement of the true distance (see equation (3.1)) from the satellites to the GNSS receiver. The most significant factors that affect the measurement accuracy are:

- **Ionospheric error**: The ionosphere is the part of the upper atmosphere that affects the satellite signal when it passes through. The ionosphere's lower bound starts around 60 km above Earth's surface and the upper bound stretches to around 1000 km [15]. It consists off multiple free electrons that have sufficient density that when the satellite signal travels through it, it disperses some of the signal [15]. One problem with the ionospheric error is that the ionosphere changes during the day depending on its interaction with the sun, hence it is hard to fully estimate its effect on the signal [15]. The ionosphere affects the transmitted signals based on the frequency of the signals which means that a significant portion of the error can be removed with the usage of multiple frequency bands [15]. The ionospheric effect will not be covered further in this thesis, instead see [13, 15].

- **Tropospheric error**: The troposphere is the first part of the Earth's atmosphere and covers up to around 50-60 km above ground [13, 16]. It does not affect the signal based on the signal's frequency but rather delays the signal based on the temperature, pressure and humidity [13].

- **Receiver clock bias**: Receiver clock bias refers to the error in the receiver's internal clock. The range measurement performed by the GNSS receiver is based on the time between the signal being sent and the signal being received (as explained in the example). Hence, an error in the GNSS receiver's internal clock adds to a range measurement error as the signal has been perceived to have traveled either further or less than in reality.

- **Satellite clock error and orbital error**: The first error, satellite clock error, refers to the same type of error as for the receiver clock bias, but for the satellite instead of the receiver. Although the satellite clock is one of the best and most expensive in the world, they are still not perfect [13]. The orbital error refers to the deviation of the satellite from its orbit. The parameters needed to account for the satellite clock error are sent through the navigation message of the satellites, similarly for the orbital error [13].

- **Multipath error**: The multipath error refers to the multipath taken by the signal to arrive at the receiver as the signals get reflected from other objects and surfaces before arriving at the destination [13].

After the pseudoranges have been measured, the GNSS receiver's position can be calculated. This calculation uses multiple measurements, each with its own error, which introduces another problem that affects the positioning error. This problem is called Dilution Of Precision (DOP) [17] and is based on the position of the satellites in relation to the receiver. A 2D illustration of this error can be seen in Figure 3.2 where, instead of a definite range like in the previous trilateration example (Figure 3.1), the range of each satellite has an error. The overlapping of the circles used to decide the correct position will then mean an overlapping area in which the receiver could be. This propagates to a combined error which gives worse accuracy if the satellites are close to each other (see Figure 3.2).

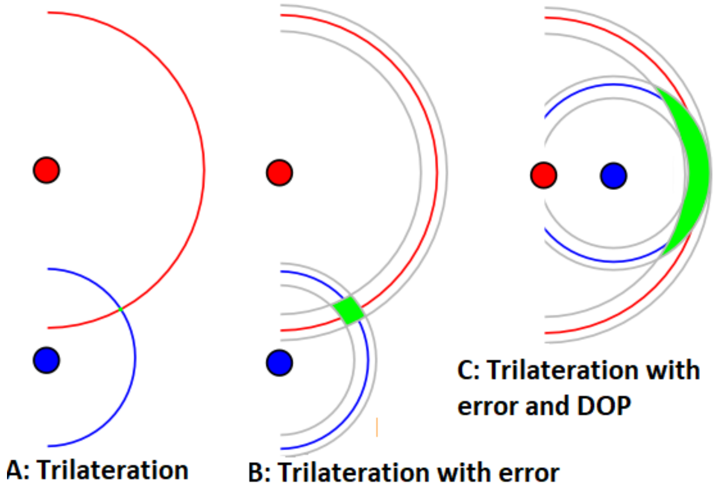

A: Trilateration
B: Trilateration with error
C: Trilateration with error and DOP

Figure 3.2: Different scenarios to determine the potential location of the receiver, detonated in green. (Image taken from [18]).

3.3 Trilateration

The beforementioned example, as seen in Figure 3.1, gives a basic understanding of how the GPS receiver position is found in 2D-space. To trilaterate the GPS receiver's 3D position $\mathbf{x}_r[k]$, the true range $p_r^s[k]$ to each of the available satellites is required,

$$p_r^s[k] = \sqrt{(x^s[k] - x_r[k])^2 + (y^s[k] - y_r[k])^2 + (z^s[k] - z_r[k])^2} = ||\mathbf{x}^s[k] - \mathbf{x}_r[k]|| \tag{3.2}$$

where $x[k]^s, y[k]^s$, and $z[k]^s$ denote the position for the satellite s and $x_r[k], y_r[k]$, and $z_r[k]$ denote the unknown positions for the GPS receiver r. All coordinates are in Earth-centered and Earth-fixed (ECEF) frame [13], which has its origin at Earth's center and fixed $X-$, $Y-$ and Z-axis (see Figure 3.3). In this thesis, the satellite coordinates are assumed to be known instead of calculated. It is possible to calculate the satellite position by using the ephemeris data [19] of the satellite which includes six parameters called the Keplerian elements [19]. [13, 20] explains the complete method to calculate the satellite coordinates based on the Keplerian

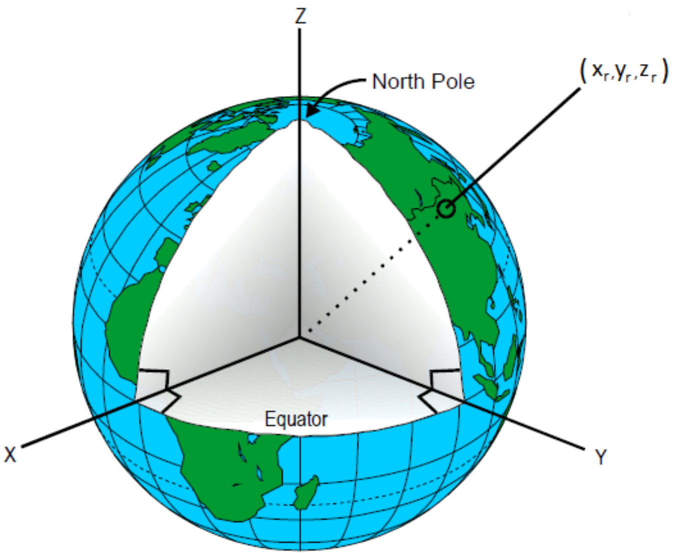


Figure 3.3: An image illustrating the ECEF coordinate system. The axes are fixed and rotates together with Earth. (Image taken from [21]).

parameters.

The pseudorange measurement, $\rho_r^s[k]$, can be modeled based on the true range $p_r^s[k]$ and some known errors included in the measurement

$$\rho_r^s[k] = p_r^s[k] + c\Delta\delta_r^s[k] + I_r^s[k] + T_r^s[k]. \tag{3.3}$$

This model contains the ionospheric error, $I_r^s[k]$, the tropospheric error, $T_r^s[k]$, and

$$\Delta\delta_r^s[k] = \delta^s[k] - \delta_r[k], \tag{3.4}$$

which is the combined error from the satellite orbit and clock error $\delta^s[k]$ as well as the receiver clock error $\delta_r[k]$, multiplied by the signal speed c (speed of light) to get the distance. The ionospheric, tropospheric, and satellite orbit and clock error is assumed to be known and already modeled for. This assumption is made because these errors only require the satellite navigation messages to be modelled [13], which are given [25]. Knowing that the system contains n_s satellites and n_t epochs, rearranging (3.3) as

$$\rho_r^s[k] - c\delta^s[k] - I_r^s[k] - T_r^s[k] = p_r^s[k] - c\delta_r[k], \tag{3.5}$$

with known variables on the left-hand side and the unknowns on the right-hand side, it is possible to discern that the system has $n_s \cdot n_t$ equations and $3 + n_t$ unknowns. To find the receiver position the number of satellites and epochs need to fulfill the following inequality $n_s n_t \geq 3 + n_t$. Having only a single time epoch means that the receiver needs four satellite pseudorange measurements.

3.3.1 Solving trilateration using least squares

Generally the GPS receiver is connected to more than four satellites which results in an overdetermined system. To account for measurement errors and multiple pseudorange measurements that would lead to an overdetermined system, least squares can be used to estimate the position and errors. The following problem is formulated

$$\begin{aligned} \delta\rho_r^s &= \rho_r^s - \rho_{r,0}^s, \quad \delta\rho_r^s \to 0, \forall s, \\ \delta\rho_r^s &= (||\mathbf{x}^s - \mathbf{x}_r|| + c\delta^s - c\delta_r + I_r^s + T_r^s + \epsilon^s) \\ &\quad - (||\mathbf{x}^s - \mathbf{x}_{r,0}|| + c\delta^s - c\delta_{r,0} + I_r^s + T_r^s), \end{aligned} \tag{3.6}$$

where $\delta\rho_r^s$ is the resulting error between the measured ρ_r^s and the estimated $\rho_{r,0}^s$, and ϵ^s is the combination of the unmodeled errors. Based on the initial estimations $\mathbf{x}_{r,0}$ and $\delta_{r,0}$, we want to iteratively find $\delta\mathbf{x}_r$ and $\delta\delta_r$ so that

$$\mathbf{x}_r = \delta\mathbf{x}_r + \mathbf{x}_{r,0}, \tag{3.7}$$

and

$$\delta_r = \delta\delta_r + \delta_{r,0}, \tag{3.8}$$

solves the expression (3.6). The solution to the least square problem is obtained by substituting (3.7) and (3.8) in (3.6). The resulting expression is obtained by

$$\begin{aligned} \delta\rho_r^s = (&||\mathbf{x}^s - (\delta\mathbf{x}_r + \mathbf{x}_{r,0})|| + c\delta^s - c(\delta\delta_r + \delta_{r,0}) + I_r^s + T_r^s + \epsilon^s) \\ &- (||\mathbf{x}^s - \mathbf{x}_{r,0}|| + c\delta^s - c\delta_{r,0} + I_r^s + T_r^s), \end{aligned} \tag{3.9}$$

which can be simplified to

$$\delta\rho_r^s = ||\mathbf{x}^s - (\delta\mathbf{x}_r + \mathbf{x}_{r,0})|| - c\delta\delta_r - ||\mathbf{x}^s - \mathbf{x}_{r,0}|| + \epsilon^s. \tag{3.10}$$

By using Taylor series expansion of vector modulus [22], (3.10) can be approximated as

$$\delta\rho_r^s \approx \hat{\mathbf{x}}_{us}\delta\mathbf{x}_r - c\delta\delta_r + \epsilon^s \tag{3.11}$$

where $\hat{\mathbf{x}}_{us} = \frac{\mathbf{x}^s - \mathbf{x}_{r,0}}{||\mathbf{x}^s - \mathbf{x}_{r,0}||}$ is a unit vector. Hence, the problem formulation (3.6) can be approximated as

$$\delta\rho_r^s = \begin{bmatrix} -\hat{\mathbf{x}}_{us} & -c \end{bmatrix} \begin{bmatrix} \delta\mathbf{x}_r \\ \delta\delta_r \end{bmatrix} + \epsilon^s. \tag{3.12}$$

For n_s satellites, the least squared problem can be approximated as

$$\delta\boldsymbol{\rho}_r^s = \begin{bmatrix} \delta\rho_r^{s1} \\ \vdots \\ \delta\rho_r^{n_s} \end{bmatrix} = \mathbf{G} \begin{bmatrix} \delta\mathbf{x}_r \\ \delta\delta_r \end{bmatrix} + \boldsymbol{\epsilon}, \ \mathbf{G} = \begin{bmatrix} -\hat{\mathbf{x}}_{us_1} & -c \\ \vdots & \vdots \\ -\hat{\mathbf{x}}_{us_{n_s}} & -c \end{bmatrix}. \tag{3.13}$$

This gives the least squares solution,

$$\begin{bmatrix} \delta\hat{\mathbf{x}}_r \\ \delta\hat{\delta}_r \end{bmatrix} = (\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T\delta\boldsymbol{\rho}_r^s. \tag{3.14}$$

Least squares estimation is the method that is used to estimate the relative position measurement. The proposed solution will not be using the pseudorange measurements but instead be based on GPS RTK.

3.4 GPS Real-Time Kinematics

GPS Real-Time Kinematics (RTK) is based on the principle of combining a more accurate GPS measurement, called carrier phase measurement (ϕ_r^s), from a reference receiver to calculate the receiver coordinates [23]. This relationship is usually defined as a 'base tower' and 'rover' relation. With the 'base tower' being the reference receiver (commonly with known position) and the 'rover' being the receiver with unknown coordinates. The carrier phase is a measurement based on the change in distance between the receiver and satellite from when the satellite was first observed. This, combined with a reference receiver, can produce a highly accurate relative position measurement. If the reference receiver has known absolute coordinates, the estimation will result in an accurate absolute position [24]. One flaw in this method is the inaccuracy of the first observed distance as that is sometimes initialized based on the first pseudorange measurement, which is a more inaccurate measurement [25]. This inaccuracy is called phase integer ambiguity and is something that needs to be found to gain accurate measurements. This is also a problem in the form of so-called cycle slips [26, 27], which means that the connection between receiver and satellite has been lost and therefore a re-initialization of the localization algorithm needs to occur. This means that a new first observation occurs and therefore also a new phase integer ambiguity.

The carrier phase ϕ_r^s, similarly to the pseudorange ρ_r^s, is modeled based on the known errors in the GPS measurement (ionospheric, etc.)

$$\phi_r^s = \frac{1}{\lambda^s} p_r^s + \aleph_r^s + f^s(\delta^s - \delta_r) + I_r^s + T_r^s. \tag{3.15}$$

With the same variables as for the pseudorange, in addition to λ^s, which is the known wavelength of the satellite signal, $\aleph_r^s$, which is the phase integer ambiguity between the receiver and the satellite, and f^s, which is the known frequency of the satellite signal.

Rearranging the carrier phase equation so all known variables are on the left-hand side and all unknown variables are on the right-hand side, gives

$$\phi_r^s - f^s\delta^s - I_r^s - T_r^s = \frac{1}{\lambda^s} p_r^s + \aleph_r^s - f^s\delta_r. \tag{3.16}$$

From (3.15) it is possible to discern that the inequality $n_s n_t \geq 3 + n_s + n_t$ needs to be satisfied to be able to calculate the receiver position where $n_s n_t$ is the amount of equations and $3 + n_s + n_t$ is the amount of unknown variables.

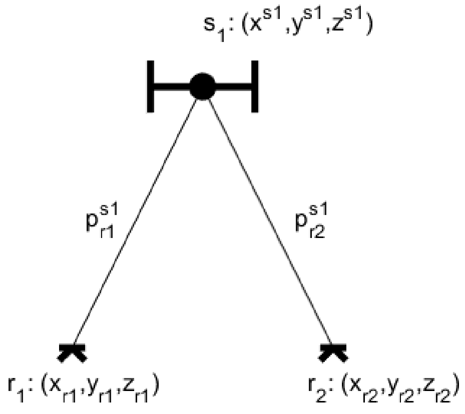


Figure 3.4: Single Difference setup which contains two receivers (r_1 and r_2) and one satellite (s_1). The system uses the difference of the carrier phase measurements between r_1 and r_2 based on their true range to the satellite ($p^{s_1}_{r_1}$ and $p^{s_1}_{r_2}$).

To simplify the estimation it is possible to combine measurements from a reference receiver, which decreases the number of unknowns. These common combinations are either a Single Difference (SD) system, Double Difference (DD) system, or Triple Difference (TD) system. A SD system is modeled based on having one satellite and two receivers, which reduces the satellite and atmospheric errors on the system. This can be observed in Figure 3.4 and gives the equation system

$$\phi^{s_1}_{r_1} - f^{s_1}\delta^{s_1} - I^{s_1}_{r_1} - T^{s_1}_{r_1} = \frac{1}{\lambda^{s_1}} p^{s_1}_{r_1} + \aleph^{s_1}_{r_1} - f^{s_1}\delta_{r_1}, \tag{3.17}$$

$$\phi^{s_1}_{r_2} - f^{s_1}\delta^{s_1} - I^{s_1}_{r_2} - T^{s_1}_{r_2} = \frac{1}{\lambda^{s_1}} p^{s_1}_{r_2} + \aleph^{s_1}_{r_2} - f^{s_1}\delta_{r_2}. \tag{3.18}$$

Since the receivers r_1 and r_2 are relatively close to each other, the ionospheric and tropospheric errors are approximated to be equal, $I^{s_1}_{r_1} = I^{s_1}_{r_2}$ and $T^{s_1}_{r_1} = T^{s_1}_{r_2}$. Subtracting $\phi^{s_1}_{r_1}$ with $\phi^{s_1}_{r_2}$ gives the following expression

$$\phi^{s_1}_{r_1} - \phi^{s_1}_{r_2} = \frac{1}{\lambda^{s_1}}(p^{s_1}_{r_1} - p^{s_1}_{r_2}) + \aleph^{s_1}_{r_1} - \aleph^{s_1}_{r_2} - f^{s_1}(\delta_{r_1} - \delta_{r_2}), \tag{3.19}$$

where the known errors have been eliminated. The expression (3.19) is rewritten as

$$\phi^{s_1}_{r_1 r_2} = \frac{1}{\lambda^{s_1}} p^{s_1}_{r_1 r_2} + \aleph^{s_1}_{r_1 r_2} - f^{s_1}\delta_{r_1 r_2}, \tag{3.20}$$

where

$$\phi^{s_1}_{r_1 r_2} = \phi^{s_1}_{r_1} - \phi^{s_1}_{r_2}, \tag{3.21}$$

$$p^{s_1}_{r_1 r_2} = p^{s_1}_{r_1} - p^{s_1}_{r_2}, \tag{3.22}$$

$$\aleph^{s_1}_{r_1 r_2} = \aleph^{s_1}_{r_1} - \aleph^{s_1}_{r_2}, \tag{3.23}$$

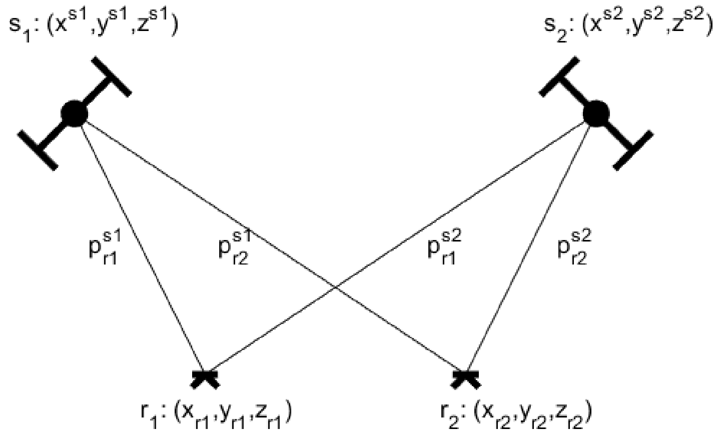


Figure 3.5: Double Difference setup which uses an additional satellite in the system. The DD system looks at the difference of the SD systems.

$$\delta^{s_1}_{r_1r_2} = \delta^{s_1}_{r_1} - \delta^{s_1}_{r_2}. \tag{3.24}$$

This notation is used in future calculations to shorten the expressions. Using SD system only eliminates the errors that are assumed to be known in this thesis, which do not simplify the objective of obtaining the receiver position. To eliminate the unknown errors the system is further extended into DD and TD.

A DD system is modeled based on having two satellites and two receivers, which, in addition to removing the satellite and atmospheric errors, also removes the receiver clock error. Using measurements from the same frequency bands means that the satellites signal frequency and wavelength are equal. The DD system can be observed in Figure 3.5 and gives the equation system

$$\phi^{s_1}_{r_1r_2} = \frac{1}{\lambda}p^{s_1}_{r_1r_2} + \aleph^{s_1}_{r_1r_2} - f\delta_{r_1r_2}, \tag{3.25}$$

$$\phi^{s_2}_{r_1r_2} = \frac{1}{\lambda}p^{s_2}_{r_1r_2} + \aleph^{s_2}_{r_1r_2} - f\delta_{r_1r_2}, \tag{3.26}$$

where (3.25) and (3.26) have been obtained by using SD system (3.20) for both the satellites s_1 and s_2. The DD system is obtained by subtracting (3.25) with (3.26) which results in

$$\phi^{s_1}_{r_1r_2} - \phi^{s_2}_{r_1r_2} = \frac{1}{\lambda}(p^{s_1}_{r_1r_2} - p^{s_2}_{r_1r_2}) + \aleph^{s_1}_{r_1r_2} - \aleph^{s_2}_{r_1r_2}, \tag{3.27}$$

where the receiver clock bias $\delta_{r_1r_2}$ has been eliminated. The DD system (3.27) can be shortened to

$$\phi^{s_1s_2}_{r_1r_2} = \frac{1}{\lambda}p^{s_1s_2}_{r_1r_2} + \aleph^{s_1s_2}_{r_1r_2}, \tag{3.28}$$

where

$$\phi^{s_1 s_2}_{r_1 r_2} = \phi^{s_1}_{r_1 r_2} - \phi^{s_2}_{r_1 r_2}, \tag{3.29}$$

$$p^{s_1 s_2}_{r_1 r_2} = p^{s_1}_{r_1 r_2} - p^{s_2}_{r_1 r_2}, \tag{3.30}$$

$$\aleph^{s_1 s_2}_{r_1 r_2} = \aleph^{s_1}_{r_1 r_2} - \aleph^{s_2}_{r_1 r_2}. \tag{3.31}$$

A TD system is modeled based on having two satellites, two receivers and two time epochs, which, in addition to removing satellite, atmospheric, and receiver clock error, eliminates phase integer ambiguity. The TD system assumes constant phase integer ambiguities through both of its time epochs, which means that the TD system assumes no cycle slip has occurred between the epochs. As it is not possible to avoid cycle slips in practice, the TD equations can only be used to identify the occurrence of cycle slips. In this thesis, the occurrence of cycle slips are not added to the simulations, which means that the TD estimation can be fully implemented. The TD system can be derived as

$$\phi^{s_1 s_2}_{r_1 r_2}[k] = \frac{1}{\lambda} p^{s_1 s_2}_{r_1 r_2}[k] + \aleph^{s_1 s_2}_{r_1 r_2}, \tag{3.32}$$

$$\phi^{s_1 s_2}_{r_1 r_2}[k-1] = \frac{1}{\lambda} p^{s_1 s_2}_{r_1 r_2}[k-1] + \aleph^{s_1 s_2}_{r_1 r_2}. \tag{3.33}$$

where (3.32) and (3.33) has been obtained by the DD system (3.28). Subtracting (3.32) with (3.33) gives the expression for the TD system

$$\phi^{s_1 s_2}_{r_1 r_2}[k] - \phi^{s_1 s_2}_{r_1 r_2}[k-1] = \frac{1}{\lambda}(p^{s_1 s_2}_{r_1 r_2}[k] - p^{s_1 s_2}_{r_1 r_2}[k-1]) \tag{3.34}$$

where the phase integer ambiguity ($\aleph^{s_1 s_2}_{r_1 r_2}$) has been eliminated. The TD can be written as

$$\phi^{s_1 s_2}_{r_1 r_2}[k, k-1] = \frac{1}{\lambda} p^{s_1 s_2}_{r_1 r_2}[k, k-1]. \tag{3.35}$$

where

$$\phi^{s_1 s_2}_{r_1 r_2}[k, k-1] = \phi^{s_1 s_2}_{r_1 r_2}[k] - \phi^{s_1 s_2}_{r_1 r_2}[k-1], \tag{3.36}$$

$$p^{s_1 s_2}_{r_1 r_2}[k, k-1] = p^{s_1 s_2}_{r_1 r_2}[k] - p^{s_1 s_2}_{r_1 r_2}[k-1]. \tag{3.37}$$

The TD and DD equation systems that are derived in this section is the equation systems that the previous mentioned least squares estimation is applied to, to estimate the relative position between the rovers. The implementation of least squares estimation onto the DD and TD estimations is presented below.

