NTNU – Trondheim
Norwegian University of
Science and Technology

AUTONOMOUS LANDING OF FIXED WING UAV IN A STATIONARY NET PATH AND NAVIGATION SYSTEM

Kjetil Hope Sørbø

Submission date: June 2016
Responsible professor: Thor Inge Fossen, Tor Arne Johansen

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Title: AUTONOMOUS LANDING OF FIXED WING UAV IN A STATIO
PATH AND NAVIGATION SYSTEM

Student: Kjetil Hope Sørbø

Problem description:

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

This is the second paragraph. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Responsible professor: Thor Inge Fossen, Tor Arne Johansen

Supervisor:

Abstract

This thesis present a path and navigation system, which is used in a autonomous net landing system for a fixed wing Unmanned Aerial Vehicle (UAV). A landing path of a UAV can be constructed as a straight line path, however in order for a UAV to follow the landing path it must be in a position from which it has a feasible path to the start position of the landing path. This is not a guaranty during a UAV operation, which motivates the development of a approach path towards the landing path with the guaranty that the UAV has a feasible path to the start position of the landing path from any initial start position.

In addition to a path generation system the UAV require a robust high accurate navigation system. This is accomplished by applying Real Time Kinematic GNSS (RTK-GNSS), which can provide centimeter level position accuracy. However due to the RTK-GNSS system can lose its lock on satellites, a secondary Global Navigation Satellite System (GNSS) system is used from a external navigation system. To handle a RTK-GNSS drop out a robust RTK-GNSS system is proposed, where previous valid RTK-GNSS position solutions are fused together with the secondary GNSS system, to be used as a compensator for the external navigation system. The compensator is designed to enable the external navigation system to achieve the same position accuracy level as the RTK-GNSS system for a short duration, until the RTK-GNSS is either reconnected or completely disconnected. With the compensator the UAV navigation system becomes robust against short drop out of the RTK-GNSS, and the availability of the RTK-GNSS is prolonged.

A mobile sensor unit was created to be used as a position reference for net placement, which eased the operational execution of a autonomous landing system. The sensor unit was used then the autonomous landing system was tested in the field at Agdenes airfield. The performance of the autonomous landing system showed that the system is capable of performing a autonomous landing.

Preface

Contents

Acronyms **vii**

1 Introduction **1**
1.1 Background and motivation 1
1.2 Previous work 1
1.3 Scope 3
1.4 Contributions 4
1.5 Outline 5

2 Basis and modelling **7**
2.1 UAV model 7
2.2 Landing path modelling 8
2.2.1 Straight lines 9
2.2.2 Dubins path 10

3 Path and navigation **15**
3.1 Landing plan 15
3.1.1 Landing Path 15
3.1.2 Approach path 17
3.2 Navigation system 21
3.2.1 Position estimation RTK-GPS 22
3.2.2 Error sources 23
3.2.3 Navigation state control system 24
3.3 Summary 30

4 Applied software and hardware **31**
4.1 LSTS toolchain 31
4.1.1 IMC 31
4.1.2 Dune 32
4.1.3 Neptus 32
4.1.4 Glued 33
4.2 RTKLIB 33

4.3 Pixhawk . . . 35
4.4 Ardupilot . . . 35
4.5 JSBsim . . . 36
4.6 X8 and nest payload . . . 36

5 Implementation **39**
5.1 Navigation system . . . 40
5.1.1 RTK-GPS system . . . 41
5.1.2 Navigation state control system . . . 42
5.1.3 Mobile sensor unit . . . 44
5.1.4 Navigation source monitor . . . 44
5.2 Landing plan generator . . . 45
5.2.1 Landing plan generation API . . . 45
5.2.2 Approach path . . . 47
5.2.3 Landing path . . . 49
5.3 Software in the loop simulation . . . 50
5.3.1 Outline of testing . . . 50
5.3.2 Landing plan generation . . . 51
5.3.3 Spiral path creation . . . 53
5.3.4 Result of simulations . . . 55
5.4 Summary . . . 56

6 Experimental field tests **59**
6.1 Outline of testing . . . 59
6.2 Landing plan generation system . . . 60
6.2.1 Day 1 . . . 60
6.2.2 Day 2 . . . 69
6.3 Navigation . . . 76
6.3.1 RTK-GNSS performance . . . 76
6.3.2 Short loss compensator . . . 77
6.3.3 Summary of navigation system performance . . . 79
6.4 Summary . . . 80

7 Conclusion and recommendation for further work **83**
7.1 Conclusion . . . 83
7.2 Recommendation for further work . . . 85

References **87**

Appendices

A Landing plan generation API **91**

B Guidance and control system **93**

B.1 Lateral controller 93
B.2 Longitudinal controller 94

C Landing plan parameters **97**
C.1 Path parameter used in the SIL simulation 97
C.2 Start path parameter used the first flight day 98
C.3 Start path parameter used the second flight day 99

D Addition field experiment results **101**

E Rtklib Configuration **103**

Acronyms

API Application Programming Interface.

ENU East North Up.

GNSS Global Navigation Satellite System.

GPS Global Positioning System.

IMC Inter-Module Communication.

MAV Miniature air vehicle.

NED North East Down.

RTK-GNSS Real Time Kinematic GNSS.

RTK-GPS Real Time Kinematic GPS.

RTKLIB Real-Time Kinematic Library.

SIL Software In the Loop.

UAS Unmanned aircraft system.

UAV Unmanned Aerial Vehicle.

Chapter 1

Introduction

1.1 Background and motivation

Recent development of flying UAVs has been recognized to provide an attractive alternative to work previously performed by manned operations. Typical work which has attracted attention includes inspection, aerial photography, environmental surveillance and search and rescue. Today UAV operation are becoming more autonomous, however in order to become fully autonomous a fixed wing UAV must be able to perform a autonomous landing.

An important premise for successful and safe UAV operation, is the provision of a robust system for safe landing of the UAV following completed operations. A autonomous landing system require a path generation system that can create a flyable landing path during flight operation from any initial position. In addition the navigation system must have centimeter level accuracy in order for the UAV to perform a autonomous landing in a net. However a accurate navigation system must be able to handle position accuracy degeneration, in order to prevent system failure. Together with a accurate navigation system and path generation system the placement of the net must be known, and made available to the UAV. With a known position of the landing net the UAV must gracefully perform a decent, preferable a glide slope towards the landing net position, with a landing path length specified by the operator.

1.2 Previous work

There has been perform several studies on autonomous landing system, and there currently exist commercial available systems. However these are typical expensive, and mostly focused on either military or air traffic industry. An available system for UAVs is the SkyHook that apply Instrumental Navigation System (INS) with

GNSS[Insitu], however this system require expensive equipment and is limited to a few UAV systems. The limitation on type of UAV and high cost restricts the usage of the recovery system, and motivates the research of a low cost recovery system for fixed wing UAV.

Studies that has been performed on autonomous landing has mostly focused on vision-based guidance, due to previously limited accuracy in low-cost GNSS receiver system, which is typically single frequency receivers. In the paper [Barber et al., 2007] a landing system was proposed, which compared the use of barometric pressure measurement and optic-flow measurement for estimation of height above ground. The landing path composed of a spiral path down to a given altitude where a glide slope were used to guide the MAV down to the landing area. The papers showed that optic-flow measurement reduced the average landing error with several meters, however the technique used to guided the UAV is not suitable for precision landing due to large average error from the landing target. A low cost recovery system for fixed wing UAV is proposed in the paper [Kim et al., 2013], where computer vision is used to find and identify the recovery net. The system was successful in performing an autonomous landing, however it require that the visual image is sent from the UAV to the ground station. In addition the system require a clear image in order to calculate the guidance commands for the UAV, which restricts when the system can used. In the paper [Huh and Shim, 2010] a vision-based landing system is presented, which was successful in performing an automatic landing. The system was aided by a standard IMU and GPS, together with a vision system relaying on color and moments based detection. The system is sensible to lighting condition, however an filtering rule was used to find the landing area. The sensibility to lighting condition is a disadvantage with vision-based guidance system, and therefore it's preferable to create an high accurate positioning system.

A net recovery system for UAV with single-frequency Real Time Kinematic GPS (RTK-GPS) was described in the paper [Skulstad et al., 2015], which was a result of the work done in the master thesis [Skulstad and Syversen, 2014]. The system presented applied RTKlib together with low-cost single frequency Global Positioning System (GPS) receivers as navigation system with a customized Ardupilot software. The complete system was able to perform a net landing, however the result showed that further work would require better controllers, and a robust navigation system. An continuation of the work done in [Skulstad and Syversen, 2014] was done in [Frølich, 2015]. The work simulated an autonomous net landing, however no physical experiment was perform. The work done in [Frølich, 2015] moved the autonomous landing system into the DUNE runtime environment, however the autonomous landing system created cannot be used in the field due to the inability of setting spacial restrains on the landing path.

The work done in the master thesis [Spockeli, 2015] resulted in the implementation of a RTK-GNSS system into the DUNE runtime environment, however the RTK-GNSS system was not integrated with the UAV navigation system. Other work done at the UAVlab with the goal of creating a autonomous landing system for fixed wing UAV was performed in the master thesis [Gryte, 2015], where a mathematical model of the skywalker X8 fixed wing UAV was created. However since the dynamical parameter for the X8 has yet to be determined, deviation in behaviour between simulation and physical flights are expected.

1.3 Scope

The objective of this work is to design, implement and test a autonomous landing system for fixed wing UAV. The focus area for this work has been the design and implementation of a landing plan, and a high accurate navigation system. The landing plan generator is an improved version of the landing path used in [Skulstad and Syversen, 2014] moved into the DUNE environment, combined with the landing plan generator interface created in the master thesis [Frølich, 2015]. The navigation system continues the work started in the master thesis [Spockeli, 2015], where RTK-GNSS navigation data from RTKlib was made available to the DUNE environment.

This thesis propose a path generation system capable of creating a flyable landing plan from any initial position with the guaranty that the UAV will be able to enter the landing path at the correct height with the correct orientation. The landing plan is composed of a landing path and a approach path where the latter is used to ensure the creation of a flyable path for the UAV to the start position of the landing path. In addition is a strategy for handling RTK-GNSS drop out presented, which includes fusing previous valid RTK-GNSS position solutions together with a external navigation system, to be used as a compensator for the external navigation system during a short duration after an RTK-GNSS drop out. The compensator is implemented in the DUNE runtime environment, thus avoiding alteration in the RTK-GNSS system software and ensures that the UAVs robust RTK-GNSS navigation system is independent from the RTK-GNSS system software. The control system tested in this autonomous landing system has been proposed as part of the master thesis [Nevstad, 2016], which has been tested both in a simulator and in the field. The simulator used to verify the autonomous landing system is based on the mathematical model created in the master thesis [Gryte, 2015]. For field test a operation study on the execution of a autonomous landing at Agdenes has been performed, in addition to the creation of a mobile sensor unit with RTK-GNSS to be used as an reference position for stationary net placement.

1.4 Contributions

The contributions of this thesis are a landing plan generator, a navigation system able to provide high accurate position and velocity information, a redundancy strategy for short loss of RTK-GNSS, a mobile sensor unit used as a GPS reference location for net placement and a experimental testing and operational study of the autonomous landing system, summarised as follows:

1. **A landing plan generator** has been created and implemented in DUNE, which guaranty a flyable path from any initial position to a landing zone, with a specified decent angle. In addition to the implementation of the landing plan an **Application Programming Interface (API)** has been created to generate a landing plan.

2. **A navigation state control system** has been created to manage which positioning system should be used in the DUNE environment. The state machine is designed to switch between the position and velocity information provided by the external navigation system and the RTK-GNSS system. In addition a **robust RTK-GNSS system** has been designed and implemented in DUNE, which includes fusing previous valid RTK-GNSS position solutions together with a external navigation system, to be used as a compensator for the external navigation system during a short duration after an RTK-GNSS drop out. With the compensator the UAV navigation system is able to keep a high accurate positioning system, for a short duration after a RTK-GNSS drop out has occurred.

3. **A navigation source monitor** has been created in the command and control software, Neptus, to provide visual indication on the source of the navigation data used in the DUNE environment, in order to give feedback to the operator on the state of the UAV navigation system. The navigation source monitor based on a color code, in order for the operator to quickly notice alteration in the state of the navigation system.

4. **A mobile sensor with RTK-GNSS** has been created, to be used as an reference position for stationary net placement. The mobile sensor unit allows for accurate position solution of the net placement, which is critical for a autonomous stationary net landing system.

5. **Experimental testing** of the navigation system and landing path generator in the field. The autonomous landing system was tested on Agdenes airfield with a virtual net placed $25m$ above the runway. Test result gathered from the field test has been used in an **operational study** on performance and feasibility of a autonomous landing at Agdenes airfield.

1.5 Outline

Chapter 2 outlines two path planing strategies which is used in the development of the landing path system. The chapter also contains a model of an Miniature air vehicle (MAV).

Chapter 3 presents a path and navigation system for a autonomous landing system. The path system consist of the creation of a landing plan, which contain a landing path and an approach path. The landing path is created as s straight line path relative to the net position and orientation, while the approach path is defined relative to both the landing path and the initial position of the UAV. The approach path is created as a Dubins path in the lateral plane, and as a straight line path in the longitudinal plane. The navigation system presented consist of a navigation state control system used to integrate a robust RTK-GNSS system into the UAV navigation system. The robust RTK-GNSS consist of the RTK-GNSS solution from RTKlib and a short RTK-GNSS loss compensator, which fuse valid RTK-GNSS position with a external navigation system together to form a compensator that allows the UAV navigation system to keep a high accurate position solution after a RTK-GNSS drop out for a short duration.

Chapter 3 proposes a path and navigation system for a autonomous landing system. The landing path system is separated into two planes, which is the lateral and longitudinal plane. The lateral path is created as a Dubins path, and the longitudinal path is created as a straight line path. The navigation system is controlled by a state machine which controls the source of the positioning and velocity solution, in addition to increasing the robustness of the RTK-GNSS.

Chapter 4 outlines the software used to create and test the autonomous landing system as well as the hardware configuration used as a basis for the X8 fixed wing UAV.

Chapter 5 outlines the implementation details of the path and navigation system, including simulation verification of the system. In addition the mobile sensor unit and navigation source monitor are presented.

Chapter 6 present experimental testing of the path and navigation system in the field, with the results used in an operation study of a autonomous landing operation at Agdenes.

Chapter 7 present the closing discussion with conclusion and recommendation for further work.

Chapter 2

Basis and modelling

2.1 UAV model

A fixed wing UAV model is presented in [Beard and McLain, 2012], which contain the kinetic equations of a general MAV. The kinetic equations used to described a general MAV can represent a fixed wing UAV given that the size of the UAV is small enough, which is the case with the UAV used in this thesis. First the kinematic equations used to describe a MAV are given as:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \mathbf{R}(\mathbf{\Theta})_{Body}^{NED} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{2.1a}$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \mathbf{T}(\mathbf{\Theta}_{nb}) \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{2.1b}$$

where $\mathbf{R}(\mathbf{\Theta})_{Body}^{NED}$ is the rotation matrix from the body frame to the NED frame, with $\mathbf{\Theta} = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T$. The transformation matrix $\mathbf{T}(\mathbf{\Theta}_{nb})$ is given in [Fossen, 2011] as:

$$\mathbf{T}(\mathbf{\Theta}_{nb}) = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \frac{\sin(\phi)}{\cos(\theta)} & \frac{\cos(\phi)}{\cos(\theta)} \end{bmatrix} \tag{2.2}$$

The kinetic equations presented from [Beard and McLain, 2012] is given in the body frame, which is a frame fixed to the body of the vehicle and rotated relative to an

inertia frame, e.q the Earth center. The kinetic equations are given as:

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = \mathbf{R}(\mathbf{\Theta})_{NED}^{Body} \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} - \frac{1}{2}\rho V_a^2 S \mathbf{R}(\alpha)_{Stability}^{Body} \begin{bmatrix} F_{Drag} \\ 0 \\ F_{Lift} \end{bmatrix} \tag{2.3a}$$
$$+ \frac{1}{2}\rho V_a^2 S \begin{bmatrix} 0 \\ C_y(\beta, p, r, \delta_a, \delta_r) \\ 0 \end{bmatrix} + \frac{1}{2}\rho S_{Prop} C_{Prop} \begin{bmatrix} (K_{Motor}\delta_t)^2 - V_a^2 \\ 0 \\ 0 \end{bmatrix}$$
$$\begin{bmatrix} L \\ M \\ N \end{bmatrix} = \frac{1}{2}\rho V_a^2 S \begin{bmatrix} C_L(\beta, p, r, \delta_a, \delta_r) \\ C_M(\alpha, q, \delta_e) \\ C_N(\beta, p, r, \delta_a, \delta_r) \end{bmatrix} + \begin{bmatrix} -k_{T_p}(K_\Omega \delta_t)^2 \\ 0 \\ 0 \end{bmatrix} \tag{2.3b}$$

where ρ is the air density in kg/m^3,mg is the weight of the MAV, S is the platform area of the MAV wing, C_i are nondimensional aerodynamic coefficients and V_a is the speed of the MAV through the surrounding air. α and β is the attack and side slip angle respectfully. F_{Drag} is the drag force acting on the fuselage, and F_{Lift} is the lift force. $\mathbf{R}(\alpha)_{Stability}^{Body}$ are the rotation matrix from the stability frame to the body frame. The stability frame is orientated with respect to the MAV movement through the surrounding air, which is defined as a standard rotation around the y-axis of the body frame. S_{Prop} is the area swept out by the propeller, and K_{Motor},K_{T_p} and K_Ω are propeller specific constants. The control surface on the MAV is defined into two groups; the wings and the rudder. On the rudder δ_e controls the elevator deflection and δ_r the rudder deflection. For the wings δ_a is the control input from the aileron deflection, while δ_t is the throttle control input.

2.2 Landing path modelling

A landing path can be view as a path following problem, which is the UAV attempts to follow a desired path without a time demand. A minimum requirement for a path is that the path is connected, where the connection level can be described by the path smoothness. Smoothness can be described with parametric continuity, which is denoted C^n were n is the degree of smoothness. The order of n implies that the n first parametric derivatives match at a common point for two subsequent paths [Barsky and DeRose, 1989]. Geometric continuity is a relaxed form of parametric continuity in which discontinuousness in speed is allowed. A table 2.1 of geometric and parametric continuity lists the requirement for each smoothness level, which is based definitions presented in [Barsky and DeRose, 1989]. Geometric continuity is sufficient for a path following system, which is the main focus of this thesis. Geometric continuity is denoted as G^n were n is the order of continuity.

Geometrical smoothness level	**Description**
G^0	All subpaths are connected
G^1	The path-tangential angle is continuous
G^2	The center of curvature is continuous
Parametric smoothness level	**Description**
C^0	All subpaths are connected
C^1	The velocity is continuous
C^2	The acceleration is continuous

Table 2.1: Smoothness definitions for both parametric and geometric

The definition used for path in this thesis is equation 1.2 in [Tsourdos et al., 2010] which state:

$$P_s(x_s, y_s, z_s, \theta_s, \psi_s) \xrightarrow{r(\varpi)} P_f(x_f, y_f, z_f, \theta_f, \psi_f) \tag{2.4}$$

where the subscripts s and f denotes the start pose and finish pose respectfully with $r(\varpi)$ as the path and ϖ the path variable.

2.2.1 Straight lines

The simplest path from P_s and P_f is a straight line path between the poses. A straight line path is given, where for for simplicity and without loss of generality the path is reduced to a 2 dimensional case:

$$x(\varpi) = a_x\varpi + b_x \tag{2.5a}$$
$$y(\varpi) = a_y\varpi + b_y \tag{2.5b}$$

with $\varpi \in [0, 1]$, where ϖ has not necessary a physical meaning. Then the parametrisation of the straight line becomes:

$$P(0) = \begin{bmatrix} x(0) \\ y(0) \end{bmatrix} = \begin{bmatrix} b_x \\ b_y \end{bmatrix} = \begin{bmatrix} x_s \\ y_s \end{bmatrix} \tag{2.6a}$$
$$P(1) = \begin{bmatrix} x(1) \\ y(1) \end{bmatrix} = \begin{bmatrix} a_x + b_x \\ a_y + b_y \end{bmatrix} = \begin{bmatrix} x_f \\ y_f \end{bmatrix} \rightarrow \begin{bmatrix} a_x \\ a_y \end{bmatrix} = \begin{bmatrix} x_f - b_x \\ y_f - b_y \end{bmatrix} \tag{2.6b}$$

Further the path tangential for a straight line path is given as:

$$\psi(\varpi) = \text{atan2}(a_y, a_x) \tag{2.7}$$

which shows that the path-tangential for a straight line path is discontinues, as seen in figure 2.1. That define a straight line path with the smoothness of G^0, which

allows the path to be used in a path following system, however with discontinuity in the path-tangential. A straight line path is shown in figure 2.2.

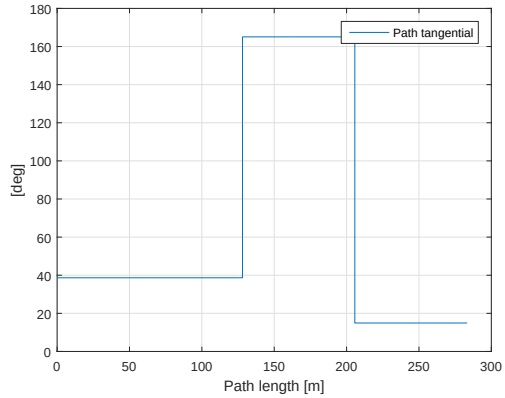

Figure 2.1: Path-tangential to a straight line path

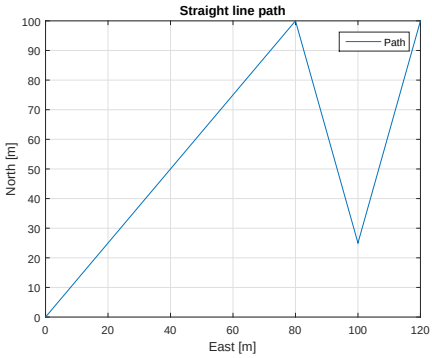

Figure 2.2: Straight line path

2.2.2 Dubins path

An alternative to a straight line path is a path constructed by straight lines and circle, like the Dubins path proposed in the paper [Dubins, 1957], which showed that the shortest possible path for a particle that moved with unit speed with maximum curvature would consist of two circles and a straight line which is tangential to both circles.

A Dubins path which is constructed with the final orientation fixed has four ways to be constructed, which are determined by the rotation directions. The four rotation combination with fixed finish orientation are given in table 2.2.

