NTNU – Trondheim
Norwegian University of
Science and Technology

Low-cost instrumentation system for recovery of fixed-wing UAV in a net

Robert Skulstad
Christoffer Lie Syversen

Master of Science in Engineering Cybernetics (2 year))
Submission date: June 2014
Supervisor: Tor Arne Johansen, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Task description

The purpose of this thesis is to implement a Real-Time Kinematic Global Positioning System (RTK-GPS) for accurate positioning of an Unmanned Aerial Vehicle (UAV) through its final approach to a landing target. Horizontal and vertical controllers, which can accommodate this new position reference, shall also be designed to guide the aircraft through its final approach to target. The below item-list summarizes the main tasks of the project.

- Obtain stable RTK-GPS solution for relative positioning of a UAV
- Design new longitudinal and lateral guidance controllers for final approach
- Verification of guidance controllers are to be performed in a Hardware-In-Loop environment
- Validate the accuracy of the procedure and the system through experimental testing

The X8 Flying wing will be used as a platform for experimental testing.

Abstract

This thesis considers the feasibility of using an inexpensive Real-Time Kinematic Global Positioning System (RTK-GPS) unit to provide position data to support a high precision landing of an Unmanned Aerial Vehicle (UAV) into a net. Both dynamic- and stationary-net platforms were considered and custom guidance controllers were designed for both applications. Fully functioning, the system is intended to remove the need of a pilot when operating UAVs from confined platforms.

The ArduPilot Mega (APM) was the selected micro-controller for providing the approach guidance. It already supported autonomous waypoint flight, based on absolute GPS-based position reference. Accommodation of the new RTK-GPS distance vector, a measure of the relative distance between base station and UAV, in custom guidance algorithms constituted the main bulk of work. Real-Time Kinematic Library (RTKLIB), an open source RTK software, was used for calculation of the relative distance vector, also known as baseline vector.

The custom guidance controllers utilized model-less controller logic, and the longitudinal and lateral axes were completely uncoupled. The longitudinal controller, governing throttle- and pitch setpoints, was comprised of two Proportional, Integral and Derivtive (PID) controllers. The lateral controller, a non-linear lookahead-based controller, produced a setpoint for the desired aircraft roll. Other main topics of the thesis were; HIL-simulation based on the software XPlane and MissionPlanner and real-life testing with the model aircraft X8 Skywalker.

HIL-simulations confirmed the feasibility of the guidance controllers, while real-life testing showed sufficient precision of the entire system, baseline vector and custom guidance controllers combined, for retrieval of a UAV in a 3 m high by 5 m wide net.

Sammendrag

Denne avhandlingen vurderer muligheten for å bruke en lav-kostnads Real-Time Kinematic Global Positioning System (RTK-GPS) enhet for å støtte en høypresisjonslanding av en Ubemannet Flygende Drone (UFD) i et nett. Både dynamiske- og stasjonære plattformer ble vurdert for plassering av nettet og den autonome navigeringen av dronen ble utført av kontrollere laget spesielt til denne applikasjonen. Fullt fungerende er systemets hensikt å fjerne behovet for en pilot når en betjener en UFD på steder hvor en ikke kan lande på en flystripe.

ArduPilot Mega (APM) var den valgte mikrokontrolleren for å kjøre programvaren som sørger for navigering ved innflyging mot målet. Den støtter allerede autonom flyging, men da basert på geodetiske koordinater som posisjonsreferanse. Innpassing av en mer nøyaktig posisjonsreferanse, den relative RTK-GPS avstandsvektoren, sto for hoveddelen av arbeidet. Den er et mål på avstanden mellom basestatjonen og den ubemannede dronen. Real-Time Kinematic Library (RTKLIB), en fritt tilgjengelig programvare, ble benyttet for å produsere posisjonsvektoren som ble brukt for navigering.

De tilpassede navigeringskontrollerne benyttet ingen matematisk modell av dronen og den langsgående og tverrgående aksen er fullstendig dekoblet. Den langsgående kontrolleren, som styrer pådrag til motoren samt settpunkt til pitch basert på høyde, besto av to Proporsjonal, Integral og Derivat (PID) kontrollere. Den tverrgående kontrolleren, en ulineær navigasjonskontroller, produserte settpunkt for ønsket rullvinkel. Andre hovedtema var; Hardware-In-Loop (HIL) -simuleringer basert på programvaren XPlane og MissionPlanner og eksperimentell testing med modellflyet X8 Skywalker som plattform.

HIL-simuleringer bekreftet at navigasjonskontrollernes funksjon var korrekt og eksperimentell testing viste tilstrekkelig presisjon av hele systemet for å kunne lande en UFD i et 3 m høyt og 5 m bredt nett.

Acknowledgments

This thesis is submitted as partial fulfillment of the requirements for the degree MSc. at the Norwegian University of Science and Technology (NTNU).

We would like to thank Professor Tor Arne Johansen, and co-supervisor Mariann Merz for their support and guidance for the duration of this project. We would also like to thank the UAV-pilots, Carl Erik Stephansen and Lars Semb, for their patience and help during experimental testing. Moreover, a thank you to Jakob Mahler Hansen for valuable feedback along the way.

Trondheim, June 08, 2014

Christoffer Lie Syversen and Robert Skulstad

Contents

List of Figures

List of Tables

Introduction

In the pursuit of fully autonomous, low-cost Unmanned Aerial Vehicle (UAV) systems, precision landing is perhaps the greatest remaining obstacle. As landing autonomously requires knowledge of the UAV's position relative to the landing target's position, multiple challenges present themselves when trying to implement this relative distance in the control scheme. As will be pointed out later in this section there are two main ways of obtaining this relative distance; by image processing of images taken from a forward mounted camera, and by Differential Global Positioning System (DGPS). This report considers only the latter approach for both moving base station (e.g. on a moving ship) and stationary base station.

Finding the relative positioning vector is a demanding task in itself. Taking into account that the UAV is a highly dynamic platform adds to the difficulty by requiring high update rates of the relative position, especially during final approach (FA) to target. Cost is always a factor and puts a limit on several operational capabilities. Low cost often results in longer time to fix (single frequency receivers receiving only Global Positioning System (GPS) satellites), limited operational range due to the quality of radio communication hardware (the radio hardware is also subject to certain laws that puts additional limitations on their use) and a relatively low relative position update rate.

For vision based landing in a static recovery net, [17] develops a scheme where the UAV follows a path of waypoints with the aid of ordinary GPS positioning. As the UAV reaches its FA point, about 50 m away from the target, corrections in order to adjust aircraft attitude are made by interpreting an image of the view in the forward direction. A similar approach with a forward mounted camera and vision-based landing is performed by [16]. The recovery net was substituted with a red inflated dome. Larger and more expensive systems have achieved takeoff and landing using DGPS [25]. Separating autonomous vision-based landing and DGPS-based landing is the need for radio communication with a ground station for the latter approach.

In this paper, some methods for retrieval of a fixed-wing UAV in a net, both moving and static, are discussed and tested by simulation and actual flight. The topic of relative position between UAV and landing target is covered through theory and simulation. Precise target-relative positioning is obtained through RTK-GPS. Hence, topics such as; reliability of Real-Time Kinematic GPS (RTK-GPS), precision of controller, implementation aspects of autonomous UAV landing, considerations when landing in a net etc., are discussed. Two relatively inexpensive single-frequency GPS receivers are used in all the tests described in this thesis. One for mounting on the Ground Control Station (GCS) and

one for the UAV. Real-Time Kinematic LIBrary (RTKLIB), an open source RTK-GPS solution software, was run on a data processing board intended to fit in the compartment bay of a fixed-wing UAV and communicate with the on-board APM used for control of the UAV. Using APM interfaced with necessary software enabled Hardware-In-Loop (HIL) testing via XPlane 10, an aircraft simulator, and thereby facilitated verification of the suggested landing procedures.

Abbreviations

GPS	Global Positioning System
GNSS	Global Navigation Satellite System
RTK	Real-Time Kinematics
DGPS	Differential GPS
RTKLIB	Real-Time Kinematics Library
APM	ArduPilot Mega
UAV	Unmanned Aerial Vehicle
HIL	Hardware In the Loop
PRN	Pseudo Random Noise
C/A	Coarse/Acquisition
EKF	Extended Kalman Filter
RMS	Root Mean Square
NMEA	National Marine Electronics Association
UTC	Universal Time Coordinated
ECEF	Earth-Centered Earth-Fixed
ENU	East-North-Up
NED	North-East-Down
WGS-84	World Geodetic System 1984
PID	Proportional-Integral-Derivative
TECS	Total Energy Control System
LT	Landing Target
CP	CheckPoint
FA	Final Approach
USB	Universal Serial Bus
UART	Universal Asynchronous Receiver/Transmitter
RAM	Random-Access-Memory
GHz	Giga Hertz
WiFi	Wireless Fidelity
LAN	Local Area Network
SMA	SubMiniature version A
PoE	Power over Ethernet
Mbps	Mega bit-per-second
dBm	desiBel-milliwatts
EEPROM	Electrically Erasable Programmable Read-Only Memory
FPS	Frames-Per-Second
SDK	Software Development Kit
MP	MissionPlanner
UDP	User Datagram Protocol
GUI	Graphical User Interface
GCS	Ground Control Station
MAVLink	Micro Air Vehicle Link
AR	Ambuguity Resolution
AP	Application Programming
TCP	Transmission Control Protocol

DOP	Dilution of Precision
m	metre
mm	millimetre
TX	Transmit
AoD	Age of Differential

Chapter 1

Background

In this chapter theory substantiating the algorithms and controllers used throughout the thesis will be presented. A brief introduction of the test platform will also be included.

1.1 Test platform

The UAV used during experimental testing was an off-the-shelf model named X8 Skywalker, a flying wing. This was chosen for its low cost and robustness. Two long range Wireless Fidelity (WiFi) tranceivers were used to transmit telemetry and RTK-GPS data. As a main payload data-processor, the Pandaboard was chosen. Two identical low-cost GPS receivers receiving only the L1 frequency sampled the GPS signal. APM 2.6 performs the autopilot duties in both real life experimental testing, as well as HIL simulation.

For HIL simulation XPlane was the chosen engine, and MaxiSwift [26] was the flying wing model used in simulation.

1.2 GPS

NAVigation Satellite Timing And Ranging GPS (NAVSTAR GPS) is a satellite navigation system based on the following basic principles [41]:

- The distance travelled by a signal through space is determined by its velocity and the time of travel.
- An electromagnetic signal travels through the atmosphere at the speed of light, which is roughly 300000 km per second.

GPS satellites transmit two signals, L1 and L2, which have their separate carrier-phase frequency (signal onto which the data is modulated). The structure of these signals is given below by:

$$L1(t) = A_1 p(t) d(t) \cos(f_1 t) + A_1 c(t) d(t) \sin(f_1 t) \tag{1.2.1}$$
$$L2(t) = A_2 p(t) d(t) \cos(f_2 t) \tag{1.2.2}$$

where A_1 and A_2 are signal amplitudes and $d(t)$ is the data message containing satellite-specific information. The time-varying functions $c(t)$ and $p(t)$ represent pseudo random sequences and are referred to as Precision-code (P-code) and Coarse/Acquisition (C/A) code, respectively [45]. The carrier frequencies ,f_1 and f_2, are given by:

$$f_1 = 1575.42\ MHz$$
$$f_2 = 1227.60\ MHz$$

Included in the L1 signal is the navigation message, which is transmitted at 50 bits per second and modulated onto the Pseudo Random Noise (PRN) generated by each satellite. Each satellite has a unique PRN and the P- and C/A-codes are known by the receiver. Using correlation of the received PRN and the receiver-generated PRN enables the receiver to distinguish between satellites transmitting on the same carrier frequency. Information contained in the navigation message may be summarized as follows:

- Satellite time; time correction for satellite clock and synchronization signal
- Ephemeris; describes the precise orbit of the satellite (specific to each satellite)
- Almanac; approximate orbital data for all satellites in system
- Information regarding the "health" of the satellite

When determining a range from a satellite with a known location in space to a receiver located on earth the true distance is the speed of the signal multiplied by the transmission time. As clock errors come into play the calculated range is not equal to the true range and is therefore named pseudorange in the literature and given by the below equation [45]

$$\rho = c\left[(T_u + t_u) - (T_s + t_s)\right] \tag{1.2.3}$$

where T_u is the time the signal reached the receiver and T_s is the time at which the satellite transmitted the signal. t_s and t_u are offset in satellite clock and receiver clock respectively. The presence of clock errors requires the receiver to obtain signals from four satellites in order to determine its position and velocity in space. Due to the orbital layout of the around 30 satellites contained in the GPS system this is guaranteed at all times anywhere on earth [41].

Single point positioning involves satellite positioning using only one receiver. A set of pseudoranges for a particular constellation of satellites may then be described by [45]

$$\rho_i = \sqrt{(\mathbf{x}_{s_i} - \mathbf{x})^T(\mathbf{x}_{s_i} - \mathbf{x})} + c\tau^* + \epsilon_{\rho_i} \tag{1.2.4}$$

In the above equation $i = 1, 2, ..., n$ where n is the total number of satellites currently in view, $\mathbf{x}_{s_i}, \mathbf{x} \in \mathbb{R}^3$ and contains the ECEF distance vector for each satellite and receiver, c is the speed of light, τ^* is the receiver clock bias and ϵ_{ρ_i} is the combined error from various sources (Ionospheric, tropospheric error, etc). The set of equations are nonlinear and must be solved for τ^* and ρ_i. Solving this set of equations is generally performed using linearization and a recursive Least-Squares algorithm or an Extended Kalman Filter (EKF). See chapter 3.4 of [45] or chapter 6.2 of [41] for detailed descriptions of algorithms. An estimate of the total Root-Mean-Square (RMS) error of a single point positioning scheme is roughly 4 m [41].

1.2.1 Typical sources of error

Typical sources of error associated with satellite navigation are discussed below

1.2.1.1 Multipath

When a direct signal from a satellite is reflected off of a structure or the ground before reaching the GPS receiver. This causes the signal to display an increase in propagation time compared to the straight line path between the satellite and receiver and thus results in an error in pseudorange [9]. In order to minimize the effects of multipath signals [9] suggests a ground plane in the form of a metal plate or a choke ring with the receiver antenna mounted in the middle. Other approaches may involve rejecting use of satellites that are below a certain elevation angle.

1.2.1.2 Ionospheric error

The Ionosphere is a layer of the atmosphere stretching from an altitude of approximately 60 km to an altitude of about 1000 km. This part of the atmosphere contains molecules which, upon exposure to ultraviolet rays from the sun, are ionized. A bi-product of this process is the release of free electrons that influences the propagation of electromagnetic signals (see [41] and [45]). The assumption of a constant propagation velocity of electromagnetic signals is no longer valid. To reduce the error in propagation time a model of the entire vertical ionospheric refraction may be applied. [45] suggests the Klobuchar model as the coefficients of the model is broadcast by the navigation message mentioned earlier.

1.2.1.3 Tropospheric error

The troposphere encompasses the earth from sea level to an altitude of about 15 km. Temperature, pressure and humidity are factors that contributes to the tropospheric error. A simple model made up of these variables can be used to mitigate these negative effects [41].

1.2.1.4 Satellite clock error

Caused by offsets in the four atomic clocks included on each satellite [41].

1.2.1.5 Ephemeris data

Precision of the calculated position of the satellite in orbit at the time of transmission [41].

1.2.1.6 Dilution of precision

The precision of the position solution is affected by the geometry of the constellation of satellites from which the receiver is receiving as well as the precision of the individual pseudorange measurements. It is preferable to base a solution on widely separated satellites [41].

1.2.1.7 Summary of errors

Tropospheric-,Ionospheric- and clock errors are removed in the process of single- and double differencing performed to produce the RTK-GPS solution. Multipath errors may be mitigated by ground planes or choke rings. Ephemeris data and dilution of precision are given by the system.

1.2.2 Carrier Phase measurement and relative positioning

A different, and more accurate, method of obtaining a position solution for a GPS-receiver involves measuring the carrier phase of the incoming signal. In the case of GPS single frequency applications, the wavelength of the carrier signal is

$$\lambda_{L_1} = \frac{c}{Hz} = \frac{299792458\frac{m}{s}}{1.57542 \times 10^9 Hz} = 0.19m \tag{1.2.5}$$

The precision of the phase measuring sensor is generally about 0.01 cycles, yielding a phase measurement resolution of 1.9 mm [45]. A technique known as differencing is used to eliminate errors and noise common to observations from the same satellite.

$$\lambda_{L_1}\phi_i^A = \rho_i^A + \delta\rho_i^A - N_i^A\lambda_{L_1} + \tau_i - \tau^A + \partial_{i,atm}^A + \partial_{i,mpath}^A + \epsilon^A \tag{1.2.6}$$

By combining the nonlinear phase measurement of Equation 1.2.6 for two separate receivers you get what is known as single differencing.

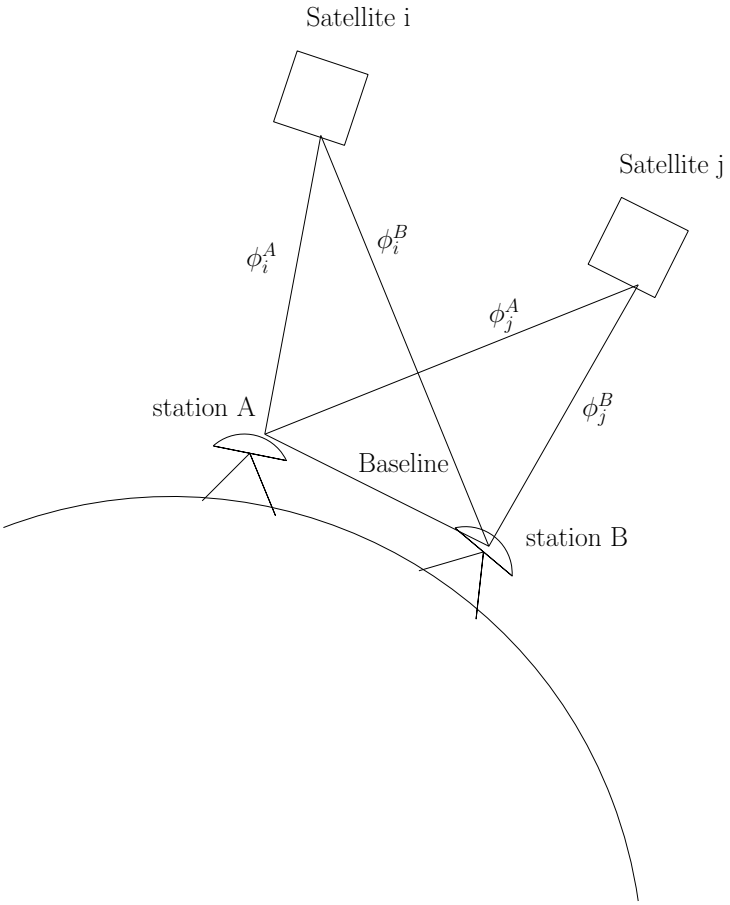


Figure 1.1: Sketch of principle for differencing.

Using the notation of [45], the phase measurement from a satellite, i, to a station, A, (seen in Figure 1.1), converted to length in meters by λ_{L_1}, is given in Equation 1.2.6. In reality several additional satellites are present and four satellites is the minimum requirement.

In Equation 1.2.6 variables may be categorized by their type. Error signals due to the atmosphere, layout of receiver location, measurement of carrier phase and the satellite's deviation from the pre-calculated orbit (ephemeris) are $\partial^A_{i,atm} = \partial^A_{i,iono} + \partial^A_{i,tropo}$, $\partial^A_{i,mpath}$,ϵ^A and $\delta\rho^A_i$, respectively. ρ^A_i is the pseudorange and N^A_i is the unknown integer number of cycles of the carrier phase signal between the satellite and receiver. τ_i is satellite clock error and τ^A is receiver clock error. All variables in Equation 1.2.6 reflects a distance on the line between satellite i and receiver A. Single differencing yields

$$\lambda_{L_1}\Delta\phi^{AB}_i = \lambda_{L_1}(\phi^A_i - \phi^B_i) = \rho^A_i - \rho^B_i - N^{AB}_i\lambda_{L_1} - \tau^A + \tau^B + \partial^{AB}_{i,mpath} + \epsilon^{AB} \tag{1.2.7}$$

where the satellite clock error is eliminated. Double differencing, which is generally the preferred representation of the phase measurement for baselines shorter than 10 kilometres [34], further eliminates errors.

$$\lambda_{L_1}\nabla\Delta\phi^{AB}_{ij} = \lambda_{L_1}(\Delta\phi^{AB}_i - \Delta\phi^{AB}_j) = \rho^A_i - \rho^B_i - \rho^A_j + \rho^B_j - N^{AB}_{ij}\lambda_{L_1} + \partial^{AB}_{ij,mpath} + \epsilon^{AB}_{ij} \tag{1.2.8}$$

1.2.3 Integer ambiguity resolution

To arrive at the desired solution for the double differenced ambiguities in Equation 1.2.8, and thereby obtain an accurate position solution, the estimated ambiguities must be constrained to an integer number [21]. Given a real valued estimate of the ambiguities, possibly produced by an EKF (see [34], Appendix E.7), one obtains a solution type called *float*. Resolving these into integer values results in a *fixed* solution, which yields highly accurate relative positioning. The accuracy of this solution is seen in [37].

A popular method for resolving the ambiguities into integer values is the LAMBDA method introduced by [36]. A simplified explanation of the method is given in [37], and reiterated in this section. The nonlinear observation equation, Equation 1.2.8, is simplified by disregarding multipath errors as well as assuming that tropospheric errors are perfectly modelled. Ionospheric delay is cancelled by the assumption that the baseline between receivers is short, giving approximately similar signal paths for both receivers. Considering these simplifications, Equation 1.2.8 is condensed into the set of equations

$$y = Aa + Bb + e \tag{1.2.9}$$

where y is the difference between observed and computed carrier phases, e is a vector containing measurement noise, a is the unknown vector of double differenced integer ambiguities (given as N in the notation of [45]), b is an unknown vector containing all real-valued parameters and the matrices A and B are design matrices. It is now noted that the vector b contains the three elements of the baseline vector, a key component in relative positioning. The least-squares objective for solving the system is thus

$$\min_{a,b}(y - Aa - Bb)^T Q_y^{-1}(y - Aa - Bb) \tag{1.2.10}$$

subject to the constraints $a \in \mathbb{Z}^m$ and $b \in \mathbb{R}^3$ and where Q_y^{-1} represents the inverse of the variance-covariance matrix for the double-differenced observables [38]. Due to the first of these two constraints the above problem is called an Integer Least-Squares (ILS) problem in the literature and may be solved by:

- Solving the unconstrained least-squares problem ($a \in \mathbb{R}^m$), which yields the real-valued ambiguity ($\hat{a}$) and baseline ($\hat{b}$) vectors as well as the matrix $\begin{bmatrix} Q_{\hat{a}} & Q_{\hat{a}\hat{b}} \\ Q_{\hat{b}\hat{a}} & Q_{\hat{b}} \end{bmatrix}$ containing variance and covariance matrices for the vectors
- Solving the minimization problem given by Equation 1.2.11 to obtain $\check{a}$, which denotes the *fixed* solution of the integer ambiguities
- Obtaining $\check{a}$ allows us to find the fixed baseline solution, $\check{b}$ calculated in Equation 1.2.12.

$$\min_a (\hat{a} - a)^T Q_{\hat{a}}^{-1} (\hat{a} - a), \quad a \in \mathbb{Z}^m \tag{1.2.11}$$

$$\check{b} = \hat{b} - Q_{\hat{b}\hat{a}} Q_{\hat{a}}^{-1} (\hat{a} - \check{a}) \tag{1.2.12}$$

The abovementioned approach of solving for the integer ambiguities is not very effective due to the correlation between the ambiguities. A re-parametrization of the Equation 1.2.11 results in Equation 1.2.13.

$$z = Z^T a, \ \hat{z} = Z^T \hat{a}, \ Q_{\hat{z}} = Z^T Q_{\hat{a}} Z \tag{1.2.13}$$

$$\min_z (\hat{z} - z)^T Q_{\hat{Z}}^{-1} (\hat{z} - z), \quad z \in \mathbb{Z}^m \tag{1.2.14}$$

where the transformation matrix, Z, is of full rank and needs to qualify as an ambiguity transformation [37]. This class of valid ambiguity transformations contains matrices which are volume preserving and have only integer valued elements, for instance the identity matrix of correct dimension. The purpose of re-parametrization of the minimization problem of Equation 1.2.11 into Equation 1.2.14 is to decorrelate the ambiguities (which are extremely correlated when observations of carrier phase are made with short intervals [36]). An optimal transformation yields a close-to-diagonal variance-covariance matrix $Q_{\hat{z}}$ which renders the transformed ambiguities close to decorrelated and the transformed minimization problem may be solved more efficiently. The ambiguities are transformed back to the original domain by $a = Z^{-T}\hat{z}$. Further details can be found in [38] and an improvement on this method is described in [8].

1.2.4 NMEA protocol

In its entirety, the National Marine Electronics Association (NMEA) 0183 protocol [23] includes over 90 sentences describing various data for communication between electronic

devices, predominately for marine applications. Relevant to this application is the two sentences named; $\$--RMC$ (Recommended Minimum Specific Global Navigation Satellite System (GNSS) Data) and $\$--GGA$ (Global Positioning System Fix Data). They comprise the specific messages searched for in the GPS-stream by the standard GPS-parser in the APM. As per the naming convention [5] the first two characters of the sentence, the two hyphens, identifies the talker. In this case the talker is identified as GPS.

1.2.4.1 RMC message

The variable used from the RMC message is the 'Speed over ground', used as the ground speed.

$$\$ \underbrace{GPRMC}_{Name\ of\ sentence}, \overset{1}{x}, \overset{2}{x}, \overset{3}{x}, \overset{4}{x}, \overset{5}{x}, \overset{6}{x}, \overset{7}{x}, \overset{8}{x}, \overset{9}{x}, \overset{10}{x}, \overset{11}{x}, \overset{12}{x}$$

Position	Description
1	Time (Universal Time Coordinated (UTC))
2	Status, V = Navigation receiver warning
3	Latitude (centidegrees)
4	North or South
5	Longitude (centidegrees)
6	East or West
7	Speed over ground, knots
8	Track made good, degrees true
9	Date (ddmmyy)
10	Magnetic Variation, degrees
11	East or West
12	Checksum

Table 1.1: Contents of GPRMC sentence

1.2.4.2 GGA message

From this message, the GPS time and geodetic Earth Centered Earth Fixed (ECEF) position is used.

$$\$ \underbrace{GPGGA}_{Name\ of\ sentence}, \overset{1}{x}, \overset{2}{x}, \overset{3}{x}, \overset{4}{x}, \overset{5}{x}, \overset{6}{x}, \overset{7}{x}, \overset{8}{x}, \overset{9}{x}, \overset{10}{x}, \overset{11}{x}, \overset{12}{x}$$

Position	Description
1	Time (UTC)
2	Latitude (centidegrees)
3	North or South
4	Longitude (centidegrees)
5	East or West
6	GPS Quality Indicator
7	Number of satellites in view
8	Horizontal Dilution of precision
9	Antenna Altitude above/below mean-sea-level (geoid)
10	Units of antenna altitude (meters)
11	Geoidal separation, the difference between the WGS-84 earth ellipsoid and mean-sea-level (geoid), "-" means mean-sea-level below ellipsoid
12	Units of geoidal separation (meters)
13	Age of differential GPS data. Null field when DGPS is not used
14	Differential reference station ID, 0000-1023
15	Checksum

Table 1.2: Contents of GPGGA sentence

1.3 Path following

Path following constitutes the task of following a predetermined path without any time constraints. The constraint on time is what separates path following from the more complex task of trajectory tracking. A path in space may be described by discrete points consisting of e.g. latitude, longitude and height, also called waypoints. Connecting a straight line between the first and second waypoint, second and third wayoint and so on defines the path in which to follow [11].

1.3.1 Line-of-sight guidance Law

For straight-line path following, line-of-sight guidance principles are often applied. In general the goal is to converge on the line drawn between two points. The vehicle seeking to follow the path, given by the straight line, may consider either Enclosure-based steering or Lookahead-based steering [11]. An implementation of Lookahead-based steering will be discussed below.

1.3.1.1 Lookahead-based steering

Initially, a lookahead distance is set. It influences how aggressively the vehicle will steer toward the path between the two waypoints as it is an input to the computation of desired course angle. The desired course angle, along with the sideslip angle, provides a setpoint for the heading controller. The sideslip angle is non-zero when cross-track disturbances affect the vehicle, but side slip effects are not specifically addressed in this thesis. Cross-track- and along-track errors are found by computing the horizontal coordinates of the

vehicle in the path-fixed reference frame. This is done by applying the rotation matrix

$$\boldsymbol{R}_p(\alpha_k) \triangleq \begin{bmatrix} \cos(\alpha_k) & -\sin(\alpha_k) \\ \sin(\alpha_k) & \cos(\alpha_k) \end{bmatrix} \tag{1.3.1}$$

where α_k is the angle between the north direction and the line between the previous and current waypoint. The equation below yields vehicle position relative to the path reference frame defined by the previous waypoint and the straight line between previous and current waypoint.

$$\boldsymbol{\epsilon}(t) = \boldsymbol{R}_p(\alpha_k)^T(\boldsymbol{p}^n(t) - \boldsymbol{p}_k^n(t)) \tag{1.3.2}$$

where $\boldsymbol{\epsilon}(t) = \begin{bmatrix} s(t) & e(t) \end{bmatrix}^T$. The variable, $e(t)$, denotes cross-track error and $s(t)$ denotes along-track error. Only the former of the two is of relevance for straight line path following. The following equations are implemented to produce the desired course of the vehicle.

$$\chi_d(e) = \chi_p + \chi_r(e) \tag{1.3.3}$$

$$\chi_p = \alpha_k \tag{1.3.4}$$

$$\chi_r(e) \triangleq \arctan\left(\frac{-e}{\Delta}\right) \tag{1.3.5}$$

Where $\chi_d(e)$ represents the desired course angle and $\chi_r(e)$ is the velocity-path relative angle [11].

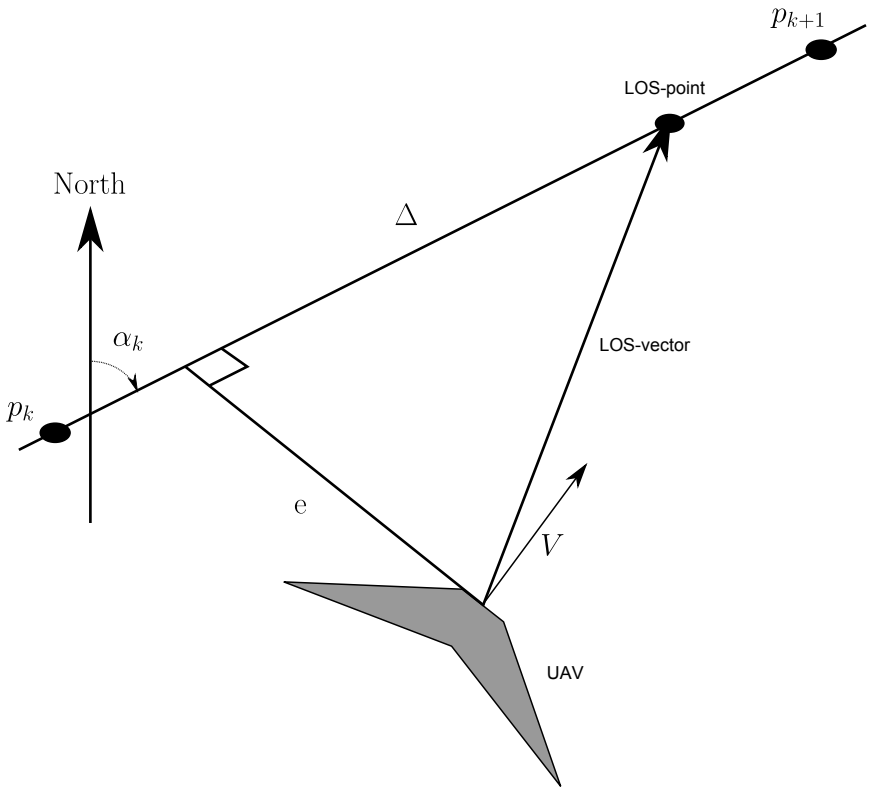


Figure 1.2: The principle of Lookahead based guidance. The original figure is found in [11].

Figure 1.2 illustrates the different variables needed for a lookahead based guidance.

1.3.2 Path following guidance logic implemented in APM

The horizontal guidance scheme implemented in the flight controller APM is similar in principle to the lookahead-distance method described in Section 1.3.1. It may be summarized by the points below [29].

- Find the reference point on the desired path forward of the vehicle. The distance (L_1) between vehicle and reference point is user defined (shown in Figure 1.3).
- Find the angle between L_1 and the desired flight path
- Find the angle between the velocity vector and the desired flight path
- A lateral acceleration command is computed by obtaining vehicle velocity, V, and the angle between the vehicle velocity and the L_1-vector.

The rationale behind this scheme is to provide the autopilot with a desired lateral acceleration equal to the centripetal acceleration experienced when travelling on a circle of radius R, at a speed V (see Figure 1.3). A commanded acceleration to meet those objectives is calculated in Equation 1.3.6.

$$a_{s_{cmd}} = K_{L_1} \frac{V^2}{L_1} \sin(\eta) \tag{1.3.6}$$

Where $a_{s_{cmd}}$ is the resulting desired acceleration and K_{L_1} is a design variable.

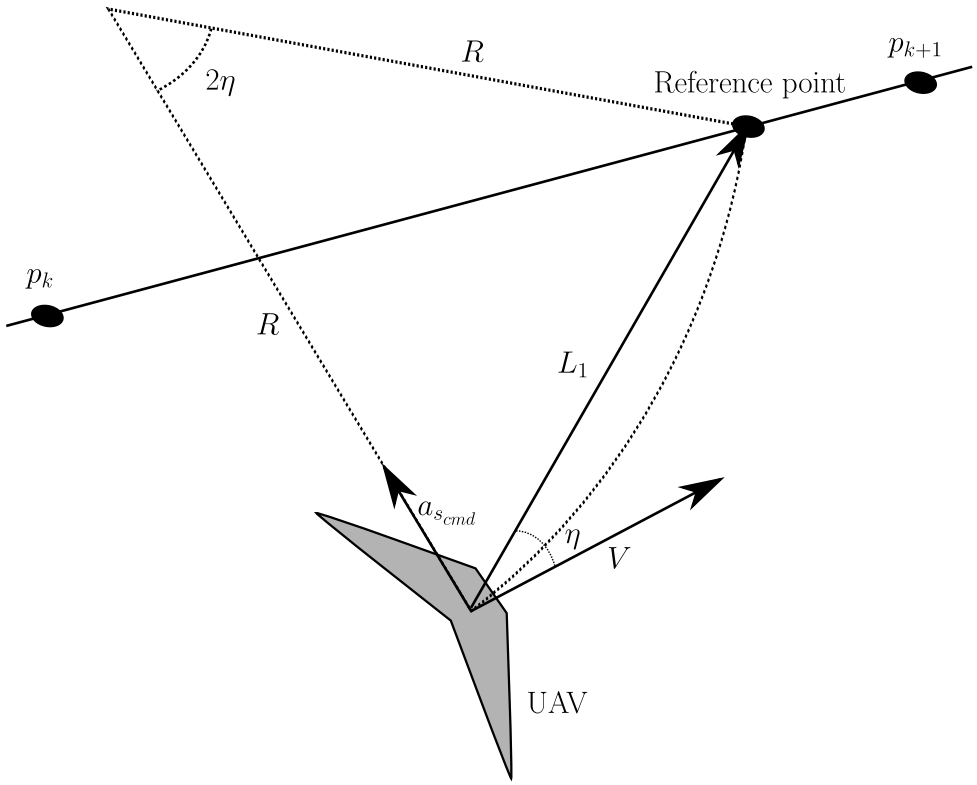


Figure 1.3: The principle of nonlinear L_1 guidance. The original figure found in [29].

For straight-line path following, which is the main area of use for this guidance controller in this application, a nonlinear Lyapunov stability analysis is given in the Appendix of [29].

1.3.3 Vertical guidance logic implemented in APM

Total Energy Control System (TECS) provides an approach to implement an altitude autopilot based on first principles. It is, in version 2.78b of ArduPlane, the default top-level controller for throttle and pitch [31]. The total energy of the aircraft may be expressed as a sum of the potential energy and kinetic energy of the aircraft [10].

$$E_T = \frac{1}{2}mV(t)^2 + mgh(t) \tag{1.3.7}$$

where E_T is the total energy, m is the mass (which is assumed constant), g is the gravitational constant, h is the current height and V is the current speed. In the longitudinal direction the forces acting on the aircraft may be described by

$$T - D = mg\left(\frac{\dot{V}}{g} + \gamma\right) \tag{1.3.8}$$

where γ is the aircraft vertical flight path angle in radians, T is the thrust produced by the motor and D is the drag corresponding to the fuselage shape. "In level flight, initial thrust is trimmed against the drag of the airplane. So the correct thrust control strategy is to develop the incremental thrust command as follows." [10].

$$\Delta T_{cmd} = mg\left(\left(\frac{\dot{V}}{g}\right)_e + \gamma_e\right) \tag{1.3.9}$$

The subscript, e, denotes error and the error of each variable is found by subtracting the true value from the commanded value. To control altitude, this scheme applies both the elevator control surface and thrust magnitude. Both of them directly influence the total energy of the aircraft. The aircraft's total specific energy rate is given as

$$\dot{E} = \frac{\dot{V}}{g} + \gamma \tag{1.3.10}$$

and the total specific energy rate error is

$$\dot{E}_e = \left(\frac{\dot{V}}{g}\right)_e + \gamma_e \tag{1.3.11}$$

The specific energy distribution rate is

$$\dot{L} = \frac{\dot{V}}{g} - \gamma \tag{1.3.12}$$

and the specific energy distribution rate error is

$$\dot{L}_e = \left(\frac{\dot{V}}{g}\right)_e - \gamma_e \tag{1.3.13}$$

The control objective is defined as; driving the two specific energy rate errors, $\dot{E}_e$ and $\dot{L}_e$, to zero. Equation 1.3.14 and Equation 1.3.15 seek to obtain this objective using a proportional and integral controller.

$$T_{cmd} = \left(K_1 T k_{p,thrust} + \frac{K_1 T k_{i,thrust}}{s} \right) \dot{E}_e \tag{1.3.14}$$

$$\delta_{e_{cmd}} = \left(K_2 E k_{p,elevator} + \frac{K_2 E k_{i,elevator}}{s} \right) \dot{L}_e \tag{1.3.15}$$

The variables K_1 and K_2 are the feedback gains of the closed loop aircraft dynamic system. k_p and k_i are proportional and integral gains and $\delta_{e_{cmd}}$ is the commanded elevator angle. Stability proof of a linear model describing the Aerospace Technologies Demonstrator airplane is given in section 8 and 9 of [10] and further insight and improvements to the overall scheme is found in [20].

1.4 PID control

The Proportional-Integral-Derivative (PID) controller provides an output based on the error signal $e(t) = y^c(t) - y$, where $y^c(t)$ is the commanded magnitude of a certain state whilst y is the measured state. Equation 1.4.1 provides a means of converting the error into a corrective signal without any knowledge of the mathematical model of the system. By transferring Equation 1.4.2 into the z-domain, introducing a band-limited differentiator and transforming the result into discrete-time, equations readily available for digital implementation is achieved [14].

$$u(t) = k_p e(t) + k_i \int_{-\infty}^{t} (e(\tau) d\tau) + k_d \frac{de(t)}{dt} \tag{1.4.1}$$

Where $u(t)$ is the output signal and k_p, k_i, k_d are the proportional, integral and derivative gains, respectively. Transforming Equation 1.4.1 into the Laplace domain yields;

$$U(s) = k_p E(s) + k_i \frac{E(s)}{s} + k_d s E(s) \tag{1.4.2}$$

The transformed error is called $E(s)$.

1.4.1 Digital implementation

A short description of the conversion from a continuous- to a digital implementation of a PID controller is given below. The algorithm is found in its entirety on pages 114-116 of [4], but is repeated below for convenience.

- Introduce band-limited differentiator: $U(s) = k_p E(s) + k_i \frac{E(s)}{s} + k_d \frac{s}{\tau s + 1} E(s)$
- Use the trapezoidal rule to convert to discrete time by replacing s with: $s \mapsto \frac{2}{T_s} \left(\frac{1 - z^{-1}}{1 + z^{-1}} \right)$
- Define Laplace-domain integrator: $I(s) \triangleq \frac{E(s)}{s}$

- z-domain integrator: $I(z) = \frac{T_s}{2}\left(\frac{1+z^{-1}}{1-z^{-1}}\right)E(z)$
- Discrete-time integrator: $I[n] = I[n-1] + \frac{T_s}{2}\left(E[n] + E[n-1]\right)$
- Define Laplace-domain differentiator: $D(s) \triangleq \frac{s}{\tau s+1}E(s)$
- z-domain differentiator: $D(z) = \frac{\frac{2}{2\tau+T_s}\left(1-z^{-1}\right)}{1-\left(\frac{2\tau-T_s}{2\tau+T_s}\right)z^{-1}}E(z)$
- Discrete-time differentiator: $D[n] = \frac{2\tau-T_s}{2\tau+T_s}D[n-1] + \frac{2}{2\tau+T_s}\left(E[n] + E[n-1]\right)$

The proportional component is simply a product of the error signal; $P[n] = E[n]-E[n-1]$. Before outputting the signal it is constrained by a saturating function given below

$$u[n] = sat(k_p P[n] + k_i I[n] + k_d D[n],\ limit) \tag{1.4.3}$$

1.5 Reference frames and transformations

The various sub-systems used in the payload operate in different reference frames. To utilize the measurements from all sub-systems they have to be rotated and transformed to a common reference frame where the data can be combined.

1.5.1 Earth-Centered Earth-Fixed (ECEF)

As the name would suggest, this frame has its origin located in the center of the earth and the frame rotates with it. The x-axis points from the origin of the frame toward the intersection of 0° latitude and 0° longitude (Greenwich meridian/Equator intersection). Upward from the origin, toward the North Pole, the z-axis points along the earth's rotational axis. The right hand orthogonal coordinate system is completed by the remaining y-axis [45]. In satellite navigation positions are given in either cartesian or ellipsoidal coordinates relative to the ECEF frame.

1.5.2 North-East-Down/East-North-Up (NED/ENU)

Currently the GPS system employs the reference ellipsoid named World Geodetic System 1984 (WGS-84). The purpose of this reference is to approximate the surface of the earth, and thus act as a reference when for example ellipsoidal height is provided to a user [41]. The NED- and ENU frames are defined relative to the WGS-84. Perpendicular to the plane, which is tangential to the reference ellipsoid in the user position, the z-axis is positive downward for the NED frame and upward for the ENU frame. The tangential plane is spanned by by the remaining x and y axes. In a NED frame representation the x-axis points toward true north and the y-axis points east and completes the orthogonal coordinate system. In ENU the x- and y-axis are switched [11].

1.5.3 Body

The origin of the Body-fixed reference frame is coincident with the origin of the local tangent plane (NED or ENU) [45]. The x-axis points in the forward direction of the vessel, the y-axis to the right and the z-axis upward.