3.4.1 Solving TD and DD systems with least squares

With both the DD and TD systems defined, it is now possible to apply least squares estimation to estimate the unknown receiver coordinates $\mathbf{x}_{r_2}[k]$. As mentioned previously, the reference receiver coordinates $\mathbf{x}_{r_1}[k]$ are known in both the DD and TD system. The previous receiver coordinate ($\mathbf{x}_{r_2}[k-1]$) occurring in the TD system are assumed to be the previous epochs estimation and therefore known. Starting with the TD estimation and expanding (3.35) gives

$$\begin{aligned}\phi_{r_1r_2}^{s_1s_2}[k,k-1] = \frac{1}{\lambda}(p_{r_1}^{s_1}[k] - p_{r_2}^{s_1}[k] - p_{r_1}^{s_2}[k] + p_{r_2}^{s_2}[k]- \\ p_{r_1}^{s_1}[k-1] + p_{r_2}^{s_1}[k-1] + p_{r_1}^{s_2}[k-1] - p_{r_2}^{s_2}[k-1]),\end{aligned} \tag{3.38}$$

where

$$p_r^s[k] = ||\mathbf{x}^s[k] - \mathbf{x}_r[k]||. \tag{3.39}$$

As mentioned previously, all variables are known in the system with the exception of $\mathbf{x}_{r_2}[k]$, which are the coordinates of the unknown receiver. The known quantities can, therefore, be contained in a constant

$$C = p_{r_1}^{s_1}[k] - p_{r_1}^{s_2}[k] - p_{r_1}^{s_1}[k-1] + p_{r_2}^{s_1}[k-1] + p_{r_1}^{s_2}[k-1] - p_{r_2}^{s_2}[k-1], \tag{3.40}$$

thus expression (3.38) can be rewritten as

$$\phi_{r_1r_2}^{s_1s_2}[k,k-1] = \frac{1}{\lambda}(C - ||\mathbf{x}^{s_1}[k] - \mathbf{x}_{r_2}[k]|| + ||\mathbf{x}^{s_2}[k] - \mathbf{x}_{r_2}[k]||). \tag{3.41}$$

Combining the TD system (3.35) together with the least squares estimation method means that we want to find $\delta\mathbf{x}_{r_2}[k]$ so that the true value $\mathbf{x}_{r_2}[k] = \delta\mathbf{x}_{r_2}[k] + \mathbf{x}_{r_2}[k]_0$ will solve

$$\delta\phi_{r_1r_2}^{s_1s_2}[k,k-1] = \phi_{r_1r_2}^{s_1s_2}[k,k-1] - \phi_{r_1r_2}^{s_1s_2}[k,k-1]_0, \quad \delta\phi_{r_1r_2}^{s_1s_2}[k,k-1] \to 0, \forall s, \tag{3.42}$$

using the initial estimation $x_{r_2}[k]_0$. This gives the equation

$$\begin{aligned}\delta\phi_{r_1r_2}^{s_1s_2}[k,k-1] = \frac{1}{\lambda}(C - ||\mathbf{x}^{s_1}[k] - \mathbf{x}_{r_2}[k]|| + ||\mathbf{x}^{s_2}[k] - \mathbf{x}_{r_2}[k]||)- \\ \frac{1}{\lambda}(C - ||\mathbf{x}^{s_1}[k] - \mathbf{x}_{r_2}[k]_0|| + ||\mathbf{x}^{s_2}[k] - \mathbf{x}_{r_2}[k]_0||) + \epsilon^{s_1s_2},\end{aligned} \tag{3.43}$$

where $\epsilon^{s_1s_2}$, same as in (3.6), is the combination of unmodeled errors. Rearranging and simplifying (3.43) gives

$$\begin{aligned}\lambda\delta\phi_{r_1r_2}^{s_1s_2}[k,k-1] = -(||\mathbf{x}^{s_1}[k] - \mathbf{x}_{r_2}[k]|| - ||\mathbf{x}^{s_1}[k] - \mathbf{x}_{r_2}[k]_0||) \\ +(||\mathbf{x}^{s_2}[k] - \mathbf{x}_{r_2}[k]|| - ||\mathbf{x}^{s_2}[k] - \mathbf{x}_{r_2}[k]_0||) + \lambda\epsilon^{s_1s_2}.\end{aligned} \tag{3.44}$$

Replacing $\mathbf{x}_{r_2}[k]$ with $\delta\mathbf{x}_{r_2}[k] + \mathbf{x}_{r_2}[k]_0$ gives

$$\begin{aligned}\lambda\delta\phi^{s_1 s_2}_{r_1 r_2}[k, k-1] = -(||\mathbf{x}^{s_1}[k] - (\delta\mathbf{x}_{r_2}[k] + \mathbf{x}_{r_2}[k]_0)|| - ||\mathbf{x}^{s_1}[k] - \mathbf{x}_{r_2}[k]_0||) \\ +||\mathbf{x}^{s_2}[k] - (\delta\mathbf{x}_{r_2}[k] + \mathbf{x}_{r_2}[k]_0)|| - ||\mathbf{x}^{s_2}[k] - \mathbf{x}_{r_2}[k]_0||) + \lambda\epsilon^{s_1 s_2}.\end{aligned} \tag{3.45}$$

This can be further simplified by using Taylor series expansion of vector modulus [22] resulting in the expression

$$\lambda\delta\phi^{s_1 s_2}_{r_1 r_2}[k, k-1] = -(-\frac{\mathbf{x}^{s_1} - \mathbf{x}_{r_2}[k]_0}{||\mathbf{x}^{s_1} - \mathbf{x}_{r_2}[k]_0||}\delta\mathbf{x}_{r_2}[k]) + (-\frac{\mathbf{x}^{s_2} - \mathbf{x}_{r_2}[k]_0}{||\mathbf{x}^{s_2} - \mathbf{x}_{r_2}[k]_0||}\delta\mathbf{x}_{r_2}[k]) + \lambda\epsilon^{s_1 s_2}. \tag{3.46}$$

The end result will then become

$$\delta\phi^{s_1 s_2}_{r_1 r_2}[k, k-1] = \left[\tfrac{1}{\lambda}(\hat{\mathbf{x}}^{s_1}_{ur_2}[k] - \hat{\mathbf{x}}^{s_2}_{ur_2}[k])\right]\left[\delta\mathbf{x}_{r_2}[k]\right] + \epsilon^{s_1 s_2}, \tag{3.47}$$

where

$$\hat{\mathbf{x}}^{s_1}_{ur_2}[k] = \frac{\mathbf{x}^{s_1} - \mathbf{x}_{r_2}[k]_0}{||\mathbf{x}^{s_1} - \mathbf{x}_{r_2}[k]_0||}, \tag{3.48}$$

and

$$\hat{\mathbf{x}}^{s_2}_{ur_2}[k] = \frac{\mathbf{x}^{s_2} - \mathbf{x}_{r_2}[k]_0}{||\mathbf{x}^{s_2} - \mathbf{x}_{r_2}[k]_0||}. \tag{3.49}$$

Repeating this for n_s satellites results in

$$\delta\boldsymbol{\phi}_{r_1 r_2}[k, k-1] = \begin{bmatrix}\delta\phi^{s_1 s_2}_{r_1 r_2}[k, k-1] \\ \vdots \\ \delta\phi^{s_1 s_n}_{r_1 r_2}[k, k-1]\end{bmatrix} = \mathbf{G}\left[\delta\mathbf{x}_{r_2}[k]\right] + \boldsymbol{\epsilon}, \tag{3.50}$$

where

$$\mathbf{G} = \begin{bmatrix}\frac{1}{\lambda}(\hat{\mathbf{x}}^{s_1}_{ur_2}[k] - \hat{\mathbf{x}}^{s_2}_{ur_2}[k]) \\ \vdots \\ \frac{1}{\lambda}(\hat{\mathbf{x}}^{s_1}_{ur_2}[k] - \hat{\mathbf{x}}^{s_{n_s}}_{ur_2}[k])\end{bmatrix}, \quad \boldsymbol{\epsilon} = \begin{bmatrix}\epsilon^{s_1 s_2} \\ \vdots \\ \epsilon^{s_1 s_n}\end{bmatrix}. \tag{3.51}$$

This gives the least squares solution

$$\delta\hat{\mathbf{x}}_{r_2} = (\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T\delta\boldsymbol{\phi}_{r_1 r_2}[k, k-1]. \tag{3.52}$$

For the DD system the least squares estimation also has to estimate the unknown phase integer ambiguities $\aleph^{s_1 s_2}_{r_1 r_2}$. The interesting difference with the ambiguities are that they are constant given that no cycle slip occurs. This means that once the ambiguities are found the estimations only needs to estimate the receiver coordinates. The DD estimation is a highly researched field within the area of GPS as it is of interest to both find ways to increase the accuracy of the estimations, and to quickly find the ambiguities to improve the estimations.

The DD estimations requires solving the expression

$$\delta\phi^{s_1s_2}_{r_1r_2} = \phi^{s_1s_2}_{r_1r_2} - \phi^{s_1s_2}_{r_1r_2 0}, \quad \delta\phi^{s_1s_2}_{r_1r_2} \rightarrow 0, \forall s. \tag{3.53}$$

Same as for the TD estimation but with the addition of the ambiguities, the unknowns becomes

$$\mathbf{x}_{r_2}[k] = \delta\mathbf{x}_{r_2}[k] + \mathbf{x}_{r_2}[k]_0, \tag{3.54}$$

and

$$\aleph^{s_1s_2}_{r_1r_2} = \delta\aleph^{s_1s_2}_{r_1r_2} + \aleph^{s_1s_2}_{r_1r_2 0}. \tag{3.55}$$

Following the same idea as in the TD estimation (3.38)-(3.52) the DD estimation (3.53) results in

$$\delta\boldsymbol{\phi}_{r_1r_2}[k] = \mathbf{G}\begin{bmatrix}\delta\mathbf{x}_{r_2}[k]\\ \delta\aleph^{s_1s_2}_{r_1r_2}\\ \vdots\\ \delta\aleph^{s_1s_{n_s}}_{r_1r_2}\end{bmatrix} + \boldsymbol{\epsilon}, \tag{3.56}$$

where

$$\mathbf{G} = \begin{bmatrix}\frac{1}{\lambda}(\hat{\mathbf{x}}^{s_1}_{ur_2}[k] - \hat{\mathbf{x}}^{s_2}_{ur_2}[k]) & \\ \vdots & \mathbf{I}^{n_s\times n_s}\\ \frac{1}{\lambda}(\hat{\mathbf{x}}^{s_1}_{ur_2}[k] - \hat{\mathbf{x}}^{s_{n_s}}_{ur_2}[k]) & \end{bmatrix}, \quad \boldsymbol{\epsilon} = \begin{bmatrix}\epsilon^{s_1s_2}\\ \vdots\\ \epsilon^{s_1s_n}\end{bmatrix}. \tag{3.57}$$

The variable $\mathbf{I}^{n_s\times n_s}$ in (3.57) represents the identity matrix of size n_s. The least squares solution of DD system is given by

$$\begin{bmatrix}\delta\mathbf{x}_{r_2}[k]\\ \delta\aleph^{s_1s_2}_{r_1r_2}\\ \vdots\\ \delta\aleph^{s_1s_n}_{r_1r_2}\end{bmatrix} = (\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T\delta\boldsymbol{\phi}_{r_1r_2}[k]. \tag{3.58}$$

The TD and DD estimations ((3.52) and (3.58)) are used to estimate the unknown receiver coordinates and used as the relative position measurement in the system. In Section 4.2.5 it is further explained how the TD and DD estimations are combined with the rest of the system.

4
Sensor fusion

The idea of sensor fusion is to combine different sensor data to get an overall improvement in the estimation of a systems states [28]. It also includes combining known information about the system together with the measurements to improve the information further [29]. An example of using known information about what is measured could be seen in the GPS example in Section 3, where the two different solutions of the GPS trilateration would either end up on a road or in water. With awareness of the movement of the vehicle on a road, the system can disregard the position over water as it is known that the vehicle is traveling on a road. In general, by being aware of the motion of the observable object, the estimated state could be significantly increased as certain erroneous values can be disregarded.

4.1 Kalman filter

The Kalman filter is a commonly used sensor fusion algorithm that takes measurement data from multiple sensors to better estimate the future state of the model [29]. The basic Kalman filter enables a higher accuracy by considering the joint probability of multiple sensors over time instead of looking at individual measurements [29]. It works on linear systems and the calculated estimation of the Kalman filter is the linear minimum mean square error (LMMSE) [29]. For a nonlinear system, there exists a different version of the Kalman filter, the Extended Kalman Filter (EKF), which includes the linearization of the system around an operating point [29], see Section 4.1.1 for more details. The Kalman filter uses the system's motion model (also called process model) to improve the estimation based on the known dynamics of the system. It also uses the system's measurement model which describes the relation between the states and the measurements, which is used to improve the estimation based on the combined measured values of the system. All equations presented in this chapter and their derivation can be found in [30].

The motion model describes the update of the states, $\mathbf{x}_k \in \mathbb{R}^n$, from time step $k-1$ to k. A linear discrete-time motion model is described by

$$\mathbf{x}_k = \mathbf{A}_{k-1}\mathbf{x}_{k-1} + \mathbf{B}_{k-1}\mathbf{u}_{k-1} + \Gamma\mathbf{q}_{k-1}. \tag{4.1}$$

- Transition matrix $\mathbf{A}_{k-1} \in \mathbb{R}^{n\times n}$ which describes the transition of the states $\mathbf{x}$ from time step $k-1$ to k.
- Input matrix $\mathbf{B}_{k-1} \in \mathbb{R}^{n\times l}$ which describes the effect of the control input, $\mathbf{u}_{k-1} \in \mathbb{R}^l$, upon the states.

- Motion noise $\mathbf{q}_{k-1} \in \mathbb{R}^q$ which is a random variable that describes the unknown motion noise of the model. It is assumed to be zero mean normally distributed noise with covariance matrix $\mathbf{Q}_{k-1}$, $\mathbf{q}_{k-1} \sim \mathcal{N}(0, \mathbf{Q}_{k-1})$.
- Noise matrix $\Gamma \in \mathbb{R}^{n \times q}$ which describes which states that are affected by the motion noise.

The measurement model describes the relation between the states $\mathbf{x}_k$ and the measurement $\mathbf{y}_k \in \mathbb{R}^m$. The measurement model,

$$\mathbf{y}_k = \mathbf{H}_k \mathbf{x}_k + \Phi \mathbf{r}_k, \tag{4.2}$$

includes the measurement matrix, $\mathbf{H}_k \in \mathbb{R}^{m \times n}$, the measurement noise, $\mathbf{r}_k \in \mathbb{R}^r$, and the noise matrix $\Phi \in \mathbb{R}^{m \times r}$. The measurement matrix $\mathbf{H}_k$ describes the exact relation between the states and measurements while the measurement noise $\mathbf{r}_k$ is a zero mean normally distributed random variable with covariance matrix $\mathbf{R}_k$, $\mathbf{r}_k \sim \mathcal{N}(0, \mathbf{R}_k)$, and the noise matrix Φ describes which measurements are affected by the noise.

The Kalman filter firstly calculates a predicted estimate, $\hat{\mathbf{x}}_{k|k-1}$, and predicted covariance, $\mathbf{P}_{k|k-1}$, using knowledge based on the previous time step $k-1$. This step is called the prediction step. Thereafter, using the calculated $\hat{\mathbf{x}}_{k|k-1}$ and $\mathbf{P}_{k|k-1}$, the Kalman filter calculates the estimated value of $\mathbf{x}_k$ based on the new measurement $\mathbf{y}_k$. This step calculates the *posteriori* estimate ($\hat{\mathbf{x}}_{k|k}$) and is called the measurement update step. The posteriori is the mean value of the joint probability $p(\mathbf{x}_k|\mathbf{y}_{1:k}) = \mathcal{N}(\hat{\mathbf{x}}_{k|k}, \mathbf{P}_{k|k})$, and has the covariance $\mathbf{P}_{k|k}$. Similiarly for the predicted estimate $\hat{\mathbf{x}}_{k|k-1}$, which is the mean of the normally distributed joint probability $p(\mathbf{x}_k|\mathbf{y}_{1:k-1}, \mathbf{u}_{k-1}) = \mathcal{N}(\hat{\mathbf{x}}_{k|k-1}, \mathbf{P}_{k|k-1})$, with a covariance of $\mathbf{P}_{k|k-1}$.

In the prediction step, the predicted estimate $\hat{\mathbf{x}}_{k|k-1}$ is calculated using the motion model based on the previous posteriori $\hat{\mathbf{x}}_{k-1|k-1}$ and input signal $\mathbf{u}_{k-1}$,

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{A}_{k-1} \hat{\mathbf{x}}_{k-1|k-1} + \mathbf{B}_{k-1} \mathbf{u}_{k-1}, \tag{4.3}$$

$$\mathbf{P}_{k|k-1} = \mathbf{A}_{k-1} \mathbf{P}_{k-1|k-1} \mathbf{A}_{k-1}^T + \Gamma \mathbf{Q}_{k-1} \Gamma^T. \tag{4.4}$$

In the measurement update step, the posteriori $\hat{\mathbf{x}}_{k|k}$ is calculated using the predicted estimate ($\hat{\mathbf{x}}_{k|k-1}$), the measurement $\mathbf{y}_k$ and the measurement model,

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \mathbf{v}_k, \tag{4.5}$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{S}_k \mathbf{K}_k^T. \tag{4.6}$$

Where $\mathbf{K}_k$ is the Kalman gain

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{S}_k^{-1}, \tag{4.7}$$

$\mathbf{v}_k$ is the innovation

$$\mathbf{v}_k = \mathbf{y}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k|k-1}, \tag{4.8}$$

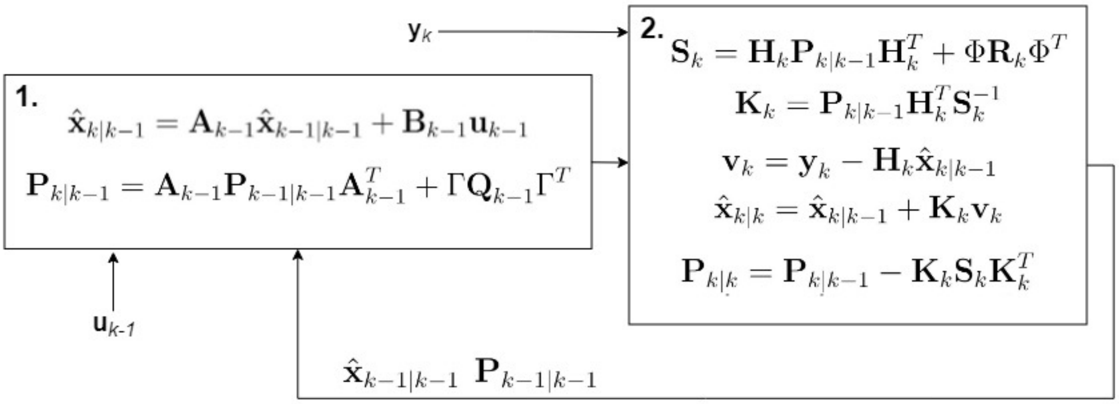

$\mathbf{y}_k$
2.
$\mathbf{S}_k = \mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \Phi\mathbf{R}_k\Phi^T$
$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^T\mathbf{S}_k^{-1}$
$\mathbf{v}_k = \mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}$
$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\mathbf{v}_k$
$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{S}_k\mathbf{K}_k^T$
1.
$\hat{\mathbf{x}}_{k|k-1} = \mathbf{A}_{k-1}\hat{\mathbf{x}}_{k-1|k-1} + \mathbf{B}_{k-1}\mathbf{u}_{k-1}$
$\mathbf{P}_{k|k-1} = \mathbf{A}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{A}_{k-1}^T + \Gamma\mathbf{Q}_{k-1}\Gamma^T$
$\mathbf{u}_{k-1}$
$\hat{\mathbf{x}}_{k-1|k-1}$ $\mathbf{P}_{k-1|k-1}$

Figure 4.1: Overview of the Kalman filter. Step 1 refers to the prediction step, and step 2 refers to the measurement update step.

and $\mathbf{S}_k$ is the innovation covariance

$$\mathbf{S}_k = \mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \Phi\mathbf{R}_k\Phi^T. \tag{4.9}$$

The complete Kalman filter model can be observed in Figure 4.1. The important thing to note is that the only input to the Kalman filter is the control input signal $\mathbf{u}_{k-1}$ and the measurement $\mathbf{y}_k$, assuming the state matrices are constant with respect to time. The filter requires the definition of the initial condition $\hat{\mathbf{x}}_0$ and $\mathbf{P}_0$.

4.1.1 Extended Kalman Filter

As mentioned previously, the normal Kalman filter only works for linear systems. To deal with nonlinear systems, one possible algorithm would be the EKF. The idea of the EKF is to use the Kalman filter on the linearized motion and measurement model. The linearized model of the non-linear motion model (4.1),

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}) + \mathbf{B}_{k-1}\mathbf{u}_{k-1} + \Gamma\mathbf{q}_{k-1}, \tag{4.10}$$

can be found by the first order Taylor expansion of the non-linear model. The first order Taylor series expansion of a nonlinear function, $\mathbf{y} = g(\mathbf{x})$, around a point $\hat{\mathbf{x}}$ can be approximated as

$$\mathbf{y} \approx g(\hat{\mathbf{x}}) + g'(\hat{\mathbf{x}})(\mathbf{x} - \hat{\mathbf{x}}), \tag{4.11}$$

where $g'(\hat{\mathbf{x}})$ is the jacobian of $g(\hat{\mathbf{x}})$, and $\mathbf{x} \sim \mathcal{N}(\hat{\mathbf{x}}, \mathbf{P})$. An interesting observation from this is that the distribution of the linearized system has changed to $\mathbb{E}(\mathbf{x}) \approx g(\hat{\mathbf{x}})$ and $\mathrm{cov}(\mathbf{y}) \approx g'(\hat{\mathbf{x}})\mathbf{P}g'(\hat{\mathbf{x}})^T$, which affects the prediction and update step.

The first order Taylor expansion of the motion model results in

$$\mathbf{x}_k \approx f(\hat{\mathbf{x}}_{k-1|k-1}) + f'(\hat{\mathbf{x}}_{k-1|k-1})(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1|k-1}) + \mathbf{B}_{k-1}\mathbf{u}_{k-1} + \Gamma\mathbf{q}_{k-1} \tag{4.12}$$

as the motion model is linearized around the estimated values, $\mathbb{E}(\mathbf{x}_{k-1}) = \hat{\mathbf{x}}_{k-1|k-1}$. The linearization of the measurement model

$$\mathbf{y}_k = h(\mathbf{x}_k) + \Phi\mathbf{r}_k, \tag{4.13}$$

similarly results in

$$\mathbf{y}_k \approx h(\hat{\mathbf{x}}_{k|k-1}) + h'(\hat{\mathbf{x}}_{k|k-1})(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1}) + \Phi\mathbf{r}_k. \tag{4.14}$$

After the linearization of the motion and measurement model the prediction step becomes

$$\hat{\mathbf{x}}_{k|k-1} = f(\hat{\mathbf{x}}_{k-1|k-1}) + \mathbf{B}_{k-1}u_{k-1}, \tag{4.15}$$

$$\mathbf{P}_{k|k-1} = f'(\hat{\mathbf{x}}_{k-1|k-1})\mathbf{P}_{k-1|k-1}f'(\hat{\mathbf{x}}_{k-1|k-1})^T + \Gamma\mathbf{Q}_{k-1}\Gamma^T. \tag{4.16}$$

And the measurement update step becomes

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\mathbf{v}_k, \tag{4.17}$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{S}_k\mathbf{K}_k^T, \tag{4.18}$$

$$\mathbf{v}_k = \mathbf{y}_k - h(\hat{\mathbf{x}}_{k|k-1}), \tag{4.19}$$

$$\mathbf{S}_k = h'(\hat{\mathbf{x}}_{k|k-1})\mathbf{P}_{k|k-1}h'(\hat{\mathbf{x}}_{k|k-1})^T + \Phi\mathbf{R}_k\Phi^T, \tag{4.20}$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}h'(\hat{\mathbf{x}}_{k|k-1})^T\mathbf{S}_k^{-1}. \tag{4.21}$$

The EKF is the sensor fusion algorithm that is used to estimate the leader's and follower's states. The exact EKF implemented in this thesis is presented in Section 4.2, which ends with the combination of the EKF and the remaining parts of the system (Section 4.2.4 and 4.2.5).

