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

Abstract

Contents

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 Unmanned Aerial Vehicles (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 with a accurate navigation system the case of what to do when the positioning system degenerates must be resolved such that system failure does not occur. An other premise is that the position of the net is known, and available for the path system. With a known position of the landing net the UAV must gracefully perform a graceful decent, preferable a glide slope towards the landing net position. The length of the glide slope will be limited by the operator, which dictates that the UAV must be in the correct pose before starting the decent.

1.2 Previous work

There has been perform several studies on autonomous landing system, and there currently exist commercial available system. 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 INS/Global Positioning System (GPS)[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 Global Navigation Satellite System (GNSS) receiver system, which is typically single frequency receivers. In the paper [Barber et al., 2007] a landing system was proposed that 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 was 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 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 a autonomous landing, however it require that the visual image is sent from the UAV to a ground station. In addition the system require a clear image in order to calculate guidance command 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 a 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 a 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 a 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 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 more robust navigation system. A continuation of the work done in [Skulstad and Syversen, 2014] was done in [Frølich, 2015]. The work simulated a autonomous net landing, however no physical experiment was perform. The result in the work indicated that further work on the controllers was required, in addition to the landing path which was not suited for a Visual Line Of Sight (VLOS) UAV operation due to no spacial restrictions.

1.3 Contributions

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 path generator is a improved version of the landing path used in [Skulstad and Syversen, 2014] moved into the DUNE environment, and combined with the landing path generator interface created in the master thesis [Frølich, 2015]. The navigation system continues the work started in the master thesis [Spockeli, 2015], where Real Time Kinematic GNSS (RTK-GNSS) was made available to the DUNE environment. This thesis propose a strategy for handling RTK-GNSS drop out for a short duration, which is a frequent problem with RTK-GNSS. The strategy proposed includes fusing data from a secondary GNSS system in case of loss of RTK-GNSS lock. This work will not look into the internal algorithm for calculation of RTK-GNSS, which is performed in the open-source program library RTKlib. The control system used during the testing of the system was developed at the UAVLab by a fellow master student [Nevstad, 2016]. 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, which guaranty a flyable path from any initial position to a landing zone, with a specified decent angle. The landing plan generator has been implemented in the DUNE runtime environment, which is capable to be used in both a stationary and moving net landing. In addition to the implementation of the landing plan a **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 position and velocity solution. In addition the **robustness of the RTK-GNSS** has been increased by creating a bias estimator, which estimate the bias between RTK-GNSS and a standalone GNSS receiver. The bias is used to compensate the external navigation position solution such that the external navigation position solution equals a FIX RTK-GNSS position solution.

3. **A navigation source monitor graphical interface** has been created to provide visual indication on the source of the navigation data used in the DUNE environment, in order to create a feedback loop on the state of the

navigation system to the operator. The graphical interface is color coded such that changes in the state of the navigation system detected with more ease.

4. **A mobile sensor with RTK-GNSS** has been created to be a reference position for a stationary net. The mobile sensor unit can be place on a runway, and used as a reference point for net placement in the command and control monitor, and provide accurate position solution for placement of net through the use of RTK-GNSS.

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

1.4 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 a Miniature air vehicle (MAV).

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.

Chapter 6 present experimental testing of the path and navigation system in the field

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

Chapter 2

Basis and modelling

2.1 UAV model

A UAV model is presented in [Beard and McLain, 2012], which present the kinetic equations of a general MAV in the. The kinetic equations is given in the body frame, which is fixed to the frame of the UAV. The kinematics equations is 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 is 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 UAV, S is the platform area of the MAV wing, C_i is 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. $\mathbf{R}(\alpha)_{Stability}^{Body}$ is 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 is 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. The control input for the control input is δ_t.

2.2 Landing path modelling

A landing path is a path following problem where the minimum requirement is that the path is connected, and is flyable. The connection level can be described by the paths 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 from 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

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 form on path is a straight line between P_s and P_f. For simplicity the path is reduced to a 2 dimensional case, where the straight line is given as

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

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

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

The tangential vector for a straight line is given as:

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

with it's derivative:

$$\dot{\psi}(\varpi) = 0 \tag{2.8}$$

A path constructed by straight lines is G^0, however since the tangential vectors derivative is zeros, it is discontinuous between two line segments with different heading and therefore not G^1[TODO: LEGG INN FIGUR SOM VISER DISKONINUITETEN]. The disadvantage with a path which is G^0 is that large discontinuity between two tangential vectors will cause problem for a control system.

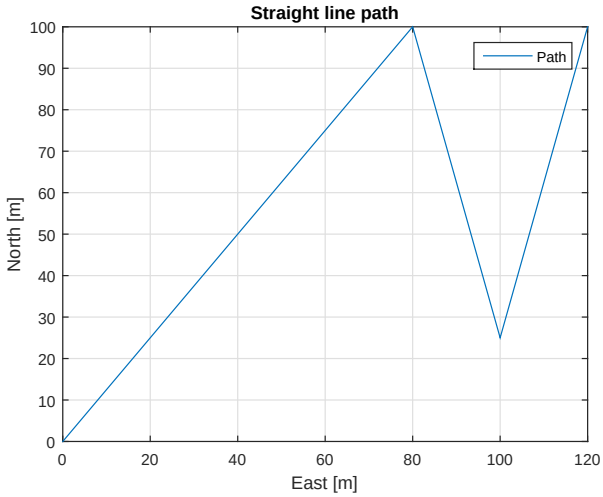


Figure 2.1: Straight line path

2.2.2 Dubins path

An alternative to a straight line path is a path constructed by straight lines and circle. Such a path is Dubins path[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 disadvantage with Dubins path is that the curvature is discontinues, which gives a path from P_s to P_f with smoothness level of G^1.

A Dubins path that is constructed where the final orientation is fixed has four different ways to be constructed, which is determined by the rotation directions. The four types of Dubins path that is used in this thesis is 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 that is used to construct the path is found in [Tsourdos et al., 2010] section 2.2.1, with a constructed path shown in figure 2.2. In figure 2.2 the whole line is the path, with the doted lines used to constructed the path.

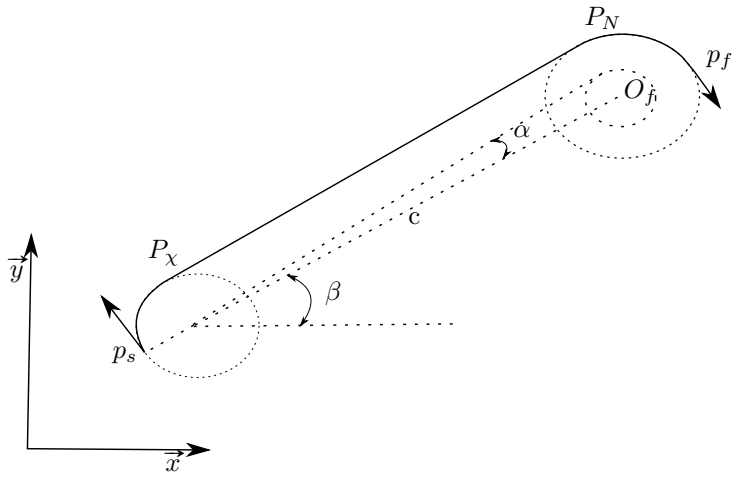


Figure 2.2: Dubins path

The first step if to determine the start and final turning circle center. The center is found with the equations:

$$X_{cs} = X_s - R_s \cos(\psi_s \pm \frac{\pi}{2}) \tag{2.9a}$$

$$Y_{cs} = Y_s - R_s \sin(\psi_s \pm \frac{\pi}{2}) \tag{2.9b}$$

$$X_{cf} = X_f - R_f \cos(\psi_f \pm \frac{\pi}{2}) \tag{2.9c}$$

$$Y_{cf} = Y_f - R_f \sin(\psi_f \pm \frac{\pi}{2}) \tag{2.9d}$$

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 final heading. The centres for the start and final turning circle is defined as:

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

$$\mathbf{O}_{cf} = \begin{bmatrix} X_{cf} \\ Y_{cf} \end{bmatrix} \tag{2.11}$$

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

$$|c| = ||\mathbf{O}_{cs} - \mathbf{O}_{cf}||_2 \tag{2.12}$$

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

$$\alpha = \arcsin\left(\frac{R_f - R_s}{|c|}\right) \tag{2.13a}$$

$$\beta = \arctan\left(\frac{Y_{cf} - Y_{cs}}{X_{cf} - X_{cs}}\right) \tag{2.13b}$$

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 point is given as:

$$x_{P_\chi} = x_{cs} + R_s\cos(\phi) \tag{2.14a}$$

$$y_{P_\chi} = x_{cs} + R_s\sin(\phi) \tag{2.14b}$$

$$x_{P_N} = x_{cf} + R_f\cos(\phi) \tag{2.14c}$$

$$y_{P_N} = x_{cf} + R_f\sin(\phi) \tag{2.14d}$$

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

$$\mathbf{P}_\chi = \begin{bmatrix} x_{P_\chi} \\ y_{P_\chi} \end{bmatrix} \tag{2.15a}$$

$$\mathbf{P}_N = \begin{bmatrix} x_{P_N} \\ y_{P_N} \end{bmatrix} \tag{2.15b}$$

The length of the 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 final pose. The length of the path is given as:

$$d = R_s\phi_s + d_t + R_f\phi_f \tag{2.16}$$

where $d_t = ||\mathbf{P}_N - \mathbf{P}_\chi||_2$, ϕ_s and ϕ_f is the arc angle for the start and final circle respectfully.

Chapter 3

Path and navigation

3.1 Landing plan generation system

The path system 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 net position. The approach path is design separately as a lateral Dubins path and a longitudinal straight line path. The approach path is designed to ensure that the UAV is able to enter the landing path at the correct height with the correct attitude.

3.1.1 Landing Path

The landing path is inspired by the work done in [Skulstad and Syversen, 2014] where waypoint 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 angles in the straight line path must be kept small to avoid large transition behaviour from switching way-point, 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. At sea this could be other ships or platforms, and on land it could be threes or hills. The straight line path is constructed relative to the net as shown in figure 3.1, with way-points given

as:

$$\mathbf{WP4} = \begin{bmatrix} -a0 \\ 0 \\ h_{nc} + a1\tan(\gamma_n) \end{bmatrix} \tag{3.1a}$$

$$\mathbf{WP3} = \begin{bmatrix} a1 \\ 0 \\ h_{nc} - a1\tan(\gamma_n) \end{bmatrix} \tag{3.1b}$$

$$\mathbf{WP2} = \mathbf{WP3} + \begin{bmatrix} a2 \\ 0 \\ -a2\tan(\gamma_l) \end{bmatrix} \tag{3.1c}$$

$$\mathbf{WP1} = \mathbf{WP2} + \begin{bmatrix} a3 \\ 0 \\ 0 \end{bmatrix} \tag{3.1d}$$

where the description of the parameters used is given in table 3.1. The net is placed between the fourth and third way points such that transitional behaviour do not occur during the finale stage of the net landing.

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: Net approach parameters

The way point vectors are rotated into the NED frame by a rotation around the z-axes.

$$\mathbf{WP}^n = \mathbf{R}(\psi_{net})\mathbf{WP}^b \tag{3.2}$$

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

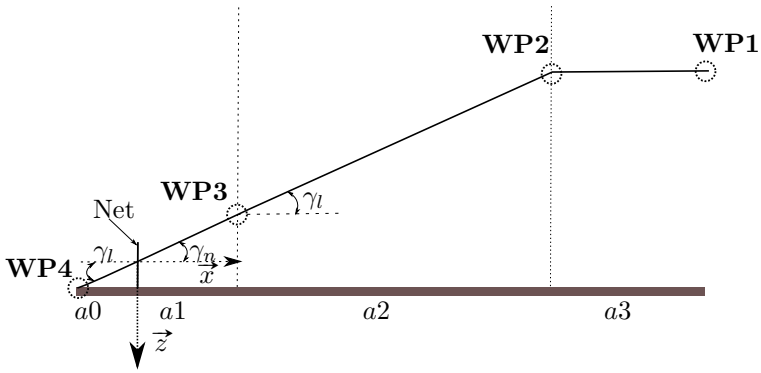


Figure 3.1: The landing path

3.1.2 Approach path

The landing 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 attitude from any initial position. The lateral and longitudinal path are created separately

Lateral path

The lateral path is designed as a Dubins path, with start pose, P_s, at the point where the landing plan generation request was made, and final pose, P_f, at the start of the landing path. Dubins path was chosen since it can be followed by a UAV, and meet the requirement that the UAV enters the landing path with the correct heading.

The lateral path is constructed following the equations in section 2.2.2, however the rotation direction in each circle must first be determined. The desired behaviour is to find the shortest path of the four different rotation pairs given in table 2.2. 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.

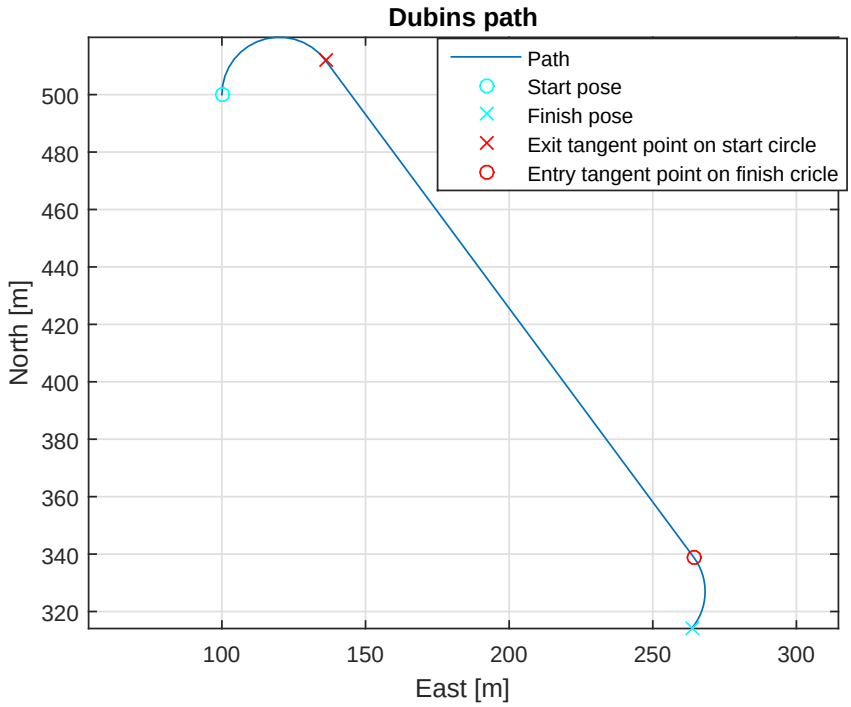


Figure 3.2: Lateral Dubins path

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 start and finish heading defined as ψ_0 and ψ_1 respectfully:

$$\psi_0 = \begin{cases} \text{atan2}(Y_s - Y_{cs}, X_s - X_{cs}) & \text{if start circle} \\ \text{atan2}(Y_{P_N} - Y_{cf}, X_{P_N} - X_{cf}) & \text{otherwise} \end{cases} \tag{3.3a}$$

$$\psi_1 = \begin{cases} \text{atan2}(Y_{P_\chi} - Y_{cs}, X_{P_\chi} - X_{cs}) & \text{if start circle} \\ \text{atan2}(Y_f - Y_{cf}, X_f - X_{cf}) & \text{otherwise} \end{cases} \tag{3.3b}$$