1.5.4 ENU-Body transformation

The transformation between the Body frame and the ENU frame differs sligthly from the NED rotation matrix (see for example [11]) and is given below [1]

$$\boldsymbol{p}_{body} = \boldsymbol{R}_{y,\phi}\boldsymbol{R}_{x,\theta}\boldsymbol{R}_{z,psi}\boldsymbol{p}_{enu} \tag{1.5.1}$$
$$= \boldsymbol{R}^{b}_{enu}\boldsymbol{p}_{enu} \tag{1.5.2}$$
$$\tag{1.5.3}$$

where $\boldsymbol{p}_{body}$ is the position vector in the Body frame, $\boldsymbol{p}_{enu}$ is the position vector in the ENU frame and $\boldsymbol{R}_{axis,angle}$ is a simple rotation about an axis.

$$\boldsymbol{R}^{b}_{enu}(\boldsymbol{\Theta}_{en\ b}) = \begin{bmatrix} c(\phi)c(\psi) + s(\phi)s(\theta)s(\psi) & -c(\phi)s(\psi) + s(\phi)s(\theta)c(\psi) & -s(\phi)c(\theta) \\ c(\theta)s(\psi) & c(\theta)c(\psi) & s(\theta) \\ s(\phi)c(\psi) - c(\phi)s(\theta)s(\psi) & -s(\phi)s(\psi) - c(\phi)s(\theta)c(\psi) & c(\phi)c(\theta) \end{bmatrix} \tag{1.5.4}$$

1.5.5 ECEF to ENU transformation

To convert from geodetic ECEF coordinates to local ENU frame, multiple transformations has to be performed.

1.5.5.1 Geodetic to cartesian ECEF coordinates

To convert from geodetic ECEF coordinates to cartesian ECEF, equation Equation 1.5.5, found in [11] is used

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} (N+h)\cos(\mu)\cos(l) \\ (N+h)\cos(\mu)\sin(l) \\ \left(\frac{r_p^2}{r_e^2}N + h\right)\sin(\mu) \end{bmatrix} \tag{1.5.5}$$

with the variable description found in Table 1.3 below

Variable	Description
μ	Latitude of current position
l	Longitude of current position
$r_e = 6378137m$	Equatorial radius of ellipsoid (semi-major axis)
$r_p = 6356752m$	Polar axis radius of ellipsoid (semi-minor axis)
$N = \frac{r_e^2}{\sqrt{r_e^2\cos^2(\mu) + r_p^2\sin^2(\mu)}}$	Radius of curvature in the prime vertical
h	Current height
X,Y,Z	Cartesian ECEF coordinates

Table 1.3: Variable description of Equation 1.5.5

1.5.5.2 Cartesian ECEF to local ENU

To convert from ECEF to local ENU, a reference value in cartesian- and geodetic ECEF coordinates is needed (which acts as the origin of the ENU frame). Now, using the reference value, the position relative to it can be found. Following [13], the expression for the rotation is found to be

$$\begin{bmatrix} e \\ n \\ u \end{bmatrix} = \begin{bmatrix} -\sin(l_{ref}) & \cos(l_{ref}) & 0 \\ -\sin(\mu_{ref})\cos(l_{ref}) & -\sin(\mu_{ref})\sin(l_{ref}) & \cos(\mu_{ref}) \\ \cos(\mu_{ref})\cos(l_{ref}) & \cos(\mu_{ref})\sin(l_{ref}) & \sin(\mu_{ref}) \end{bmatrix} \begin{bmatrix} X - X_r \\ Y - Y_r \\ Z - Z_r \end{bmatrix} \tag{1.5.6}$$

with the variable description found in Table 1.4 below

Variable	Description
μ_{ref}, l_{ref}	Geodetic ECEF coordinates of reference position
e	East value of local ENU frame
n	North value of local ENU frame
u	Up value of local ENU frame
X,Y,Z	Cartesian ECEF position of new position
X_r, Y_r, Z_r	Cartesian ECEF coordinates of reference position

Table 1.4: Variable description of Equation 1.5.6

Chapter 2

Method

2.1 Retrieval of UAV

As this thesis considers both static-net-, and moving-net retrieval of UAVs, the procedure for each of the two cases will be laid out in separate sections. Furthermore, this chapter serves as general description of the procedures, including takeoff and regular auto flight. Details of the inner workings of algorithms will be explained in Section 2.5 following a description of hardware components in Section 2.2. While static-net retrieval is easier to accomplish than moving-net-retrieval, much of the methodology is similar. The starting point for this study was landing in a static net, hence this procedure receives the most focus in this thesis. Moving-net retrieval employs much of the same algorithms and hardware and the primary differences will be made clear in the following sections.

2.1.1 Static-net retrieval

Static-net retrieval of a UAV demands the position and attitude of the net to remain fixed for the duration of the flight. This particular application is relevant when the UAV is to be retrieved in a ground-based net. This way, no update of the attitude of the net needs to be transmitted to the UAV and thus no sensing equipment needs to be attached to the net mounting.

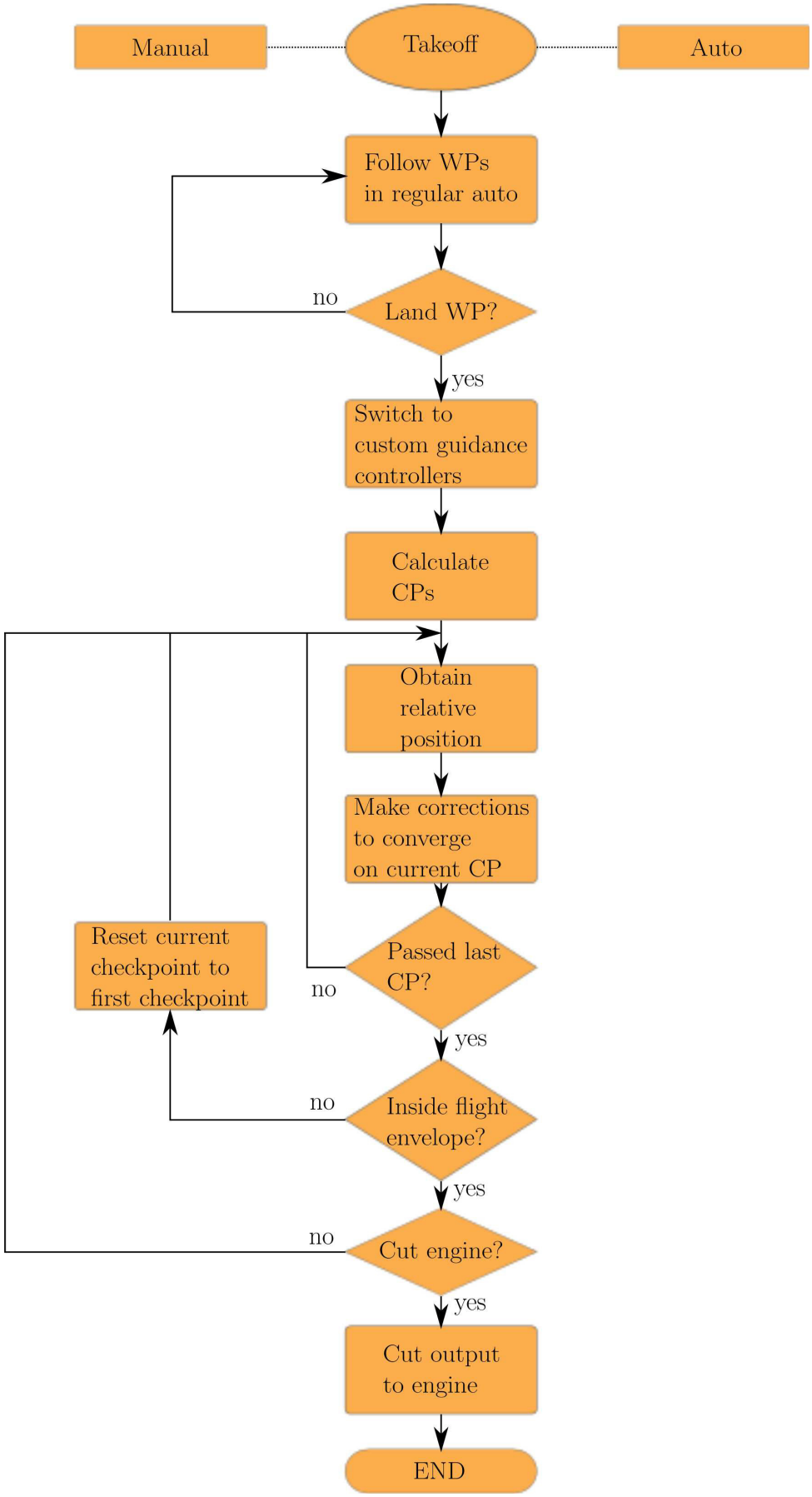


Figure 2.1: Flow chart indicating the intention of the static-net retrieval application. The abbreviation CP stands for CheckPoint and relates to e.g. Figure 2.14

Depending on how the flight is conducted, if it is simulated or performed by a pilot, auto or manual mode is utilized during takeoff. As pilots are on-site, all real-life testing involves a manual takeoff. For takeoff performed in the HIL-environment a simple takeoff procedure was written in order to bring the aircraft off the ground. The commanded roll, ϕ_c, was made dependent on GPS course, χ_{GPS}, because our real-life aircraft did not utilize the magnetometer to provide heading measurements. A short description of how it functions is given in Algorithm 2.1.1.

Algorithm 2.1.1: TAKEOFF(void)

if First takeoff == true
  $\psi_c =$ heading at takeoff
  First takeoff = false
$\theta_c =$ constant (pitch command, e.g. 10 degrees)
$\delta_c = 100$ (throttle percentage)
$\phi_c =$ constrain($\psi_c - \chi_{GPS}$, -10°, 10°)

Where θ_c is the commanded pitch angle, δ_c is the commanded throttle and ϕ_c is the commanded roll angle.

A brief expansion on the events described by Figure 2.1 is in order. Depending on test location and its inherent topology and wind conditions, a suitable waypoint track was laid out and uploaded to the autopilot before the switch over to regular auto mode. Leaving the TECS- (longitudinal) and L1- (lateral) controllers to provide corrections to maintain a desired altitude, and follow the given track, a waypoint marked with the LAND command was placed a certain distance from the Landing Target (LT). Upon entering the LAND command area, the controllers mentioned earlier are replaced by custom controllers for this application. The CheckPoints (CPs) are similar to waypoints, requiring that the aircraft passes a certain distance from it during FA.

Next, the CPs are calculated based on UAV position and desired land course, the latter being a user-settable variable. These CPs will, in the case of an evasive maneuver, later on be re-used. With CPs calculated and defined in the ENU-frame, position is updated and corrections are calculated and distributed to the actuators. When a certain CP has been reached, boundaries on RTK-GPS solution quality, horizontal error (cross-track error) and vertical error (altitude error) are imposed and deviations beyond these boundaries, over a time period, result in an evasive maneuver. If an evasive maneuver is ordered, certain variables are reset leaving the system ready for a second attempt at landing.

The engine of the plane is mounted at the rear of the aircraft and its propellers are designed for belly-landing on a soft surface. However, the engine needs to be cut in order for the propeller blades to fold. This is also the case when performing net-landings. Relative position between net and UAV is quite precise. Therefore this distance may be used to determine when to cut the engine. Due to a brake system implemented in the Electronic Speed Controller (ESC) controlling the engine, the command to cut power to the engine may be given only meters before impact.

2.1.2 Moving net

The procedure for creating the CPs when entering LAND mode is the same in both the static-net and moving-net application. They are dependent on the RTK-GPS solution quality at the moment of calculation and are referenced relative to the LT with a certain desired land course. Assuming that the retrieval net is static on the platform it is mounted, rotations of CPs are necessary in order to approach the moving landing platform at a constant heading referenced to its frame.

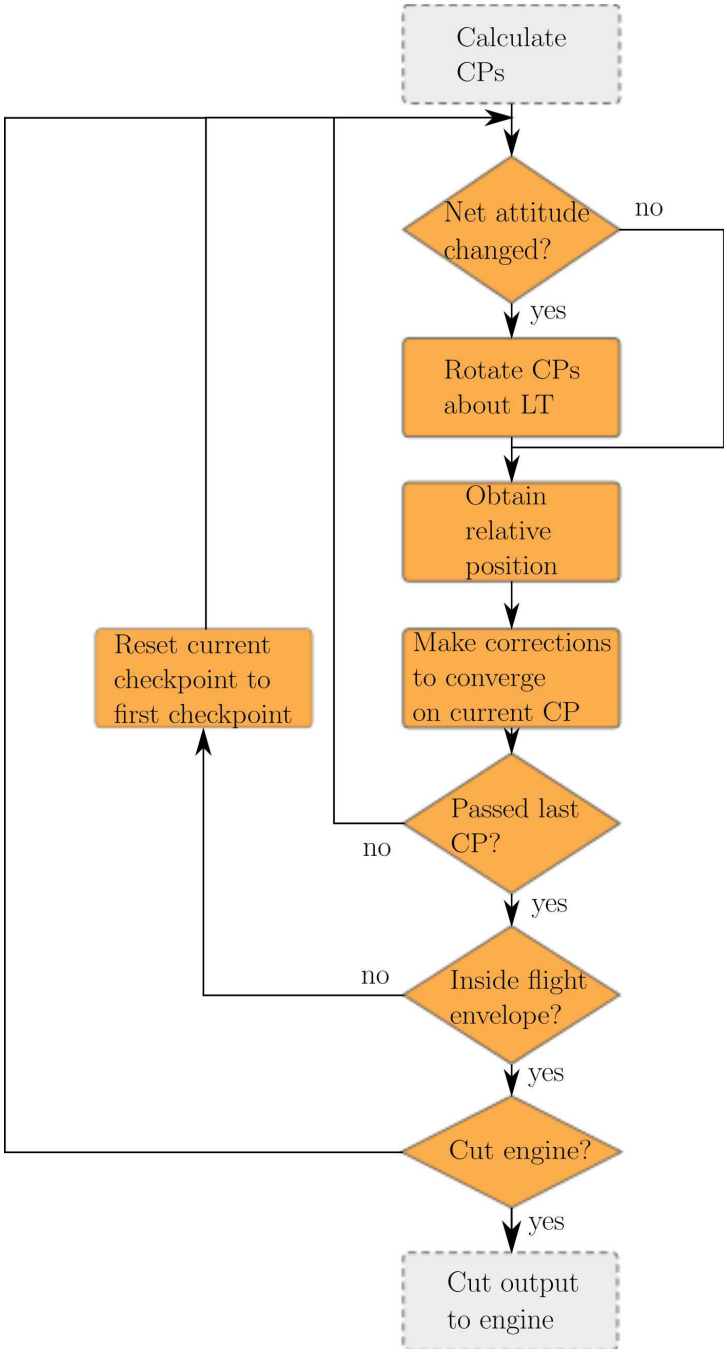


Figure 2.2: Flow chart indicating the intention of the moving net retrieval application

A natural platform to perform net retrieval on is a vessel moving on the surface of the

ocean. This puts constraints on the allowable motion of the ship in all six degrees of freedom. The heading of the platform is the one most sensitive to change, but excessive pitch, roll and heave will likely cause the UAV to perform an evasive maneuver due to large horizontal and vertical errors. Showing only the procedures performed repeatedly while in LAND-mode Figure 2.2 displays the changes made to perform net-retrieval on a moving platform, relative to a static platform.

The addition to the flow-chart seen in Figure 2.2 relative to Figure 2.1 is the rotation of the CPs about the LT. If the heading of the vessel carrying the net changes more than a threshold value, the CPs position in the ENU frame is updated to reflect the change in heading.

2.2 Equipment

The various hardware used to comprise the payload will be described in this section.

2.2.1 Overview of hardware

The hardware used to assemble the payload and base station in order to achieve the intended function is described in Figure B.4. It, visually, describes the connection between each component and may serve as a reference when reviewing Section 2.2.

After performing the preliminary test of RTK-GPS, the limitations of the standard telemetry (see [2]) with respect to range and throughput became obvious along with the limited Transmit (TX) power and radiation angles of the NanoStation [28]. This obsolete setup is shown in Figure B.3. In order to extend the reach of the telemetry and secure uninterrupted transmission between base station and UAV, the telemetry stream was altered to be transmitted using the WiFi transceivers. The NanoStation was replaced by a Rocket M5 and two new roof-mounted dipole antennas ([44]). Figure B.4 shows the new configuration.

2.2.2 UAV fuselage

Containing all the electronics mentioned in this section is the Skywalker X8 Flying Wing UAV fuselage [15]. Built from Expanded Polyolefin (EPO) foam it is very durable and lightweight.

- Wingspan: 2120 mm
- Aircraft gross weight (All-Up-Weight): 3500 gram

Motor power and battery capacity is dependent on the weight of the payload contained in the equipment bays. However, tentative values have been given as 400-800 watts for motor power and 3-5 ampere hours for battery capacity.

2.2.3 GPS-receiver and antenna

Two identical single frequency (L1) GPS-receivers were used in this project. They are manufactured by the Swiss company u-blox. The model is called LEA-6T and a summary of relevant specifications is given below [42].

- Maximum update rate 5 Hz
- Type of receiver: GPS L1 C/A code
- Able to transmit raw GPS data via Universal Asynchronous Receiver/Transmitter (UART) or Universal Serial Bus (USB)

Two types of antenna were initially considered for GPS use on the UAV and base station. Initially, the active antenna included in the u-blox LEA-6T evaluation kit was used on the base station. For mounting in the UAV, these were deemed too bulky and replaced with an embedded precision GPS L1 antenna (detailed in [35]). A more stable and durable base station antenna (AT-1675-295 GPS antenna from Aero Antenna Technology) was introduced when testing commenced at the new location at Agdenes.

2.2.3.1 Ground plane

A ground plane has positive effects on both patch-antenna gain and the ability to reject multipath. The shape of the ground plane as well as its thickness defines how much it contributes to the total gain of the antenna [39]. The placement of the patch antenna on the ground plane is also significant, with a center mounting being preferable to mounting the antenna close to an edge.

Rejection of ground-bounce multipath error is usually performed with a threshold for allowable satellite elevation angle. A ground plane supplements this protection by physically rejecting signals below 0° elevation angle[9].

2.2.4 Data processing board

For running the RTKLIB, and thereby obtain the absolute and relative position of the UAV, a Pandaboard was used. Relevant specifications are summarized below.

- Processor: Dual-core ARM ®1.2 GHz.
- Temporary storage: 1 GB DDR2 Random Access Memory (RAM).
- Storage: Support for High-speed, High-capacity memory cards.
- Wired connections: Ethernet, 2xUSB with optional expansion available, UART/RS-232.
- Dimensions: 114.3x101.6 (length by breadth in mm), weight is 81.5 grams.

2.2.5 Communication

Wireless communication between base station and UAV was carried out with two Rocket M5 radios from Ubiquity Networks. Pairing two of these units allows a user to set up a wireless, point-to-point TCP/IP network boasting a theoretical range of 50 km and transfer speeds of 150 Mbps. The radio transmitter placed on the UAV as part of the payload was stripped of its plastic casing to minimize weight. It was also equipped with two dipole antennas, mounted vertically on the fuselage and connected through 20 cm of 8 mm coaxial cable, as shown in Figure 2.3a.

2.2.5.1 Rocket M5

"Rocket M5 is a rugged, hi-power, very linear 2x2 MIMO radio with enhanced receiver performance."[27]. It allows for a connection between two or more Rocket M5s or in combination with a NanoStation (described in Section 2.2.5.3) using WiFi between the transceivers and Local Area Network (LAN)-connection at each end. A short summary of the specifications can be seen in Table 2.1, and the stripped-down version used during testing is seen in Figure 2.3b

Frequency range	Output power	Transmission rate	Power options	Connector	Power Consumption
5470-5825	27 dBm (max)	Up to 150Mbps (theoretical)	PoE	2xRP-SMA	8 Watts (max)

Table 2.1: Summary of specifications of the Rocket M5 - Based on [27]

2.2.5.2 Antennas

As there are weight and size limitations when choosing antennas to use on the UAV, different antennas were used on the base station and the UAV.

WiMo 18720.3H

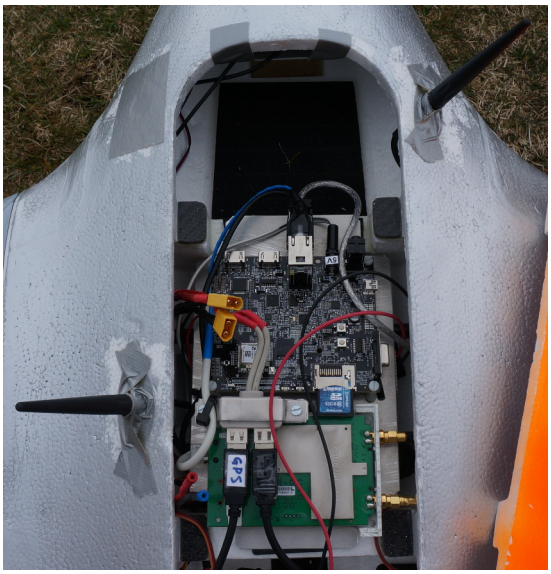
This antenna was chosen for its size, frequency range, weight and TX power, which fit with the UAV and the Rocket M5. The specification for this antenna can be seen in Table 2.2

Frequency range	Length	Weight	Max TX power
5.5 - 5.8 GHz	10.5cm	25g±2g	5Watts

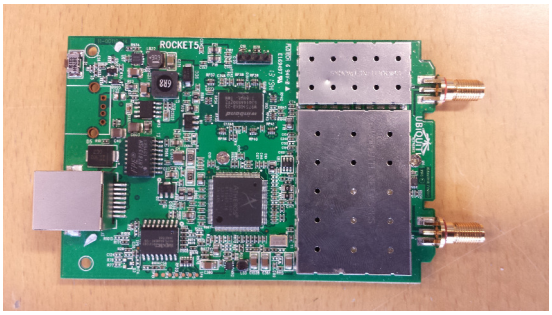
Table 2.2: Specifications of WiMo 18720.3H - Based on [43]

WiMo GP5000-10

As there are no weight and size restrictions on the base station, an antenna with higher gain and power rating can be used. The specifications of the antenna chosen can be seen in Table 2.3



(a) Overview of UAV with payload and three vertical dipole antennas. The two larger ones transmit base station correction while the smaller transmits telemetry from the autopilot.



(b) A stripped Rocket M5

Figure 2.3: Pictures showing mounting of antennas and transciever

Frequency range	Length/Width	Weight	Max TX power
5.1 - 5.9 GHz	430mm/20mm	230gr	20Watts

Table 2.3: Specifications of WiMo GP5000-10 - Based on [44]

2.2.5.3 Nanostation M5

The Nanostation M5 works in the same way as Rocket M5 (described in Section 2.2.5.1), as a communication tool between either multiple Nanostation M5s or in combination with Rocket M5s. It has a larger frequency range than the Rocket M5, however, it does not have the option of connecting an external antenna for improved range and gain. The specification for Nanostation M5 is summarized in Table 2.4.

Frequency range	Output power	Transmission rate	Power options	Connector	Power Consumption
5170-5850	27 dBm (max)	Up to 150Mbps (theoretical)	PoE	-	8 Watts (max)

Table 2.4: Summary of specifications of the Nanostation M5 - Based on [28]

2.2.6 APM

The APM is an Arduino-compatible open source software used by the APM autopilot to control fixed-wing aircraft. The APM provides an entire UAV control system with scriptable missions, 3D waypoints, in-flight uploading of commands and powerful ground

station software ([30]). The APM comes with ready-made navigation algorithms discussed in Section 1.3.2.

Features
The features (and layout of the APM version 2.5) are shown in Figure B.5. APM is built with an ATMega 2560 board as its base, and the features added are:

- GPS connector
- Wireless telemetry connector
- Magnetometer
- External power connector
- IMU
- Ready made control of fixed-wing UAV (with waypoints)

2.2.6.1 Navigation

In autonomous flight GPS data and an Inertial Measurement Unit (IMU) are used in combination for navigation. The combination of the two systems provides increased stability with respect to GPS-dropout [45]. A magnetometer measures the strength and sometimes the direction of a magnetic field which can, when combined with the IMU, give a reliable estimate of heading [19].

2.2.6.2 Communication

The APM communicates with a GCS through Micro Air Vehicle Link (MAVLink). MAVLink packs C-structs over serial channels with high efficiency and send these packets to the ground control station [32]. This means information from GCS to the UAV can easily be sent using transmitters without a large baudrate.

2.2.6.3 Waypoints

When waypoints are sent through MAVLink in the correct waypoint form, the waypoints are stored in the Electrically Erasable Programmable Read-Only Memory (EEPROM) of the APM. There they are kept until the APM is set to auto mode. From there it reads the first command from the stored waypoints in the EEPROM, and then executes that command. When the command has finished, it loads the next command and keeps going until the list of commands is empty. If the command list is empty the UAV will attempt to return to the point it was launched from.

2.2.6.4 GPS

The APM works out of the box together with u-blox GPS. However, the output from the Pandaboard (which provides our RTK-GPS solution) is not in the same format as one standalone GPS receiver from u-blox. Instead, the output from the Pandaboard is a custom NMEA message, where the baseline information is added to it.

This means that the RTK-GPS solution from the Pandaboard is transmitted through the custom NMEA message to the APM. The APM takes this as a normal GPS signal, therefore no further modification is needed to get RTK-GPS on the APM. If the signal to the base station is lost, the signal APM treats the single GPS signal from the rover as it would any standard non-RTK-GPS signal. No code modification is needed to handle the event where RTK-GPS solution is lost.

2.2.7 Retrieval net

The landing target is composed of a 3 m high by 5 m wide net, fastened with ropes at each corner to the mounting fixtures. This net is not optimal for UAV retrieval as the UAV will fall to the ground after impact. However, it can be used to prove the concept of net retrieval. The mounted net can be seen in Figure 2.4

Figure 2.4: The landing net, mounted

2.2.8 Payload

The collective term payload is in this case used to describe the separate units of hardware, that need to fit in the centre compartment of the UAV. Each one of the separate devices is necessary to perform the task of obtaining a RTK-GPS solution and steering the UAV towards the centre of a retrieval net. Figure B.1 and Figure B.2 shows the layout of the payload assembly. Notice the slightly angled profile of the mounting plate, shown in Figure 2.5. A thin carbon fibre tube runs across the centre of the fuselage, making the angled profile a necessity for a secure fit of the payload.

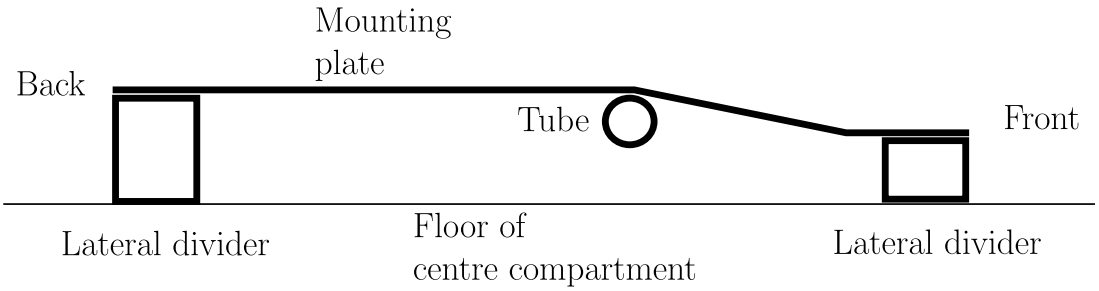


Figure 2.5: Profile of mounting plate for payload system.

The payload is powered by a 3S LiPo battery. The uptime of the payload was tested, and the results are shown in Appendix B.4.

2.2.8.1 Telememetry- and GPS cable

The connections on the APM for GPS and telemetry are both serial connections. Therefore a standard Future Technology Devices International (FTDI) TTL-232-R-3V3 cable was customized to fit with the connector (DS13 connector) the APM uses for those connections. This FTDI cable allows conversion of the signals from TTL serial UART to USB [24], which gives us a connection from the Pandaboard to the APM (USB $\leftrightarrow$ serial). The cable was made using the datasheet for the FTDI cable (see [24]) and the eagle files of the GPS connector for the APM (See Appendix C - APM eagle files). The resulting cable can be seen in Figure 2.6. The connections from the USB $\rightarrow$ DS13 connector can be seen in Table 2.5

USB		DS13
1	$\rightarrow$	5
2	$\rightarrow$	N/C
3	$\rightarrow$	1
4	$\rightarrow$	3
5	$\rightarrow$	2
6	$\rightarrow$	N/C

Table 2.5: Connections in the GPS/Telemetry cable

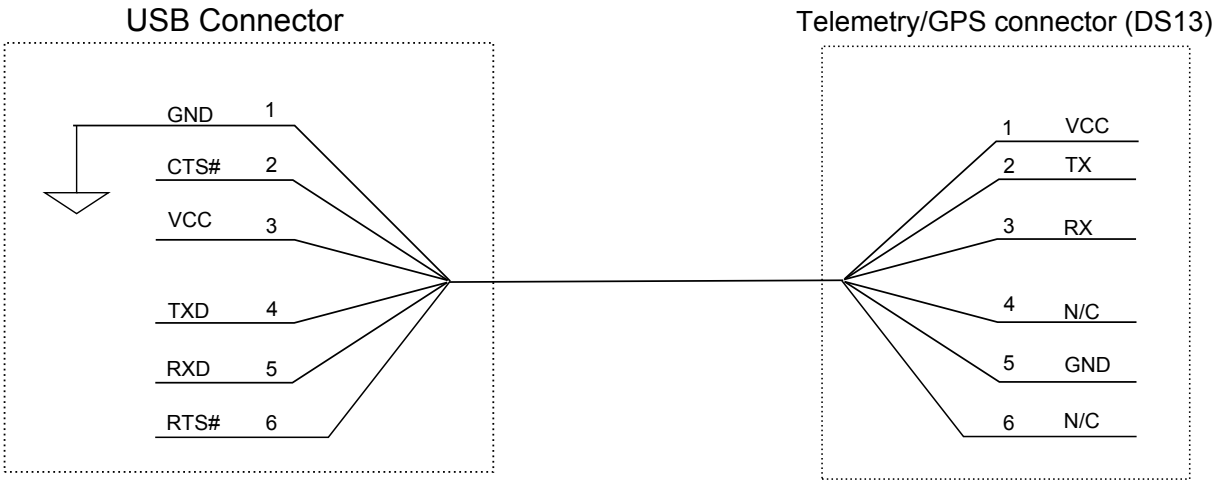


Figure 2.6: FTDI USB to DS13 Connection

2.2.8.2 Rocket power and data cable

The Rocket M5 is powered through PoE (Power over Ethernet), optimally with a working voltage of ± 12V [27], which is available from the step-up transformer. To reduce weight, the original PoE/LAN adapter was removed and a standard CAT-5e cable was modified to utilize the 2 free pairs available in the cable to power the Rocket M5. With this solution, the cable is split in two, where 2 pairs go to the voltage transformer, and 2 pairs connect to the Pandaboard for data transfer. The overview of this connection is seen in Figure 2.7, whereas the specific connection for both Pandaboard and the step-up transformer can be seen in Table 2.6 and Table 2.7 respectively.

RJ45 Rocket M5		RJ45 Pandaboard
1	→	1
2	→	2
3	→	3
4	→	N/C
5	→	N/C
6	→	6
7	→	N/C
8	→	N/C

Table 2.6: Data connection RJ45 Rocket to RJ45 Pandaboard

RJ45 Rocket M5 Rocket M5		Step-up transformer
1	→	N/C
2	→	N/C
3	→	N/C
4	→	+12V
5	→	+12V
6	→	N/C
7	→	-12V
8	→	-12V

Table 2.7: Power connection RJ45 Rocket to step-up transformer

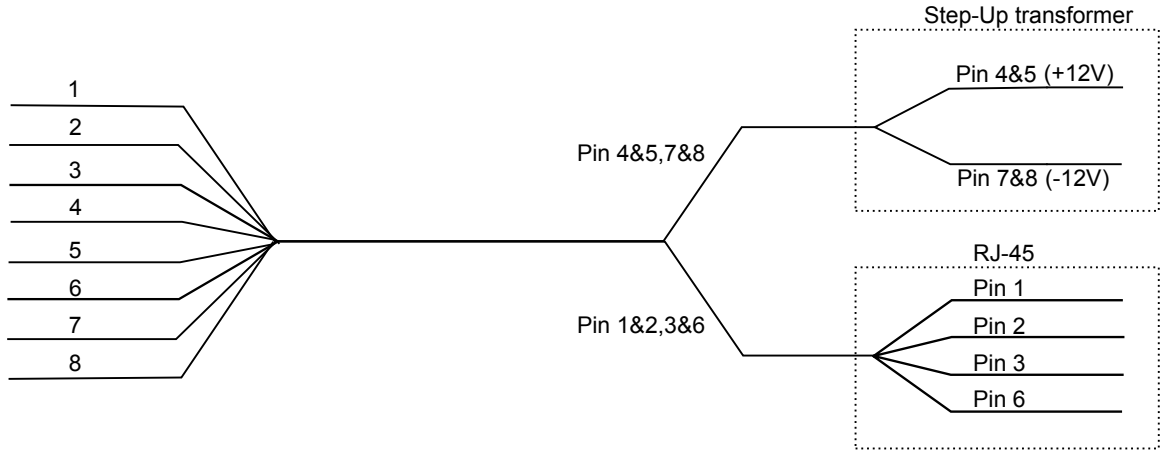


Figure 2.7: Rocket M5 to power and data

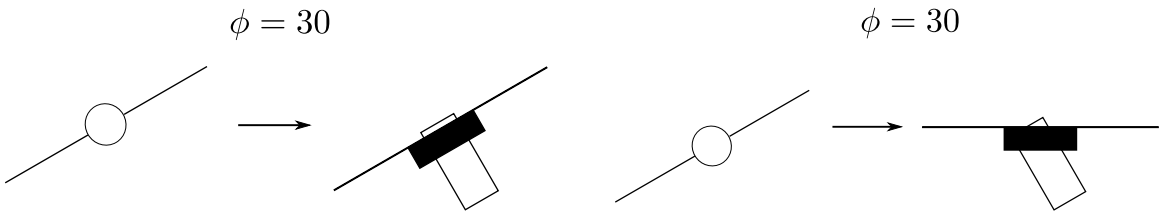
2.2.8.3 Roll stabilization of GPS antenna

To increase the possibility for a fixed RTK-GPS solution (described in Section 1.2.3), a system to stabilize the GPS antenna about the UAV's Body frame x-axis was implemented. The setup for this consists of

- HiTec HS-5245MG Digital Servo
- Lightweight aluminium ground plane (8cm by 5cm)
- Lightweight aluminim fastening plate

with this setup, the loss of fix as a result of continuous high roll angle turns can be negated, which in turn leads to a higher availability of a fixed RTK-GPS solutions.

Looking at Figure 2.8a, the effects of a high roll angle turn is visualized. As the plane banks, the GPS signals are halted by the ground plane that is there to remove electric interference and reject multipath errors. Depending on the position of the satellites currently used to solve the Integer Ambiguity Resolution (described in Section 1.2.3), all satellites used to calculate the baseline might be lost. This leads to an inability to calculate an accurate baseline between the UAV and base GPS antenna, which in turn leads to the UAV not being able to/attempting to land at the desired LT. Figure 2.8b shows the roll stabilization in effect, which allows the antenna to stay level during rolls.



(a) No roll compensation (b) With roll compensation

Figure 2.8: UAV with and without roll compensation

2.2.9 UAV launcher

The launcher was used for manual takeoff during testing. Shown in Figure 2.9, it was operated by the pilot and provided the UAV with additional thrust making it possible to perform takeoff without landing gear.

Figure 2.9: The launcher used to perform takeoff.

2.3 Settings of the GPS-receiver

Appendix A of the u-blox manual [40] describes the default settings for the GPS-receivers used in this project. However, some of the default settings had to be changed in order to allow interaction between systems and optimal utilization of hardware. These are listed below along with a description of what it governs. The settings are altered by changing the contents of messages contained in the message class **UBX-CFG** (configuration).

- Message section (**UBX-CFG-MSG**). Messages **02-10 RXM - RAW** and **02-11 RXM - SFRB** needs to be activated for UART in order to interact with RTKLIB with enough throughput for 5 Hz update rate.

- Port configuration (message **UBX-CFG-PRT**). Set output on specific physical port of the receiver, including the protocol and transfer rate in bauds. The transfer rate must support the amount of data that the receiver outputs. This amount varies depending on number of satellites and update rate.

- Rate of update (message **UBX-CFG-RATE**). Determines the time intervals between each measurement update.

- Dynamic platform setting in message (**UBX-CFG-NAV5**). Options for setting the type of dynamic platform the GPS-receiver is located on. Here Airborne $< 4g$ is chosen, as it's designed for a highly dynamic platform.

As a basis for testing the following settings were applied to the GPS-receivers

Setup	Dynamic platform	Update rate	Baudrate
Static	Stationary/Airborne $< 4g$	5 Hz	115200
Moving	At Sea/Airborne $< 4g$	5 Hz	115200

Table 2.8: Basic settings for base station receiver on different dynamic platforms.

The moving target scenario was not tested in real life, but the last row of Table 2.8 is a suggestion for settings.

2.4 Settings of Real-Time-Kinematic Library

RTKLIB is the positioning program of choice for this application and produces a position solution. The term solution is in this case the term used for the position output produced by programs that process GPS-signals. As stated in the previous section, RTKLIB needs raw measurement data. This is provided from both receivers and is combined in the software to produce a solution with centimetre-level-precision. Depending on the type of operation, different settings are relevant. A description of three scenarios relevant to this project is given below.

1 Precise Point Positioning (PPP)-Static mode (used in preparation): Used to obtain absolute position of base station in order to utilize relative positioning with a fixed base station at a later stage. The sampling takes place over a large time interval and a ”sample mean” calculation yields an accurate position.

2 Kinematic: By knowing the absolute position of the base station, carrier-phase relative positioning may be applied with the positioning mode called Kinematic in RTKLIB.

3 Moving base: Desired setup for an application where base station receiver is located on a ship and the roving receiver is part of the payload on a UAV. The baseline retains its accuracy, but the absolute position of both receivers is reduced to the accuracy of a single receiver solution (see [34] page 169).

Scenario 1, given above, is the classic GPS setup, where a single, L1-frequency receiver is used to obtain a position solution. However, as shown in [46], when applying the precise point positioning scheme in RTKLIB the quality of absolute positioning increases. This is done by utilizing pseudorange measurements as well as phase observations, satellite orbits and the satellite clock, which are all inherent in the raw GPS data stream.

Common to scenario two and three is that they both need to resolve the ambiguous number of L1 carrier cycles between the satellite and receiver in order to yield accurate position solutions. The former has the options Continuous and Fix-and-Hold and the latter has only the real numbered ratio as input. Default threshold value for the ambiguity resolution validation ratio is 3.0. A lower ratio value yields less accurate solutions. The main methods of resolving these ambiguities are [34];

- Continuous: ”Continuously static integer ambiguities are estimated and resolved.”

- Fix and Hold: "Continuously static integer ambiguities are estimated and resolved. If the validation is OK, the ambiguities are tightly constrained to the resolved values."

The ambiguity resolution mode is selected on tab two of Figure 2.10. Mentioned in the procedure for fix and hold ambiguity resolution, is the ratio of squared residuals of the best integer vector divided by the second best. It is performed to be able to measure the reliability of a fixed solution. Minimization of Equation 1.2.14 provides the two best solutions of the integer least squares problem. In turn these are used to obtain the mentioned residual.

$$R = \frac{(\check{Z}_2 - \hat{Z})^T \mathbf{Q}_Z^{-1} (\check{Z}_2 - \hat{Z})}{(\check{Z}_1 - \hat{Z})^T \mathbf{Q}_Z^{-1} (\check{Z}_1 - \hat{Z})} \tag{2.4.1}$$

where $\check{Z}_2$ and $\check{Z}_1$ are the best and second best fixed integer solution to Equation 1.2.14, respectively. $\hat{Z}$ is the integer ambiguity vector.

The following paragraphs deal with other settings available in the configuration of RTK-LIB's RTK-GPS program. They are mentioned by order of appearence in the configuration file and explanation of what they govern may be found in [34].

Used in order to exclude satellites below a certain elevation angle, the setting "Elevation mask" can substantially change the Dilution of Precision (DOP) values of the current satellite constellation received by the two receivers. The user may specify a lower bound ranging from 0° to 70° with a default value of 15°. This seems a reasonable value as the data from satellites below this angle is usually corrupted by phenomena such as multi-path error, cycle slips and a low SNR [7]. In the same manner satellites may be excluded based on the SNR value of their signal perceived by the receiver by the setting SNR mask.

Receiver dynamics (Rec Dynamics) was set to OFF along with Earth Tides Correction. The former setting requires a mathematical model of the receiver platform for estimation of acceleration and velocity (for use in prediction of position) and the latter requires earth tides correction input. Neither model, nor tide corrections were available so the settings remained default. As for the correction of ionospheric error and tropospheric error, both settings were left default, meaning the former was set to Broadcast and the latter set to the Saastamoinen-setting [33]. Broadcast allows the user to implement the Klobuchar model, as its parameters are broadcast with the navigation message (see Section 1.2.1). Receiving on only the L1-GPS frequency requires corrective measures to be taken [18]. Broadcast is also the setting of choice for ephemeris and clock information.

The next user-setable parameter of the configuration file is the Receiver Autonomous Integrity Monitoring Fault Detection and Exclusion (RAIM FDE). If enabled, this feature excludes a satellite if the sum of squared errors of residuals (the difference between estimated and observed value) gets above a certain limit. Along with the PRN exclusion setting, this was not employed. The last setting, of page one of settings (see Figure 2.10),

involves selecting type of navigation system. GPS was the only viable option as the receivers used in the application are limited to this system [40].

Page two of settings (see Figure 2.10) is largely devoted to settings concerning RTK variables. Firstly, the integer ambiguity resolution strategy must be selected (mentioned in above bullet points). Using the manual provided by the creators of RTKLIB (see page 167 of [34]) to investigate the intended use of each mode and the principle of trial-and-error, fix-and-hold was selected as strategy for integer ambiguity resolution. The minimum ratio to fix ambiguity was held at its default value, 3.0. On this page the remaining settings are:

- Min Lock/Elevation to Fix Ambiguity: Demands a certain amount of locks on satellites and a certain elevation angle to attempt to fix ambiguities. The default value of 0 locks and 0 elevation angle was used due to not wanting to put strict bounds on when to fix ambiguities, but rather test if the prduced solution is good enough at a later stage.
- Min Fix/Elevation to hold ambiguity: Minimum number of valid satellites used in solution set to 5 instead of the default 10 satellites. This to allow fix-and-hold to perform its function with a lower number of valid satellites. Elevation angle is already constrained in Elevation Mask.
- Max Age of Differential (AoD): The default, and applied, value is 30 seconds. This is, however, not very relevant as much more frequent updates are necessary. The setting was not altered due to measures taken by RTKLIB if lowered.
- Sync Solution: Set to OFF in order to minimize latency in solution.
- Number of iterations: Applies when operating with short baselines (< 1 m) and the non-linearity that occurs in the measurement equation (see page 39 of [34]).
- Baseline Length Constraint: The changes in baseline experienced during flight renders this setting irrelevant. Intended for attitude applications.