4.2 Complete Sensor Model

The complete sensor model includes both rovers, where one acts as a base tower and the other as a rover in accordance to the RTK setup (see Section 3.4). Both rovers have individual EKFs to estimate the states. The states that are used differ slightly in comparison to the vehicle state space model defined in Section 2.4, specifically that it calculates the global position instead of the local coordinates. The reason why the sensor model uses the global coordinates is because the position measurement is in global coordinates and that the global coordinates are needed in the path planning algorithm (see Chapter 5). The local coordinates is only used in the implementation of the Linear-Quadratic Regulator (LQR) (see Section 6.3). The states

$$\mathbf{x}_k = \begin{bmatrix} X_k \\ Y_k \\ v_k \\ \psi_k \\ \dot{\psi}_k \end{bmatrix}, \tag{4.22}$$

contain the global position (X_k, Y_k), the velocity v_k, the heading ψ_k, and the yaw rate $\dot{\psi}_k$. The most important note is that the global position of the rovers does not refer to the ECEF frame, but to a self-defined global coordinate system. This is because the system does not include a third reference point, an actual base tower, which means that the RTK solution only gives a high accuracy result for the relative distance between the two rovers. The self-defined global coordinate systems are based around the leader's initial position being an arbitrary value. Thereafter, the leader's position and states will update according to the Kalman filter. Important to note is that the GPS will never be used in reference to calculate the global coordinate of the leading rover. The GPS will only be used with the RTK solution to accurately estimate the relative distance between both rovers.

4.2.1 Motion Model

The motion model of the rovers is based on the coordinated turn model found in [31] and the vehicle state space model defined in Section 2.4. It is a nonlinear motion model that updates the global coordinates based on the heading and speed of the vehicle,

$$\begin{bmatrix} X_k \\ Y_k \\ v_k \\ \psi_k \\ \dot{\psi}_k \end{bmatrix} = \begin{bmatrix} X_{k-1} + Tv_{k-1}\cos\psi_{k-1} \\ Y_{k-1} + Tv_{k-1}\sin\psi_{k-1} \\ 0 \\ \psi_{k-1} + T\dot{\psi}_{k-1} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{r}{2} & \frac{r}{2} \\ 0 & 0 \\ -\frac{r}{2L} & \frac{r}{2L} \end{bmatrix} \begin{bmatrix} \omega_l \\ \omega_r \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} q_v \\ q_{\dot{\psi}} \end{bmatrix}. \tag{4.23}$$

This model's velocity and yaw rate is purely based upon the control input of the model and uses the same variables as defined in Section 2.2. In (4.23), T is the time between time steps $k-1$ and k, q_v is the motion noise of the vehicles velocity and $q_{\dot{\psi}}$ is the motion noise of the vehicles yaw rate.

The motion model can be described in the same way as the nonlinear motion model (4.10) described in Section 4.1.1 with

$$f(\mathbf{x}_{k-1}) = \begin{bmatrix} X_{k-1} + Tv_{k-1}\cos\psi_{k-1} \\ Y_{k-1} + Tv_{k-1}\sin\psi_{k-1} \\ 0 \\ \psi_{k-1} + T\dot{\psi}_{k-1} \\ 0 \end{bmatrix}. \tag{4.24}$$

The jacobian of $f(\mathbf{x}_{k-1})$ becomes

$$f'(\mathbf{x}_{k-1}) = \begin{bmatrix} 1 & 0 & T\cos(\psi_{k-1}) & -Tv_{k-1}\sin(\psi_{k-1}) & 0 \\ 0 & 1 & T\sin(\psi_{k-1}) & Tv_{k-1}\cos(\psi_{k-1}) & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{4.25}$$

The defined motion model is implemented in both the leading and the following rover's EKF.

4.2.2 Measurement Model

The measurement model differs between the different rovers as the leading rover does not have the GPS value as reference for the estimation of position. Instead, the position is only estimated based on the measurement of the IMU and wheel speed sensors (see Appendix A.2 and A.3). The measurements that are of interest for the leader are

$$\mathbf{y}_k^L = \begin{bmatrix} \tilde{\omega}_{l,k}^L \\ \tilde{\omega}_{r,k}^L \\ \tilde{\psi}_k^L \\ \tilde{\dot{\psi}}_k^L \end{bmatrix}. \tag{4.26}$$

- Measurement $\tilde{\omega}_{l,k}^L$, the left wheels rotational speed, is the output of the wheel speed sensor on the rover's left wheels measured in angular velocity.
- Measurement $\tilde{\omega}_{r,k}^L$, the right wheels rotational speed, is the output of the wheel speed sensor on the rover's right wheels measured in angular velocity.
- Measurement $\tilde{\psi}_k^L$, the absolute orientation, is the output of the inbuilt sensor fusion algorithm between the gyroscope and the magnetometer from the IMU.
- Measurement $\tilde{\dot{\psi}}_k^L$, the yaw rate, is the output of the gyroscope from the IMU.

The conversion from state $\mathbf{x}_k^L$ to measurement $\mathbf{y}_k^L$ for the different states are all linear

$$\tilde{\omega}_{l,k}^L = \frac{v_k^L}{r} - \frac{L\dot{\psi}_k^L}{r} + r_k^{\omega_l}, \tag{4.27}$$

$$\tilde{\omega}_{r,k}^L = \frac{v_k^L}{r} + \frac{L\dot{\psi}_k^L}{r} + r_k^{\omega_r}, \tag{4.28}$$

$$\tilde{\psi}_k^L = \psi_k^L + r_k^{\psi}, \tag{4.29}$$

$$\tilde{\dot{\psi}}_k^L = \dot{\psi}_k^L + r_k^{\dot{\psi}}, \tag{4.30}$$

with added noise $r_k^{\omega_l}$, $r_k^{\omega_r}$, r_k^{ψ} and $r_k^{\dot{\psi}}$. This results in a completely linear measurement model which we can directly use the standard Kalman filter on

$$\begin{bmatrix} \tilde{\omega}_{l,k}^L \\ \tilde{\omega}_{r,k}^L \\ \tilde{\psi}_k^L \\ \tilde{\dot{\psi}}_k^L \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & \frac{-L}{r} \\ 0 & 0 & 1 & 0 & \frac{L}{r} \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_k^L \\ Y_k^L \\ v_k^L \\ \psi_k^L \\ \dot{\psi}_k^L \end{bmatrix} + \mathbf{I}^{4x4} \begin{bmatrix} r_k^{\omega_l} \\ r_k^{\omega_r} \\ r_k^{\psi} \\ r_k^{\dot{\psi}} \end{bmatrix}. \tag{4.31}$$

For the following rover, the measurement gets extended with the GPS output $(\tilde{X}_k^F, \tilde{Y}_k^F)$,

$$\mathbf{y}_k^F = \begin{bmatrix} \tilde{X}_k^F \\ \tilde{Y}_k^F \\ \tilde{\omega}_{l,k}^F \\ \tilde{\omega}_{r,k}^F \\ \tilde{\psi}_k^F \\ \tilde{\dot{\psi}}_k^F \end{bmatrix}, \tag{4.32}$$

which are direct measurements of a state similar to (4.29) and (4.30) and results in the following measurement model for the following rover

$$\begin{bmatrix} \tilde{X}_k^F \\ \tilde{Y}_k^F \\ \tilde{\omega}_{l,k}^F \\ \tilde{\omega}_{r,k}^F \\ \tilde{\psi}_k^F \\ \tilde{\dot{\psi}}_k^F \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & \frac{-L}{r} \\ 0 & 0 & 1 & 0 & \frac{L}{r} \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_k^F \\ Y_k^F \\ v_k^F \\ \psi_k^F \\ \dot{\psi}_k^F \end{bmatrix} + \mathbf{I}^{6x6} \begin{bmatrix} r_k^X \\ r_k^Y \\ r_k^{\omega_l} \\ r_k^{\omega_r} \\ r_k^{\psi} \\ r_k^{\dot{\psi}} \end{bmatrix} . \tag{4.33}$$

The derived measurements models for the leader (4.31) and the follower (4.33) are the models used in the EKF of each rover.

4.2.3 Filter parameters and initial condition

To enable the usage of the EKF, the two filter parameters $\mathbf{Q}_k$ and $\mathbf{R}_k$, the initial condition of the state estimate, $\hat{\mathbf{x}}_0$, and its covariance, $\mathbf{P}_0$, need to be defined. The only difference between the two rovers will be the measurement noise covariance matrix $\mathbf{R}_k$ as the following rover includes the GPS measurements, as well as their initial condition.

For the initial state, $\hat{\mathbf{x}}_0^L$, of the leading rover, the states will be defined as follows:

- The global coordinate, X_0^L and Y_0^L, will be initialized to an arbitrary value as mentioned in Section 4.2.
- The velocity v_0^L and yaw rate $\dot{\psi}_0^L$ will also be initiated to zero as the rovers start at stand-still.
- The yaw angle ψ_0^L will be initialized with the first measurement it receives.

The initial state of the following rover, $\hat{\mathbf{x}}_0^F$, are defined the same as the leading rover but with the global coordinates initialized to the first measurement it receives from the GPS RTK estimation.

The initial state covariance matrix will be initialized to zero for all states that are initialized to zero and, otherwise, set to corresponding measurement noise covariances. This means that for the leading rover, $\mathbf{P}_0^L$ will be initialized to zero for all states except for the heading ψ_0^L, which will be initialized to the measurement noise covariance σ_ψ^2. The same will be true for the following rover with an addition on the global coordinates which will also be initialized to the corresponding measurement noise covariance.

The motion noise covariance matrix $\mathbf{Q}_k$ is set to constant ($\mathbf{Q}$) and includes the covariance of the noise affecting the velocity and yaw rate of the rovers

$$\mathbf{Q} = \begin{bmatrix} \sigma_v^2 & 0 \\ 0 & \sigma_{\dot{\psi}}^2 \end{bmatrix} . \tag{4.34}$$

Similarly the measurement noise covariances, $\mathbf{R}_k^L$ and $\mathbf{R}_k^F$, are also set to constant,

$\mathbf{R}^L$ and $\mathbf{R}^F$,

$$\mathbf{R}^L = \begin{bmatrix} \sigma_{\omega_l}^2 & 0 & 0 & 0 \\ 0 & \sigma_{\omega_r}^2 & 0 & 0 \\ 0 & 0 & \sigma_{\psi}^2 & 0 \\ 0 & 0 & 0 & \sigma_{\dot{\psi}}^2 \end{bmatrix}, \tag{4.35}$$

$$\mathbf{R}^F = \begin{bmatrix} \sigma_X^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_Y^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{\omega_l}^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{\omega_r}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{\psi}^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{\dot{\psi}}^2 \end{bmatrix}. \tag{4.36}$$

The motion noise covariance matrix $\mathbf{Q}$ was found through tuning the EKF (Appendix B.1) and the measurement noise covariance matrices ($\mathbf{R}^L$ and $\mathbf{R}^F$) was found through measuring the different sensors and finding their characteristics, see Appendix B.2.

4.2.4 Combination with the control system

The estimated state vector of the leading and following rover will be sent as an input to the LQR controller on the following rover. Figure 4.2 shows the complete control system with integrated sensor fusion for both the leading and following rover and the connection between them. The leader only performs the EKF as the control input is given, while the follower takes both rovers' state vectors as input for the LQR control. The LQR controller then calculates the control input as defined in Chapter 6 which gets sent to the follower's EKF algorithm to update the estimate.

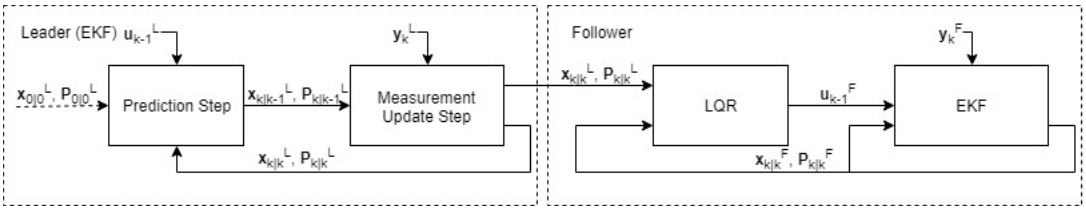


Figure 4.2: Illustration of the complete control system with both LQR controller and EKF algorithm.

4.2.5 Combination with GPS RTK estimation

The initial condition of the following rover's coordinates will be based on the relative distance between the rovers,

$$\mathbf{x}_0^F = (\hat{X}_0^F, \hat{Y}_0^F) = (r_X, r_Y) + (\hat{X}_0^L, \hat{Y}_0^L). \tag{4.37}$$

This distance is found through the use of the GPS RTK estimation. The GPS RTK estimation (Section 3.4) uses the base tower (reference receiver) coordinates to calculate the rover's coordinates. In our system, we have a moving base tower, the leading rover, and therefore unknown leader coordinates. Therefore, the GPS RTK estimation will be using the leader's estimated position to calculate the follower's position.

The carrier phase measurements from the two GPS receivers include the information needed to calculate the true distances (r_X, r_Y) between the rovers (see Figure 4.3). Instead of this distance being added to the true leader position, as in (4.37), it will be added to the estimated leader position $\mathbf{x}^L_{k|k}$. This means that the DD and TD estimation will never estimate the true follower's position but rather the follower's position translated the true distance between the rovers. Therefore the GPS RTK estimation will produce $\mathbf{x}^F_{ls} = \mathbf{x}^L_{k|k} + (r_X, r_Y)$ and not $\mathbf{x}^F$, as the translation is done on the estimated leader position $(\mathbf{x}^L_{k|k})$ and not the true leader position $(\hat{X}^L, \hat{Y}^L)$. This means that the GPS RTK measurement of the follower's position will include the same drift as the leader's position. So, although the leader's estimated position is not close to the true leader position, the relation between the true leader position and the true follower position will exist in the relation between both rovers estimated positions. In the example presented in Figure 4.3, the rovers move forward with a velocity v, which move the estimated position with a similar velocity v, which cause the following rover to move forward to keep the distance d.

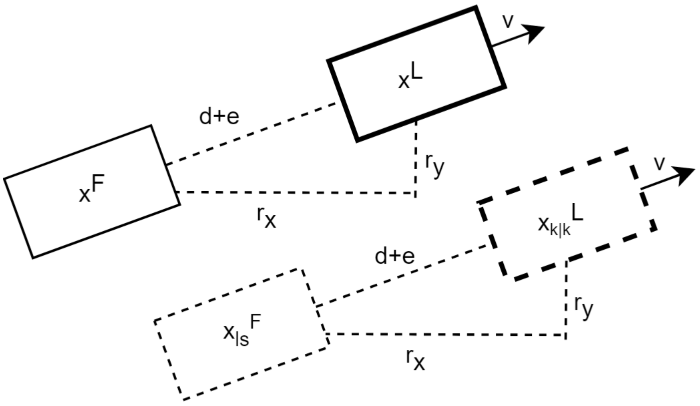


Figure 4.3: Illustration of the relation between the true distances and the estimated distances.

In the implementation of the EKF and GPS RTK estimation (DD and TD), the least squares estimation will be used as the GPS measurement in the EKF, $((\tilde{X}^F_k, \tilde{Y}^F_k) = \mathbf{x}^F_{ls})$ (see Figure 4.4). In addition the follower's prediction $\hat{\mathbf{x}}^F_{k|k-1}$ will be used as the least squares initial estimation $\mathbf{x}_{r_2}[k]_0$, and the previous estimation $\hat{\mathbf{x}}^F_{k-1|k-1}$ will be used as the previous estimation in the least squares estimation $\mathbf{x}_{r_2}[k-1]$.

As the least squares estimation is based on the ECEF frame which uses X, Y and Z, while the simulation implementation is based on just X and Y, a conversion has been implemented. This conversion just translates the X, Y coordinates by an offset $\mathbf{x}_{offset}$. This offset is based on the longitude and latitude coordinates close to our office at DevPort, which were thereafter converted to the ECEF frame through

[32]. In practice, the movement in one direction would cause a movement in all axes in the ECEF instead of just X and Y. This difference is minimal as we are at most moving a few meters, which in comparison to the overall coordinates that have a magnitude of $10^5 - 10^6$ is insignificant.

Similarly for the implementation of the carrier phase measurements, eight different satellite coordinates where found based on real-life GPS satellites visible from Gothenburg [33].

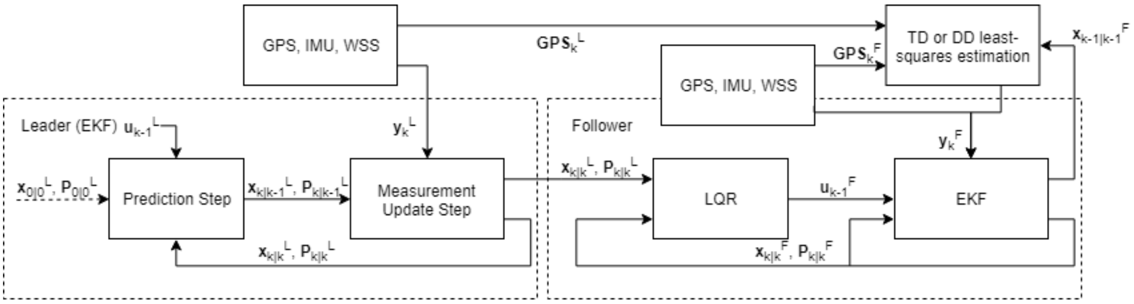


Figure 4.4: Illustration of the complete control system with both LQR controller and EKF algorithm, including the measurement origins. With the used GPS measurement in the following rover is based on the least squares estimation.

5

Path Planning

This chapter introduces the path planning algorithm which is used to create the trajectory for the following rover. The algorithm is mainly based on the concept of using breadcrumbs, explained in Section 5.1. Thereafter the tracking of the lead rover and the breadcrumb path is defined. This is firstly defined based on shortest path (Section 5.2) and later extended with Bézier curves to better track large distance differences (Section 5.3).

5.1 Breadcrumbs

During the operation of platooning, the lead vehicle is manually controlled while the follower is attempting to mimic the exact route of the lead vehicle. This is accomplished by using a concept called breadcrumb path [34]. The lead vehicle's estimated position is stored in an array with a set frequency and the follower has to drive towards these position in chronological order. Each breadcrumb also has an orientation attached towards it in order to determine a method to drive towards the breadcrumb. When a follower is within a distance D from a breadcrumb, it will swap focus towards the next breadcrumb and drive towards it. The distance D is a tuning parameter for the path planner. This concept is illustrated in Figure 5.1.

A path planning algorithm is used which makes use of straight lines and curves in order to reach each individual breadcrumb. The straight lines are created by using the shortest distance (Pythagoras' theorem [35]) between two points and the curves generated are Bézier curves [9]. Both of them are used in conjunction in order to achieve a smooth path planner which imitates the movements of the lead vehicle. The shortest distance is used in situations where the vehicle does not have to change its orientation drastically in order to match its breadcrumb's orientation. A solution using Bézier curves is used in situations where the vehicle has to change its orientation in order to match the orientation of the breadcrumb. The idea is that once the follower has caught up to the lead rover it transitions into using the shortest distance algorithm. Hence, the Bézier curves are only used in the beginning of the simulations if the coordinates and orientations differs.

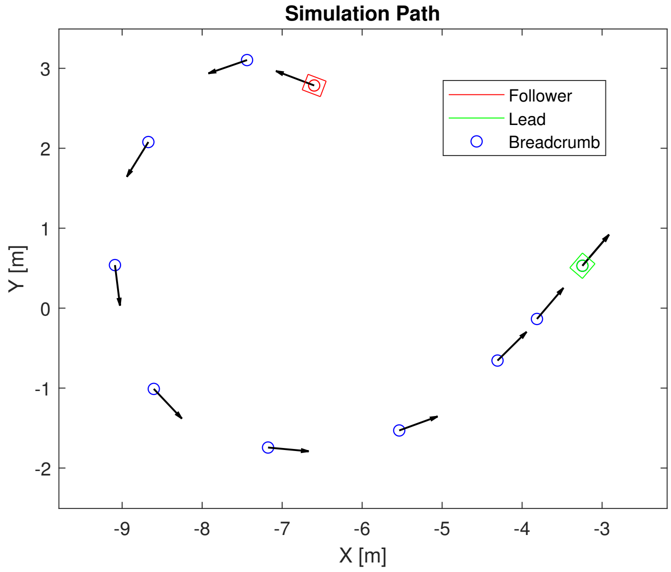


Figure 5.1: A breadcrumb path dropped from the lead vehicle where the follower has to pick them up in chronological order.

5.2 Shortest Distance

The shortest distance is based on Pythagoras' theorem in order to generate a linear path between two points. Assume a shortest path is generated between two points P_1 and P_2

$$P_1 : (x_1, y_1), \tag{5.1}$$

$$P_2 : (x_2, y_2). \tag{5.2}$$

The distance d between the points P_1 and P_2 is obtained by

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}. \tag{5.3}$$

The shortest path between two points can be seen in Figure 5.2. This path planner does not need to be more complex given that the vehicle maintains the same orientation in order to reach its destination and match its breadcrumb's orientation.

5.3 Bézier Curve

A Bézier curve is a curve based on chosen points in a plane. The number of points may be chosen arbitrarily but a four points curve P_0, P_1, P_2 and P_3 will be used in this thesis' path planning algorithm. Two points P_0 and P_3 specify the starting location and the destination respectively while the other two points P_1 and P_2 specify the orientations of the start and end location respectively. The generated Bézier curve based on the four points can be observed in Figure 5.3.

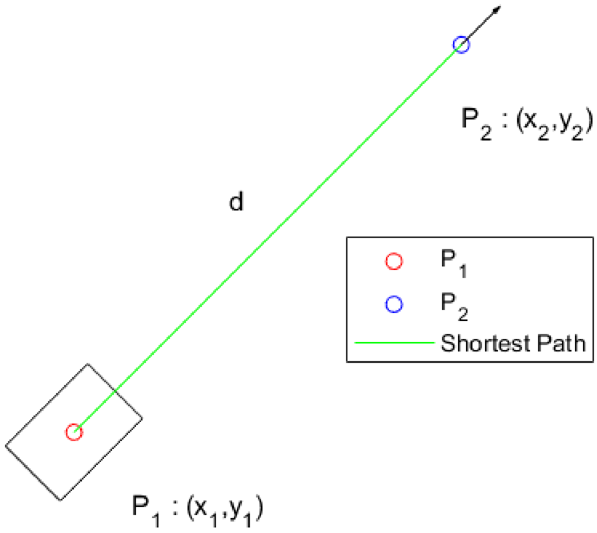


Figure 5.2: A shortest path algorithm used to reach a breadcrumb.

Point P_0 and P_3 are chosen based on starting and end location. Point P_1 and P_2 are chosen at a distance a from point P_0 and P_3 heading orientation. The relation between the points can be described according to the following equations

$$P_i : (x_i, y_i), \tag{5.4}$$

$$x_1 = x_0 + a\cos\psi_1 \qquad y_1 = y_0 + a\sin\psi_1, \tag{5.5}$$

$$x_2 = x_3 - a\cos\psi_2 \qquad y_2 = y_3 - a\cos\psi_2, \tag{5.6}$$

where ψ_1 and ψ_2 are the orientation of the follower and the lead vehicle respectively. With all four points P_0, P_1, P_2 and P_3 obtained explicitly, a Bézier curve can be generated by

$$x(t) = \sum_{i=0}^{3} b_{i,3}(t) \cdot x_i, \quad t \in [0,1], \tag{5.7}$$

$$y(t) = \sum_{i=0}^{3} b_{i,3}(t) \cdot y_i, \quad t \in [0,1], \tag{5.8}$$

where $b_{i,3}$ is given by

$$b_{i,n}(t) = \sum_{i=0}^{n} \binom{n}{i} (1-t)^{n-i} t^i. \tag{5.9}$$

The variables x_i and y_i represents the coordinates of the points P_i and t is a continuous parameter which generates the path when t goes from 0 to 1.

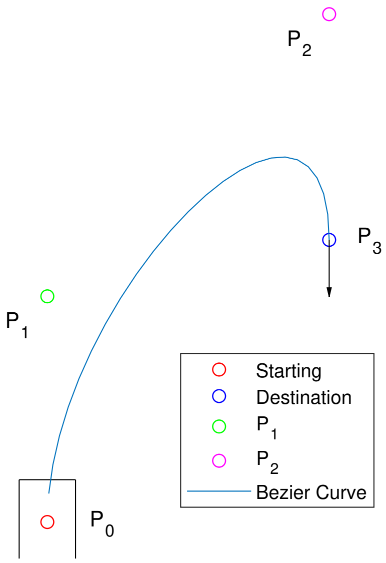


Figure 5.3: A path generated by Bezier curve.

6
Control system for platooning

There are several different controllers that may be used in order to control the rovers in this thesis. One being a Linear-Quadratic Regulator (LQR) which is a model based controller. The tuning process of the LQR is simple and it can be specified which states are of higher priority to control. An LQR is only viable if the vehicle model fulfills certain criteria. This section investigates and proves that all criteria are met by the vehicle model and defines how the LQR is used to control the rover.