Right to Right
Right to left
Left to Right
Left to left

Table 2.2: Turning direction for Dubins path with fixed final orientation

The equations used to construct a Dubins path are found in [Tsourdos et al., 2010] section 2.2.1, with a constructed path shown in figure 2.3. In figure 2.3 the whole line is the path, with the doted lines used express the parameter used to construct the path.

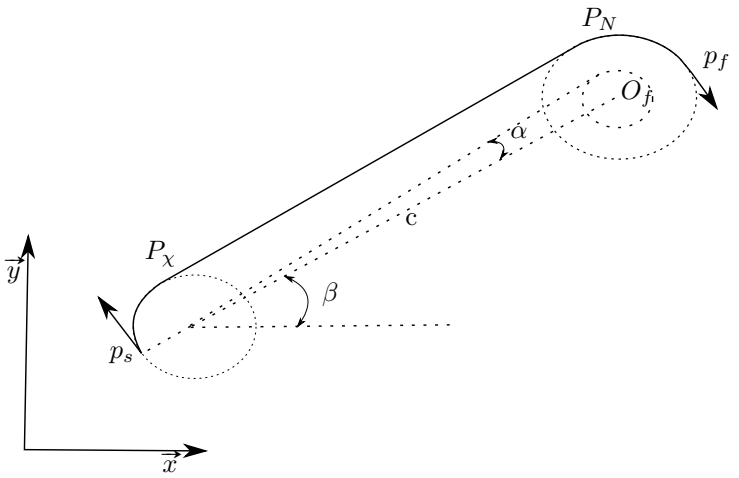

Figure 2.3: The whole line is the Dubins path, while the doted lines are used to express the parameter used to construct the path

Dubins path is constructed by first determine the start and finish turning circle

center. The centres are found with the equations:

$$X_{cs} = X_s - R_s \cos(\psi_s \pm \frac{\pi}{2}) \tag{2.8a}$$

$$Y_{cs} = Y_s - R_s \sin(\psi_s \pm \frac{\pi}{2}) \tag{2.8b}$$

$$X_{cf} = X_f - R_f \cos(\psi_f \pm \frac{\pi}{2}) \tag{2.8c}$$

$$Y_{cf} = Y_f - R_f \sin(\psi_f \pm \frac{\pi}{2}) \tag{2.8d}$$

where R_s and R_f is the radius of the start and final turning circle respectfully, with ψ_s and ψ_f the start and finish orientation. The centres for the start and finish turning circle are defined as:

$$\mathbf{O}_{cs} = \begin{bmatrix} X_{cs} \\ Y_{cs} \end{bmatrix} \tag{2.9}$$

$$\mathbf{O}_{cf} = \begin{bmatrix} X_{cf} \\ Y_{cf} \end{bmatrix} \tag{2.10}$$

Continuing the centres O_{cs} and O_{cf} are connected with a centreline c, where the length is given as:

$$|c| = ||\mathbf{O}_{cs} - \mathbf{O}_{cf}||_2 \tag{2.11}$$

where $||\cdot||_2$ is the second norm. Continuing the arc exit and entry point for the start and finish circles are calculated by first applying the equations:

$$\alpha = \arcsin\left(\frac{R_f - R_s}{|c|}\right) \tag{2.12a}$$

$$\beta = \arctan\left(\frac{Y_{cf} - Y_{cs}}{X_{cf} - X_{cs}}\right) \tag{2.12b}$$

where α is the angle between the length of the center line between the two circles, and the length of the line from the start circle to the exit tangent point. β is the angle of the center line with respect to the inertial frame. The exit and entry tangent point is found with the use of table 2.3.

	Turn angle
ϕ_{right}	$\alpha + \beta + \frac{\pi}{2}$
ϕ_{left}	$\beta - \alpha + \frac{3\pi}{2}$

Table 2.3: Turn angle

With the angle of the exit and entry tangent point the points are given as:

$$x_{P_\chi} = x_{cs} + R_s \cos(\phi) \tag{2.13a}$$
$$y_{P_\chi} = x_{cs} + R_s \sin(\phi) \tag{2.13b}$$
$$x_{P_N} = x_{cf} + R_f \cos(\phi) \tag{2.13c}$$
$$y_{P_N} = x_{cf} + R_f \sin(\phi) \tag{2.13d}$$

which is used to define the exit and entry points as:

$$\mathbf{P}_\chi = \begin{bmatrix} x_{P_\chi} \\ y_{P_\chi} \end{bmatrix} \tag{2.14a}$$
$$\mathbf{P}_N = \begin{bmatrix} x_{P_N} \\ y_{P_N} \end{bmatrix} \tag{2.14b}$$

The length of the Dubins path is calculated in three parts. The first the is the arc length from the start pose to the exit tangent point, then the length of the straight line before the arc length from the entry point to the finish pose. The length of the path is given as:

$$d = R_s\phi_s + d_t + R_f\phi_f \tag{2.15}$$

where $d_t = ||\mathbf{P}_N - \mathbf{P}_\chi||_2$, ϕ_s and ϕ_f is the arc angle for the start and finish circle respectfully. The path-tangential of the Dubins path is given as:

$$\psi = \text{atan2}(1, -tan(\phi)) \tag{2.16}$$

which determine that the path-tangential is continues, which makes Dubins path G^1. The path-tangential for Dubins path is shown in figure 2.4.

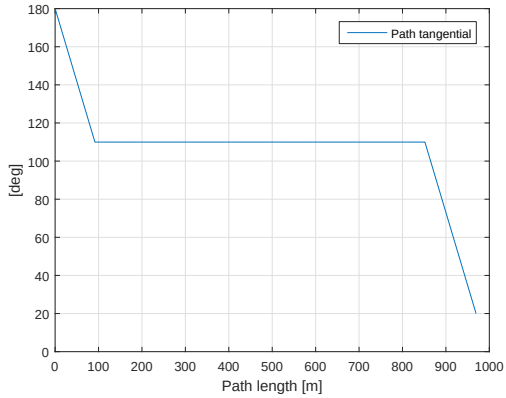

Figure 2.4: Path-tangential for a Dubins path

Chapter 3

Path and navigation

This chapter contain the system description of the landing plan generator and the navigation system.

3.1 Landing plan

The landing plan consist of two main parts, which the landing path and the approach path. The landing path is a straight line path orientated with respect to a reference position, which in this system is the position of the net. The approach path is design as a lateral Dubins path and a longitudinal straight line path, with the guaranty that the UAV is able to enter the landing path at the correct height with the correct orientation.

3.1.1 Landing Path

The landing path is inspired by the work done in [Skulstad and Syversen, 2014] where waypoints was used to create a straight line path towards the net. This method proved successful, and thus this landing system continues on the work. The decent angle of the straight line path should be kept small to avoid build up in speed, however the trade off is that the start high in of the landing path must be above any obstacles that is around the landing area. A waypoint is here defined as:

$$\mathbf{WPn} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{3.1}$$

where $x, y, z \in \mathbb{R}^3$. The straight line path is constructed relative to the net as shown in figure 3.1, with way-points given as:

$$\mathbf{WP4} = \begin{bmatrix} -a0 \\ 0 \\ h_{nc} + a1\tan(\gamma_n) \end{bmatrix} \tag{3.2a}$$

$$\mathbf{WP3} = \begin{bmatrix} a1 \\ 0 \\ h_{nc} - a1\tan(\gamma_n) \end{bmatrix} \tag{3.2b}$$

$$\mathbf{WP2} = \mathbf{WP3} + \begin{bmatrix} a2 \\ 0 \\ -a2\tan(\gamma_l) \end{bmatrix} \tag{3.2c}$$

$$\mathbf{WP1} = \mathbf{WP2} + \begin{bmatrix} a3 \\ 0 \\ 0 \end{bmatrix} \tag{3.2d}$$

where the description of the parameters used is given in table 3.1. The net is placed between the fourth and third way points, in order for the fourth waypoint to be a aiming point for the UAV and avoid transitional behaviour before hitting the net.

Parameter	**Description**
h_{nc}	The height from ground to the net center
$a0$	The distance behind the net
$a1$	The distance in front of the net
$a2$	The length of the glide slope
$a3$	The length of the approach towards the glide slope
γ_n	The net attack angle
γ_l	The landing glide slope angle

Table 3.1: Landing path parameters

Continuing the waypoints are rotated into the NED frame by a rotation around the z-axes.

$$\mathbf{WP}^n = \mathbf{R}(\psi_{net})\mathbf{WP}^b \tag{3.3}$$

were ψ_{net} is the heading of the net, and $\mathbf{R}(\psi_{net})$ is the standard rotation matrix around the z-axis.

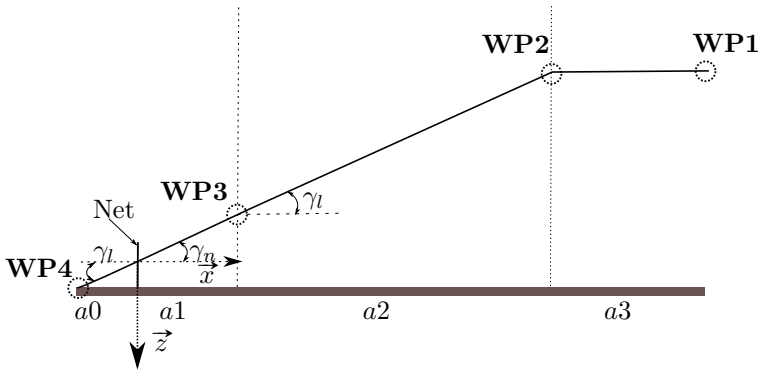

Figure 3.1: The landing path

3.1.2 Approach path

The approach path is separated into two parts, which is a lateral and longitudinal path. The purpose of the path is to ensure that the UAV can enter the landing path at the correct height with the correct orientation from any initial position. The lateral and longitudinal paths are created separately

Lateral path

The lateral path is designed as a Dubins path, with start pose, P_s, which is the pose where the landing plan generation request was made, and final pose, P_f, which is the start position of the landing path with the orientation towards the net. Dubins path was chosen due to its circular turns, simplistic design , and meet the requirement that the UAV enters the landing path with the correct orientation.

The lateral path is constructed with the equations presented in section 2.2.2. The standard approach path is the shortest path of the four different rotation pairs given in table 2.2, however when implemented there exits the option of manually setting the rotation direction for both circles. The shortest path is determined by calculating the length of each variants, where the shortest is chosen. The resulting the path is shown in figure 3.2.

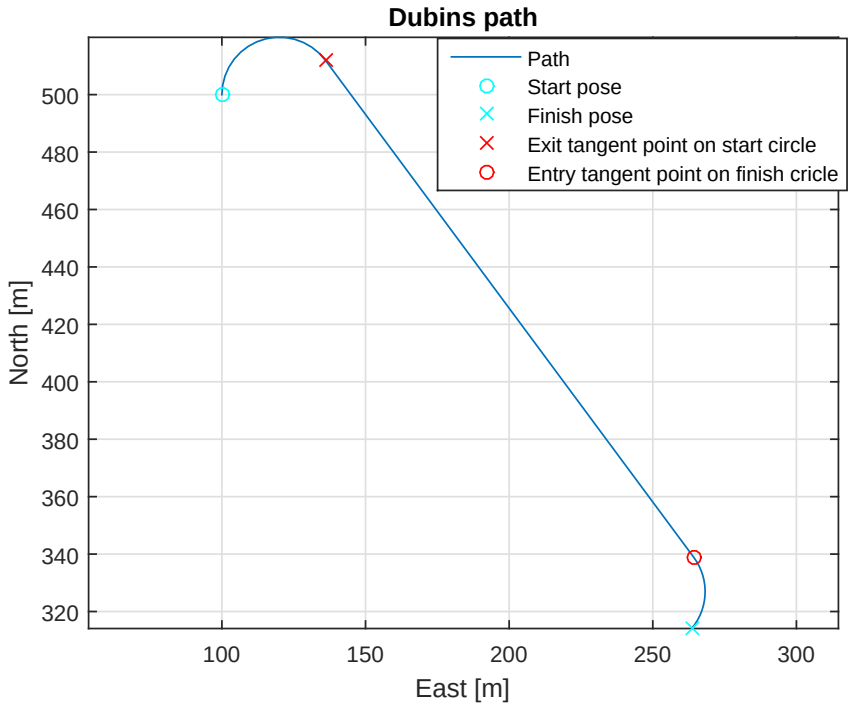

Figure 3.2: Lateral Dubins path

The construction of the lateral path consists of two arc with a straight line between the arcs. The arcs are constructed by first finding the entry and exit angle with respect to the inertia frame, defined as ψ_0 and ψ_1 respectfully:

$$\psi_0 = \begin{cases} \text{atan2}(Y_s - Y_{cs}, X_s - X_{cs}) & \text{if start circle} \\ \text{atan2}(Y_{P_N} - Y_{cf}, X_{P_N} - X_{cf}) & \text{otherwise} \end{cases} \tag{3.4a}$$

$$\psi_1 = \begin{cases} \text{atan2}(Y_{P_\chi} - Y_{cs}, X_{P_\chi} - X_{cs}) & \text{if start circle} \\ \text{atan2}(Y_f - Y_{cf}, X_f - X_{cf}) & \text{otherwise} \end{cases} \tag{3.4b}$$

Continuing the turn angle must be defined, which is the difference between ψ_1 and ψ_0. However the periodic behaviour of the unit circle must be respected, in addition to the rotation direction. The maximum turning angle becomes:

$$\psi_{max} = \begin{cases} -|\psi_1 - \psi_0| & \text{if counter clockwise rotation and } \psi_1 - \psi_0 \leq 0 \\ -(2\pi - |\psi_1 - \psi_0|) & \text{if counter clockwise rotation and } \psi_1 - \psi_0 > 0 \\ |\psi_1 - \psi_0| & \text{if clockwise rotation and } \psi_1 - \psi_0 \geq 0 \\ (2\pi - |\psi_1 - \psi_0|) & \text{if clockwise rotation and } \psi_1 - \psi_0 < 0 \end{cases} \tag{3.5}$$

where $|\psi_1 - \psi_0| \in (-\pi, \pi]$. From the maximum turning angle the angle step and number of angle segments in the arc can be determined:

$$h = \text{sign}\frac{d_{arc}}{R} \tag{3.6a}$$

$$N = \left\lceil \frac{\text{sign}(\psi_{max})\psi_{max}}{|h|} \right\rceil + 1 \tag{3.6b}$$

where h is arc angle step and N the total number of steps in the arc. The step angle must have the same sign as ψ_{max} to ensure the correct rotation direction. Continuing the heading function $\psi(\varpi)$ can be defined as:

$$\psi(\varpi) = \begin{cases} \psi_{max} & \varpi = N - 1 \\ \varpi h & \text{otherwise} \end{cases} \tag{3.7}$$

where $\varpi = 1, ..., N - 1$. Finally the arc path can be defined as:

$$\mathbf{p}(\varpi) = \begin{bmatrix} \mathbf{O}_c \end{bmatrix} + R \begin{bmatrix} \cos(\psi_0 + \psi(\varpi)) \\ \sin(\psi_0 + \psi(\varpi)) \end{bmatrix} \tag{3.8}$$

A summary of the lateral path is:

$$\mathbf{p}(i) = \begin{cases} \begin{bmatrix} \mathbf{O}_{cs} \end{bmatrix} + R_s \begin{bmatrix} \cos(\psi_0 + \psi(\varpi)) \\ \sin(\psi_0 + \psi(\varpi)) \end{bmatrix} & \text{Start circle} \\ \begin{bmatrix} \mathbf{O}_{cf} \end{bmatrix} + R_f \begin{bmatrix} \cos(\psi_0 + \psi(\varpi)) \\ \sin(\psi_0 + \psi(\varpi)) \end{bmatrix} & \text{Finish circle} \end{cases} \tag{3.9}$$

where $i \in [1, ..., N_s + N_f]$ with N_s and N_f as the number of segments in the start and finish circle respectfully.

Longitudinal path

The longitudinal path is designed as a straight line path along the lateral path, which fused together with the lateral path forms a spiral path towards the landing path. The approach path hold a constant decent angle until the correct height is reached, which is defined as the start height for the landing path. The approach decent angle is then adjusted in order for the correct height to be reach. The approach decent angle is defined as:

$$\gamma_d = \begin{cases} \text{atan2}(\Delta z, ||\mathbf{p}(i+1) - \mathbf{p}(i)||_2) & \text{if atan2}(\Delta z, ||\mathbf{p}(i+1) - \mathbf{p}(i)||_2) \leq \gamma_{d_{Max}} \\ \gamma_{d_{Max}} & \text{otherwise} \end{cases} \tag{3.10}$$

where $\gamma_{d_{Max}}$ is the maximum decent angle for the approach path, and Δz is defined as:

$$\Delta z = z_d - z(i) \tag{3.11}$$

where z_d is the z component in $WP1$. Finally the longitudinal path is included into the approach path, which is given as:

$$\mathbf{r}(i) = \begin{bmatrix} \mathbf{p}(i) \\ ||\mathbf{p}(i) - \mathbf{p}(i-1)||_2 \tan(\gamma_d) \end{bmatrix} \tag{3.12}$$

where $\mathbf{r}(i)$ is the landing path, with $i \in [2, ..., N_s + N_f]$ and $p = \begin{bmatrix} x(i) & y(i) \end{bmatrix}^T$ is the lateral path. The resulting height profile of the approach path is shown in figure 3.3.

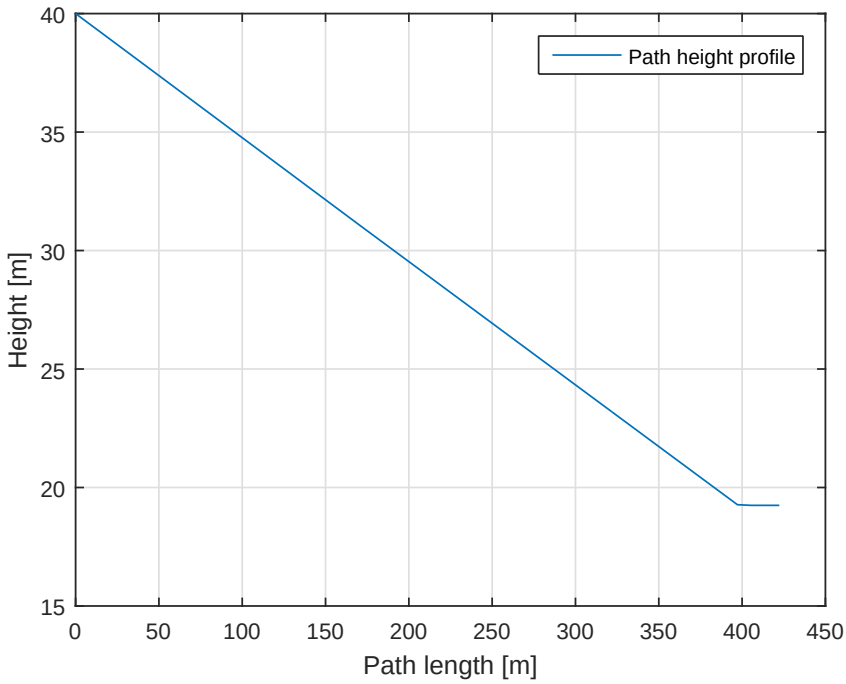

Figure 3.3: Height profile of the landing path

Spiral path In the case where the approach path has yet to reach the correct heigh at the end of the combined lateral and longitudinal path a new spiral path is created in order for the approach path to reach the height of the landing path. The spiral path is designed to have the same turning radius, turning direction and decent angle as the approach path. The longitudinal path continues along the spiral until the correct height can be reached with and decent angle equal or less then $\gamma_{d_{Max}}$. Continuing an arc is created such that the path ends with the correct heading. The arc has the same rotation direction as the lateral path, with start point where

the correct height was reached and end point of the start of the landing path. The complete approach path combined with the landing path is shown in figure 3.2.

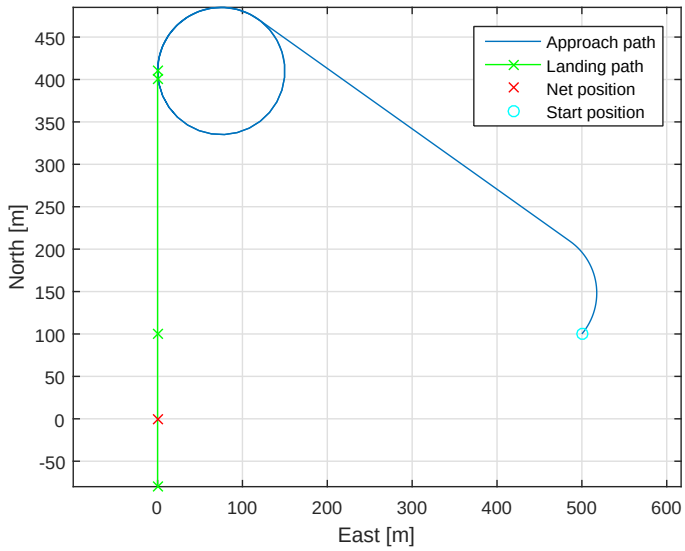

Figure 3.4: Approach path connected to the landing path

3.2 Navigation system

The navigation system consist of two position and velocity measurement system, where one is a high accurate positioning system and the other a reliable backup system. The high accurate positioning system apply RTK-GNSS, which is able to provide high accurate position solution. The backup positioning system consist of a standard package of standalone GPS and Inertial Measurement Unit (IMU) together with a Kalman filter, which is a proven reliable system in Ardupilot together with a Pixhawk. Ardupilot and Pixhawk are explained further in section 4.4 and section 4.3. However the RTK-GNSS is subject to drop out, which must be handled by the navigation system. A state switch controller has been created to handle the state switching between RTK-GPS and the external navigation data, which is navigation data from the Pixhawk. A short drop out of the RTK-GNSS could be resolved by fusing data from the external navigation source together with previous position solutions from the RTK-GNSS, and create a compensator for the external navigation system. The compensator can then be used to achieve the same accuracy as the RTK-GNSS for a short period.

3.2.1 Position estimation RTK-GPS

Real Time Kinematic GNSS (RTK-GNSS) is in [Misra and Enge, 2011] section 7.2.2 defined as a rover that receive raw GNSS measurements from a reference receiver which is transmitted over a radio link. A key feature with RTK-GNSS is that the rover is able to estimate the integer ambiguities while moving. The reference receiver is usually defined as a base station, and the integer ambiguity is the uncertainty of the number of whole phase cycles between the receiver and a satellite. With the measurements from the base station the rover is able to calculated the distance between itself and the base station, where the distance is referred to as a baseline. The length of the baseline affect the accuracy of the RTK-GNSS solution, due to increased effect of atmospheric disturbance, which is further explain in 3.2.2. However with a short baseline, e.g. $1 - 2km$, the atmospheric condition can be considered equal for the base station and the rover, which keeps the solution at centimetre level accuracy. The concept of RTK-GNSS is depicted in figure 3.5.