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 respected, including the rotation direction. The maximum turning angle becomes:

$$\psi_{max} = \begin{cases} -|\psi_1 - \psi_0| & \text{if counter clockwise rotation and } \psi_1 - \psi_0 \leq 0 \\ -(2\pi - |\psi_1 - \psi_0|) & \text{if counter clockwise rotation and } \psi_1 - \psi_0 > 0 \\ |\psi_1 - \psi_0| & \text{if clockwise rotation and } \psi_1 - \psi_0 \geq 0 \\ (2\pi - |\psi_1 - \psi_0|) & \text{if clockwise rotation and } \psi_1 - \psi_0 < 0 \end{cases} \tag{3.4}$$

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:

$$h = \frac{d_{arc}}{R} \tag{3.5a}$$

$$N = \left\lceil \frac{\text{sign}(\psi_{max})\psi_{max}}{h} \right\rceil + 1 \tag{3.5b}$$

$$h = \text{sign}(\psi_{max})h \tag{3.5c}$$

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:

$$\psi(\varpi) = \begin{cases} \psi_{max} & \varpi = N - 1 \\ \varpi h & \text{otherwise} \end{cases} \tag{3.6}$$

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

$$\mathbf{p}(\varpi) = \begin{bmatrix} \mathbf{O}_c \end{bmatrix} + R \begin{bmatrix} \cos(\psi_0 + \psi(\varpi)) \\ \sin(\psi_0 + \psi(\varpi)) \end{bmatrix} \tag{3.7}$$

A summary of the lateral path is:

$$\mathbf{p}(i) = \begin{cases} \begin{bmatrix} \mathbf{O}_{cs} \end{bmatrix} + R_s \begin{bmatrix} \cos(\psi_0 + \psi(\varpi)) \\ \sin(\psi_0 + \psi(\varpi)) \end{bmatrix} & \text{Start circle} \\ \begin{bmatrix} \mathbf{O}_{cf} \end{bmatrix} + R_f \begin{bmatrix} \cos(\psi_0 + \psi(\varpi)) \\ \sin(\psi_0 + \psi(\varpi)) \end{bmatrix} & \text{Finish circle} \end{cases} \tag{3.8}$$

where $i = \varpi_s + \varpi_f$ with ϖ_s and ϖ_f as the number of segments in the start and finish circle respectfully.

Longitudinal path

The longitudinal path is designed as a straight line along the lateral path, which results in a spiral path in the arcs created by the lateral 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 accused such that the correct height is reach. The approach decent angle is defined as:

$$\gamma_d = \begin{cases} \text{atan2}(\Delta z, ||\mathbf{p}(i+1) - \mathbf{p}(i)||_2) & \text{if } \text{atan2}(\Delta z, ||\mathbf{p}(i+1) - \mathbf{p}(i)||_2) \leq \gamma_{d_{Max}} \\ \gamma_{d_{Max}} & \text{otherwise} \end{cases} \tag{3.9}$$

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

$$\Delta z = z_d - z(i) \tag{3.10}$$

where z_d is the z component in $WP1$. Continuing the longitudinal path is given as:

$$\mathbf{r}(i+1) = \begin{bmatrix} \mathbf{p}(i) \\ ||\mathbf{p}(i+1) - \mathbf{p}(i)||_2 \tan(\gamma_d) \end{bmatrix} \tag{3.11}$$

where $\mathbf{r}(i)$ is the landing path 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.

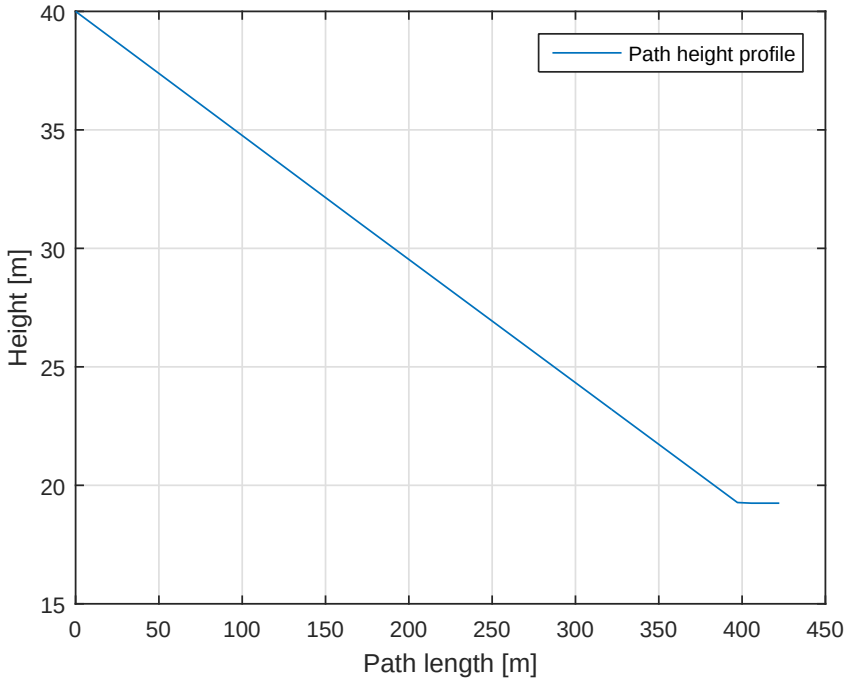


Figure 3.3: Height profile of the landing path

Spiral path The longitudinal path will in the case where the correct height is not reach enter a spiral such that the correct height is reached. The spiral keeps the same turn radius as the finish circle in the lateral 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.

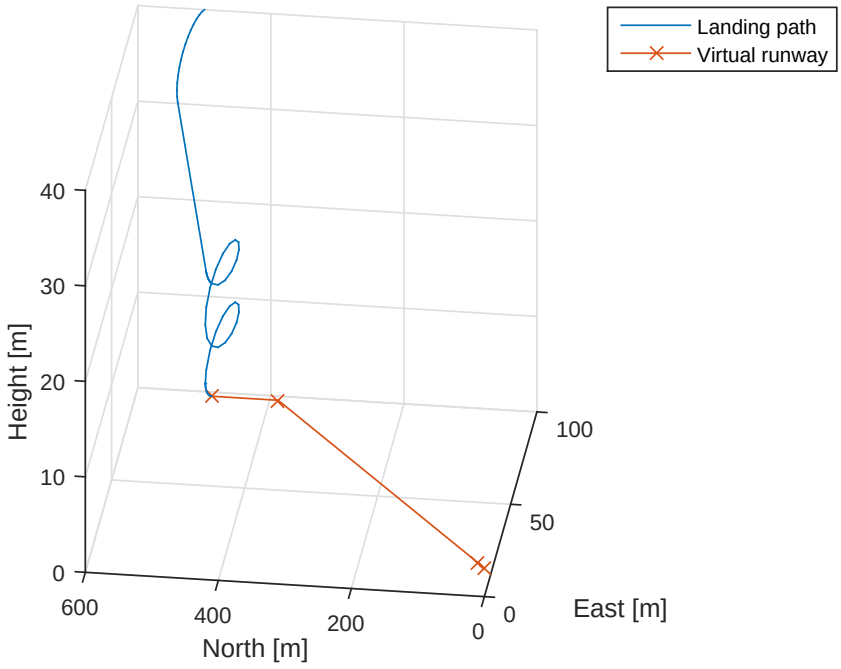


Figure 3.4: Approach path connected to the landing path

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-GPS, 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-GPS is subject to drop out, which create a situation where the navigation system should switch to standalone GPS or wait for the RTK-GPS to return. A state machine has been created to handle the state switching between RTK-GPS and the external navigation data, which is navigation data from the Pixhawk. This is handled in two steps. The first is a short loss compensator stage, where the backup standalone GPS is compensated to get a position solution closer to

the RTK-GPS solution. The second stage fully disconnect the RTK-GPS. First the term RTK-GPS will be further explained, together with typical error sources that affect a GNSS positioning system.