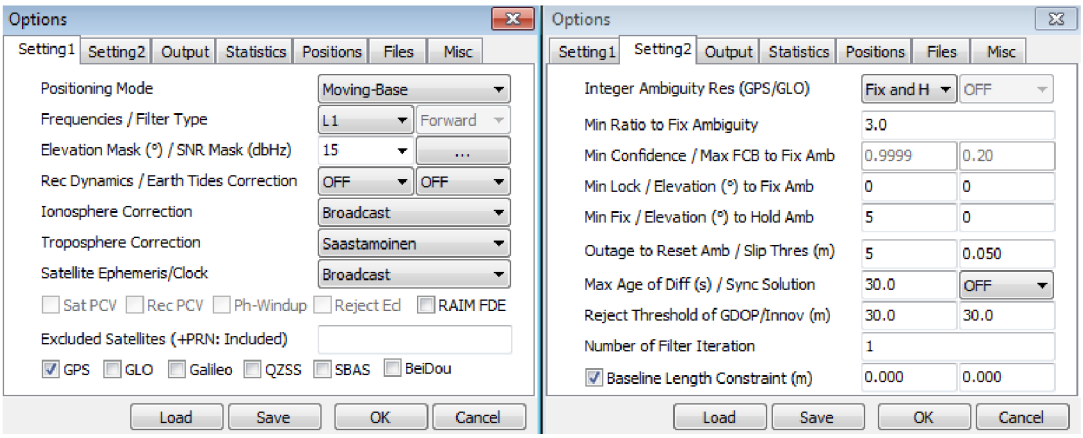


Figure 2.10: The Option window of RTKLIB and its two Setting tabs

2.5 Software

A variety of software is necessary both to perform HIL simulations and during real-life flight. Here, the software used for both will be discussed.

2.5.1 XPlane

XPlane is used to visualize and simulate the environment during the HIL simulation of the UAV. It was quite demanding when it came to computer performance needed, but worked sufficiently with rendering details reduced so that the simulation Frames-Per-Second (FPS) could stay at a level which provided the UAV with a realistic environment. As XPlane is used by world-leading defence contractors, air forces, aircraft manufacturers, and even space agencies for applications ranging from flight training to concept design and flight testing [47], its ability to emulate real-world conditions are good. The model used for simulation is found in Appendix C - XPlane - MaxiSwift [26].

XPlane has its own Software Development Kit (SDK), so both modifications and new plugins were created to append our needs to the already existing possibilities in XPlane. These plugins were used in evaluation of both path- and height controllers.

The landing targets used in the HIL simulation were created and placed by the software described in Appendix E

2.5.1.1 Data from landing target

To be able to simulate the baseline given by RTKLIB (described in Section 2.5.5 in real life, the position of the landing target is necessary. This plugin therefore extracts

- The geodetic ECEF position
- The objects heading

which is sent to MP using User Datagram Protocol (UDP), to be used to replicate a baseline between the landing target and the UAV. This plugin can be found in Appendix C - XPlane - plugins - Position of landing target.

2.5.1.2 Moving target

This plugin takes a 3D model (.obj) created in Blender, described in Appendix E.1, and animates it in the XPlane world. This plugin is created by the user 'mroe' in the XPlane forum and the manual can be found in Appendix C - XPlane - plugins - Animate an object.

There are several possiblities when it comes to animating this object, such as

- Set object height
- Set object speed
- Set path in ECEF frame

which allows for simulation of a moving landing target (boat for example). The plugin also sets certain *datarefs* (used by plugins to read XPlane's internal variables) for the animated object, such as

- geodetic ECEF position
- Heading

This information is extracted by the plugin previously described in Section 2.5.1.1, and sent to MP.

2.5.1.3 Target objects

The size of the target object was set to the same size as the net the UAV had to land in during real tests (described in Section 2.2.7). The target position was set in the center of the object.

Static landing targets in the simulation environment were added using OverlayEditor (described in Section E.2). The static objects were set in a circle (see Figure E.2) with different positions and heading, such that desired landing course and position could be changed easily.

2.5.2 MP

MP is a Graphical User Interface (GUI) for interaction with (among others) APM as a GCS. This means that MP can send commands live to the APM, either to add waypoints or to go in for landing if needed.

MP was developed for the APM, such that all features from the APM are represented in MP, in a graphical package. This is the reason why MP was chosen to be used in this HIL simulation, as creating a flightpath with waypoints were a significant part of the test parameters.

Commands and waypoints are created by clicking on a map and inserting desired height and setting the type of command the user wants to send. Among others, these are the most used

- Take-off
- Land
- Waypoint

MP provides a MAVLink connection, which is the protocol used for transferring data to and from the APM. Because of this, in addition to setting waypoints etc, it's also possible to edit the control parameters on the APM from the MP GUI.

To create a HIL environment more suited to our needs, some modifications to the existing MP were needed. These are described in detail in the section below.

2.5.2.1 Calculating and drawing points for final approach

The path generated during landing is, as opposed to normal WP flying, not created by the user, therefore not shown in MP. This leads to uncertainty as to what the APM intends to do while in the FA control loop. For the operator of the GCS, it's a necessity to know what the APM desires to do, to see if it is expected behaviour, and if not, tell the pilot to take control over the UAV. To make this information available to the GCS operator, the vectors connecting the desired path of the APM during FA are sent to MP. The pseudo-code for this function is given in Algorithm 2.5.1 below.

Algorithm 2.5.1: DRAW FA POINTS(BL,LLH)

comment: Here BL is the RTK-GPS solution, and LLH is the

comment: geodetic ECEF position of the UAV

if $BL \neq 0$
  Store current geodetic ECEF position (LLH)
  Rotate ENU values from BL vector to LLH
  Draw polygon markers based on rotated BL LLH values

The transformation between frames are described in Section 1.5.5.

2.5.2.2 GUI additions for HIL environment

A big step in creating a more dynamic HIL environment, is to make sure that both the environmental factors and the measurements can be changed quickly without the need to recompile or add variables in the code. Therefore multiple additions were put in the existing MP HIL setup, such as

- RTK-GPS solution
- Landing target position
- Force evasive maneuver
- Landing target heading

These additions made it possible to rapidly change our test-setup without much delay.

The RTK-GPS solution quality selector allows the user to change the accuracy of the baseline vector sent to the APM. Any RTK-GPS solution less than 1 (Fix) will put randomly generated values on the existing perfect baseline values created from X-Plane values. This allows the observer to see how the horizontal- and height controllers act with 'unstable' measurements, and the oscillations/reactions the inaccurate measurements created. With this the user can monitor the RTK-GPS solution and implement measures to reduce the impact inaccurate measurements have on the system.

2.5.2.3 MAVLink additions, on both MP and APM

To allow for the additional information required of the MP changes, the MAVLink messages were modified to include the data required. The variables are added to a common .xml file, which is used by both APM and MP to generate .c files and C# files respectively.

2.5.3 Changes/additions made to ArduPilot software

As mentioned previously in this thesis, the autopilot of choice for this application was the APM. It allows the user to select between several firmwares depending on the type of vehicle to control. For this application the ArduPlane firmware version 2.78b was selected to provide both autopilot functionality during autonomous flight and distribution of servo output during manual flight.

The governing part of this firmware is a scheduler which dictates the sequence in which to run the top-level functions of the firmware. These top-level functions, in turn, perform their separate task. Refer to Appendix C and sub-folders *ArduPlane* and *libraries* for a closer look at the entire code. All files in the folder *ArduPlane* is compiled into arduino-executable code. In order to achieve the goal of this project, autonomous net retrieval by way of RTK-GPS as positional reference, additions to the standard version of the firmware were necessary. In the following sections these additions are described, both its location in the software and its purpose.

2.5.3.1 Acquisition of GPS data in APM

The critical relative position reference, provided by the combined effort of the GPS hardware and RTKLIB software, is supplied to the APM via its GPS port. Responsible for parsing the data available at the port for this application. The *AP_GPS_NMEA* library contains algorithms for processing GPS measurements. Other libraries supporting different protocols are located in the same general library folder named *AP_GPS*, found in the Appendix C. As mentioned previously, the standard NMEA protocol was expanded to include the three-dimensional baseline vector along with the quality of the RTK-GPS solution. As a result the APM reads a total of three standard NMEA messages and one custom message. The number given on top of each x in the sentence below is described in Table 2.9. The standard sentences used are described in Section 1.2.4.

$$\$ \underbrace{GPENU}_{Name\ of\ sentence} \ ,\overset{1}{x},\overset{2}{x},\overset{3}{x},\overset{4}{x},\overset{5}{x},\overset{6}{x},\overset{7}{x},\overset{8}{x},\overset{9}{x},\overset{10}{x},\overset{11}{x},\overset{12}{x},\overset{13}{x},\overset{14}{x}$$

Position	Description
1	East component of baseline vector (in meters)
2	North component of baseline vector (in meters)
3	Up component of baseline vector (in meters)
4	Solution quality
5	Number of satllites
6	Standard deviation along east axis (in meters)
7	Standard deviation along north axis (in meters)
8	Standard deviation along up axis (in meters)
9	Standard deviation in east-north plane (in meters)
10	Standard deviation in north-up plane (in meters)
11	Standard deviation in up-east plane (in meters)
12	Age of differential (in seconds)
13	Ambiguity resolution validation ratio
14	Checksum

Table 2.9: Contents of custom message embedded in the stream from RTKLIB

2.5.3.2 Target offset relative to base station antenna

As the base station antenna cannot be placed in the center of the net, offsets in the form of floating point variables were added to library folder *AP_FA_NAV_Control*. For ease of use they can be changed during flight via the MP environment to accomodate several target locations. A pure addition of distances to the components of the baseline vector only applies to the case of stationary target location. In the case where the net used for retrieval is mounted on a moving platform, a rotation between the body-frame and ENU-frame is required (see Section 1.5.4) before addition to the baseline vector. Rotation about any of the three body axes would result in a change in the 3D offset vector.

Returning to static-net retrieval, the offset in ENU baseline vector of target relative to base station antenna is determined by placing the airframe on the ground at the location of the target net. Switching on the payload and acquiring a fixed RTK-GPS solution yields an accurate baseline vector which during flight is used as fixed offsets to obtain the real-time position of the UAV relative to the target. A similar procedure may be applied on a moving platform for obtaining the lever arm between target net and base station antenna in the body frame. A requirement is that the vessel is pointing directly north at the moment of sampling the fixed RTK-GPS baseline vector.

Distance	Variable name
d_1	NAVFA_OFFSET_E
d_2	NAVFA_OFFSET_N
d_3	NAVFA_OFFSET_U
d_4	HGTFA_ZERO_ALT

Table 2.10: Relates distances of Figure 2.11 and Figure 2.12 to variables listed in the APM parameter file.

Figure 2.11 and Figure 2.12 indicate the different distances implemented as user input through the parameter file of the APM, its interface being the ground control station software MP. In Figure 2.11 the distance d_1 indicates the relative distance between target and base station antenna in the East-direction and d_2 indicates the relative distance in the North-direction. Moving on to Figure 2.12, d_3 indicates the center of the net in the vertical direction (at ground-level at the location of the target in the NE-plane) and d_4 is the vertical offset to zero the Up-component of the baseline vector at ground level of target location. See Table 2.10 for description on how the distances in Figure 2.11 and Figure 2.12 relates to variable names in the parameter list of APM.

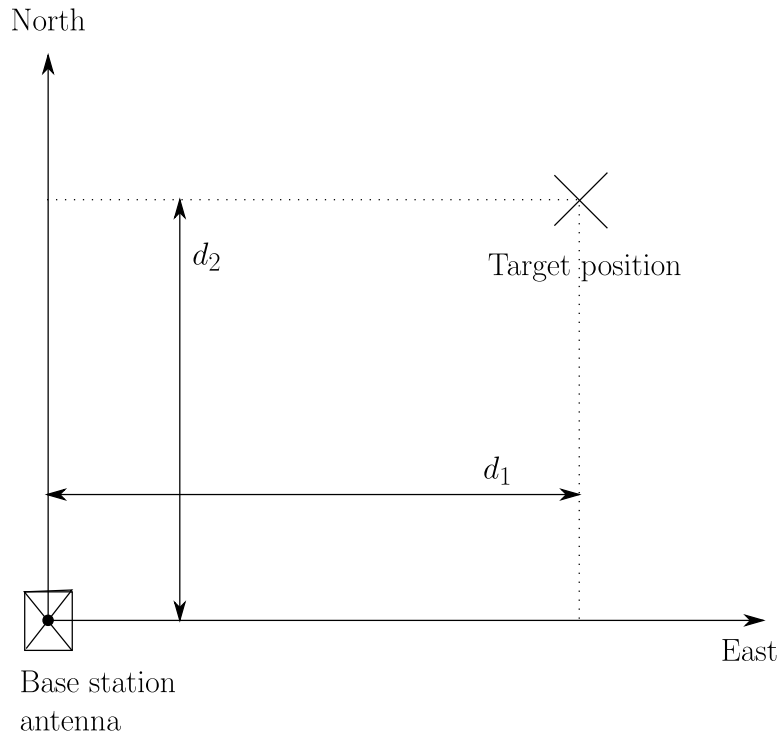


Figure 2.11: Offsets in the North/East-plane of the ENU-frame.

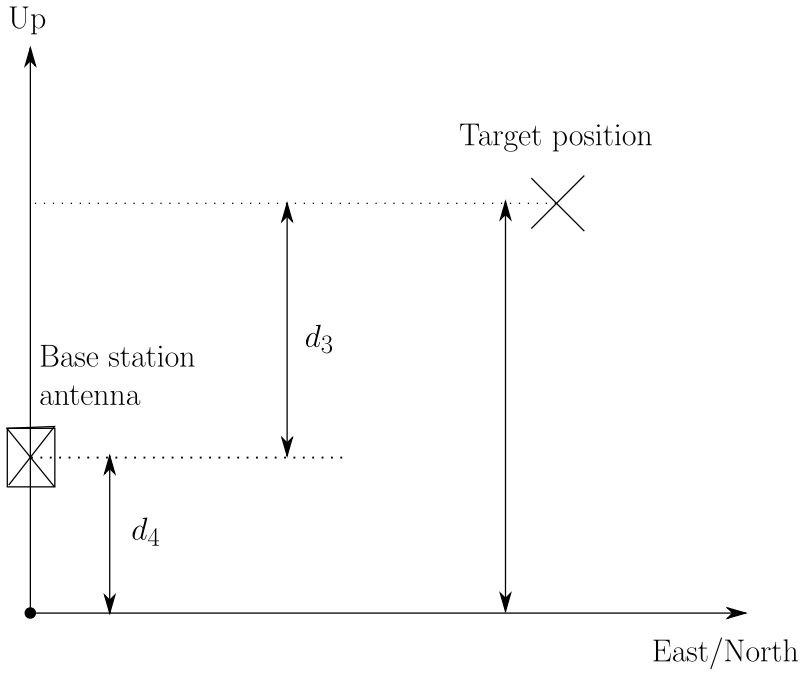


Figure 2.12: Offsets in the Up-direction of the ENU-frame.

2.5.3.3 Longitudinal controller for final approach

Upon entering LAND mode the standard version firmware would proceed with the same longitudinal controller, namely the TECS controller for determining throttle and pitch commands to guide it safely to the ground, whether it be an airstrip or an open, flat field.

As the TECS controller is designed to minimize the use of energy, it's not suited well for navigation during FA. With the additions made to achieve autonomous net-retrieval, this was changed. Although both the standard longitudinal controller and the custom one inherits from the same class, the throttle and pitch setpoints are now generated by a PID controller as opposed to the TECS controller. Considering the simplicity and robustness of the PID controller, and the lack of a mathematical model describing the UAV, the switch was natural.

A new library folder, named AP_FA_HGT_Control (see the *libraries*-folder Appendix C), was added to contain the functions supporting the new longitudinal controller. Its main purpose is to return setpoints for the PID controllers governing the pitch angle and throttle output. The top-level PID controllers, for pitch and throttle, get their error signal from Equation 2.5.1 and Equation 2.5.2 or Equation 2.5.3, respectively. Depending on the configuration of the APM, whether an airspeed sensor is available and in use, either Equation 2.5.2 or Equation 2.5.3 is applied.

$$e_{pitch}(t) = h_d - h_{UAV} \tag{2.5.1}$$

Where $e_{pitch}(t)$ is the error between desired and actual height.

$$e_{throttle}(t) = V_{setp,air} - V_{air} \tag{2.5.2}$$
$$= V_{setp,ground} - V_{ground} \tag{2.5.3}$$

Where $e_{throttle}$ is the error in speed, either airspeed or groundspeed. Two variations of throttle PI controllers were considered and implemented. The first returns the percentage of throttle desired at a given error based on Equation 2.5.4

$$\delta_t = k_p e_{throttle} + \frac{k_i}{s} e_{throttle} \tag{2.5.4}$$
$$\tag{2.5.5}$$

Where δ_t is the throttle output, k_p, k_i are controller gains. It was discovered that as the error approached zero this would lead to zero throttle output, apart from the integral portion of the output. This would lead to a less-than satisfactory behaviour, likely resulting in an offset in desired speed. A slightly different approach, where the throttle output was carried over from the previous calculation was chosen [4]. A PI controller then calculates the size of the increment based on the error. The performance of this scheme is also dependent on how often the calculation is run. Equation 2.5.6 shows the control scheme.

$$\delta_t = \delta_t^* + k_p e_{throttle} + \frac{k_i}{s} e_{throttle} \tag{2.5.6}$$
$$\tag{2.5.7}$$

Where $delta_t^*$ is the output throttle from the previous iteration. A visual display of the desired height h_d, providing a setpoint to the top-level pitch controller, is given in

Figure 2.13 and Figure 2.14. Both schemes provide a viable height profile for the FA, although the Figure 2.14 was preferred for its simplicity. Considering Figure 2.13, the initial slope of descent is determined by the relative altitude between Checkpoint 0 and 1 and the distance between the same checkpoints in the NE-plane.

Distance	Variable name
h_1	Difference between WP altitude and altitude of checkpoint 1
h_2	Vertical distance from center of target to Checkpoint 1 and 2
h	Vertical distance from target center to waypoint height of last waypoint given through MP
d_1	Distance from target center to the position of checkpoint 2 projected onto the East-North plane
d_2	Distance between checkpoint 1 and 2 projected onto the East-North plane
d_3	Distance between checkpoint 0 and 1 projected onto the East-North plane
d_4	Horizontal distance between checkpoint 1 and target projected onto the East-North plane
d_5	Distance between checkpoint 0 and 1 projected onto the East-North plane

Table 2.11: Describes variables mentioned in Figure 2.13 and Figure 2.14.

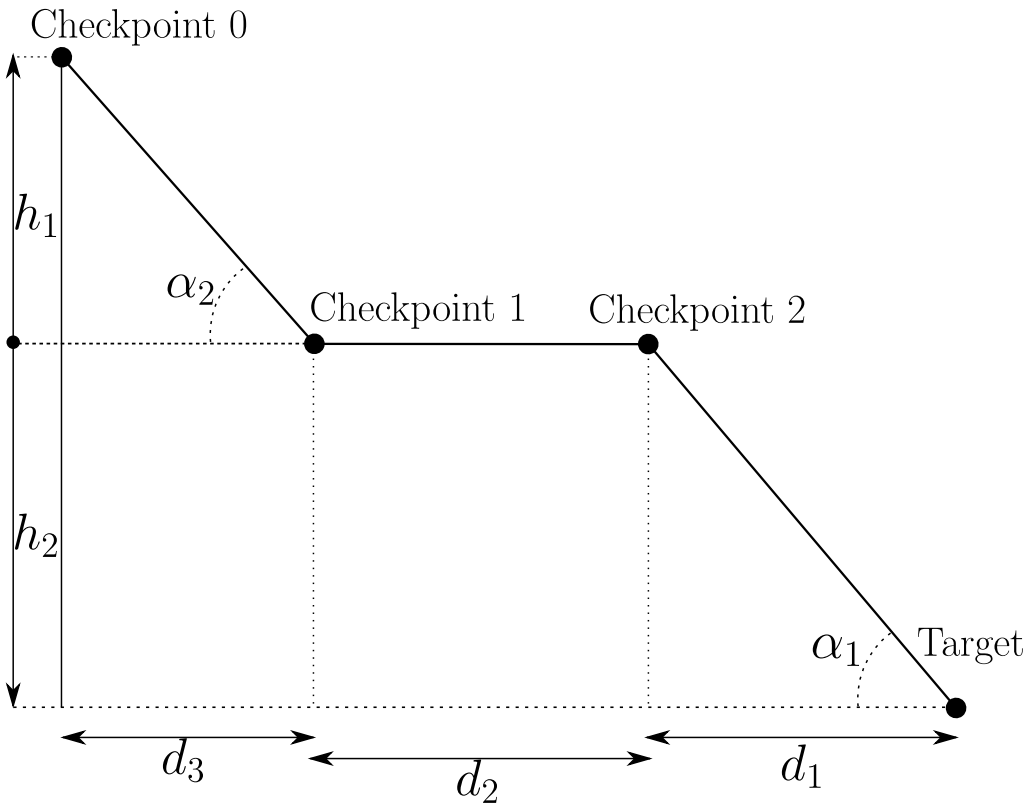


Figure 2.13: Option 1

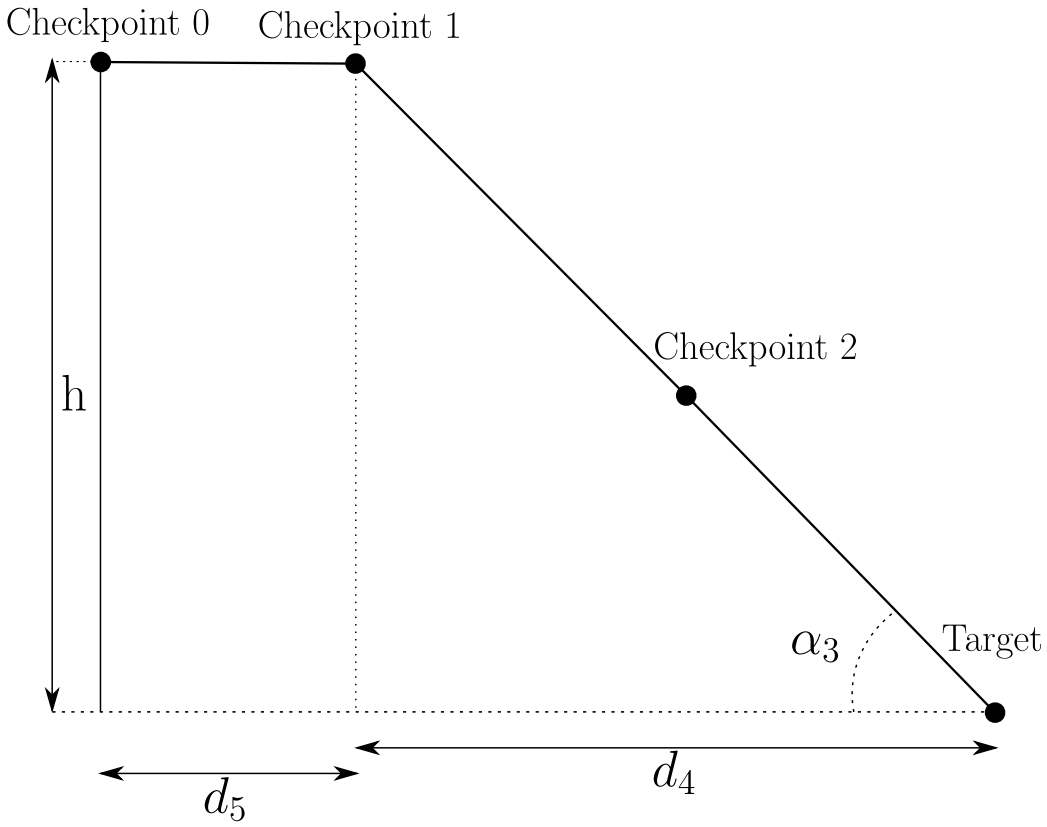


Figure 2.14: Option 2

2.5.3.4 Lateral controller for final approach

The lateral controller used for FA is based on the existing implementation of the L_1 controller (described in Section 1.3.2), with some modifications to be able to utilize the new baseline measurements given from the RTK-GPS solution. As the existing implementation of the L_1 controller converts WP positions and UAV position into a local frame for navigation, not much was needed to convert the controller into utilizing the baseline values instead.

The main difference, however, is that the accurate moving-base RTK-GPS solution only exists in the local ENU frame and not in the ECEF frame. This is because the geodetic ECEF position, while using moving-base positioning mode, has the same accuracy as a single GPS receiver [34]. Because of this, it is not possible to use the standard WP setup, as the RTK-GPS solution would have to be transformed (described in Section 1.5.5) into geodetic ECEF position to be used in the standard controller. This will lead to inaccuracies. Instead, the CPs for FA are calculated in the ENU frame based on design parameters and the UAV position when the landing command is given.

There were two solutions developed for the FA lateral controller, one using linear paths between all points and one using a semi-circle between checkpoint one and checkpoint two.

The implementation of this lateral controller can be found in AP_FA_NAV_Control. See Appendix C - ArduPlane - Static target - libraries, as well as finalApproach_nav.ino in

Appendix C - ArduPlane - Static target - ArduPlane.

Calculating checkpoints - linear path

The number of checkpoints were chosen as two, as this was found through HIL simulation to be a sufficient amount for a repeatable landing. The two checkpoints are calculated based on the desired landing course. The outline of the algorithm, the illustration and the implementation can be seen in Algorithm 2.5.2, Figure 2.15 and finalApproach.ino in Appendix C Software - Static Target - ArduPlane, respectively.

Algorithm 2.5.2: ROTATE FIRST CHECKPOINT($baseLine, landCourse, d_{lp_2}$)

comment: Here baseLine and landCourse both point from origin and out

comment: Look at Figure 2.15 for the basis of this algorithm

θ_l = Angle between baseLine and landCourse
if $\theta_l > 45°$
  then $\theta_l = 45°$
  else if $\theta_l < 45°$ **and** $baseLine.length < d_{lp_2}$
  then $\theta_l = 45°$
  else $\theta_l = \theta_l$
Rotate LP_2Vec
comment: Which way this vector is rotated is explained in Appendix D.2

The 'else if' requirement is included because if the UAV is close to the landing target when FA starts, and has $\theta_l < abs(45°)$, which means the corresponding flight path would include a turn $> 90°$ (See Figure D.1 for an illustration).

As seen from Algorithm 2.5.2 and Figure 2.15, the first checkpoint position is based on the distance between checkpoint two and checkpoint one ($d_{p_2p_1}$ in Figure 2.15), the distance between checkpoint two and the landing target (d_{lp_2} in Figure 2.15) and the rotate angle (θ_l) found in Algorithm 2.5.2. The reason as to why θ_l is limited to 45° is that the smaller angle between the vectors LP_2Vec and P_2P_1Vec, the quicker the UAV is able to converge to the new desired lateral flight path.

Parameter	Description
θ_l	Angle between landCourse and baseLine
P_2P_1Vec	Vector from checkpoint two to one (also desired flight path)
LP_2Vec	Vector from landing target to checkpoint two (also desired flight path)
$d_{p_2p_1}$	Length of P_2P_1Vec
d_{lp_2}	Length of LP_2Vec

Table 2.12: Description of Figure 2.15 with distance parameters

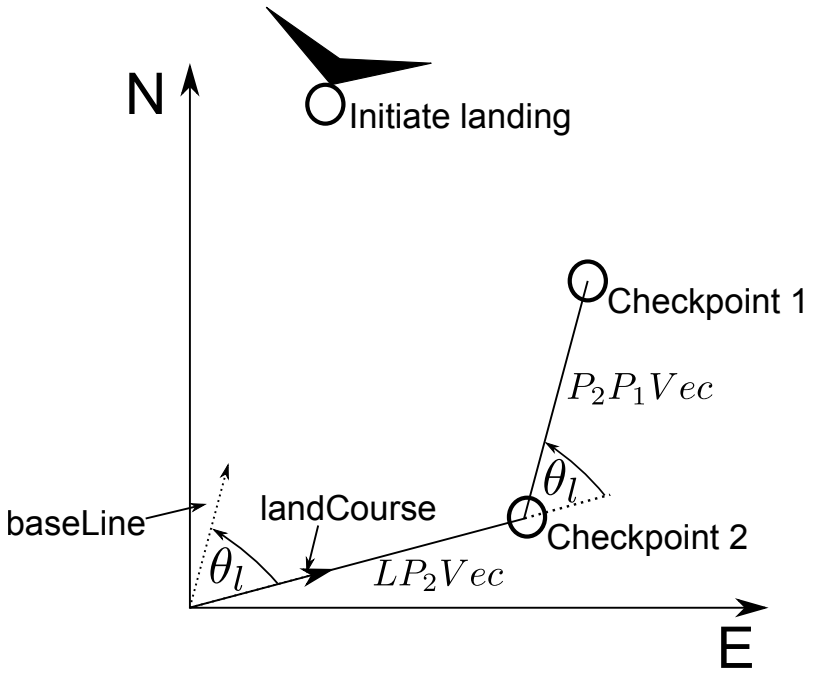


Figure 2.15: Rotating first checkpoint - linear path

Here the inverse of P_2P_1Vec and LP_2Vec becomes the desired flight path. Both d_{lp_2} and $d_{p_2p_1}$ are design variables to make the FA checkpoints fit the area it's being used in.

Calculating checkpoints - circular path

As opposed to the previously described way to calculate checkpoint, this method utilizes a circle as a path to converge to CP two from CP one. As a circular path is used between checkpoint two and checkpoint one, a larger θ_l is acceptable. Therefore, the algorithm now used to calculate θ_l can be seen in Algorithm 2.5.3.

Algorithm 2.5.3: ROTATE FIRST CHECKPOINT($baseLine, landCourse, d_{lp_2}$)

comment: Here baseLine and landCourse both point from origin and out

comment: Look at Figure 2.15 for the basis of this algorithm

θ_l = Angle between baseLine and landCourse
if $\theta_l > 90°$
  then $\theta_l = 90°$
  else if $\theta_l < 90°$ **and** $baseLine.length < d_{lp_2}$
  then $\theta_l = 90°$
  else $\theta_l = \theta_l$
Rotate LP_2Vec
comment: Which way this vector is rotated is explained in Appendix D.2

The reason as to why $\max(\theta_l)$ is increased is because the tangent of the semi-circle in

CP two is LP_2Vec, meaning as long as the UAV can follow the arced path, it will immediately be able to follow the linear path described by LP_2Vec. The new parameter description and an illustration can be seen in Table 2.13 and Figure 2.16 respectively.

Parameter	Description
θ_l	Angle between landCourse and baseLine
P_2P_1Vec	Vector from checkpoint two to one (also desired flight path)
FP_{sc}	Arc of the circle (also desired flight path)
d_{sc}	Diameter of the circle
c	Chord of the circle with specified θ_l
d_{lp_2}	Length of LP_2Vec

Table 2.13: Description of Figure 2.16 with distance parameters

The position of CP one now depends on θ_l and the radius, $\left(\frac{d_{sc}}{2}\right)$, of the circle used for convergence to CP two. The calculation of the chord distance is found in Appendix D.1.

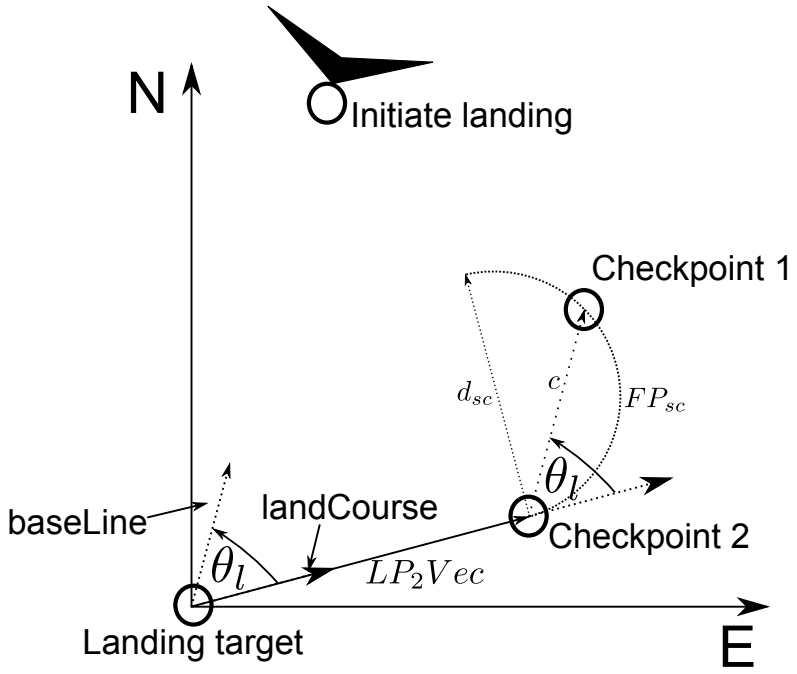


Figure 2.16: Rotating first checkpoint - semi-circular path

Choosing between linear- and circular path

The option to choose which path to follow during FA was added to give more freedom when choosing how to converge to the last CP, as the environment around the landing target might change. The choice as to which lateral path to use highly depends on the UAV's previous flight path, and the environment. Although both the linear and the circular path can handle different wind conditions and topography, one might be preferable over the other in various situations.

There are different advantages to both. For the linear path, the following points are some of the advantages:

- The UAV settles earlier at the desired landing course
- The UAV follows a controlled descent in a near straight line

Whereas for the circular path, advantages are:

- Ability to better handle a large difference in course vs landing course
- Improved FA control due to low roll angles during course changes
- Greater airspeed reduction without headwind as the descent is more gradual

The linear path works best when there's an open environment where the UAV can fly during FA, as the UAV can stay close to the desired landing course and follow a close to straight path from start of FA to landing. However, if FA is started with a course opposite of the desired landing course, there will be sharp turns and the area needed for FA will be large, as the distance between the CPs has to be sufficient such that the UAV is on the desired path when changing CPs. The circular path on the other hand, requires less area as it keeps a constant circular motion from checkpoint one to checkpoint two, and the path between landing target and CP two is a tangent from the circular path followed.

Converting desired acceleration to roll angle

Looking at Figure 2.17, an expression for the desired roll angle can be found by

$$\tan(\phi) = \frac{a_{s_{cmd}}}{g} \tag{2.5.8}$$

$$\phi = \arctan\left(\frac{a_{s_{cmd}}}{g}\right) \tag{2.5.9}$$

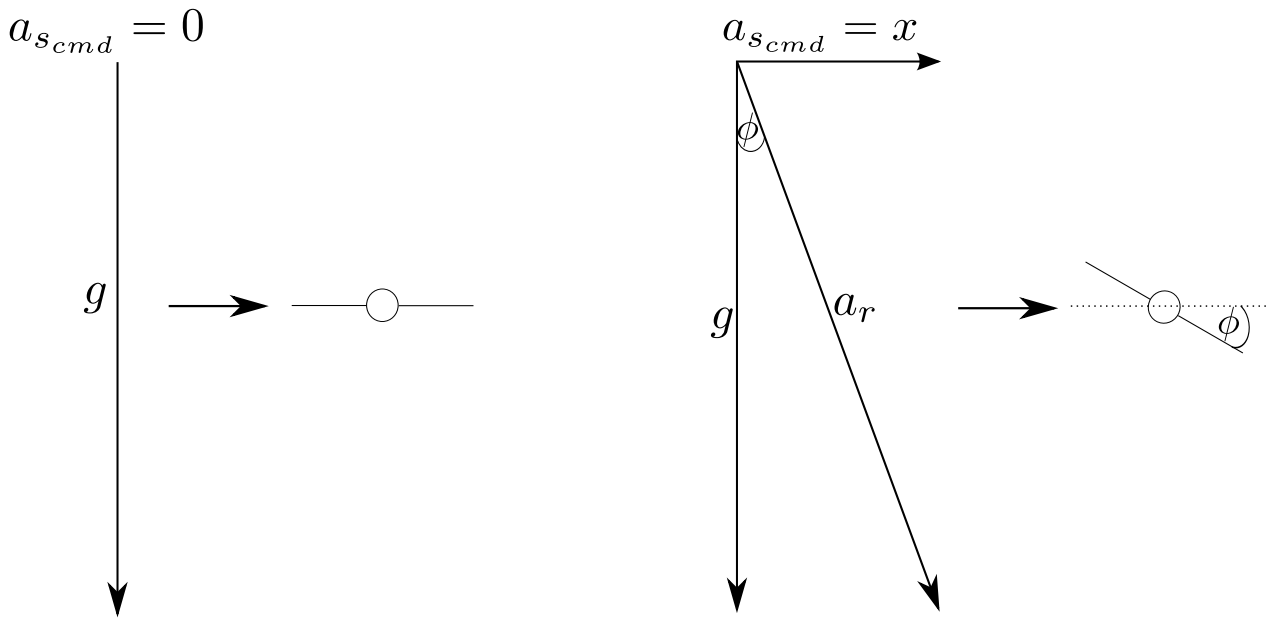


Figure 2.17: Commanded roll angle with and without desired acceleration

where g is the gravitational constant, ϕ is the desired roll angle and $a_{s_{cmd}}$ is the desired acceleration found in Equation 1.3.6 [29]. This equation follows from Figure 2.17. To obtain acceleration in the desired direction, a roll angle (found in Equation 2.5.9) is necessary. However, this is not enough. As the X8 is a flying wing, a high-valued desired roll command combined with an aircraft that is either in a severe positive or negative pitch angle, can result in the aircraft spinning out of control, due to lack of lift. Because of this, a parameter limiting the desired roll angle is implemented in the standard L_1 controller. The equation used for translation of the desired acceleration to roll becomes:

$$\phi = \cos(\theta) \arctan\left(\frac{a_{s_{cmd}}}{g}\right) \tag{2.5.10}$$

where θ is the current pitch angle, and $\cos(\theta)$ limits the value of the desired roll angle ϕ.

2.5.3.5 Parsing NMEA message

The APM has several options when it comes to protocol of GPS data, which may vary depending on hardware supplier. RTKLIB provides the option of streaming RTK-GPS solution in the well known NMEA 0183 protocol. Being limited to one GPS port on the APM, when both geodetic ECEF coordinates and RTK-GPS solution were necessary for regular AUTO mode and LAND mode, respectively, a decision was made to create a new message in the NMEA protocol (see Section 2.5.5.1). Adding a message to the input stream required additions to the standard GPS parser.

The standard GPS parser is modified to include the newly created message. The message is recognized by the initial letters $GPENU (described in Section 2.5.3.1, and the variables were extracted according to their position in the message.

2.5.3.6 Resetting of logic for re-entry to FA

After a successful net retrieval, or at the end of a successful test, after which the pilot takes control of the aircraft, a reset of certain parameters is necessary for restarting AUTO mode at a certain waypoint. This is performed in the function change_command(), located in Appendix C - ArduPlane - commands_process.ino. The reset allows:

- Switch from custom controllers to standard controllers for pitch, roll and throttle
- Reset of CPs for visualizing in MP
- Makes sure a re-entry into LAND mode yields intended response from the guidance system each time. This is relevant after a successful net-retrieval or during testing (post MANUAL mode).

2.5.3.7 Evasive maneuver

In case the UAV diverges from the desired path during FA, both horizontally and/or vertically, an evasive maneuver has to be performed to avoid getting into a potentially hazardous situation. The criteria which has to be fulfilled for an evasive maneuver to be performed were set as:

- Cross-track error above certain threshold
- Altitude error above certain threshold
- RTK-GPS solution quality during final stages of FA
- Course over ground vs heading
- Ignore evasive maneuvers

Cross-track error criterion

As the landing target (in this case, a net) has a limited area where the UAV can land, a constraint has to be set on the distance from the desired horizontal path. In the test setup for the landing attempts, a net of 5x3 m were used (described in Section 2.2.7). As the X8 has a wing span of 2.1 m (described in Section 2.2.2), the approximate distance from the antenna position to the end of each wing is 1 m. The net is fixed on metal poles which will damage the UAV during landing if hit during landing. For this a cross-track error of ± 1 m was chosen as an acceptable criterion for landing. The pseudocode for the cross-track error is seen in 2.5.4

Algorithm 2.5.4: CROSS-TRACK EVASIVE MANEUVER(cross-track error)

```
Get current cross-track error
   Compare the last 2 seconds of cross-track error
   if last 2 seconds of cross-track error > cross-track error criterion
      Evasive maneuver flag to TRUE
  else Evasive maneuver flag to FALSE
```

The reason the evasive maneuver flag is set by a timeseries of errors, and not just one single cross-track error, is based on the fact that:

- A single gust of wind can push the UAV out of position, triggering an error, even if it has time to correct itself
- RTK-GPS noise can cause a 'spike' in the baseline measurements, leading to a 'false' cross-track error

Altitude error criterion

The net is only 3 m in height, however, as there are no wingspan to take into consideration, an error of ± 1 m can be used here as well. The pseudocode for the altitude error can be seen in 2.5.5

Algorithm 2.5.5: ALTITUDE EVASIVE MANEUVER FLAG(Altitude error)

```
Get current altitude error
   Compare the last 2 seconds of altitude error
   if last 2 seconds of altitude error > altitude error criterion
      Evasive maneuver flag to TRUE
  else Evasive maneuver flag to FALSE
```

The reasoning behind requiring a series of measurements to be outside the error criterion, compared to one single error measurement, is the same as described previously in 'Cross-track error criterion'.

RTK-GPS solution ratio criterion

The ambiguity resolution (AR) ratio is a ratio describing the quality of the RTK-GPS solution. Any value above zero means the integer ambiguity resolution has enough data to start estimating a relative position between the rover and the basestation. The pseudocode for the evasive maneuver flag can be seen in Algorithm 2.5.6

Algorithm 2.5.6: AMBIGUITY RESOLUTION RATIO FLAG(AR_{ratio})

```
Get AR_Ratio
   Compare the last second of AR_ratio
   if 0 ≤ AR_ratio ≤ 1
      Evasive maneuver flag to TRUE
  else Evasive maneuver flag to FALSE
```

The basis for this value can be seen from Figure 2.18. As the AR_{ratio} increases, the standard deviation in the EN plane goes down. Any AR_{ratio} above 1 leads to a low standard deviation in the EN plane, which leads to a high precision baseline that's possible to navigate after, even without a fix solution.