6.1 Stability of the system

It is of interest in control theory to investigate whether a system is stable or not. The possibility for a state to diverge to infinity might present a case where the system is not controllable, thus increasing the difficulty of controlling the system.

The stability condition [36] for a discrete state space representation is fulfilled when all eigenvalues λ_i of $\mathbf{A}_d$ satisfy

$$||\lambda_i|| \leq 1,\ \forall\lambda_i,\ i = 1, 2, ..., n\ \ , \tag{6.1}$$

where n is the dimension of $\mathbf{A}_d$. The matrix $\mathbf{A}_d$, which was derived in (2.21), has the following eigenvalues obtained by

$$det(\mathbf{I}^{n\times n}\lambda - \mathbf{A}_d) = 0 \implies det\begin{bmatrix}\lambda-1 & 0\\ 0 & \lambda-1\end{bmatrix} = 0 \implies \tag{6.2}$$
$$\implies (\lambda-1)^2 = 0 \implies \lambda_1 = 1,\ \lambda_2 = 1,$$

Since all eigenvalues fulfill the condition (6.1), it implies that the system is stable [36].

6.2 Reachability and Controllability of the system

The vehicle model needs to fulfill the reachability and controllability conditions in order to use an LQR as a controller. Given a system modeled by

$$\boldsymbol{\chi}_k = \mathbf{A}\boldsymbol{\chi}_{k-1} + \mathbf{B}\mathbf{u}_{k-1}, \quad \boldsymbol{\chi}_0 = \begin{bmatrix}0 & 0\end{bmatrix}^T, \tag{6.3}$$

the system (6.3) is reachable if any arbitrary state $\boldsymbol{\chi}_N \in \mathbb{R}^2$ can be reached from $\boldsymbol{\chi}_0 = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$ by a discrete set of inputs $\{\mathbf{u}_i : i = 0, 1, ...N - 1\}$ where N is an integer [36].

A system

$$\boldsymbol{\chi}_k = \mathbf{A}\boldsymbol{\chi}_{k-1} + \mathbf{B}\mathbf{u}_{k-1}, \quad \boldsymbol{\chi}_0 \in \mathbb{R}^2 \tag{6.4}$$

is controllable if for any arbitrary initial state $\boldsymbol{\chi}_0 \in \mathbb{R}^2$ the system can reach $\boldsymbol{\chi}_c = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$ by a discrete set of inputs $\{\mathbf{u}_i : i = 0, 1, ...c - 1\}$ where c is an integer [36]. If both of these conditions are fulfilled, then $\boldsymbol{\chi}$ is able to reach all values in $\mathbb{R}^4$ given the right inputs.

The reachability condition is fulfilled if

$$\text{rank } \mathcal{R}_2 = 2, \tag{6.5}$$

where

$$\mathcal{R}_2 = \begin{bmatrix} \mathbf{B} & \mathbf{AB} \end{bmatrix}. \tag{6.6}$$

Inserting the discrete matrices $\mathbf{A}_d$ and $\mathbf{B}_d$ from (2.21) in (6.6) gives the following expression

$$\mathcal{R}_2 = t_s \begin{bmatrix} \frac{r}{2} & \frac{r}{2} & \frac{r}{2} & \frac{r}{2} \\ \frac{-r}{2L} & \frac{r}{2L} & \frac{-r}{2L} & \frac{r}{2L} \end{bmatrix}, \tag{6.7}$$

The matrix $\mathcal{R}_2$ obtained in (6.7) is of rank two, which implies that the system is reachable [36]. Since reachability implies controllability, all the requirements for using an LQR are met [36].

6.3 Linear-Quadratic Regulator

A Linear-quadratic controller uses feedback based on the current states to give the input $\boldsymbol{u}_{k-1}$ according to

$$\mathbf{u}_{k-1} = -\mathbf{K}_d\boldsymbol{\chi}_{k-1}, \tag{6.8}$$

where $\mathbf{K}_d$ is a matrix which is dependant on the model matrices $\mathbf{A}_d$ and $\mathbf{B}_d$ and the penalty matrices $\mathbf{Q}_{LQR}$ and $\mathbf{R}_{LQR}$ which are tuning parameters. The general idea to solve for $\mathbf{K}_d$ is to minimize the expression

$$V = \int_0^\infty (\boldsymbol{\chi}^T \mathbf{Q}_{LQR} \boldsymbol{\chi} + \boldsymbol{u}^T \mathbf{R}_{LQR} \boldsymbol{u}) dt. \tag{6.9}$$

The matrix $\mathbf{K}_d$ can be obtained explicitly by solving the Discrete-time Algebraic Riccati Equation (DARE) which is explained in more detail in [37] .

Substituting expression (6.8) for $\mathbf{u}_{k-1}$ in the state space representation gives the expression

$$\boldsymbol{\chi}_k = (\mathbf{A}_d - \mathbf{B}_d\mathbf{K}_d)\boldsymbol{\chi}_{k-1}. \tag{6.10}$$

The input $\boldsymbol{u}_{k-1}$ given by (6.8) will normally result in a decreased value of the states until it reaches zero. Therefore, what is of interest is to have the errors as states so that the controller reduces the error to zero given enough time. The error states are given by the following expression

$$\boldsymbol{\chi}_e = \begin{bmatrix} x_e \\ \psi_e \end{bmatrix}. \tag{6.11}$$

The error ψ_e is defined as the shortest angle between the current heading of the follower and the desired heading towards the destination, see Figure 6.1. The error x_e is a self defined distance given by

$$x_e = d\frac{(\pi - |\psi_e|)}{\pi} \tag{6.12}$$

where d is the shortest distance between the follower and the desired destination. It is desired that x_e assumes a low value when turned away from the destination. This is to prioritize the correction of the orientation before the distance. These error states exist so that the follower will have a similar driving pattern as the lead vehicle as the follower is approaching the breadcrumb as described in Chapter 5.

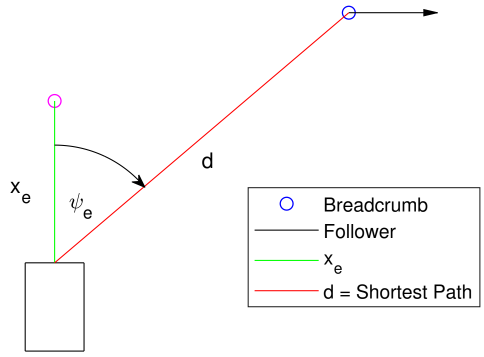


Figure 6.1: Schematic illustrating the error states.

7
Results

In this chapter, the results of the thesis are presented. Firstly, the final tuning of the LQR is presented in Section 7.1 which will be used in the platooning simulation. The main result of the LQR is the final tuning matrices $\boldsymbol{Q}_{LQR}$ and $\boldsymbol{R}_{LQR}$ defined in Section 6.3 as well as to give a baseline of the platoon's performance with a completely known system. Thereafter in Section 7.2, the platooning result is presented using measurements based on the GPS sensor in the hardware platform (Appendix A.4). The result show how well the system can work in theory if the ideal accuracy of the GPS RTK measurement is achieved.

The Sections 7.3, 7.4 and 7.5 introduces the results of the simulated least squares TD and DD estimations followed by a full platoon simulation using the developed TD estimation instead of the ideal measurement. The main result from the TD and DD estimations (Section 7.3 and 7.4) are the accuracy of the estimated receiver coordinates as well as the accuracy of the estimated ambiguities. Other things of importance are to evaluate whether the leader's estimated position drift would follow into the GPS RTK estimation, as presented in Section 4.2.5, and how the fixation of the phase integer ambiguities affect the estimation. Finally the result of the complete platoon simulation using the developed TD estimation is presented which shows how well the platoon can function using the proposed solutions of the thesis.

7.1 Performance of the LQR

The simulation presented in Figure 7.1 illustrates platooning with the tuned LQR controller. The EKF was not included in this simulation. The ground truth states were assumed to be known for both of the rovers in order to evaluate the performance of the controller. The performance of the controller is measured in two ways, the follower's ability to stay on the leader's path and its ability to maintain a distance of 20 cm from the leader (45 cm center-to-center). The controller was tuned until a reasonably good performance was achieved which can be observed in Figure 7.2. The finalized tuning parameters of the controller is given by the matrices

$$\mathbf{Q}_{LQR} = \begin{bmatrix} 5000 & 0 \\ 0 & 1000 \end{bmatrix}, \tag{7.1}$$

$$\mathbf{R}_{LQR} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{7.2}$$

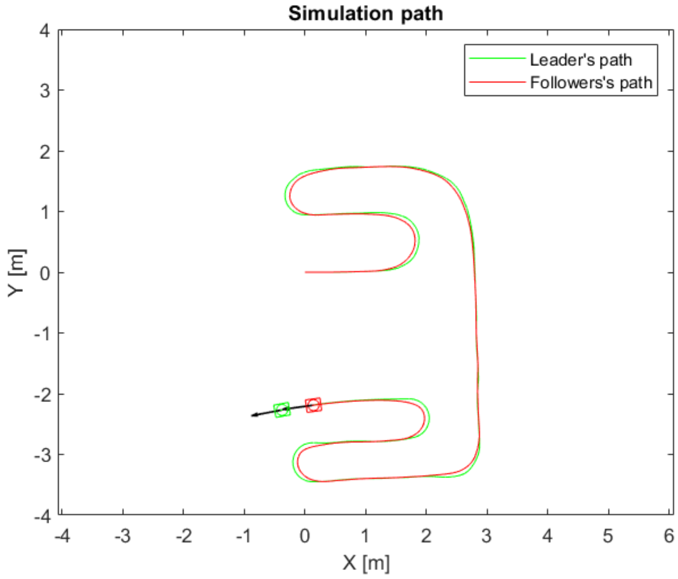


Figure 7.1: Platooning illustrating the leader's and the follower's path.

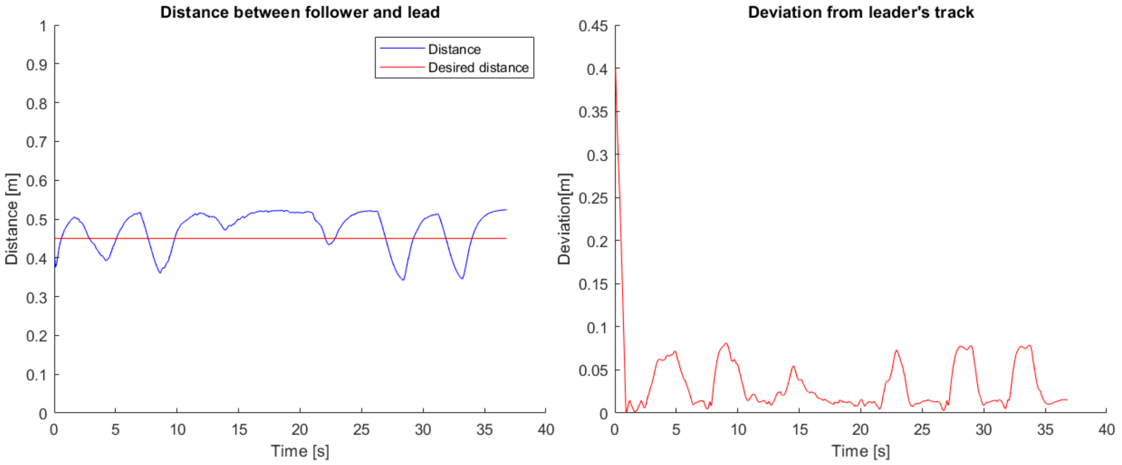


Figure 7.2: The left plot illustrates the distance between the lead and the following rovers during the platooning. The right plot illustrates the follower's deviation from the leader's path.

These finalized tuning parameters are used in the results presented in Section 7.2 and Section 7.5.

7.2 Platooning with ideal GPS RTK

Using the precision given by the GPS receiver sensor (Appendix A.4) for the GPS measurements, the platoon is successful, seen in Figure 7.3. The errors of the simulation can be seen in Figure 7.4 which shows a deviation from the track and a deviation from the desired distance of around a decimeter. This result shows that the platoon is capable of maintaining the desired distance, while staying close to the leaders track, with a high accuracy.

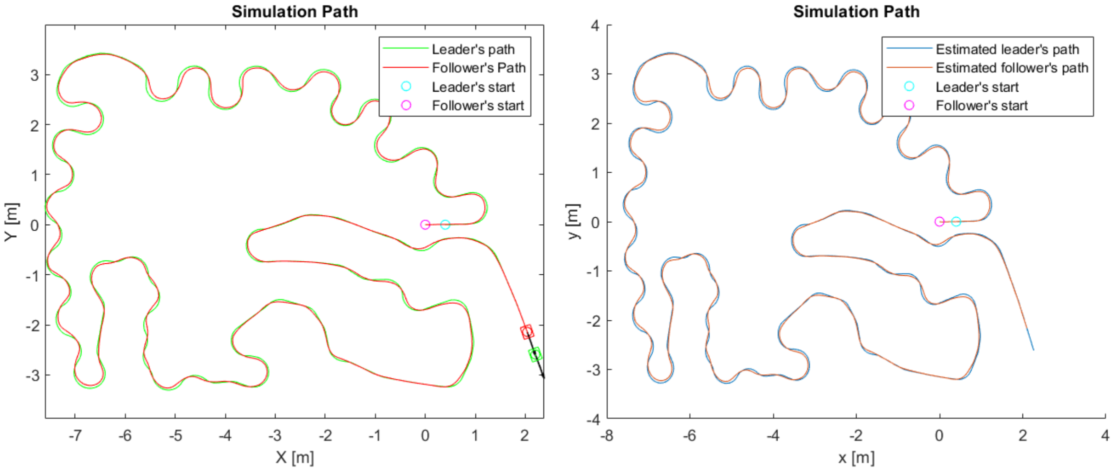


Figure 7.3: Simulated platooning path using ideal GPS RTK measurements. The left simulation shows the ground truth position while the right shows the estimated position.

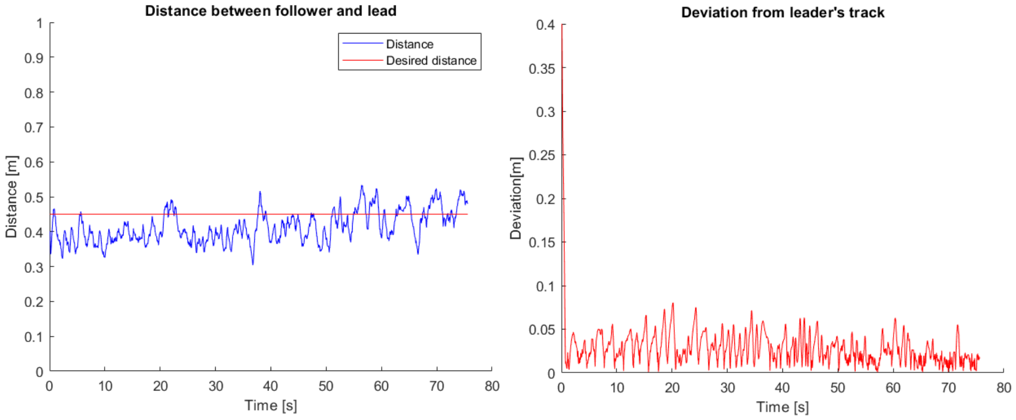


Figure 7.4: The distance from the leader and the deviation from path using ideal GPS RTK measurements.

7.3 GPS trilateration using TD estimation

The TD estimation, using four satellites and two time epochs, has seven known variables (see Section 4.2.5) that are required to perform the estimation: The satellite positions $\mathbf{x}^{s_1}[k]$, $\mathbf{x}^{s_2}[k]$, $\mathbf{x}^{s_3}[k]$, $\mathbf{x}^{s_4}[k]$, the leader's position $\mathbf{x}_{r_1}[k]$, the previous estimation $\mathbf{x}_{r_2}[k-1]$, and the initial estimation $\mathbf{x}_{r_2}[k]_0$. The satellites' positions are assumed to be known as mentioned in Section 3.3. Similarly the leader's position ($\mathbf{x}^L$) should be irrelevant to the performance of the estimation as the leader's position is only used to calculate the translation (r_X, r_Y, r_Z) that the estimated follower position ($\mathbf{x}_{ls}$) is affected by (see Section 4.2.5). The following baseline assumptions are made during the investigation of the TD estimation result. The leader's position used in the estimation will be the true position as the value should be irrelevant to the result. The initial estimation ($\mathbf{x}_{r_2}[k]_0$) is set to the follower's true position ($\mathbf{x}^F$) with an added noise $\mathbf{r}_{XYZ}$. Similarly, the previous estimation ($\mathbf{x}_{r_2}[k-1]$) is set to the true previous position ($\mathbf{x}[k-1]^F$) of the follower with an additional noise $\mathbf{r}_{XYZ}$, with $\mathbf{r}_{XYZ} = 0.1$.

For the baseline assumptions, the standard deviation of the least squares estimation after 1000 samples (see Figure 7.5 for the X-axis result) becomes

$$\begin{bmatrix}\sigma_0^X\\ \sigma_0^Y\\ \sigma_0^Z\end{bmatrix} = \begin{bmatrix}0.3184\\ 0.3162\\ 0.3139\end{bmatrix}, \tag{7.3}$$

see Appendix C.1 and C.2 for Y and Z.

Increasing the noise $\mathbf{r}_{XYZ}$ for the previous estimation to the obtained standard deviation ($\mathbf{r}_{XYZ} = 0.316$) increases the inaccuracy even further to

$$\begin{bmatrix}\sigma_1^X\\ \sigma_1^Y\\ \sigma_1^Z\end{bmatrix} = \begin{bmatrix}0.5639\\ 0.5633\\ 0.5642\end{bmatrix}, \tag{7.4}$$

see Appendix C.3, C.4 and C.5.

If we instead increase the noise $\mathbf{r}_{XYZ}$ for the initial estimation the result becomes similar to the first run with

$$\begin{bmatrix}\sigma_0^X\\ \sigma_0^Y\\ \sigma_0^Z\end{bmatrix} = \begin{bmatrix}0.3161\\ 0.3000\\ 0.3158\end{bmatrix}, \tag{7.5}$$

see Appendix C.6, C.7 and C.8.

Increasing the noise $\mathbf{r}_{XYZ}$ for both the initial estimations and the previous estimation gives a similar result to when only increasing the noise for the previous estimation,

$$\begin{bmatrix}\sigma_1^X\\ \sigma_1^Y\\ \sigma_1^Z\end{bmatrix} = \begin{bmatrix}0.5536\\ 0.5646\\ 0.5533\end{bmatrix}, \tag{7.6}$$

see Appendix C.9, C.10 and C.11.

From (7.3)-(7.6) it is possible to discern that the estimation accuracy is dependent on the accuracy of the previous estimation $\mathbf{x}_{r_2}[k-1]$, and not on the accuracy of the initial estimation $\mathbf{x}_{r_2}[k]_0$.

Increasing the noise for 100 iterations and for each iteration using the found deviation of the last 1000 samples ($\mathbf{r}_{XYZ} = \sigma_k^{X,Y,Z}$) when initializing the initial and previous estimation shows (Figure 7.6) that the accuracy of the least squares estimation converges with time to an average standard deviation

$$\begin{bmatrix}\sigma_{100}^X\\ \sigma_{100}^Y\\ \sigma_{100}^Z\end{bmatrix} = \begin{bmatrix}1.0232\\ 1.0076\\ 1.0235\end{bmatrix}. \tag{7.7}$$

Similar result is obtained if the noise $\mathbf{r}_{XYZ}$ remains constant for the initial estimation and the increasing inaccuracy is only applied to the previous estimation.

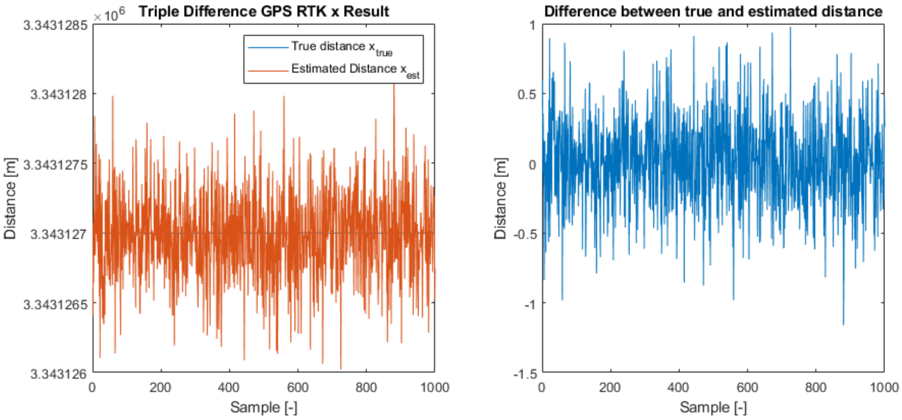


Figure 7.5: TD GPS RTK measurement samples, X axis, assuming known leader position and adding noise onto the previous and initial estimation.

As mentioned previously and in Section 4.2.5, the simulated LQR controller with EKF and GPS RTK is based on the fact that the measured GPS signal is translated the same distance as the lead rover's drift. Instead of using the true leader position $\mathbf{x}^L[k]$, as in the baseline, we use the estimated leader position $\hat{\mathbf{x}}^L[k] = \mathbf{x}^L[k] + \mathbf{b}$, as well as an assumed equally drifted previous receiver location $\mathbf{x}_{r_2}[k-1] = \mathbf{x}^F[k-1]+\mathbf{b}$ ($\mathbf{b} = 10$ in X-axis). This results in a similarly translated least squares estimation as seen in Figure 7.7 ($\mathbf{x}_{ls} = \mathbf{x}^F[k-1] + \mathbf{b}$).

The TD least squares estimation results in a standard deviation of

$$\begin{bmatrix}\sigma_{100}^X\\ \sigma_{100}^Y\\ \sigma_{100}^Z\end{bmatrix} = \begin{bmatrix}1.0232\\ 1.0076\\ 1.0235\end{bmatrix}, \tag{7.8}$$

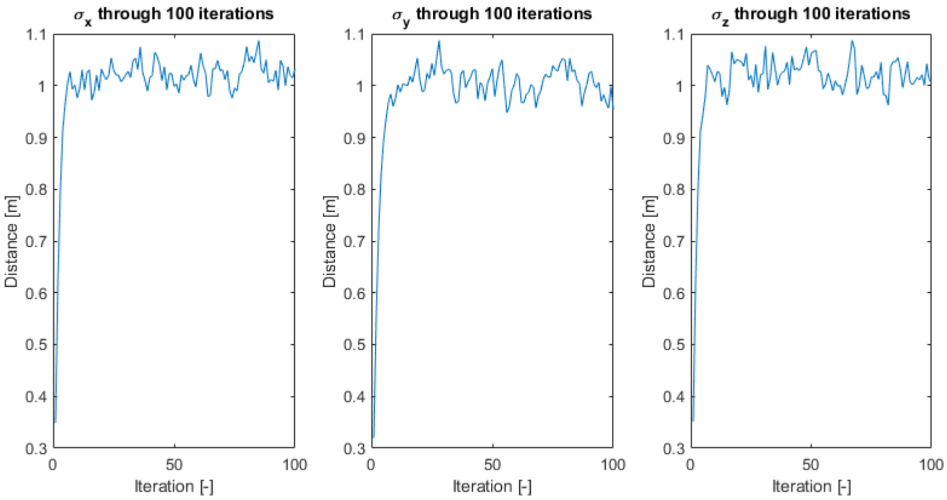


Figure 7.6: The standard deviation of X, Y, and Z axis through 100 iterations where each iteration used the previous iterations uncertainty.

based on Figure 7.6, which used four satellites and two time epochs. Increasing the number of satellites results in an slightly better overall estimation of the coordinates (see Figure 7.8). With eight satellites the result is

$$\begin{bmatrix} \sigma_{100}^{X} \\ \sigma_{100}^{Y} \\ \sigma_{100}^{Z} \end{bmatrix} = \begin{bmatrix} 0.9977 \\ 1.0063 \\ 1.0047 \end{bmatrix}. \tag{7.9}$$

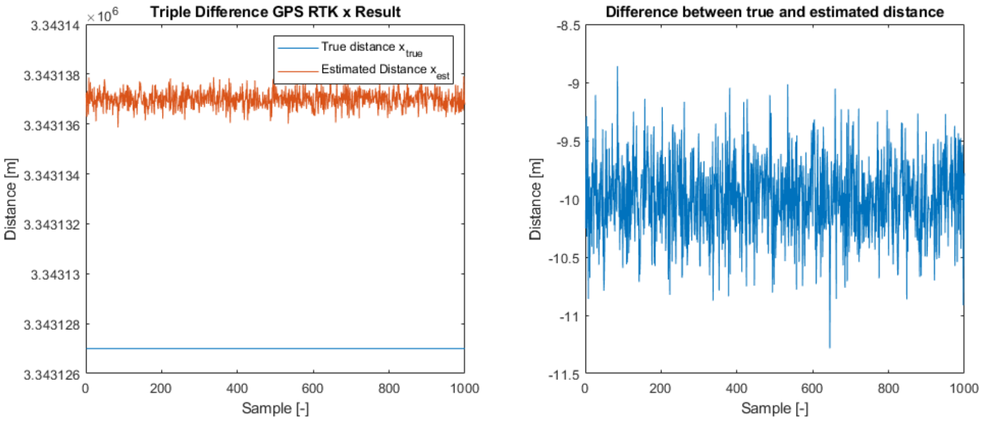


Figure 7.7: X-axis result of 1000 samples using a leader position and previous estimation translated 10 meters in X-axis.

