NTNU – Trondheim
Norwegian University of
Science and Technology

Autonomous landing UAV in a net on ship

Kjetil Hope Sørbø

Submission date: December 2015
Responsible professor: Thor Inge Fossen, Tor Arne Johansen
Supervisor: Firstname Lastname, Affiliation

Norwegian University of Science and Technology
Department of Telematics

Abstract

Abstract

Preface

Contents

List of Figures

List of Tables

List of Acronyms

DGPS Differential GPS.

ECEF Earth-Centered-Earth Fixed.

ECI Earth-Centered-Inertial.

ENU East North Up.

GLONASS Global Navigation Satellite System.

GNSS Global Navigation Satellite System.

GPS Global Positioning System.

LAMBDA Least-squares AMBiguity Decorrelation Adjustment.

NED North East Down.

NTNU Norwegian University of Science and Technology.

RTK-GPS Real Time Kinematic GPS.

UAV Unmanned Aerial Vehicle.

WGS-84 World Geodetic System, 1984.

Chapter 1

Introduction

1.1 Background and motivation

In the resent past the development of flying Unmanned Aerial Vehicles (UAVs) have seen to provide an attractive alternative to task previously performed by manned alternatives. Typical task which has attracted attention includes inspection, aerial photography, environmental surveillance and search and rescue. Today UAV are mostly operated over land, but in the future this will include the out at sea as well. This will give some challenges that must be overcome. One of these challenges is that the UAV should be able to perform a autonomous landing.

A UAV can increase performance in many maritime operation where today only other manned aircraft or satellites are the only solution. In the maritime sector they can be used in iceberg management, monitoring of oil spills, search and rescue and maritime traffic monitoring. To enable a safe UAV operation at sea there must be a system in place to ensure a safe landing.

There exist landing system that can guide the UAV towards a net, but they are expensive and restricted to a few UAVs. A pilot could always land the UAV, but it would be better if the UAV could hit the net by it self. In order to make the UAV able to perform a automatic landing the minimum requirement is that it know where it is at all time. This require a accurate position estimation in real time. A highly accurate position sensor is typically expensive, however it's possible to achieve a accurate solution with low cost sensors. By combining two Global Navigation Satellite System (GNSS) receivers it's possible to estimate the relative position of one of the receivers in respect of the other receiver highly accurately. This work will test a new generation of GNSS receiver, and use Global Positioning System (GPS) to find the relative position of the UAV. The demand on the accuracy is that the error must be in decimetres to ensure safe landing in the net.

In order to complete an automatic landing a path planner, guidance system and a accurate position estimation system, in addition to the low level control system in the UAV is required. Such complete system is still subject for development, where only method for achieving the different objectives has been developed, i.e. path planner and guidance system respectively. The path planer and guidance system as basis for this work was developed as previous master thesis work in the UAVLAB by [Frølich, 2015]. The gudiance system is currently under further development in two project thesis. A key component in the guidance system is the position estimation system, which for the guidance referenced was developed by [Spockeli, 2015]. However, this system has been concluded to be in-sufficient with respect to provide means required for automatic landing. This work will continue the research done by Spockeli, and introduce a new GNSS receiver that will be used with the open source program, rtklib. The motivation is to have a system with accurate local position estimate, such that in the future a landing can be performed automatic.

The automatic landing system will use Real Time Kinematic GPS (RTK-GPS) for position estimation. The main motivation for this work is to describe the gaps of the available position system sufficient to scope further work required for closure of such gaps ultimately providing means for position estimating sufficient for completion of automatic landing.

1.2 Literature Review

Citation checking [Frølich, 2015, Spockeli, 2015] or Frølich [2015], Spockeli [2015] or Frølich [2015], Spockeli [2015]

GNSS navigation has been As part of the NTNU MSc thesis of [Skulstad and Syversen, 2014] they successfully demonstrated a net landing with a fixed wing UAV using a low-cost RTK GNSS system. More about how they did it.

An other NTNU MSc thesis by [Spockeli, 2015] simulated a net landing with a fixed wing UAV. Here the net was allowed to be dynamical, and the system was design with landing on a ship in mind. The system used artificial neural network to estimate the position of the net.

As part of the Stellenbosch University MSc thesis [Smit, 2013] demostraded a autonomous landing on a airfield using decoupled linear controllers and DGPS.

An other vision [Kim et al., 2013]

The uavlab at NTNU has in the last year studied how to perform a autonomous landing with a fixed wing UAV. Automatic landing of a UAV in a net using low-cost single frequency gps is described in [Skulstad and Syversen, 2014], and a path

generation system that is integraded with neptus is described in [Frølich, 2015]. Research on RTK GPS intergration in Dune is described in [Spockeli, 2015].

Work done on ambiguity Blewitt [1989], Teunissen and Tiberius [1995], Teunissen [1994, 1995] Work done with rtk gps [Stempfhuber and Buchholz, 2011] [Stempfhuber, 2013]

There has been done work on autonomous landing system using a vision aided system. The work done in [Williams and Crump, 2012] describe a intelligent vision aided landing system that detect and generate landing waypoints for a unsurveyed airfield. Other work

Commercial landing system: [INSITU]

1.3 System layout

An integrated system required for automatic landing will typically consist of four main sub-systems as shown in figure 1.1. Today these systems are individually available or in development; however not integrated into a proven working system allowing for automatic landing of UAVs. The four main systems comprises of the navigation part, guidance part, control part and the user interface, where further validation and development of the first is the main topic for this work.

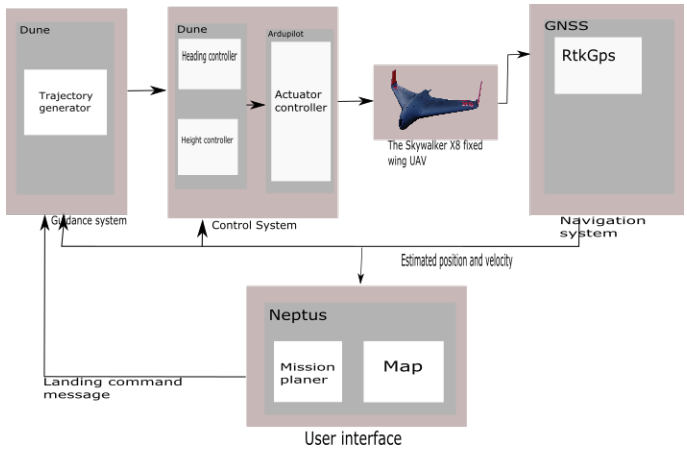

Figure 1.1: The structure of the autonomos landing system

The navigation system will apply RTK-GPS to estimate the relative position of the UAV. More details about what RTK-GPS is, will be given in 3.5.3.