3.2.1 Position estimation RTK-GPS

Real Time Kinematic GPS (RTK-GPS) 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-GPS 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-GPS solution, due to increased effect of atmospheric disturbance, which is further explain in 3.2.1. 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-GPS is depicted in figure 3.5.

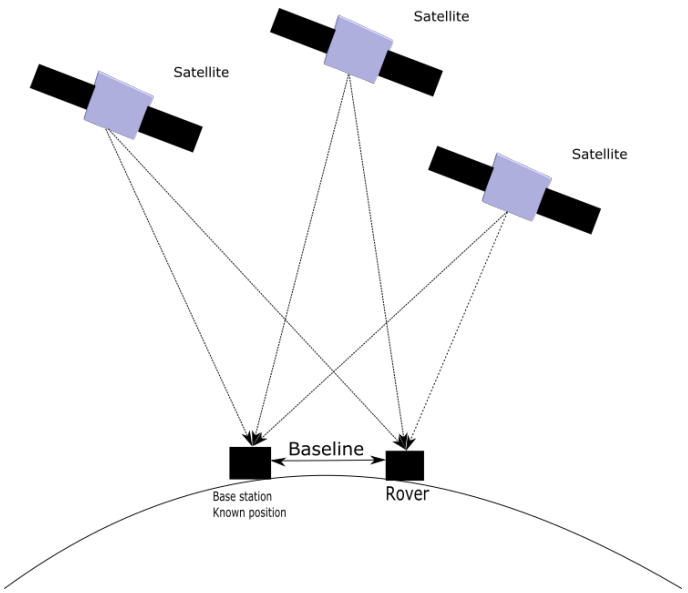


Figure 3.5: Concept figure of Real Time Kinematic GPS (RTK-GPS)

The ability for the rover to resolve the integer ambiguity is a key feature in RTK-GPS. 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 imminently degrade to decimetre level.

In RTK-GPS 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-GPS 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-GPS the relative position between the rover and base station can be determined in real time. This will be the case in automatic ship landing system, where the base station is on a ship, thus must be allowed to move. 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.

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. 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. 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. 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. 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.

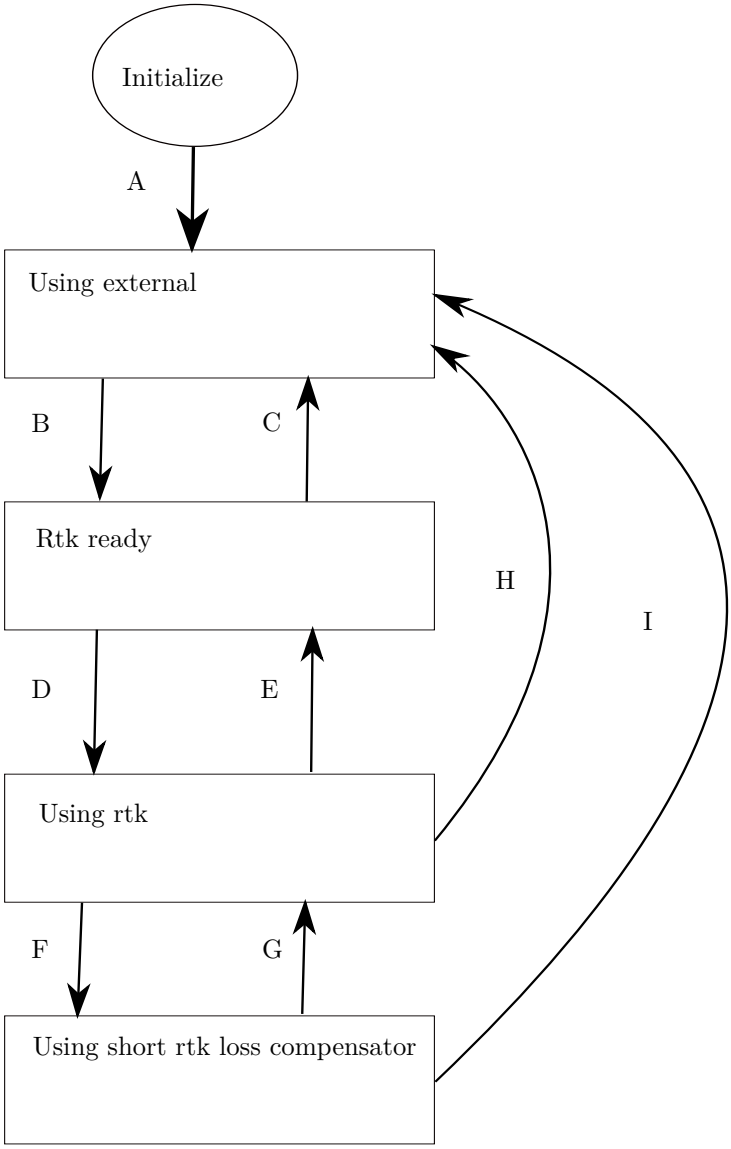
3.2.2 Navigation state control system

The navigation state control system manage the current state of the navigation system, which controls if the RTK-GPS usage and responsible of dispatching the current state of the UAV to the rest of the DUNE system. The structure of the navigation state control system is shown in figure 3.6, with edge description given in table 3.2. The basic state of the navigation system function as if the RTK-GPS system is not available, which makes the navigation system independent from the RTK-GPS system. This is a requirement for the navigation system since it should function in a UAV where RTK-GPS is not present.

In order for the navigation system to switch state into "Rtk ready" the RTK-GPS message must contain a valid base position, which must be set by the user. The setting of the base station is specified further in section 5.2.1. In addition the RTK-GPS solution type must be valid for x seconds such that a stable desired solution type is available.

The navigation state control system will only use the RTK-GPS system if the user has configured the navigation system to apply RTK-GPS and if it's available.

The state machine can enter a state during loss of RTK-GPS, where the position solution from the Pixhawk is compensated with the average difference between the RTK-GPS position solution and the Pixhawk solution. This state will be further explained in section 3.2.2.


Initialize
A
Using external
B
C
Rtk ready
H
I
D
E
Using rtk
F
G
Using short rtk loss compensator

Edge	Event	Guard
A	Event: Received External Nav message	None
B	Time out: Fix RTK-GPS solution for x seconds with valid base position	None
C	Time out: x seconds since last valid GpsFixRtk	None
D	Flag: Using Rtk is set true	None
E	Flag: Using Rtk is set false	None
F	Time out: x seconds since last valid GpsFixRtk Event: Received GpsRtkFix with type == None	Short loss compensator: Enabled
G	Event: Received valid GpsFixRtk message	None
H	Time out: x seconds since last valid GpsFixRtk Event: Received GpsRtkFix with type == None	Short loss compensator: Disabled
I	Time out: x seconds since last valid GpsFixRtk	None

Table 3.2: Description of the edges used in the state machine

The state machine is designed to alter behaviour when the state machine enters a new state, with the entries action listed in table 3.4. A summary of the states in the navigation state control system is given in table 3.3.