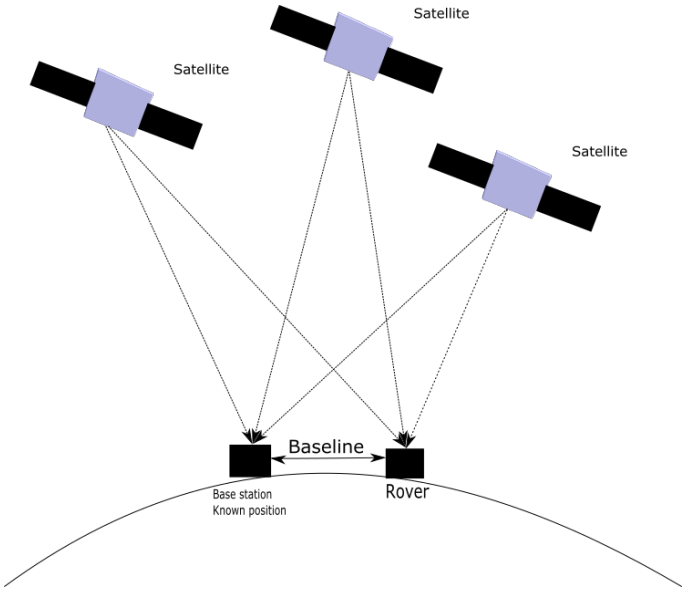

Figure 3.5: Concept figure of Real Time Kinematic GNSS (RTK-GNSS)

The ability for the rover to resolve the integer ambiguity is a key feature in RTK-GNSS. A well used method was purposed in the article [Teunissen, 1994] which decorrelate the integer ambiguities such that a efficient computation of the least square estimate can be performed. The search method is further explained in [Teunissen, 1995]. A estimate of the integer ambiguity with sufficient high degree of certainty is referred to as a FIX solution, otherwise the solution is degraded to

FLOAT where the integer ambiguity is allowed to be a decimal or a floating point number. When the solution is categorised as FIX the accuracy of the solution is considered on centimetre level, while with a FLOAT solution the accuracy is at a decimetre level. However when a FIX solution is lost, the solution accuracy will not immediately degrade to decimetre level.

In RTK-GNSS the position of the base station must be resolved. This can be achieved by either knowing the position beforehand, which is defined as a kinematic configuration. If the base station position is unknown the RTK-GNSS solver calculates the position on the fly, which is defined as a moving baseline configuration. The unknown position is then calculated as a standalone GNSS receiver, with the accuracy that entails. Therefore the RTK-GPS system with a moving baseline configuration can never have better global accuracy then what it will get with a single receiver. The advantage with the moving baseline configuration is that the base station is allowed to moved, and with RTK-GNSS the relative position between the rover and base station can be determined in real time. The advantage with kinematic mode is that it can give a more accurate position estimate, however this require that the base station is known and stationary.

3.2.2 Error sources

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

Clock error

There is drift in both the satellite clock and the receiver clock [Misra and Enge, 2011]. The atomic clock in the satellites makes the clock drift negligible from the user perspective. The receiver clock tend to drift, and if not taken into account will cause large deviations in the position estimate from the true position. This error is remove by including a fourth satellite in the position computation. The satellite clock error is given in the satellite message.

Ionospheric and tropospheric delays

When the GPS signals travel though the atmosphere there will be a delay caused by the different atmospheric layers [Misra and Enge, 2011]. The atmosphere change the velocity of wave propagation for the radio signal, which results in altered transit time of the signal.

Ionospheric delay Gas molecules in the ionosphere becomes ionized by the ultraviolet rays that is emitted by the sun, which release free electrons. These electron can influence electromagnetic wave propagation, such as GNSS signals. In [Vik, 2014] section 3.5.1 it's stated that the delay caused by the ionosphere usually is in the order of $1 - 10$meters. The error can be mitigated by using a double frequency receiver, or by applying a mathematical model to estimate the delay. Both those methods are with a single receiver, however by including a second receiver in a network, e.g. RTK-GPS, the GNSS solution system can assume that both receiver receive signal in the same epoch, which means that the signals have experienced the same delay. The rover is then able to remove the error induced from ionospheric disturbance.

Tropospheric delay The tropospheric delay is a function of the local temperature, pressure and relative humidity. The effect of tropospheric delay can vary from 2.4 meters to 25 meters depending on the elevation angle of the satellites,[Vik, 2014] section 3.5.1. The error can be mitigated by applying a mathematical model to estimate the tropospheric delay, or by using a elevation mask can remove all satellites with a elevation angle bellow a certain threshold. Similar to ionospheric delay, tropospheric delay can be removed when using two receivers in a network by assuming that the single received by both receivers has experienced the same delay.

Multipath

One of the primary source of error in in a GNSS receiver is multipath [Misra and Enge, 2011]. Multipath happens when the satellite signal is reflected by a nearby surface before if reach the GNSS antenna. The delay introduced in the signal can make the receiver believe that its position is several meters away form its true position. The easiest way to mitigated multipath is to place the antenna at a location with open skies, with no tall structures nearby. The effect can also be mitigated by choosing a antenna with good multipath rejection capability.

Multipath error uncorrelated between receivers, thus the local receiver must be able to correct for multipath error locally.

3.2.3 Navigation state control system

The navigation state control system manage the current state of the navigation system, which controls the RTK-GNSS usage and is responsible for dispatching the current state of the UAV to the rest of the DUNE system. A state diagram of the navigation state control system is shown in figure 3.6, which included trigger event on the edges and entry actions when entering a new state. The entry actions ensure that only functionality that is connected to the current state is active, and that monitors are updated with the state switch. The navigation state control system is designed to function without RTK-GNSS available, which makes the navigation

system independent from the RTK-GNSS system. This is a requirement for the navigation system since it should function in a UAV where RTK-GNSS is not present.

In order for the navigation system to switch state into "Rtk ready" the RTK-GPS message must be considered valid, which is defined in section 5.1.1. In addition the RTK-GPS solution type must be FIX for x seconds such that the RTK-GNSS solution can be considered highly accurate. For the navigation state control system to switch to RTK-GNSS, the user has to enabled it from the command and control software used to monitor the UAV. In the case where a loss of RTK-GNSS is experienced the short RTK-GNSS compensator will be enable to prevent the loss of RTK-GNSS accuracy, thus prolonging the availability of the RTK-GNSS. This state will be further explained in section 3.2.3. A summary of the states in the navigation state control system is given in table 3.2.

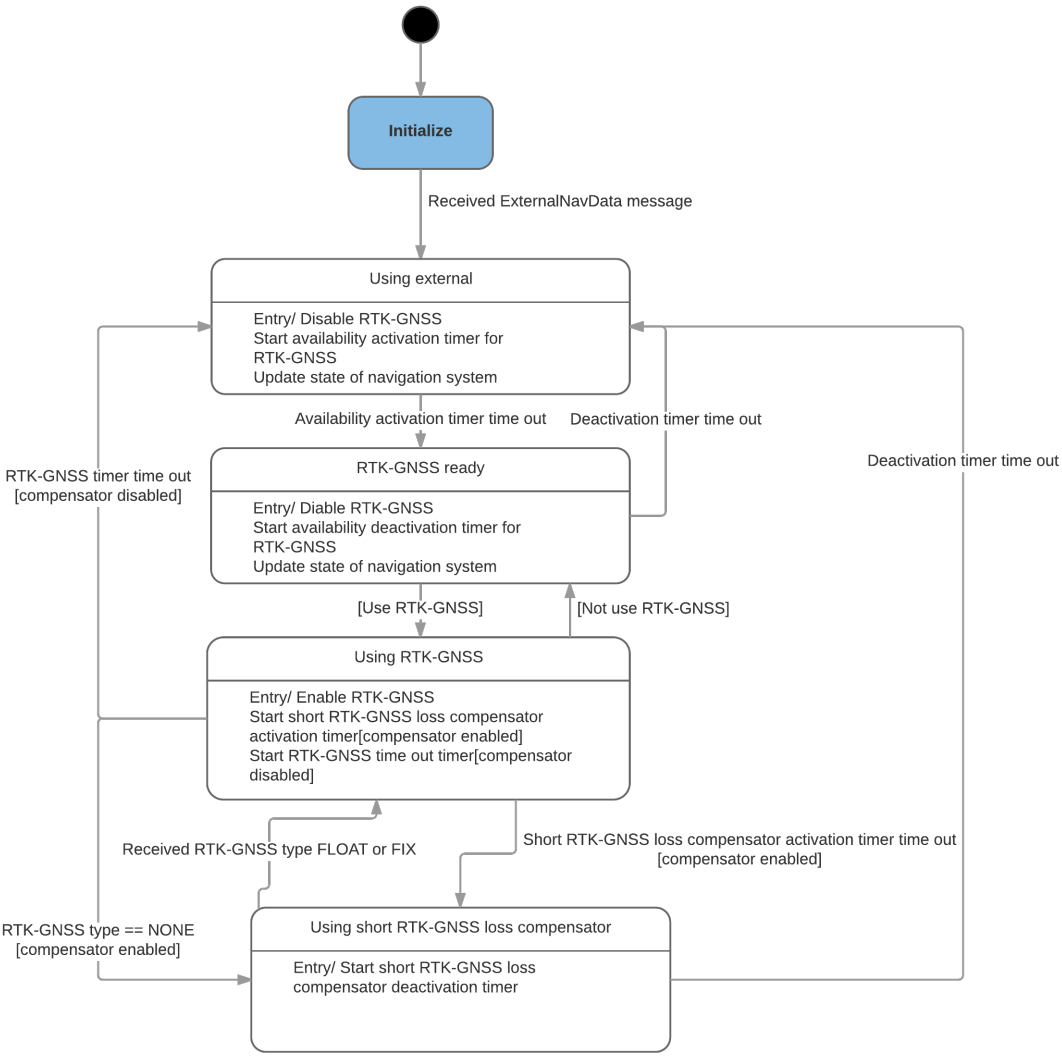

Figure 3.6: State machine in the DUNE Navigation task

State	Description
Initialize	The task starting up
Using external	The navigation task apply the external navigation source in the state message
RTK-GNSS ready	The RTK-GNSS is ready for use, however the external navigation source is still used
Using RTK-GNSS	The navigation task apply the RTK-GNSS in the state message
Using short RTK-GNSS loss compensator	The navigation task apply the external navigation source with a compensation term to reduce the effect of RTK-GNSS loss.

Table 3.2: States in the navigation system with description

Short loss of RTK-GNSS

In order for the navigation system to handle short loss of RTK-GPS a short loss RTK-GPS system is presented. The compensator is based on that the position solution from the external navigation system is almost constant with respect to the RTK-GPS solution. As seen in figure 3.7 the position of the RTK-GNSS and the external navigation system remain close to each other, given that the RTK-GNSS has a FIX solution.

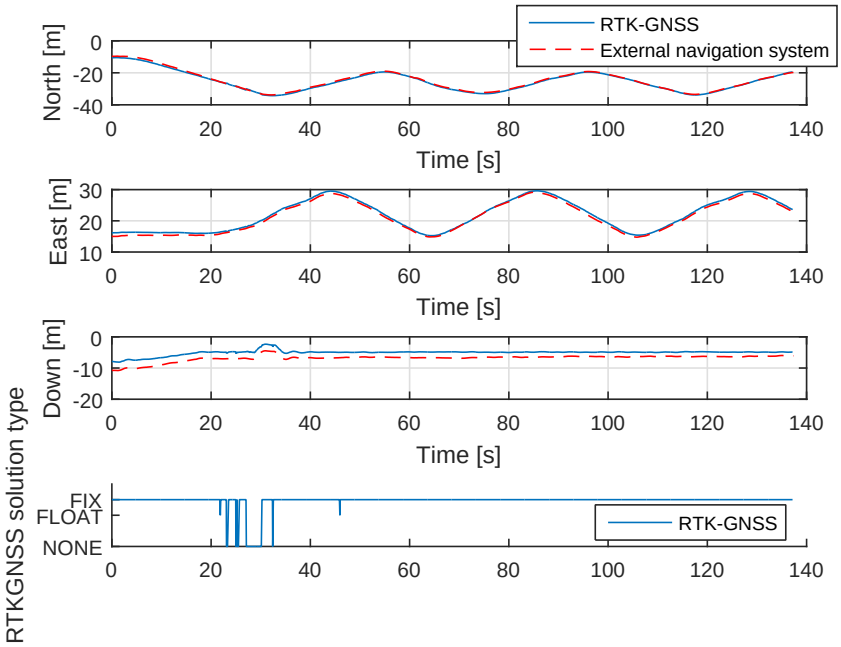

Figure 3.7: Plot where the RTK-GNSS solution is displayed together with the position solution from the external navigation system, include the solution type of the RTK-GNSS system.

The slow moving difference between the RTK-GNSS and external navigation system position solution is confirmed in figure 3.8.

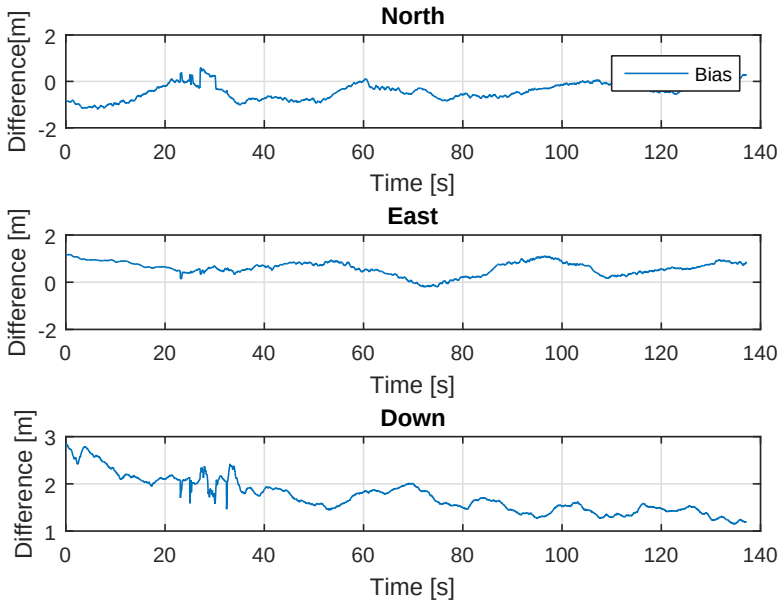

Figure 3.8: The difference between the RTK-GNSS solution and the external navigation system position solution

Therefore the average difference between the RTK-GNSS position solution and the external navigation system should be able to move the external navigation position solution closer to the RTK-GPS position solution. The short loss compensator is given as:

$$\mathbf{e}(n) = \mathbf{p}_1(n) - \mathbf{p}_2(n) \tag{3.13}$$

$$\delta = \frac{1}{N}\sum_{n=0}^{N}(\mathbf{e}(n)) \tag{3.14}$$

where $\mathbf{p}_1(n)$ and $\mathbf{p}_2(n)$ is the position solution sample for the RTK-GNSS system and the external navigation system respectfully. N is the total number of samples with $n \in [0, N-1]$ as the counting variable. Adding δ to the external navigation position with the assumption of slow varying bias between the two position solution gives:

$$\mathbf{p}_2(t) + \delta \rightarrow \mathbf{p}_1(t) \tag{3.15}$$

where $\mathbf{p}_1(t)$ and $\mathbf{p}_2(t)$ is the current position solution for the RTK-GNSS system and the external navigation system respectfully. The short loss RTK-GNSS compensator will trigger if there is a delay in the RTK-GNSS system, or a temporary drop out

occur. The output frequency of the short loss compensator is set to be the same as the RTK-GNSS system, which is estimate by comparing the time each RTK-GNSS message is dispatched. This results in prolonging the time where the RTK-GNSS is available for the navigation system, and will ensure that the navigation system outputs at a stable frequency. The goal with the compensator is to prevent mission abortion, and make the RTK-GNSS system more robust.

3.3 Summary

This chapter has presented a system description of both a landing plan generator and a navigation system with robust RTK-GNSS. The landing plan presented is separated into two parts, which is the landing path and the approach path. The landing path is a straight line path towards the net, whereas the approach path is a combination of Dubins path in the lateral plane and straight line path in the longitudinal plane. The approach path is designed to guaranty that UAV has a flyable path which enable the UAV to enter the landing path at the correct height with the correct orientation from any initial position.

The UAV navigation system has been design with a navigation state control system, which is used to control the current state of the source of the navigation data. Currently the different navigation source available to the UAV is a external navigation system and a RTK-GNSS system. In order to prolong the availability of the RTK-GNSS during short drop outs a short RTK-GNSS loss compensator system is presented, which is used to compensate the external navigation system position solution in order for the UAV navigation system to keep the a RTK-GNSS accuracy level during a RTK-GNSS drop out.

Chapter 4

Applied software and hardware

This chapter presents the third party software and existing hardware used to create the autonomous landing system.

4.1 LSTS toolchain

The software toolchain used in the autonomous landing system was developed by the Underwater Systems and Technology Laboratory (LSTS), which is called the LSTS toolchain [Pinto et al., 2013]. The toolchain was developed for support of networked heterogeneous air and ocean vehicle systems over wireless network. The toolchain contain four modules, namely Inter-Module Communication (IMC), DUNE, NEPTUS and Glued.

4.1.1 IMC

The Inter-Module Communication (IMC) message protocol [Martins et al., 2009] is design to enable interconnections between systems of vehicles, sensors and human operators, which enable the pursuit of an common goal by cooperatively exchange real-time information about the environment and updated objectives. The message protocol is oriented around the message, which abstracts hardware and communication heterogeneity with a provided shared set of messages that can be serialized and transferred over different means. The IMC protocol is defined in a single eXtensible Markup Language (XML) document, which simplify the definition of exiting messages and the creation of new messages. Table 4.1 includes the IMC messages which is used in the UAV navigation system.

Message name	**Description**
EstimtedState	Contain the current state information of the UAV, which has been made available for the DUNE system
ExternalNavData	Contain the navigation data from a external navigation system
NavSources	Contain the current source of the navigation data used in the UAV navigation system, in addition to all available navigation sources
GpsFixRtk	Contain the RTK-GNSS data

Table 4.1: Common IMC messages used in the UAV navigation system

4.1.2 Dune

DUNE (DUNE Uniform Navigation Environment) is a runtime environment for unmanned systems on-board software written in C++. DUNE is capable to interact with sensors, payload and actuators, in addition to communication, navigation, control, manoeuvring, plan execution and vehicle supervision. The software separate operations into different task where each has its own thread of execution. DUNE apply a message bus which is responsible for forwarding IMC messages from the producer to all registered consumers, which is the only way different DUNE tasks is communicating. The terminology dispatch and consume is used to describe whether a task send or receive a IMC message, with dispatch is used for a sent message and consume for a received message.

A DUNE task can be configured to be enabled in different profiles, where ArduPilot Software In the Loop (AP-SIL) and Hardware are the profiled used to test and verify the autonomous landing system. Separation is program execution profile allows for verification of a task in a simulation, where the DUNE system is executed as if it's connected to the hardware configuration.

4.1.3 Neptus

Neptus is a Command and Control software used to command and monitor unmanned systems. Neptus is able to provide coherent visual interface to command the vehicles in the DUNE system despite the heterogeneity in the controlled system which Netpus is interacting with. This allows the operator to command and control unmanned system without the need to dwell into specific command and control software in the unmanned system. The main communication channel for Neptus is the IMC message protocol, which makes it interoperable with DUNE or other IMC- based peer.

The Neptus console can be configured to suit an unmanned systems operational requirements by including plug-ins that express information critical to the unmanned system, in addition to plug-in used to execute certain actions from the unmanned system. This flexibility makes Neptus ideal for experimental testing of new features applied to unmanned systems.

Neptus can be used to review the mission logs generated by DUNE in the Mission Review and Analysis (MRA). Mission analysed in MRA can be exported to a Matlab format, which allows for a deeper analyse of the performance of a system after a completed mission.

4.1.4 Glued

Glued is a minimal Linux operating system distribution, designed with embedded system in mind. It is platform independent, easy to configure and contain only the necessary packages to run on a embedded system. This makes GLUED a light and fast distribution, which is ideal for a on-board operating system for a unmanned system where payload size is normally limited. GLUED is configured through a single configuration file that which can be created for a specific system. A advantage with Glued is that it can be cross-compiled, which allows for compilation of software before it's transferred to the embedded computer.

4.2 RTKLIB

RTKlib [Takasu and Yasuda, 2009] is an open source program package for standard and precise positioning with GNSS developed by T. Takasu. RTKlib can be configured as a RTK-GNSS system, where raw GNSS data from the base station and rover is fused together, and used to estimate the relative position of the rover with respect to the base station in real time. The communication flow for a RTK-GNSS system using RTKlib in a DUNE system is shown in figure 4.1, where the modules str2str and rtkrcv in RTKlib is the base station software and rover software respectfully. The version of Real-Time Kinematic Library (RTKLIB) used in this thesis is RTKLIB2.4.2 [RTKLIB].

RTKlib can be configured as both a moving baseline and kinematic configuration, which is described in section 3.2.1. The configuration used in this autonomous landing system is the moving baseline configuration, due to increased operational mobility of the base station. The effect of this configuration is that the global positioning accuracy is lower then the relative position accuracy with respect to the base station. However this is not a problem for the autonomous landing system, since as long as all system participants apply glsrtk-gnss with the same base station position, then high baseline accuracy is good enough to safely perform a autonomous net landing.

The RTKlib configuration file used in the autonomous landing system is given in appendix E.