The control and guidance system has currently only been tested in Software In the Loop (SIL) simulations with Ardupilot, which shows promising result likely to be sufficient for automatic landing applications. However this module has not yet been implemented to support the use of RTK-GPS as required for performing automatic landing.

This work will identify and describe the navigation system necessary to allow for automatic landing of UAVs. The RTK-GPS solution will be compute by two different system ,which will be compared against each other. The first system is called Piksi, and is developed by Swift navigation. The second system will is called Rtklib, and is developed by T. Takasu. The latter is indepente on the type of receiver, and thus will be the main focus of this work. More detail about Piksi and Rtklib will be given in 4.2.5 and 4.2.6.

The comparison test will be used to summarize further work required for complete identification of gaps and work required for closure of these gaps such that a control system including provision for automatic landing may be developed and implemented in a integrated UAV control system. The UAV that will be used in the test is a Skywalker X8 Fixed Wing UAV. More details about the X8 will be given in 4.1.1

This work will identify and describe the gaps which will need to be closed on order to develop and implement a solution which will allow for automatic landing of UAVs. Two different system that will be compared against each other. The current state of the system is that it consist of two parts that has not yet been integrated with each other. The part were which is the main focus of this thesis is the positioning part. The plan is to use RTK-GPS for positioning estimation. The second part is the path and control path. Currently there has only been done simulation of the guidance system. It shows promising results, but is yet to be integrated with RTK-GPS. For more information on the subject please read the MSc thesis by [Frølich, 2015]. Figure 1.1 gives a overview on how the system may look like. Ideas on how the intergration can be acheived will be given in the future work section.

1.4 Scope of work

The scope of this work is to test the performance of ublox-LEA M8T GNSS receiver in a real time differential Differential GPS (DGPS) configuration. The RTK-GPS solution will be calculated with the open source program rtklib, which will communicate with a task in DUNE. The solution from rtklib will be compared against the solution from Piksi, and a post processed solution from rtklib. The result from the experiment will be used in the discussion on how to perform a autonomous landing.

1.5 Layout of thesis

This section is currently not up to date

Chapter 1 Intro

Chapter 2 Theory about coordinate system

Chapter 3 Theory about GNSS system with focus on the GPS

Chapter 4 Hardware and software

Chapter 5 Test and result

Chapter 6 Conclusion, discussion and further work

Chapter 2

Reference frames

This chapter will explain the reference frame that will be mentioned in this thesis. There are four different frames that will be explained.

2.1 ECI

Earth-Centered-Inertial (ECI) frame is considered a inertial frame for terrestrial navigation. The origin is fixed in the center of the Earth, and the axis is fixed in space. This frame can be considered as a non-accelerating where Newton's laws of motion applies.

2.2 ECEF

The Earth-Centered-Earth Fixed (ECEF) coordinate system is defined in the center of the Earth with it's x-axis point toward the intersection between the Greenwich meridian and Equator (0 deg longitude, 0 deg latitude). The z-axis points along the Earth's rotational axis, and the y-axis complete the right handed orthogonal coordinate system. The ECEF system can be represented in either Cartesian coordinates (xyz) or ellipsoidal coordinates (longitude, latitude and height). The ECEF frame rotate relative to the ECI frame at a angular rate of $\omega_e = 7.2921 \times 10^{-5} rad/s$, where ω_e is the Earth rotation. Due to the relatively low rotation speed some system can consider the ECEF frame as inertial.

2.3 NED and ENU

The North East Down (NED) and East North Up (ENU) frame is defined as relative to the Earth reference ellipsoid (World Geodetic System, 1984 (WGS-84)). For the NED frame the x axis points in the direction to the true North, y axis towards

East while the z axis points downward to completed the right handed orthogonal coordinate system. The ENU has the x and y axis exchange place in respect of the NED frame, and the z axis point upwards instead of downwards.

Chapter 3

Real time kinematic GPS

In the following the behaviour of a receiver is denoted using term like rover and base station. The term base station means a receiver that is assumed to have a known position by the other receivers. The term rover means a receiver that is allowed to move, is the main focus for position estimation.

The outline of this chapter is first give a overview of current GNSS and system that is under development. For the rest of the chapter the main focus will be GPS, however the sections about error sources and error mitigation is common in all GNSS constellations.

This chapter outline the basic of the GPS signal. how RTKGPS works. The first section give a brief summary on what differential GPS is, and how that principle is applied in RTK-GPS. The two following sections is directly used in RKT-GPS(maybe write some more). The last section give a quick overview over the error sources that effect the measurement.

A short description of GPS signals and error sources. How to find the Ambiguity resolution and why it's important. What is differential gps, and why use RTK-GPS.

This chapter will explain what is meant by the term rtk-gps.

3.1 GNSS constelations

There are currently two operation GNSS constellations with global coverage, which is American GPS and Russian Global Navigation Satellite System (GLONASS). Other GNSS constellations that will be operational in the near future is the Chinese BeiDou and European Galileo.

3.2 GPS attributes

The GPS satellites transmits continuously using two radio frequencies in the L-band referred to as Link 1 (L1) and l2. The L-band covers frequencies between 1 GHz and 2GHz, and is a subset of the Ultra-High Frequency (UHF) band.

Previously only the L1 signal was intended from civil users, but in the future a new l2 signal as well as the Link 5 (L5) signal will be available for civil users.

GPS uses it's time reference called GPS Time (GPST), which is independent from the Coordinated Universal Time (UTC). Because of this independent the UTC diverge away from the GPST, and to correct this the GPS keeps track of the offset between GPST and UTC. The offset is included in the GPS message as leap seconds to be added to the GPST. GPST can be given as Time Of Week (TOW), which include the number of weeks since 1980-01-06T00:00Z. TOW is given in seconds, and is reset each week when the week number is incremented. More information about the GPS can be found in [Misra and Enge, 2011, Vik, 2014]

The two basic ways to measure the psedurange is code and phase measurement. Phase measurement is the most accurate of the two, but also least reliable. In code measurement the information in signal is used to calculate the psudorange between the receiver and the satellite. In phase measurement the signal itself is used to calculate the psudorange by counting the number of cycles between the receiver and the satellite. It's phase measurement that is used in RTK-GPS.

The receiver needs at least four satellite to be able to estimate the receiver position. Three of the satellite is used for the position, and the fourth if used to calculate the receiver clock bias.

3.3 Error sources

In order to get high accuracy in the position estimation the different error sources must be identified and removed if possible. This section will identify some of the larger error sources that can affect the GPS signal, and how to remove or mitigate them in the estimation.

3.3.1 Clock error

There is drift in both the satellite clock and the receiver clock. The atomic in the satellites makes the clock drift negligible from the user perspective. The receiver clock tend to drift, and if not taken into account will cause large deviations in the position estimate from the true position. This error is remove by including a fourth

satellite in the position computation. The satellite clock error given in the satellite message.