7.4 GPS trilateration using DD estimation

The DD estimation uses the inbuilt MATLAB function *lsqlin* instead of the least square estimation 3.58. This is because pseudoinverse of the **G** matrix has rounding errors which causes the search to diverge. Therefore, a strict bound has been set on

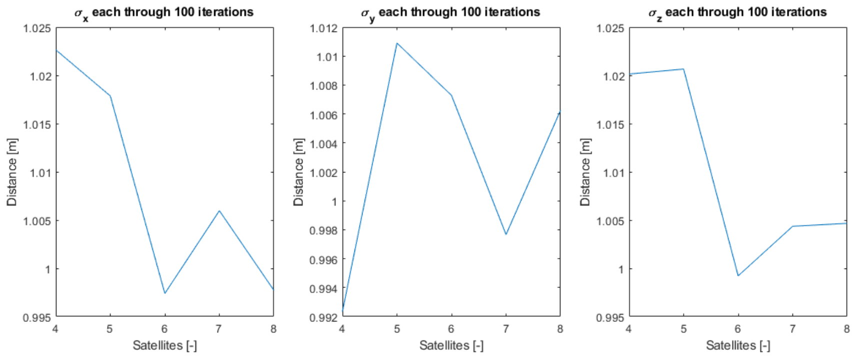


Figure 7.8: Average standard deviation calculated based on 100 iterations of 1000 samples for different amount of satellites.

the search algorithm to not allow it to diverge. This causes the result to be skewed as certain values cannot be explored.

The baseline of the DD estimation is the same as for the TD estimation with the addition of a baseline for the phase integer ambiguities, $\aleph^{s1s2}_{r1r2}$, $\aleph^{s1s3}_{r1r2}$ and $\aleph^{s1s4}_{r1r2}$. The phase integer ambiguities are estimated and therefore needs an initial estimation $\aleph^{s1s2}_{r1r20}$, $\aleph^{s1s3}_{r1r20}$, and $\aleph^{s1s2}_{r1r20}$. The baseline for the initial estimation of the ambiguities is set to the true value with an additional noise $r_\aleph = 1$.

The DD estimations baseline assumptions results in a standard deviation of the coordinates

$$\begin{bmatrix} \sigma_0^X \\ \sigma_0^Y \\ \sigma_0^Z \end{bmatrix} = \begin{bmatrix} 0.7054 \\ 0.3989 \\ 0.6465 \end{bmatrix}, \tag{7.10}$$

after a run of 1000 samples, see Figure 7.9 for X-axis (Appendix C.12 and C.13 for Y and Z). Similarly the phase integer ambiguities has an updated standard deviation

$$\begin{bmatrix} \sigma_0^{\aleph^{s1s2}_{r1r2}} \\ \sigma_0^{\aleph^{s1s3}_{r1r2}} \\ \sigma_0^{\aleph^{s1s4}_{r1r2}} \end{bmatrix} = \begin{bmatrix} 0.6557 \\ 0.8869 \\ 0.9053 \end{bmatrix}, \tag{7.11}$$

see Figure 7.10, Appendix C.14 and Appendix C.15.

In Section 3.4 it is mentioned that with a fixed integer ambiguity the accuracy of the estimation would increase as the carrier phase measurements are accurate with the exception of the ambiguities. Setting the ambiguities initial estimation noise $r_\aleph$ to zero, emulating a fixed ambiguity, gives an updated deviation for the coordinates

$$\begin{bmatrix} \sigma_0^X \\ \sigma_0^Y \\ \sigma_0^Z \end{bmatrix} = \begin{bmatrix} 0.2662 \\ 0.3052 \\ 0.2742 \end{bmatrix}, \tag{7.12}$$

see Appendix C.16, C.17 and C.18.

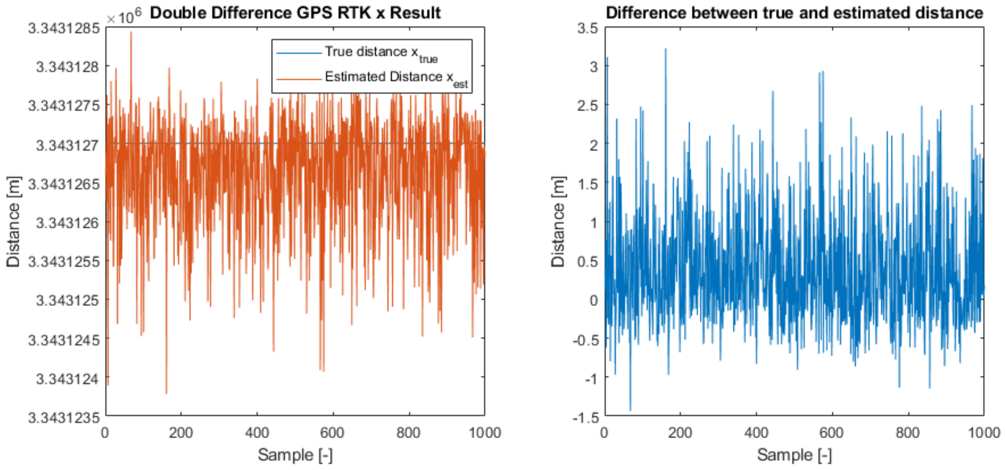


Figure 7.9: DD GPS RTK measurement samples, X axis, using baseline assumptions.

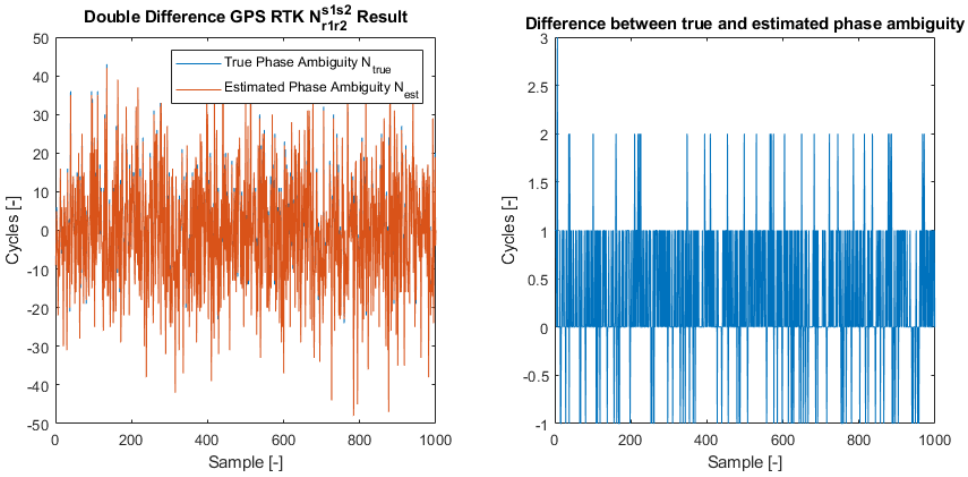


Figure 7.10: DD GPS RTK measurement samples, ambiguity $\aleph_{r1r2}^{s1s2}$, using baseline assumptions.

Increasing the noise $r_{\aleph} = 100$ to emulate the initial condition with completely unknown ambiguities increases the inaccuracy drastically,

$$\begin{bmatrix} \sigma_0^X \\ \sigma_0^Y \\ \sigma_0^Z \end{bmatrix} = \begin{bmatrix} 8.8916 \\ 2.7898 \\ 7.0257 \end{bmatrix}, \tag{7.13}$$

see Appendix C.22, C.23 and C.24.

For the ambiguities the deviations stays similar for the fixed scenario

$$\begin{bmatrix} \sigma_0^{\aleph_{r1r2}^{s1s2}} \\ \sigma_0^{\aleph_{r1r2}^{s1s3}} \\ \sigma_0^{\aleph_{r1r2}^{s1s4}} \end{bmatrix} = \begin{bmatrix} 0.0891 \\ 0.0547 \\ 0 \end{bmatrix}, \tag{7.14}$$

see Appendix C.19, C.20 and C.21.

For the unknown initial ambiguity scenario the deviation for the ambiguities improves drastically in comparison to the initial noise $r_{\aleph}$,

$$\begin{bmatrix} \sigma_0^{\aleph_{r1r2}^{s1s2}} \\ \sigma_0^{\aleph_{r1r2}^{s1s3}} \\ \sigma_0^{\aleph_{r1r2}^{s1s4}} \end{bmatrix} = \begin{bmatrix} 7.8275 \\ 8.2229 \\ 9.1117 \end{bmatrix}, \tag{7.15}$$

see Appendix C.25, C.26 and C.27.

Repeating the run with fixed integer ambiguities for 100 iterations, and each iterations having an updated noise ($\mathbf{r}_{XYZ} = \sigma_k^{X,Y,Z}$ and $r_{\aleph} = \sigma_k^{\aleph}$), results in a final deviation of

$$\begin{bmatrix} \sigma_{100}^{X} \\ \sigma_{100}^{Y} \\ \sigma_{100}^{Z} \end{bmatrix} = \begin{bmatrix} 1.1060 \\ 0.9154 \\ 1.2076 \end{bmatrix}, \tag{7.16}$$

as seen in Figure 7.11. The final deviation for the phase integer ambiguity has had a slight increase with each iteration with a final value of

$$\begin{bmatrix} \sigma_{100}^{\aleph_{r1r2}^{s1s2}} \\ \sigma_{100}^{\aleph_{r1r2}^{s1s3}} \\ \sigma_{100}^{\aleph_{r1r2}^{s1s4}} \end{bmatrix} = \begin{bmatrix} 0.8225 \\ 1.2454 \\ 0.93763 \end{bmatrix}, \tag{7.17}$$

see Figure 7.12).

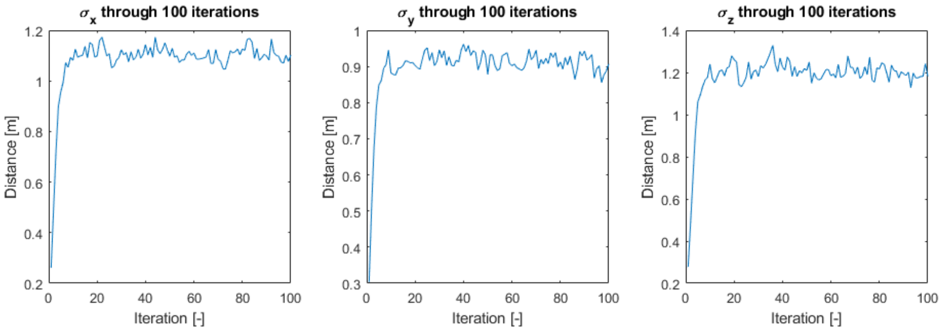


Figure 7.11: The standard deviation of X, Y, and Z axis through 100 iterations of DD estimation where each iteration used the previous iterations uncertainty.

Repeating the run with unknown initial ambiguities for 100 iterations shows a opposite trend for the coordinates and ambiguities (Figure 7.13 and 7.14). In the first iterations the initial estimation of the ambiguities will improve with each iteration and therefore produce a overall better result. Eventually the deviation of the coordinates and the ambiguities will converge towards

$$\begin{bmatrix} \sigma_{100}^{X} \\ \sigma_{100}^{Y} \\ \sigma_{100}^{Z} \end{bmatrix} = \begin{bmatrix} 1.1065 \\ 0.9167 \\ 1.2085 \end{bmatrix}, \tag{7.18}$$

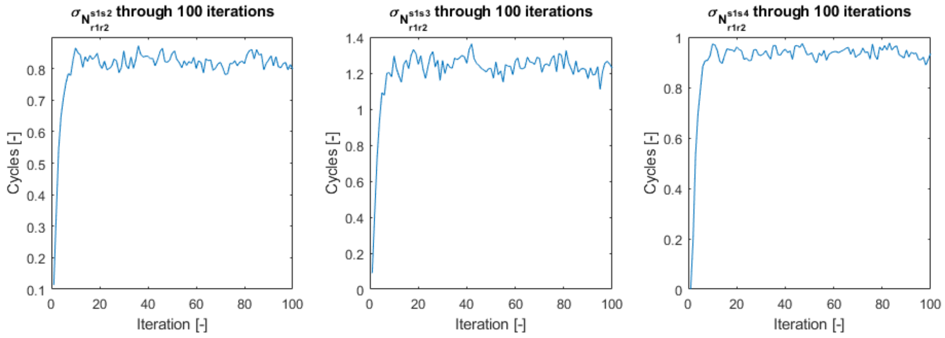


Figure 7.12: Standard deviation of ambiguity $\aleph^{s1s2}_{r1r2}$, $\aleph^{s1s3}_{r1r2}$ and $\aleph^{s1s4}_{r1r2}$, through 100 iterations of DD estimation where each iteration used the previous iterations uncertainty.

$$\begin{bmatrix} \sigma_{100}^{\aleph^{s1s2}_{r1r2}} \\ \sigma_{100}^{\aleph^{s1s3}_{r1r2}} \\ \sigma_{100}^{\aleph^{s1s4}_{r1r2}} \end{bmatrix} = \begin{bmatrix} 0.8239 \\ 1.2449 \\ 0.9390 \end{bmatrix}, \tag{7.19}$$

same as the run with fixed ambiguities.

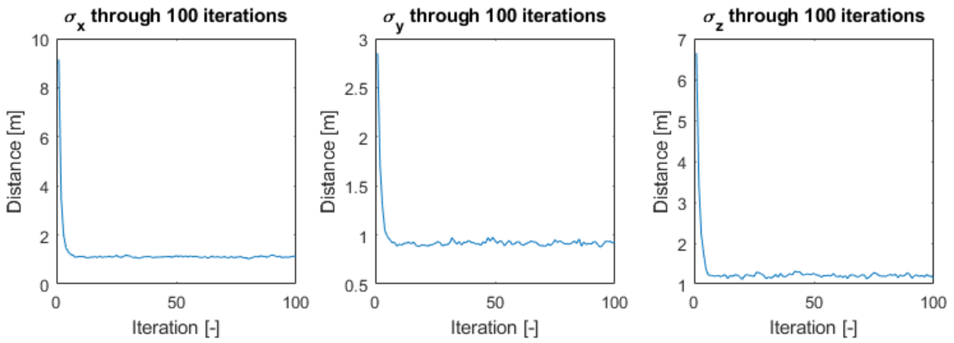


Figure 7.13: The standard deviation of X, Y, and Z axis through 100 iterations where each iteration used the previous iterations uncertainty.

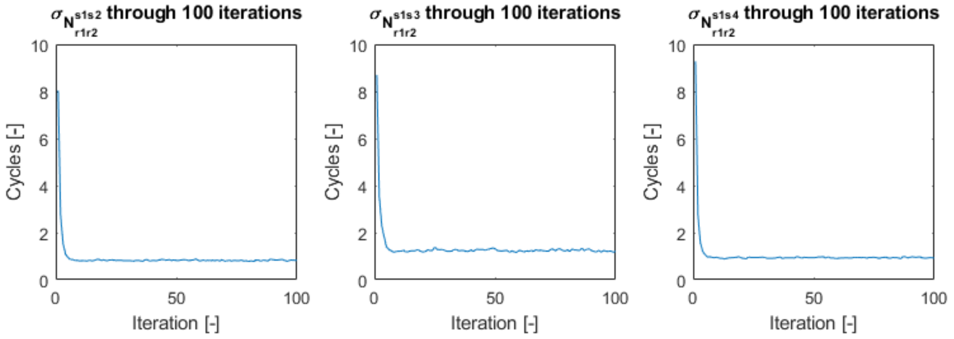


Figure 7.14: Standard deviation of ambiguity $\aleph^{s1s2}_{r1r2}$, $\aleph^{s1s3}_{r1r2}$ and $\aleph^{s1s4}_{r1r2}$, through 100 iterations where each iteration used the previous iterations uncertainty.

7.5 Platooning with TD GPS RTK

Same simulation as in Section 7.2 but instead of using the ideal GPS RTK measurements the designed least squares TD GPS RTK estimation with eight satellites is used. The initial estimation ($\mathbf{x}_{r_2}[k]_0$) is based on the predicted value in the EKF ($\hat{\mathbf{x}}^F_{k|k-1}$) and the previous estimated value ($\mathbf{x}_{r_2}[k-1]$) is based on the previous EKF estimation ($\hat{\mathbf{x}}^F_{k-1|k-1}$) as explained in Section 4.2.5. The carrier phase measurements get generated based on the true position of the follower and leader.

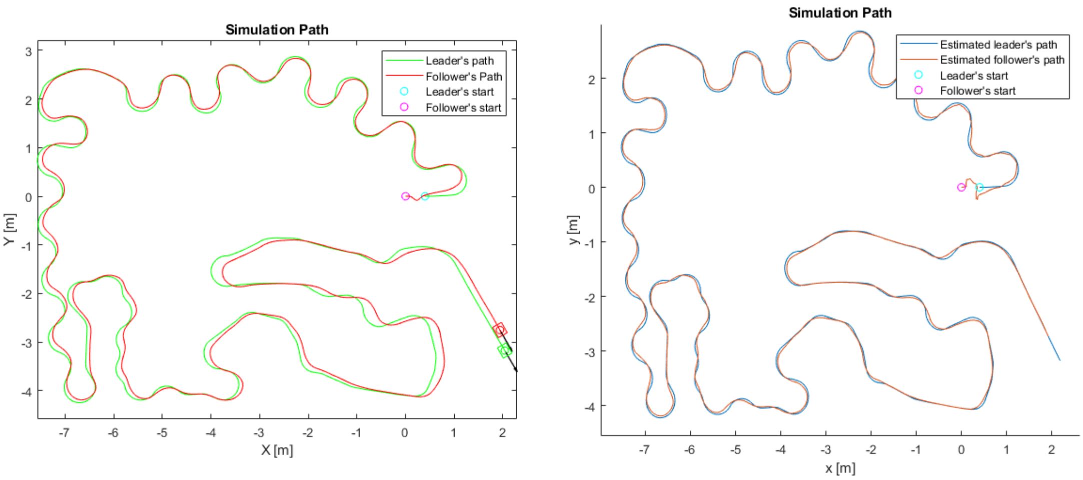


Figure 7.15: Simulated platooning path using TD GPS RTK estimation as measurement. The right simulation shows the ground truth position while the left shows the estimated position.

As observed in Section 7.3 the GPS RTK estimation is highly dependent on the previous estimation. Assuming unknown initial estimation of the follower's coordinates will give an accuracy of $\sigma \approx 1$ for all axes. As we are interested to coordinate the platoon to keep a distance of 20 cm between the two rovers, the accuracy is insufficient. Instead the initial coordinates are assumed to be known to keep the accuracy closer to the achievable accuracy. In addition to this the standard deviation of the GPS measurement in the noise matrix **R** in the EKF will be based on the standard deviation of the baseline simulation in Section 7.3 ($\sigma_X = \sigma_Y = 0.316$).

The platoon simulation result can be observed in Figure 7.15, where both the simulated and estimated path is shown. Comparing Figure 7.16 and Figure 7.4 it is possible to discern that the platoon simulation with the proposed TD estimation performs worse than the ideal system. This is unsurprising as the accuracy of the TD estimation is much worse than the hardware sensor.

In Figure 7.17 the follower's true position translated the same distance as the leader's estimation has drifted is plotted together with the GPS RTK estimation. As mentioned in Section 7.3 and seen in the result of the TD estimation (Figure 7.7) the GPS RTK estimation should estimate the translated follower position. In Figure 7.17 it

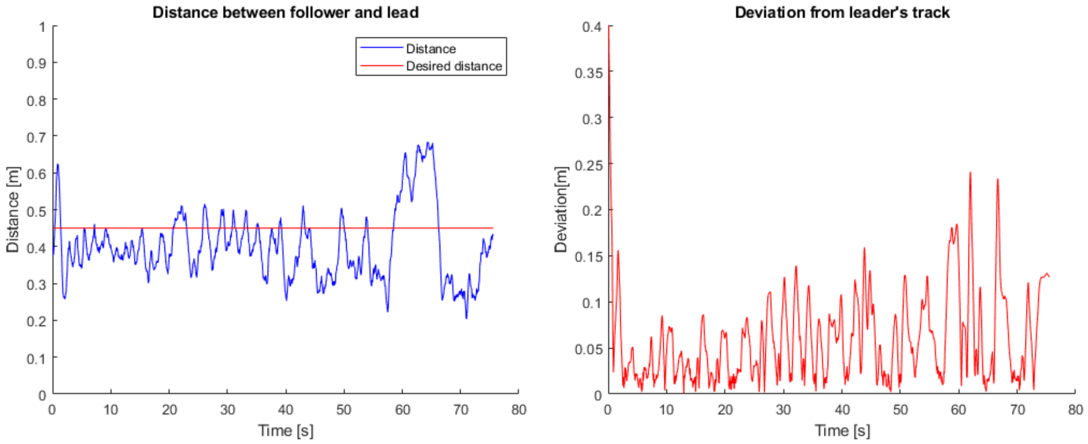


Figure 7.16: The distance from the leader and the deviation from path using the least squares TD GPS RTK estimation as GPS measurement.

is possible to see that the GPS RTK estimation does not successfully estimate the correct value as it deviates from the translated follower position, but it is not as inaccurate as the independent estimation result (7.9).

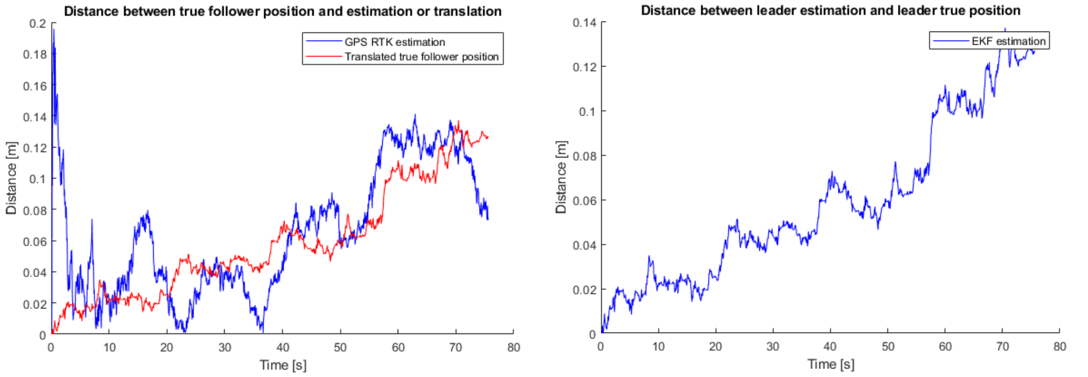


Figure 7.17: The left plot shows the distance from the GPS RTK estimation (blue) and the translated true follower position (red) to the follower's true position. The right plot shows the distance between the leader's estimatin and its true position.

8
Discussion

In this chapter, an analysis of the obtained results is made as well as a comparison between the different results. The methods are of importance when analyzing the results and will thus also be evaluated and compared with other works. Future improvements to the position estimation are suggested as well as how to improve the performance of the platooning.

8.1 Evaluation of results

Two simulations of the platooning were implemented, platooning with ideal GPS RTK, which can be observed in Figure 7.3, and platooning with GPS RTK using TD estimation, which can be observed in Figure 7.15. For the implementation of the ideal GPS RTK, the deviation from the leader's path was fluctuating between zero to ten centimeters as seen in Figure 7.4. The other metric used to evaluate the platooning is the follower's ability to maintain a distance of 45 cm from its center to the lead vehicle's center. The result is an oscillating error of 10 cm which can be seen in Figure 7.4. For the implementation of the GPS RTK using TD estimation, the deviation from the leader's path was significantly larger than using ideal GPS-RTK values which can be seen in Figure 7.16. The error is also larger for the follower's ability to maintain a desired distance from the leader.