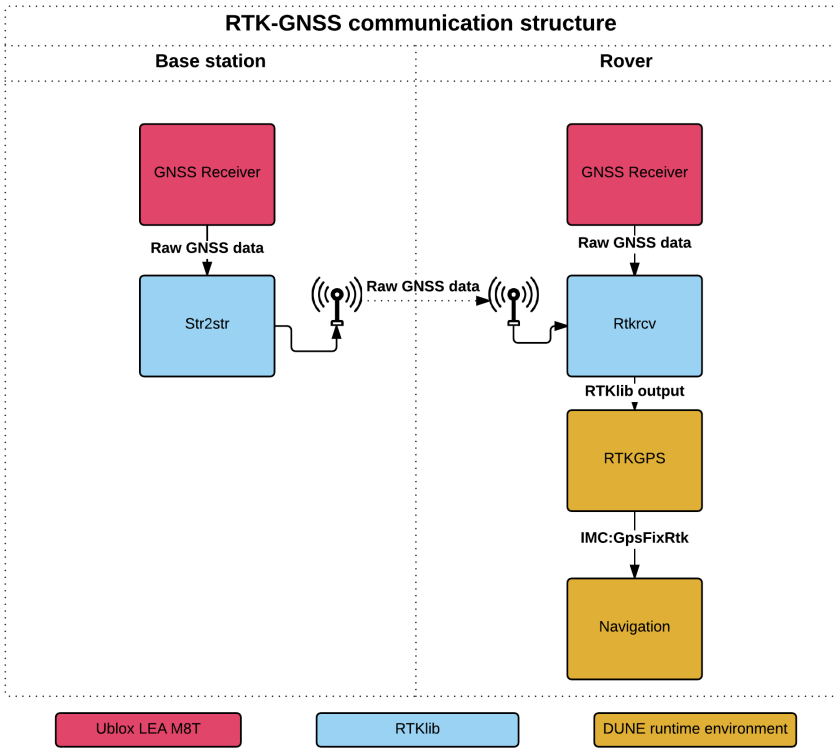

Figure 4.1: The communication structure a RTK-GNSS system with RTKlib, connected to DUNE

4.3 Pixhawk

The 3DR Pixhawk is a high-performance autopilot suitable for fixed wing and multi rotors UAV and other robotic platform which can move. The Pixhawk system comes complete with GPS, imu, airspeed sensor and magnetometer, is used as an external navigation system for the DUNE environment.

4.4 Ardupilot

Ardupilot is an open-source unmanned aerial vehicle platform, able to control both fixed wing and multi rotor UAVs, and can run on the Pixhawk platform. Ardupilot allows for both manual flight control and autonomous flight operations, with a command and control software called Missionplaner. Ardupilot can be used in a Software In the Loop (SIL) simulation, where ardupilot connects to a simulator of the desired vehicle platform. This allows for verification of software before attempting a hardware test. The simulator used in a SIL simulation can be a third party software, which motivates the creation of mathematical models for the desired UAV platform. Ardupilot is used in the autonomous landing system presented in this thesis, where the low level controller is controlled in Ardupilot. The control system in Ardupilot can be separated into high and low level control. The low level control system is used in the control loops for the actuators, while the high level controller manage the desired state of the low level controllers. Ardupilot can be configured to allow for third party high level controller, which involves three levels of high level control outsourcing. The guided mode accepts way point from a third party software, however both high and low level control is managed in Ardupilot. Fly By Wire-A (FBWA) is the mode where all high level controllers are in a third party software, while Fly By Wire-B (FBWB) is a hybrid between FBWA and guided where only the lateral high level controller is in the third party software. The different mode is listen in table 4.2.

Mode	Description
Guided	Ardupilot accept third party waypoints, and has both high and low level controllers
FBWB	Third party lateral controller with desired height controlled in a set-point regulated controller in Ardupilot
FBWA	Third party lateral and longitudinal controller, with control input is sent directly to the low level controllers in Ardupilot

Table 4.2: Guidance and control modes in autopilot viable for the landing system

4.5 JSBsim

JSBSim [Berndt, 2004] is an open-source flight dynamic model that is able to simulate a physical model of an arbitrary aircraft without the need of specific compiled and linked program code. The simulator is design such that a third party software e.g. Ardupilot can expose the model to external forces and moments. This is useful in a SITL simulator for verification of software, which is design for a hardware configuration. The physical model that was used in this thesis was developed in the master thesis [Gryte, 2015].

4.6 X8 and nest payload

The Skywalker X8 is fixed wing UAV in a flying wing configuration, which indicate that the UAV has no tail and clear distinction between the wings and fuselage. The X8 is a popular choice for experimental missions at the UAV-lab at the Deparment of Engineering Cybernetic since it's durable, cheap and enough space to carry experimental payload.

The hardware configuration used in the X8 and nest systems is based on the proposed hardware in the paper [Zolich et al., 2015]. The nest system is a mobile base station, which is used as both a base station and communication link between the X8 and Neptus. The X8 and the nest systems are installed with a BeagleBone embedded computer with the Glued operating system, which is used to run the Dune system, as well as RTKlib. The autopilot used in the X8 is a 3DR Pixhawk with ArduPilot ArduPlane software. For the RTK-GPS system Ublox Lea M8T GNSS receivers [U-blox, a,b] are connected to the BeagleBone with uart cable. The antenna used in the X8 is a Maxtena M1227HCT-A-SMA L1/L2 GPS-GLONASS

Active Antenna [Maxtena], and the antenna used in the base station is a Novatel GPS-701-GG [Novatel].

The communication between the X8 and the nest systems is done with Ubiquiti M5 rocket [roc] radios, where the communication between each unit can be done with TCP/UDP/IP.

Chapter 5

Implementation

The autonomous landing system for stationary net landing consist mainly of three modules, which is the Navigation, LandingPlan, and Path Controller tasks. A simplified figure of the system flow in the autonomous landing system in DUNE is showed in figure 5.1. The implementation and system description of the path controllers used in the autonomous landing system is not a focus area in this thesis, however a short description is given in appendix B and a closer analyse of the path control system can be found in the master thesis [Nevstad, 2016]. The DUNE task Ardupilot is used as interface towards the Ardupilot software which either runs as in a simulation mode or on a Pixhawk. The DUNE system is command and controlled with Neptus, which has been modified to suit the needs of the autonomous landing system.

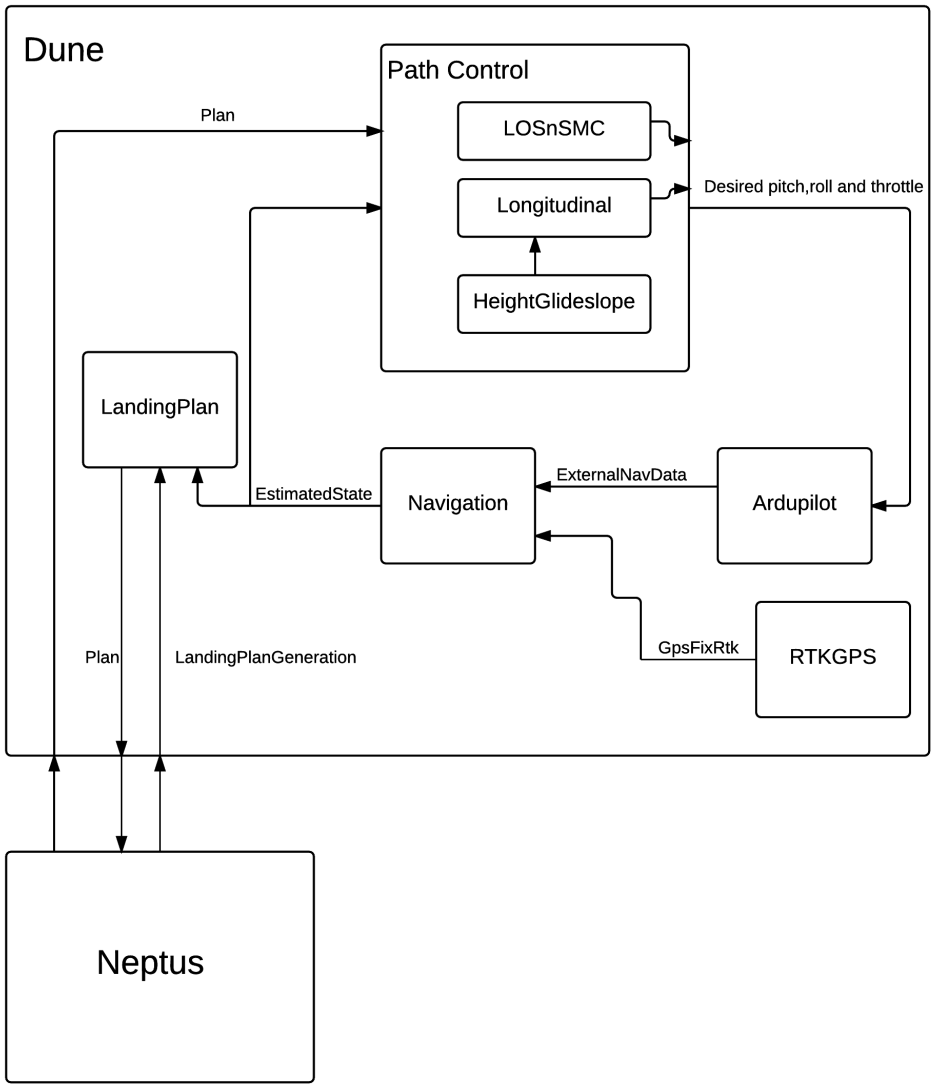

Figure 5.1: A simplified depiction of the interaction between the major components in the autonomous landing system during stationary net landing

5.1 Navigation system

The navigation state control system described in section 3.2.3 is implemented in the DUNE task Navigation, which is used to control the position and velocity information source of the navigation state IMC message EstimatedState. Depending

on the current state of the navigation system the IMC EstimatedState message will either have position solution form the RTK-GPS system or the external navigation system. During a short loss of the RTK-GNSS the external navigation position is compensated with the average difference between the RTK-GNSS solution and the external navigation solution. The state of the navigation system is monitored through the IMC message NavSources, which contain the information about the source of the state information, including which alternative navigation system is available for the UAV navigation system.

5.1.1 RTK-GPS system

The RTK-GNSS solution is made available to the DUNE system through the DUNE task RTKGPS, which is an modified version of the same DUNE task developed in the master thesis [Spockeli, 2015]. The RTK-GNSS solution is included in the IMC GpsFixRtk message, however in order for the message to be valid the following flags must be set true:

- Valid velocity
- Valid position
- Valid time
- Valid base

The three first flags are set automatically when receiving a output solution from RTKLib, however since the base station position is not included in the RTKLib output the valid base flag will not be set automatically. In order for the GpsFixRtk message in the UAV to get a valid base station position, the base station must calculate it's own position and send it to the UAV. Figure 5.2 shows the message flow which is needed in order for the GpsFixRtk message in the UAV to be considered valid by the UAV navigation system. The DUNE task basestationFix must be reconfigured that the base station position is fixed from Neptus in order for the base station to start transimiiting its own position to the UAV. The advantage with a fixed base station position in the DUNE system is that all vehicle that uses RTK-GNSS will be in the same reference frame, which enable high accurate vehicle coordination.

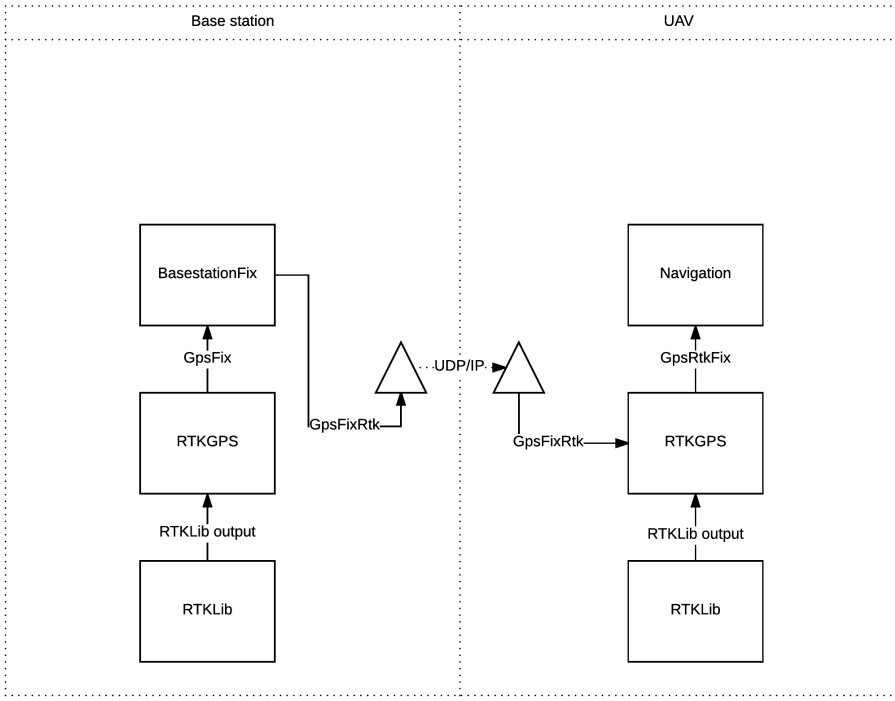

Figure 5.2: Message flow for validation of GpsFixRtk

5.1.2 Navigation state control system

The navigation state control system described in section 3.2.3 is implemented in the DUNE task Navigation. Figure 5.3 shows the system flow in the navigation state control system, which is trigger by either receiving a ExternalNavData or GpsFixRtk IMC message. Both types of messages is stored internally in the DUNE task, however the GpsFixRtk is used to trigger update of the short rtk loss compensator. The state machine performs its state transition action while entering the new state. The state change will then trigger an alteration in the content of the EstimatedState IMC message whether the position and velocity information should be from RTK-GNSS or from the external navigation system.

In the case where RTK-GNSS is lost the short RTK-GNSS loss compensator described in section 3.2.3 is implemented in the Navigation task. The compensator is used to prolong the availability of the RTK-GNSS, however if the Navigation task does not receive a valid GpsFixRtk before the deactivation timer triggers the resulting action from the navigation state control system would be to set the RTK-GNSS system as unavailable.

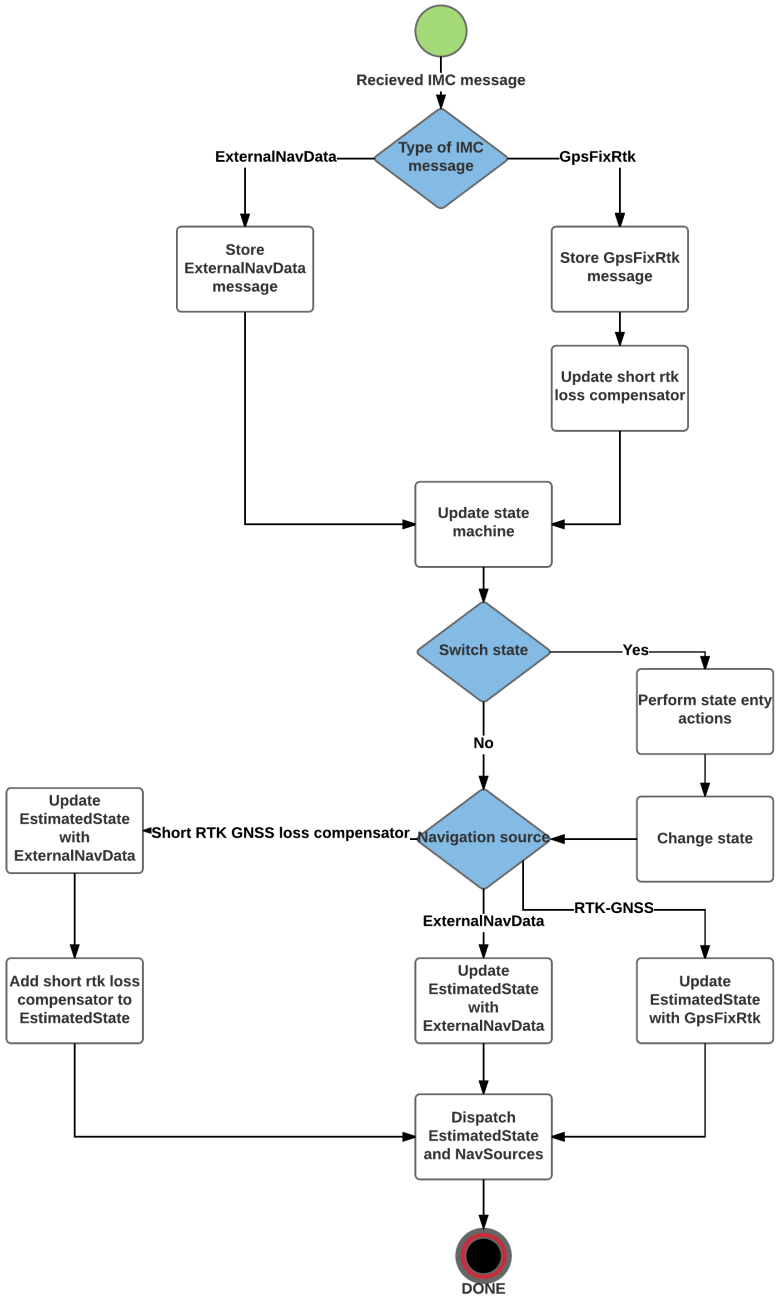

Figure 5.3: Flow chart of the navigation state control system

5.1.3 Mobile sensor unit

The mobile sensor unit is a box that can moved around and act as a reference position for other vehicles in the DUNE systems, e.q net placement for the autonomous landing system. The mobile sensor unit apply RTK-GNSS in the same manor as a UAV, however the mobile sensor unit does not include a external navigation system. Thus a simplified version of the Navigation task has been created for the mobile sensor unit, which is the DUNE task RtkNavigation. The DUNE task RtkNavigation handle the GpsRtkFix message similar to the Navigation task, however due to not having a external navigation system only the GpsFixRtk position solution is used in the EstimatedState message. Figure 5.4 shows a picture of the mobile sensor unit.

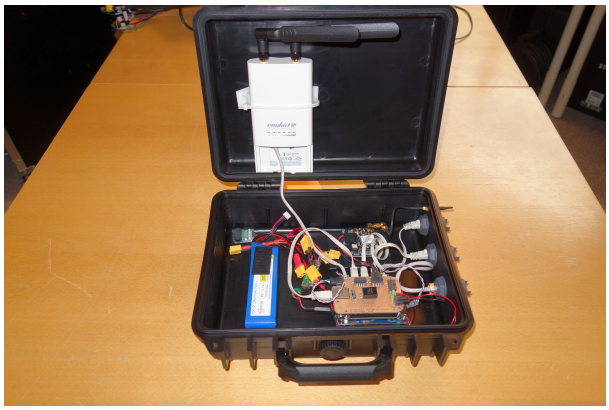

Figure 5.4: The mobile sensor unit

5.1.4 Navigation source monitor

The navigation source monitor is a plug-in in Neptus used to monitor which navigation source a vehicle is using. The monitor consume the NavSources IMC message, which contain the navigation sources that are currently in use and which are available. The monitor apply a color code to indicate which source is currently in use, in addition to all sensor system that are available. A figure of the navigation source monitor is seen in figure 5.5, with the color code description given in table 5.1.

Color	**Description**
White	Not available
Yellow	Available, but not in use
Green	Available, and in use

Table 5.1: Net approach parameters

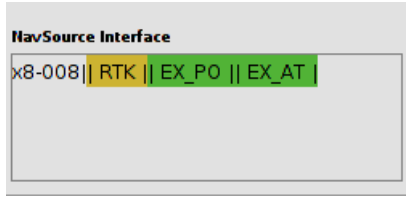

Figure 5.5: Navigation source interface

5.2 Landing plan generator

The DUNE task LandingPlan is the implementation of the landing plan generation system described in section 3.1. The DUNE task receive its plan parameter through a API, which is a IMC message called LandingPlanGeneration. The task is triggered by the event a LandingPlanGeneration message is consumed, which results in the generation of the approach and landing path. Together the paths form the landing plan. The API can be accessed from Neptus through the plug-in LandMayLayer, enabling a graphical interface to be used for net placement.

5.2.1 Landing plan generation API

The landing plan generation API is a IMC message used to structure the input parameter used to create the landing plan. With the API the desired path parameter can be set, in addition to behaviour setting used to create a specific landing plan. The API can be used to set the rotation direction of the start and finish turning circle, by setting the "Automatic" flag to false. In addition a loiter manoeuvre can be added to the landing path, which act as a waiting manoeuvre. The behaviour settings in the API are listed in table 5.2, with the entire API listed in appendix A.

Parameter name	**Description**
Automatic (boolean)	If true a standard path where the shortest Dubins path is chosen as the approach path. Otherwise a user specific path is chosen
Start circle turning counter clockwise (boolean)	If true the start turning circle is created with a turning direction which is counter clockwise. Otherwise clockwise. Require Automatic==false
Finish circle turning counter clockwise (boolean)	If true the finish turning circle is created with a turning direction which is counter clockwise. Otherwise clockwise. Require Automatic==false
Wait at loiter (boolean)	If true a unlimited loiter is included into the landing plan.

Table 5.2: Landing plan behaviour setting in Landing plan generation API

Neptus API graphical interface

In Neptus the plug-in LandmapLayer, which is an altered version of the Neptus plug-in developed in the master thesis [Frølich, 2015], is used to configure the landing plan generation API. The alteration in the plug-in includes new parameters, the inclusion of the IMC message LandingPlanGeneration and the ability to manually write the global position coordinates of the net. The API graphical interface which is used in Neptus is shown in figure 5.6. The LandmapLayer plug-in works by first placing the net in Neptus, continued by setting the desired parameters of the landing plan in the graphical interface.

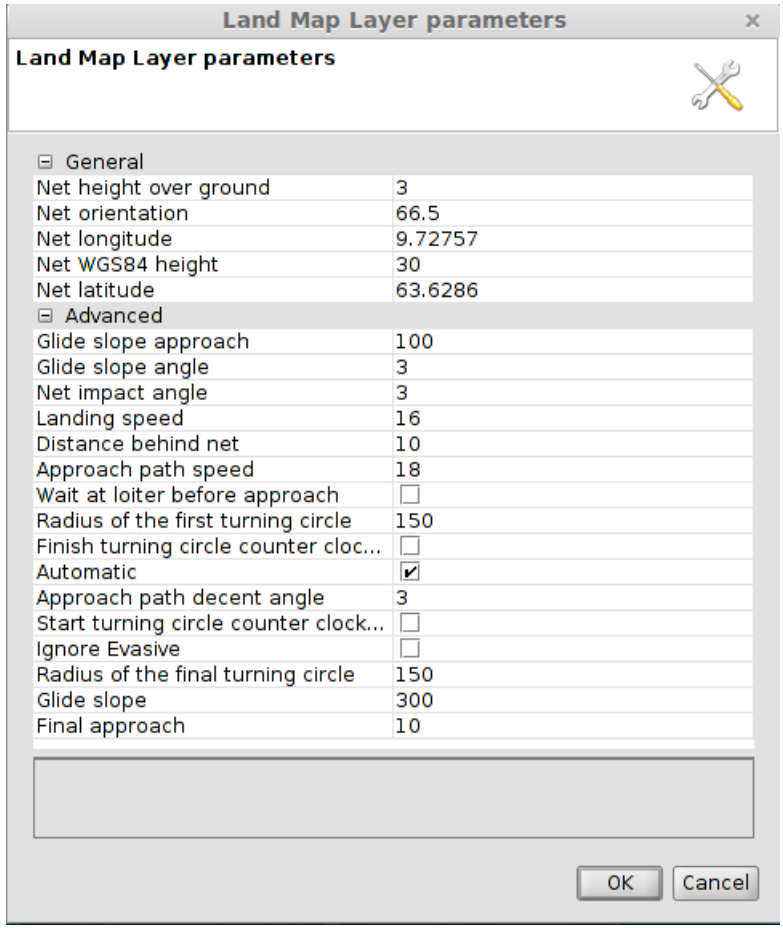

Figure 5.6: Graphical interface for the landing plan generator API in Neptus

5.2.2 Approach path

The approach path described in section 3.1.2 is implemented in the DUNE task LandingPlan, where the start position of the approach path is the initial position of the fixed wing UAV at the time the LandingPlanGeneration message is sent. The final position of the approach path is the first **WP** in the landing path. Figure 5.7 shows the structure of the creation of the approach path in the landing plan. The creation is trigger by the consumption of a LandingPlanGeneration message, which is extracted in order gain access to the landing plan parameters.