3.3.2 Ionospheric and Trophospheric Delays

When the GPS signals travel though the atmosphere there will be a delay caused by the different layers, as further explained in this section.

Ionospheric delay

Gas molecules in the ionosphere becomes ionized by the ultraviolet rays that is emitted by the sun, which release free electrons. These electron can influence electromagnetic wave propagation, such as GNSS signals. The delay that the single get from the ionosphere may cause a error the the order of $1 - 10 meters$. The error can be mitigated by using a double frequency receiver, or by applying a mathematical model to estimate the delay. Both those cases is with a single receiver, but by having a second receiver the GNSS solution system can assume that both receiver receive signal in the same epoch, which means that the signals have experienced the same delay. More on this in section 3.5.2.

Tropospheric delay

The tropospheric delay is a function of the local temperature, pressure and relative humidity. The delay can vary from 2.4meters to 25 meters depending on the elevation angle of the satellites. The error can be mitigated by applying a mathematical model to estimate the tropospheric delay, and by using a elevation mask can remove all satellites with a elevation angle bellow a certain threshold. Error caused by tropospheric delay can be removed in the same manners as ionospheric delay when using two or more receivers. More on this in section 3.5.2.

3.3.3 Ephemeris Errors

A satellite isn't able to perfectly follow a given orbit, and therefore there will be a deviation between satellite position given to the receiver and the true position of the satellite. This is called the ephemeris error. The true position of a satellite is monitored and corrected by the owner of the GNSS constellation, but error between each correction can be expected.

3.3.4 Multipath

One of the primary source of error in in a GNSS receiver is multipath. Multipath happens when the satellite signal is reflected by a nearby surface before if reach the antenna. The delay introduced in the signal can make the receiver believe that its

position is several meters away form its true position. The easiest way to mitigated multipath is to place the antenna at a location with open skies, and not tall structures nearby.

3.4 Dilution of Precision

The geometry of the GNSS constellation affect the accuracy of the position solution. Pore geometry will also enhance the effect that different error sources has on the position solution.

3.5 Differential GPS

Differential GPS consist of at least two receivers, where one is called a base station and the rest rovers. The two receivers are within range of a communication channel over which they are communicating. There are two basic ways to implement DGPS. There is the position-space method and the range-space method. Only the latter will be covered in this thesis. Want it as low as possible

3.5.1 Interger Ambiguity Resolution

The integer ambiguity is the uncertainty of phase cycles between the receiver and the satellites.

There are several strategies on how to resolve the integer ambiguity. A well used strategy is the Least-squares AMBiguity Decorrelation Adjustment (LAMBDA) method. LAMBDA starts by reducing the integer search space by decorrelation adjustment. The LAMBDA method has two types of outputs. One is called the fixed solution, and the other is called the float solution. The float solution is the first solution given by the LAMBDA method and is used to find the fixed solution. When the right fixed solution is reached the position estimation in from a DGPS can be considered highly accurate. The solution program can calculate the wrong fixed solution, or experience a cycle slip. In order to reduce the possibility of letting a wrong solution become the fixed solution the program need a good integer ambiguity validation strategy. One validation strategy is to check if the ratio between the best ambiguity estimate and the second best estimate in greater then a certain threshold. High Dilution Of Precision (DOP) will effect the time the LAMBDA method needs to find a fixed solution.

3.5.2 Error mitigation in DGPS

The advantage with DGPS is that two or more receivers can share the same error sources. This enable the solution system almost completely remove them. In the

case of a moving baseline situation the GNSS system assumes that the rover is close enough to the base station such that they shear the same atmospheric conditions. If this assumption holds the system should be able to almost remove the error caused by atmospheric delay.

3.5.3 RTK GPS

RTK GPS scarifies correctness in order to give a position estimate in real-time. ADD HERE RESEARCH IN THE RTK FIELD. Real time position is critical for a autonomous system to navigate to a given position.

PASS PÅ FOR Å GÅ INNOM SYSTEM SPEC. KUN TEORI OM RTK GPS Dynamic system can be solved in kinematic mode, or with a moving baseline. In kinematic mode the rover is allowed to move, but the base station is assumed stationary with a known position. In the case of a moving baseline both the rover and base station is allowed to move. The position of the base station is calculated in single mode. Without a known position of the base station the global position of the rover can never be better then if calculated in single mode. However the relative position of the rover from the base station is calculate accurately. There for from a local control systems per Need to write about baseline restrictions. In the case of this thesis is a moving baseline relevant.

Trade off between getting the position fast, and getting it right

Moving baseline restrictions. The base stations position is calculated with in single mode. The error in position to the base station is inherit by the rover. Source of error.

Chapter 4

System Components

This chapter contains a brief description of the system that has been used.

4.1 Hardware

This section outline the physical components in the x8 and the base station.

4.1.1 UAV

The Skywalker X8, see 4.1, is a fixed wing UAV that is moulded out of Expanded PolyOlefin (EPO), which makes it a cheap and robust platform for prototype testing. The large space within the fuselage makes it ideal for experimental payload.

Figure 4.1: X8 Skywalker from Skywalker Technologies, Picture from www.campilot.tv/blog/win-x8

The X8 has a wingspan of 2120mm which allow for All Up Weight (AUW) of 3500g. More specification about the X8 can be found at [hob]. The components that is used in the X8 at the UAVLAB is given in table 4.1.

TX	Spektrum DX7s
RX	Spektrum Remote Receiver SPM9645
Servo	Hitec HS-5085MG
Motor	Hacker A40-12S V2 14-Pole
ESC	Master Spin 66 Pro
Propeller	Aeronaut 13x8 folding propeller
Battery	1 Zippy Compact 4S 5000 mAh and 1 Haiyin 800 mAh
Autopilot	3D Robotics Pixhawk w/3DR uBlox GPS with Compass kit and airspeed sensor
Telemetry link	3D Robotics radio (433MHz)

Table 4.1: Components in the X8 at the UAVLAB

4.1.2 Embedded Computer

The embedded computer in the UAV is a BeagleBone Black List why this embedded computer is chosen.

Demands: Must run LSTS in real time, calculate rtkgps wight and size number of output pins.