The results shown in Figure 7.4 and Figure 7.16 can be improved by tuning the LQR. It can be tuned towards having a more precise tracking of the leader's path, but as a trade off, the ability to follow the desired distance might suffer. The result as of now uses a simplified model and it is not as representative of reality since only 2-dimensions is taken into consideration. A more realistic model that takes 3-dimensions into consideration could be derived with the help of [39]. Another improvement could be to use another tracking method, such as [40], which uses the nonlinear state in the global frame rather than the local frame. It could also be possible to improve the platooning result, and more easily allowing the addition of multiple vehicles, by using a decentralized platoon controller [41, 42].

Regarding the GPS trilateration, it is interesting to note that the least squares estimation is heavily dependent on the previous estimation. Increasing the inaccuracy of the previous estimation gives an even more inaccurate estimation (see Section 7.3). This problems shows clearly for both estimations which eventually reaches an even worse distance measurement than just using pseudorange (Figure 7.6 and 7.11). For the DD estimation, having a small deviation to the initial estimation of the ambiguities do similarly cause a worse deviation in the next iteration (see Figure 7.11). In Figure 7.13 it is clear that with the initial estimation of the ambiguities being completely unknown the accuracy improves with time until they converge to a similar result as in Figure 7.11. Having fixed phase integer ambiguities shows that the overall performance of the DD estimation is similar to the TD estimation with the DD estimation being slightly more inaccurate.

From our analysis of the code and calculations it seems that the inaccuracies of the estimations are partly because of rounding errors during the least squares estimation (3.58) and (3.52). Another reason for the inaccuracies of the estimations is that we use a simple linear least squares algorithm which does not perform an effective search. An alternative to the simple estimation method would be conditional least squares estimation as seen in [43] which also estimates the DD system. The conditional least squares estimation defines bounds of the search space which creates a more effective search. Similarly, integer least squares estimation should be used for the estimation of the phase integer ambiguities as they are integers [43]. These search issues cause greater problems for the DD estimation as a wrongful initial estimation of the phase integer ambiguities adds additional inaccuracies to the position estimation.

Based on the independent result of the GPS trilateration the least squares estimation is clearly not enough to give an accurate estimation if the system has an unknown initial position (as explained in Section 7.5). In theory, without rounding errors for the least squares estimation, it should be possible to run the estimation further to achieve a more accurate result with the downside of adding computations. Combining this with a more effective search algorithm would both improve the result and make the estimation more effective. Looking at the difference between the individual result of the TD estimation and the DD estimation (Figure 7.6 and 7.11) it is clear that the deviation of the DD estimation is slightly worse than the TD estimation if it has known initial ambiguities. This is because the DD estimation has additional unknowns that it has to estimate. It is not clear why the deviation of the DD estimation is worse in X and Z axis in comparison to Y axis when using unknown ambiguities (see (7.13)). We believe this has to do with the usage of the MATLAB function *lsqlin* instead of (3.58), which constraints the search space and causes the result for some of the axes to deviate.

The previously mentioned solution of removing the rounding errors should improve the TD estimation result but not comparatively for the DD estimation. This is because the phase integer ambiguity estimation has an inaccuracy which the implemented least squares simulation cannot solve, which is the inaccuracy in the ambiguity estimation that is caused by the correlation between the ambiguities, as all depend on the ambiguity ($\aleph^{s1}_{r1r2}$) of the first SD system (3.20). This inaccuracy is only removable by decorrelating the estimations. This is done in a method called Least squares AMBiguity Decorrelation Adjustment (LAMBDA), which enables the search for the estimations to be much more efficient while also increasing precision

[44]. The LAMBDA method introduces an ambiguity transformation Z^* that decorrelates the phase integer ambiguities and simplifies the estimation search [44]. The LAMBDA method also makes use of more complex search algorithms which are used on the transformed ambiguity problem which further improves the result. [44] shows that decorrelating the ambiguities improves the result and efficiency in the estimation, but it merely states which qualities the ambiguity transformation needs, not the exact method to use. There are different methods that can be used to construct the ambiguity transformation, three such methods are the integer Gaussian decorrelation [45], inverse integer Cholesky decorrelation [46] and Lenstra-Lenstra-Locacs decorrelation [47]. In [48] it is shown that all three methods are usable for the ambiguity transformation but that both the inverse integer Cholesky decorrelation and the Lenstra-Lenstra-Lovacs decorrelation performs better than the integer Gaussian decorrelation.

Moving from the independent trilateration result and looking at the complete platoon simulation using the TD estimation it is clear that the estimation has much less variance than previously. Firstly it is dependent on the fact that the system starts with known initial coordinates. Secondly, for following epochs the difference between the wanted estimation (4.37) and the prediction $\hat{\mathbf{x}}^F_{k|k-1}$ is much smaller than the difference between the randomized initial estimation $(\mathbf{x}_r[k]_0)$ and the true position $(\mathbf{x}^F)$. Another factor is the improved accuracy of the previous estimation $\hat{\mathbf{x}}_{k-1|k-1}$ in comparison to randomizing the previous estimation based on the noise $\mathbf{r}_{XYZ}$. These three factors are the main reason for the TD solution to have a much better result than in the independent simulations. Another important factor is that the vehicle model is not representative of the hardware platform. It has not been possible during this project to measure the performance of just the vehicle controller on the hardware platform to use as a reference in the simulations. Instead we have assigned random noise to the different states which might be less than in reality. Having a less noisy model means that the EKF can trust the model more (lower values in $\mathbf{Q}$) and will therefore not be as affected by bad sensor data. The reason why the TD estimation was used instead of the DD estimation is that the TD estimation has a better overall result and that the simulations include no cycle slips which means that the TD estimation is applicable.

The hardware platform developed in this thesis has been tested with the proposed solutions without GPS receiver. It has been found that the platform is capable of using the developed methods without issue. These tests are not presented in the thesis as we did not prepare any way of measuring the rovers ground truth. This means that we are incapable of evaluating the results beyond observing that the methods are executable. Our recommendations for the evaluation of the hardware platform would be to add an identification marker (e.g. QR-code) on each rover, and film the rovers during a run. If the camera is in a known stationary position it should be possible to calculate the rovers ground truth position and thereafter evaluate the performance.

8.2 Conclusion

Based on the obtained results, it is possible to have a working platoon with 'blind' vehicles given that the relative position measurement is accurate. The complete simulation result has been made with a simplified vehicle model and controller, and shows that the following vehicle will not deviate more than a few decimeters from the desired distance. This result can be improved, and made more realistic, by implementing a more complex vehicle model and controller.

Our simulations of GPS RTK shows that an accurate position estimation can be achieved but even this estimation drifts given indefinite simulation time. To improve the GPS RTK estimation, mainly two approaches should be taken. Firstly, the simple least squares estimation should be disregarded and a more complex estimation method should be used that encapsulates the known constraints and bounds in the estimation, such as conditional least squares estimation. Secondly, the phase integer ambiguities should be decorrelated with an ambiguity transformation as with the usage of the LAMBDA method, to further improve the result.

For the implementation of the method in practice the TD estimation can only be used to check when a cycle slip is detected as it assumes no slip can occur. This means that when the slip occurs the fixed phase integer ambiguity estimation is no longer applicable. The DD estimation can be implemented in practice but needs a more complex estimation solution than just a simple least squares estimation as the DD estimation with unknown initial ambiguities are highly inaccurate. Although not tested in practice, it is theoretically possible to broadcast the needed carrier phase measurement and satellite navigation messages between two unrelated vehicles to calculate an accurate relative position.

For future work of the GPS RTK estimation, our recommendation is to either focus on implementing a more complex estimation method together with the ambiguity transformation to decorrelate the phase integer ambiguities or implement the found solutions in practice to observe the differences between theory and reality. Besides evaluating the overall result it would be of interest to evaluate the modularity of the solution between several vehicles in the platoon, or other vehicle coordination scenarios.

Another area that we recommend investigating is the usage of a decentralized platoon controller. As some information is already shared with the GPS RTK estimation, it might be possible to use a decentralized platoon controller without much additional information. This could be used to both more easily model the system for multiple vehicles and improve the platoon accuracy.

It is also of interest, as the result of the combined system with the TD estimation and EKF shows a remarkable improvement in comparison to the stand-alone estimation results, to evaluate how a complete integration of the GPS RTK estimation into the EKF would change the result (similarly done in [23]). For DD estimation one could add the ambiguities to the motion model that gets estimated through the filter instead of through the least squares estimation.

To fully evaluate whether the proposed solutions in this thesis could be implemented in practice, requires a more in-depth study of the GPS RTK solution, the vehicle model, and the controller. Specifically focusing on making the system representative of reality.

Bibliography

[1] Grand View Research. 2018. *Self driving Cars and Trucks Market Size, Share & Trends Analysis Report By Application (Transportation, Defense), By Region (NA, Europe, APAC, South America, MEA), And Segment Forecasts, 2020 - 2030.*

[2] AB Volvo. 2017. *Volvo pioneers autonomous, self-driving refuse truck in the urban environment.*
`https://www.volvogroup.com/en-en/news/2017/may/news-2561936.html`
Date accessed 2019-01-30

[3] Alam A A, Gattami A and Johansson K H. 2010 *An experimental study on the fuel reduction potential of heavy duty vehicle platooning.* 13th International IEEE Conference on Intelligent Transportation Systems, Funchal, 2010, pp. 306-311.

[4] Gheyssens T. (2016). *Aerodynamic analysis of a platoon of bluff bodies subjected to cross wind.* Aerospace Engineering, Delft University of Technology .

[5] Janssen G R, Zwijnenberg J, Blankers I J, and de Kruijff J S. 2015. *Truck platooning : Driving the future of transportation.* TNO - innovation for life.

[6] Tsugawa S, Jeschke S and Shladover S E. 2016. *A Review of Truck Platooning Projects for Energy Savings.* IEEE Transactions on Intelligent Vehicles, vol. 1, no. 1, pp. 68-77.

[7] van de Hoef S, Johansson K H, and Dimarogonas D V. 2015. *Truck platooning: Driving the future of transportation.* 2015 American Control Conference (ACC), Chicago, IL, 2015, pp. 3740-3745

[8] Kamali M, Dennis L A, McAree O, Fisher M, and Veres S M. *Formal verification of autonomous vehicle platooning.* Science of Computer Programming Volume 148, 15 November 2017, Pages 88-106

[9] Helmut P, Asperl A, Hofer M, Kilian A. 2007. *Architectural Geometry.* Bentley Institute Press.

[10] Wu X, Xu M, Wang L. 2013. *Differential Speed Steering Control for Four-Wheel Independent Driving Electric Vehicle.* International Journal of Materials, Mechanics and Manufacturing, Vol. 1, No. 4

[11] Elshazly O, Abo-Ismail A, Abbas H, Zyada Z. 2014. *Skid Steering Mobile Robot Modeling and Control.* 2014 UKACC International Conference on Control (CONTROL).

[12] Kozłowski K, Pazderski D. 2004. *Modeling and control of a 4-wheel skid-steering mobile robot.* Int. J. Appl. Math. Comput. Sci., 2004, Vol. 14, No. 4, 477–496.

[13] Sanz J, J.M, Zornoza J, Hernández-Pajares M. 2011. *GNSS DATA PROCESSING Volume I: Fundamentals and Algorithms.* ESA Communications.

[14] Ciuban S, Krainski M. 2019. *GNSS Compare Documentation*
`https://media.readthedocs.org/pdf/gnss-compare/latest/gnss-compare.pdf`
Date accessed 2019-02-26

[15] Sickle J. 2018.*The Ionospheric Effect.*
`https://www.e-education.psu.edu/geog862/node/1715`
Date accessed 2019-05-07

[16] Sickle J. 2018.*The Tropospheric Effect, dtrop.*
`https://www.e-education.psu.edu/geog862/node/1719`
Date accessed 2019-05-07

[17] Langley R. 1999 .*Dilution of Precision.*
`http://www2.unb.ca/gge/Resources/gpsworld.may99.pdf`
Date accessed 2019-06-03

[18] Xoneca. 2013. *Geometric Dilution Of Precision.*
`https://upload.wikimedia.org/wikipedia/commons/7/7b/Geometric_Dilution_Of_Precision.svg`
Date accessed: 2019-06-14

[19] Narom Andøya Space Center. 2018. *Introduction of the six basic parameters describing satellite orbits.*
`https://www.narom.no/undervisningsressurser/sarepta/rocket-theory/satellite-orbits/introduction-of-the-six-basic-parameters-describing-satellite-orbits/`
Date accessed 2019-05-07

[20] Hofmann-Wellenhof B, Lichtenegger H, Collins J. 2001. *GPS Theory and practice.* Springer; 5th edition.

[21] U.S. Department of Transportation Federal Aviation Administration - Airway Facilities Division. 2003. *Earth Centered Inertial Coordinate System.* FAA Academy Training Manual - GPS concepts, Course 44221.
https://upload.wikimedia.org/wikipedia/commons/3/32/Earth_Centered_Inertial_Coordinate_System.png
Date accessed: 2019-06-01

[22] Mohan A. 2017. *Calculating Position from Raw GPS Data.*
http://www.telesens.co/2017/07/17/calculating-position-from-raw-gps-data/?fbclid=IwAR1McnbXO6qWJAr21cu3hmFbWEc0eneA9rVCp4qTK1HBeIb23sdqgMK8-qY#Conversion_between_Geodetic_Ellipsoidal_and_Cartesion_Coordinates
Date accessed 2019-05-11

[23] Stenborg E, Hammarstrand L. 2016. *Using a single band GNSS receiver to improve relative positioning in autonomous car.* 2016 IEEE Intelligent Vehicles Symposium (IV), Pages 921-926.

[24] ublox. 2018. *ZED-F9P u-blox F9 high precision GNSS module (Datasheet).*
https://cdn.sparkfun.com/assets/8/3/2/b/8/ZED-F9P_Data_Sheet.pdf
Date accessed 2019-02-26

[25] ublox. 2018. *u-blox ZED-F9P Interface Description.*
https://cdn.sparkfun.com/assets/learn_tutorials/8/5/6/ZED-F9P_UBX_NMEA_and_RTCM_protocols.pdf
Date accessed 2019-06-02

[26] Kim D, Langley B. R. 2001. *Instantaneous Real-time Cycle-slip Correction of Dual-frequency GPS Data.* Department of Geodesy and Geomatics Engineering, University of New Brunswick.

[27] Sickle J. 2018. *Cycle Slip.*
https://www.e-education.psu.edu/geog862/node/1728
Date accessed 2019-05-08

[28] Mitchell H. 2007. *Multi-Sensor Data Fusion.* Springer-Verlag Berlin Heidelberg.

[29] Grewal M, Andrews A. 2001. *Kalman Filtering: Theory and Practice using MATLAB.* John Wiley & Sons, Inc, 2nd Edition.

[30] Brown R, Hwang P. 2012. *Introduction to Random Signals and Applied Kalman Filtering.* John Wiley & Sons, Inc.

[31] Yuan X, Lian F, Han C. 2014 . *Models and Algorithms for Tracking Target with Coordinated Turn Motion.* Mathematical Problems in Engineering Volume

2014, Article ID 649276, 10 pages.

[32] *Latitude,Longitude,Height to/from ECEF (X,Y,Z).*
`https://www.oc.nps.edu/oc2902w/coord/llhxyz.htm`
Date accessed: 2019-06-02

[33] Ford D. 2019. *Live Map of Satellite Positions.*
`https://in-the-sky.org/satmap_worldmap.php`
Date accessed: 2019-06-02

[34] Jon L. Vavrus and Brant Miller Clark. *Method and apparatus for hybrid routing using breadcrumb paths.* U.S. Patent 8.498,808 B2, issued July 30, 2013.

[35] Robinson E, Clark D. 2017. *Basic Geophysics.* SEG (Society of Exploration Geophysicists), pages 88-92.

[36] Zak S. 2003. *Systems and Control.* OUP USA.

[37] Dilip A, Pillai H, Jungers R. 2017. *On discrete algebraic Riccati equations: A rank characterization of solutions.* Linear Algebra and its Applications, volume 527 (2017), page 184–215.

[38] Bosch. 2014. *BNO055 Intelligent 9-axis absolute orientation sensor.*
`https://cdn-shop.adafruit.com/datasheets/BST_BNO055_DS000_12.pdf`
Date accessed 2019-05-23

[39] Kiencke U, Nielsen L. 2005. *Automotive Control Systems For Engine, Driveline, and Vehicle.* Springer-Verlag Berlin Heidelberg.

[40] Kanayama Y, Kimura K, Miyazaki F, Noguchi T. 1990. *A Stable Tracking Control Method for an Autonomous Mobile Robot.* IEEE International Conference on Robotics and Automation, Cincinnati, OH, USA, pp. 384-389

[41] Neuendorf N, Bruns T. 2004. *The vehicle platoon controller in the decentralised, autonomous intersection management of vehicles.*Proceedings of the IEEE International Conference on Mechatronics. ICM ’04., Istanbul, Turkey, pp. 375-380.

[42] Stankovic S S, Stanojevic M.J, and Siljak D D. 2000. *Decentralized overlapping control of a platoon of vehicles.* IEEE Transactions on Control Systems Technology, vol. 8, no. 5, pp. 816-832.

[43] Teunissen P. 1993. *Least-Squares Estimation of the Integer GPS Ambiguities.* Section IV "Theory and Methodology", at the General Meeting of the International Association of Geodesy, Beijing, China, August 1993.

[44] Teunissen P. 1994. *The least-squares ambiguity decorrelation adjustment: a method for fast GPS integer ambiguity estimation.* Journal of Geodesy November 1995, Volume 70, Issue 1–2, pp 65–82.

[45] Borno M, Chang X. 2013. *On 'decorrelation' in solving integer least-squares problems for ambiguity determination.* Survey Review Volume 46, 2014 - Issue 334.

[46] Zhou Y, He Z. 2013. *Variance reduction of GNSS ambiguity in (inverse) paired Cholesky decorrelation transformation.* GPS Solutions Volume 18 Issue 4, October 2014 Pages 509-517.

[47] Xie K, Chai H, Pan Z, Wang H, Dong B, Ming L. 2014. *Application of Improved LLL Lattice Reduction in BDS Ambiguity Decorrelation.* China Satellite Navigation Conference (CSNC) 2014 Proceedings: Volume III. Lecture Notes in Electrical Engineering, vol 305. Springer, Berlin, Heidelberg.

[48] Lou L, Grafarend W. E. 2003. *GPS integer ambiguity resolution by various decorrelation methods.* Published in zfv 3/2003,

[49] Seeed Studio. 2018. *Skeleton Bot - 4WD hercules mobile robotic platform.*
`http://wiki.seeedstudio.com/Skeleton_Bot-4WD_hercules_mobile_robotic_platform/`
Date Accessed 2019-02-05

[50] Townsend K. 2015. *Adafruit BNO055 Absolute Orientation Sensor.*
`https://learn.adafruit.com/adafruit-bno055-absolute-orientation-sensor/overview`
Date accessed 2019-02-05

[51] Siciliano. B, Khatib, O. 2016. *Springer Handbook of Robotics.* Springer International Publishing, edition nr 2, pages 479–482.

[52] Caron. F, Duflos. E, Pomorski. D, Vanheeghe. P. 2004. *GPS/IMU Data Fusion using Multisensor Kalman Filtering : Introduction of Contextual Aspects.* Information Fusion Volume 7, Issue 2, Pages 221-230.

[53] Sparkfun. 2019. *SparkFun GPS-RTK2 Board - ZED-F9P (Qwiic).*
`https://www.sparkfun.com/products/15136`
Date accessed 2019-02-05

[54] Arduino. *Arduino due*
`https://store.arduino.cc/arduino-due`
Date accessed 2019-02-05

[55] Curt Franklin and Julia Layton. 2000. *How Bluetooth Works.*

https://electronics.howstuffworks.com/bluetooth2.htm
Date accessed 2019-06-14

[56] Sparkfun. 2009. *SparkFun Bluetooth Mate Silver.*
https://www.sparkfun.com/products/12576
Date accessed 2019-02-05

[57] U-blox. 2018. *ZED-F9P u-blox F9 high precision GNSS module.*
https://cdn.sparkfun.com/assets/8/3/2/b/8/ZED-F9P_Data_Sheet.pdf
Date accessed 2019-05-23

A
Hardware

This section describes the hardware which was used to build the rover in this thesis. It explains the functionality of each component and how they are used to make the platoon function. A circuit diagram is presented at the end to illustrate how the components are connected.

A.1 Rover

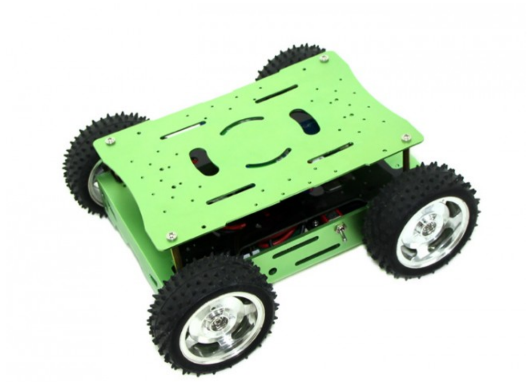


Figure A.1: The four wheel driven rover 'Skeleton Bot' [49]. Image taken from [49].

The rovers used in this thesis are called "Skeleton Bot - 4WD Hercules mobile robotic platform" [49]. The dimensions for this model are 242×207×94 mm and it weights 3 kg with a wheel diameter of 8.5 mm. The vehicle is equipped with four motors, one motor mounted to each wheel. A motor controller ”Hercules Dual 15A 6-20V Motor Controller” [49] is used to control the motors so that different wheel speeds may be achieved. This particular motor controller cannot supply four different inputs to each wheel instead the left and right side are controlled independently.

The Skeleton Bot is differentially steered, where the steering is achieved by having different velocities on each side of the vehicle. The advantage of differential steering is the ability to take extremely sharp turns. The disadvantage, however, is the inaccuracy of the turns since the wheel speeds on each side has to be accurate in order to take the desired turn.

A.2 Inertial Measurement Unit

An Inertial Measurement Unit (IMU) is an electronic component which composes of a combination of accelerometer, gyroscope, and magnetometer. An IMU that consists of all three components measures the mounted body's specific force, angular rate and the magnetic field surrounding the body respectively. The IMU used in this master thesis is of the model BNO055 [50] and has 9 degrees of freedom. This means that the IMU is capable of measuring the specific force, angular rate, and magnetic field in all three axes.

An IMU is mainly used in navigation systems for instance in airborne or ground vehicles. The measurement unit is particularly useful in environments where other external sensors (like GPS) are not able to locate the vehicle. An IMU is an internal sensor located in the vehicle which is able to estimate the vehicles position and orientation by integrating the measurement data. The estimation is given by

$$x = \int (\int \ddot{x} dt) dt, \qquad y = \int (\int \ddot{y} dt) dt, \qquad z = \int (\int \ddot{z} dt) dt, \tag{A.1}$$

$$\varphi = \int \dot{\varphi} dt, \qquad \theta = \int \dot{\theta} dt, \qquad \psi = \int \dot{\psi} dt. \tag{A.2}$$

where x, y, z positions are obtained by integrating the data from the accelerometer twice and φ (roll), θ (pitch) and ψ (yaw) orientations are obtained by integrating the data from the gyroscope.

The way how an IMU is typically being used to estimate position and orientation leads to problems where the measurement errors are getting accumulated with time because of integration [51]. This leads to drift which is an increasing error between the measured orientation or position of the vehicle compared to the actual value. The drift can be negated with the combined use of other sensors such as speed encoders and GPS [52].

A.3 Speed encoder

A speed encoder (also called rotary encoder) is a sensor used for measuring the speed of a wheel. In this thesis, the speed encoders are inbuilt in two out of the four motors, one on each side of the vehicle. It measures the angular velocity of the motor which can be translated to a linear velocity of the wheel mounted on the motor according to

$$v_{wheel} = r_{wheel} \cdot \omega \tag{A.3}$$

where v_{wheel} is the linear velocity of the mounted wheel, r_{wheel} is the wheel radius and ω is the angular velocity of the motor.