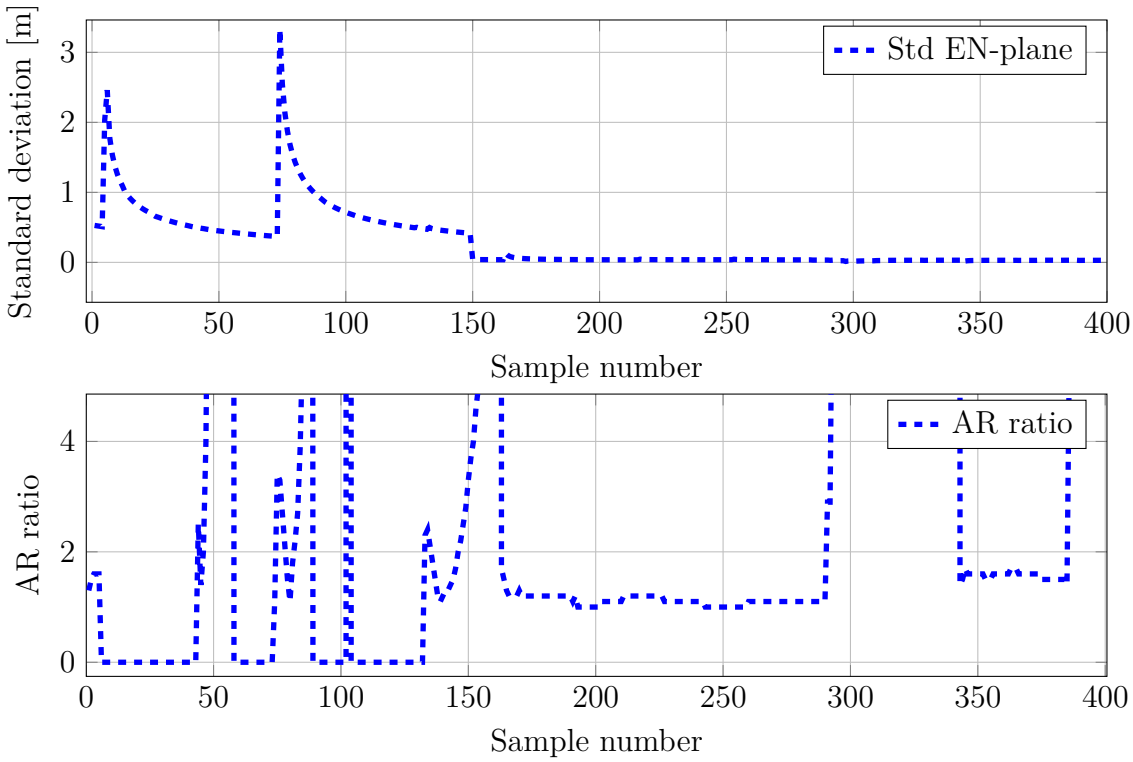


Figure 2.18: Standard deviation in East-North plane and AR ratio versus sample number (sampled at 5 Hz).

Course over ground vs heading criterion

With strong crosswinds during FA, the Course Over Ground (COG) will differ from the UAV's heading. In the extreme cases (see Figure 2.19), the angle between the heading vector and COG vector might be $> 45°$ which would lead to the UAV hitting the net, wing first. As this should be avoided, to reduce the stress on the UAV fuselage, the algorithm shown in 2.5.7 is implemented

Algorithm 2.5.7: COG vs Heading(COG,Heading)

```
Get COG and Heading
   Compare the last 5 samples of COG vs heading
   if all samples >= Threshold
      Evasive maneuver flag to TRUE
  else Evasive maneuver flag to FALSE
```

As wind gusts can occur, which may cause a temporary large angle between COG and heading, a series of samples are again used to decide whether or not an evasive maneuver has to be performed.

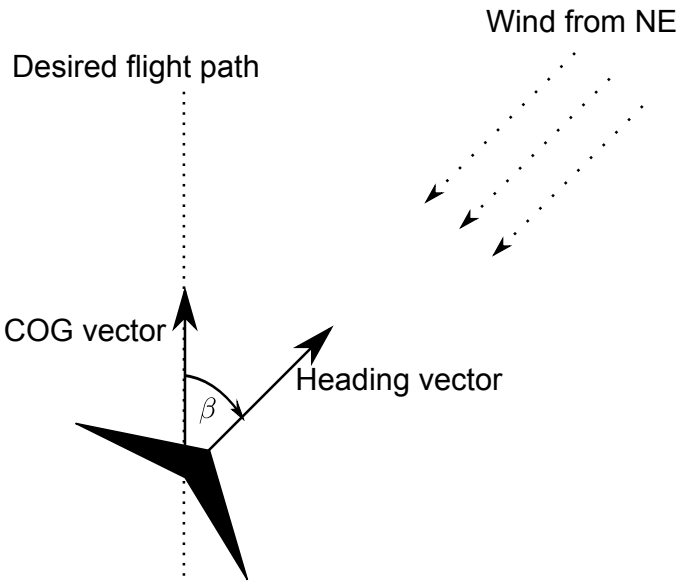


Figure 2.19: Course over ground vs heading - NE plane

Ignoring evasive maneuver

There is also the option to ignore evasive maneuvers when landing. The reasoning for this is that the UAV might operate in weather/GPS environment where being constrained by the evasive maneuver criteria previously set, will never lead to a successful landing. There are multiple scenarios where this option might be viable to use, for example:

- Sudden weather change during UAV mission
- Low battery on UAV (have to land, or risk damaging valuable payload

Addressing 'Sudden weather change' first. Sudden change in both wind strength and direction can mean that the UAV is limited by its physical capabilities when it comes to FA navigation, as the X8 only has 2 control surfaces. This means the UAV can land approximately in the net, but violating the constraints set by the evasive maneuver criteria, which may lead to damage of UAV fuselage. If this scenario occurs, the different options has to be weighed against each other. Either try and force landing in the landing target, risking damage to the UAV fuselage, or loiter/fly in a new route to try FA again. Both of these options can lead to the second reason for this option. If the option of loiter/new route to FA is chosen, the UAV might end up repeatedly attempting but always violating constraints, which will drain the battery to the point where a loss of power to the actuators is a real possibility, in which case only damage to the fuselage will be preferable to damaging both payload and fuselage.

2.5.3.8 Smoothing altitude

To minimize the effect an inaccurate height measurement has on the PID height controller during FA when a fixed RTK-GPS solution is unavailable, an average value of height is used. An average value over a set number of samples reduces the severe changes in altitude

error. Equation 2.5.11 shows the implementation

$$alt_{avg} = \frac{1}{4}\sum_{t=1}^{4} alt(t) \tag{2.5.11}$$

where $alt(t)$ is the altitude of the UAV at time t. The four most recent UAV altitudes were used.

2.5.4 Ser2Net

Ser2Net (Serial to network) is used to stream the telemetry (GPS position/attitude angles etc.) between the UAV and the GCS. This program takes the serial stream from the APM and converts it to a network stream (in our case, a Transmission Control Protocol (TCP) stream which can be read by MP). With this, the telemetry from the APM can be sent over the wireless connection (described in Section 2.2.5.1) already established between the payload and GCS. A simple diagram of this can be seen in Figure 2.20

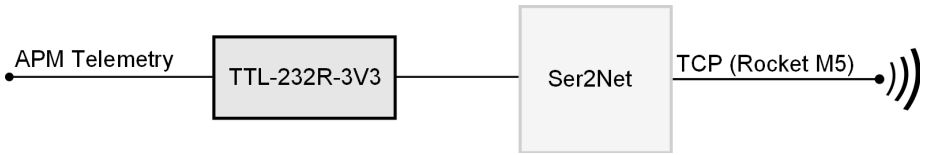


Figure 2.20: Diagram of telemetry connection ⇔ TCP through ser2net

2.5.5 RTKLIB

RTKLIB is an open source program package for standard and precise positioning with GNSS (global navigation satellite system). RTKLIB consists of a portable program library and several Application Programs (APs) utilizing the library [34].

2.5.5.1 Changes made

As RTKLIB lacks the possibility to stream ENU values combined with standard NMEA 0183 messages (described in Section 1.2.4), the standard NMEA stream in RTKRCV, an AP for linux, was customized to contain a GPENU sentence. The reasoning as to why the NMEA stream was modified compared to just using the already existing ENU stream is that the ENU stream does not contain geodetic ECEF positions, which are needed during normal auto-flight.

Chapter 3

Results and discussion

Data collected in HIL simulations, as well as real-life experimental testing, is presented in this chapter to substantiate the functionality of the system.

3.1 HIL-tests

Hardware-in-loop (HIL) simulation allows extensive testing for any desired hardware without the risk of damaging valuable equipment by prematurely testing in the real world. As tuning of a PID-controller for flight control, or weighing different engines of an aircraft up against each other can be both time consuming, and potentially result in failure. This is why HIL simulation is used as a test bench before the changes/equipment are applied to the real world application.

For realistic HIL simulation however, an accurate model of the aircraft and the different environmental parameters is required. This ranges from moments of inertia, to accurately deciding how much a propeller affects the aircraft when full throttle is applied.

As a mathematical model for the X8 Flying wing is not available, HIL simulations have been performed with the intent to show proof of concept for the newly designed final approach controllers, both horizontal and vertical. The logs used for data-plots are telemetry logs, which log the information at $\tilde{0}.5$s intervals, which is the log available when flying in the real world. More accurate logs are available for HIL simulation, but as not all variables are available at the same sampling rate, the telemetry logs were used for creating the figures shown in the result section.

A list of different HIL scenarios which will be tested follows.

- No wind, static target
- Wind disturbance, static target
- Wind disturbance and GPS noise, static target
- No wind, moving target

- Wind, moving target
- Wind, Evasive maneuver, static target

All HIL tests followed the same procedure; takeoff, waypoint navigation, final approach, land with specified course.

3.1.1 HIL Setup

The outline of how the connections are made during a HIL simulation is illustrated in Figure 3.1 and the specific connection types are seen in Table 3.1. MP acts as a central hub where all information passes through. When a command is sent from the GCS using MP this command is sent to the APM. The APM then decides an action based on the command, which is sent back to MP. MP then relays this command to XPlane, which in turn converts the command into action. Normal status information (IMU, GPS etc) from the UAV during simulation is sent from XPlane to APM via MP.

Connection nr	1	2	3
Connection type	MAVLink	Command	UDP

Table 3.1: Connection setup for Figure 3.1

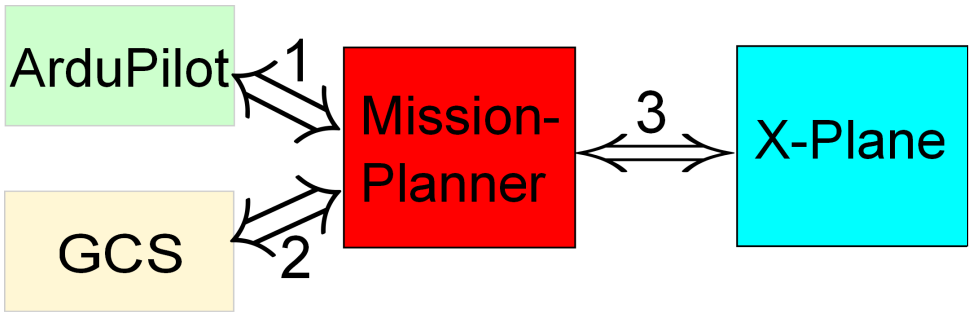


Figure 3.1: Outline of HIL simulation

3.1.2 HIL Simulation - no wind

The first HIL simulation was set up with no wind disturbance, and with a baseline signal without noise (fixed RTK-GPS solution). The reasoning for this is to show consistency in the FA navigational controllers.

A plot of the flightpath in 3D, with desired flight path and waypoints can be seen in Figure 3.2, with the LT at (0,0,0). This figure illustrates how the UAV flies with desired altitude and waypoints. The oscillations in height seen during mid-flight is the TECS controller not tuned correctly, as this is a simulation to mainly show the FA controllers, the tuning of controllers pre-FA has been neglected. As seen in Figure 3.2, the UAV performs a takeoff with a locked heading, steadily increasing altitude until it reaches its set altitude, then starts normal navigation, and lastly descends for landing.

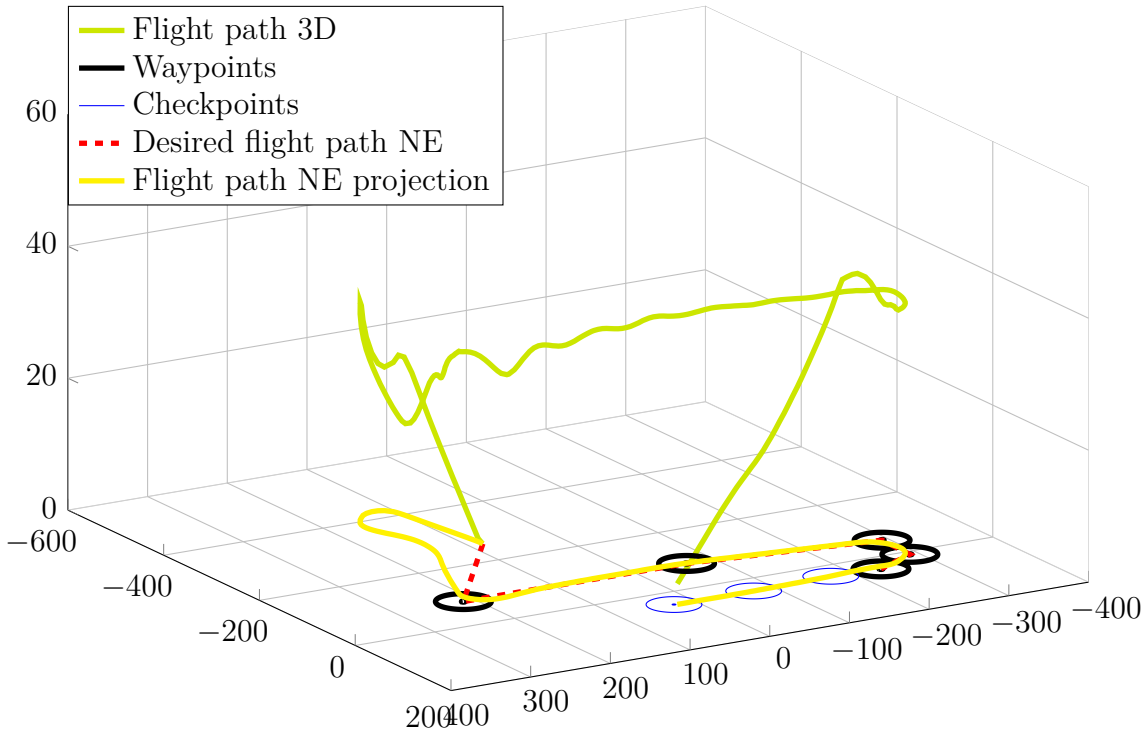


Figure 3.2: Flight path - No wind disturbance

Figure 3.3 and Figure 3.4 shows the altitude- and crosstrack errors of FA for 5 consecutive flights, respectively. Both these figures show that all errors consistently converge towards the desired value. This shows that the controllers are able to repeat the flight multiple times, as necessary.

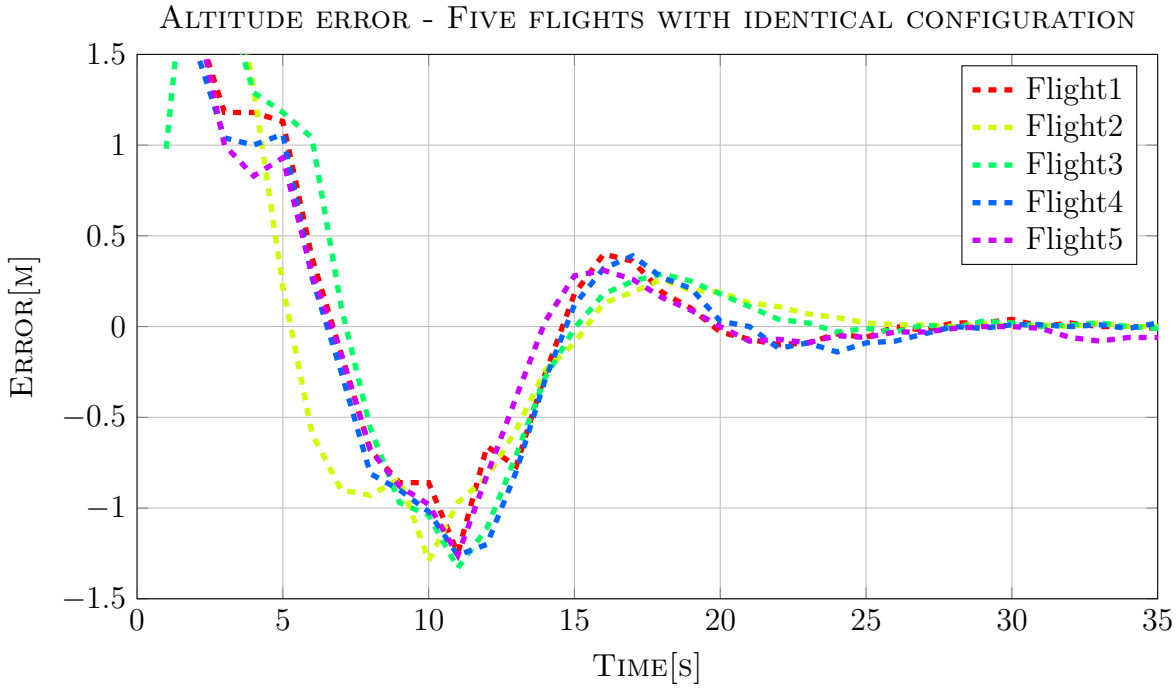


Figure 3.3: Altitude error of 5 consecutive flights - No wind disturbance

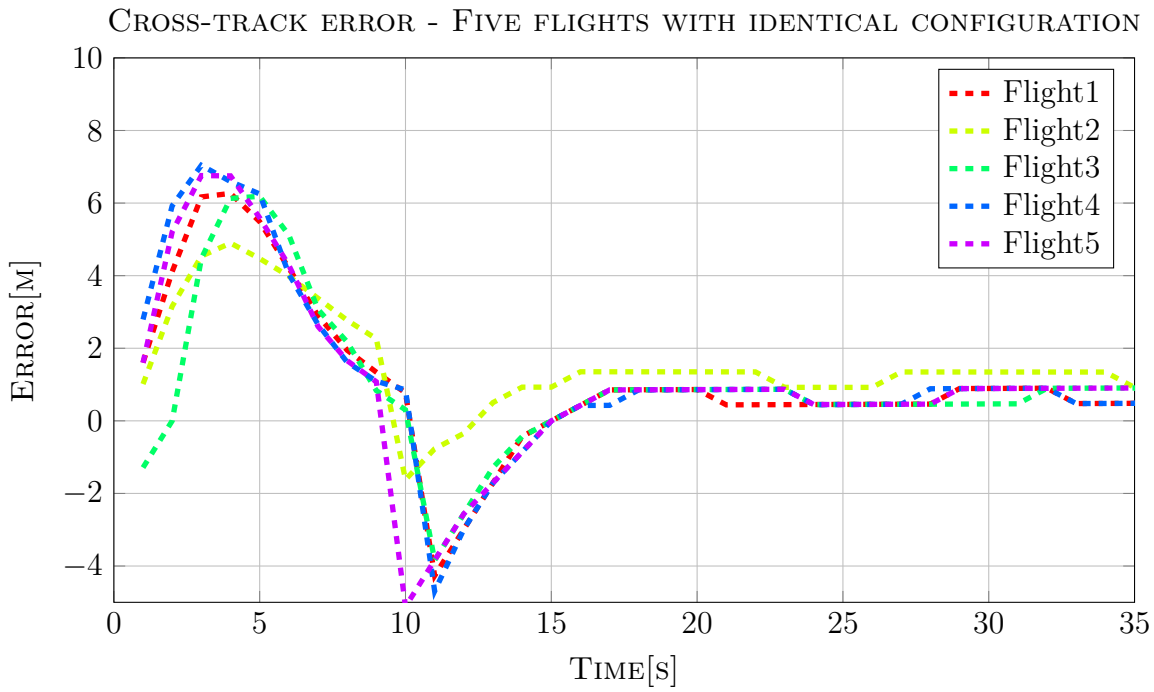


Figure 3.4: Crosstrack error of 5 consecutive flights - No wind disturbance

3.1.3 HIL Simulation - wind/turbulence

Using the same test setup as described in Section 3.1, with a 5 knot wind combined with 5 knot turbulence from south east was added. This gives us a slight head/crosswind during landing, which is what can be expected during normal landing. It can be argued that 5 knot is not a big disturbance, but the limit as to what the XPlane model can handle is significantly lower then what was experienced during real flight. With a more accurate model this would not have been an issue.

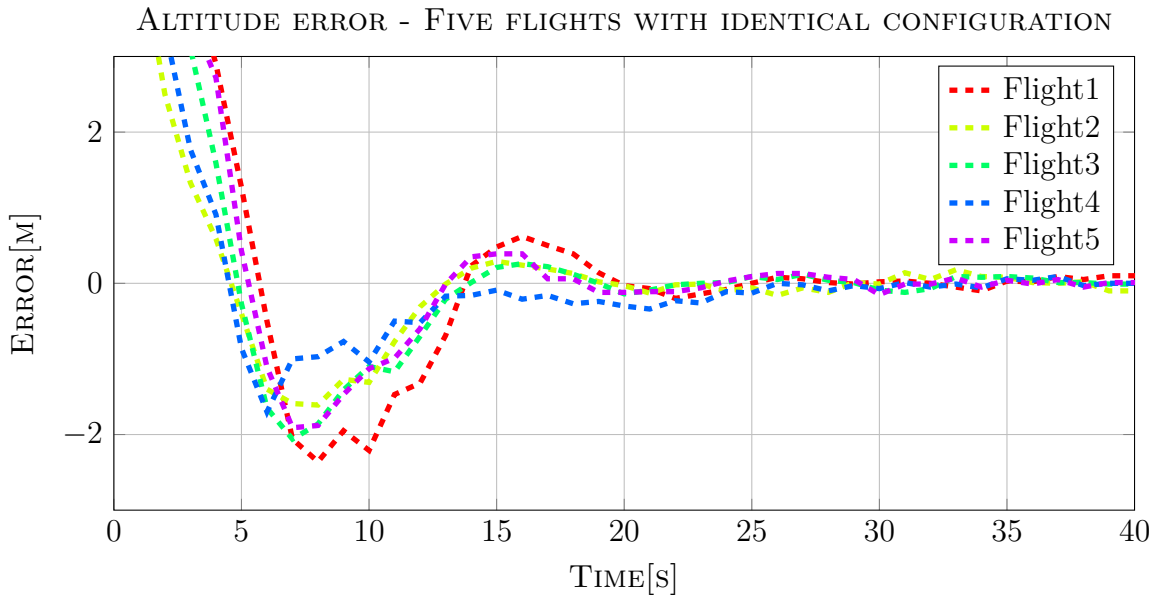


Figure 3.5: Altitude error of 5 consecutive flights - with wind disturbance

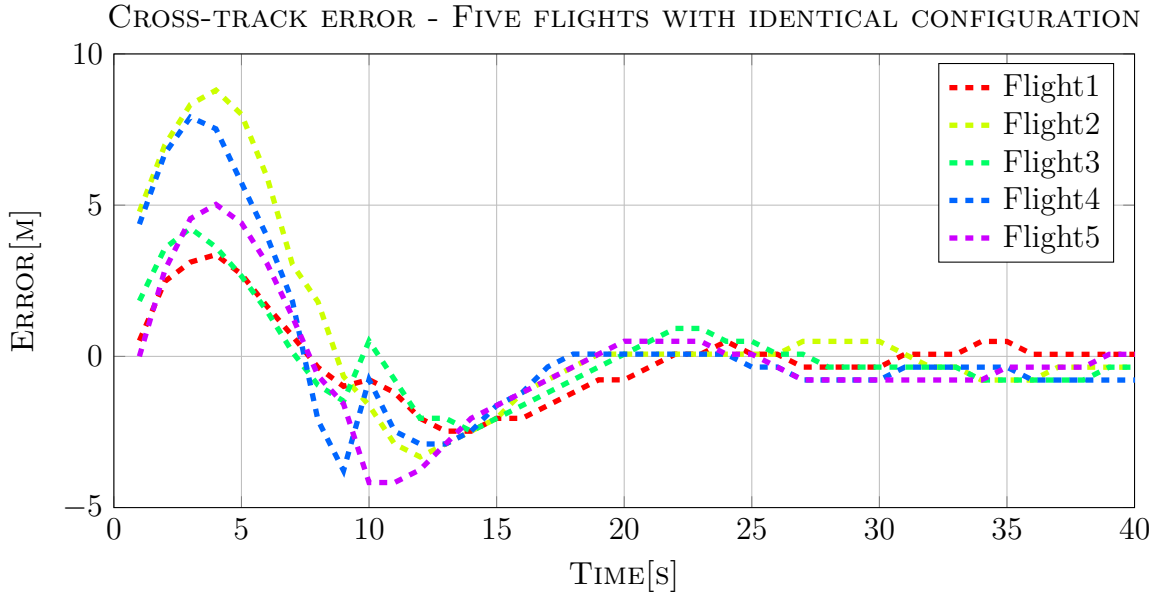


Figure 3.6: Crosstrack error of 5 consecutive flights - with wind disturbance

The resulting FA altitude- and cross-track error plots can be seen in Figure 3.5 and Fig-

ure 3.6, respectively. As seen from both plots, the flights are consistently even and similar to the no-wind tests, as all the 5 tests performed, is showing the ability to compensate for the wind disturbances.

3.1.4 HIL Simulation - wind/turbulence and GPS noise

To see how the controllers handle GPS noise (non-fix RTK-GPS solutions) random values were modulated onto the baseline values to mimic low accuracy RTK-GPS solution. The noise modulated onto the baseline were:

- Random values between ±0.6 m in North and East baseline values
- Random values between ±1 m in the Up baseline values

The basis for these values were taken from the approximate standard deviation of a stationary GPS receiver with a float RTK-GPS solution (seen in Figure 3.7). The magnitude of the standard deviation was increased in order to test robustness in case of a lower AR validation ratio value, resulting in a greater standard deviation.

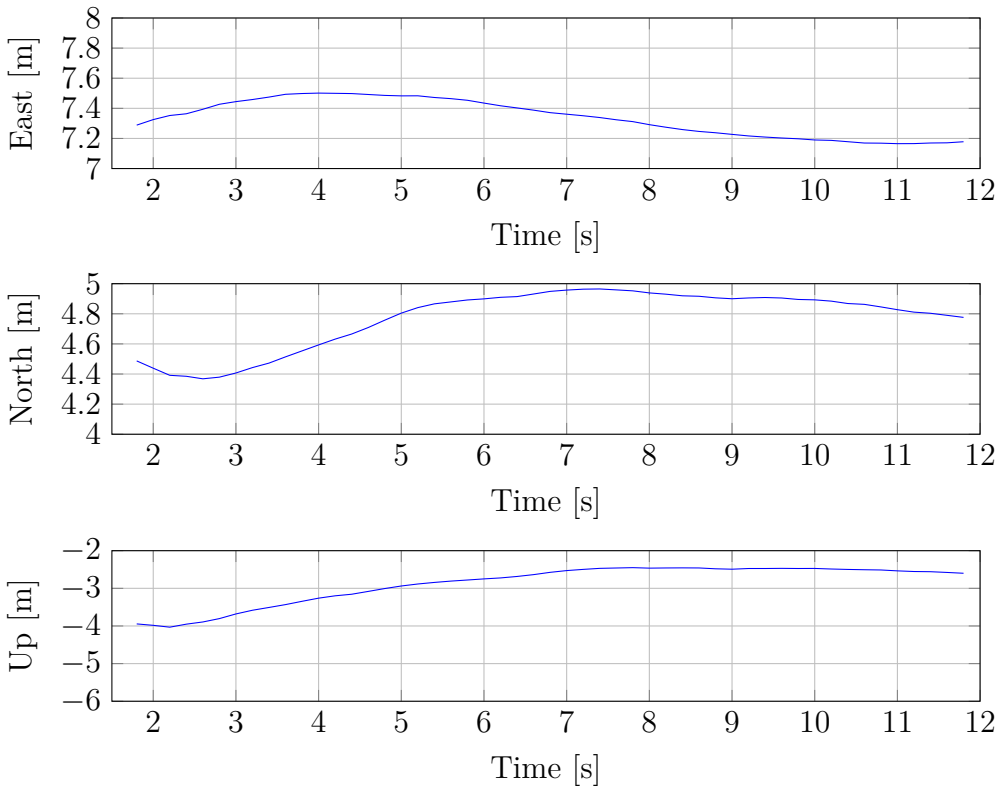


Figure 3.7: All three components of the baseline vector with float RTK-GPS solution and an AR validation ratio of between 1.0 and 1.5.

Using the formulas given in Equation 3.1.1 and Equation 3.1.2 the statistical properties of the baseline components when experiencing float RTK-GPS solution is given in Table 3.2.

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{3.1.1}$$

$$s_N = \left(\frac{1}{N} \sum_{i=1}^{N} (x_i - \bar{x})^2 \right)^{\frac{1}{2}} \tag{3.1.2}$$

Baseline component	Standard deviation (m)	Sample mean (m)
East	0.1206	7.3365
North	0.1949	4.7718
Up	0.5121	-2.8802

Table 3.2: Contains statistical properties of datasets visualized in Figure 3.7.

Looking at the altitude error plot shown in Figure 3.8, the effectiveness of the average altitude described in Section 2.5.3.8 is seen. With noise of ±1 m modulated onto the baseline Up signal, the vertical controller is still able to regulate the altitude error towards zero, and obtain a stable low altitude error, even if it looks a bit noisy.

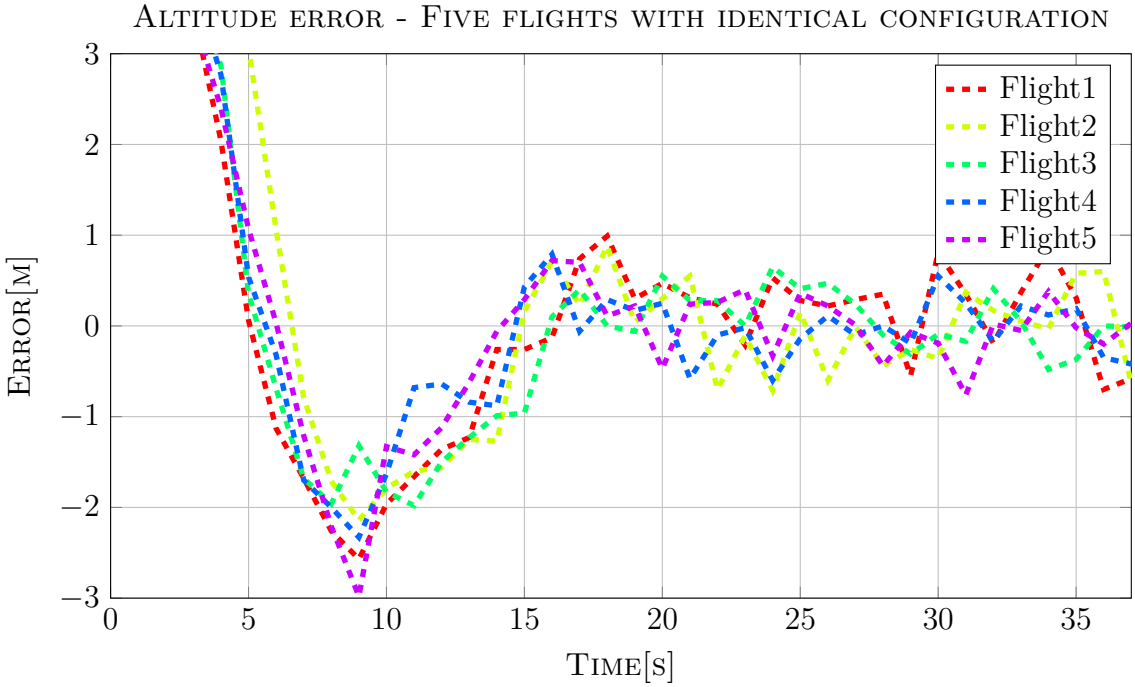


Figure 3.8: Altitude error of 5 consecutive flights - with wind and GPS disturbance

The cross-track error seen in Figure 3.9, even though the signal looks abit more noisy, the overall cross-track error is still low. This is because the lookahead based nonlinear horizontal controller is not that susceptible to noise as i.e. a PID controller might be. Based on the lookahead distance, the impact the noise has on the desired acceleration is low.

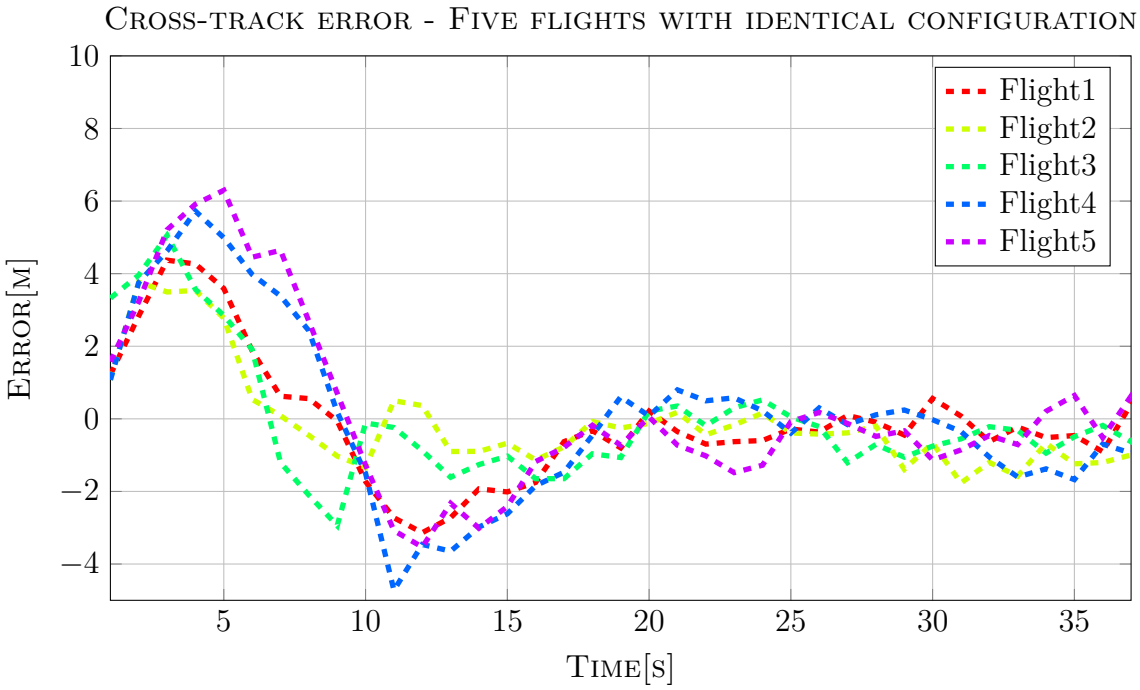


Figure 3.9: Crosstrack error of 5 consecutive flights - with wind and GPS disturbance

3.1.5 Discussion - Static target

Looking at Section 3.1.2, Section 3.1.3 and Section 3.1.4, the modified FA controllers are able to withstand both wind- and GPS disturbances.

Looking at the horizontal controller first, in both Section 3.1.3 and Section 3.1.2 (with and without wind, respectively) the cross-track error is stable and converging to zero at the end of the flight. However, there is a small offset, as is the weak point of this nonlinear controller. Since the controller relies on a desired acceleration to converge to its desired flight path (described in Section 2.5.3.4), the smaller the offset, the lower the acceleration will be required. It's possible to tune for an aggressive controller, such that cross-track error goes towards zero. However, this will lead to a very aggressive controller which might fail when disturbances occur (wind and GPS) earlier in FA. Therefore a compromise has to be made to make sure the controller can safely handle disturbances, as well as getting the lowest cross-track error possible, which is why a small static cross-track error is seen. The result of this is seen in Section 3.1.4, when a GPS disturbance is added. Even if the cross-track error is a bit varying, the error itself is kept low, meaning the controller is robust.

In Section 3.1.2, Section 3.1.3 and Section 3.1.4, the altitude error can be seen. Even with the severe disturbance added in the last test (seen in Figure 3.8), the altitude error stays within the limitations set in Section 2.5.3.7. This shows that averaging height (described in Section 2.5.3.8) provides the PID with a useful input altitude. Even though the PID controller is not optimal when controlling an aircraft susceptible to as much disturbance as can occur during flight (when it is as aggressively tuned), an aggressive controller is needed to counteract any changes forcing the UAV to deviate from its path.

This makes the PID controller acceptable when considering the lack of a mathematical model for the system, limiting the choices of guidance controllers.

3.1.6 HIL Simulation - Evasive maneuver

To force evasive maneuvers, the lateral controller was tuned to react slower and have an offset, as opposed to the previous tests. Looking at Figure 3.10 the 2D path of the evasive maneuver is seen.

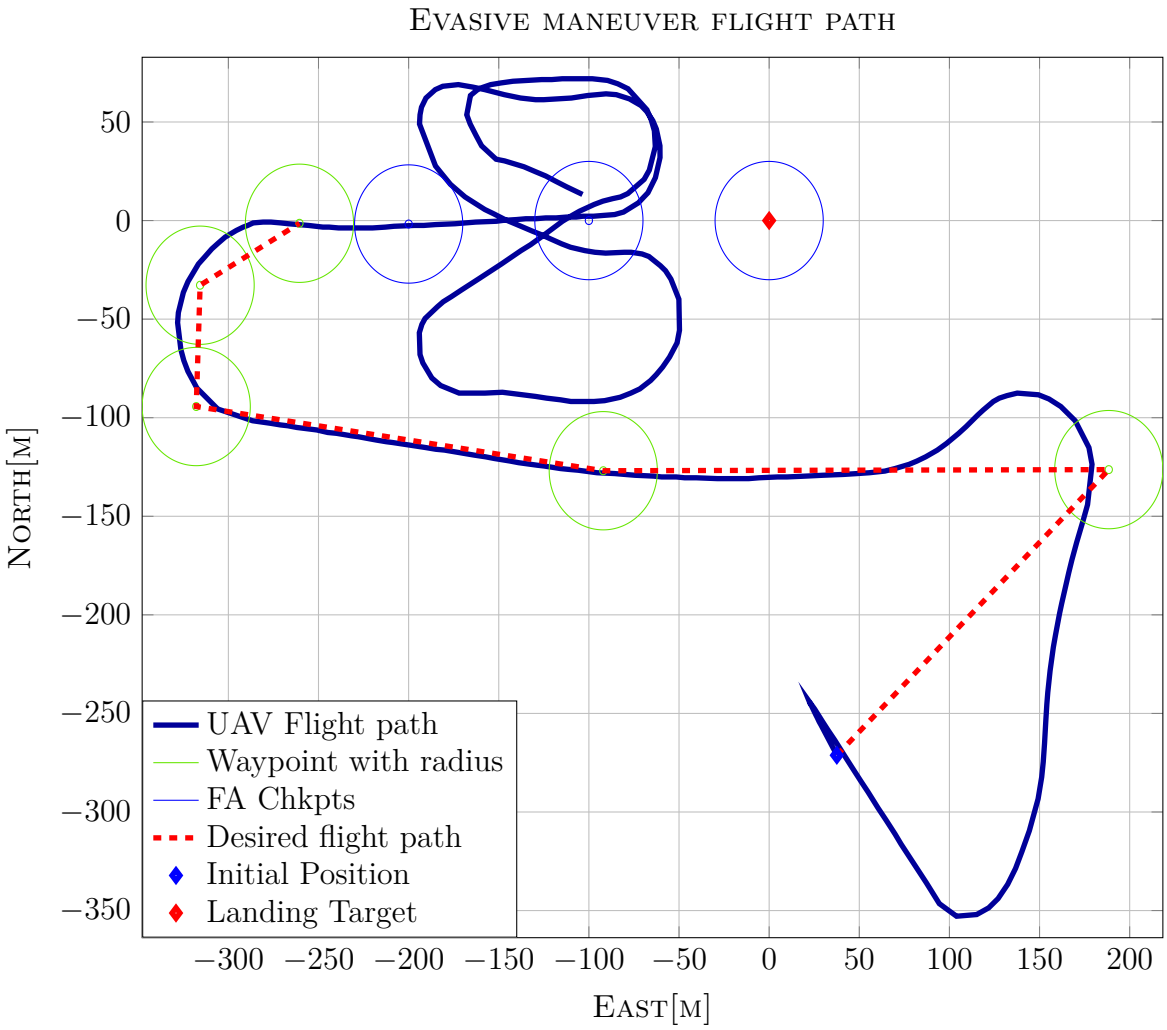


Figure 3.10: Plot of the horizontal flight - forced evasive, with wind

This plot shows that the UAV performs an evasive maneuver after passing checkpoint 2, as this is the point where the criteria for evasive maneuver starts to be checked. As the controller is still poorly tuned, the UAV keeps performing evasive maneuvers when it passes checkpoint 2 next time. Looking at Figure 3.11, it's clear that the evasive maneuver algorithm reacts to the cross-track error, as expected. An improvement would be that the evasive maneuver uses the same CPs for the evasive lap every time, e.g. the northern lap seen in Figure 3.10. This is to ensure that the evasive maneuver is performed in an area where this is allowed.

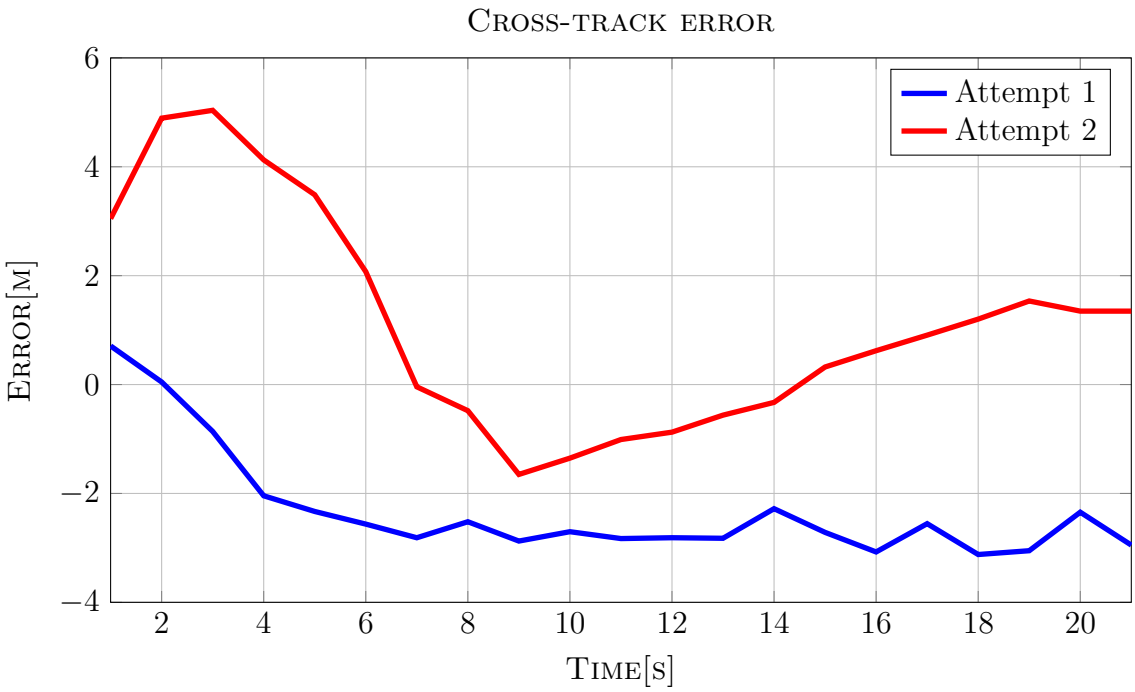


Figure 3.11: Plot of the cross-track error - forced evasive, with wind

3.1.7 HIL Simulation - moving target, no wind

An almost identical configuration to the one described in Section 3.1.2 was taken for the HIL simulation described in this section. The only change made was to the dynamic properties of the target object. The path flown by the aircraft is shown in Figure 3.12. Initially the path is determined by WPs, which serves the purpose of guiding the UAV through takeoff and into level flight. At the southern-most left turn, the custom guidance controllers take over from the regular controllers. The blue track marks the path taken by the target. Travelling at a constant speed of 4 m per second, the straight-lined path of the target results in a piece-wise constant heading.