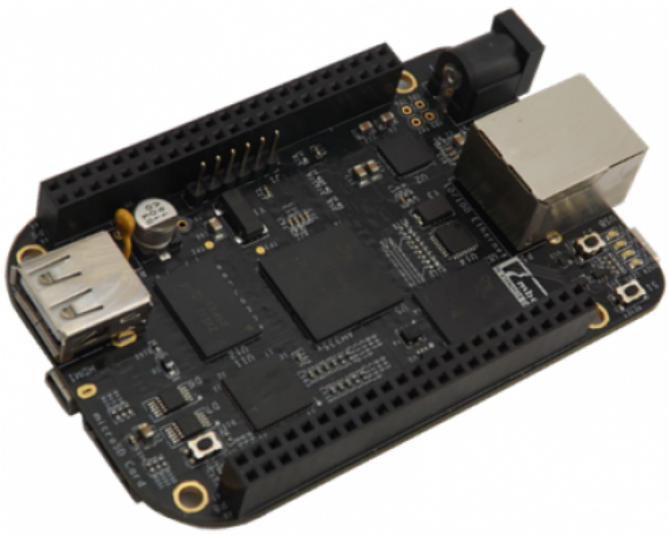

Figure 4.2: BeagleBone Black element 14, Picture from http://www.element14.com

4.1.3 GNSS receiver

Write about the Ublox LEA M8T gnss receiver. Also include that it support sending GPS and GLONASS data at the same time. Need to be configured. A receiver were prepare and mounted in the x8.

Figure 4.3: Ublox LEA M8T, Picture from http://www.csgshop.com

4.2 Software

This section contain the different software that is used in the x8 system. The following sections contain the operating system that runs on the base station and rover, the middleware used to connect the different tasks, the messages protocol, the missionplaner program and rtklib which will have the main focus.

4.2.1 GLUED

Glued is a minimal Linux distribution developed by LSTS, and design with embedded system in mind. It is platform independent, easy to configure and contain only the necessary packages to run on a embedded system. This makes GLUED a light and fast distribution. GLUED is configured through a single configuration file that which can be created for a specific system.

4.2.2 Dune

Dune is a runtime environment for unmanned systems on-board software created by LSTS (Underwater Systems and Technology Laboratory) in Porto, Portugal. The environment type is called a middleware, which is seeing increase usage in unmanned systems. Can refer to ROS or MOOS middleware.

Dune works by setting up individual task that can dispatch and subscribe to different IMC messages. The IMC messages will be explained in 4.2.3 A type of middleware. write how to link rtklib with dune Refer to the dune wiki page

4.2.3 IMC

The IMC (write gls here when fixed) protocol that is designed and implemented by LSTS (gls) that is build to interconnect systems of vehicles,sensors and human operators. The protocol is a messages-oriented protocol that enable exchange of real-time information about the environment and updated objectives, such that the participant in the communication can pursue a common goal cooperatively. IMC has a standard way of dispatching and consuming messages, which abstracts hardware

Other feature: Not assuming a specific software architecture, can generate native support for different programming languages and/or computer architectures. Used in network nodes, inter-process and inter thread communication

4.2.4 Netpus

4.2.5 Piksi

Piksi, see figure 4.4, is a low cost, high performance GPS receiver with RTK-GPS functionality with capabllity for centimeter level relative positioning accuracy developed by Swift Navigation. Piksi is ideal for autonomous vehicles because of its small form factor, fast position solution update rate and low power consumption. More detailed information about Piksi can be found in [Pik]

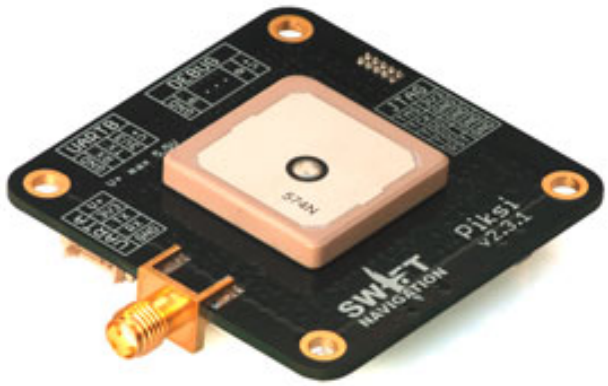

Figure 4.4: Piksi, Picture from www.swiftnav.com

4.2.6 RTKLIB

Rtklib is a open source program package for standard and precise positioning with GNSS developed by T. Takasu. Rtklib can use raw GNSS data to estimate the position of the rover. Rtklib can be configured to give a position solution in real time in differential mode. Figure 6.8 shows how rtklib can be used in a RTKGNSS mode. The two main moduels here is str2str and rtkrcv. Both will be explalined more closely in the following sections. More information about rtklib can be found in [Rtk].

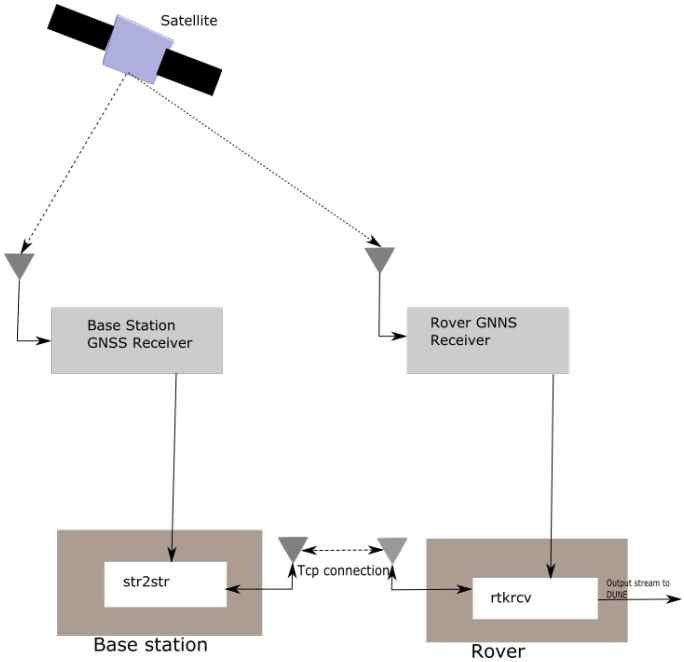

Figure 4.5: The communication structure of rtklib

rtkrcv

Rtkrcv is the app program that calculate the position of the rover. Rtkrcv can be configured to have two output streams. The structure of the output stream is given the rtklib manual, however there has been done some alteration in the structure of the output. It's desired in a autonumos landing system that have that velocity solution. This is was not provided in the newest version of rtklib, and therefore the source code was altered to send out the velocity data. A quick note here is that it is only available in the output solution is in a enu format.

When set configured as a differential GPS rtkrcv uses the LAMBDA method to resolve the integer ambiguity. The solution is considered fixed if the ration between the best estimate and the second best estimate is above a certain threshold. The solution body is given is table The position solution is calculated with a Extended Kalman Filter, that can have different structure depending on how rtkrcv is configured

str2str

Str2str is the app program that retrieve the ublox signal from the gps and sends over tcp to the rtkrcv app. The str2str is setup to either send RMTC 3 messages, or whatever is send in from the GPS. Since the str2str do not support to send ublox

Header	**Content**
1 Time	The epoch time of the solution indicate the true receiver signal reception time. Can have the following format: yyyy/mm/dd HH:MM:SS.SSS: Calender time in GPST, UTC or JST. WWWW SSSSSSS.SSS: GPS week and TOW in seconds
2 Receiver Position	The rover receive antenna position
3 Quality flag (Q)	The flag which indicates the solution quality. 1:Fixed 2:Float 5:Single
4 Number of valid satellites (ns)	The number of valid satellites for solution estimation.
5 Standard deviation	The estimated standard deviation of the solution assuming a priori error model and error parameters by the positioning options
6 Age of differential	The time difference between the observation data epochs of the rover receiver and base station in second.
7 Ratio factor	The ratio factor of "ratio-test" for standard integer ambiguity validation strategy
8 Receiver velocity	The velocity of the rover. Given only when output is in enu format

Table 4.2: Table 1.

signal directly. How to write that the user should not specify the input format or the output format.