The rover considered in this project is a differentially steered vehicle which means that two speed encoders are necessary since the vehicle may have different velocities on each side. The angular velocity from the speed encoders is used together with the measurements from the IMU and GPS to estimate the vehicle states.

A.4 Global Navigation Satellite Systems

A Global Navigation Satellite Systems (GNSS) works through the use of satellite transmitted information that gets received and decoded by a GNSS receiver. The decoded information from multiple different satellites can then be used to calculate certain information, e.g. position, velocity, and heading (see Section 3.3 for GPS trilateration). In this thesis, Global Positioning System (GPS) is used, and it is one of the most common GNSS in the world [14].

There are currently only four available GNSS in the world: GPS made by the United States, GLONASS made by Russia, BeiDou made by China, and lastly, Galileo made by the European Union [14]. Although the systems were developed separately the principle behind their use are the same, the difference lies in how the data is transmitted and decoded. As the accuracy and coverage of a GNSS are dependent on the number of satellites in orbit, (see Section 3) an important difference between the GNSS is the number of available satellites. Currently, GPS has 31 satellites in orbit, GLONASS 25, BeiDou 28, and Galileo 22 [14].

The GPS receiver used in this project is the GPS-RTK2 Board - ZED-F9P [53] which not only decodes the information for navigation but is also able to deliver raw GPS data [24], e.g. pseudorange signals and carrier phase, which is used for our self-developed trilateration. It also allows the usage of Real-Time Kinematic (RTK) (see Section 3.4) which uses the cooperation between receivers to allow high relative accuracy.

A.5 Microcontroller

The microcontroller, Arduino Due [54], is used in this rover due to its ease of implementation and its relatively high operating frequency at 84 MHz in comparison to other Arduinos.

Microcontrollers are normally used in embedded systems, electronic devices or automatic processes such as autonomous vehicles. The microcontroller in this project serves as the central electrical control unit in the vehicle which collects all the sensor data through input peripherals and sends output signals through the output peripherals.

A.6 Bluetooth

Bluetooth is a wireless communication standard which is used to exchange data over short distances. It uses short wavelengths (radio waves) from 2.400 to 2.485 GHz [55] to exchange data between other Bluetooth devices. The Bluetooth devices used in the rover is called 'SparkFun Bluetooth Mate Silver' [56] which has a communication range of 20 m. The Bluetooth device is used for three purposes in this thesis, first to manually control the lead vehicle from an application, second to exchange information between the rovers to determine the optimal control input to the motors so that platooning can be achieved. Thirdly is the communication from the rovers to a separate computer to acquire the data for further analysis after a run.

A.7 Circuit diagram

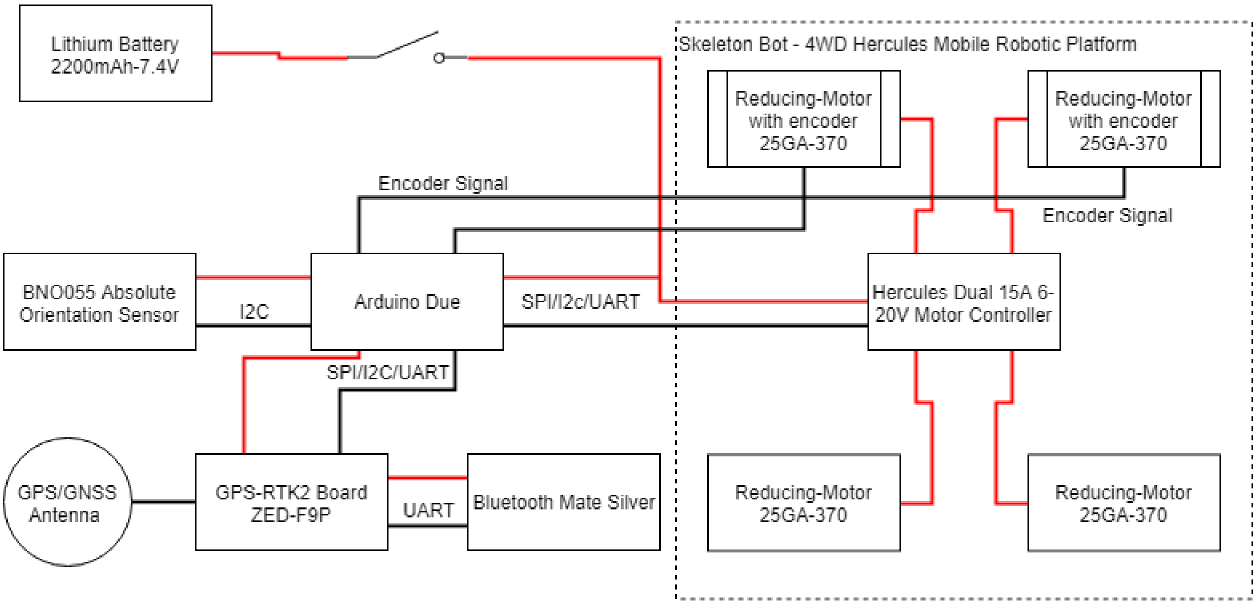


Figure A.2: Simplified schematic of the electrical wiring of the complete rover. Power is defined in red while other connections are defined in black.

In Figure A.2 the overall electronic wiring can be seen. This schematic includes all the different electrical components that are included in the rover, as well as how they are connected. An important observation is the available communications protocol between the different components. This is important as it defines the limits for the wiring and programming.

B

Filter parameter estimation

This chapter describes the method of estimating the filter parameter covariance matrices **Q** and **R** for the sensor fusion filter.

B.1 Filter parameter estimation for the motion covariance matrix

In order to obtain the standard deviation noise matrices **Q** and **R** presented in Section 4.2.3, the variance of the noise $\mathbf{r}_k$ and $\mathbf{q}_{k-1}$ have to be assigned for the simulations. As the true motion noise of the hardware used is unknown, a motion noise was added directly onto the vehicle model in the simulations. The motion noise $\mathbf{q}_{k-1}$ added to the velocity v_k and yaw rate $\dot{\psi}_k$ were set arbitrarily to

$$\mathbf{q}_{k-1} = \begin{bmatrix} 0.01 \\ 0.01 \end{bmatrix}. \tag{B.1}$$

This means that the tuning of the **Q** matrix is unnecessary as the noise of the model is directly known and does not have to be found. However, what is of interest in this thesis is the effect of the relative position measurement on the platoon. Hence a different motion noise matrix where initialized,

$$\mathbf{Q} = \begin{bmatrix} 4 \cdot 10^{-4} & 0 \\ 0 & 4 \cdot 10^{-4} \end{bmatrix}, \tag{B.2}$$

to reduce the performance of the EKF and more clearly observe the effect of the GPS RTK measurement onto the platoon. The values were initialized arbitrarily with the only focus being to not know the true performance of the model.

The variance for the measurement noise was assigned based on the sensor characteristics which is investigated in the next Section B.2.

B.2 Sensor characteristic analysis

The standard deviation noise for each sensor is of interest in order to estimate the noise covariance matrix **R**. This analysis is carried out by taking a sample size of data for each sensor and calculate the standard deviation σ for each one.

For the first test, the rovers are initialized and placed stationary in a horizontal plane.

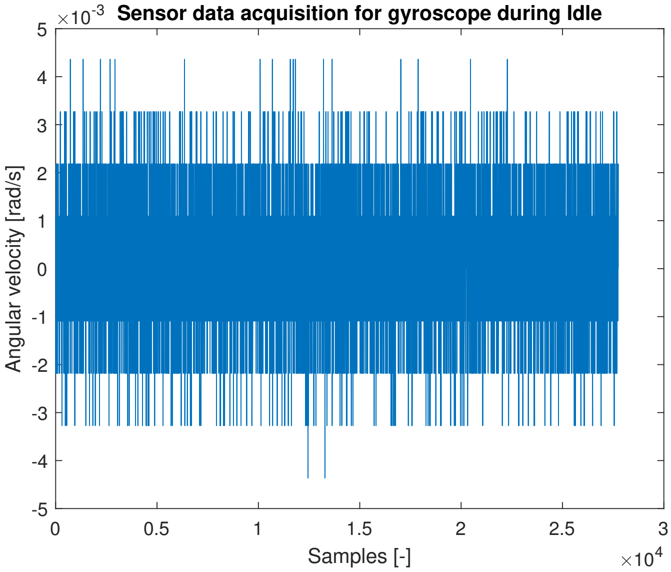


Figure B.1: Sensor data obtained from the gyroscope (Appendix A.2).

The sensor data presented in Figure B.1 is the acquired gyroscope data. The mean $\bar{y}_{gyro}$ and the standard deviation σ_{gyro} of the gyroscope data is obtained by

$$\bar{y}_{gyro} = \frac{1}{n}\sum_{i=1}^{n} y_i = 1.6647 \cdot 10^{-4}, \tag{B.3}$$

$$\sigma_{gyro} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2} = 1.2 \cdot 10^{-3}, \tag{B.4}$$

where y_i is i-th the sample data illustrated in Figure B.1. The gyroscope is mostly subjected to noise with a negligible offset.

The sensor data for the inbuilt orientation sensor is presented in Figure B.2. The orientation sensor is a combination of a magnetometer and a gyroscope. The mean and the standard deviation for the data acquired in Figure B.2 is given by

$$\bar{y}_{orientation} = \frac{1}{n}\sum_{i=1}^{n} y_i = 6.5 \cdot 10^{-3}, \tag{B.5}$$

$$\sigma_{orientation} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2} = 5.6 \cdot 10^{-3}. \tag{B.6}$$

The data is only affected by offsets and is not subjected to Gaussian white noise. This is because the data is filtered by an inbuilt function in the IMU.

The data acquisition for the GPS RTK is omitted from this section because the GPS sensor (see Appendix A.4) wasn't successfully integrated in the hardware platform.

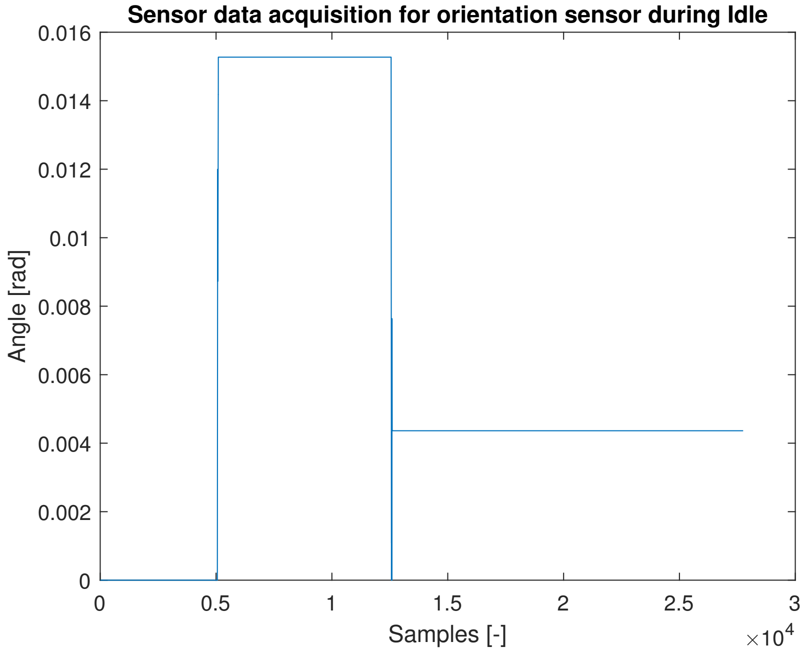


Figure B.2: Sensor data obtained from the orientation sensor (Appendix A.2).

Therefore, the standard deviation is directly taken from the datasheet of the GPS [57] which is equal to

$$\sigma_{GPSrtk} = 0.015. \tag{B.7}$$

Assuming the standard deviation of equation (B.7) being equally distributed in the X and Y axis, the standard deviation for the X position σ_X and the Y position σ_Y is given by

$$\sigma_{GPSrtk}^2 = \sigma_X^2 + \sigma_Y^2, \tag{B.8}$$

$$\sigma_Y = \sigma_X = \frac{0.015}{\sqrt{2}}. \tag{B.9}$$

The data acquisition of the wheel speed sensors is introduced in Appendix A.3. It counts the number of pulses in a time interval which is then translated to an angular velocity. These sensors are not subjected to Gaussian normal distributed noise but can count different pulses for the same velocity which acts as disturbance. The rover has to be in motion in order to obtain pulses which means the sensor data acquisition cannot be done while idling.

The experiment is performed by giving a constant input u_0 to the motors and log the output data from the wheel speed sensors. The data for this experiment is presented in Figure B.3 for the left wheel speed sensor and in Figure B.4 for the right wheel speed sensor. At around sample 180, both the left and right wheel speed sensors produces a measurement of zero. It is unclear exactly why this occurred, but we believe the pulses where not correctly measured by the microcontroller. This loss of measurement is accounted for in the calculation of the sensor accuracy.

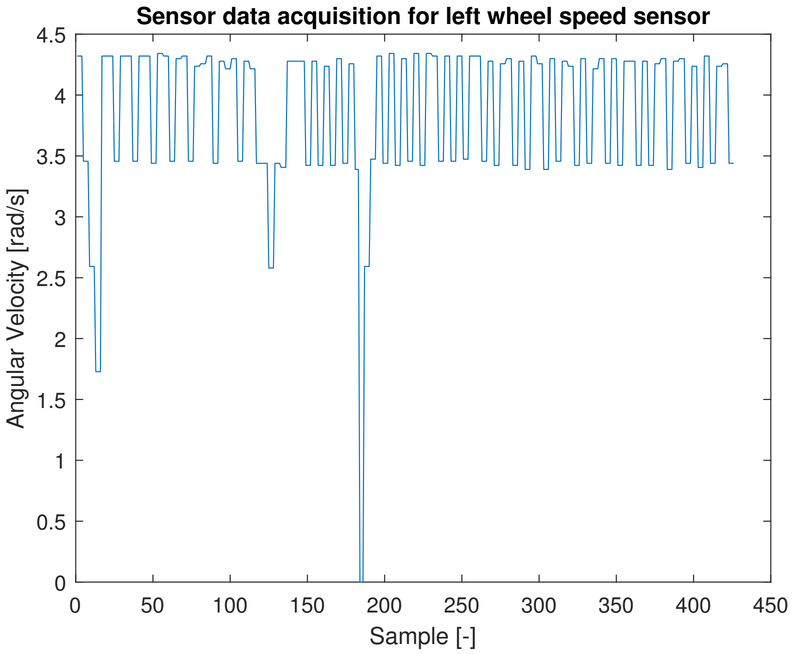


Figure B.3: Sensor data obtained from the left wheel speed sensor (Appendix A.3).

The standard deviation for the wheel speed sensors are given by

$$\bar{y}_{\omega_l} = \frac{1}{n}\sum_{i=1}^{n} y_i = 3.8524, \tag{B.10}$$

$$\sigma_{\omega_l} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2} = 0.6082. \tag{B.11}$$

and

$$\bar{y}_{\omega_r} = \frac{1}{n}\sum_{i=1}^{n} y_i = 3.8826, \tag{B.12}$$

$$\sigma_{\omega_r} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2} = 0.5467. \tag{B.13}$$

With the standard deviation noise obtained for all of the sensors, the matrices $\mathbf{R}^L$ and $\mathbf{R}^F$ presented in Section 4.2.3 is set equal to the variance of each sensor

$$\mathbf{R}^L = \begin{bmatrix} 0.6082^2 & 0 & 0 & 0 \\ 0 & 0.5467^2 & 0 & 0 \\ 0 & 0 & 0.0056^2 & 0 \\ 0 & 0 & 0 & 0.0012^2 \end{bmatrix}, \tag{B.14}$$

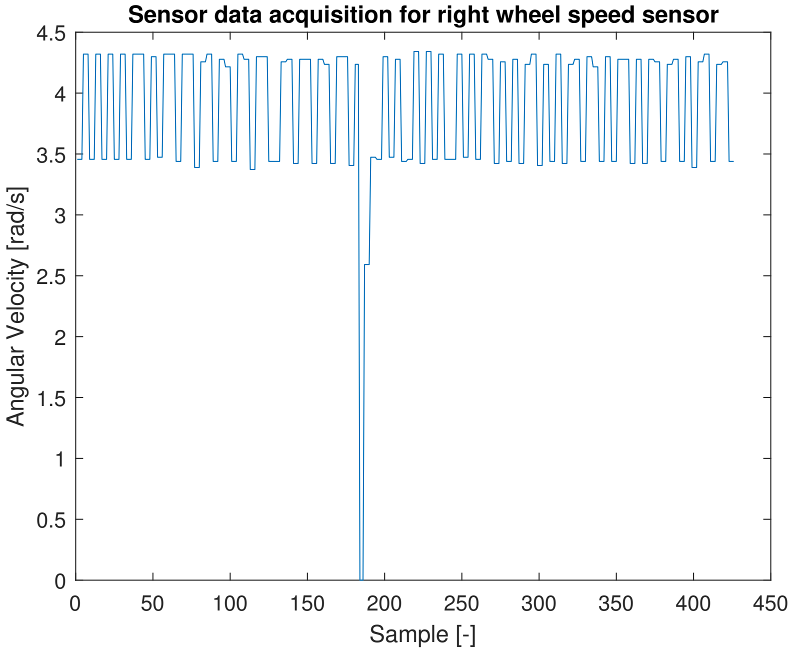


Figure B.4: Sensor data obtained from the right wheel speed sensor (Appendix A.3).

$$\mathbf{R}^F = \begin{bmatrix} \frac{0.015^2}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{0.015^2}{\sqrt{2}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.6082^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5467^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0056^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.0012^2 \end{bmatrix}. \tag{B.15}$$

These matrices are used in the simulations when generating the results.

C

Figures for results

This part of the Appendix includes figures that are referenced in the result. These figure are mainly added to give a better understanding to the result but are not necessary to show the result the thesis focuses on.

C.1 Results for TD

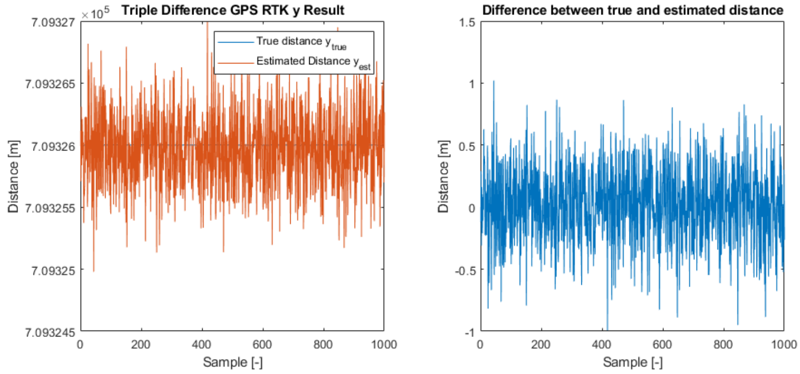


Figure C.1: TD GPS RTK measurement samples, Y axis, assuming known leader position and adding noise onto the previous and initial estimation.

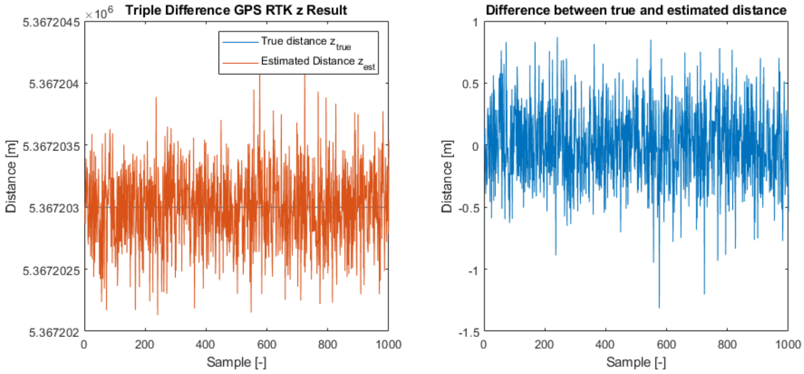


Figure C.2: TD GPS RTK measurement samples, Z axis, assuming known leader position and adding noise onto the previous and initial estimation.

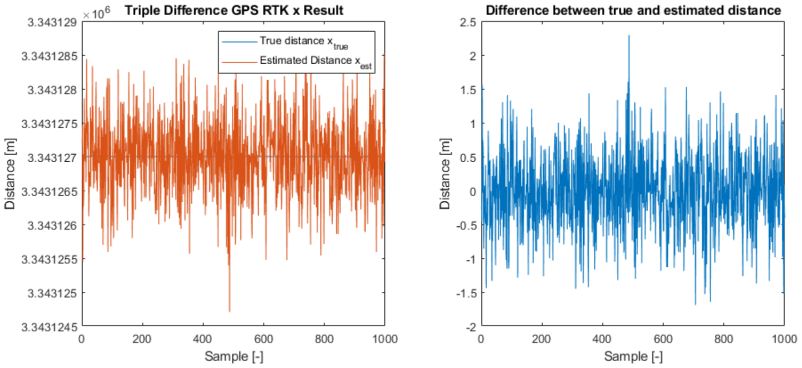


Figure C.3: TD GPS RTK measurement samples, X axis, assuming known leader position. Adding noise onto the previous estimation with a standard deviation based on the previous found $\sigma_0^X = 0.3184$, $\sigma_0^Y = 0.3162$ and $\sigma_0^Z = 0.3139$.

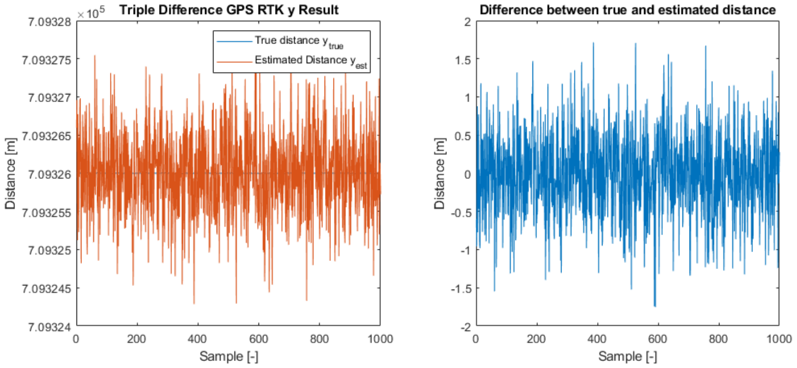


Figure C.4: TD GPS RTK measurement samples, Y axis, assuming known leader position. Adding noise onto the previous estimation with a standard deviation based on the previous found $\sigma_0^X = 0.3184$, $\sigma_0^Y = 0.3162$ and $\sigma_0^Z = 0.3139$.

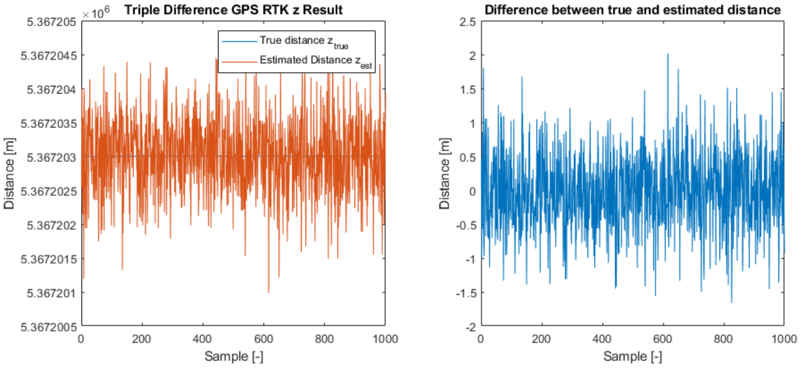


Figure C.5: TD GPS RTK measurement samples, Z axis, assuming known leader position. Adding noise onto the previous estimation with a standard deviation based on the previous found $\sigma_0^X = 0.3184$, $\sigma_0^Y = 0.3162$ and $\sigma_0^Z = 0.3139$.