The creation of the approach path is designed to enable the user to specify the rotation direction of the start and finish turning circles. This design choice was made such that the user has a guaranty of the turning direction in both circles, which

is not given when calculating the shortest Dubins path. Thus the two mode have different guaranties, such that the landing plan generator becomes more flexible.

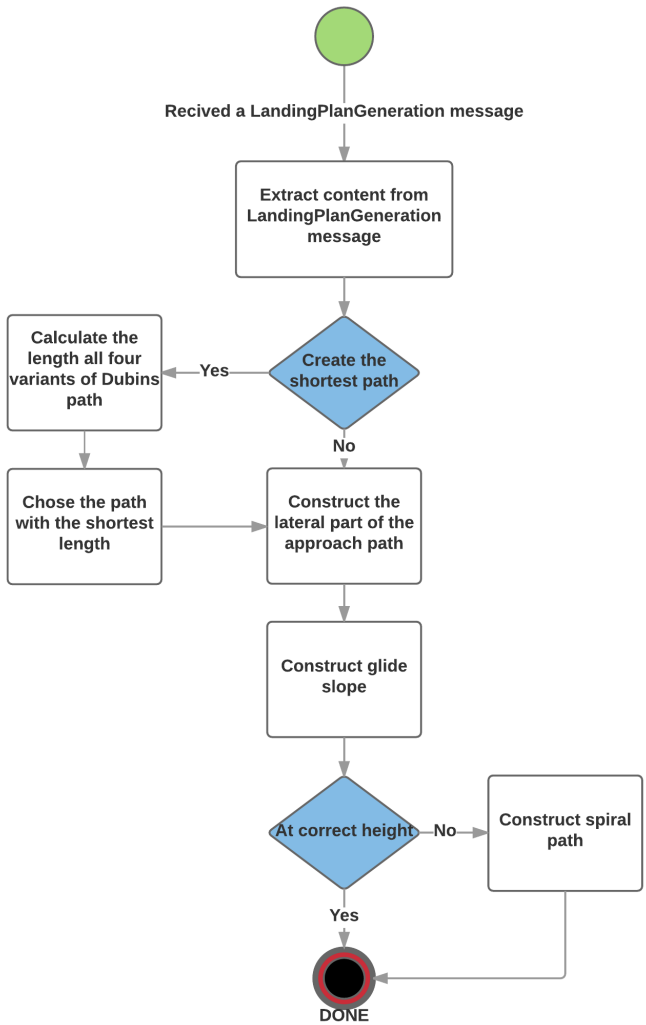

Figure 5.7: Flow chart of approach path creation

The approach path is created as a FollowPath manoeuvre, which is a manoeuvre where each point in the manoeuvre is defined relative to a fixed reference position. This manoeuvre is suited for more complex manoeuvres, which is the reason it's used in the approach path. The task configuration parameter "Distance Between Arc Segments" is used to specify the distance between each point in both the turning circles, thus given the total number of segments in the circles. The parameter can be

Configuration parameter name	Default value	Description
Distance Between Arc Segment	$25m$	Distance between each arc segments in the turning circles

Table 5.3: Configuration parameter for the landing plan

used to increase the performance of the lateral control system, by continues switching point to create a smoother circle. The default value of the configuration parameters "Distance between Arc Segments" is given in table 5.3.

5.2.3 Landing path

The landing path described in section 3.1.1 is implemented in the DUNE task LandingPlan, where the path is created relative to the position and heading of the net retrieved from the LandingPlanGeneration message. Figure 5.8 shows the system flow for the creation of the landing path. The creation is trigger when the approach path has successfully been created. A option for the landing path is that it includes a loiter manoeuvre at the beginning, which is defined as a circular manoeuvre around a fixed position with a constant radius. The loiter manoeuvre increase the flexibility of the autonomous landing system, by introducing a manoeuvre in which the UAV can wait for the landing zone to be prepared. In the case of a dynamic net landing the loiter manoeuvre can be used as a waiting manoeuvre as a final check before the UAV starts to track the position of the net. An other possible application is to apply the loiter manoeuvre in a net landing where the net is carried by multi-copter UAVs, where the copters can wait on the ground until the fixed wing UAV enters the loiter manoeuvre.

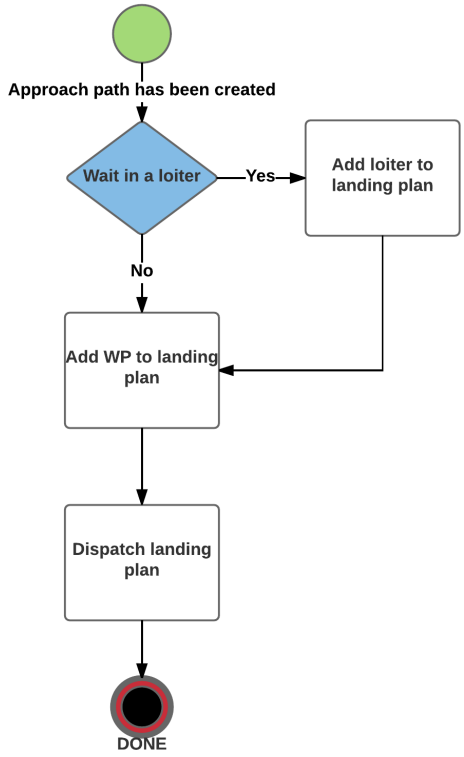

Figure 5.8: Flow chart of the landing plan generation

5.3 Software in the loop simulation

5.3.1 Outline of testing

The landing plan was verified and test through the use of a Software In the Loop (SIL) simulation, where the landing plan generation code runs as if it's connected to the actual hardware. A SIL simulation is used to verify that the code function as it's design to do. During a SIL simulation Ardupilot enters a simulation mode, where the JSBSim simulator is used as replacement of the actual X8 fixed wing UAV. The result obtain from a simulation is used as a ideal test case, from which the performance of a flight test can be compeered against. However the current model of the X8 used in the simulation has not been completely verified, such that deviation in results and behaviour is expected. Figure 5.9 shows a screenshot of a landing plan created in Neptus with the green triangle symbolizing the net placement.

The autonomous landing system flies in the Ardupilot mode FBWA, where the

high level control system is implemented in DUNE. The longitudinal control system during SIL simulation of the autonomous landing system is designed and tested in the master thesis [Nevstad, 2016], and the lateral control system was design and implemented as part of the paper [Fortuna and Fossen, 2015]. A short description of both control system is presented in appendix B. The low level controllers in Ardupilot used in the SIL simulation are fined tuned for autonomous flight, which allows for ideal test conditions on the performance of the high level controllers. In DUNE the time of arrival factor control when the path control system switches between waypoints, which can be used during turning manoeuvres to force the UAV to closer follow the path. During the SIL simulation the time of arrival factor is set to $2s$.

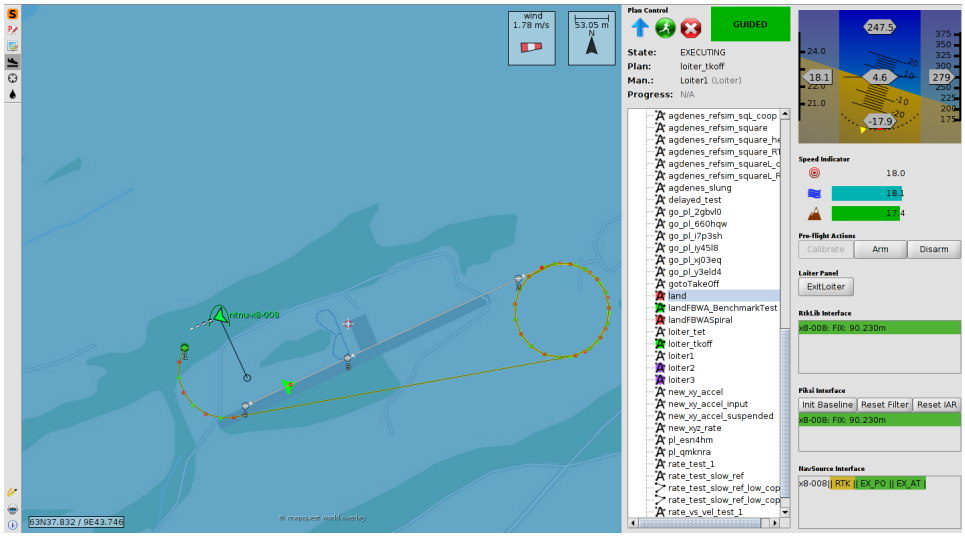

Figure 5.9: Path generated from the landing plan generator, with the net placement as a green triangle

5.3.2 Landing plan generation

A landing path was created to simulate a real landing, with the lateral path is shown in figure 5.10 and the height versus the desired height shown in figure 5.11. The landing plan is design to fit the operation area in which the fixed wing UAV can operate during LOS flight operation.

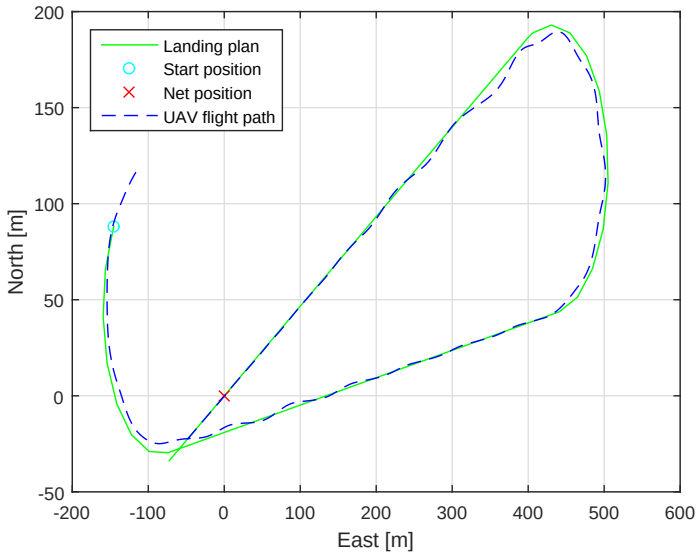

Figure 5.10: North-East plot of a SIL simulation of the autonomous landing system.

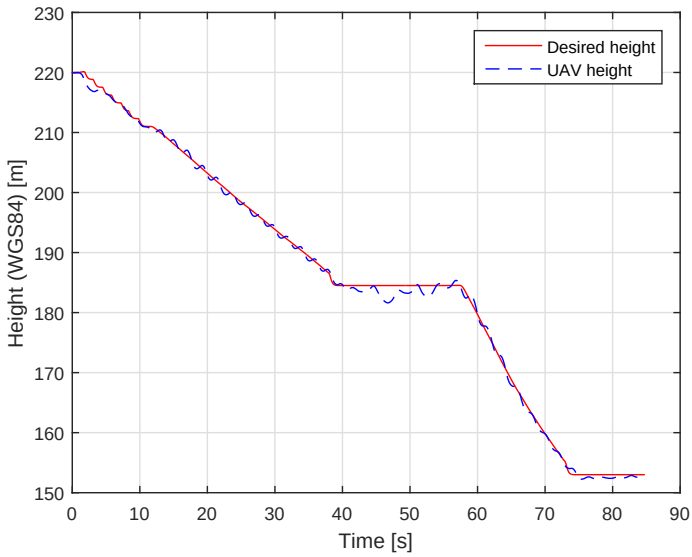

Figure 5.11: The desired height and UAV height when executing the landing plan.

5.3.3 Spiral path creation

During a landing plan the height at the end of the approach path may not match the start height of the landing path. In those cases the spiral path described in section 3.1.2 is created in order for the approach path to reach the correct height. In figure 5.12 the spiral function of the lateral path was tested, with the resulting height profile in figure 5.13. The simulation was performed with a wind disturbance of $9m/s$ from the west of the net position, to simulate how the UAV would perform during the landing plan with wind disturbance. The direction of the wind is set such that the landing path is directed against the wind, which the optimal direction to land in order to reduce the ground speed of the UAV.

The lateral control system struggles when flying with the wind, however when flying against the wind it's able to stay on the straight line between the way-points. This is as expected and makes physical sense as the fundamentals of flying is lift created by air velocity over the wing, where the flaps are used to create steering force based on same principle, i.e. air flow over the wing from front to aft with a certain velocity is required to create any steering force/rotational moment. During the turn in the spiral the lateral control system is unable to stay on the circle, thus overshooting the desired path. The longitudinal control system behave similar to the simulation without wind, indicating that as long as the UAV can fly in the wind the performance from the longitudinal control system would remain the same as with negligible wind conditions.

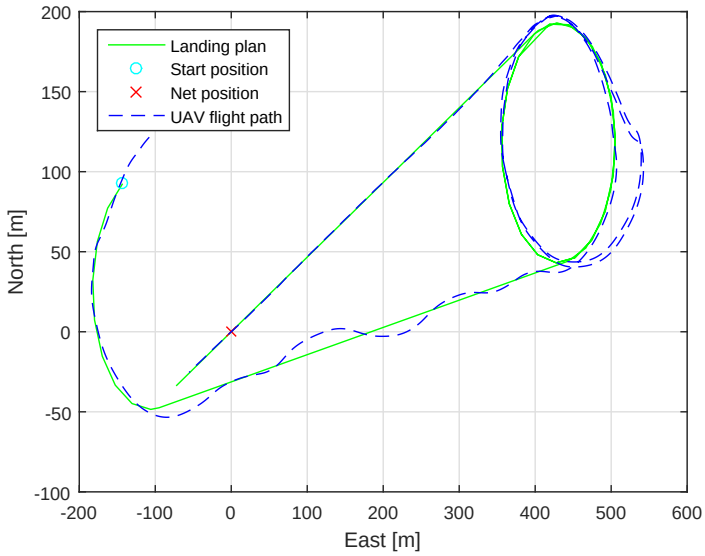

Figure 5.12: North-East plot where the approach path enters a spiral in order to find a path to the correct height. The simulation was performed with $9m/s$ wind from west

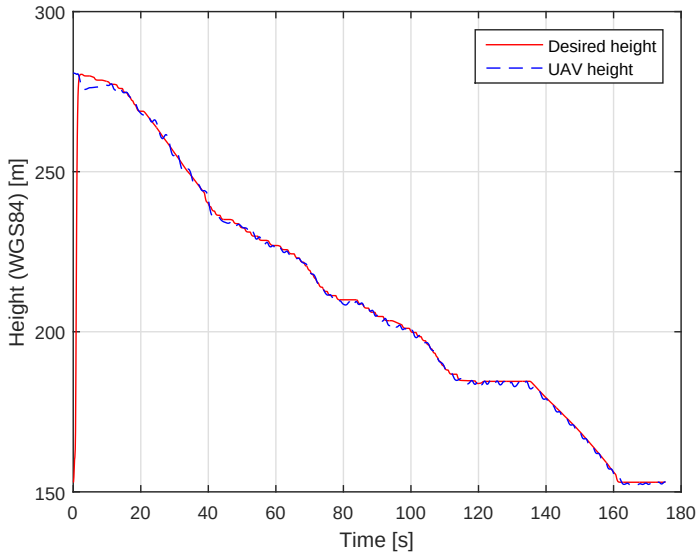

Figure 5.13: The desired height and UAV height when executing the landing plan from a height that trigger a spiral path towards the correct height with maximum decent angle $\gamma_{d_{Max}}$. The simulation was performed with $9m/s$ wind from west

5.3.4 Result of simulations

The system performance in a simulation environment is presented, where the success criteria is if the UAV was within the net acceptance criteria at the time the along track error equalled the distance to the aiming point behind the net position. The acceptance criteria used to indicate if the UAV would have hit the net is given in table 5.4, which is related to a net with the dimensions 3 meter height and 5 meter width.

Height acceptance	**Cross track error acceptance**
± 1.5	± 2.5

Table 5.4: Net hit acceptance criteria

The result from six simulations is given in figure 5.14, with the overall performance of the path following capability of the control system given in table 5.5. All the simulations of the autonomous landing system gave a result where the UAV was able to pass the net with a accuracy that is within the acceptance criteria, showing that the system is able to perform autonomous landing. The height difference between the net placement and the start of the landing path was $31.5m$, where the glide slope

angle was set to 6 deg. A higher glide slope angle would result in the UAV to build up speed, thus with the current longitudinal control system the angle of the glide slope should not exceed 8 deg.

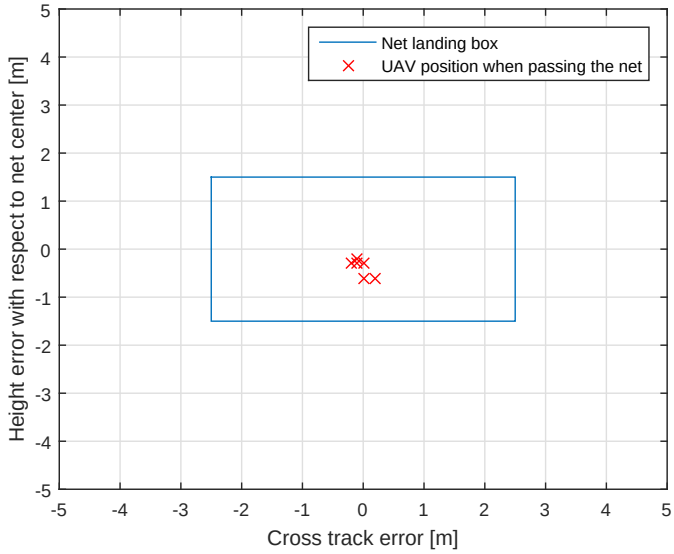

Figure 5.14: UAV position at time of net passing during SIL simulation.

Nr.	Average height error [m]	Average cross track error [m]
1	−0.3	−3.1
2	0.7	−4.0
3	0.2	−3.3
4	0.5	−1.2
5	0.4	−2.5
6	0.2	0.3

Table 5.5: Average cross track error and height error relative to the path.

5.4 Summary

This chapter has presented the system implementation for the path and navigation system described in chapter 3, as well as presented a mobile sensor unit. The landing plan generator has been tested and verified in a SIL simulation, together with the control system which will be used in experimental flight tests. The results from the SIL simulation shows that the system is capable of performing a autonomous

landing, however it's expected a deviation in behaviour due to the use of a X8 model that has not been completely verified. In addition the low level controller in the simulator are fined tuned for autonomous flights, which is not the case of the physical X8. The landing path generator is design to as flexibly as possible when it comes to creating a operational valid landing plan. By having the option of selecting the rotation direction of both the turning circles, a landing plan can be created to suit the surrounding environment to minimize the risk of a crash or the UAV leave the line of sight. The loiter manoeuvre that included in the landing plan allows a coordination stage, in which the landing zone can be prepared when the UAV enters the loiter.

The UAV navigation system has been design to enable the use of two positioning system, where one is more accurate then the other. The RTK-GNSS has been made more robust by the creation of a short RTK-GNSS loss compensator, which apply sensor fusion with navigation data from a secondary GNSS system in the case of RTK-GNSS drop out. The navigation source monitor has been design to allow the operator to get a feedback on which navigation source the navigation system is currently using, and which other navigation systems are available.

The mobile sensor unit has been design to function as position reference in Neptus for net placement. The unit comes with it's own DUNE system with RTK-GNSS, which can be expanded with a closer interaction with the autonomous landing system. As the system is designed today, the net position must either be placed or written in manually in Neptus. An other solution would be that the mobile sensor units position becomes the net position. This would allow for a more advance autonomous landing where target tracking is used by the fixed wing UAV to land at a offset position above the mobile sensor unit, which is essential in autonomous landing system where the net is dynamical. A similar system has been developed in the master thesis [Moe, 2016], however here the net is controlled by multirotor UAVs.

Chapter 6

Experimental field tests

The autonomous landing system has successfully been tested in the field, where the results from two subsequent days is presented.

6.1 Outline of testing

The experimental field tests was performed at Agdenes airfield with a virtual net placed $26m$ above the runway, which was performed by placing the mobile sensor unit out on the runway to act as a reference position in Neptus. The runway at Agdenes has the direction west-east, which is used to define to possible direction from which the UAV can perform a autonomous landing. The autonomous landing system is applied with the same control system which was used in the SIL simulation in section 5.3.1, however it should be noted that the low level controllers in the hardware configuration of the X8 has not been fined tuned for autonomous flight, which will decrease the performance of the high level controllers compared to the SIL simulation. Similar to the SIL simulation the time of arrival factor is set to $2s$ during the landing plan. The UAV navigation system used in the autonomous landing operation is the system presented in section 5.1. The flight operation is a Line Of Sight operation, which means that the UAV must be within the line of sight of the pilot at all time.

The weather condition differed over the course of the two days, where the first had windspeeds between $8-9m/s$ from the west while the second day was considered calm. Hence the performance of the autonomous landing system was tested during two different wind condition, where one strained the performance of the system while the other could be considered as ideal conditions. All landing plan was generated when the UAV was in a loiter manoeuvre, in order for the plan to be reviewed and the correct controllers assigned to the plan before the execution of the landing plan.

6.2 Landing plan generation system

6.2.1 Day 1

First plan

The first test of the autonomous landing system was performed with the landing plan parameter listen in appendix C.2, which resulted in the path shown in figure 6.1, where the the start position of the landing plan is marked with a circle, the net position with a cross, the desired path as a whole line and the actual flight path as a stippled line. The desired heigh as well as the actual height is shown in figure 6.2. The rotation direction of the start turning circle was chosen to be opposite to the finish turning circle, which resulted in a straight line path between the two turning circles bringing the UAV into the cross wind. The effect of flying in the cross wind introduced oscillatory motion in the UAV, which the lateral control system was unable to compensate for. The effect of the oscillatory motion affected the UAV when entering the finish turning circle, where the result was the UAV overshooting the desired path. The consequence of the UAV overshooting the finish turning circle can in the worst case result in the UAV to leave the line of sight of the pilot, which is a failure when flying in a LOS operation. The UAV continues to oscillate when flying along the landing path, however the cross track error was within the acceptance criteria listed in table 5.4 at the time the UAV passed the virtual net.