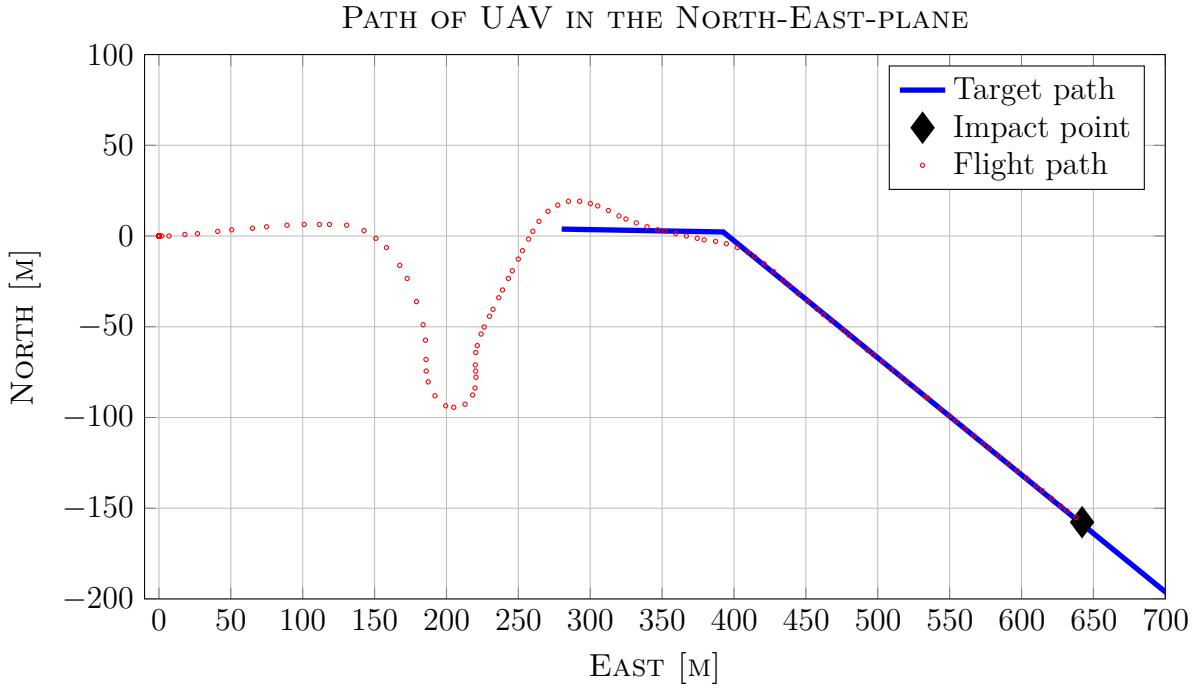


Figure 3.12: Flight path

Showing the final 20 seconds of the flight before impacting the target, Figure 3.13 and Figure 3.14 display the magnitude of the altitude error and cross-track error, respectively. Altitude errors at impact are centered at 0 m and are within ±0.2 m. As for the cross-track error, this is between 0.2-0.7 m at impact. As the figure shows, the errors are centered at a certain distance which remains relatively constant for each flight. At each update the target has moved a certain distance, resulting in an incremented altitude error as seen by the PID controller since the desired altitude is a function of the baseline distance.

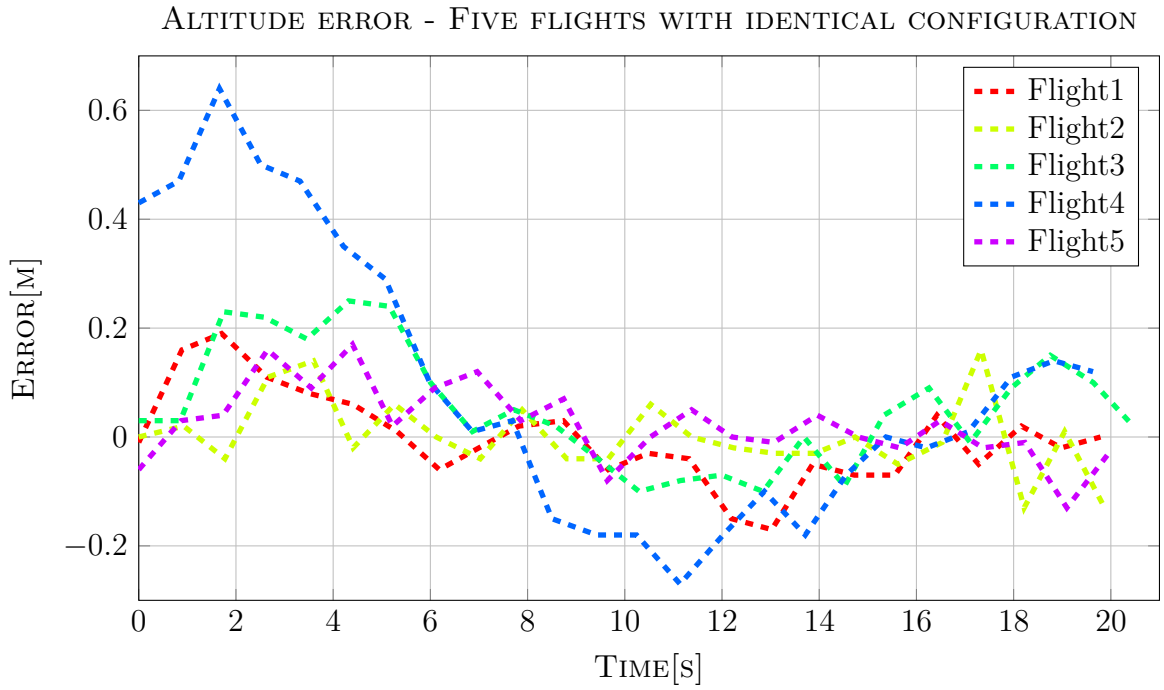


Figure 3.13: Altitude error

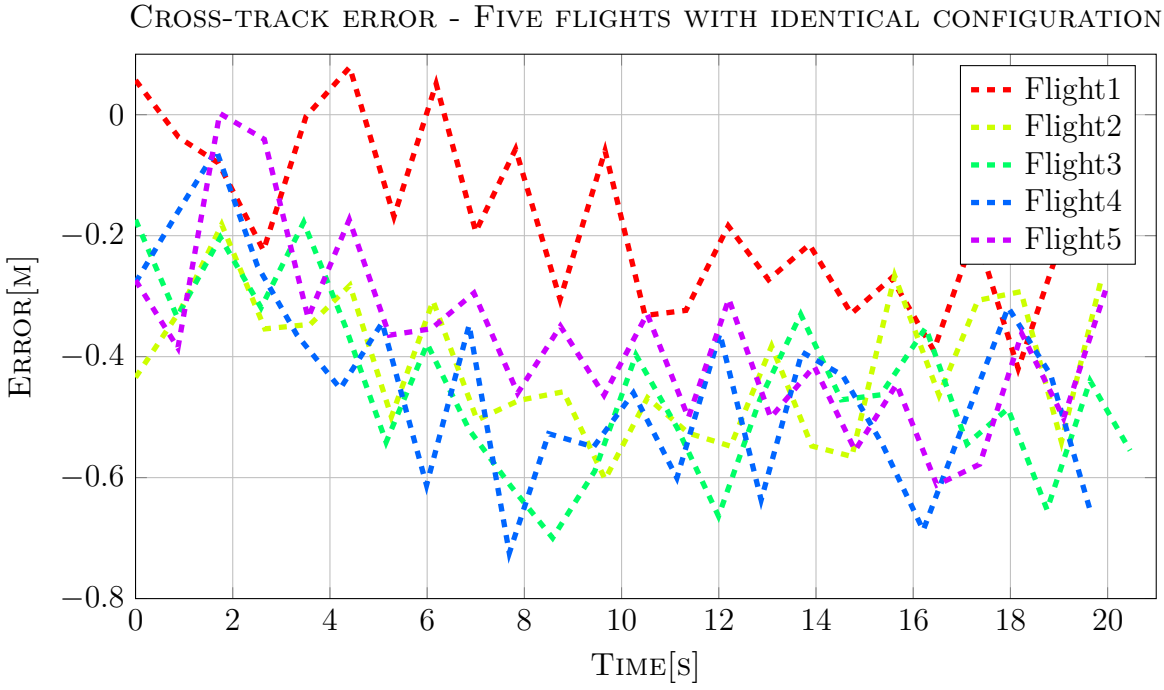


Figure 3.14: Cross-track error

3.1.8 HIL Simulation - moving target, wind

With wind conditions identical to the ones used for static-target HIL simulation with wind, 5 knot constant wind and 5 knot gusts, the only change made relative to the previous section was reducing the gains of the pitch servo controller. This was made in order to resist oscillations induced on the airframe due to wind gusts combined with a large P-gain. The old k_p value was 0.4, which was reduced to 0.3.

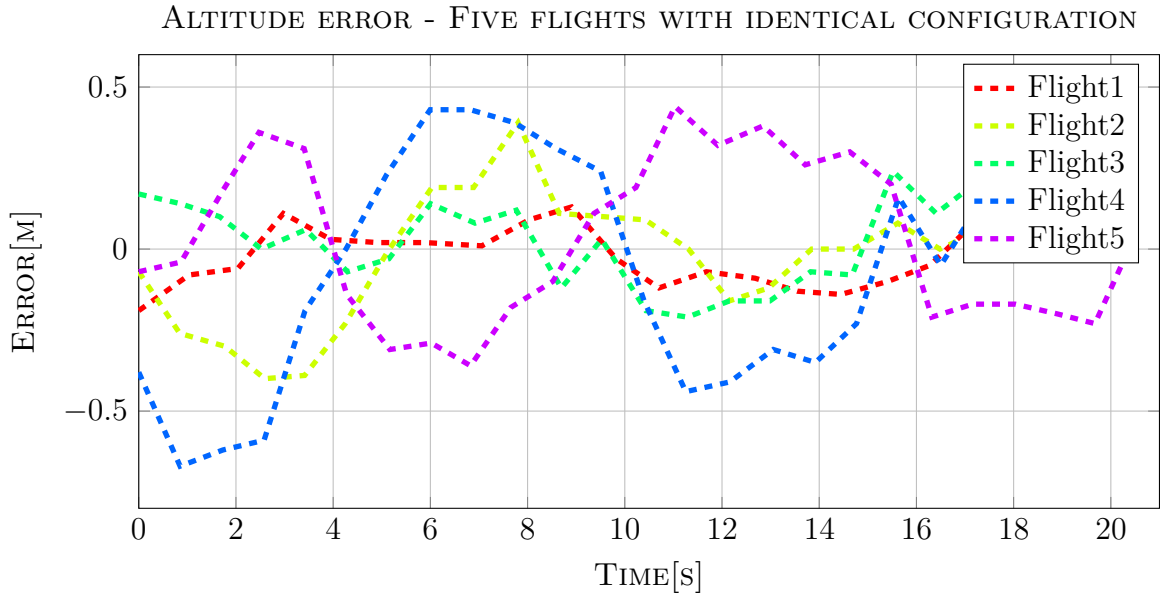


Figure 3.15: Altitude error

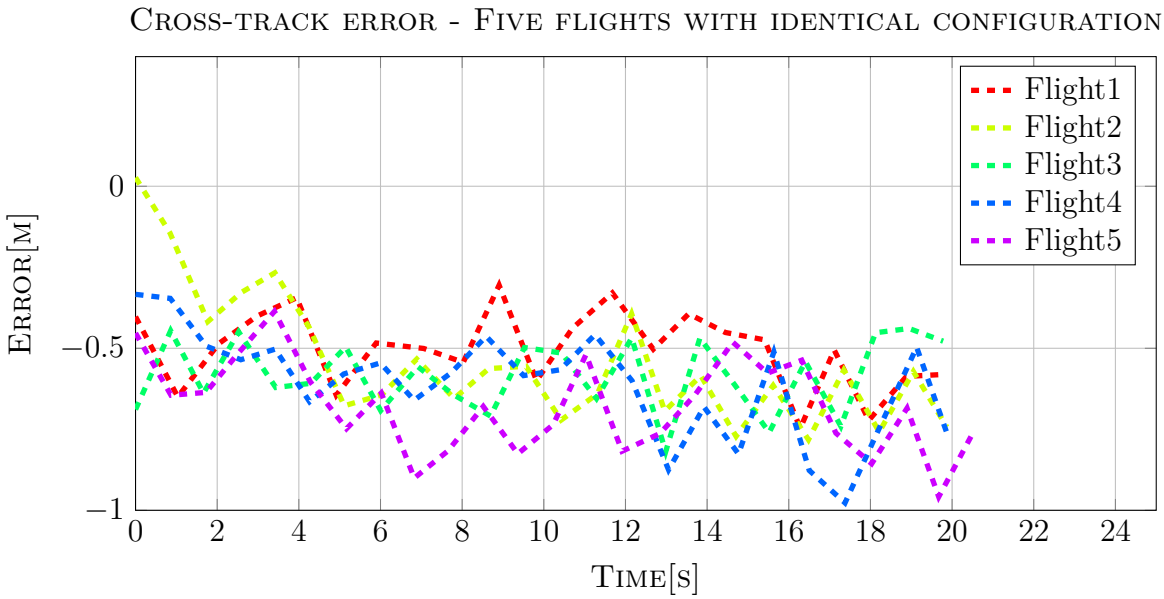


Figure 3.16: Cross-track error

The altitude error and cross-track error of the last 20 seconds of the flight are shown in Figure 3.15 and Figure 3.16. By comparison with the no-wind simulation, the altitude

error displayed a more erratic behaviour. With respect to cross-track error, no significant change was witnessed.

Simulation with wind present, yielded the path shown in Figure 3.17. It only slightly deviated from the no-wind-simulation by requiring a while longer to reach its intended track. This is caused by the south-easterly wind direction.

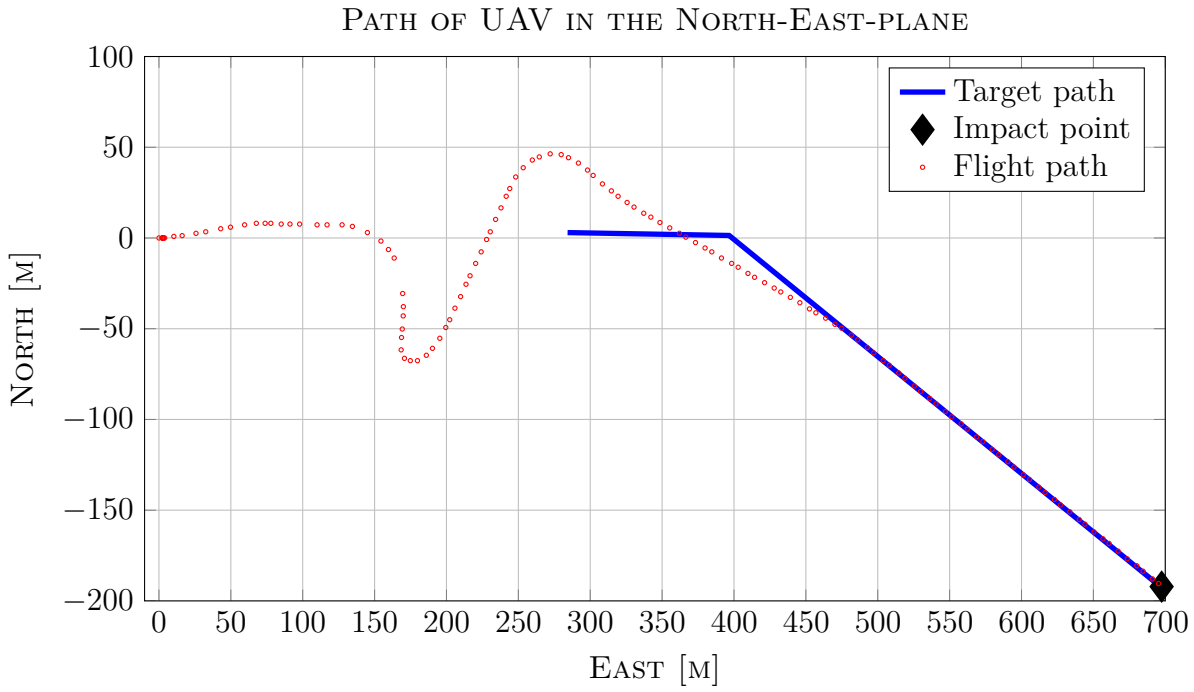


Figure 3.17: Flight path

3.1.9 Discussion - Moving target

This section will discuss the results from HIL-simulations involving a moving target.

Starting with simulations with no wind disturbance, described in Section 3.1.7, both cross-track error and altitude error are well within their maximum values. This suggests that retrieving a UAV under restricted conditions is feasible with the current controllers and hardware. In this case the word restricted requires the vessel used for retrieval to maintain a constant heading for an extended period of time. Section 3.1.8 shows that the system, in addition to a moving target, could handle disturbances in the form of constant wind and wind gusts. Moderate wind was added to prove the concept and the lack of significant change in altitude- and cross-track error shows the system's ability to reject minor wind-disturbances.

3.2 Preliminary GPS tests

During the preliminary project, spanning the fall semester of 2013, tests to investigate the feasibility of RTK-GPS on a UAV platform were performed. Located near Hønefoss, Norway, Eggemoen airstrip provided ideal facilities and conditions for flying a UAV. There are no surrounding hills or mountains and the airstrip itself is 2100 m long. Running alongside the airstrip, on both sides, is an additional 150 m of grass fields before tall trees appear. Estimating the height of the trees at 15 m this yields an unobstructed view of the sky at greater elevation angles than approximately 6° (see Equation 3.2.1).

$$\alpha = \arctan\left(\frac{15m}{150m}\right) = 5.71° \tag{3.2.1}$$

The result of this is a high number of visible satellites seen in Figure 3.18. Having a large number of satellites visible allows for lower DOP values which improves the absolute positioning used in the regular auto mode of flight. An increased number of satellites also improves the reliability of the ambiguity resolution process, but does not increase the accuracy of the baseline vector, as concluded by [22].

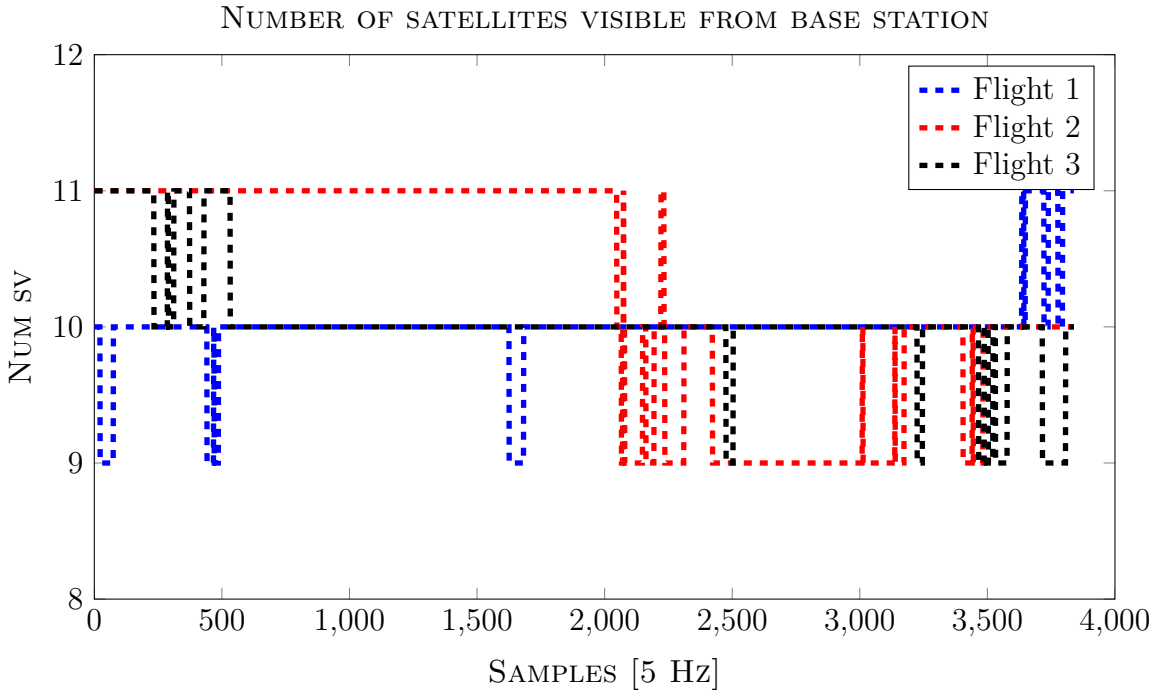


Figure 3.18: Shows visible satellites from base station for three test flights performed on the same day.

Depending on elevation mask, satellites which are regarded as valid (used in calculation of solution) are a subset of the ones displayed in Figure 3.18. It is assumed that the visible satellites seen during one day of flight is representable for an average day of flight. The configuration of the payload selected for these preliminary tests is described in Section 2.2.1, specifically in Figure B.3. Antennas used in the setup are listed below.

- GPS antenna UAV: Embedded GPS antenna with ground plane (1 m coaxial cable). The first two days of testing, this was fastened to the top of the interior front-section

of the UAV. Final day of testing saw the addition of an aluminium ground plane for the antenna to be mounted on (see Section 2.2.3).

- GPS antenna base station: An identical antenna was mounted on a ground plane and placed at a height of 2.5 m above ground with a clear view of the sky. This antenna had a longer (5 m) coaxial cable attached.

Due to the small size and aerodynamic lines of the UAV, the nose was considered the only feasible mounting location inside the fuselage. Mounting the antenna on the outside of the fuselage meant exposing it to the elements and also meant that adding an additional ground plane would be difficult to manage. As the L1 frequency does not experience significant attenuation when travelling through the EPO of the aircraft fuselage, as demonstrated in Figure 3.19, inside-mounting was a natural choice.

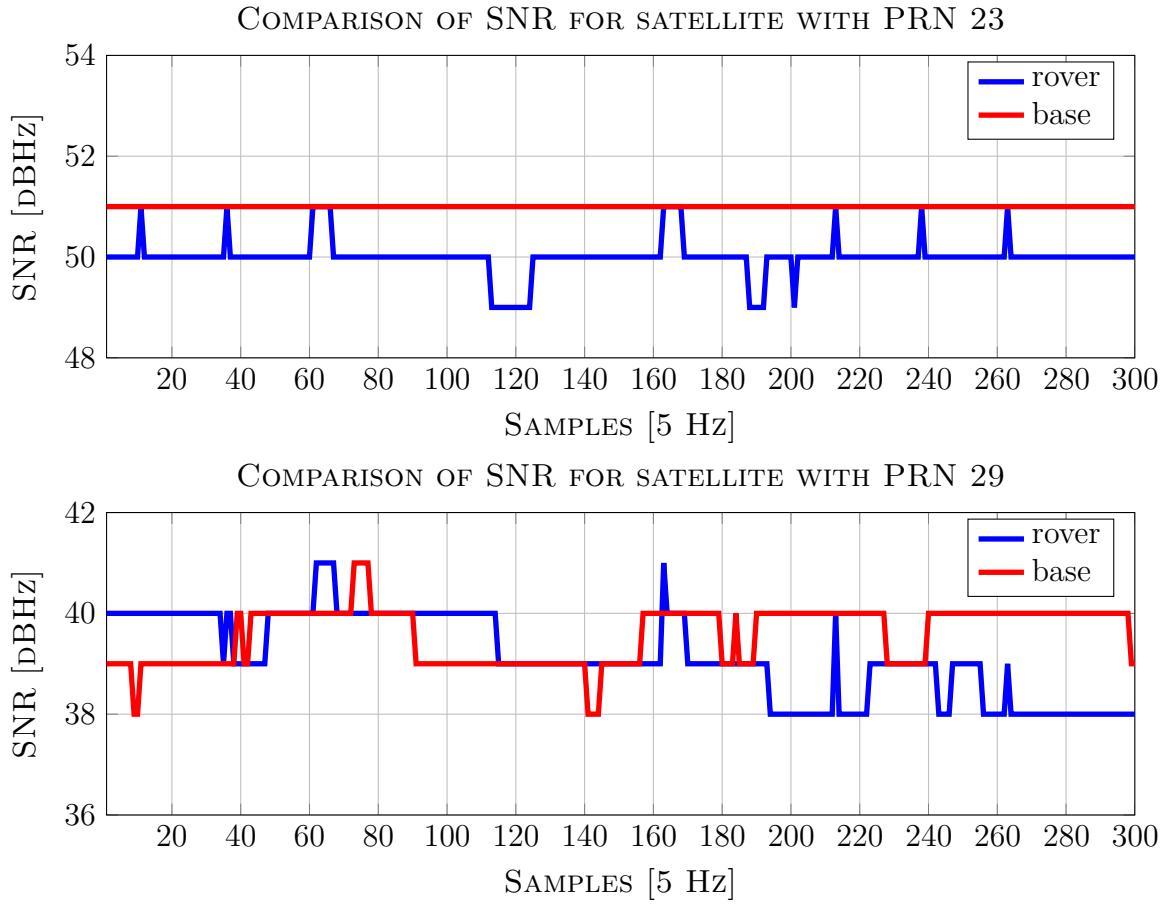


Figure 3.19: Displays measured SNR at both rover and base station receiver. Antennas are mounted inside UAV (rover) and on top of ground station (circa 2.5 m above ground).

Appendix A shows the specific RTKLIB settings tested and the resulting percentage of different solution qualities. Verification of the accuracy of the RTK-GPS solution, with fixed- and float ambiguities was tested and the results shown in Figure 3.20.

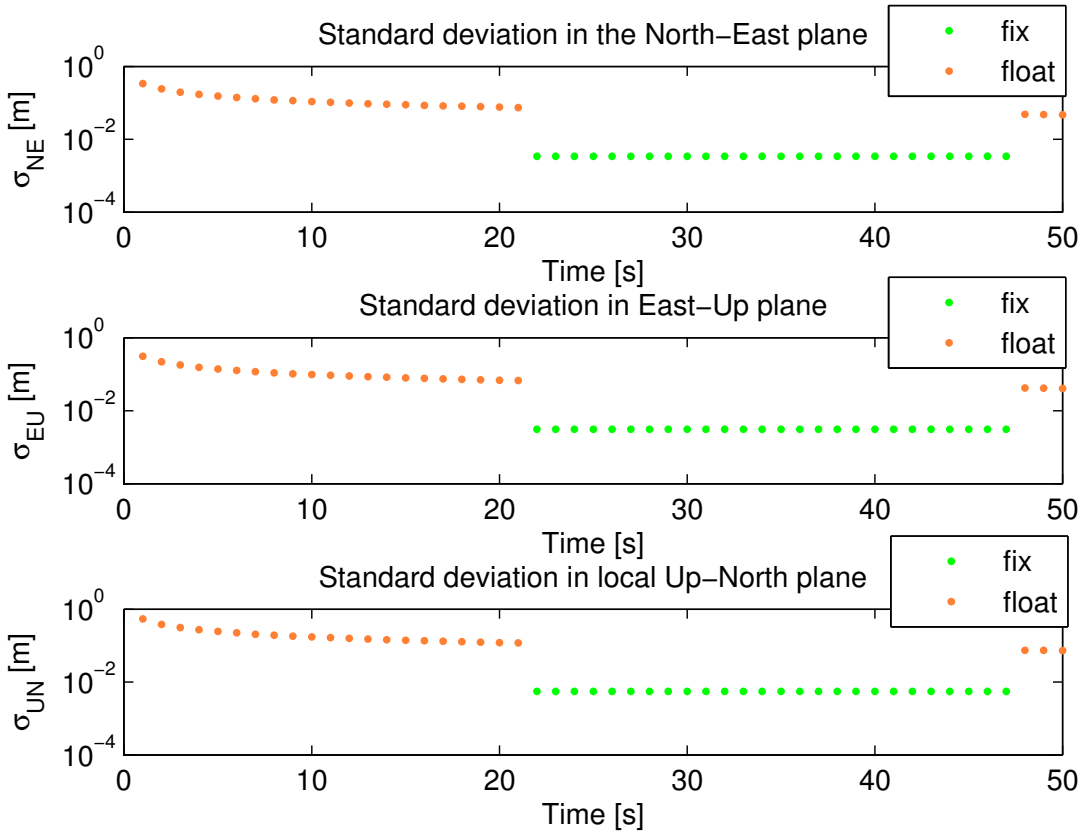


Figure 3.20: Shows difference in standard deviation from experimental data sampled, and processed, using the techniques given in MLAMBDA method. The fixed solutions are consistent and accurate. Note the logarithmic scale on the Y-axis.

Figure 3.20 shows the standard deviation in the planes; North-East, East-Up and Up-North. As shown in [37], the magnitude of the standard deviations were as expected. The estimated standard deviations of the solution were calculated assuming a priori error model and error parameters by the positioning options [34].

3.2.1 Discussion

A main part of this thesis was dependent upon achieving a high-precision RTK-GPS solution. Several hardware configurations were tested throughout the duration of the project. These changes were mostly related to antennas, with the intention of increasing signal strength, reduce multipathing and achieve the highest possible number of satellites common to both base station and UAV. The most notable addition was to the GPS antenna mounted on the UAV. An extra, significantly larger, ground plane was added to supplement the small 5 cm ground plane inherent to the antenna. Illustrating the difference is Table A.3 and Table A.4, where the former shows statistics from data sampled before the addition of the ground plane. A large increase in samples displaying the fixed RTK-GPS solution, rather than float, speaks to its significance. Comparing figures given in Appendix A.3 with figures in Appendix A.1 and Appendix A.2 shows the slightly increased

SNR value for the post-ground plane sampled data.

3.3 Roll compensation

This section covers sampled data of roll-attitude of the aircraft and how it relates to a change in satellites visible from the UAV. Section 3.3.1 has no roll-gimbal installed for roll-compensation of the GPS antenna with extended ground plane, whilst Section 3.3.2 shows the roll-compensated configuration.

3.3.1 Without roll compensation

As the aircraft pitches, banks or performs a combination of motion about these axes, satellites might be lost from view of the receiver mounted on the UAV. This impacts the number of satellites common to both the base station receiver and the UAV and thus impacts the stability of the RTK solution. Factors determining if satellites become ”invisible” are:

- Which way the plane banks or pitches relative to the current constellation of satellites
- The magnitude of the angle

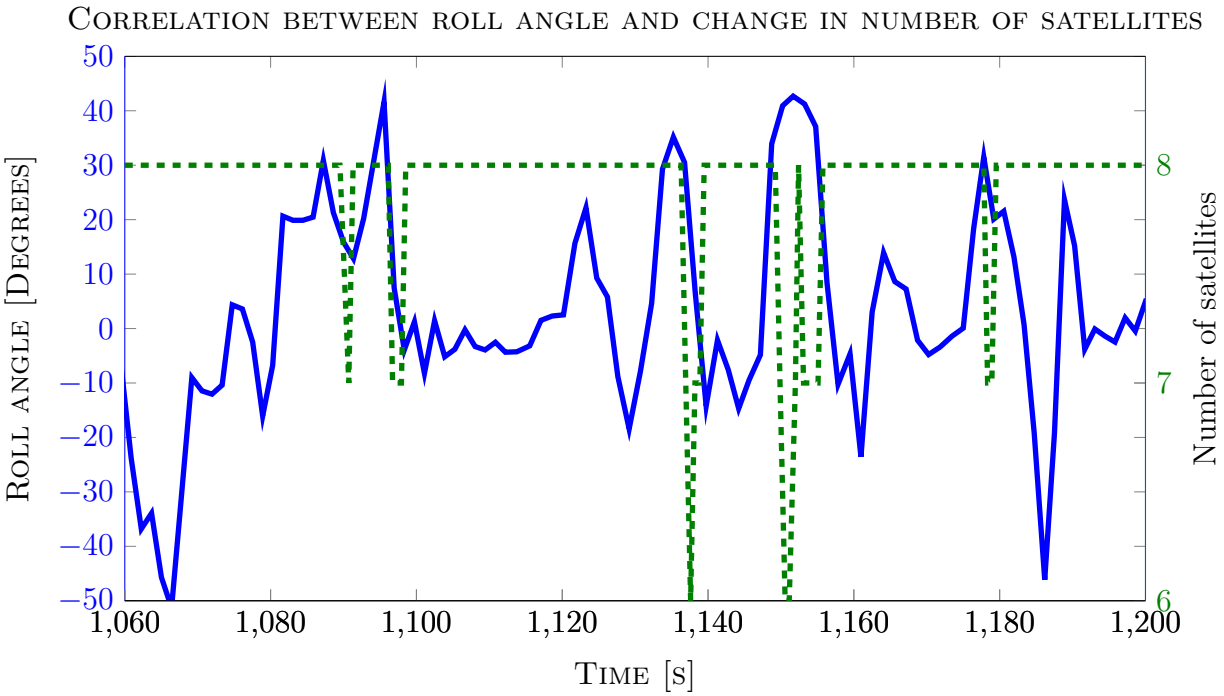


Figure 3.21: Shows correlation between visible satellites and change in roll angle.

Evident from Figure 3.21 is the loss of one or two satellites due to a significant change in roll angle. Figure 3.22 and Figure 3.23 captures the aircrafts movement about both roll

and yaw axes through a 180° turn. The large roll angle causes the ground plane of the antenna to block out two satellites, initially.

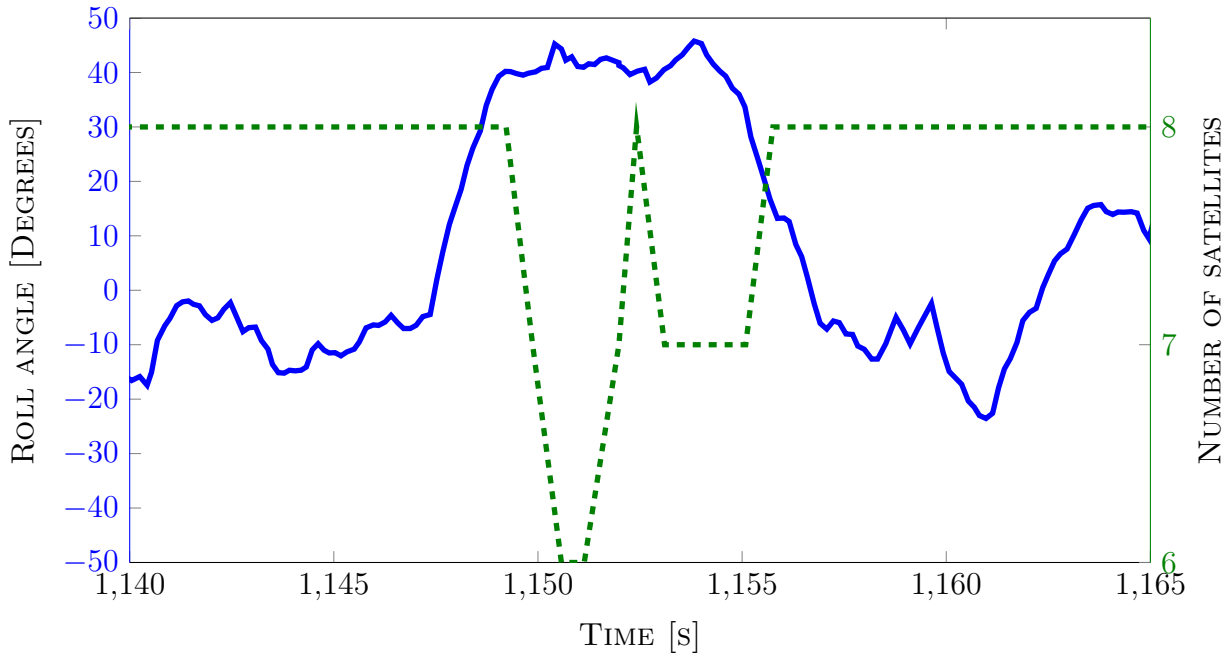


Figure 3.22: Shows correlation between visible satellites and change in roll angle. Zoomed version of Figure 3.21

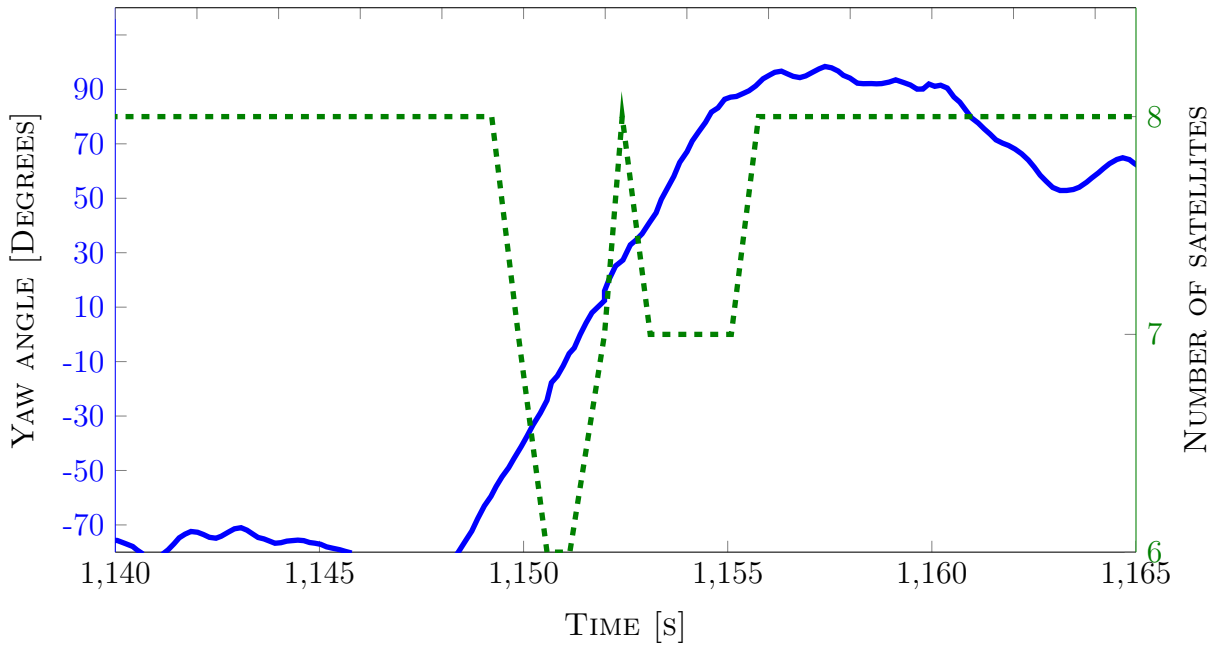


Figure 3.23: Shows correlation between visible satellites and change in yaw angle.

As the aircraft yaw reaches 90°, or halfway through the turn, the satellite count goes back to eight satellites. Passed the 90° mark, yet another portion of the sky is off-limits to the

UAV GPS receiver, resulting in seven visible satellites. Finally, as the aircraft levels out, its original level flight attitude is restored and all satellites not ruled out by the constraints given in Section 2.5.5, and test specific constraints given in the results section, are yet again visible. Other parameters capable of altering the number of satellites visible from the UAV, other than its own attitude, is;

- Local topography
- The natural movement of the orbiting satellites causing them to drop below the set elevation threshold

3.3.2 With roll compensation

In Figure 3.24 the roll angles (compensated and actual) combined with the visible satellites are shown. The maximum angle of the roll gimbal was limited to ±35°, because of the spatial limitations in the nose of the UAV. The figure shows that the number of visible satellites is reduced when the roll angle is greater than compensated roll angle. This result is logical as the ground plane the antenna is mounted on will have a roll angle equal to the difference between the roll angle and the compensated roll angle. This might lead to loss of visible satellites (depending on the satellite constellation in relation to the UAV).

Naturally, the other environmental limitations described in Section 3.3.1 are still in effect. However, the issues regarding roll angle and loss of satellites (shown in Figure 3.21) can now be negated.

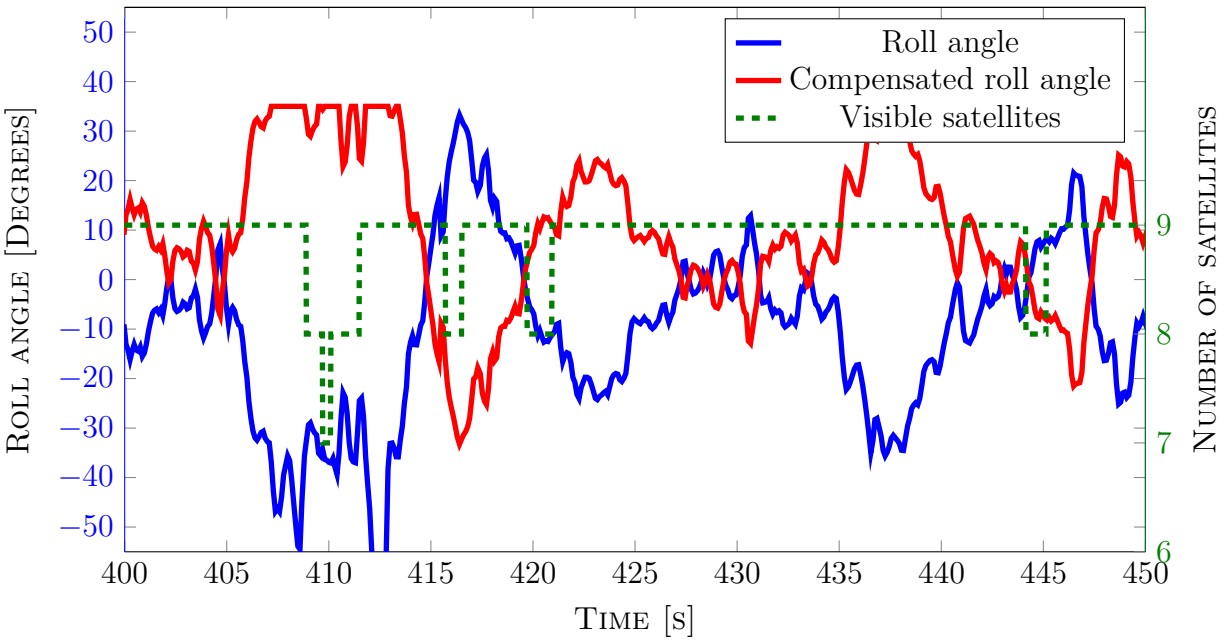


Figure 3.24: Shows the effectiveness of the roll compensation setup

3.3.3 Discussion

It can be seen from this section that the GPS antenna itself is compensated in roll, and keeps the antenna level. Figure 3.24 clearly shows that if the roll angle goes outside the bounds of the compensation setup, satellites are still lost. However, within the area of compensation the visible satellites stay constant and the same set of satellites can be used for the integer ambiguity resolution.

Although a direct correlation between roll angle compensation and fixed RTK-GPS solution has not been found, it helps indirectly by allowing the satellites used to compute the solution to stay in sight. In the cases where few valid satellites are available, this will be the deciding factor between a RTK-GPS fix solution and not, as the importance of each valid satellite grows.

3.4 Flight tests

In this section a selection of the experimental tests are presented. The total number of tests exceeds 50 over a period of 8 days. The tests presented are both tests that show the feasibility of landing in a net, but also tests that presents limitations in the system.