Chapter 5

Implementation

The chapter will explain how the different software are connected and what information is sent between them. All the configuration and start up commands can be found in the appendix.

5.1 Software implementation

The software implementation consist mainly of rtklib and dune. Piksi is also used in the experiment, however it's rtklib and the Dune task RTKGPS that will be discussed thoroughly.

5.1.1 Rtklib

Rtklib is sepperated into the base station and the rover. The base station implementation runs the app str2str were it communicate with the ublox over a uart cable, and start up a tcp server.

The rover uses the rtkrcv app from rtklib to estimate the position of the rover. Rtkrcv connect itself as a tcp client to the tcp server that str2str create. Rtkrcv is configured in a moving baseline configuration to simulate the behavior that is expected during a landing on a ship. (Referer her til teori kap om rtklib)

Rtkrcv output the solution data over a virtual uart connection that is created by a program called socat. The output structure is given i (se rtklib com kap)

5.1.2 Rtkgps

Rtkgps is a task in Dune that takes the output from rtklib and create a rtkfix imc message that is dispatch into the Dune/Neptus network. Rtkgps consists of two parts.

One reads the message from the virtual uart, and the other create and dispatch the rtkfix imc message.

5.1.3 Neptus

Neptus is used as interface for the human operator, and for after mission reviews. In this thesis it's only used for after mission reviews, however it has the cabillity to give a landing command to the rover (henvis her til frølich) PLASSERES I APPENDIX

str2str configuration

The system has to instances of rtklib. The base station uses the str2str and the x8 uses rtkrcv. str2str is configured to receive raw data from the ublox at a frequency of 10 Hz. The connection between str2str and the GNSS receiver is a uart cable that is configured with a baudrate at 115200. The program starts a tcp serve were rtkrcv becommes a client.

Very short about how Glued is used: Start up,

About rtklib: What does rtklib do in the system, were do it run, what instances is used, how do it connect to dune, what is the output message

About dune: What task are used to communicate with rtklib, how do they communicate, what is the output of the given task

About Neptus: What do neptus do, how is it used in the system, how will it be used in a automatic landing senario.

About Ardupilot: Very short on what Ardupilot do in the system

All in one section unless something need more space. Write here how rtklib is configured and how the system is connected Write here how Dune receive data from rtklib and what task is used. Include also what imc message is involeved in the task Write here how neptus is configured for the test, and how it is used How Glued is configured to run rtklib and Dune

5.2 Hardware implementation

This section contain how all the physical components are connected at both the rover and the base station. Include also how everything was prepared.

About Beaglebone: What runs on the beaglebone, connections, devices, what is it place in the system

About Ublox: Explain the ublox from a system perspective, how it's connected

Pixhawk: What do it do in the system:

Piksi: Same as ublox

The X8: How do it fit in the system

The base station: Same as x8

Antennas:

Wifi router

Chapter 6

Experimental testing

This chapter contain the result from the test that were performed. The goal with these test was to evaulate the ublox receiver againgst the pixi receiver, and to get a impresion on the accuracy to the rtklib solution with ublox. The comparison test was performed with the pixi and ublox connected to the same antenna at both the rover and the base station. Then the deviation in the position estimate can only come from the receivers. The accuracy test of the ublox was tested by performing the same manoeuvre several times.

All position and velocity data is given in the NED frame. Can mension the gnss radar used to find good gps windows.

6.1 Physical testing

The physical experiment were done in two parts. During the first part the x8 was carried around on a open field. The goal with this experiment was to log data from rtklib and piksi, and then compare how they deviated from each other. The second part of the physical experiment was flying (SKRIV NÅR FLYVING ER GJENNNOMFØRT)

6.1.1 GPS test

In this part two sessions of navigation data will be presented. Both sessions were performed on the same day, which was cloudless and at a time with low DOP. The raw data from the Ublox receiver was post processed with rtklib, and is should be more accurate than it's real time counter part. Therefore a error was defined as:

$$e(t) = p_r(t) - p_p(t) \tag{6.1}$$

where $p_r(t)$ and $p_p(t)$ is defined as the position solution from the real time system and the position solution from the post processed solution respectfully. It should be noted that in order to compare the different time-series the position data was synchronized with each other. From the error the cumulative mean error and cumulative standard deviation for the error was calculated using the matlab function "mean" and "std".

Figure 6.1 shows a North East plot of how the walk was. The plot contain only the fixed solution from both the piksi and rtklib.

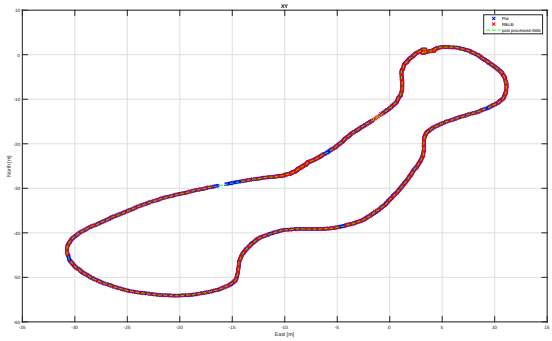

Figure 6.1: A xy plot of the piksi,rtklib real time solution and the rtklib post processed solution

Figure 6.2 shows the down position, as well as how the integer ambiguity solution was during the experiment. As seen in the figure both the Piksi and Rtklib manage to keep there fixed solution. The position solution from both the Piksi and Rtklib agrees with the post processed solution.