The longitudinal control system was able to follow it's desired height reference during the approach path, however the UAV height starts to diverge from the desired height during the landing path. The divergence corresponds to the overshot in the lateral plan, which could be the reason for the divergence. However the desired height from the longitudinal control system was higher then the net center height at the time the UAV passed the virtual net. The reason behind this behaviour is due to the reference model in the longitudinal control system attempt to create a smooth glide slope towards the aiming position behind the net, which cause the desired height to lag behind the desired path. This behaviour can be removed by setting the final approach angle to zero, thus ensuring that the straight line through the net center has a constant height value.

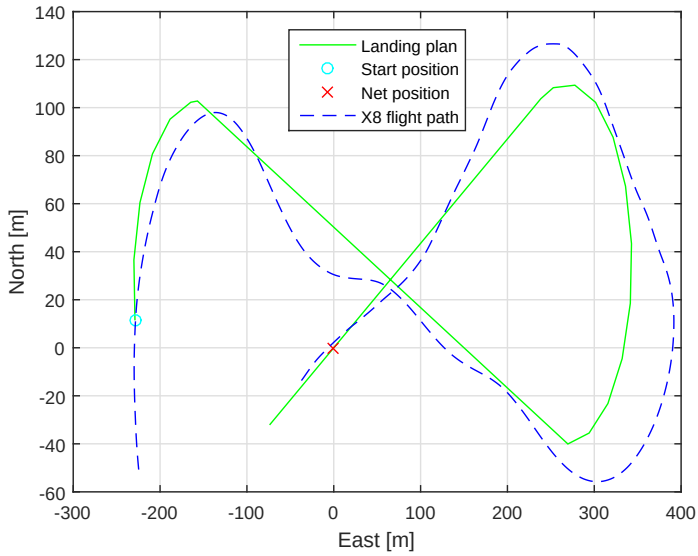

Figure 6.1: North-East plot where the start and finish turning circles have opposite turning directions

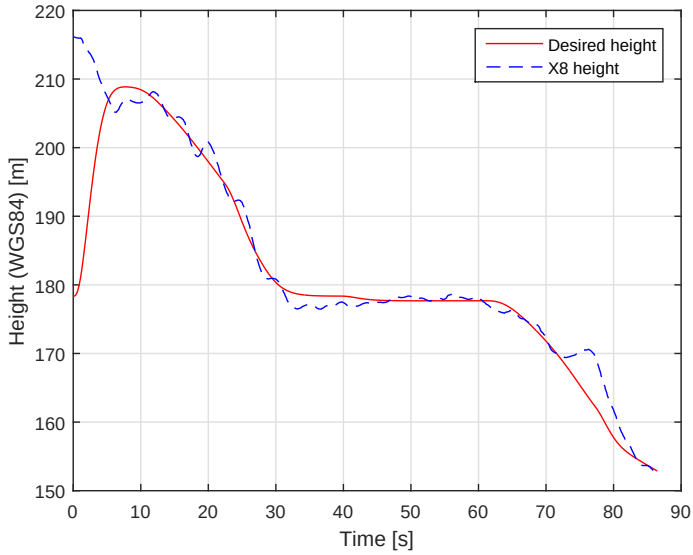

Figure 6.2: Height profile of a landing plan with 3 deg net impact angle

Path with $\gamma_n = 0$

A landing plan with the final approach angle ,$\gamma_n = 0$, was created, where the desired and actual height of the UAV is shown in figure6.3. The effect of setting the net impact angle to zero gave a better performance from the longitudinal control system, due to the desired height now converging to the net center height. At the time the UAV passed the virtual net the hight error with respect the height of the net center was within the height error acceptance.

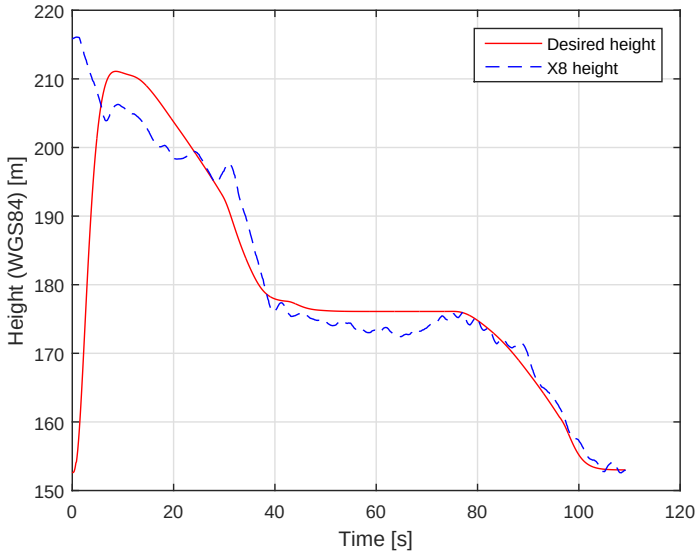

Figure 6.3: Height profile of a landing plan with 0 deg net impact angle

Path with inverted rotation direction in the start circle

A landing plan with the rotation direction of the start turning circle inverted with respect to the previous landing plan, as shown in figure 6.4. The result of this alteration was reduced oscillation in the lateral plane for the UAV when flying along the straight line between the turning circles. The UAV still experience overshot in the finish turning circle, however the overshooting is reduced due to more stable entry into the finish turning circle.

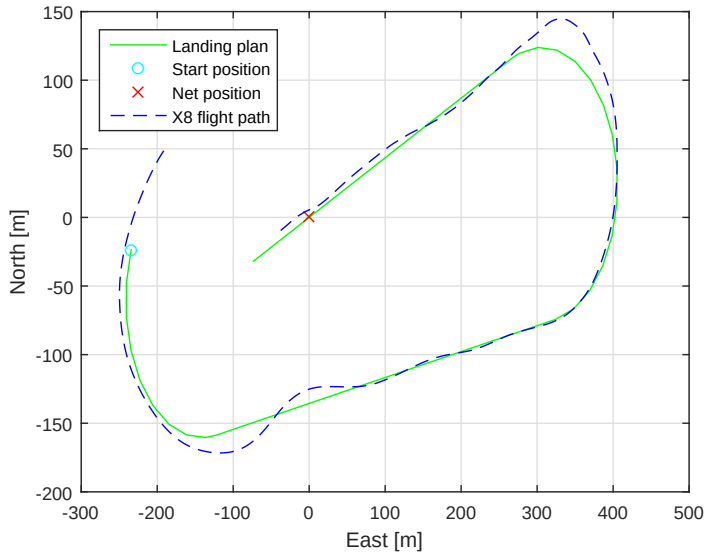

Figure 6.4: North-East plot where the start and finish turning direction have the same turning direction

Path with reduced lookahead distance in lateral controller

In order to further reduce the oscillatory motion in the lateral plan the lookahead distance in the lateral control system was reduced to make the controller more aggressive towards the wind. The effect of this change is shown in figure 6.5, where the oscillatory motion is almost completely removed. However the reduced lookahead distance did not affect the the overshot in the finish turning circle.

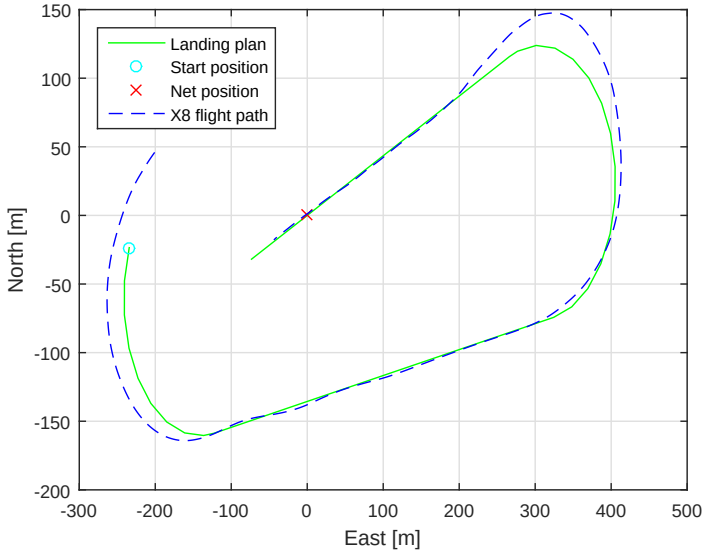

Figure 6.5: North-East plot where the lookahead distance of the lateral controller was reduced to increase performance of the autonomous landing system when flying against the wind

By reviewing the desired roll (ϕ_d) and the actual roll (ϕ) of the UAV at the time of the final turn, shown in figure 6.6, it's observed that the lateral control system decrease the desired roll in the middle of the turn. This happens since the lateral control system only sees the next point on the circle, and not the circle as a whole.

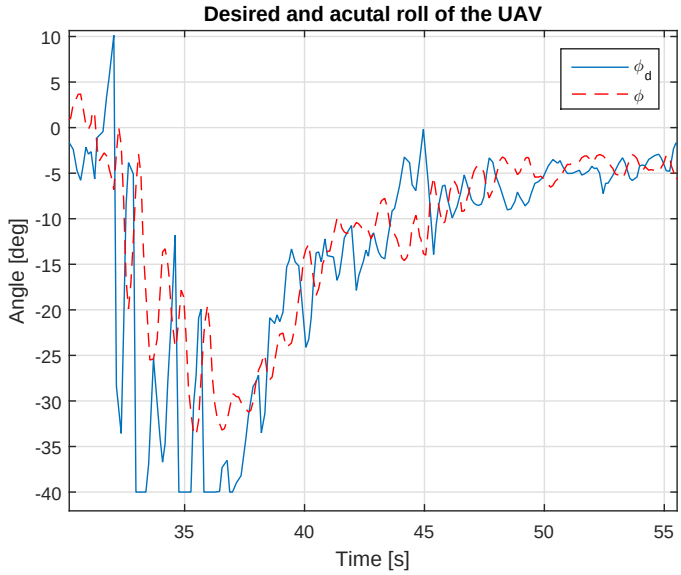

Figure 6.6: The desired roll and actual roll of the UAV during the finish turning circle

Summary of day 1

The result from the first day was affected with strong wind condition, in which the UAV struggled to stay on the path and overshooting in the finish turning circle. The heigh and cross track error for 11 landing plan missions, is given in table 6.1. The average height error vary less then the average cross track error, with a variance of $0.4m$ against $6.2m$. However the performance of the UAV in both height error and cross track error is worse then the results obtain during SIL simulation. This was expected, however the magnitude of the variance in the average cross track error shows that the performance of the lateral control system must increase in order for the autonomous landing system to be considered reliable.

Nr.	**Average height error [m]**	**Average cross track error [m]**
1	1.5	6.1
2	2.6	6.7
3	0.9	5.5
4	0.1	2.8
5	1.7	2.0
6	1.3	6.8
7	1, 8	9.1
8	1.2	8.2
9	1.9	5.9
10	1.5	4.4
11	1.5	1.4
Average	1.5	5.4
Variance	0.4	6.2

Table 6.1: Mean height and cross track error from day 1

The variance of the longitudinal control system shows that the control system is reliable in performance, however the average error must be reduced in order for the autonomous landing system to be able to hit the stationary net with increased probability of success. The results of whether the UAV was within the net acceptance criteria at the time of net passing, is given in table 6.2. Most of the alteration of the path and controller parameters was aimed towards the lateral control system, which is shown be be increased performance during the later attempts. In the comparison to the simulation results, where the average cross track error was almost zero, the performance has clearly decreased. This behaviour was expected due the simulation model used in the simulation has not been verified.

Nr.	**Height error [m]**	**Cross track error [m]**	**Height acceptance**	**Cross track error acceptance**	**Net hit**
1	2.8	2.1	X	OK	X
2	2.7	−4.5	X	X	X
3	0.9	−1.6	OK	OK	OK
4	0.0	5.4	OK	X	X
5	0.8	5.3	OK	X	X
6	2.1	−1.6	X	OK	X
7	0.7	2.3	OK	OK	OK
8	−1.5	−5.4	X	X	X
9	1.9	0.8	X	OK	X
10	0.3	1.1	OK	OK	OK
11	−1.3	0.2	OK	OK	OK

Table 6.2: Table containing the result of each landing attempt

The content of table 6.2 is shown in figure 6.7, where the net is marked as a whole line and all landing attempts are marked as crosses. The oscillatory motion in the lateral plane by the UAV is reflected in the placement of the crosses. However the placement of the cross shows the effect of an high average height error by either passing over the net or in the upper part of the net.

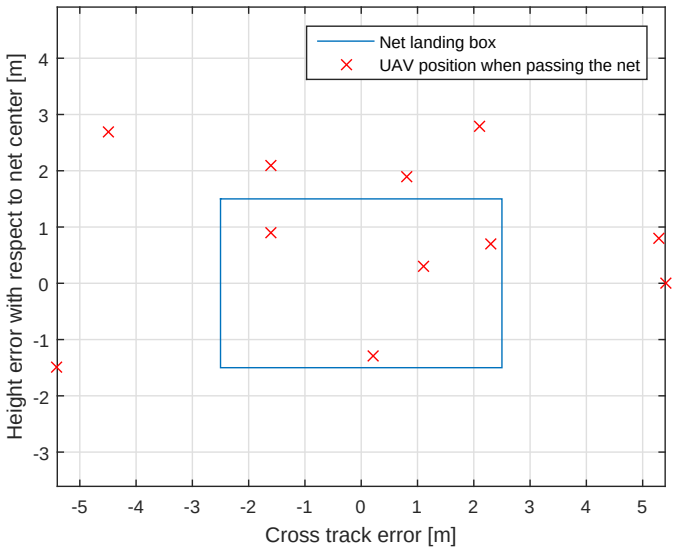

Figure 6.7: Position of UAV relative to the net center at the time of net passing

During the execution of the landing plan it was discovered that in order for the UAV to stay above the highest tree tops it had to start its decent from an height of $56m$ above the airfield. Flying bellow this height would result in the pilot losing sight of the UAV, which is unacceptable in a LOS UAV operation. This didn't provide a problem during landing attempt with a virtual net which was placed $26m$ above the airfield, however during a real stationary net landing it would push the limitation of the operation area in which the UAV can fly. A $56m$ decent with a glide slope angle of 6 deg would require a glide slope of $500m$. An estimate of the the available length of which the UAV can use to perform a autonomous landing in a stationary net is estimated to be $700m$. This would require the use of the entire runway at Agdenes. An alternative solution is to attempt to land with a greater glide slope angle, or simply start the landing path from the west with respect to the runway. Landing attempts from the west has yet to tried, the reason being limited time and that would mean that the UAV would have to land in tail wind. This would increase the ground speed of the UAV, which is undesired. In the case where the wind is calm enough to be considered negligible, a autonomous landing from west would be a valid option.

6.2.2 Day 2

First plan

The second day had calm wind condition, which is considered as ideal field test conditions for the autonomous landing system. In an attempt to increase the height difference between the start height of the landing path and the net center, a longer glide slope was attempted. Among the alteration was moving the virtual net further to the west with respect to the runway, and the length and angle of the glide slope was increase to $280m$ and 8 deg respectfully. The full path specification for the first path is given in appendix C.3. Figure 6.8 shows a North-East plot of the path created with the new specification, which show the lateral path overshoots both the start and finish turning circles. The overshot in the start circle is a result of the roll angle is not able to follow the desired roll angle fast enough, as seen in figure 6.9, which is due to the low level roll controller being poorly tuned for autonomous flights where high precision performance from the UAV is desired. In addition the control surface used to control the roll of the UAV is also used to control the pitch, which results in having to weight the performance in heading against the ability to follow a height reference.

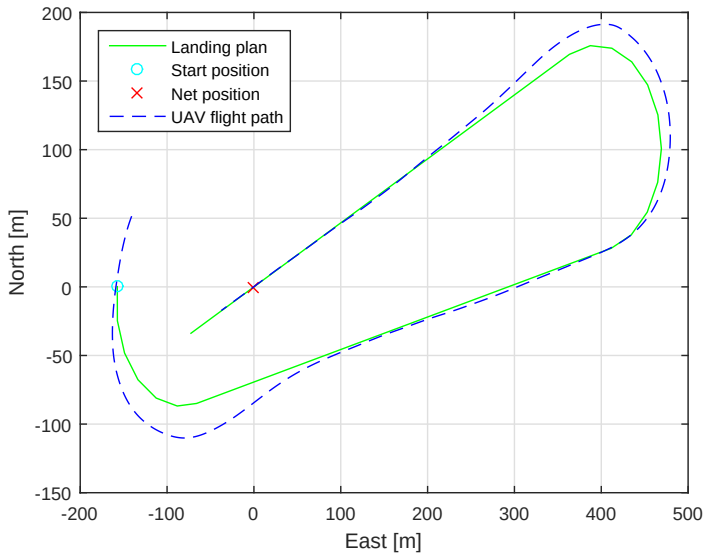

Figure 6.8: North-East plot created with the path parameter listed in table C.3

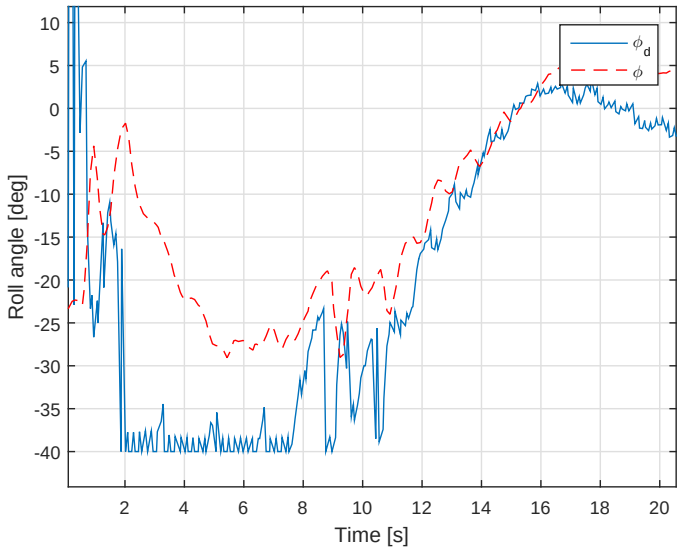

Figure 6.9: Desired and actual roll angle during the start turning circle

The desired height and the actual heigh of the UAV is shown in figure 6.10, which shows that the UAV is unable to follow a glide slope with a slope angle of $\gamma_l = 8\,\text{deg}$. A key reason for this behaviour is due to the inability of the UAVs low level pitch controller to follow the desired pitch, as shown in figure 6.11. The pitch appear to be saturated in the low level controller since it does not attempt to reach the desired pitch. The low level pitch controller must be further fine tuned, and similar to the low level roll controller, has not been fine tuned for autonomous flights where high precision performance is desired.

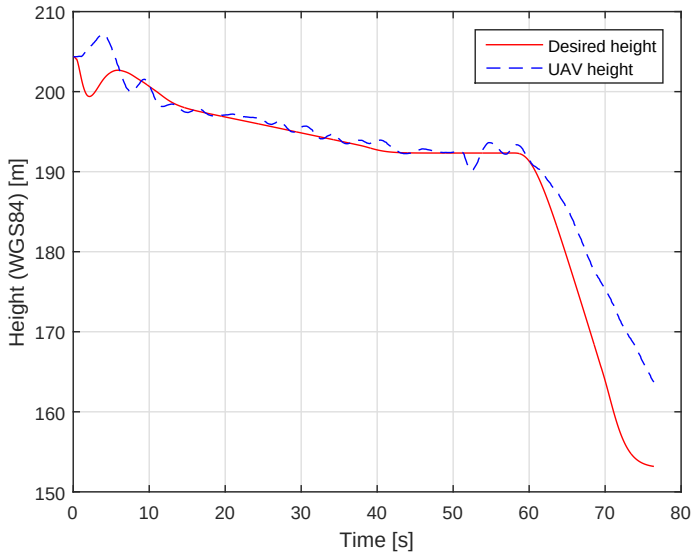

Figure 6.10: Desired and actual height of the UAV with a glide slope angle of $\gamma_l = 8\,\text{deg}$

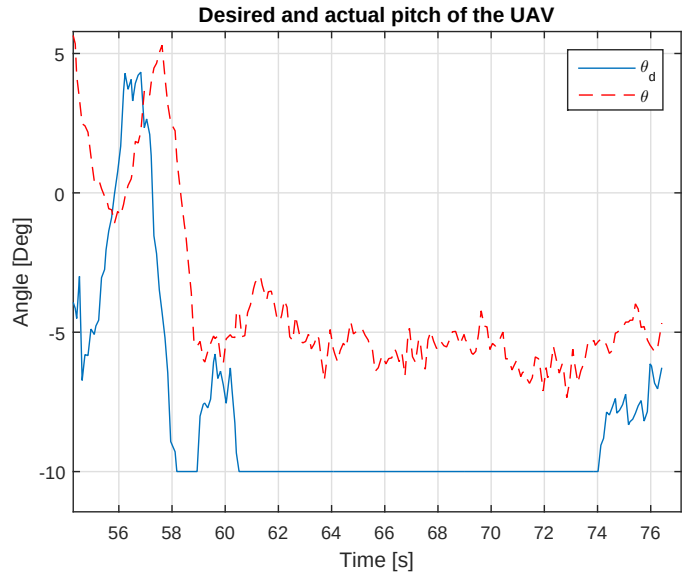

Figure 6.11: Desired θ_d and actual pitch θ, of the UAV at the time the UAV is in the glide slope

Reducing arc segments distance

In an attempt to further reduce the overshot in the finish turning circle the distance between each arc segments in the approach path circles was reduced from $25m$ to $10m$. The desired response with this alteration of the approach path was to ensure that the lateral control system keeps a high desired roll angle through the turning circle, thus reducing the overshot and increase the performance. A North-East plot of the resulting path is shown in figure 6.12, which shows that the UAV has reduced its overshot in the final turning circle. The desired behaviour where the lateral control system keep a high desired roll angle through is achieved and shown in figure 6.13. The confirmation that the UAV had a small overshot from the desired path is shown in the cross track error plot given in figure 6.14.