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

Table 3.3: States in the navigation system with description

State	**Entry action**
Initialize	Start up for the navigation state control system
Using external	RTK-GPS is disabled Availability activation timer for RTK-GPS is started State of navigation system is updated
RTK ready	RTK-GPS is disabled Availability deactivation timer for RTK-GPS is started State of navigation system is updated
Using RTK	RTK-GPS is enabled State of navigation system is updated If short RTK loss compensator is activated: - Short RTK loss compensator activation timer is started If short RTK loss compensator is deactivated: - RTK-GPS time out timer is started
Using short RTK loss compensator	Short RTK loss compensator deactivation timer is started

Table 3.4: Entry action for each state

Short loss of RTKGPS

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. Therefore the average difference between the RTK-GPS 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:

$$\mathbf{e}(n) = \mathbf{p}_1(n) - \mathbf{p}_2(n) \tag{3.12}$$

$$\delta = \frac{1}{N}\sum_{n=0}^{N}(\mathbf{e}(n)) \tag{3.13}$$

where $\mathbf{p}_1(n)$ and $\mathbf{p}_2(n)$ is the position solution sample for the RTK-GPS 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:

$$\mathbf{p}_2(t) + \delta \rightarrow \mathbf{p}_1(t) \tag{3.14}$$

where $\mathbf{p}_1(n)$ and $\mathbf{p}_2(n)$ is the current position solution for the RTK-GPS system and the external navigation system respectfully. The short loss RTK-GPS compensator will trigger if there is a delay in the RTK-GPS system, or a temporary drop out occur. The output frequency of the short loss compensator is set to be the same as the RTK-GPS system, which is estimate by comparing the time each RTK-GPS message is dispatched. This results in prolonging the time where the RTK-GPS 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-GPS system more robust.

3.3 Summary

Chapter 4

Applied software and hardware

4.1 LSTS toolchain

The software that the system is based on 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 different modules, namely Inter-Module Communication (IMC), DUNE, NEPTUS and Glued.

4.1.1 IMC

IMC [Martins et al., 2009] is design to enable interconnections between systems of vehicles, sensors and human operators, which enable the pursuit of 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. A single XML document ease communication between two node when both node use the same document for message definition.

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 that each has there own thread of execution. DUNE

apply a message bus that is responsible for forwarding IMC message from the producer to all registered receivers, which is the only way different DUNE tasks is communicating.

A DUNE task is enabled through a configuration file, where the user can choose in which profile the task should be enabled in. The different profiles are used to separate a SIL test from physical testing of the system. The hardware profile configures the system for a hardware setup, while AP-SIl configure DUNE to a software in the loop test. The different profile configuration in DUNE allows for testing the same system used in a hardware setting with a simulator.

4.1.3 Neptus

Neptus is a Command and Control software which is used to command and monitor unmanned systems that is written in Java. Neptus is able to provide coherent visual interface to command despite the heterogeneity in the controlled system that it is interacting with. This allow 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 IMC, which makes it interoperable with DUNE or other IMC- based peer.

Neptus is able to do MRA (Mission Review and Analysis) after a mission is finished. In the MRA phase Neptus analyse the IMC logs that is collected by e.g. DUNE, such that the result from a completed mission can be presented. In addition Neptus mission review is able to create output files of the log that can be analysed in third party software like Matlab.

4.1.4 Glued

Glued is a minimal Linux operating system distribution, and design with embedded system in mind. It is platform independent, easy to configure and contain only the necessary packages to run on a embedded system. This makes GLUED a light and fast distribution, 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

Real-Time Kinematic Library (RTKLIB)[Takasu and Yasuda, 2009] is a open source program package for standard and precise positioning with GNSS developed by T. Takasu. Real-Time Kinematic Library (RTKLIB) can be configured to apply

RTK-GPS, such that raw GNSS data is used estimate the relative position of the rover with respect to the base station in real time. Figure 4.1 shows how RTKLIB can be used in a RTK-GPS mode, where the two main modules here is str2str and rtkrcv. The version of RTKLIB used in this thesis is RTKLIB2.4.2 [RTKLIB].

Rtklib is configured as a moving baseline, where the baseline between the base station and the rover is accurately estimated with centimeter level accuracy. However the concept of moving baseline indicates that the base station is allowed to move, which require continues calculation of the base station position as a standalone GPS. Therefore the error sources that are mitigated in the RTK-GPS solution is present in the GPS position of the base station. The moving baseline configuration is used since a fixed base station location is not known, and a navigation system that should be used at sea will not have a fixed base station location present.

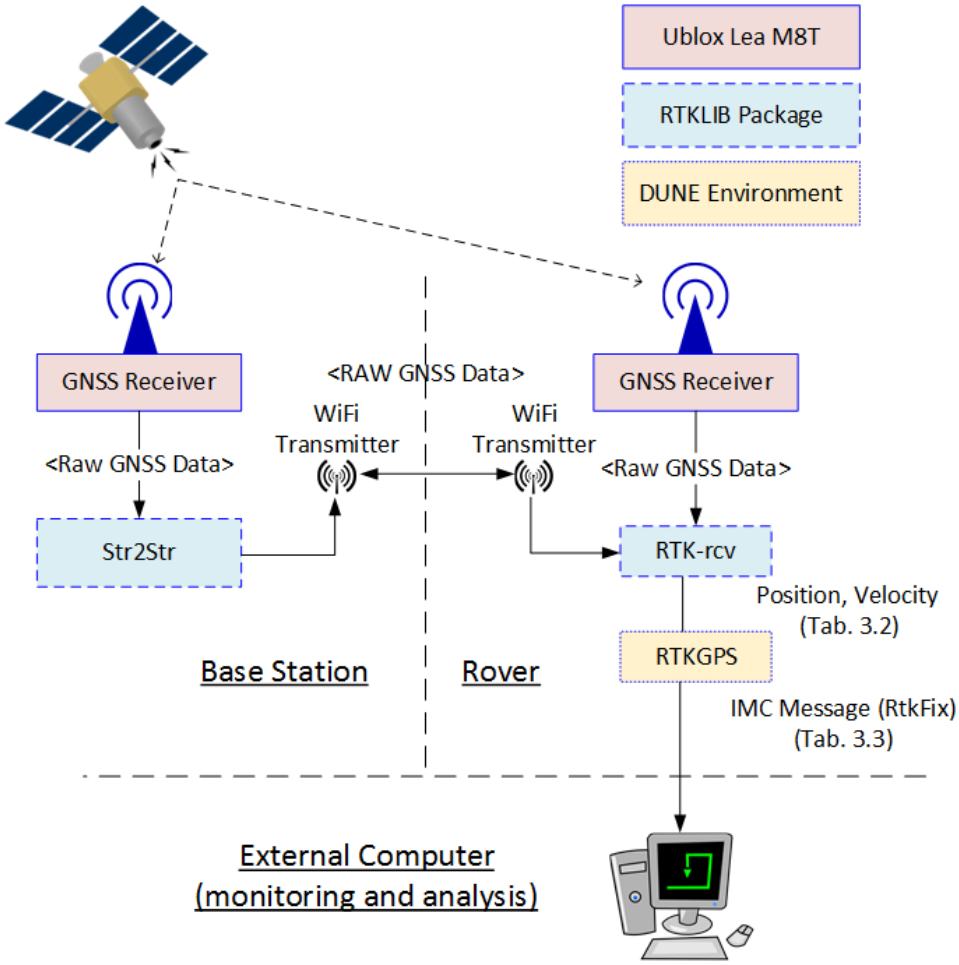


Figure 4.1: The communication structure of RTKLIB

4.3 Pixhawk

3DR Pixhawk is a high-performance autopilot suitable for fixed wing multi rotors, helicopter and other robotic platform that can move. The Pixhawk system comes complete with GPS, imu, airspeed sensor and magnetometer.

4.4 Ardupilot

Ardupilot is an open-source unmanned aerial vehicle platform, able to control fixed wing UAV and multicopters. Ardupilot is used for low level control of the UAV, and is the software that runs on the Pixhawk. Ardupilot is able to communicate to third

party software e.g Dune. Ardupilot uses the sensors in the Pixhawk to calculate the position, velocity and attitude of the UAV, which is sent to DUNE.

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 enable Software In the Loop (SIL) testing of system that is able to run in a hardware configuration with only minor configuration alteration. 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 X8 is used to test the landing path discussed in this thesis, however the navigation system has been tested in both the X8 and a multicopter system.

The hardware configuration used in the X8 and nest systems is based on the proposed hardware in the paper [Zolich et al., 2015]. 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, which is configured with a output rate of 10Hz. 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.