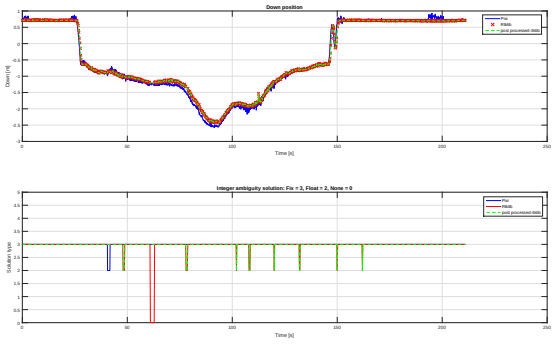

Figure 6.2: The communication structure of rtklib

As seen in figure 6.1 the difference between the different solution appear to be small, which is comfirmed in the error plot shown in figure 6.3 and 6.4

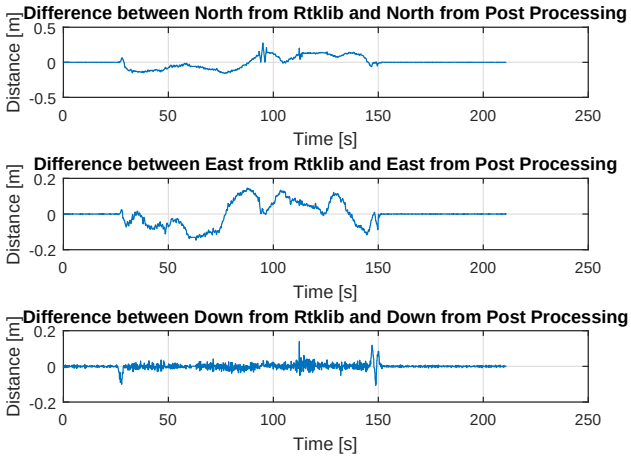

Figure 6.3: The difference between rtklib real time and post processed solution

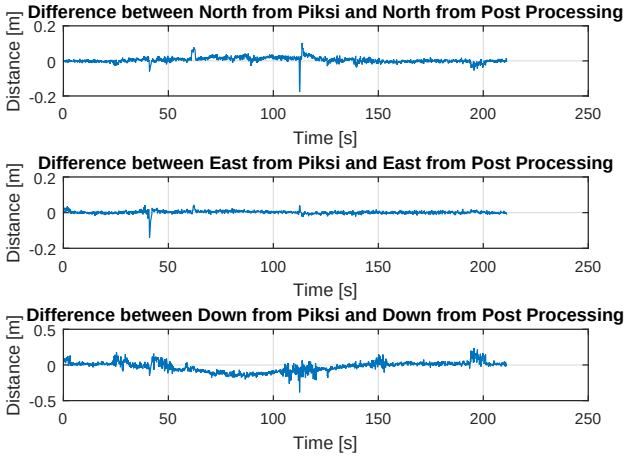

Figure 6.4: The communication structure of rtklib

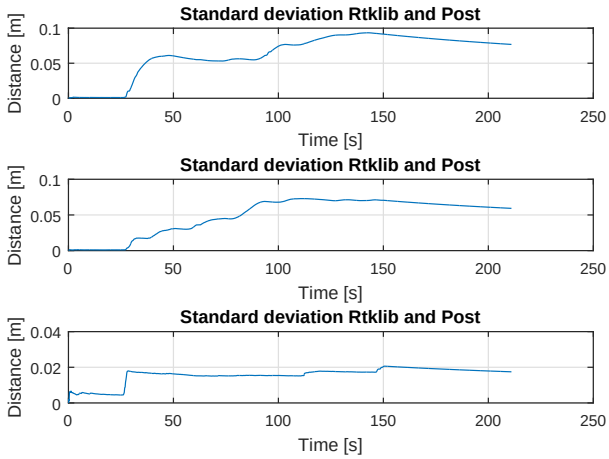

Figure 6.5: Standard deviation of the difference between rtklib real time and post processed solution

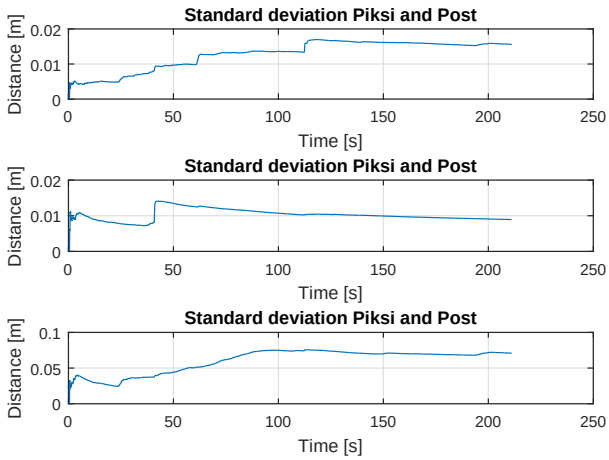

Figure 6.6: Standard deviation of the difference between piksi real time and rtklib post processed solution

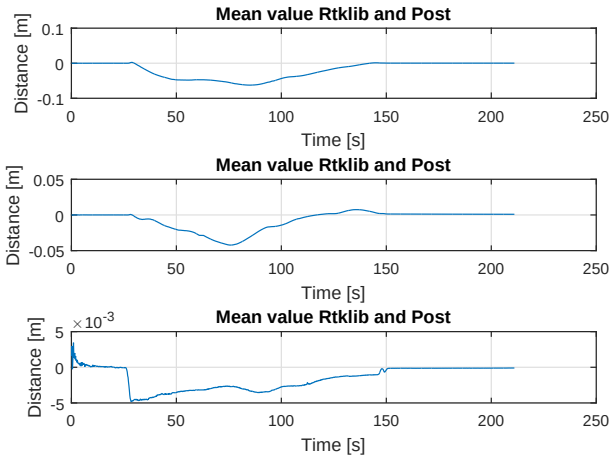

Figure 6.7: The mean difference between the rtklib real time and post processed solution

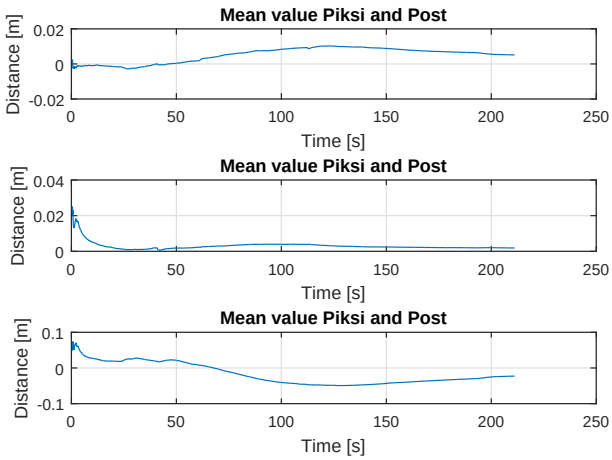

Figure 6.8: The mean difference between the piksi real time and the rtklib post processed solution

The velocity data is shown in figure 6.14. It appears that rtklib has a delayed solution, that might be because of problem with the TOW that was identified in rtklib.