3.4.1 Location of tests

A re-location of base of operations took place before the system was ready for final flight tests. A small airstrip of 500 m length at Agdenes, Norway, became the new location for testing. Topography-wise, this facility was somewhat different than Eggemoen. Closer to the coast higher winds were encountered and a nearby hill induced turbulence to the increased level of wind. Unfortunately, no equipment that reliably measured the winds at the LT existed, hence data showing the actual wind patterns will not be provided in this thesis. Agdenes represented a more challenging environment, both with respect to the increased forces acting on the airframe and hills making certain sections of the virtual dome spanned by all possible elevation- and azimuth angles off-limits. Figure 3.25 shows the visible satellites as seen from the base station antenna sampled during the course of three separate flight tests.

The data was sampled from the base station antenna, which was mounted on the rooftop of the small aircraft hangar attached to the airstrip, during tests commenced at 14:47 (flight 1), 16:23 (flight 2) and 17:44 (flight 3). It's clear from Figure 3.25 that the number of satellites, seen from the base station, changes throughout the day, as the three flights were conducted with an interval of about 1.5 hours. The results were representative of the GPS availability experienced throughout the testing at Agdenes.

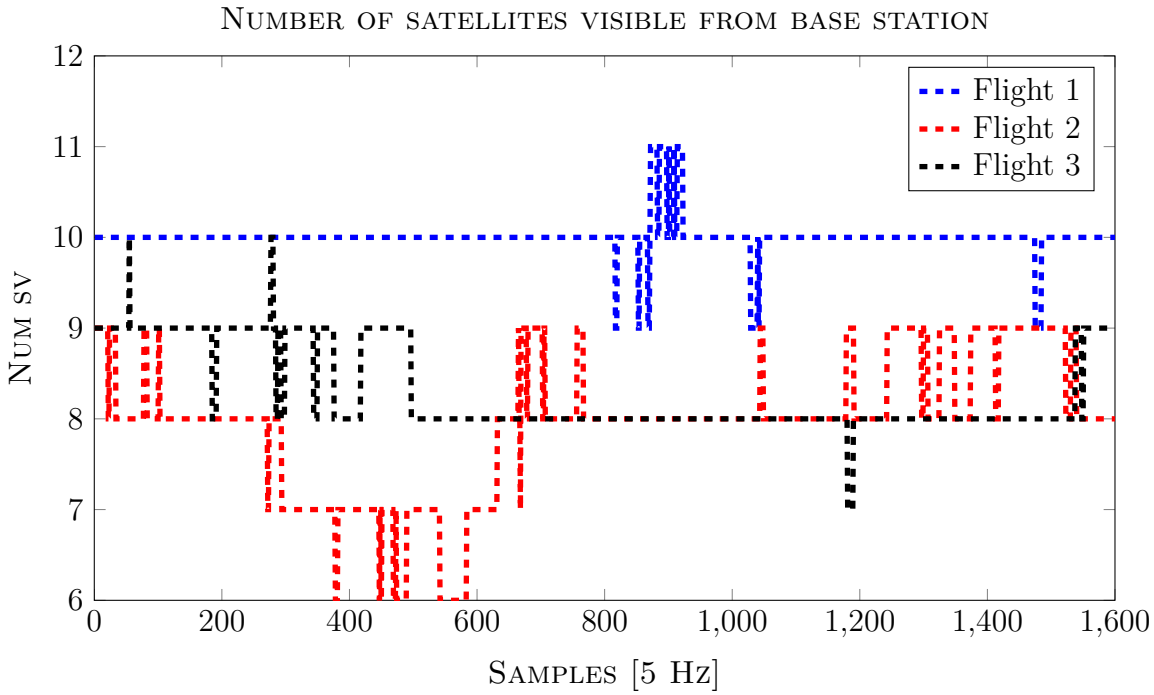


Figure 3.25: Shows visible satellites from base station for three test flights performed on the same day.

3.4.2 Hardware setup

After the move to Agdenes, the hardware configuration was changed, especially with respect to the wireless communication between the base station and UAV. A detailed description of the new setup is given in Section 2.2.1 and Figure B.4. The types of antennas used are listed below.

- GPS antenna UAV: No changes in antenna type. As testing revealed difficulty in achieving a fixed RTK-GPS solution when turbulent air induced rapid and extensive movement about the roll-axis the GPS antenna was eventually moved to a roll-stabilized ground plane (see Section 2.2.8.3)
- GPS antenna base station: A new antenna was introduced along with a more permanent mount. See Section 2.2.3 for further details.
- Base station communication antennas: Two roof-mounted dipole antennas were mounted on the aircraft hangar to increase the range of the communication system [44]. The previous arrangement only utilized the NanoStation with built-in-antennas.

3.4.3 Key results from flight tests

This section outlines how the system performed when tested on the X8 Skywalker platform. A series of tests, spanning several separate days of testing were performed in order to tune control system parameters to reliably hit the center of the retrieval net. The

challenges encountered were dealt with as they arose, and the main issues are described below.

3.4.3.1 Altitude error due to boundary on pitch

For the duration of this particular test, disturbances affecting the airframe was virtually non-existent. Seen in Figure 3.26 is a constant negative altitude error. According to the below equation, used for altitude error calculation, this translates to the aircraft being a certain distance above the target altitude, $h^c(t)$.

$$e_{alt}(t) = h^c(t) - h(t)$$

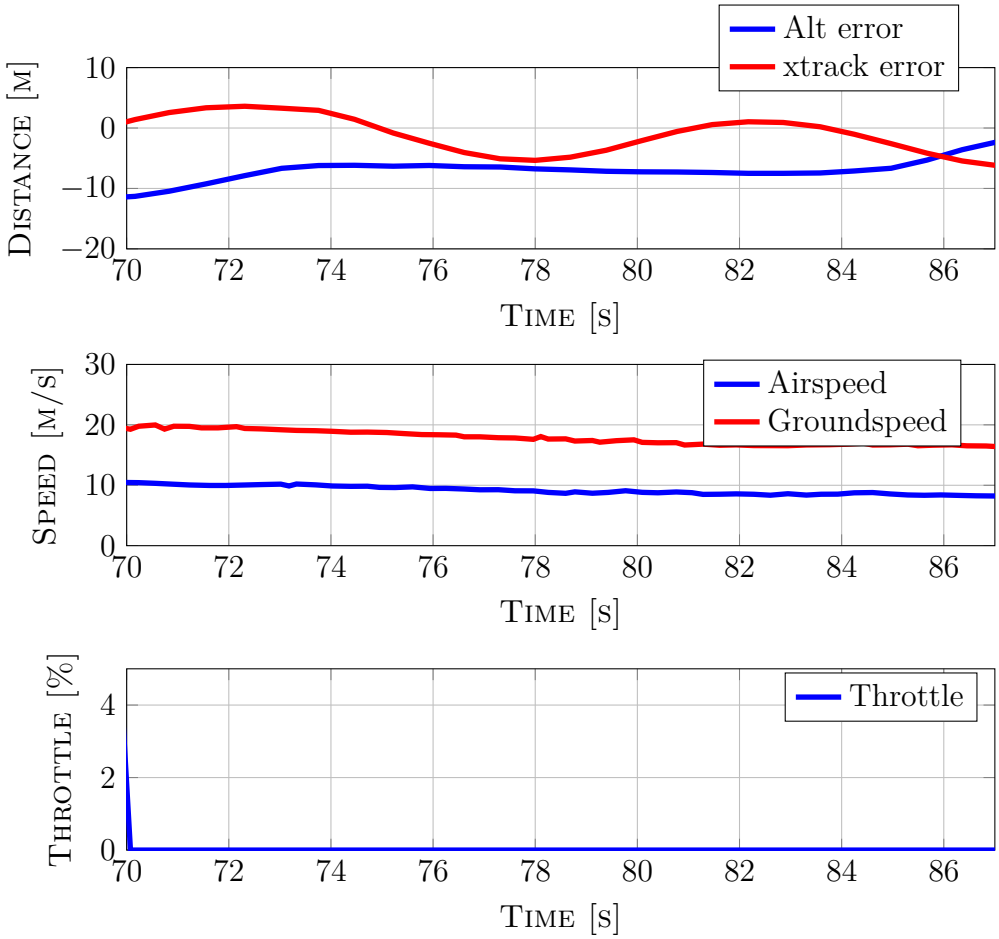


Figure 3.26: Altitude error, cross-track error and throttle output for FA.

Seen in Figure 3.27 is the effort made by the control system to reduce the altitude error mentioned above. The minimum commandable pitch angle is, however, constrained by the variable *min_limit_pitch*, which is a product of the roll angle and a fixed lower boundary. In this case the boundary was set to -15° and further constrained by the roll angle. Hitting the threshold for minimum pitch angle, the aircraft remains on a linearly descending slope (Figure 3.26 shows a constant altitude error during FA descent). With no headwind and a negative pitch angle, the aircraft maintains its speed well, even with a minimum throttle

of 0%. The throttle goes to zero as the speeds (both ground and air) are both above the setpoints.

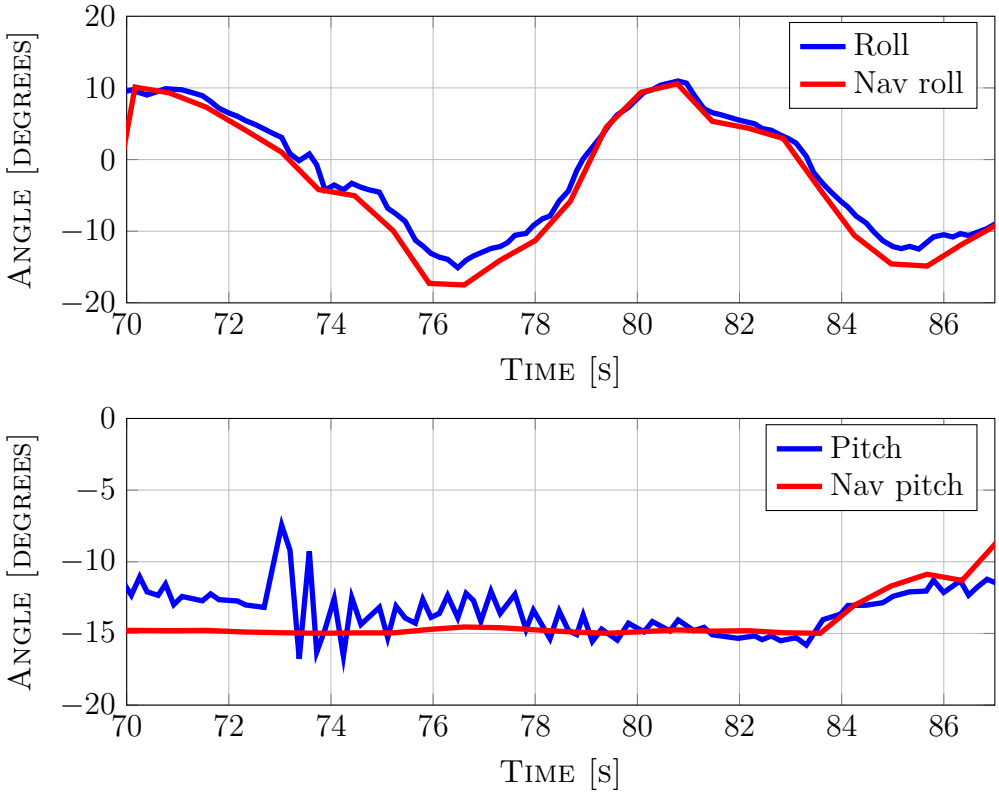


Figure 3.27: Commanded pitch/roll versus measured pitch/roll.

3.4.3.2 "Snaking" due to aggressive L_1 controller

The 'snaking' effect for the lateral controller during FA occurs when the controller is too aggressively tuned. Looking at Figure 3.28 the effect of an aggressive L_1 controller vs a less aggressive one is shown.

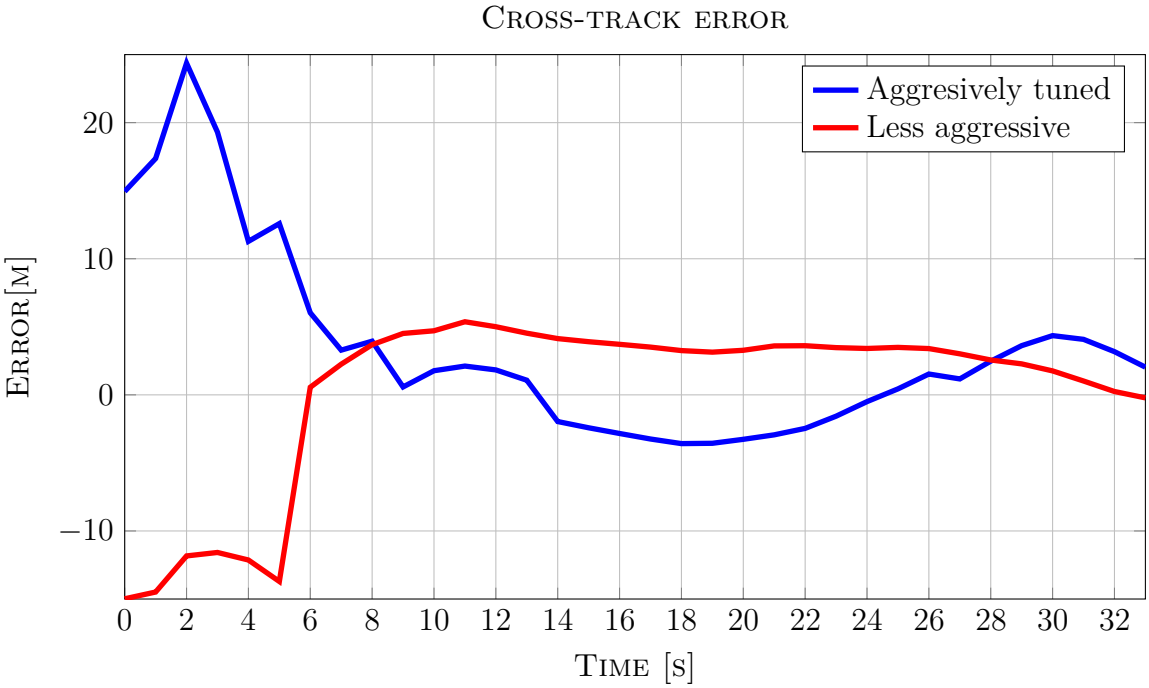


Figure 3.28: Aggressive vs more conservative L_1 controller

These cross-track errors are taken from FA during similar wind conditions. Clearly, the less aggressively tuned controller forces the cross-track error to zero in a more controlled manner than the aggressively tuned controller does. Both versions has its usage; during very windy/turbulent conditions the aggressively tuned version might reduce the cross-track error quicker/better as the shifty wind conditions might lead to the conservative version not having enough time to sufficiently reduce the cross-track error. The factors involved in tuning are the L_1 and K_{L_1} (described in Section 1.3.2) variables. A gain-schedule approach might improve the performance of this controller, as the controller gains can be tuned more aggressively depending on the distance to target, wind conditions or CPs passed.

3.4.3.3 Precision and repeatability of system

This test was setup to prove both repeatability and precision of the system. Key results from flights resulting in, for the most part, successful net retrievals are shown. For visual confirmation of impact, two thin strips of crepe paper was mounted on two poles placed about 12 m apart. These strips of paper were meant to indicate the lower and upper boundary of the net used for retrieval. As the net measures 5 m wide by 3 m high, the upper strip had an approximate altitude of 5 m above ground and the lower strip 2 m. Figure 3.29 shows the setup which had the purpose of providing visual confirmation of a net retrieval without the consequences of impacting the net.

Figure 3.29: Crepe paper "net" for visual confirmation of impact point.

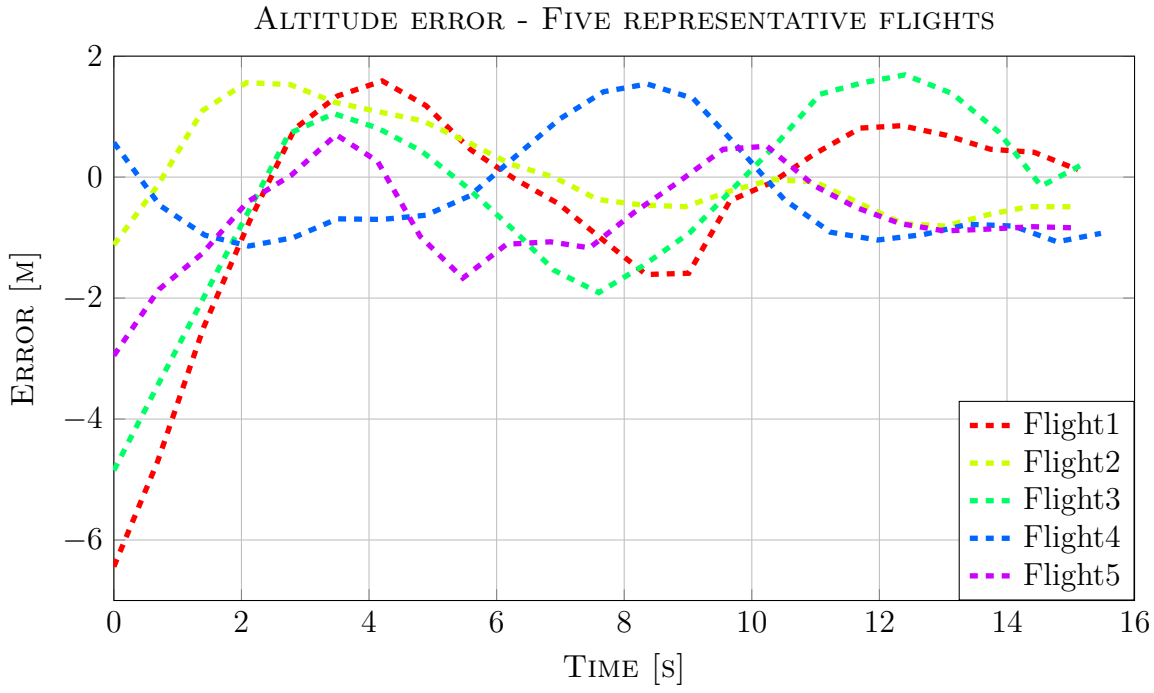


Figure 3.30: Altitude error during FA of flight tests.

All flight tests, from which the data seen in figures presented in this section, were performed without much environmental disturbances. A light, constant breeze blew from the west, making the ideal land heading approximately 270°. Figure 3.30 indicates how much the aircraft's altitude deviates from the desired altitude over time. A positive error translates to the aircraft having a lower altitude than desired. Cross-track error is given in Figure 3.31 and yields the shortest distance between the aircraft and the desired track in the horizontal plane (see Figure 1.2).

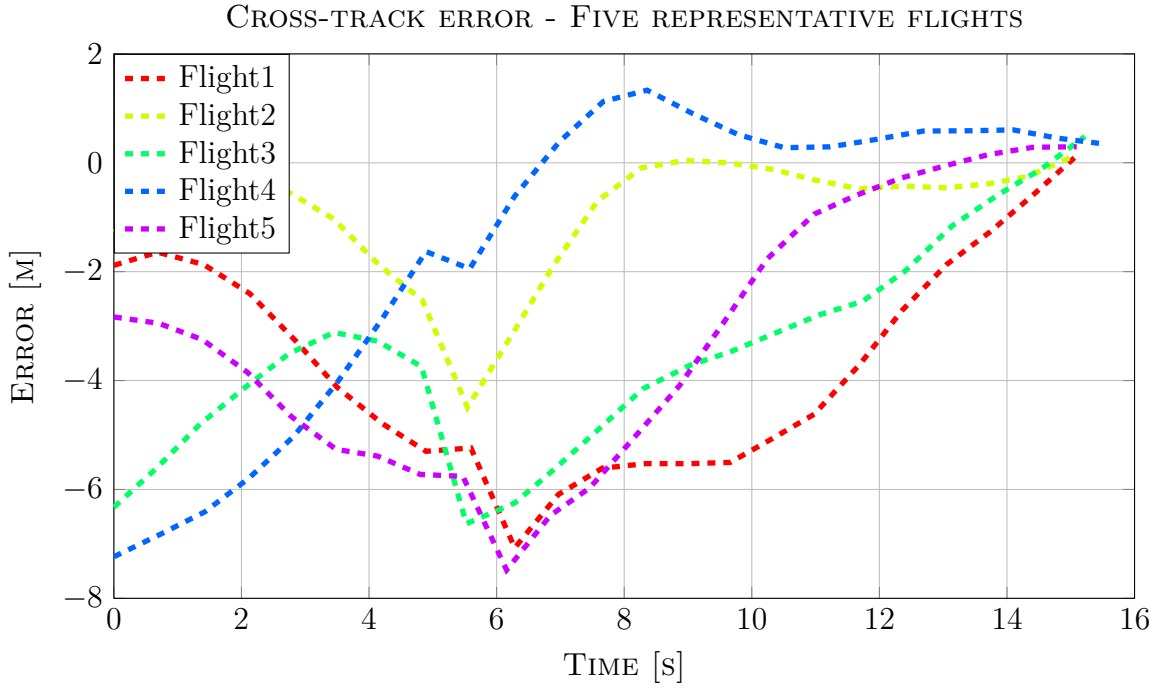


Figure 3.31: Cross-track error during FA of flight tests.

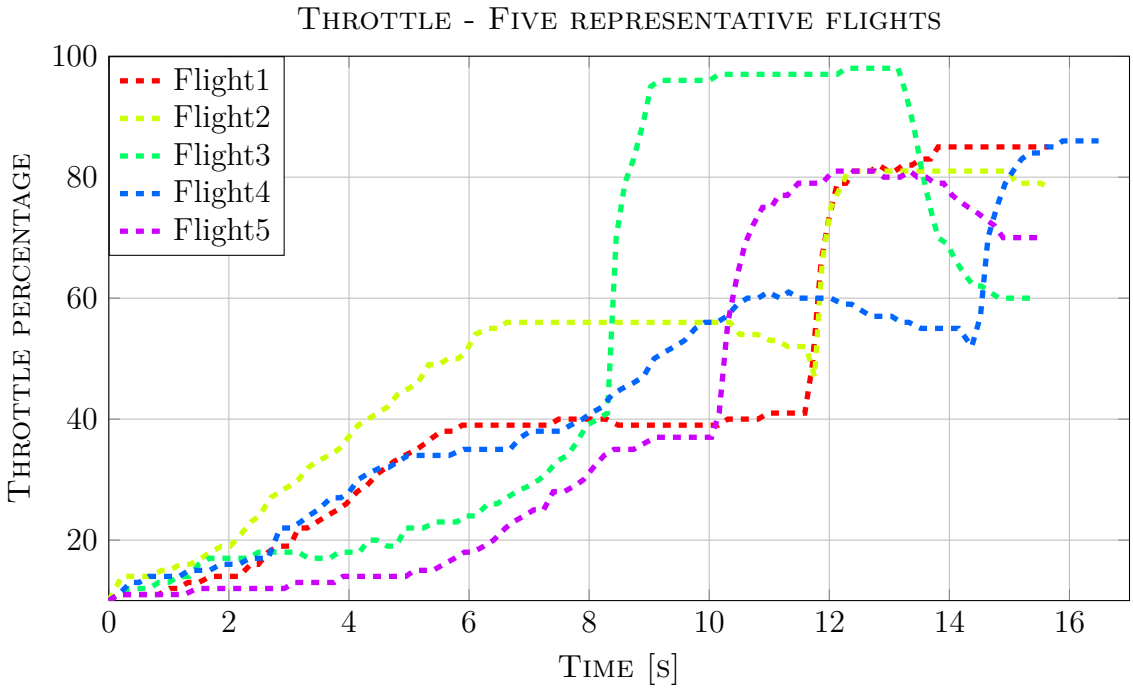


Figure 3.32: Throttle output during FA of flight tests.

Figure 3.32 and Figure 3.37 indicate the commanded throttle and the ground speed, respectively, of the UAV for the same flights as in the figures describing altitude error and cross-track error. The desired speed set for the aircraft to follow was 16 $\frac{m}{s}$ as recommended by the pilot. This was to minimize the forces acting on the UAV fuselage at impact, as

well as allowing for sufficient control authority during FA. A certain distance away from the target the commanded throttle was set to zero as to not cause damage to the net or engine after impact. To be able to safely impact the static net, groundspeed must be low and the engine must be stopped. Figure 3.36 shows the commanded throttle set to zero a certain distance from the impact point. Desired groundspeed was set to 16 $\frac{m}{s}$.

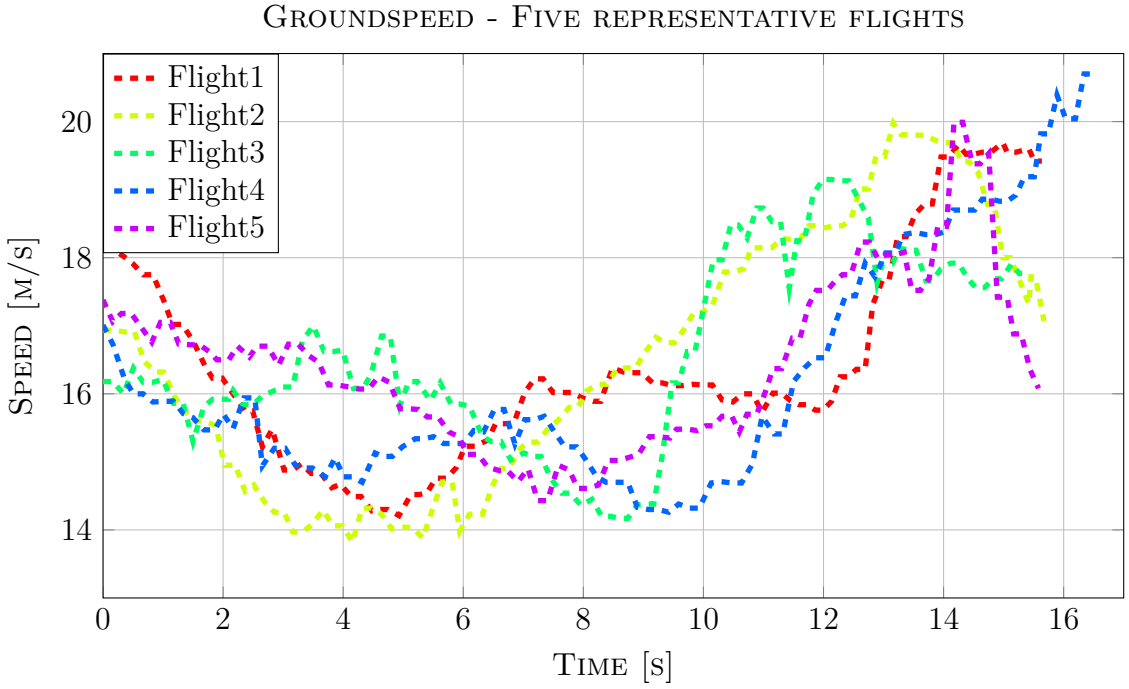


Figure 3.33: Ground speed during FA of flight tests.

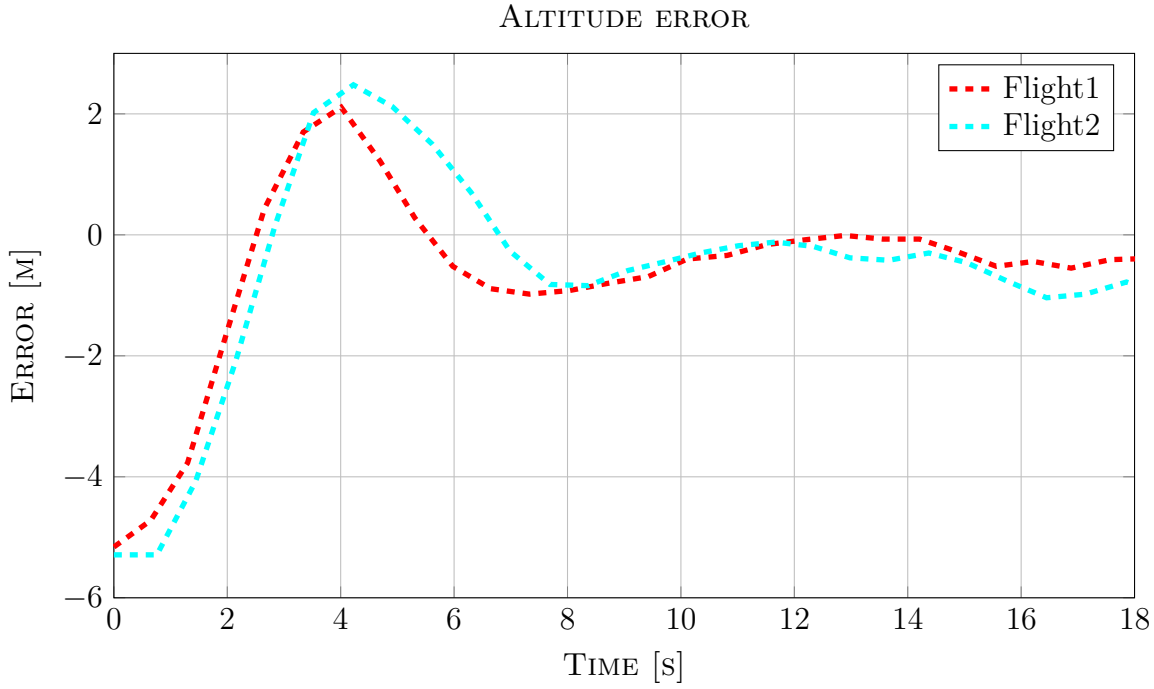


Figure 3.34: Altitude error during FA of flight tests.

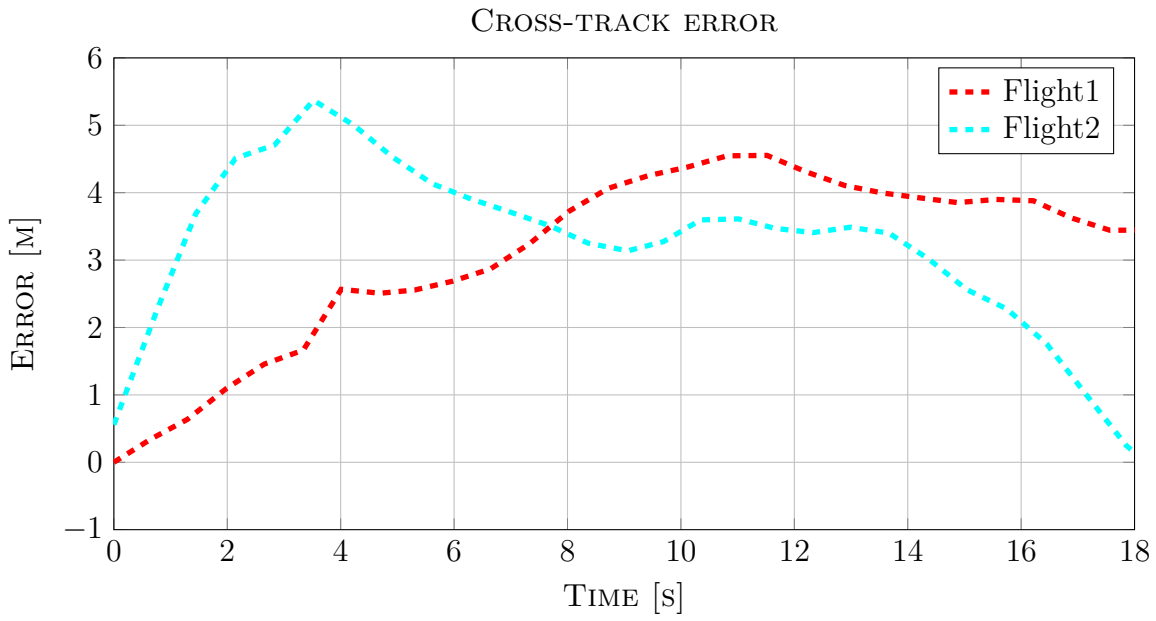


Figure 3.35: Cross-track error during fA of flight tests.

Figure 3.34, Figure 3.35, Figure 3.36 and Figure 3.37 shows the UAV performance during FA. These two flights were attempts at landing in an actual net, and flight 2 was the successful landing. Both flights performed engine cut-off as intended and the groundspeed was around the desired value. Seen from the plots, the cross-track error for flight 1 was too big, and normally an evasive maneuver would have been performed. As this was not implemented, the UAV continued its flight and missed the target.

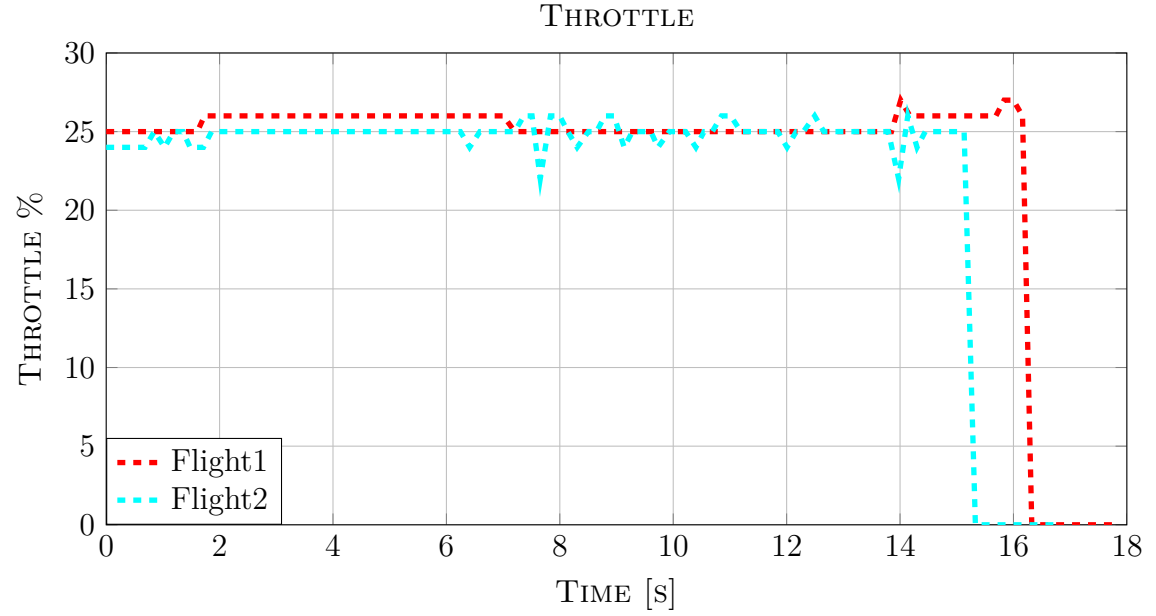


Figure 3.36: Throttle output during fA of flight tests.

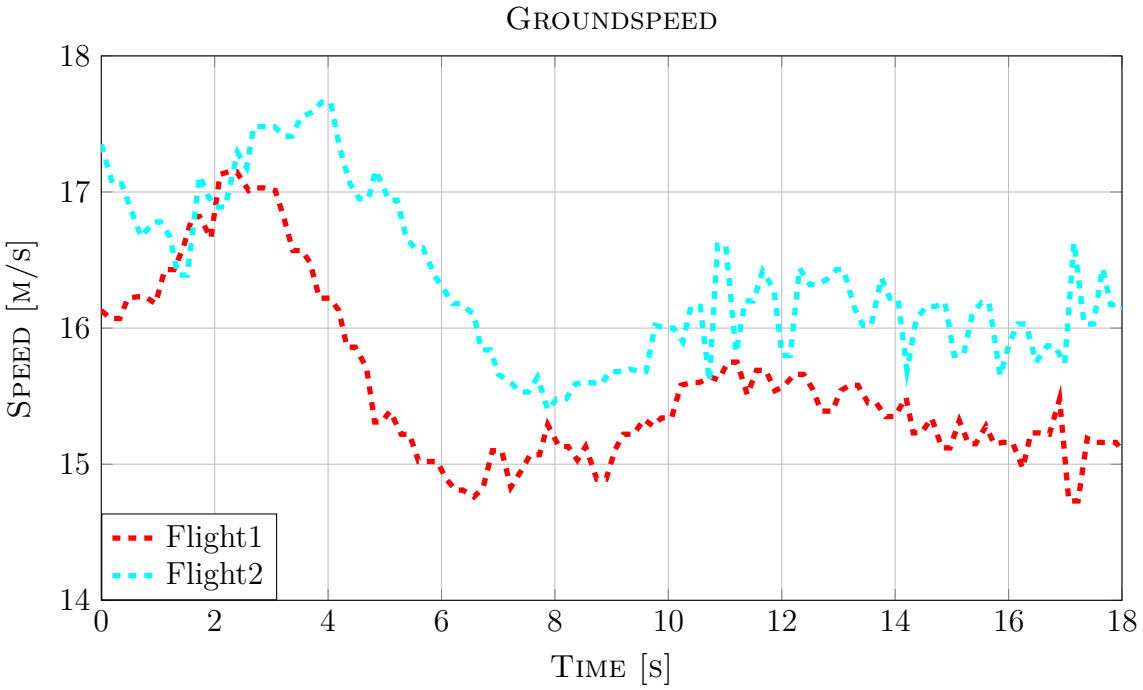


Figure 3.37: Ground speed during FA of flight tests.

3.4.3.4 RTK-GPS solution quality during flight

During a full day of testing on the 1st of June 2014, the GPS solution obtained (with roll compensation) is seen in Table 3.3. Post-takeoff data is shown.

Start time (UTC)	Fix (%)	Float (%)	Single (%)	Satellites	Duration (min)
07:14	25.9	74.1	0	10-12	48
08:36	35.3	64.7	0	8-11	44
10:14	32.7	67.3	0	7-9	40
11:53	44.7	54.6	0.8	6-8	45
13:31	49.6	48.5	1.9	7-9	43
14:50	25.2	74.8	0	7	18
16:27	29.7	70.3	0	9-10	40
18:10	60.0	39.1	0.9	5-7	4

Table 3.3: RTK-GPS solution during testing - 1st of June 2014

The pilot performs maneuvers in manual flight, which puts the UAV in a position where obtaining a RTK-GPS fix solution would be difficult (sharp turns, dives etc), which explains the relatively high percentage of float.

From the data presented it's obvious that the number of visible satellites is not the only deciding factor when attempting to obtain a fixed RTK-GPS solution. It is in combination with a stable-, high SNR from all the visible satellites used to compute the integer ambiguity resolution. Looking at Appendix F.1, it's seen that a stable SNR high is obtained leading to a RTK-GPS solution equal to fix. The most common reasons as to why the SNR signal might be noisy/low are:

- Multipathing
- Noise (low elevation)
- Severe changes in attitude

Comparing the SNR values previously seen, with Appendix F.2, it's possible to compare satellite by satellite. Now, even with a high average SNR value with a float RTK-GPS solution, the signal is temporarily noisy, which leads to difficulties when evaluating the integer ambiguities. As the signal for these measurements were taken in a short time span, it's assumed that the reason most contributing to the noise seen in Appendix F.2, is changes in attitude. As roll is compensated (to a certain point), a compensation in pitch can reduce the remaining noise due to change in pitch as well.

3.4.4 Discussion

Results put forth in Section 3.4.3.3 speaks to the repeatability of the system. There are, however, indications of less-than-optimal tuning of the controllers governing the actuators. Witnessed in Figure 3.30 is the undulating nature of the altitude error. It is the result of non-optimal gains for custom pitch guidance PID-controller. Although non-optimal, it converges towards zero as the magnitude of the oscillations decreases with time. The cross-track error, given in Figure 3.31, converges to zero as well. It decreases at different rates for a given flight. This is due to the varying groundspeed seen in Figure 3.37. An increase in demanded roll angle may be obtained by increasing gains pertaining to the L_1-controller and is the next natural step in the tuning process.

The aircraft is more responsive at greater speeds, which makes the airframe less affected by disturbances such as cross winds. In addition, errors in altitude and cross-track are more rapidly corrected. There is, of course, a limit to how fast the aircraft may travel when impacting the net. Speeds around 16 $\frac{m}{s}$ were deemed safe to land in.

At a later stage a throttle-cut function was implemented. Figure 3.36 shows two separate attempts at static-net retrieval. The net was not mounted, only the crepe paper markers. Although the two separate attempts may seem to have a different cut-off distance, they have the same pre-set distance. Speed experienced throughout the FA has shifted the two cut-off points in time since the time axis shows zero as the plane enters FA.
Looking at the figures of Section 3.4.3.3 it is clear that further tuning of all three top-level controllers;

- Throttle
- Pitch
- Roll

is a necessity. Although the controllers were not tuned to perfection, the system performed as intended and repeatable results were obtained. Improvements on overall accuracy may

be made by further tuning of both the low-level actuator controllers and the custom guidance controllers presented in this thesis.

To remedy the inability to descend to demanded altitude, seen in Section 3.4.3.1, one or more of the following actions may be taken;

- Reduce speed, in order to reduce lift, either by reducing the vertical approach angle or otherwise command the guidance of the aircraft over a longer period of time.
- Decrease boundary for minimum commandable pitch angle. This option does not take into consideration the increase in speed due to a greater downward pitch angle. Speed at target may be too great to safely impact the net.

Chapter 4

Conclusion

Two controllers, separated by the lateral and longitudinal axes, were created to accomodate the baseline vector produced by RTKLIB. Minor changes were also made to allow landing in a moving target which had the freedom to move about in the EN plane and rotate about the Up-axis. Section 3.1 shows the feasibility of the controllers utilizing the, when noise was not specifically added, completely accurate baseline vector provided by the simulator. Repeated tests were run without wind, with moderate wind, with moderate wind included noisy position measurements, for both stationary- and moving target. A 100 % success rate was observed when counting hits within ± 1 m in the horizontal plane, and ± 1 m in the vertical plane, a successful hit. When real-life tests were performed, a lot of effort went into tuning and reconfiguration of software and hardware until a satisfactory setup was found. From that point on, satisfactory repeatability of impact point, without mounting the net, was achieved (shown in Section 3.4.3.3) and crowned with a successful net retrieval for the static target scheme. The repeatability of the real-life flight tests showed that the reliable behaviour of the simulator model was retained when implemented on another aircraft platform in real-life. Visual confirmation found in Appendix C - Video.

Significant effort went into finding a stable, and efficient, hardware configuration. Early designs included antennas for transferring telemetry and correction, for the RTK-GPS solution calculation made on-board the UAV, separately. This was deemed unpractical as the range of the system was severely limited by the throughput and range of the telemetry antenna [2], as well as the radiation pattern of the NanoStation M5. The final hardware configuration included more powerful transceivers, in the form of the Rocket M5 radio, combined with higher gain, dipole antennas. This resulted in a very stable communication link for the distances experienced when operating the particular airframe at line-of-sight flight. Although the aircraft was able to carry the 1 kg payload, not much more weight may be added before reaching the maximum gross weight of the aircraft. The purpose of the system was to act as a pilot for the duration of the flight, harbouring payloads for various data sampling. As the capacity of the airframe, with respect to weight, is at its maximum already, the payload may be concluded to be to heavy. A weight reduction of the payload is therefore required in order for it to fulfill the true purpose of the system.

The software, run on the APM 2.6, is largely the standard, open-source, v.2.78 ArduPlane.