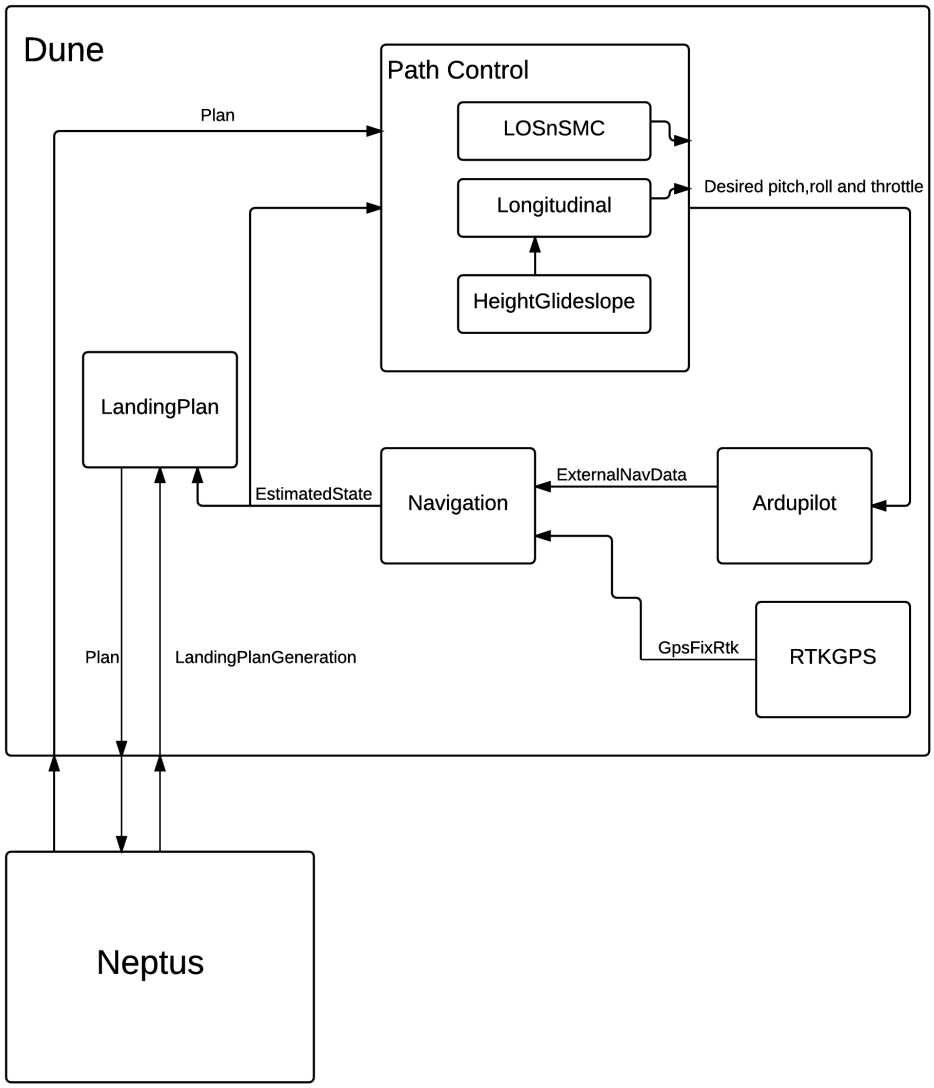


Figure 5.1: A simplified depiction of the interaction between the major components in the autonomous landing system during stationary net landing

5.1 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.

5.1.1 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 5.1, 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. Otherwise a user specific path is chosen
Start circle turning counter clockwise (boolean)	If true the start arc is created such that the turning direction is counter clockwise. Otherwise clockwise. Require Automatic==false
Finish circle turning counter clockwise (boolean)	If true the finish arc is created such that the turning direction is counter clockwise. Otherwise clockwise. Require Automatic==false
Wait at loiter (boolean)	If true a unlimited loiter is included in the path before the path continue with the path along the virtual runway.

Table 5.1: Landing path behaviour setting in LandingPlanGeneration

Neptus API graphical interface

In Neptus the plug-in LandmapLayer, which is an altered version of Neptus plug-in developed in theses [Frølich, 2015], is used to configure the landing plan generation API. The alteration in the plug-in include 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 is shown in figure 5.2. The LandmapLayer function by first placing the net in Neptus, continued by setting the

desired parameter of the landing plan in the graphical interface for the landing plan generator API.

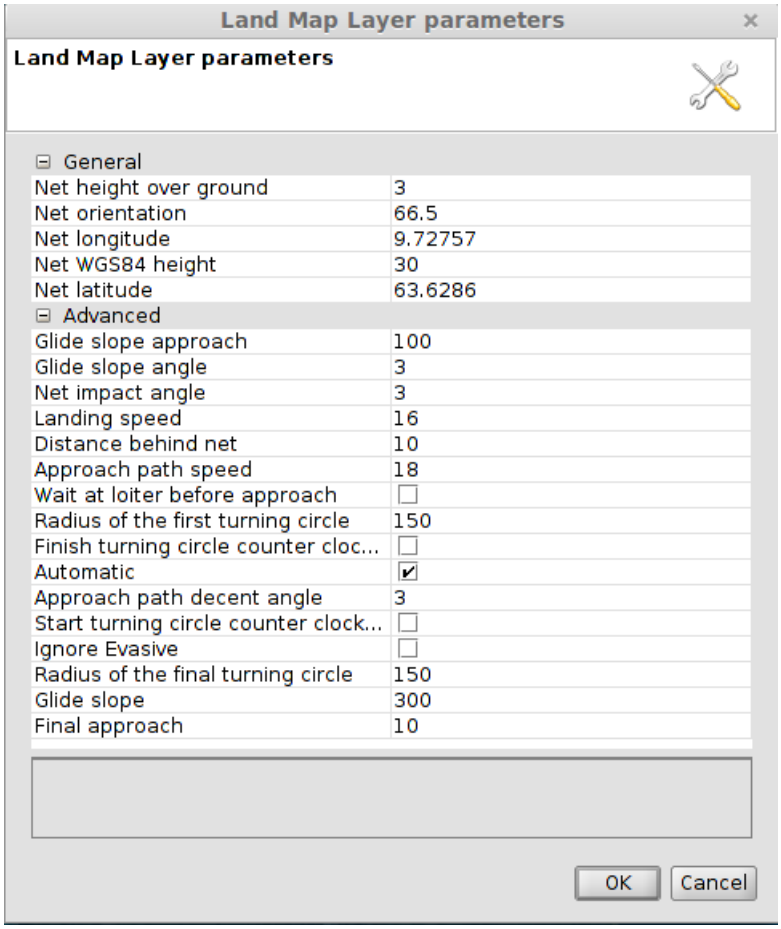


Figure 5.2: Graphical interface for the landing plan generator API in Neptus

5.1.2 Approach path

The approach path given 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 5.3 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.