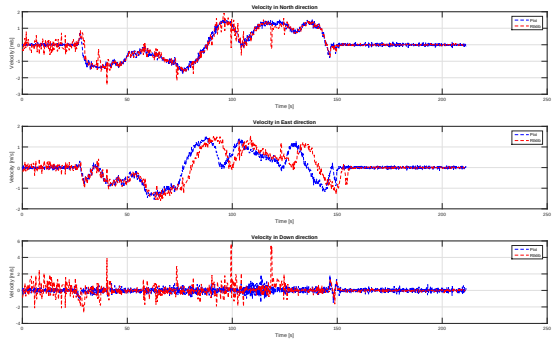

Figure 6.9: Velocity data from the piksi and rtklib real time solution

The problem with the TOW is shown more clearly in figure 6.10. The figure display the time difference between each output. The first part is an altered version of the TOW from rtklib, while the other is the timestamp of the same output given by Dune. The output from rtklib gave only seconds, and not a millisecond value. To correct for this error the TOW value from rtklib altered to fill the gap between each second. The alteration was done by counting the number of elements with the same TOW value, and then equally spread them from between the TOW value to the next TOW value.

The timestamp given by Dune indicate the delay a output message from rtklib might experience before it's available for consumption. Both the timestamp and TOW value indicate that rtklib has a mean output of 5 Hz.

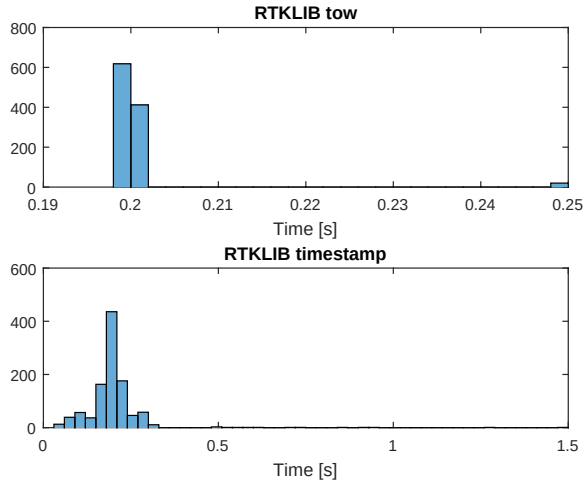

Figure 6.10: The time between time samples from rtklib

Figure 6.11 shows the time difference between each output sample. The first plot shows the difference between each TOW value from Piksi, and the other the difference between each time stamp given by Dune. Both plots indicate that Piksi is able to have a output frequency at 10 Hz.

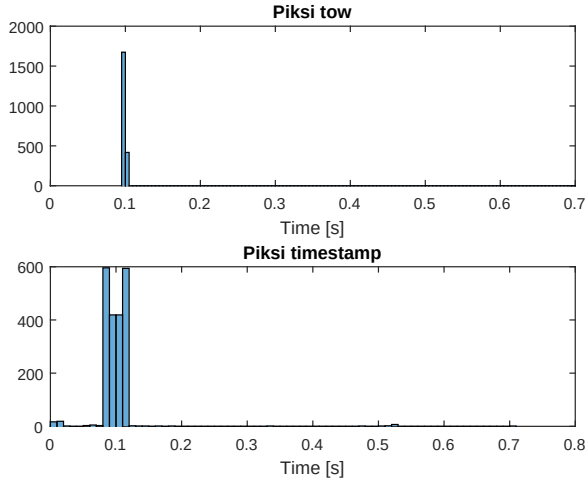

Figure 6.11: The time between time samples from rtklib

The different receiver used in Rtklib and Piksi was not able to track the same satellites at all time. Figure 6.12 shows that the Ublox LEA M8T receiver connected to Rtklib managed to track more satellite then the receiver used in Piksi. Figure

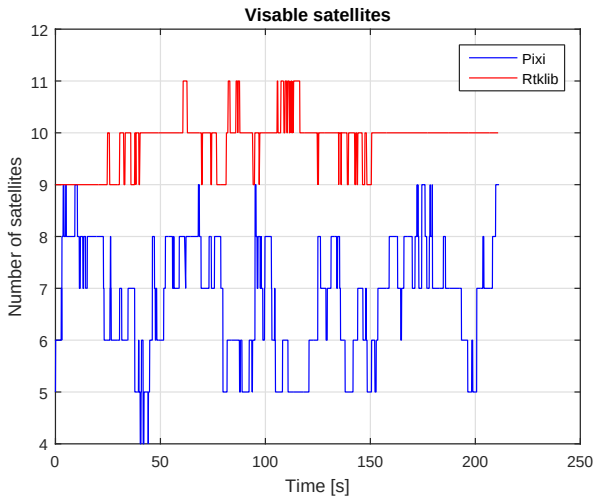

Figure 6.12: Visable statellite for the piksi and rtklib

The position estimate of the rover is a relative position in reference to the base station. Due to the fact that the position of the base station is calculated with a single receiver with one frequency, there will be introduced a bias in the position estimate. Figure 6.13 shows the North, East position of the the first walk. The true position is exactly the same, but in the figure it appear that the distance is approximately $5cm$. The distance from the base station to the estimated start and stop position was calculated to be $3.29m$ and 3.26 respectfully. The measured distance was approximately $3.3m$. That gives an initial error off $0.04m$. This gives an accuracy level at centimeter level at stationary condition.

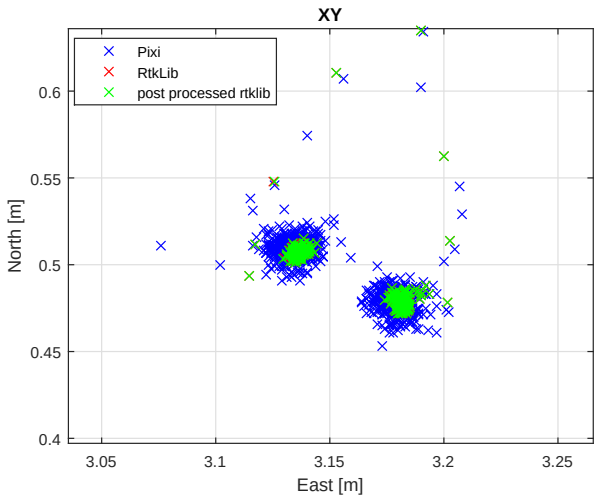

Figure 6.13: Visable statellite for the piksi and rtklib

Given that the error given in figure 6.4 and 6.3 is never has a greater absolute value then $0.2m$ it's possible to assume that the true error will be bellow $1m$ which was given as an evasion criterion in the MSc thesis by [Frølich, 2015], if the RTK-GPS system has a fixed integer solution.

6.1.2 Fly test

A flight test with the UAV was performed at Udduvoll. Because of bad weather the were only performed one flight, and before the flight started only the Rtklib had a fixed solution. That is why in this part only performance from the Rtklib is considered, as a fixed solution is a must for a automatic landing system.