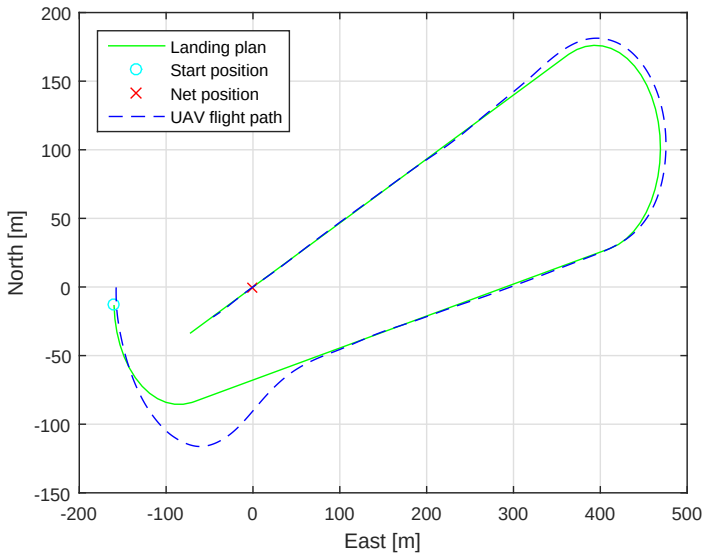

Figure 6.12: North-East plot where the distance between each arc segments has been reduced from $25m$ to $10m$

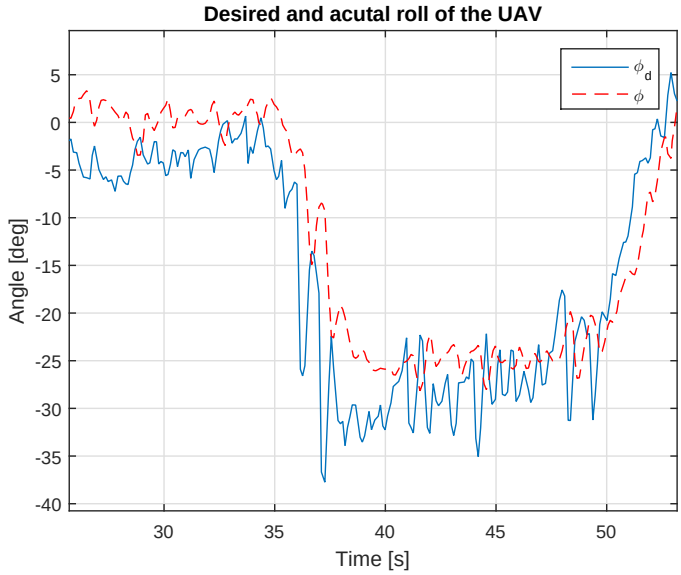

Figure 6.13: Roll and desired roll in the final turning circle

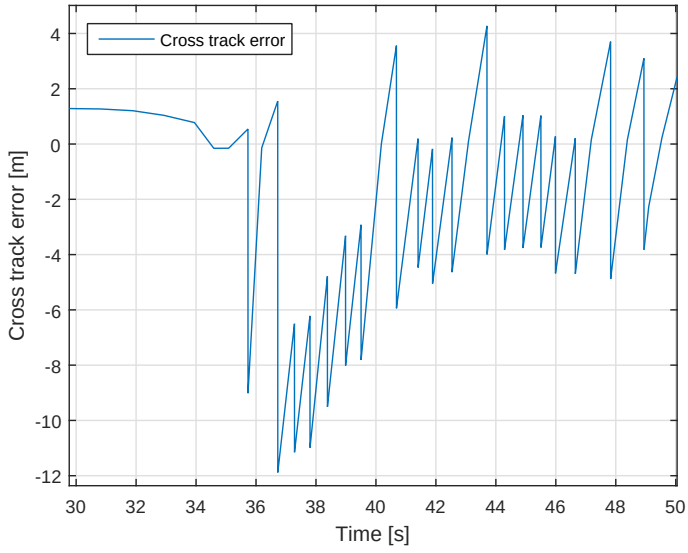

Figure 6.14: Cross track error in the final turning circle

Summary day 2

The weather condition during the second day can be considered ideal for testing the autonomous landing system, thus possible to identify the strength and weaknesses of the system where the weather affect is negligible. The goal with the second day was to increase the height from which the UAV would start its decent towards the net by increasing the glide slope angle, and to investigate path parameter that can be used to reduce the UAVs overshot during turning in the final turning circle. The attempts to increase the glide slope angle during landing plan mission are reflected in table 6.3, which contain the results of whether the UAV passed through the net or not. Attempts with glide slope angles greater then 6 deg resulted in the UAV to miss the net, however this performance could be increased with a better better tuning of the low level pitch controllers. With the current longitudinal control system the maximum glide slope angle which the X8 is able to follow during a autonomous landing is 6 deg. In addition increased performance from the longitudinal control system can be achieved by increasing the length of the final approach, in order for the UAV to have longer time to converge to the net center height. The final approach length could be increased to $120m$ by moving the net to the edge of the runway at Agdenes. The risk with a large final approach during a autonomous landing in a real net would be the prolonged time the UAV flies $3 - 4m$ above ground. A loss of high accurate positioning system could then result in a crash.

Nr.	**Height error [m]**	**Cross track error [m]**	**Height acceptance**	**Cross track error acceptance**	**Net hit**
1	14.4	0.1	X	OK	X
2	1.3	0.6	OK	OK	OK
3	1.1	−0.2	OK	OK	OK
4	1.4	0.1	OK	OK	OK
5	1.1	0.1	OK	OK	OK
6	2.0	−0.2	X	OK	X
7	2.3	0.2	X	OK	X
8	7.0	0.3	X	OK	X

Table 6.3: Table containing the result of each landing attempt

The lateral control system perform better compared to the first day during calm wind condition which is reflected in the table 6.4, where the performance from 8 landing plan missions is presented. The performance from the longitudinal control system remain similar to the first day, which shows that the longitudinal control

system is less affected by the wind then the lateral control system, however the the average height error of the longitudinal control system must be decreased in order for the autonomous landing to achieve a higher probability of successfully performing a successful net landing. The high average height error of the longitudinal control system is reflected in figure 6.15, where those crosses that are within the height acceptance criteria are in the upper part of the net.

Nr.	**Average height error [m]**	**Average cross track error [m]**
1	2.2	3.8
2	1.2	3.4
3	0.9	−1.8
4	2.5	−0.2
5	3.0	0.3
6	1.6	0.2
7	1, 9	−2.3
8	1.9	−0.1
Average	1.9	0.5
Variance	0.5	4.7

Table 6.4: Average height and cross track error from day 2

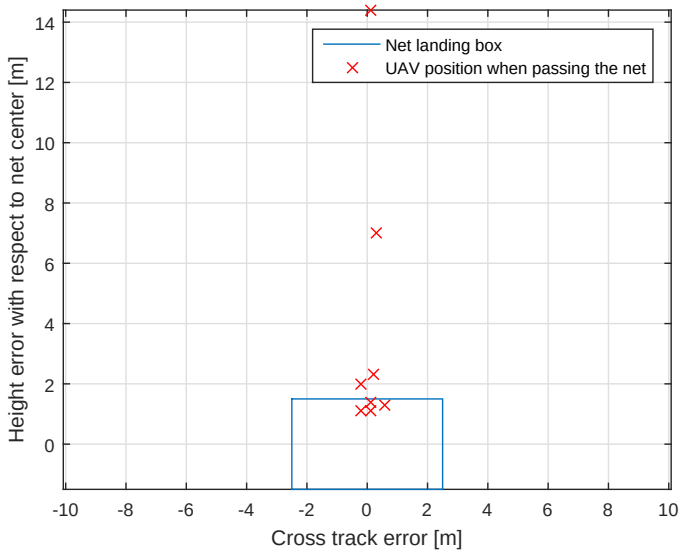

Figure 6.15: Position of the UAV relative to the net center at the time of net passing

6.3 Navigation

6.3.1 RTK-GNSS performance

The performance from the RTK-GNSS system during both day of testing the autonomous landing system resulted in similar performance, thus only results from the first day is presented in this section, while the results from the second day is given in appendix D. The results from the RTK-GNSS system during the landing plan flights is summarised in table 6.5, where the result is presented as percentage of the total number of GpsFixRtk messages.

Nr.	FIX %	FLOAT %	NONE %
1	99.5	0.5	0.0
2	99.5	0.5	0.0
3	99.8	0.0	0.0
4	100	0.0	0.0
5	100	0.0	0.0
6	100	0.0	0.0
7	99.9	0.1	0.0
8	99.7	0.3	0.0
9	99.3	0.7	0.0
10	100	0.0	0.0
11	100	0.0	0.0

Table 6.5: Performance of the RKT-GNSS system the first day during the executing of the landing plans

6.3.2 Short loss compensator

The short loss compensator was engaged during a flight where the RTK-GNSS started to experience problem. The flight plan was part of cooperative net recovery system presented in [Nevstad, 2016] and [Moe, 2016], which experience reduced RTK-GNSS performance due to decreased satellite geometry. During the flight the RTK-GNSS system experienced a drop out, as seen in figure 6.16. The main reason for the drop out is shown in figure 6.18, where at the time of RTK-GNSS drop out the number of valid satellites starts to vary rapidly. Even though the RTK-GPS position solution is unavailable the navigation system is still able to supply high accurate position solution due to the short RTK-GNSS compensator as seen in figure 6.17.

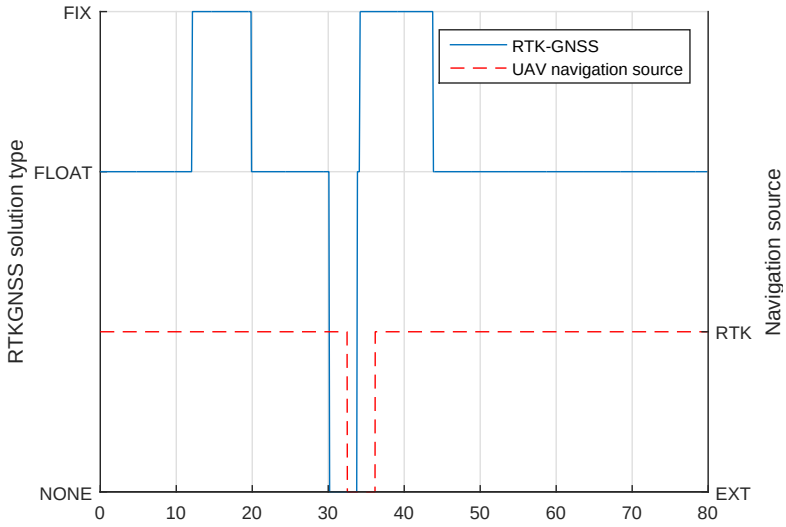

Figure 6.16: State of RTK-GNSS system and UAV navigation source.

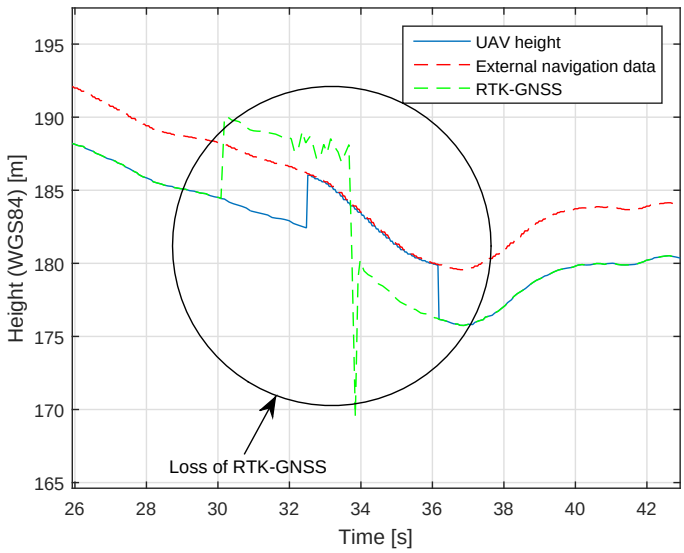

Figure 6.17: Loss of RTK-GPS triggers the short loss compensator such that the UAV keeps the RTK-GPS position solution longer

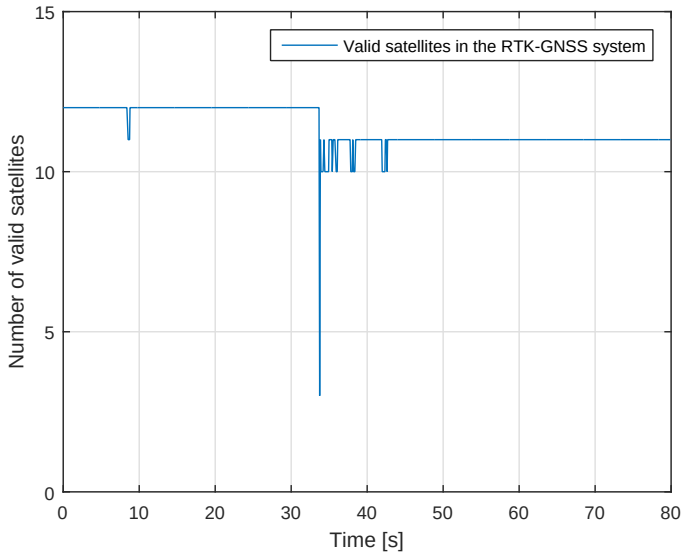

Figure 6.18: Number of valid satellites during a flight mission

6.3.3 Summary of navigation system performance

The navigation system was able to provide high accuracy position solution to the UAV during the autonomous landing flight plan, however continues drop out of the RTK-GNSS system was experienced when the satellite geometry decreased, which make the positioning system more susceptible to atmospheric disturbance. The continues drop out could be reduced by increasing the elevation mask in RTKlib, however this would limit the valid satellites to a number where a FIX solution from the RTK-GNSS system is no longer available. In order for the system to be able to perform during a prolong duration of degraded satellite geometry a state estimator must be created which can give a accurate position estimate with minimum RTK-GNSS information available. This would be a more advance estimator compared to the short loss compensator presented in this thesis.

The short loss compensator was able to compensate the solution from the external navigation system sufficiently to avoid a large step in UAV position data for the short period where the RTK-GNSS data was not available. However the limits of the short loss compensator has yet to determined, and should be further tested. The goal of the navigation system is be to be able to provide high accurate position solution to the UAV, even if the RTK-GNSS system experience a drop out.

6.4 Summary

This chapter has presented the result from the experimental testing of the autonomous landing system for a X8 fixed-wing UAV over the course of two days with different weather, where the landing target was a virtual net placed $26m$ above the runway. The different weather condition allowed for identification of strength and weaknesses with the control system, in addition to how alteration of landing plan parameters could affect the performance of the control system.

The longitudinal control system showed a stable performance, however a large average error in the height with respect to the desired height shows a performance that is not acceptable for a autonomous landing system. Compared to the results obtain in the SIL simulation 5.3.4 where the low level controllers are fined tuned, shows that the performance can be increased by further tuning of the low level pitch controller. This might allow for an increased glide slope angle, which will increase the height difference between the landing net and the start of the landing path or reduce the necessary glide slope length. However the risk with an increased glide slope angle is an increased airspeed, which could result in a large overshot with respect to the desired height.

The lateral control system performed with satisfying result during the second day when the wind condition was calm, however during the first day it experienced problems when attempting to converge to the straight line between two waypoints. This result differ from the SIL simulation of the autonomous landing system, which was expected due to the mathematical model used to represent the X8 UAV has not been verified against a physical model of the X8, in addition to the low level controllers not being fined tuned for autonomous flights. The lateral control system struggeled to avoid overshooting when following the finish turning circle, with some resulting overshot large enough to bring the UAV to the edged of the operational flight space. During the flight experiments some key factors was identified which resulted in increased performance from the lateral control system. The key factors are listed in table 6.6.

- Reducing the lookahead distance when flying in strong wind conditions.
- Creation of a approach path with the same rotation direction in both turning circles.
- Reducing the distance between each arc segments in the turning circles.

Table 6.6: Alteration in the autonomous landing system which resulted in increased performance from the lateral controller

The reduction of the lookahead distance will result in increased performance when against the wind, at the cost of reduced performance when flying in the tail wind. A possible solution to this problem would be to implement the the lookahead distance as a function of the cross track error, which in [Fossen, 2011] section 10.3.2 is given as:

$$\Delta(t) = \sqrt{R^2 - e(t)^2} \tag{6.1}$$

where $\Delta(t)$ is the lookahead distance, R is the maximum lookahead distance and $e(t)$ is the cross track error. In addition the lateral control system functionality could be expanded to increase the performance during a turning manoeuvre, with the goal of reducing the overshot.

Experimentation with the landing plan parameter showed that both the lateral and longitudinal control system was affected by the choice of parameters. During the first day the rotation direction of the start turning circle brought the UAV into the cross wind, which caused oscillatory motion in the UAV. Thus the combination of rotation direction should be considered carefully, with the recommended combination being either counter-clockwise/counter-clockwise or clockwise/clockwise. These combinations will result a smoother transition between the two turning circles, hence reduce the oscillation in the lateral controller. In addition the final approach angle must be zero, in order for the longitudinal control system to output a desire height reference which equals the net center height. Other key landing plan factors with recommended values for increase autonomous landing system performance are listed in table 6.7.

Parameter name	**Recommended value**
Time of arrival factor	$2s$
Distance between arc segments	$10m$
Final approach length	$100m$
Final approach angle	0 deg
Glide slope angle	6 deg

Table 6.7: Recommended parameter alteration to the landing plan with respect to the plan parameter given in table C.3, and the task configuration parameter given in table 5.3

The minimum height from which the autonomous landing system could start the landing path from was found to be $56m$ above the runway at Agdenes airfield when attempting to land from the east. This strain the operation boundaries in which the UAV operates, since the approach could move the UAV out of the line of sight of the pilot. An alternative approach from the west is possible, however the typical wind

condition on Agdenes is from the west, which would limit the weather window of the autonomous landing system.

The navigation system performed with satisfying results, and showed that it's able to provide a stable and reliable RTK-GNSS solution to the navigation system. However the RTK-GNSS is still prone to bad satellite geometry, which will result in the RTK-GNSS system to loose lock of the satellites. During these events the short RTK-GNSS loss compensator is able to compensate the external navigation system, such that the navigation position solution remain at the same accuracy level as with RTK-GNSS. The time limitation in which the compensator is able to operate without divergence from the RTK-GNSS accuracy level has yet to determined.

Chapter 7

Conclusion and recommendation for further work

7.1 Conclusion

The development of a landing plan generator and navigation system in Ch. 3 resulted in a system capable for a autonomous landing of a fixed wing UAV in a stationary net. The landing plan generator presented can create a flyable path towards the net from any initial position, where all parameters of the plan can be specified to suit the users needs. The separation of the landing plan into two parts allowed for usage of the landing plan generator in other system that require a fixed wing UAV to be at a specific position with a given heading. The implementation of the landing plan generator presented in Ch. 5 showed that the landing path generator can be configure through the use of the API in the form of the IMC message LandingPlanGeneration, which ensures that the parameters used to create a landing plan is available in the fixed structure of the IMC protocol. Further the Software In the Loop (SIL) simulation of the autonomous landing system presented in section 5.3 showed the the presented system is capable of performing a autonomous landing in a stationary net with a landing plan created from a arbitrary location. The testing of the control system used in the autonomous landing system showed that during a simulation the control system is able to pass through the net with acceptable precision. In Ch. 5 the necessary navigation systems need to perform a autonomous landing operation was presented. The RTK-GNSS system is able to provide high accurate position solution of the UAV relative to the base station position, with the mobile sensor unit providing a reference position for net placement. The navigation source monitor in Neptus provide the operator with feedback on the state of the UAV navigation system, which enable the operator to make informative decisions during a autonomous landing in a stationary net operation. The results from the experimental testing of

the autonomous landing system presented in Ch. 6, shows that the lateral control system is capable of performing a autonomous landing during strong wind condition, given that the lateral control system is tuned for the current wind condition. A proposed strategy for tuning the lookahead distance of the lateral control system include a lookahead distance which is a function of the cross track error of the desired path. The longitudinal control system had a stable performance during the experimental testing, which resulted in a low variance in the average height errors. However the high average height error results from the autonomous landing missions showed that longitudinal control system was slow to converge to the desired height. A tuning attempt performed in the the SIL simulation in section 5.3, where the time constant for the height reference smoothing filter was reduced, showed promising results. However due to time limitation the new time constant has not been tested in the field, and together with further tuning of the low level pitch controller in the UAV would result in a performance closer to that in the SIL simulation. The performance of the navigation system presented in Ch. 6 shows that the navigation system with RTK-GNSS is reliable enough to be used during a autonomous landing operation. The RTK-GNSS is still prone to drop out due to shift in satellite geometry, however with the introduction of the short RTK-GNSS compensator a short loss of RTK-GNSS is handled in a way that keeps RTK-GNSS availability and position accuracy. The short RTK-GNSS loss compensator was introduced in section 3.2.3 as a method of exploiting the slow moving difference between the RTK-GNSS and the external navigation system by fusing navigation data from the RTK-GNSS and the secondary GNSS system together in a compensating term for the external navigation system. The introduction of the mobile sensor unit in section 5.1.3 has provide a reference position for net placement in Neptus, in order for correct placement of the landing path. The introduction of a mobile sensor that act as a reference position for net placement allows for increase operational control over a autonomous landing operation, thus reducing the risk of a undesired landing plan being created due to wrong placement of the net. During the experimental field test in Ch. 6 at Agdenes airfield limitation in the start height of the landing path when attempting a landing from East was discovered. Due to a surrounding environment which demand a high start altitude, a landing plan created with the purpose of landing in a real stationary net would strain the operational restriction of flying a Line Of Sight operation. In the event a autonomous landing from East is attempted with the current control system the entire runway at Agdenes must be used to ensure a long enough glide slope for sufficient altitude decrease. Alternative landing direction from the west has the disadvantage of being the typical direction of the wind, which is undesired when attempting a autonomous landing due to increased ground speed. All thought during calm wind periods this would not prove a problem, thus becoming an attractive alternative due to the allowance of a lower start altitude of the landing path.

Due to limited time for experimental testing the full potential of the autonomous

landing system has yet to be determined, together with a landing attempt in a real stationary net. The overall performance of the autonomous landing system has shown that the system is capable of performing a autonomous landing in a stationary net.

7.2 Recommendation for further work

This master thesis has presented a autonomous landing system, with the main focus on the path and navigation system. The most important focus for further work is the tuning of the low level controllers in the X8 for autonomous flights. An important aspect here it that the tuning of the low level controller is only used when the pilot is not manually controlling the X8. Thus the control system must be able to switch tuning parameters depending on the mode of Ardupilot. Further work on the lateral controller would be a lookahead distance parameter which is a function of the cross track error relative to the desired lateral path. In addition the lateral control system could be improved to better follow a path through a turning manoeuvre, with the main motivation of reducing overshot. In order to shorten the distance of the landing path, a new type of longitudinal control system can be investigated with the main motivation being to use the high dynamic behaviour of the X8 to more effective decrease the altitude. The new control design would use a higher attack angle, however due to the increase stress on the wings the controller must be time dependent to prevent a crash. The UAV navigation system can be improved by expanding the functionality of the short RTK-GNSS loss compensator, with the goal of creating a more robust RTK-GNSS system. The improvement could be the use of the velocity information from the external navigation system together with the compensator term to further increase the duration where the short RTK-GNSS loss compensator is valid. The functionality of the mobile sensor unit can be expanded to enable the fixed wing UAV to apply target tracking of the position of the sensor unit. This could be used in a autonomous net landing system where the position of the net is dynamic, without the use of multirotor UAVs. This can be used in a autonomous landing system where the net is placed on a ship, of which the DUNE system has not the option to control.