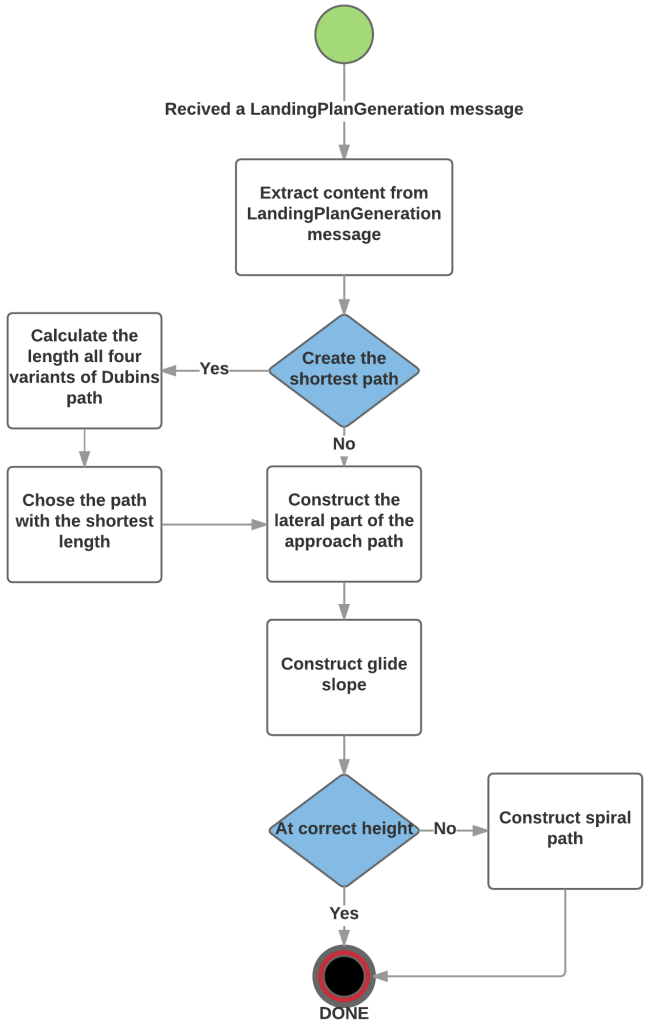


Figure 5.3: Flow chart of approach path creation

The approach path is created as a FollowPath manoeuvre, which is a manoeuvre where the path is offset point from a given 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 spesify the distance between each point both the turning circles, thus given the total number of segments in the circles. The parameter can be used to increase the performance of the lateral control system, continues switching point to keep the bank angle in order to better stay on the circle.

5.1.3 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 5.4 shows the structure of 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 that 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.

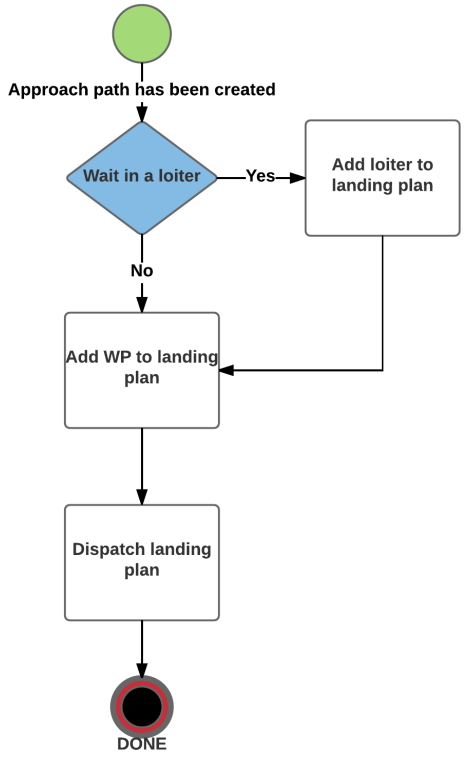


Figure 5.4: Flow chart of the landing plan generation

5.1.4 Software in the loop simulation

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. The 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 simulation 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 X8 used in the simulation has not been completely verified, such that deviation in results and behaviour is expected. Figure 5.5 shows a screenshot of a landing plan created in Neptus with the green triangle symbolizing the net placement.

A landing path was created to simulate a real landing, with the lateral path is shown in figure 5.6 and the height versus the desired height shown in figure 5.7. The plan is designed to fit a operational area where the UAV will be within the line of

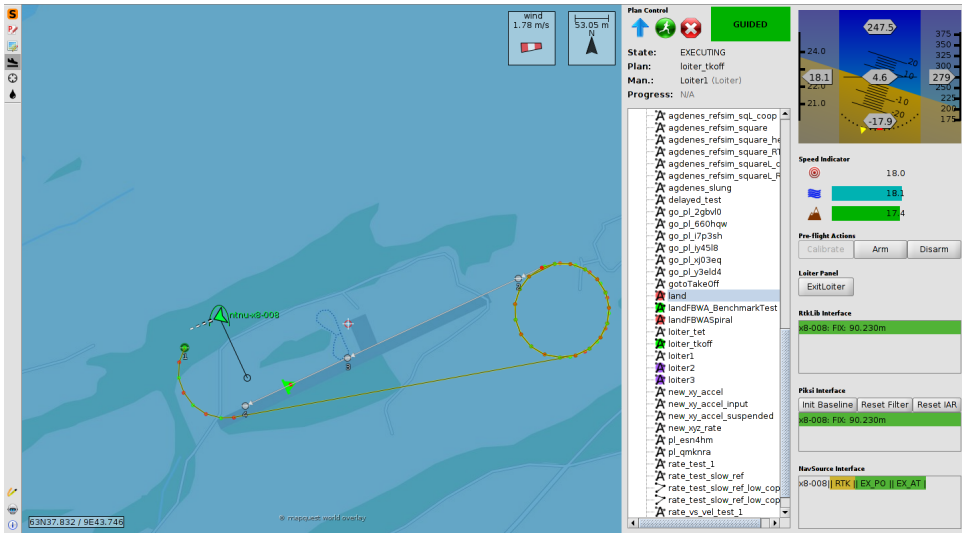


Figure 5.5: Path generated from the landing plan generator, with the net placement as a green triangle

sight of the pilot at any time during execution of the landing plan.

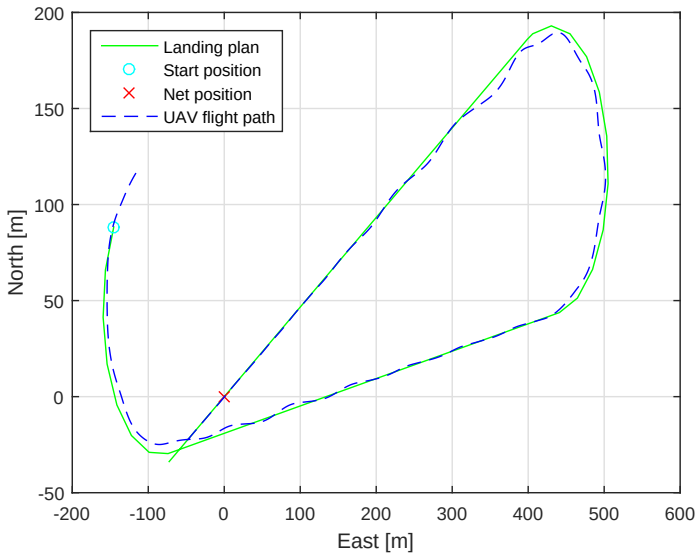


Figure 5.6: North-East plot of a SIL simulation of the autonomous landing system.

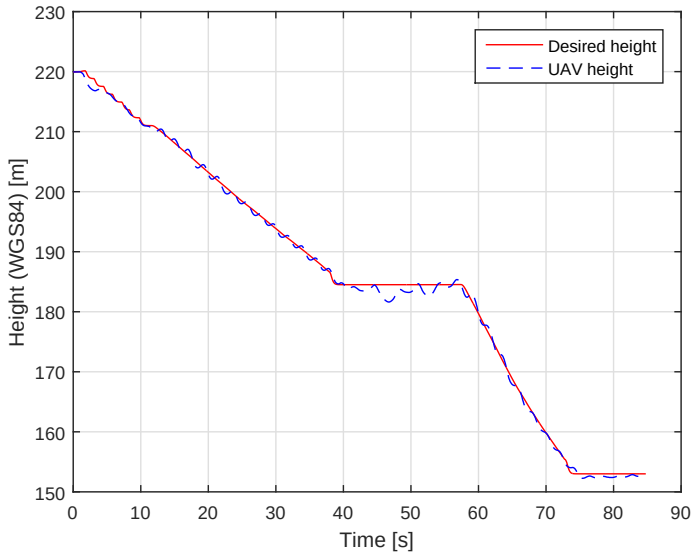


Figure 5.7: The desired height and UAV height when executing the landing plan.