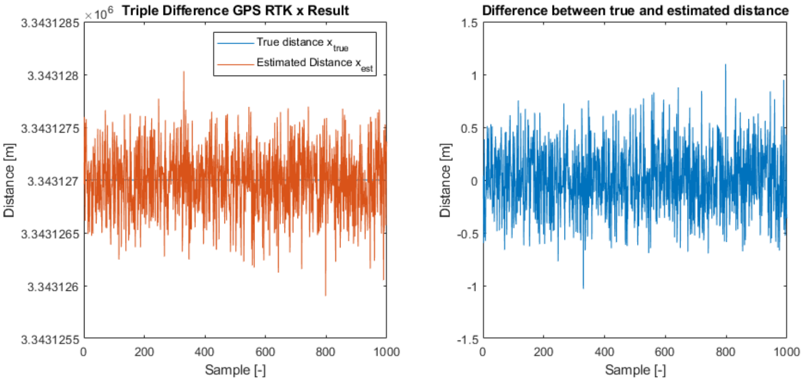


Figure C.6: TD GPS RTK measurement samples, X axis, assuming known leader position. Adding noise onto the initial estimation with a standard deviation based on the previous found $\sigma_0^X = 0.3184$, $\sigma_0^Y = 0.3162$ and $\sigma_0^Z = 0.3139$.

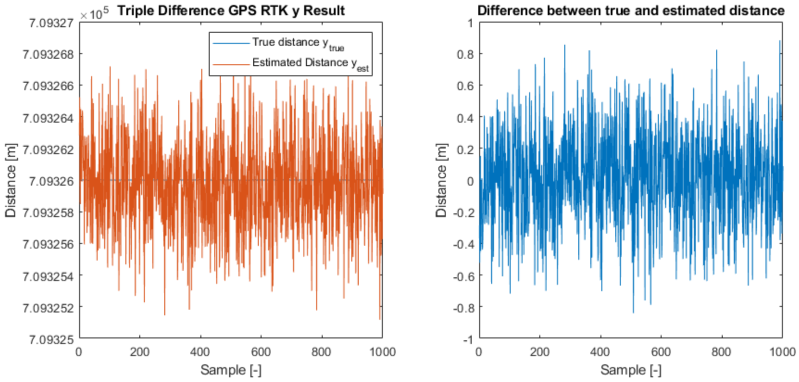


Figure C.7: TD GPS RTK measurement samples, Y axis, assuming known leader position. Adding noise onto the initial estimation with a standard deviation based on the previous found $\sigma_0^X = 0.3184$, $\sigma_0^Y = 0.3162$ and $\sigma_0^Z = 0.3139$.

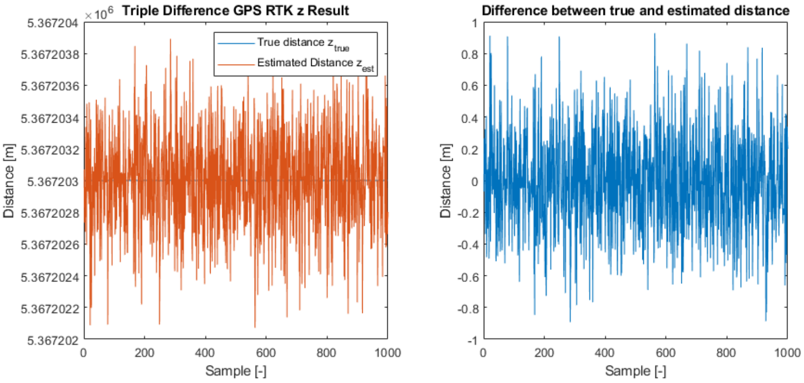


Figure C.8: TD GPS RTK measurement samples, Z axis, assuming known leader position. Adding noise onto the initial estimation with a standard deviation based on the previous found $\sigma_0^X = 0.3184$, $\sigma_0^Y = 0.3162$ and $\sigma_0^Z = 0.3139$.

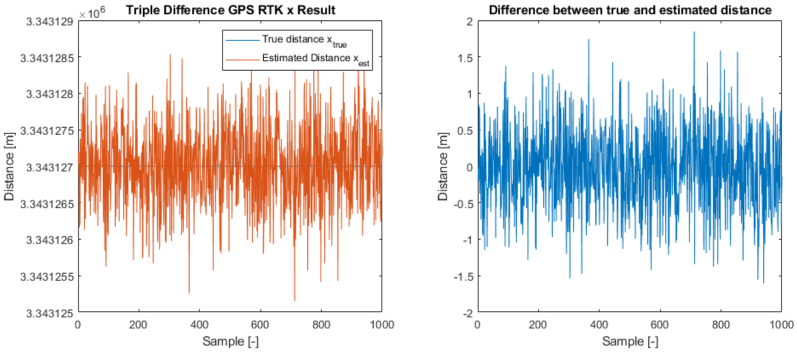


Figure C.9: TD GPS RTK measurement samples, X axis, assuming known leader position. Adding noise onto the initial and previous estimation with a standard deviation based on the previous found $\sigma_0^X = 0.3184$, $\sigma_0^Y = 0.3162$ and $\sigma_0^Z = 0.3139$.

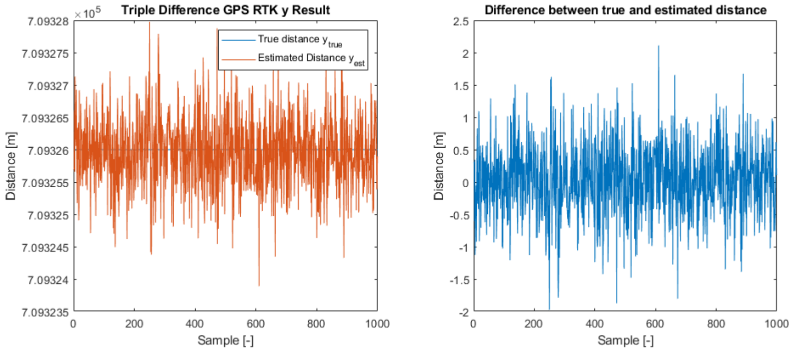


Figure C.10: TD GPS RTK measurement samples, Y axis, assuming known leader position. Adding noise onto the initial and previous estimation with a standard deviation based on the previous found $\sigma_0^X = 0.3184$, $\sigma_0^Y = 0.3162$ and $\sigma_0^Z = 0.3139$.

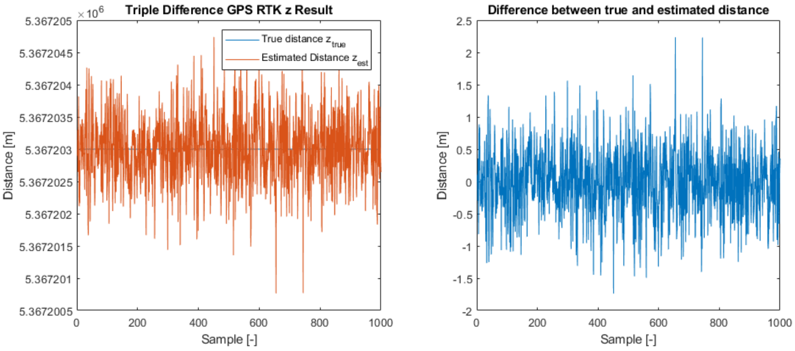


Figure C.11: TD GPS RTK measurement samples, Z axis, assuming known leader position. Adding noise onto the initial and previous estimation with a standard deviation based on the previous found $\sigma_0^X = 0.3184$, $\sigma_0^Y = 0.3162$ and $\sigma_0^Z = 0.3139$.

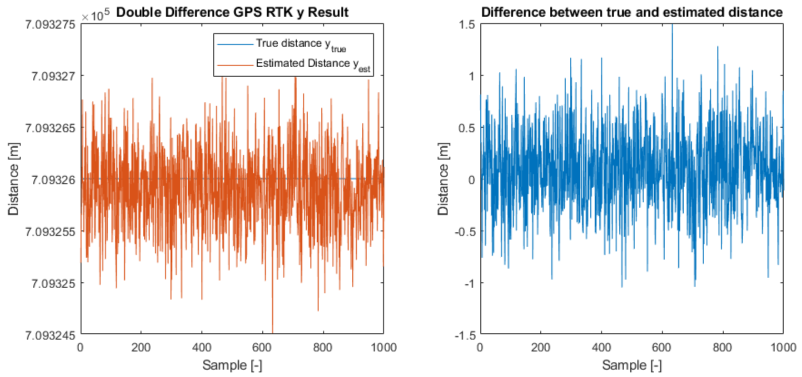


Figure C.12: DD GPS RTK measurement samples, Y axis, assuming known leader position and previous estimated follower position. Adding noise onto the previous and initial estimation with a standard deviation of 1 cm.

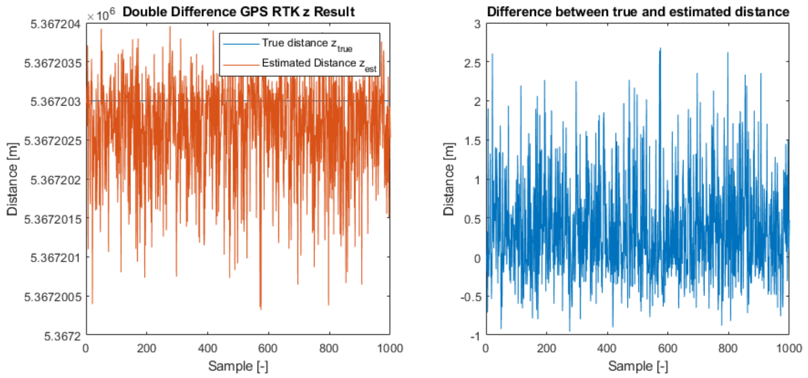


Figure C.13: DD GPS RTK measurement samples, Z axis, assuming known leader position and previous estimated follower position. Adding noise onto the previous and initial estimation with a standard deviation of 1 cm.

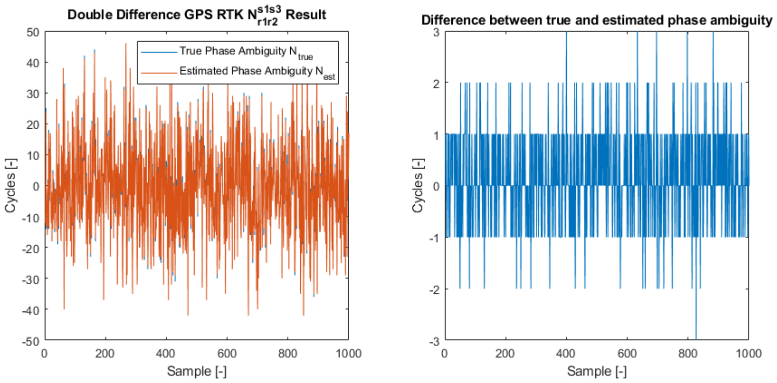


Figure C.14: DD GPS RTK measurement samples, ambiguity $\aleph^{s1s3}_{r1r2}$, assuming known leader position and previous estimated follower position. Adding noise onto the previous and initial estimation with a standard deviation of 0.01.

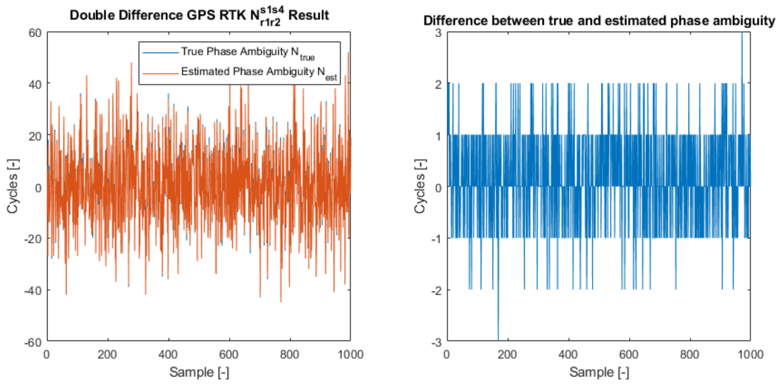


Figure C.15: DD GPS RTK measurement samples, ambiguity $\aleph^{s1s4}_{r1r2}$, assuming known leader position and previous estimated follower position. Adding noise onto the previous and initial estimation with a standard deviation of 0.01.

C.2 Results for DD

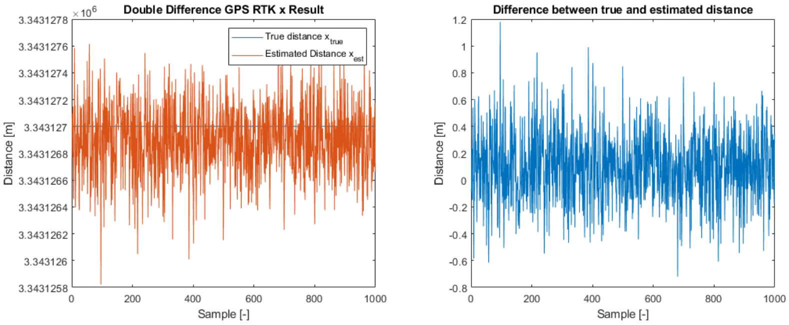


Figure C.16: 1000 DD GPS RTK measurement samples, X axis, assuming known leader position and previous estimated follower position. Having known initial estimation of the ambiguities.

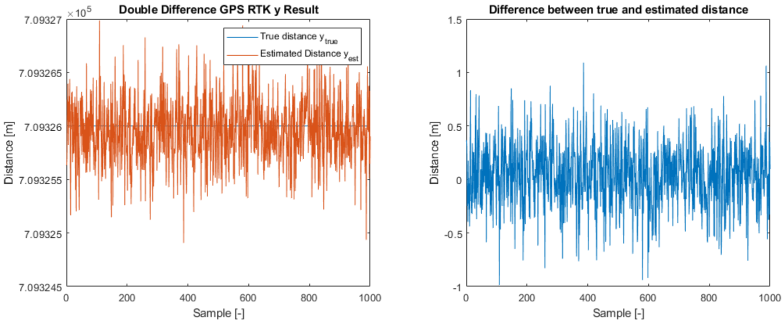


Figure C.17: 1000 DD GPS RTK measurement samples, Y axis, assuming known leader position and previous estimated follower position. Having known initial estimation of the ambiguities.

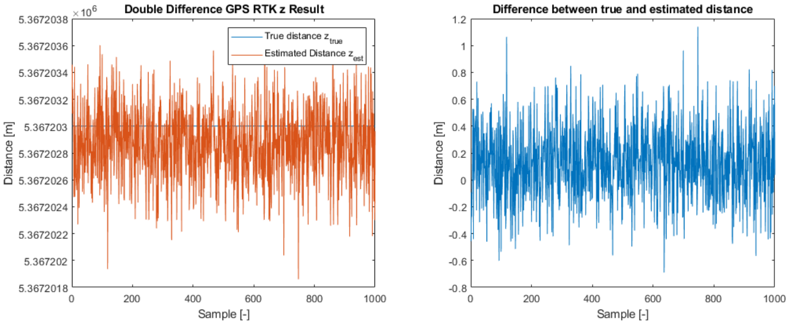


Figure C.18: 1000 DD GPS RTK measurement samples, Z axis, assuming known leader position and previous estimated follower position. Having known initial estimation of the ambiguities.

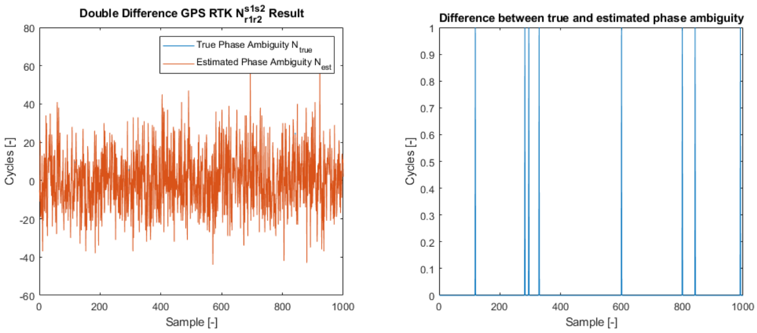


Figure C.19: 1000 DD GPS RTK measurement samples, ambiguity $\aleph^{s1s2}_{r1r2}$, assuming known leader position and previous estimated follower position. Having known initial estimation of the ambiguities.

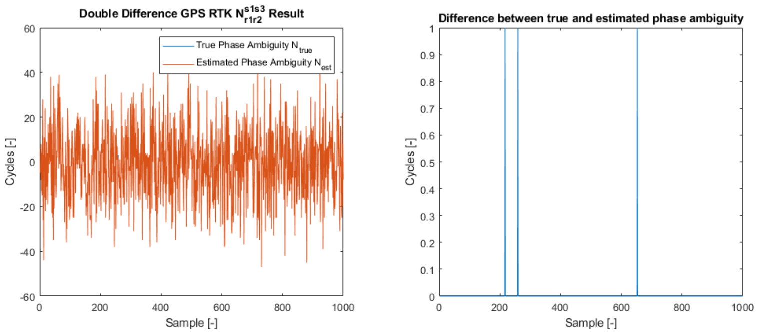


Figure C.20: 1000 DD GPS RTK measurement samples, ambiguity $\aleph^{s1s3}_{r1r2}$, assuming known leader position and previous estimated follower position. Having known initial estimation of the ambiguities.

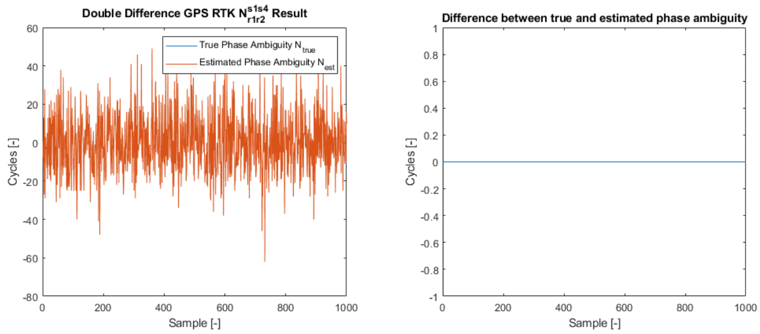


Figure C.21: 1000 DD GPS RTK measurement samples, ambiguity $\aleph_{r1r2}^{s1s4}$, assuming known leader position and previous estimated follower position. Having known initial estimation of the ambiguities.

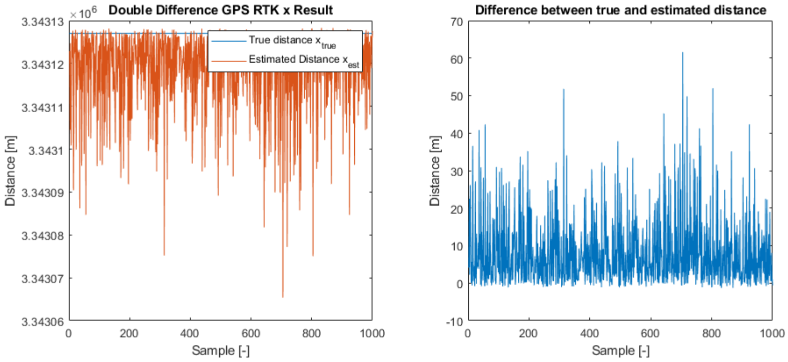


Figure C.22: 1000 DD GPS RTK measurement samples, X axis, assuming known leader position and previous estimated follower position. Having completely unknown initial estimation of the ambiguities.

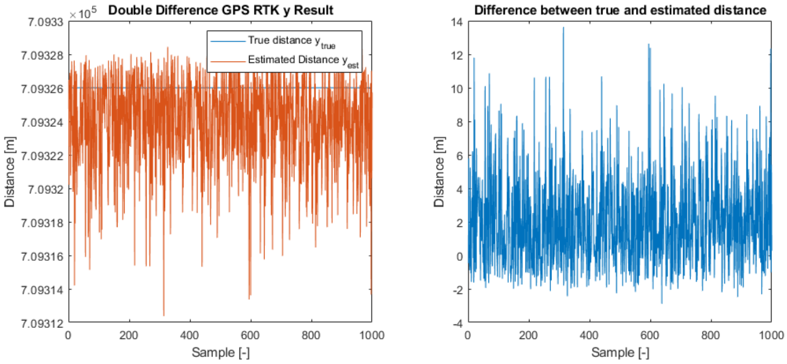


Figure C.23: 1000 DD GPS RTK measurement samples, Y axis, assuming known leader position and previous estimated follower position. Having completely unknown initial estimation of the ambiguities.

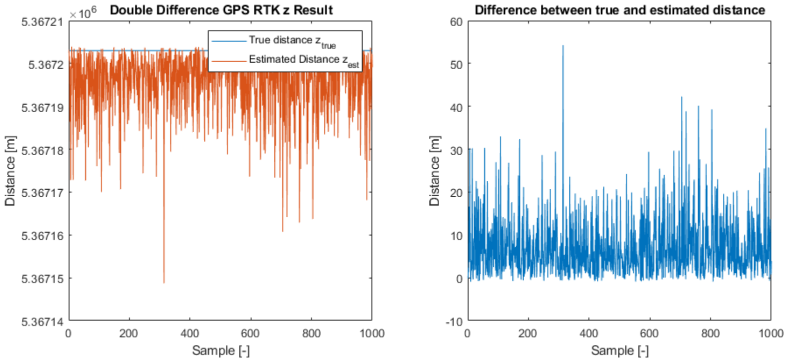


Figure C.24: 1000 DD GPS RTK measurement samples, Z axis, assuming known leader position and previous estimated follower position. Having completely unknown initial estimation of the ambiguities.

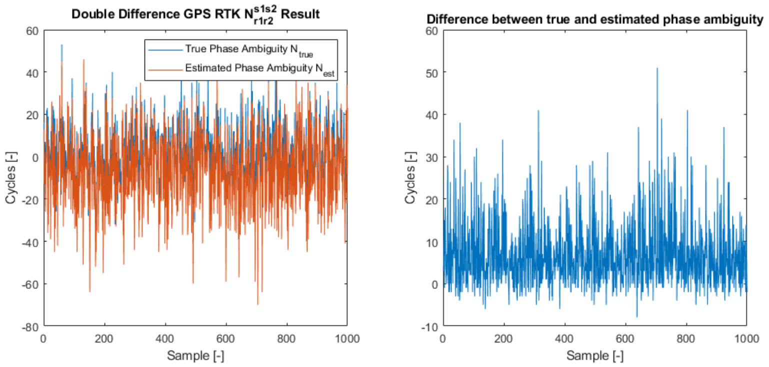


Figure C.25: 1000 DD GPS RTK measurement samples, ambiguity $\aleph_{r1r2}^{s1s2}$, assuming known leader position and previous estimated follower position. Having completely unknown initial estimation of the ambiguities.

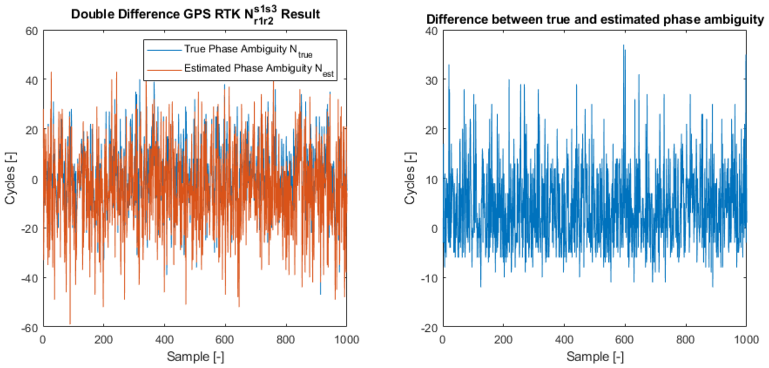


Figure C.26: 1000 DD GPS RTK measurement samples, ambiguity $\aleph_{r1r2}^{s1s3}$, assuming known leader position and previous estimated follower position. Having completely unknown initial estimation of the ambiguities.

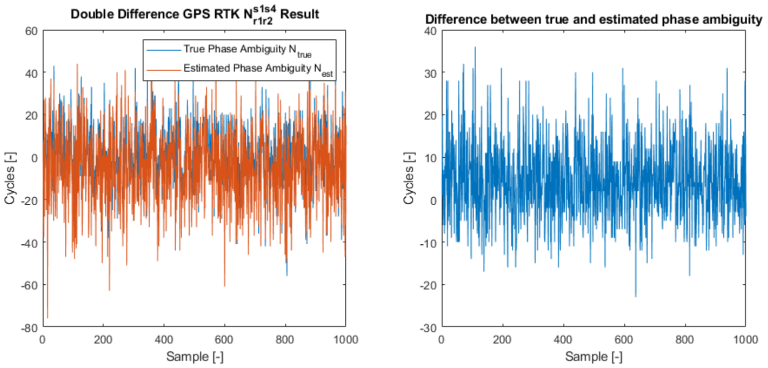


Figure C.27: 1000 DD GPS RTK measurement samples, ambiguity $\aleph_{r1r2}^{s1s4}$, assuming known leader position and previous estimated follower position. Having completely unknown initial estimation of the ambiguities.