Modifications, mentioned in Section 2.5.3, were made to enable guidance controllers utilizing the baseline vector. This particular product is an off-the-shelf product which comes at little cost compared to more advanced proprietary autopilot systems. Additions made to the software were kept modularized, with only a moderate amount of code insertions needed in the original file set. The platform worked well for the computationally cheap custom guidance algorithms, but upgrading to a more powerful platform is inevitable if more sophisticated algorithms are to be implemented.

Final approach speed was another factor needed to be monitored. This may be incorporated as being a criterion for performing an evasive maneuver. However, this feature was not active during testing as it would complicate the main task of net retrieval. The static net has no form of active- or passive dampers to absorb the energy of the impact. Only the flex in mounting materials are present to mitigate the forces acting on the airframe. Upon impact, referring to the single net retrieval achieved in this project, the flex of the mounting was sufficient to safely decelerate the aircraft from roughly $16\frac{m}{s}$ to $0\frac{m}{s}$. Its size, measuring 5 m wide by 3 m high, was also sufficient as repeatability of the less-than-optimal tuned aircraft was within the boundaries (1x1 m) on 100 % of the attempts shown in Figure 3.30 and Figure 3.31. The high success rate was, in this case, aided by a constant headwind during FA.

Chapter 5

Further work

This chapter will suggest specific steps to improve the performance of the system with the current available hardware.

5.1 Mathematical model of UAV

There are several ways to mathematically describe the motion of an aircraft depending on how the model is to be used. A complete nonlinear description of the six degrees of freedom, inherent to the aircraft, may be reduced to a linear model using perturbation theory [12]. A further simplification can be made by decoupling the linear model in the lateral and the longitudinal axes. Included in the model is aerodynamic coefficients and propeller thrust. The former may be determined either by wind-tunnel experiments or by system identification. System identification requires logged data.

A mathematical model will enable the use of more advanced controllers and also produce a more realistic model for a simulator, which, in the long run, saves time and increases UAV performance.

5.1.1 Wind estimation

A step that can be taken, after a mathematical model is found, is to implement wind estimation, as opposed to the wind compensation that already exists in the control algorithms. Strong winds can severely affect the behaviour of a UAV when attempting to follow a desired flight path, especially during FA, when the controllers are as aggressive as possible. A mathematical model with wind estimation can be used to optimize the desired flight path during FA, to ensure the landing is performed in a desirable way.

5.2 Improved attitude estimation

As the setup presented in this thesis is a development platform, the next step should be to implement an improved attitude estimation method. Looking at [48], the Extended Kalman Filter (EKF) provides a more accurate estimation of the attitude angles, which combined with a mathematical model described in Section 5.1, gives more freedom

when designing the control algorithms for the UAV. The APM is, computationally, an insufficient platform for running the EKF. As the Pandaboard is already included in the payload to compute the baseline vector, it can be used instead. The APM can send its raw gyroscope- and accelerometer readings to the Pandaboard through its serial ports, then attitude is estimated, and finally sent back to the APM. A different approach includes an external IMU of greater quality than the one included in the APM. The final estimated attitude is transmitted to the APM.

5.3 2D compensation of GPS antenna

It was argued in Section 3.3.3, based on the results given in Section 3.3, that keeping the mounting plane of the GPS antenna level was advantageous in keeping the RTK-GPS solution stable. By including compensation of the pitch axis movement, the antenna will be kept level if the roll- and pitch angles do not exceed their boundary value. Overall, this will improve the stability of the relative position measurement by maintaining a constant set of satellite PRN numbers within view. By constant, the motion of the satellites relative to the earth, causing them to disappear below the horizon, is not included.

Bibliography

[1] Inertial navigation system overview. *Bulletin of advanced tecnology research*, 5:11–16, 2011.

[2] 3DRobotics. 3dr radio telemetry set. `https://store.3drobotics.com/products/3dr-radio`, 2013. Accessed 2013-11-26.

[3] 3DRobotics. Apm power module with xt60 connectors. `https://store.3drobotics.com/products/apm-power-module-with-xt60-connectors`, 2014. Accessed 2014-05-01.

[4] Randal W. Beard and Timothy W. McLain. *Small Unmanned Aircraft, Theory and Practice*. Princeton University Press, 2012.

[5] Klaus Betke. The nmea 0183 protocol. Technical report, May, revised August 2000.

[6] Blender. About the software blender. `http://www.blender.org/about/`, 2014. Accessed 2014-05-17.

[7] Geoffrey Blewitt. Basics of the gps technique: Observation equations. Technical report, Department of Geomatics, University of Newcastle, United Kingdom, 1997.

[8] X.-W. Chang, X. Yang, and T.Zhou. Mlambda: a modified lambda method for integer least-squares estimation. *Journal of Geodesy*, 79:552–565, 2005.

[9] Anne-Marie Dinius. Gps antenna multipath rejection performance. Technical report, Lincoln Laboratory, Massachusetts Institute of Technology, 1995.

[10] L.F. Faleiro and A.A. Lambregts. Analysis and tuning of a 'total energy control system' control law using eigenstructure assignment. *Aerospace Science and Technology*, 1(3):127–140, 1999.

[11] Thor I. Fossen. *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley and Sons, Ltd., 2011.

[12] Thor I. Fossen. Mathematical models for control of aircraft and satellites, 3rd edition, April 2013.

[13] Mohinder S. Grewal, Lawrence R. Weill, and Angus P. Andrews. *Global Positioning Systems, Inertial Navigation, and Integration*. Wiley-Interscience, 2007.

[14] Finn Haugen. *Praktisk Reguleringsteknikk, 2. utgave*. Tapir Akademisk Forlag, 2009.

[15] HobbyKing.com. Description of the fuselage. https://www.hobbyking.com/hobbyking/store/uh_viewitem.asp?idproduct=27132, 2013. Accessed 2013-11-26.

[16] Sungsik Huh and David Hyubchul Shim. A vision-based automatic landing method for fixed-wing uavs. *Journal of Intelligent and Robotic Systems*, 57:217–231, January 2010.

[17] H. Jin Kim, Mingu Kim, Hyon Lim, Chulwoo Park, Seunho 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. *IEEE/ASME Transactions On Mechatronics*, 18(4):1320–1333, August 2013.

[18] J.A. Klobuchar. Ionospheric time-delay algorithm for single-frequency gps users. *IEEE Transactions on Aerospace and Electronic Systems*, AES-23(3):325–331, May 1987.

[19] Wonmo Koo, Sangkyung Sung, and Young Jae Lee. Development of real-time heading estimation algorithm using magnetometer/imu. 2009.

[20] Antonius A. Lambregts. Tecs generalized airplane control system design - an update. *Chapter in book, Advances in Aerospace Guidance, Navigation and Control*, pages 503–534, 2013.

[21] Alfred Leick. *GPS Satellite Surveying*. John Wiley and Sons, Inc., third edition edition, 2004.

[22] Timothy R. Lemmon and George P. Gerdan. The influence of the number of satellites on the accuracy of rtk gps positions. *The Austrailian Surveyor*, 44(1):64–70, June 1999.

[23] Active Research Limited. The nmea 0183 information sheet, issue 3. Technical report, Active Research Limited, Dorset, UK, 2011.

[24] Future Technology Devices International Limited. Ttl to usb serial converter range of cables, datasheet. http://www.ftdichip.com/Support/Documents/DataSheets/Cables/DS_TTL-232R_CABLES.pdf, 2014. Accessed 2014-05-14.

[25] Greg Loegering and Steve Harris. Landing dispersion results - global hawk auto-land-system. *American Institute of Aeronautics and Astronautics*, May 2002.

[26] Jean Louis Naudin. Rc maxiswift flying wing uav 1.00. http://forums.x-plane.org/index.php?app=downloads&showfile=21158. Accessed 31.05.2014.

[27] Ubiquiti Networks. Rocket m5 datasheet. http://dl.ubnt.com/datasheets/rocketmgps/Rocket_M_GPS_Datasheet.pdf, 2011. Accessed 2014-04-26.

[28] Ubiquiti Networks. Nanostation m5 datasheet. http://http://dl.ubnt.com/datasheets/nanostationm/nsm_ds_web.pdf, 2013. Accessed 2014-05-21.

[29] Sanghyuk Park, John Deyst, and Jonathan P. How. A new nonlinear guidance logic for trajectory tracking. The American Institute of Aeronautics and Astronautics, August 2004.

[30] APM Plane. About arduplane. `https://code.google.com/p/ardupilot-mega/`, 2013. Accessed 2013-12-03.

[31] APM Plane. Tecs (total energy control system) for speed and height tuning guide. `http://plane.ardupilot.com/wiki/tecs-total-energy-control-system-for-speed-height-tuning-guide/`, 2013. Accessed 2013-11-29.

[32] QGroundControl. Mavlink micro air vehicle communication protocol. `http://qgroundcontrol.org/mavlink/start`, 2013. Accessed 2013-12-03.

[33] J. Saastamoinen. Atmospheric correction for the troposphere and stratosphere in radio ranging of satellites. Technical report, Photogrammetric Research, Division of Physics, National Research Council of Canada, 1972.

[34] T. Takasu. Rtklib ver. 2.4.2 manual. `http://www.rtklib.com/prog/manual_2.4.2.pdf`, April 2013. Accessed 2013-11-13.

[35] Tallysman. Tw2105 embedded precision gps l1 antenna. `http://tallysman.com/TW2105.php`, 2013. Accessed 2013-12-12.

[36] P.J.G. Teunissen. Least-squares estimation of the integer gps ambiguities. International Association of Geodesy, August 1993.

[37] P.J.G. Teunissen, P.J. De Jonge, and C.C.J.M. Tiberius. The lambda-method for fast gps surveying. In *Proceedings of International Symposium 'GPS technology applications*, pages 26–29, 1995.

[38] P.J.G. Teunissen, P.J. De Jonge, and C.C.J.M. Tiberius. The least-squares ambiguity decorrelation adjustment: a method for fast gps integer ambiguity estimation. *Journal of Geodesy*, 70:65–82, 1995.

[39] u blox. Gps antennas. Technical report, u-blox, 2009.

[40] u blox. *u-blox 6 Receiver Description*. u-blox, revision 7.03 (public release) edition, Nov 2011. Including protocol specification.

[41] u blox. Gps, essentials of satellite navigation, 2013. Document number GPS-X-02007-D.

[42] u blox. Lea-6t/neo-6t product description. `http://www.u-blox.com/images/downloads/Product_Docs/LEA-NEO-6T_ProductSummary_%28GPS.G6-HW-09020%29.pdf`, 2013. Accessed 2013-10-28.

[43] WiMo Antennen und Elektronik GmbH. Datasheet wimo 18720.3 / 18720.3h antenna. `http://www.wimo.de/download/18720.3.pdf`. Accessed 2014-05-26.

[44] WiMo Antennen und Elektronik GmbH. Gp5000-10 5 ghz omnidirectional antenna for wlan / ism datasheet. http://www.wimo.de/download/18715-10.pdf. Accessed 2014-05-26.

[45] Bjørnar Vik. Integrated satellite and inertial navigation systems. Department of Engineering Cybernetics, NTNU, 2012.

[46] B. Wisniewski, K. Bruniecki, and M. Moszynski. Evaluation of rtklib's positioning accuracy using low-cost gnss receiver and asg-eupos. *International Journal on Marine Navigation and safety of Sea Transportation*, 7(1):79–85, March 2013.

[47] XPlane. Xplane 10 desktop manual. http://www.x-plane.com/v10-online-desktop-manual/#overview, 2013. Accessed 2013-12-03.

[48] Guoyu Zuo, Xiaogang Ruan Kai Wang, and Zhen Li. Multi-sensor fusion method using marg for a fixed-wing unmanned aerial vehicle. In *Proceedings of 2012 IEEE International Conference on Mechatronics and Automation August 5 - 8, Chengdu, China*, 2012.

Appendix A

GPS tests Eggemoen

A.1 Day 1

During the first test day, the following settings were used on the GPS receivers and in RTKLIB:

Test	RTKLIB Positioning Mode	Update rate	Rec dynamics
1	Moving base (fix and hold)	1Hz	Both: Airborne < 1g
2	Moving base (continuous)	5Hz	Both: Airborne < 4g
3	Kinematic (continuous)	5Hz	Both: Airborne < 4g
4	Kinematic (fix-and-hold, switch made from Test 3 in flight)	5Hz	Both: Airborne < 4g
5	Moving base (fix-and-hold)	5Hz	Both: Airborne < 1g
6	Moving base (continuous, switch made from Test 5 in flight)	5Hz	Both: Airborne < 1g
7	Kinematic (continuous)	5Hz	Rov: Automotive base: Stationary
8	Kinematic (fix-and-hold, switch made from Test 7 in flight)	5Hz	Rov: Automotive base: Stationary
9	Moving base (fix-and-hold)	5Hz	Rov: Airborne < 1g base: Stationary
10	Kinematic (continuous, switch made from Test 9 in flight)	5Hz	Rov: Airborne < 1g base: Stationary

Table A.1: Settings of RTKLIB and GPS receivers

With these settings, the following results were produced

Test num	FIX num samples / %	FLOAT num samples / %	SINGLE num samples / %
1	153/11,5	1153/87	20/1,5
2	84/1,4	5341/90,7	461/7,8
3	788/19	3353/81	0/0
4	0/0	1068/100	0/0
5	314/11,8	2334/88,1	2/0,1
6	71/3,4	1933/93,6	61/3
7	142/5,5	2454/94,5	0/0
8	89/4,4	1924/95,6	0/0
9	582/13,3	3799/86,7	0/0
10	94/6	1467/94	0/0
total number	2317	24826	544
total percentage	8,37	89,67	1,96

Table A.2: Table indicating quality of position solution produced by RTKLIB for duration of test (including time spent in launcher pre-takeoff).

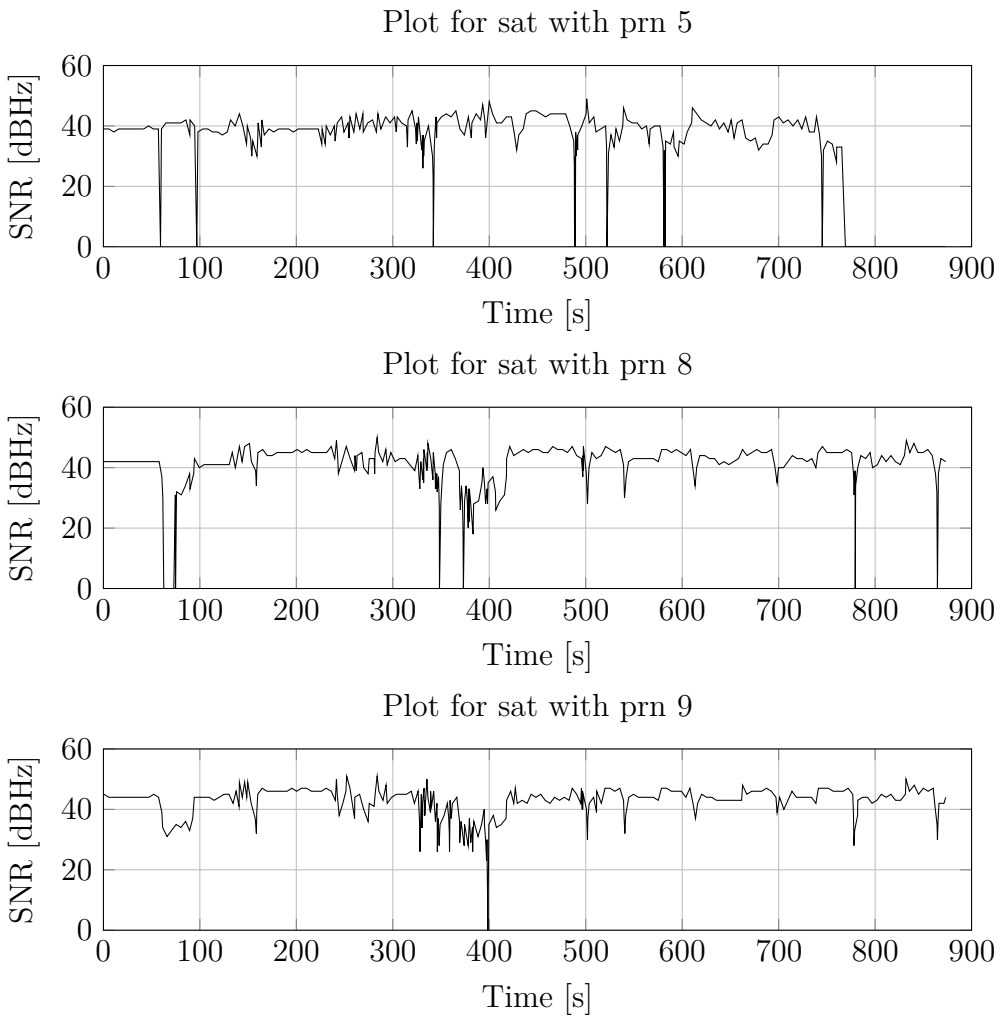


Figure A.1: SNR (black line)of certain satellites for Test 9, Day 1 at Eggemoen.

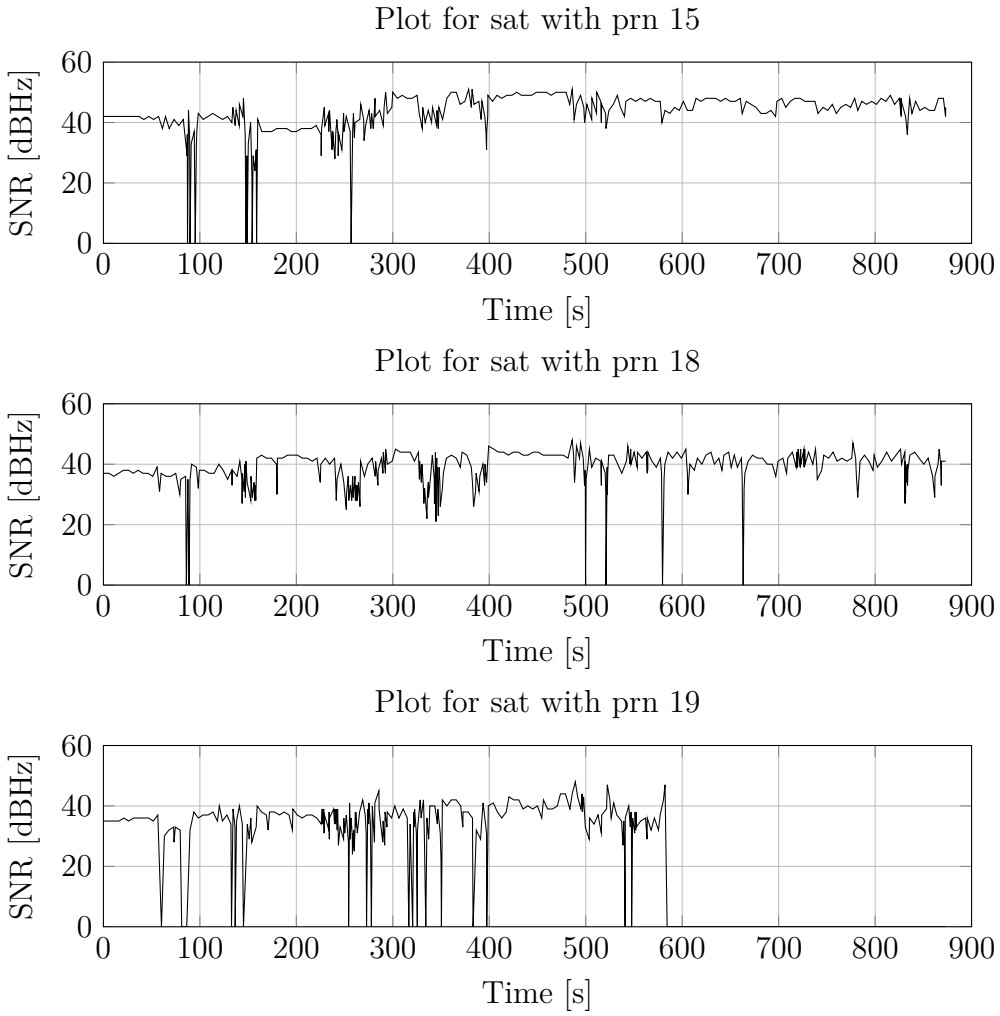


Figure A.2: SNR (black line)of certain satellites for Test 9, Day 1 at Eggemoen.

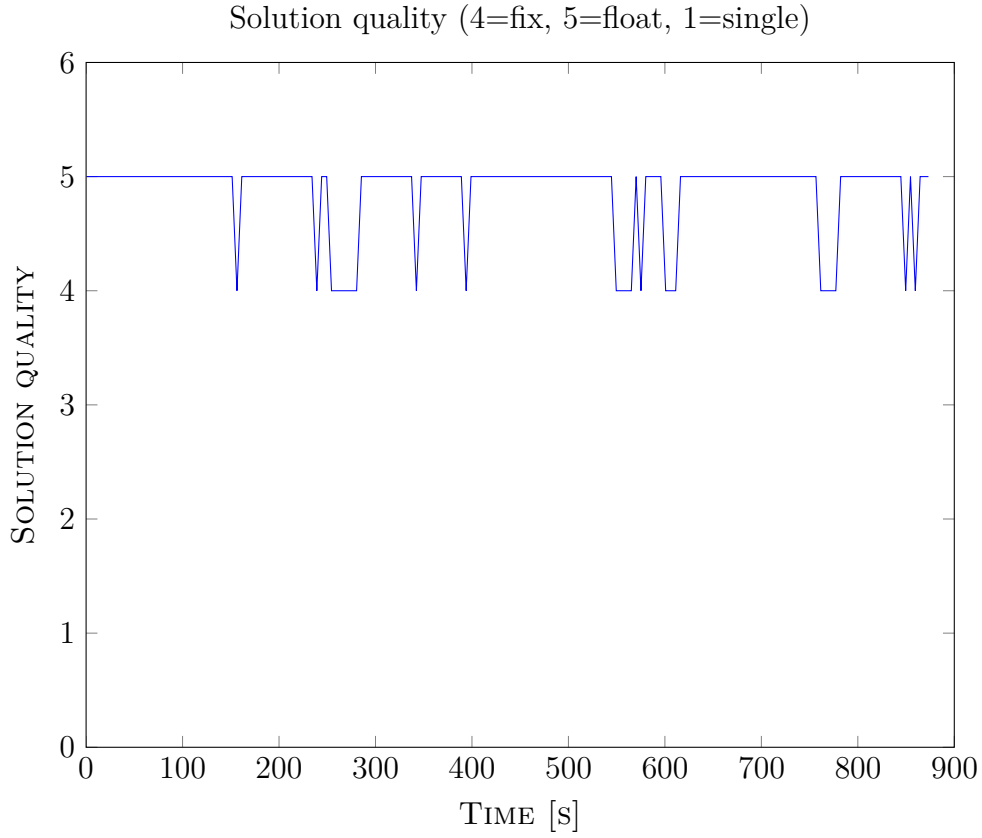


Figure A.3: Solution quality for Test 9, Day 1. As the title claims; 1 equals single receiver positioning, 4 equals RTK fix and 5 equals RTK float.

A.2 Day 2

Common settings for flight tests are:

- Base station GPS receiver has its dynamic platform setting set to *Stationary* for all tests.
- The positioning mode of RTKLIB was set to moving base for all tests.
- The minimum ratio to fix ambiguity was set to 3.0.
- Integer ambiguity resolution managed by way of the Fix-and-hold method used by RTKLIB.
- Elevation angle of satellites limited to above 10 degrees unless otherwise stated.

Test	Rover receiver dynamics	Elevation angle
1	Airborne $<$ 1g	$\geq$ 10
2	Automotive	$\geq$ 10
3	Pedestrian	$\geq$ 10
4	Airborne $<$ 4g	$\geq$ 10
5	Airborne $<$ 4g	$\geq$ 5
6	Airborne $<$ 4g	$\geq$ 20
7	Airborne $<$ 2g	$\geq$ 10

With these settings, the following results were produced

Test	FIX num samples / %	FLOAT num samples / %	SINGLE num samples / %
1	257/3,4	6556/87,4	685/9,1
2	466/15,9	2243/76,6	219/7,5
3	107/3,2	3220/95,9	30/0,9
4	815/15,3	4174/78,4	335/6,3
5	350/25,1	1035/74,1	12/0,9
6	444/14,3	1367/44	1296/41,7
7	526/17,1	2320/75,5	227/7,4
total number	2965	20915	2804
total percentage	11,11	78,38	10,50

Table A.3: Table indicating quality of position solution produced by RTKLIB for duration of test (including time spent in launcher pre-takeoff).

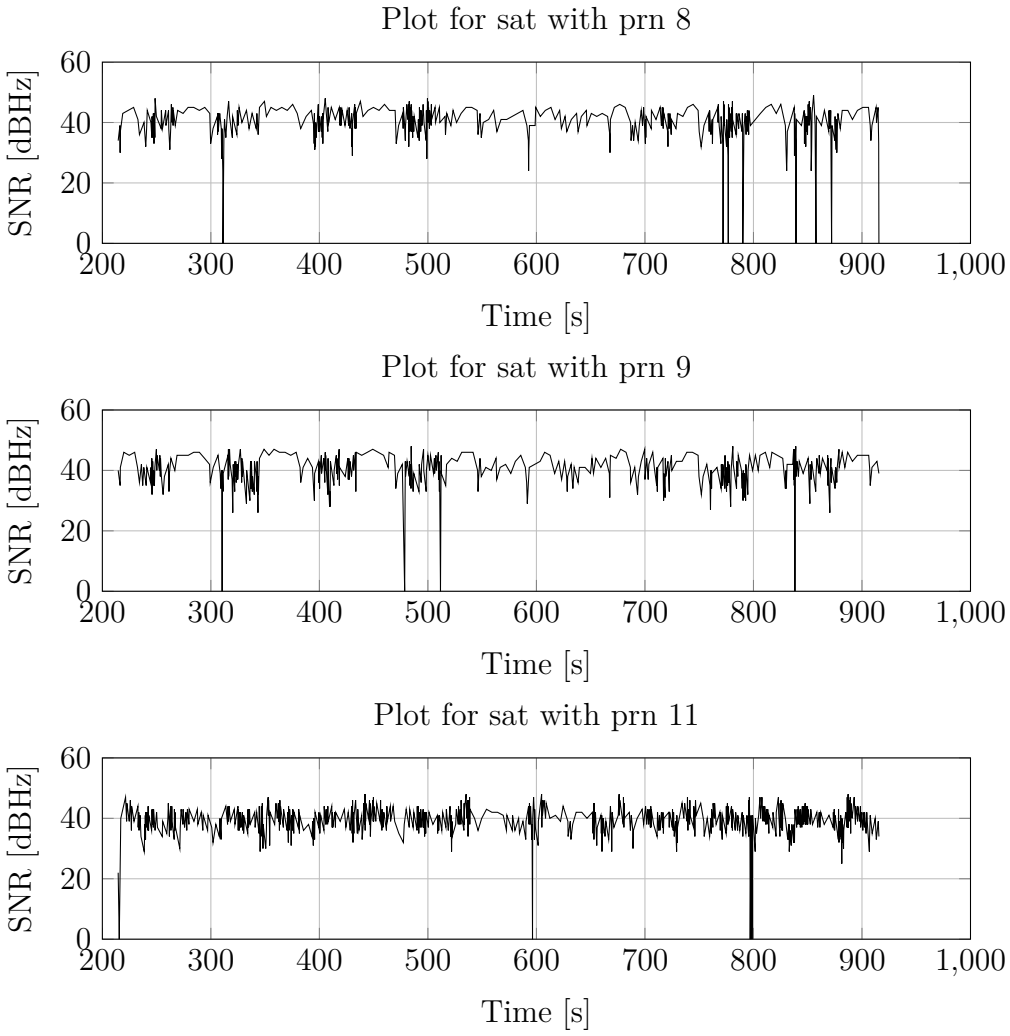


Figure A.4: SNR (black line) of certain satellites for Test 4, Day 2.

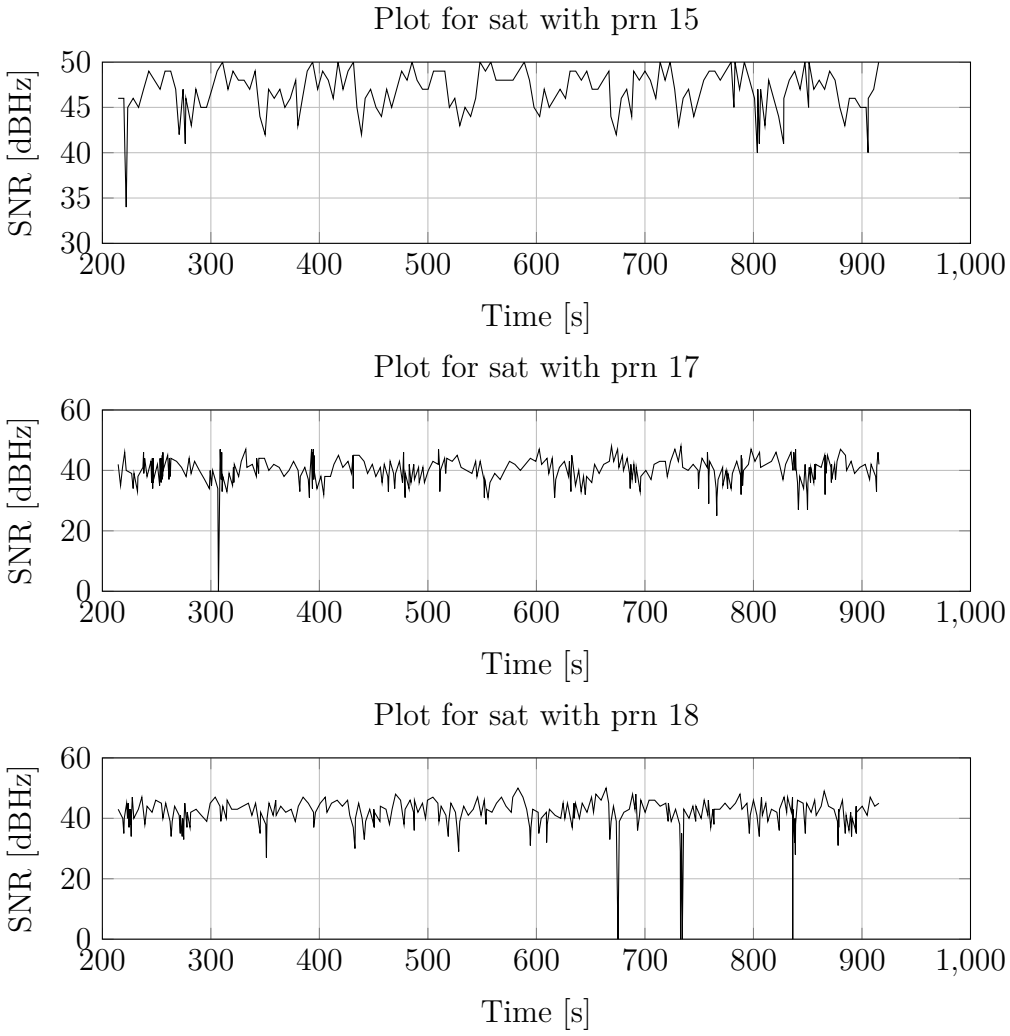


Figure A.5: SNR (black line) of certain satellites for Test 4, Day 2.

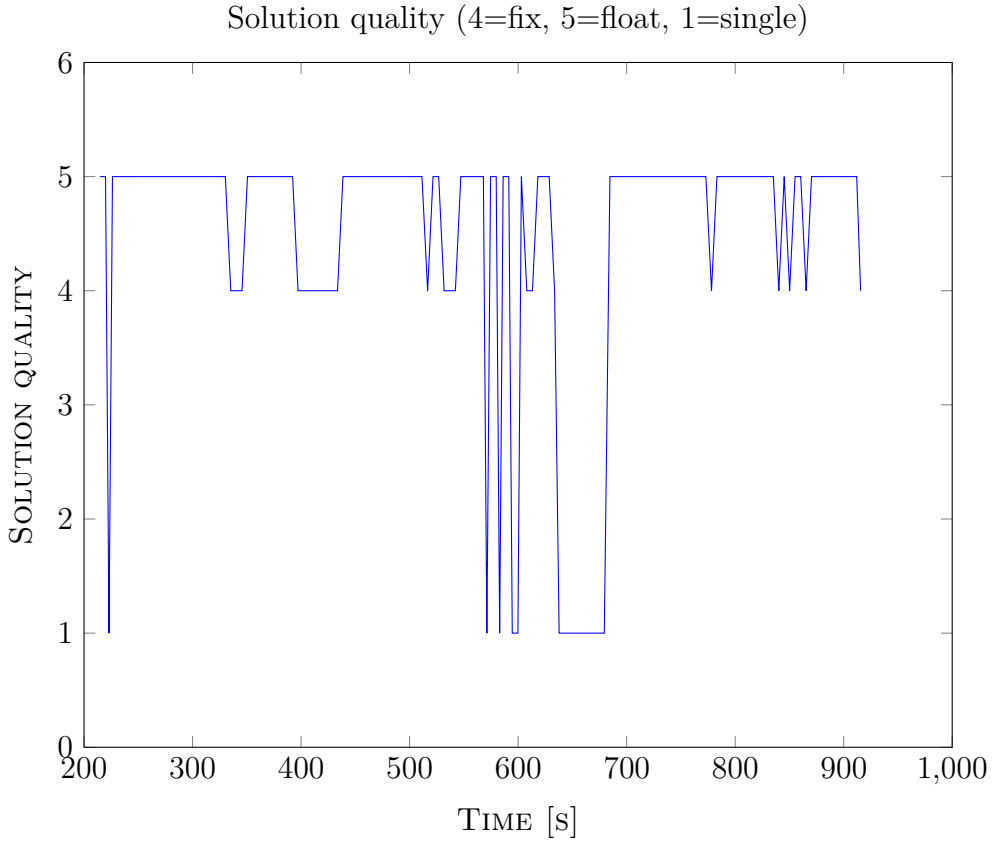


Figure A.6: Solution quality for Test 4, Day 2. As the title claims; 1 equals single receiver positioning, 4 equals RTK fix and 5 equals RTK float.

A.3 Day 3

Common settings for flight tests are the same as for day 2. Hardware-wise, however, a single change was made; an aluminium groundplane was added beneath the UAV GPS antenna. This comes as an addition to the built-in groundplane of the antenna used in the UAV.

Test	Rover receiver dynamics
1	Airborne < 4g
2	Airborne < 4g
3	Airborne < 1g
4	Airborne < 1g

With these settings, the following results were produced

Test	FIX num samples / %	FLOAT num samples / %	SINGLE num samples / %
1	2212/44,6	2501/50,4	250/5,0
2	825/21,5	2984/77,7	31/0,8
3	1292/41,5	1768/56,8	54/1,7
4	2696/53,2	2362/46,6	7/0,1
total number	7025	9615	342
total percentage	41,36	56,61	2,01

Table A.4: Table indicating quality of position solution produced by RTKLIB for duration of test (including time spent in launcher pre-takeoff).

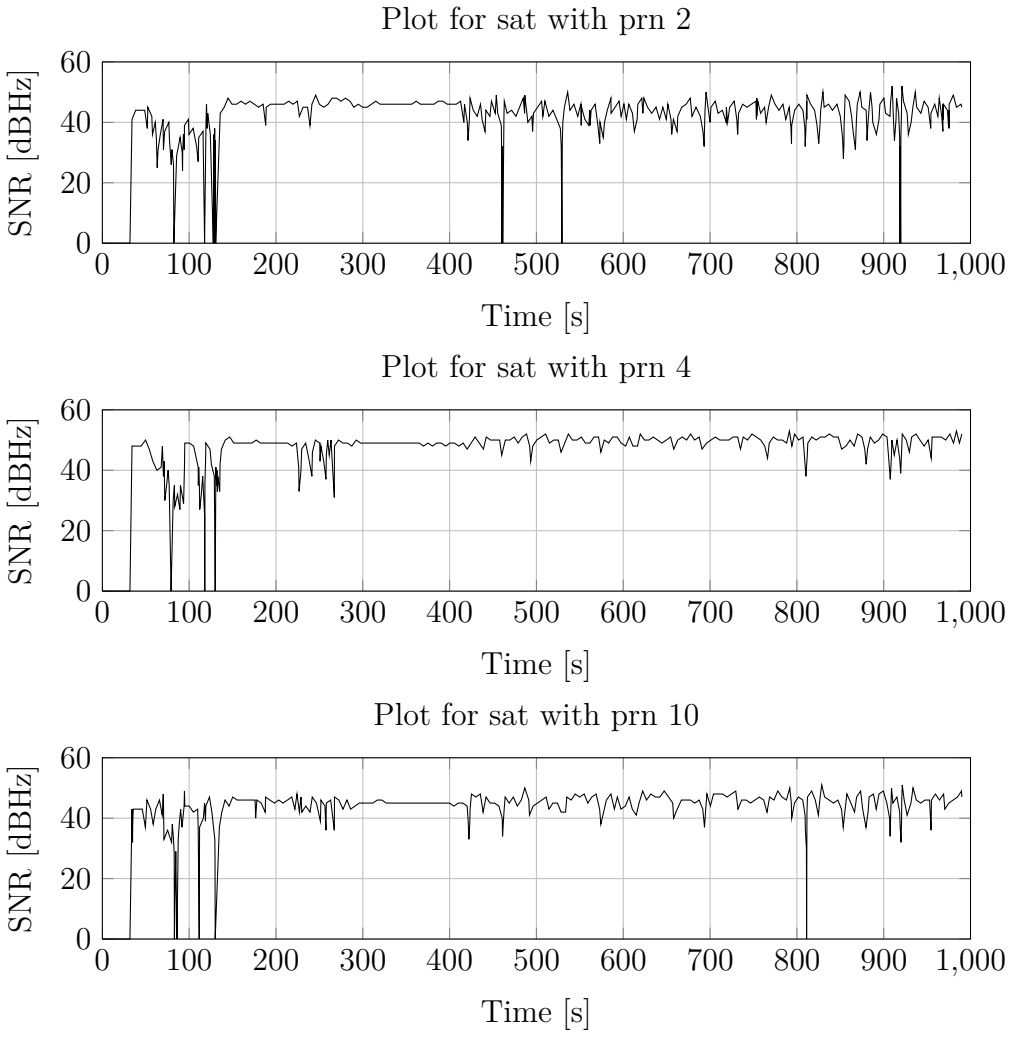


Figure A.7: SNR (black line) of certain satellites for Test 1, Day 3.

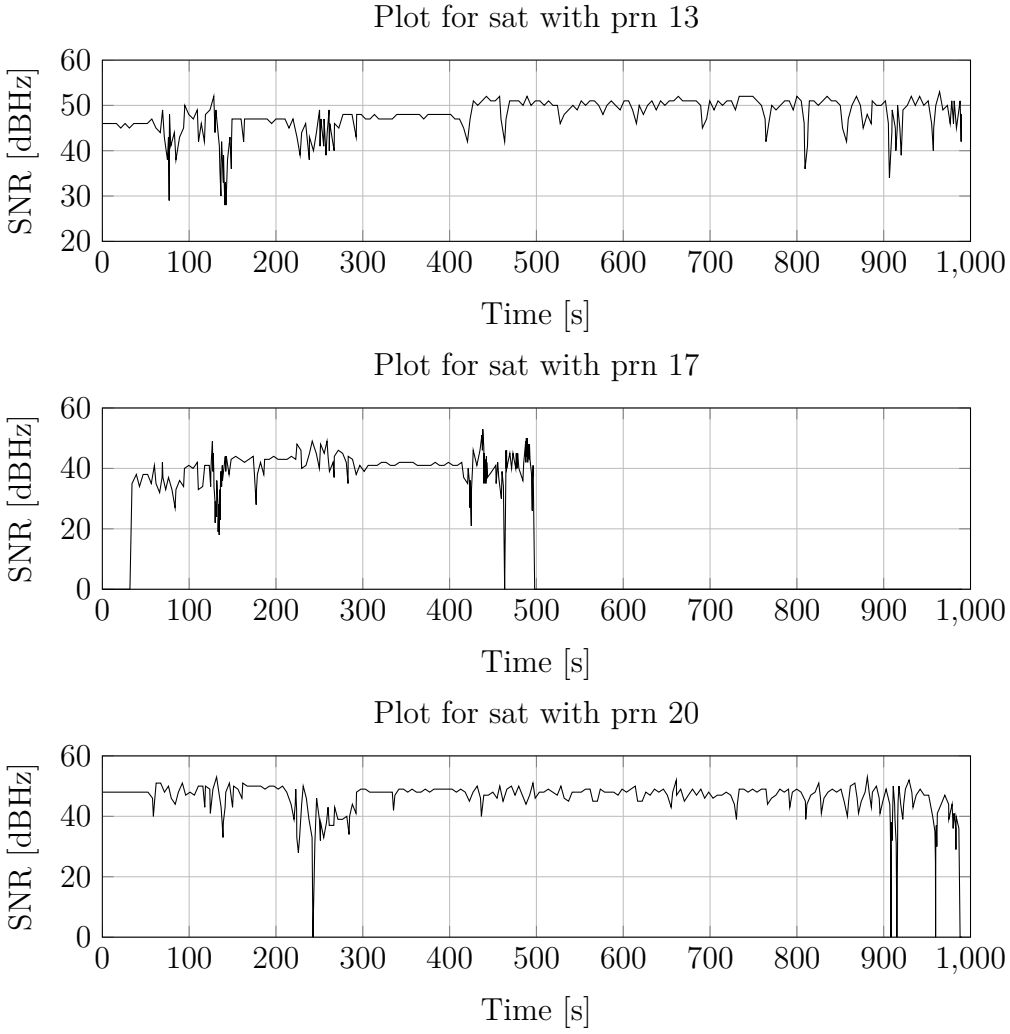


Figure A.8: SNR (black line) of certain satellites for Test 1, Day 3.

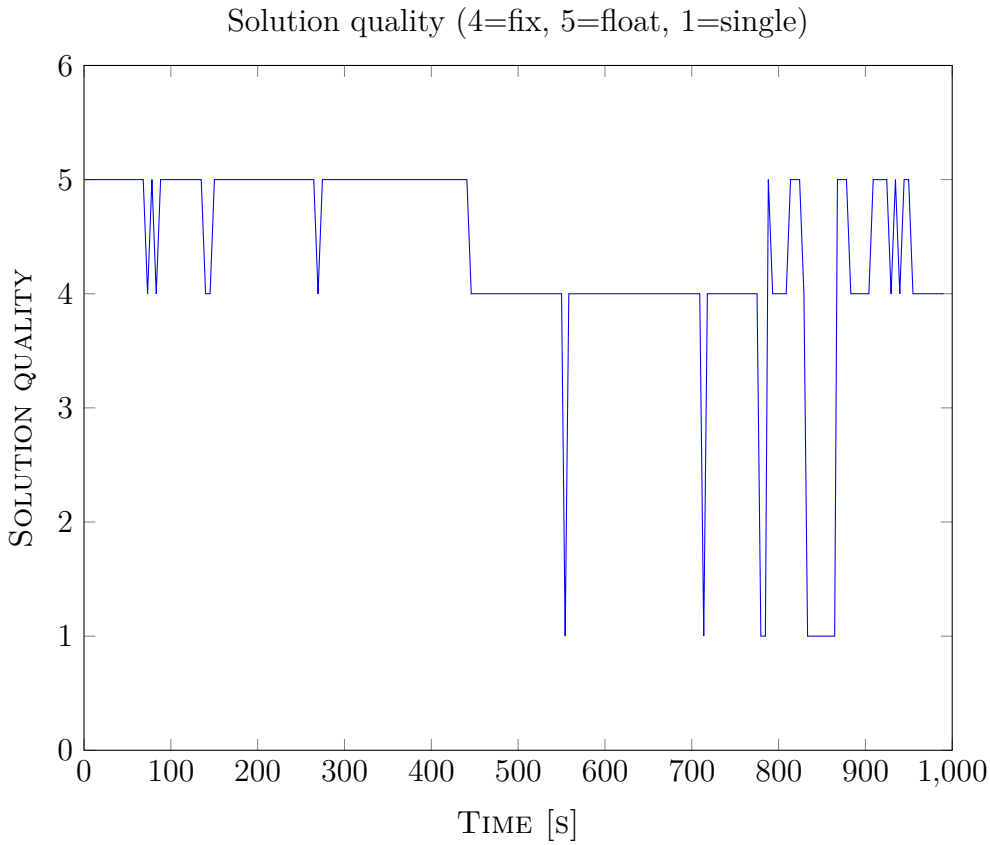


Figure A.9: Solution quality for Test 1, Day 3. As the title claims; 1 equals single receiver positioning, 4 equals RTK fix and 5 equals RTK float.