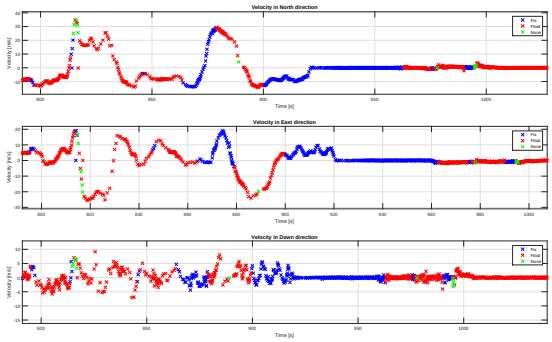

Figure 6.14: Velocity data from the piksi and rtklib real time solution

During the flight test the integer ambiguity solution were more float then fixed as seen in figure 6.15, which affected the measurement. The main reason for this behavior is because of the number of valid satellite the receiver can track experience large variation, as seen i figure 6.18.

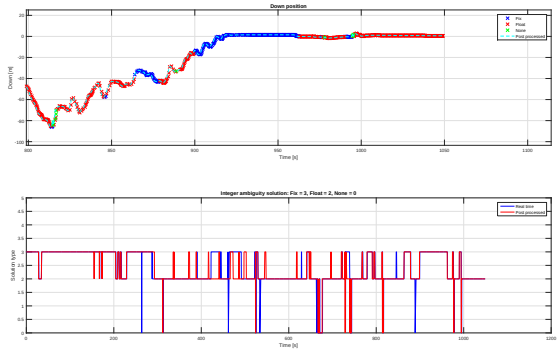

Figure 6.15: Velocity data from the piksi and rtklib real time solution

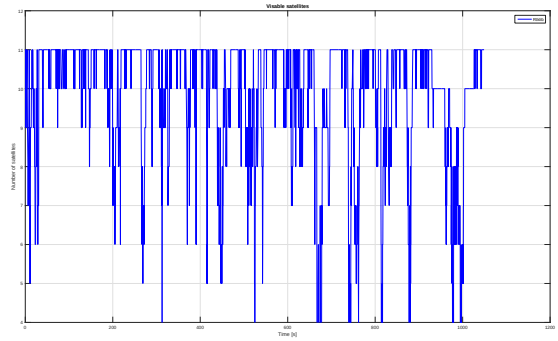

Figure 6.16: Velocity data from the piksi and rtklib real time solution

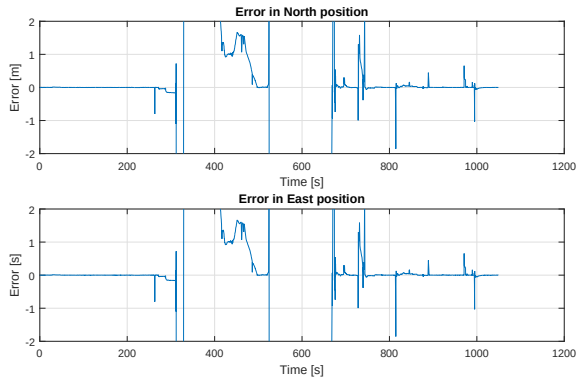

Figure 6.17: Velocity data from the piksi and rtklib real time solution

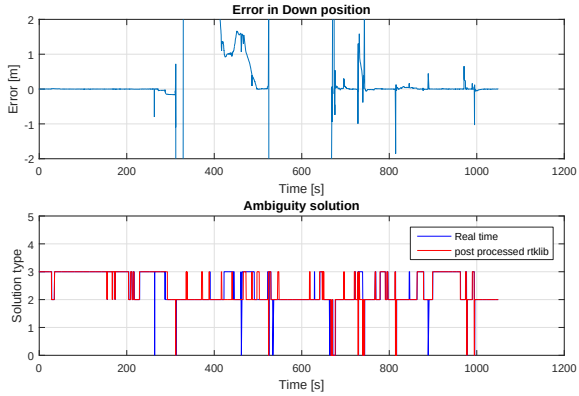

Figure 6.18: Velocity data from the piksi and rtklib real time solution

Chapter 7

Conclusion and Discussion

7.1 Further work

Setup the reciver to use both gps and glonass raw data.

References

Piksi Datasheet ver. 2.3.1.

RTKLIB ver. 2.4.2 Manual.

Skywalker x8. http://www.hobbyking.com/hobbyking/store/___27132__Skywalker_X_8_FPV_UAV_Flying_Wing_2120mm.html. Accessed: 09.12.2015.

Geoffrey Blewitt. Carrier phase ambiguity resolution for the global positioning system applied to geodetic baselines uo to 2000 km. *Journal of Geophysical Research*, 94(B8):10,187–10,203, 1989.

Marcus Frølich. Automatic ship landing system for fixed-wing uav. Master's thesis, Norwegian University of Science and Technology, NTNU, 2015.

INSITU. Scaneagle system. http://www.insitu.com/systems/scaneagle. Accessed: 07.12.2015.

H Jin Kim, Mingu Kim, Hyon Lim, Chulwoo Park, Seungho Yoon, Daewon Lee, Hyunjin Choi, Gyeongtaek Oh, Jongho Park, and Youdan Kim. Fully autonomous vision-based net-recovery landing system for a fixed-wing uav. *Mechatronics, IEEE/ASME Transactions on*, 18(4):1320–1333, 2013.

Pratap Misra and Per Enge. *Global Positioning System Signals, Measurements and Performance Revised Second Edition*. Ganga-Jamuna Press, 2011.

Robert Skulstad and Christoffer Lie Syversen. Low-cost instrumentation system for recovery of fixed-wing uav in a net. Master's thesis, Norwegian University of Science and Technology, NTNU, 2014.

Samuel Jacobus Adriaan Smit. Autonomous landing of a fixed-wing unmanned aerial vehicle using differential gps. Master's thesis, Stellenbosch: Stellenbosch University, 2013.

Bjørn Amstrup Spockeli. Integration of rtk gps and imu for accurate uav positioning. Master's thesis, Norwegian University of Science and Technology, NTNU, 2015.

W. Stempfhuber. 3d-rtk capability of single gnss receivers. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2013.

W. Stempfhuber and M. Buchholz. A precise, low-cost rtk gnss system for uav applications. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2011.

P.J. de Jonge Teunissen, P.J.G and C.C.J.M. Tiberius. The lambda-method for fast gps surveying. Presented at the International Symposium "GPS Technology Applications" Bucharest,Romania, 1995.

P.J.G. Teunissen. A new method for fast carrier phase ambiguity estimation. *IEEE*, pages 562–573, 1994.

P.J.G. Teunissen. The least-squares ambiguity decorrelation adjustment: a method for fast gps integer ambiguity estimation. *Journal of Geodesy*, 70:65–82, 1995.

Bjørnar Vik. *Integrated Satellite and Inertial Navigation Systems.* Department of Engineering Cybernetics, NTNU, 2014.

Paul Williams and Michael Crump. Intelligent landing system for landing uavs at unsurveyed airfields. In *28th International Congress of the Aeronautical Sciences*, 2012.