The autonomous landing system in a stationary net require a monitor to follow the fixed wing UAV along the landing path and detect when the UAV hit the net, or be used to trigger an abortion. In the case of abortion the autonomous landing system must include a evasive manoeuvre, however further testing of the current landing system is required to find the boundaries of when an abortion should be triggered.

References

Ubiquiti networks rocket m powerful 2x2 mimo airmax basestation models: M5, rm5-ti, m3, m365, m2, rm2-ti, m900 datasheet. https://dl.ubnt.com/datasheets/rocketm/RocketM_DS.pdf. Accessed: 16.05.2016.

D Blake Barber, Stephen R Griffiths, Timothy W McLain, and Randal W Beard. Autonomous landing of miniature aerial vehicles. *Journal of Aerospace Computing, Information, and Communication*, 4(5):770–784, 2007.

Brian A Barsky and Tony D DeRose. Geometric continuity of parametric curves: three equivalent characterizations. *IEEE Computer Graphics and Applications*, (6):60–68, 1989.

Randal W Beard and Timothy W McLain. *Small unmanned aircraft: Theory and practice*. Princeton university press, 2012.

Jon S Berndt. Jsbsim: An open source flight dynamics model. In *in C++. AIAA*. Citeseer, 2004.

Lester E Dubins. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of mathematics*, 79(3):497–516, 1957.

Joao Fortuna and Thor I Fossen. Cascaded line-of-sight path-following and sliding mode controllers for fixed-wing uavs. In *Control Applications (CCA), 2015 IEEE Conference on*, pages 798–803. IEEE, 2015.

Thor I Fossen. *Handbook of marine craft hydrodynamics and motion control*. John Wiley & Sons, 2011.

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

Kristoffer Gryte. High angle of attack landing of an unmanned aerial vehicle. Master's thesis, Norwegian University of Science and Technology, NTNU, 2015.

Sungsik Huh and David Hyunchul Shim. A vision-based automatic landing method for fixed-wing uavs. *Journal of Intelligent and Robotic Systems*, 57(1-4): 217–231, 2010.

Insitu. Skyhook universal productcard. https://insitu.com/images/uploads/pdfs/Skyhook_Universal_ProductCard_PR041615.pdf. Accessed: 13.05.2016.

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

Ricardo Martins, Paulo Sousa Dias, Eduardo RB Marques, José Pinto, Joao B Sousa, and Femando L Pereira. Imc: A communication protocol for networked vehicles and sensors. In *Oceans 2009-Europe*, pages 1–6. IEEE, 2009.

Maxtena. M1227hct-a-sma l1/l2 gps-glonass active antenna, data sheet. http://www.farnell.com/datasheets/1681376.pdf. Accessed: 18.12.2015.

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

Jostein Borgen Moe. Autonomous landing of fixed-wing uav in net suspended by multirotor uavs - a multirotor recovery system. Master's thesis, Norwegian University of Science and Technology, NTNU, 2016.

Sigurd Olav Nevstad. Autonomous landing of fixed-wing uav in net suspended by multirotor uavs - a fixed-wing landing system. Master's thesis, Norwegian University of Science and Technology, NTNU, 2016.

Novatel. Gps-701-gg and gps-702-gg, data sheet. www.novatel.com/assets/Documents/Papers/GPS701_702GG.pdf. Accessed: 18.12.2015.

Joel Pinto, Paulo S Dias, Rui P Martins, Joao Fortuna, Eduardo Marques, and Joao Sousa. The lsts toolchain for networked vehicle systems. In *OCEANS-Bergen, 2013 MTS/IEEE*, pages 1–9. IEEE, 2013.

RTKLIB. Rtklib ver. 2.4.2 manual. http://www.rtklib.com/prog/manual_2.4.2.pdf. Accessed: 18.12.2015.

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

Robert Skulstad, Christoffer Lie Syversen, Mariann Merz, Nadezda Sokolova, Thor I Fossen, and Tor A Johansen. Net recovery of uav with single-frequency rtk gps. In *Aerospace Conference, 2015 IEEE*, pages 1–10. IEEE, 2015.

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

Tomoji Takasu and Akio Yasuda. Development of the low-cost rtk-gps receiver with an open source program package rtklib. In *international symposium on GPS/GNSS*, pages 4–6. International Convention Centre Jeju, Korea, 2009.

Peter JG Teunissen. A new method for fast carrier phase ambiguity estimation. In *Position Location and Navigation Symposium, 1994., IEEE*, pages 562–573. IEEE, 1994.

Peter JG Teunissen. The least-squares ambiguity decorrelation adjustment: a method for fast gps integer ambiguity estimation. *Journal of Geodesy*, 70(1-2): 65–82, 1995.

Antonios Tsourdos, Brian White, and Madhavan Shanmugavel. *Cooperative path planning of unmanned aerial vehicles*, volume 32. John Wiley & Sons, 2010.

U-blox. Neo/lea-m8t u-blox m8 concurrent gnss timing modules, data sheet. https://www.u-blox.com/sites/default/files/NEO-LEA-M8T_DataSheet_(UBX-14006196).pdf, a. Accessed: 18.12.2015.

U-blox. U-blox m8 receiver description including protocol specification. https://www.u-blox.com/sites/default/files/products/documents/u-bloxM8_ReceiverDescrProtSpec_%28UBX-13003221%29_Public.pdf?utm_source=en%2Fimages%2Fdownloads%2FProduct_Docs%2Fu-bloxM8_ReceiverDescriptionProtocolSpec_%28UBX-13003221%29_Public.pdf, b. Accessed: 18.12.2015.

Bjørnar Vik. Integrated satellite and inertial navigation systems. *Department of Engineering Cybernetics, NTNU*, 2014.

Artur Zolich, Tor Arne Johansen, Krzysztof Cisek, and Kristian Klausen. Unmanned aerial system architecture for maritime missions. design & hardware description. In *2015 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS)*, pages 342–350. IEEE, 2015.

Appendix A

Landing plan generation API

The landing plan generator API consist of the IMC LandingPlanGeneration, which is given in table the field with description is given in table A.1.

Field name	**Type**	**Description**
Command	Enumerated	Command the plan database to generate the plan, with the option of executing the plan after generation.
Operation	Enumerated	Type of operation started.
Plan identifier	Plain text	Plan identifier.
Reference latitude	rad	Reference latitude for the landing path.
Reference longitude	rad	Reference longitude for the landing path.
Reference height	m	Reference height for the landing path.
Height over ground	m	Offset from the reference height to placement.
Reference point heading	rad	Heading of the reference point in NED frame.
Distance behind	m	Aiming point behind the reference point.
Final approach length	m	The length of the final approach towards the reference point.
Final approach angle	rad	The decent angle of the final approach vector.
Glide slope length	m	The length of the glide slope in the landing path.
Glide slope angel	rad	The decent angle of the glide slope.
Approach decent angle	rad	The decent angle of the approach path.
Approach length	m	The length of the vector from the end of the approach path to the glide slope.
Landing speed	m/s	The landing speed.
Approach speed	m/s	The approach speed.
Start turning circle radius	m	The radius of the first turning circle.
Finish turning circle radius	m	The radius of the second turning circle.
Automatic generate landing plan	Flag	The approach path will calculate the shortest path form the initial position towards the start of the landing path.
Start circle turning direction	Flag	Manually setting the rotation direction of the first circle.
Finish circle turning direction	Flag	Manually setting the rotation direction of the finish circle.
Wait at loiter	Flag	Setting if after the approach path the UAV should enter a loiter manoeuvre.

Table A.1: The landing plan generator API

Appendix B

Guidance and control system

The autonomous landing system is design to be independent from any types of guidance and control system. However this hold only true when the guidance and control system is implemented in the DUNE environment due to the requirement of an accurate navigation system.

B.1 Lateral controller

The lateral controller used in the autonomous landing system is based on the paper [Fortuna and Fossen, 2015], which is a line of sight path following and sliding mode controller. The controller is designed to converge to the straight line between two waypoints, with the ability to compensate the wind disturbance in order to stay on the path. Depending on the line of sight of the controller, the controller either react aggressive towards wind disturbance or allow the wind to guide the UAV. The lookahead distance is a constant parameter, which can be configured from either the configuration file in DUNE or Neptus. The lateral controller is implemented in the DUNE task LOSnSMC, however a slightly altered version was used in the autonomous landing field experiment, which was the DUNE task LOSnSMCuser. The configuration parameter used in the autonomous landing field test are given in table B.1

Parameter	Value
Lookahead	50.0
Rho	1.0
Lambda	0.35
Kd	1.5
Bandwidth	3.0
Roll Time Const	0.5
Maximum Bank	40.0

Table B.1: LOSnSMCuser configuration parameters used in the hardware setup

B.2 Longitudinal controller

The longitudinal control system used in the autonomous landing system is designed and implemented as part of the master thesis [Nevstad, 2016]. The controller is design to create glide slopes between waypoint, where a 3rd low pass reference model is used make the glide slope between two way point smooth enough to become flyable. The reference model create outputs a desired hight which lay above the desired path due to lag introduced from the reference model. When the desired height glide slope will reach a constant height is dependent on the time constant used in the reference model, in addition to the time of arrival factor in DUNE. The longitudinal control system is implemented in the DUNE tasks Longitudinal and HeightGlideslope, with the configuration parameter given in table B.2 and B.3 for the Longitudinal and HeightGlideslope respectfully.

Parameter	Value
Throttle Proportional gain	0.3
Throttle Integrator gain	0.1
Throttle Proportional height gain	0.3
Gamma Proportional gain	1.0
Trim throttle	44.0

Table B.2: Longitudinal configuration parameters used in the hardware setup

Parameter	**Value**
LOS Proportional gain up	0.9
LOS Integral gain up	0.1
LOS Radius up	14
LOS Proportional gain down	0.9
LOS Integral gain down	0.1
LOS Radius down	14
LOS Proportional gain line	1.0
LOS Integral gain line	0.02
LOS Radius line	25
Time constant refmodelZ	1.0
Time constant refmodelGamma	1.0
Use reference model	True
Height bandwidth	10
Vertical Rate maximum gain	0.3

Table B.3: HeightGlideslope configuration parameters used in the hardware setup

Appendix C

Landing plan parameters

The landing plan specification for both the simulation and the start parameter for the two field experiment is presented in this appendix chapter.

C.1 Path parameter used in the SIL simulation

Table C.1 includes the landing plan parameter used to create the path shown in section 5.3.

Parameter	**Value**
Net latitude	63.6281111085521 deg
Net longitude	9.724609316783464 deg
Net height	150m
Net height offset	3m
Net heading	65.0 deg
Distance behind net	80m
Final approach length	100m
Final approach angle	0 deg
Glide slope length	300m
Glide slope angle	6 deg
Glide slope approach	40m
Landing speed	16m/s
Approach speed	18m/s
Turn radius first circle	75m
Turn radius final circle	75m
Start turning direction	Counter-Clockwise
Finish turning direction	Counter-Clockwise

Table C.1: Landing plan parameter used in the SITL simulation

sss:Day1FirstPlan

C.2 Start path parameter used the first flight day

Table C.2 includes the landing plan parameter used to create the path shown in section 6.2.1.

Parameter	**Value**
Net latitude	63.62852832784504 deg
Net longitude	9.726611753451145 deg
Net height	150m
Net height offset	3m
Net heading	66.5 deg
Distance behind net	80m
Final approach length	80m
Final approach angle	3 deg
Glide slope length	220m
Glide slope angle	6 deg
Glide slope approach	10m
Landing speed	16m/s
Approach speed	18m/s
Turn radius first circle	75m
Turn radius final circle	75m
Start turning direction	Clockwise
Finish turning direction	Counter-Clockwise

Table C.2: Landing plan parameter used at the start of first day of field experiments

C.3 Start path parameter used the second flight day

Table C.3 includes the landing plan parameter used to create the path shown in section 6.2.2.

Parameter	**Value**
Net latitude	63.62831245876848 deg
Net longitude	9.72534155984453 deg
Net height	$150m$
Net height offset	$3m$
Net heading	65.0 deg
Distance behind net	$80m$
Final approach length	$80m$
Final approach angle	0 deg
Glide slope length	$280m$
Glide slope angle	8 deg
Glide slope approach	$40m$
Landing speed	$16m/s$
Approach speed	$18m/s$
Turn radius first circle	$75m$
Turn radius final circle	$75m$
Start turning direction	Counter-Clockwise
Finish turning direction	Counter-Clockwise

Table C.3: Landing plan parameter used at the start of second day of field experiments

Appendix D

Addition field experiment results

This chapter includes additional field experiment gather during the field test at Agdenes. Table D.1 shows the performance of the RTK-GNSS during the second day of field experiment at the time the autonomous landing system was tested.

Nr.	FIX %	FLOAT %	NONE %
1	99.2	0.8	0.0
2	100	0.0	0.0
3	99.9	0.1	0.0
4	99.9	0.1	0.0
5	100	0.0	0.0
6	100	0.0	0.0
7	99.6	0.4	0.0
8	99.9	0.31	0.0

Table D.1: Performance of the RTK-GNSS system during the second day of executing landing plans

Appendix E

Rtklib Configuration

This appendix chapter contain the configuration file used to configure RTKlib in the X8, which is given as:

```
# RTKNAVI options (2015/10/30 13:05:07, v.2.4.2)

pos1-posmode       =movingbase # (0:single,1:dgps,2:kinematic,
3:static,4:movingbase,
5:fixed,6:ppp-kine,
7:ppp-static)
pos1-frequency     =l1         # (1:l1,2:l1+l2,3:l1+l2+l5,
4:l1+l2+l5+l6,
5:l1+l2+l5+l6+l7)
pos1-soltype       =forward    # (0:forward,1:backward,2:combined)
pos1-elmask        =10         # (deg)
pos1-snrmask_r     =off        # (0:off,1:on)
pos1-snrmask_b     =off        # (0:off,1:on)
pos1-snrmask_L1    =0,0,0,0,0,0,0,0,0
pos1-snrmask_L2    =0,0,0,0,0,0,0,0,0
pos1-snrmask_L5    =0,0,0,0,0,0,0,0,0
pos1-dynamics      =on        # (0:off,1:on)
pos1-tidecorr      =off        # (0:off,1:on,2:otl)
pos1-ionoopt       =brdc       # (0:off,1:brdc,2:sbas,3:dual-freq,
4:est-stec,5:ionex-tec,
6:qzs-brdc,7:qzs-lex,8:vtec_sf,
9:vtec_ef,10:gtec)
pos1-tropopt       =saas       # (0:off,1:saas,2:sbas,3:est-ztd,
4:est-ztdgrad)
```

```
pos1-sateph        =brdc       # (0:brdc,1:precise,2:brdc+sbas,
3:brdc+ssrapc,4:brdc+ssrcom)
pos1-posopt1       =off        # (0:off,1:on)
pos1-posopt2       =off        # (0:off,1:on)
pos1-posopt3       =off        # (0:off,1:on)
pos1-posopt4       =off        # (0:off,1:on)
pos1-posopt5       =off        # (0:off,1:on)
pos1-exclsats      =           # (prn ...)
pos1-navsys        =7          # (1:gps+2:sbas+4:glo+8:gal+16:qzs+32:comp)
pos2-armode        =fix-and-hold # (0:off,1:continuous,2:instantaneous,
3:fix-and-hold)
pos2-gloarmode     =off        # (0:off,1:on,2:autocal)
pos2-bdsarmode     =off        # (0:off,1:on)
pos2-arthres       =3
pos2-arlockcnt     =0
pos2-arelmask      =0          # (deg)
pos2-arminfix      =10
pos2-elmaskhold    =0          # (deg)
pos2-aroutcnt      =5
pos2-maxage        =30         # (s)
pos2-syncsol       =on        # (0:off,1:on)
pos2-slipthres     =0.05       # (m)
pos2-rejionno      =30         # (m)
pos2-rejgdop       =30
pos2-niter         =1
pos2-baselen       =0          # (m)
pos2-basesig       =0          # (m)
out-solformat      =llh        # (0:llh,1:xyz,2:enu,3:nmea)
out-outhead        =off        # (0:off,1:on)
out-outopt         =off        # (0:off,1:on)
out-timesys        =gpst       # (0:gpst,1:utc,2:jst)
out-timeform       =tow        # (0:tow,1:hms)
out-timendec       =3
out-degform        =deg        # (0:deg,1:dms)
out-fieldsep       =
out-height         =ellipsoidal # (0:ellipsoidal,1:geodetic)
out-geoid          =internal   # (0:internal,1:egm96,2:egm08_2.5,
3:egm08_1,4:gsi2000)
out-solstatic      =all        # (0:all,1:single)
out-nmeaintv1      =0          # (s)
out-nmeaintv2      =0          # (s)
```

```
out-outstat        =state        # (0:off,1:state,2:residual)
stats-eratio1      =100
stats-eratio2      =100
stats-errphase     =0.003      # (m)
stats-errphaseel   =0.003      # (m)
stats-errphasebl   =0          # (m/10km)
stats-errdoppler   =1          # (Hz)
stats-stdbias      =30         # (m)
stats-stdiono      =0.03       # (m)
stats-stdtrop      =0.3        # (m)
stats-prnaccelh    =10         # (m/s^2)
stats-prnaccelv    =10         # (m/s^2)
stats-prnbias      =0.0001     # (m)
stats-prniono      =0.001      # (m)
stats-prntrop      =0.0001     # (m)
stats-clkstab      =5e-12      # (s/s)
ant1-postype       =llh        # (0:llh,1:xyz,2:single,3:posfile,
4:rinexhead,5:rtcm)
ant1-pos1          =90         # (deg|m)
ant1-pos2          =0          # (deg|m)
ant1-pos3          =-6335367.6285 # (m|m)
ant1-anttype       =
ant1-antdele       =0          # (m)
ant1-antdeln       =0          # (m)
ant1-antdelu       =0          # (m)
ant2-postype       =llh        # (0:llh,1:xyz,2:single,3:posfile,
4:rinexhead,5:rtcm)
ant2-pos1          =90         # (deg|m)
ant2-pos2          =0          # (deg|m)
ant2-pos3          =-6335367.6285 # (m|m)
ant2-anttype       =
ant2-antdele       =0          # (m)
ant2-antdeln       =0          # (m)
ant2-antdelu       =0          # (m)
misc-timeinterp    =off        # (0:off,1:on)
misc-sbasatsel     =0          # (0:all)
misc-rnxopt1       =
misc-rnxopt2       =
file-satantfile    =
file-rcvantfile    =
file-staposfile    =
```

```
file-geoidfile     =
file-ionofile      =
file-dcbfile       =
file-eopfile       =
file-blqfile       =
file-tempdir       =/tmp/
file-geexefile     =
file-solstatfile   =
file-tracefile     =
#

inpstr1-type       =serial     # (0:off,1:serial,2:file,3:tcpsvr,
4:tcpcli,7:ntripcli,8:ftp,9:http)
inpstr2-type       =tcpcli     # (0:off,1:serial,2:file,3:tcpsvr,
4:tcpcli,7:ntripcli,8:ftp,9:http)
inpstr3-type       =off        # (0:off,1:serial,2:file,3:tcpsvr,
4:tcpcli,7:ntripcli,8:ftp,9:http)
inpstr1-path       =uart/2:115200:8:n:1:off
inpstr2-path       =:@10.0.60.51:50022/:
inpstr3-path       =
inpstr1-format     =ubx        # (0:rtcm2,1:rtcm3,2:oem4,3:oem3,
4:ubx,5:ss2,6:hemis,7:skytraq,
8:gw10,9:javad,10:nvs,11:binex,
12:rt17,15:sp3)
inpstr2-format     =ubx        # (0:rtcm2,1:rtcm3,2:oem4,3:oem3,
4:ubx,5:ss2,6:hemis,7:skytraq,
8:gw10,9:javad,10:nvs,11:binex,
12:rt17,15:sp3)
inpstr3-format     =rtcm2      # (0:rtcm2,1:rtcm3,2:oem4,3:oem3,
4:ubx,5:ss2,6:hemis,7:skytraq,
8:gw10,9:javad,10:nvs,11:binex,
12:rt17,15:sp3)
inpstr2-nmeareq    =off        # (0:off,1:latlon,2:single)
inpstr2-nmealat    =0          # (deg)
inpstr2-nmealon    =0          # (deg)
outstr1-type       =serial        # (0:off,1:serial,2:file,3:tcpsvr,
4:tcpcli,6:ntripsvr)
outstr2-type       =file        # (0:off,1:serial,2:file,3:tcpsvr,
4:tcpcli,6:ntripsvr)
outstr1-path       =../tmp/ttyV0:115200:8:n:1:off
outstr2-path       =/opt/lsts/rtklib/log/rtklib_output%Y%m%d%h%M.pos
```

```
outstr1-format     =enu        # (0:llh,1:xyz,2:enu,3:nmea)
outstr2-format     =enu        # (0:llh,1:xyz,2:enu,3:nmea)
logstr1-type       =file        # (0:off,1:serial,2:file,3:tcpsvr,
4:tcpcli,6:ntripsvr)
logstr2-type       =file        # (0:off,1:serial,2:file,3:tcpsvr,
4:tcpcli,6:ntripsvr)
logstr3-type       =off        # (0:off,1:serial,2:file,3:tcpsvr,
4:tcpcli,6:ntripsvr)
logstr1-path       =/opt/lsts/rtklib/log/
rtklib_ubxstream_x8_log%Y%m%d%h%M.ubx
logstr2-path       =/opt/lsts/rtklib/log/
rtklib_ubxstream_base_log%Y%m%d%h%M.ubx
logstr3-path       =
misc-svrcycle      =50         # (ms)
misc-timeout       =30000      # (ms)
misc-reconnect     =30000      # (ms)
misc-nmeacycle     =5000       # (ms)
misc-buffsize      =32768      # (bytes)
misc-navmsgsel     =all        # (0:all,1:rover,2:base,3:corr)
misc-proxyaddr     =
misc-fswapmargin   =30         # (s)
#misc-startcmd =./rtkstart.sh
#misc.startcmd =./rtkshut.sh
file-cmdfile1 =/etc/rtklib/data/ubx_raw_10hz.cmd
file-cmdfile2 =/etc/rtklib/data/ubx_raw_10hz.cmd
```