Appendix B

Hardware

This chapter describes the layout of the payload, system hardware configuration, physical layout of APM and battery duration tests.

B.1 Payload layout

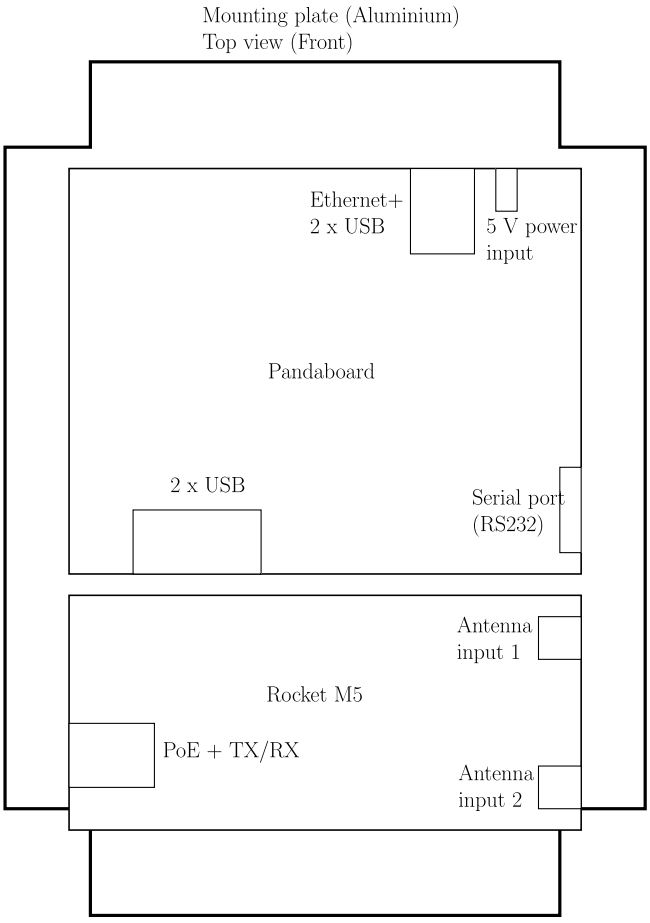


Figure B.1: Payload, showing components mounted on top-side of mounting plate. Dimensions: 14x24cm

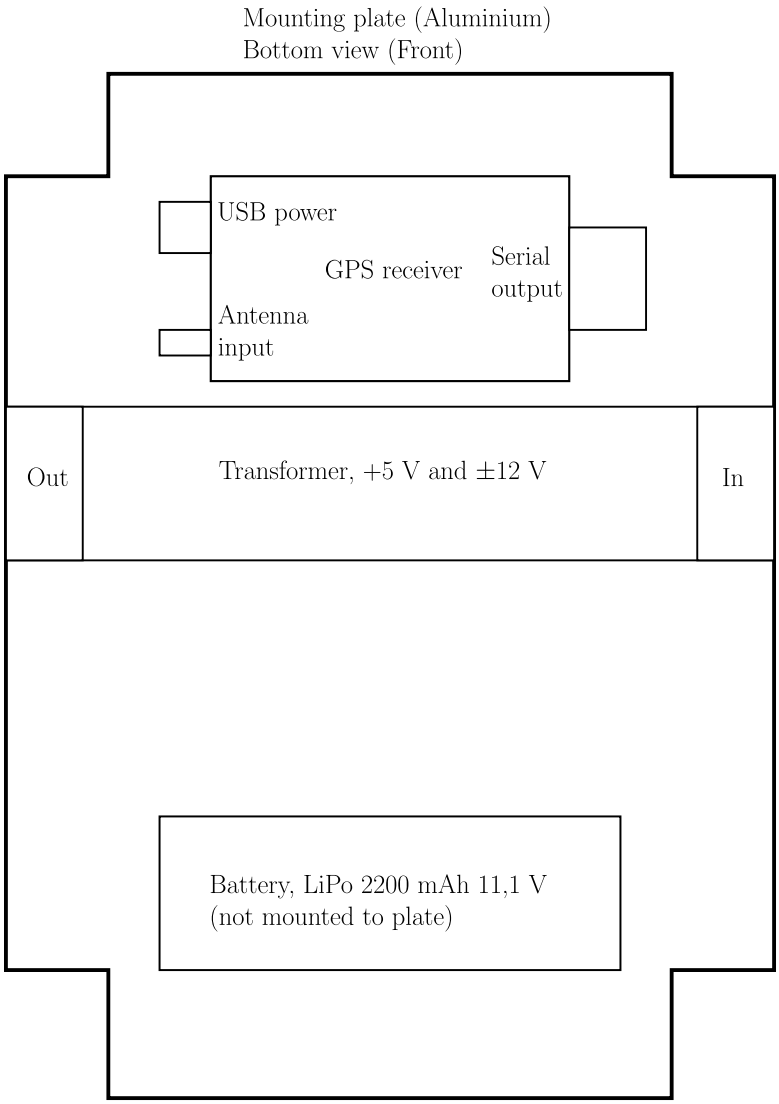


Figure B.2: Payload, showing components mounted on bottom-side of mounting plate. Dimensions: 14x24cm

B.2 System setup

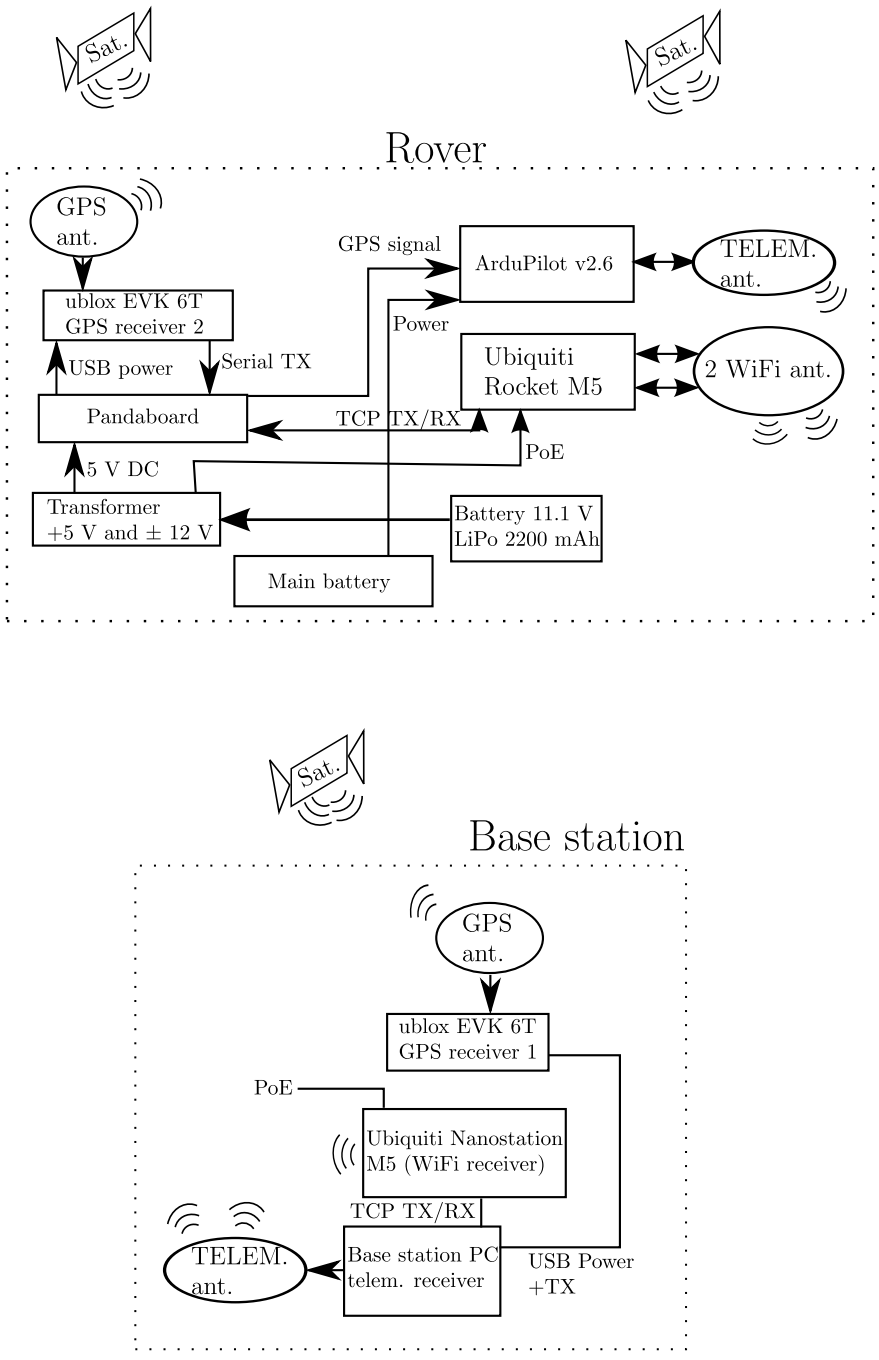


Figure B.3: Preliminary layout of hardware used on UAV and base station.

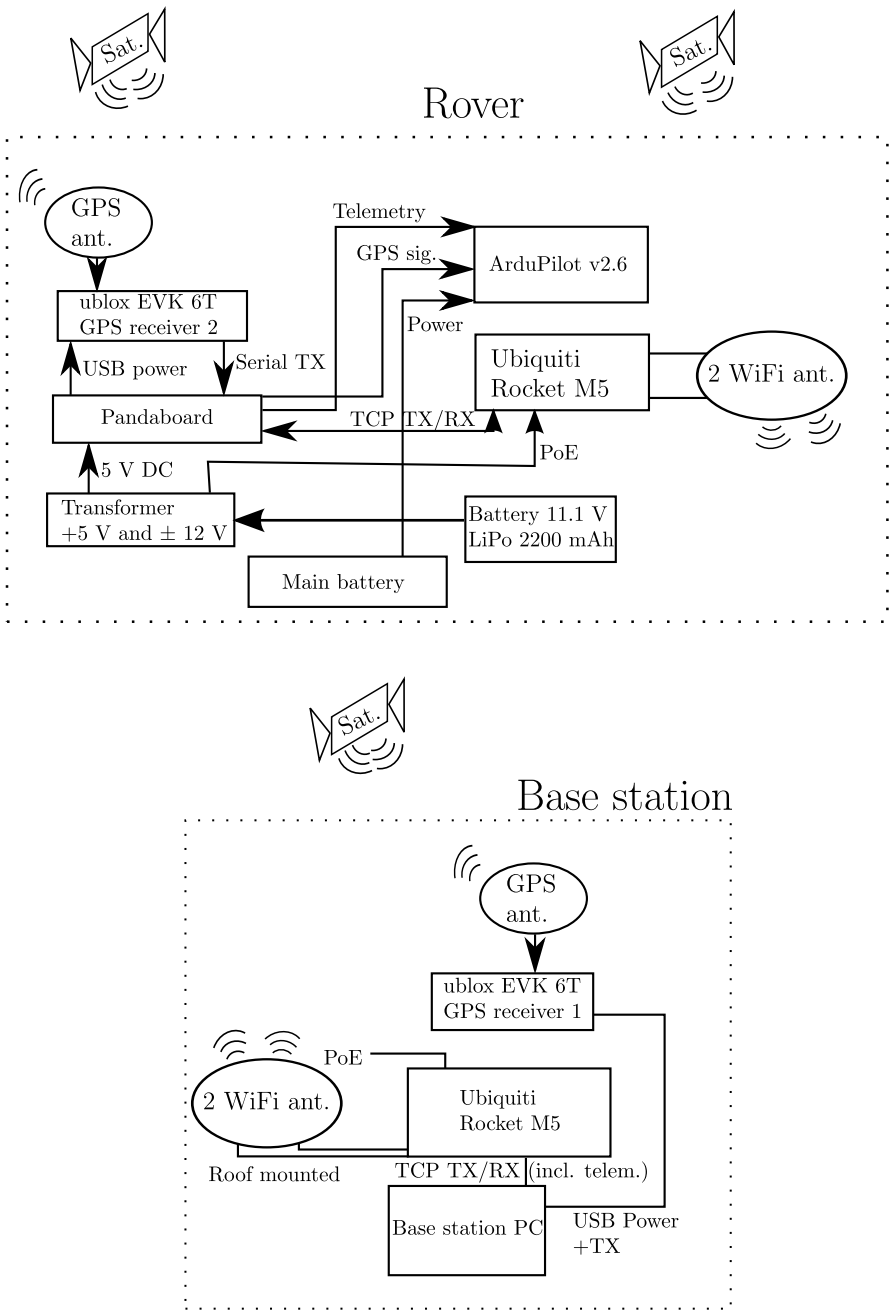


Figure B.4: Improved configuration of hardware used on UAV and base station.

B.3 APM Layout

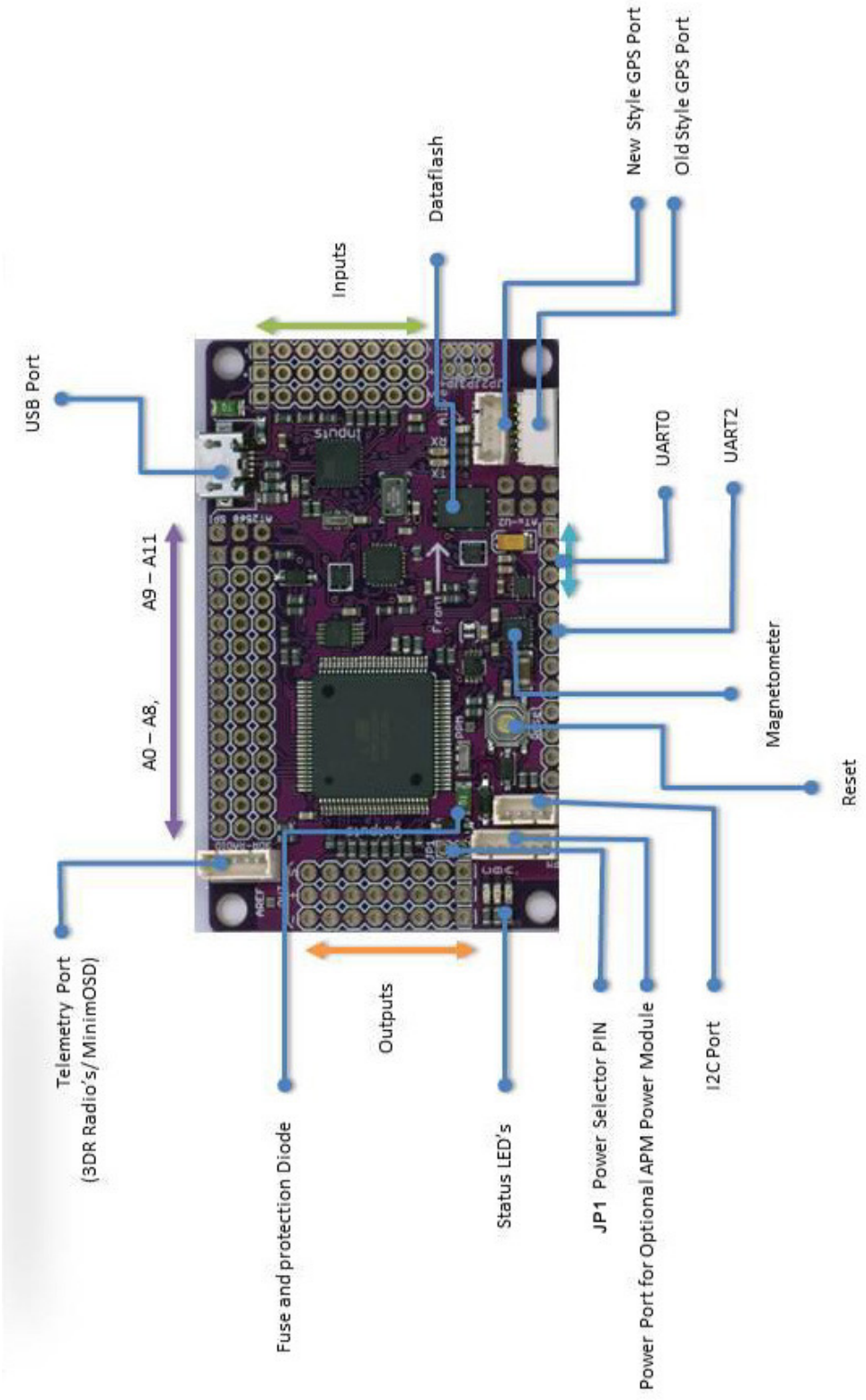


Figure B.5: Layout of the APM 2.5 board

B.4 Payload battery duration

The payload is designed to be completely separate from the rest of the UAV in terms of power dependency and basic control of the plane (i.e. RC options MANUAL and STABILIZE). The payload delivers GPS data to the APM. This makes the APM dependent on the payload for GPS data, which is important for the APM when functioning as an autopilot and critical when implementing the custom net retrieval algorithms. By considering Section 2.2.8, no meter is attached to the payload battery, similar to the APM power module [3], for detecting when battery power runs low during a flight. As this would cause the payload to power down, the APM would lose telemetry from base station and GPS data from payload. To reduce the likelihood of a payload shutdown mid-air, "endurance" tests of the payload were performed in a controlled environment. The goal being to establish a threshold value for maximum time between replacing batteries. The battery used in this test is the same as during flight; LiPo 3s 11.1V 2200mAh.

Test number	Start time (UTC)	End time (UTC)	Total uptime
1	10.58.47	13.25.54	2.27.07
2	08.20.16	10.50.59	2.30.43
3	13.10.56	15.33.14	2.22.18

Table B.1: Duration of payload battery during endurance tests in lab.

The battery in these tests was used to power the system until failure, i.e when the battery was completely empty. As this is not recommended if a long battery-life is desired, the limit on the system was set to 1h45min, when the battery is at approximately 20-30%. As the duration of a normal flight with the X8, using 2x14.8V 5000mAh batteries is around 40 minutes, batteries were swapped out well before they ran out. This power solution will also be able to support additional hardware, if necessary.

Appendix C

Digital appendix

The below table describes the position of files in the digital appendix. The left-most column holds top-level folders with sub-folders to its right and below.

Level 1	Level 2	Level 3	Level 4
Software			
	APM	Static target	ArduPlane
			libraries
		Moving target	ArduPlane
			libraries
	XPlane	plugins	Position of landing target
			Animate an object
		Custom scenery	
		MaxiSwift	
	MP	Files edited	
Video			
Hardware	APM eagle files		

Table C.1: Overview of digital appendix

Appendix D

Lateral controller

This chapter contains information relating to the lateral controller.

D.1 Calculating the chord of a circle

The chord of a circle, given θ_l (found in Section 2.5.3.4) is found by

$$c = r \cos\left((\theta_l + 90)\frac{\pi}{180}\right)$$

D.2 Direction to rotate

Which way to rotate the first vector, which will act as a desired flight path (P_2P_1Vec in Figure 2.15), is decided by the following algorithm

Algorithm D.2.1: ROTATE($landCourse, baseLine$)

θ_1 = Angle of landCourse vector in ENU frame
θ_2 = Angle of baseLine vector in ENU frame
if ($\theta_2 > \theta_1$ **and** $\theta_2 - \theta_1 < 180$) **or** ($\theta_2 < \theta_1$ **and** $\theta_1 - \theta_2 > 180$)
  then Rotate vector clockwise

  else Rotate vector anti-clockwise

Here, landCourse describes the inverse of the desired course during landing, and baseLine is a vector pointing from the landing target towards the UAV.

D.3 Landing initiated close to LT

In the event that the UAV initiates landing close to the LT (closer than the second CP), and the angle is small, there is a failsafe in creating the first CP, to make sure the UAV flies in a path that will give it the best chance possible to reach its LT. Figure D.1 shows where the checkpoints would be created if not for this failsafe

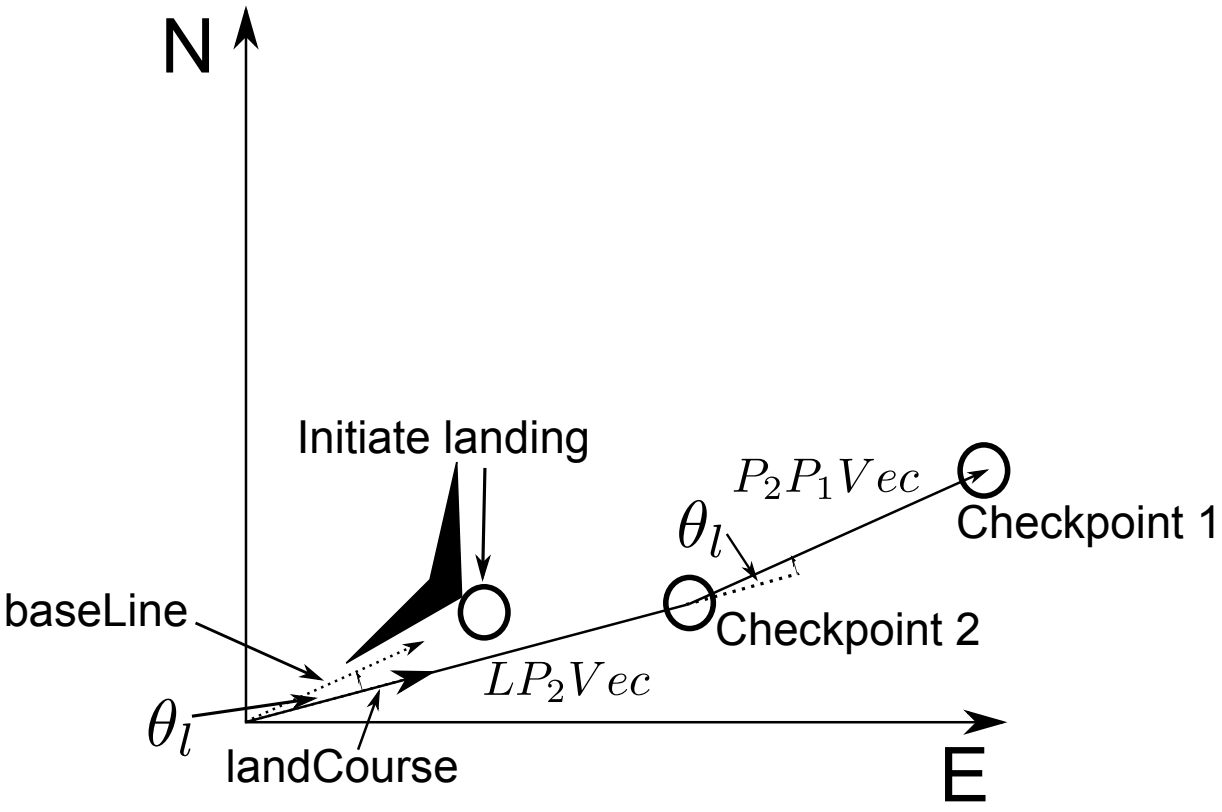


Figure D.1: Checkpoint position if UAV is too close and angle small

CP 1 would now be created at the correct angle, however, as the UAV passes CP 1, it has to perform a 180° turn. Which, depending on the distances between checkpoints and LT, might not be enough to get to the desired path, if there are disturbances. Therefore, in the same scenario, creating CP 1 at a 45° angle, will give the UAV a smaller angle to turn, and a better chance to follow the desired path.

Appendix E

HIL landing targets

E.1 Blender

Blender is a software used to create the 3D models which are used as landing targets for the HIL-simulation. The objects are editable, with the option to customize both size and height over ground, which results in a realistic landing target in the HIL simulation.

Additionally, there is the option of giving an object physical properties, such as gravity or weight. With this, the objects can act solid when stationary, which allows us to simulate a 'crash' with a landing target (a net for example). In Figure E.1, a typical development environment in Blender is seen. A more detailed description of Blender and its usages can be found in [6].

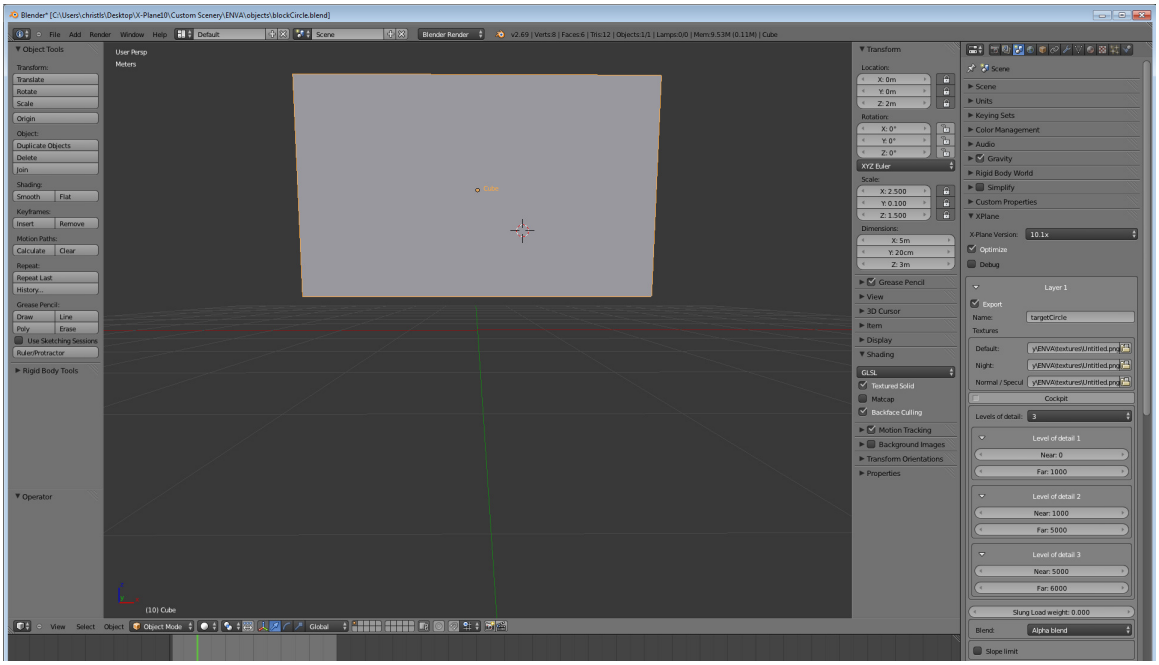


Figure E.1: Development environment in Blender

E.2 OverlayEditor

This program allows for the placement of objects created in Blender (described in Section E.1). OverlayEditor has a pre-cached map of the airports around the world, with geodetic ECEF coordinated combined with the airports, giving the possibility of placing an object at a specific location (for example at an airport). The airport used for the HIL testing in this project was Værnes airport (ENVA airport code in OverlayEditor). This program was then used to place objects at designated positions with a highly accurate GPS position. A quick snapshot of the OverlayEditor user interface can be seen in Figure E.2. When the objects are placed, OverlayEditor stores the changes as custom scenery files, which are readable from XPlane (described in Section 2.5.1).

Figure E.2: Snapshot of Overlayeditor at Værnes airport, with objects placed

Appendix F

SNR

This appendix shows the difference in SNR between a fixed RTK-GPS solution and a float RTK-GPS solution.

F.1 SNR with fix solution

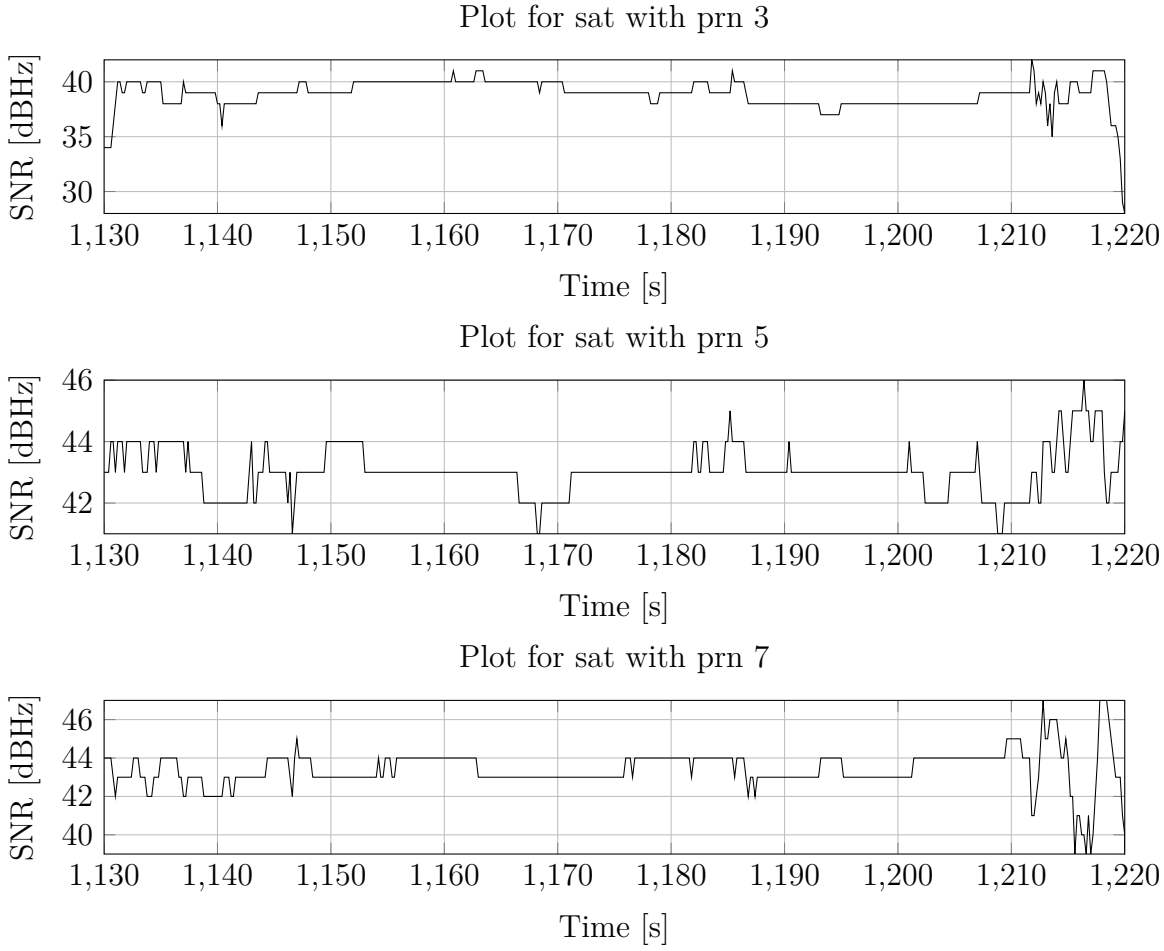


Figure F.1: Plot of 3 valid satellites with fix solution

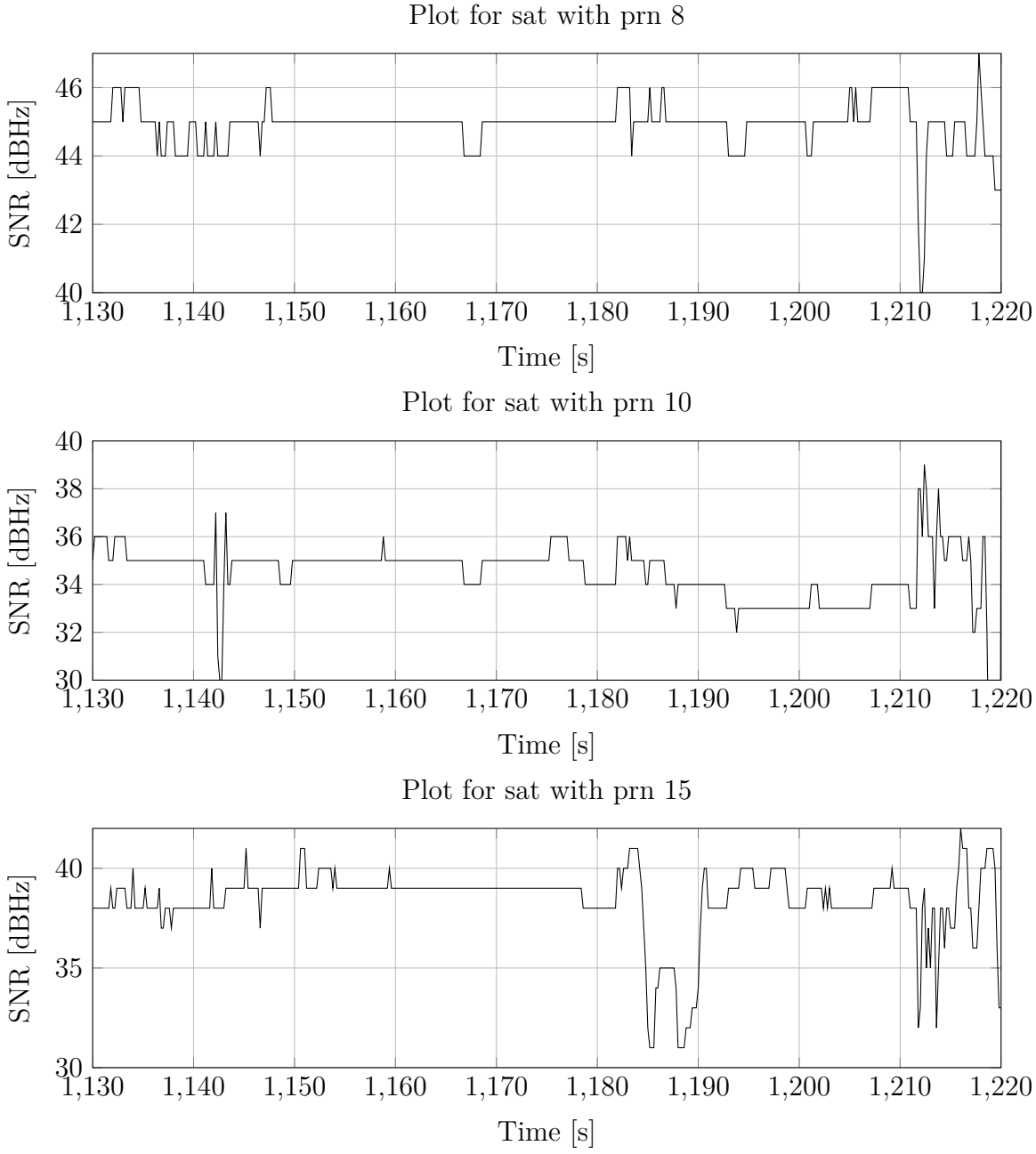


Figure F.2: Plot of 3 valid satellites with fix solution

F.2 SNR with float solution

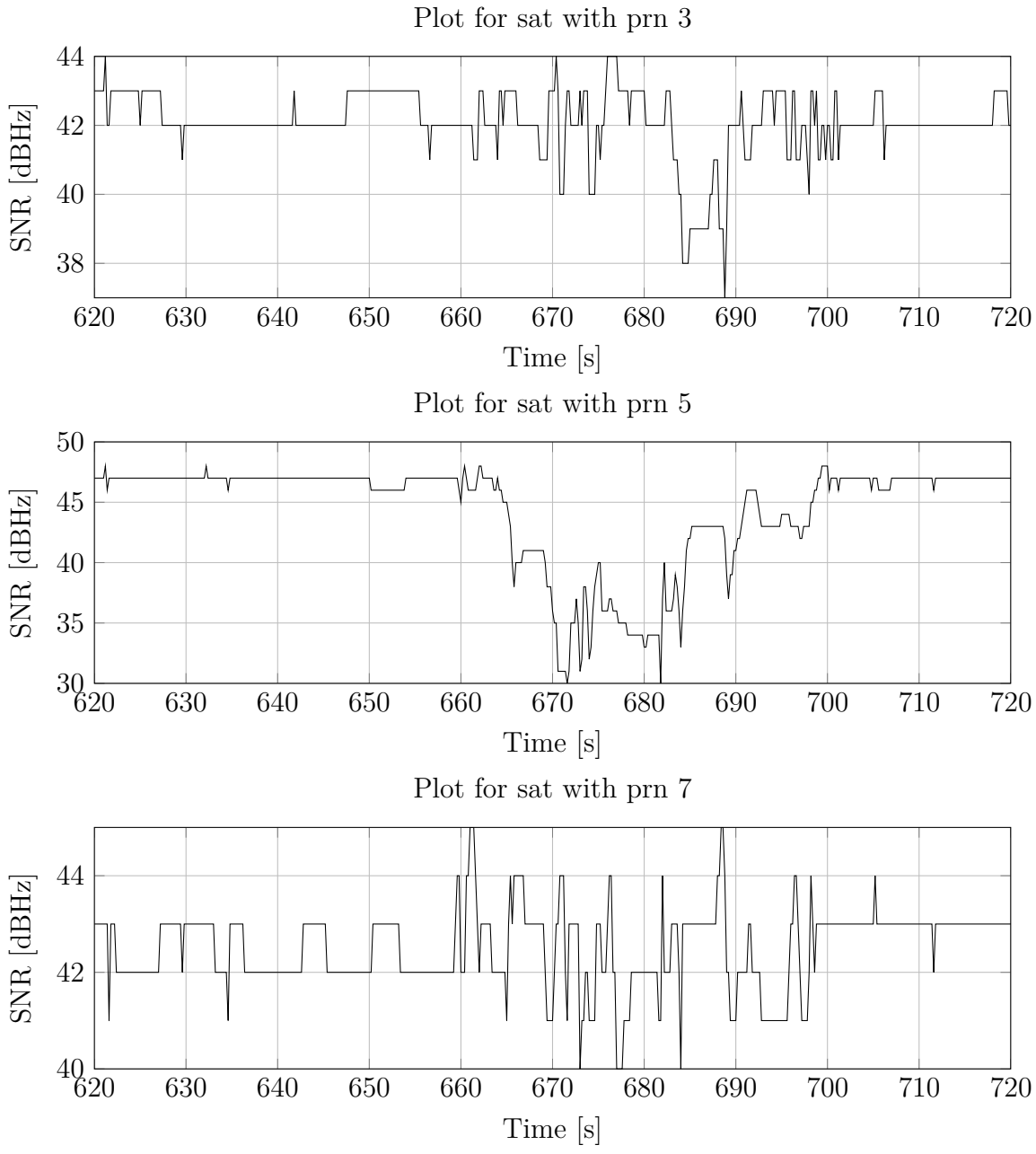


Figure F.3: Plot of 3 valid satellites with float solution

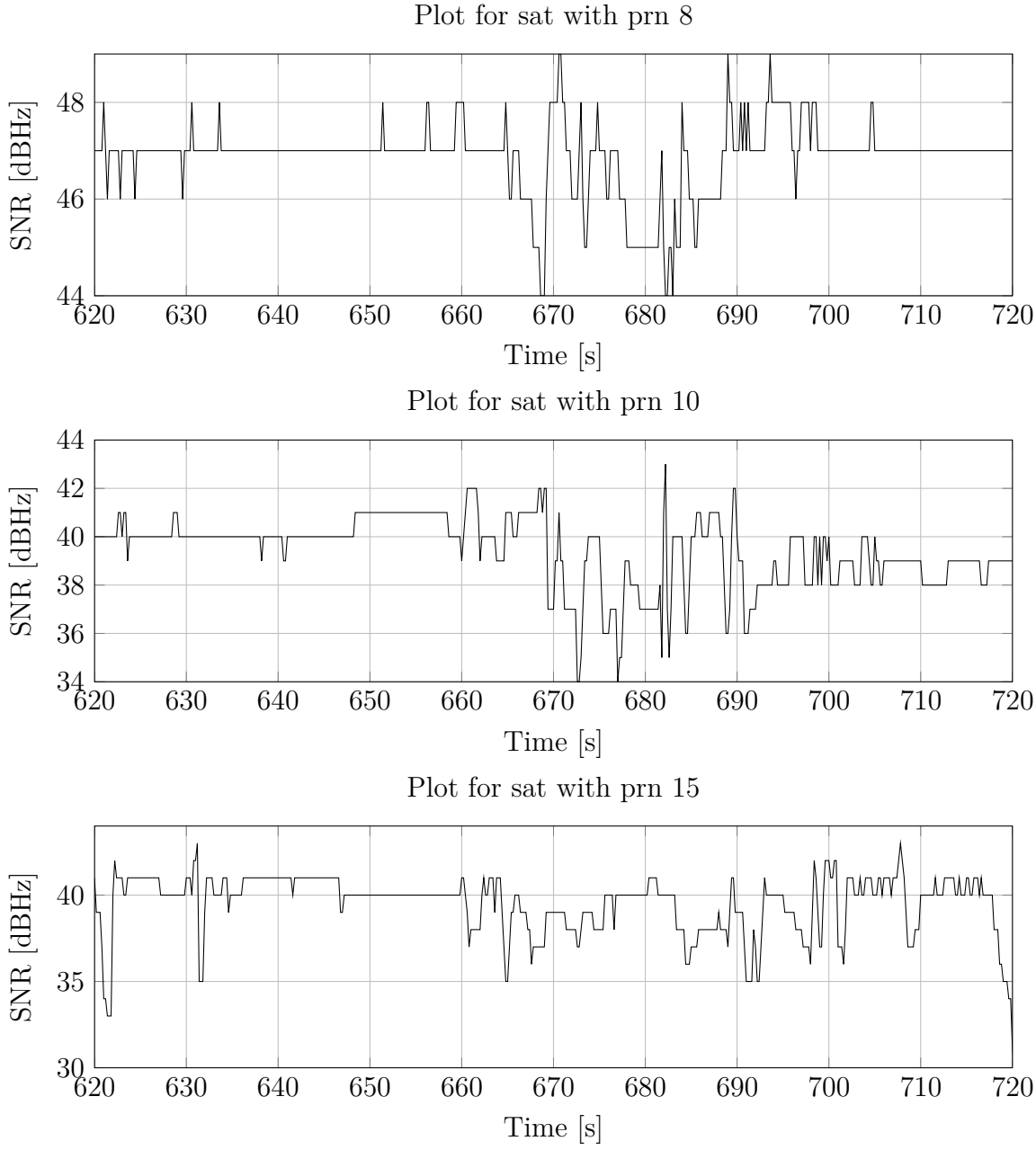


Figure F.4: Plot of 3 valid satellites with float solution