During a landing plan the height in 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 5.8 the spiral function of the lateral path was tested, with the resulting height profile in figure 5.9. The simulation was performed with a wind disturbance of $9m/s$ from west, to simulate how the UAV would perform during the landing plan with wind disturbance The direction of the wind is set such 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. 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. This indicate that during a actual flight the wind will affect the error in the longitudinal control system less then the lateral control system.

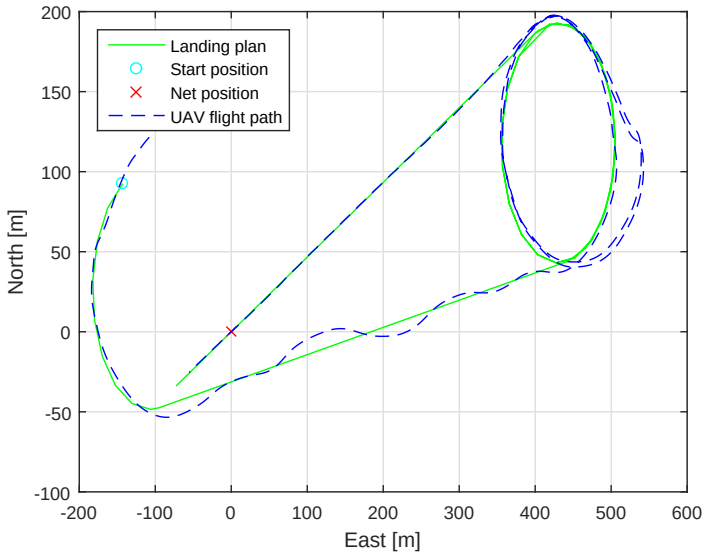


Figure 5.8: North-East plot where the approach path enters a spiral in order to find a path to the correct height. The simulation was performed with $9m/s$ wind from west

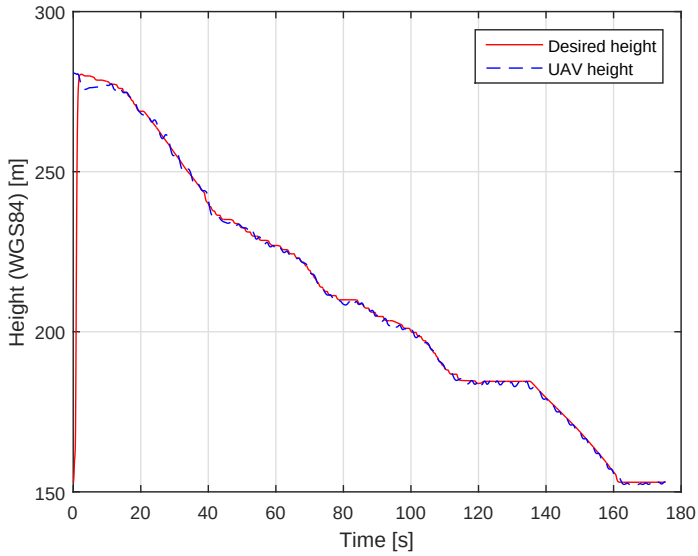


Figure 5.9: The desired height and UAV height when executing the landing plan from a height that trigger a spiral path towards the correct height with maximum decent angle $\gamma_{d_{Max}}$. The simulation was performed with $9m/s$ wind from west

Result of simulations

The system performance in a simulation environment is presented, where the criteria is where the UAV was during the time when it passed the net position. The criteria used to indicate if the X8 would have hit the net is given in table 5.2, 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 5.2: Net hit acceptance criteria

The result from six simulations is given in figure 5.10, with the overall performance of the path following capability of the control system given in table 5.3. The autonomous landing system is able to pass the net with a accuracy that is within the acceptance criteria, indicating 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 controller the angle of the glide slope should not exceed 8 deg.

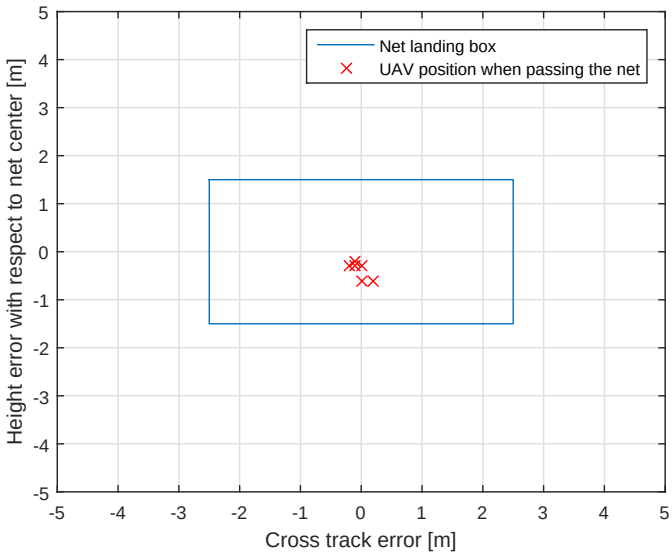


Figure 5.10: UAV position at time of net passing during SIL simulation.

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

5.2 Navigation system

The navigation state control system described in section 3.2.2 is implemented in the DUNE task Navigation, which is used to control the content of the output IMC messages EstimatedState and NavSources. 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 the external navigation position is compensated with the average difference between the RTK solution and the external navigation solution.

5.2.1 RTK-GPS system

The RTK-GNSS solution is made available to the DUNE system through the DUNE task RTKGPS, which is an altered version of the system developed in the theses [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 fist flags are set automatically when receiving a output solution from RTKLib, however the since the base station position is not included in the RTKLib output this flag will not be set. 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 5.11 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 configured from Neptus that the base station position is fixed in order for the base station to start transmitting its own position to the UAV. The advantage fixing the base station position is that all vehicle that uses RTK-GNSS will be in the same reference frame, which enable high accurate vehicle coordination.

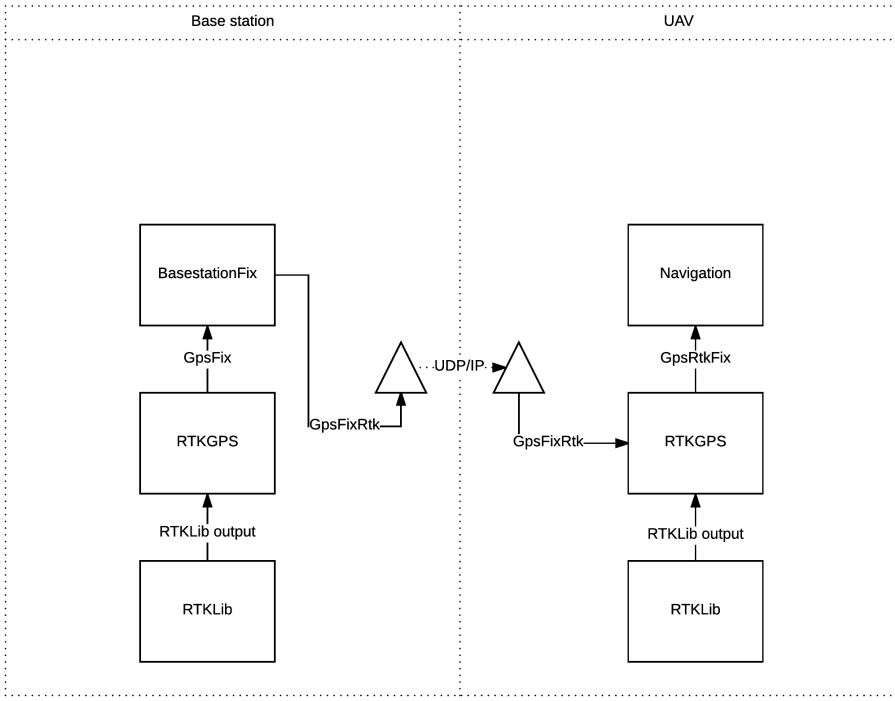


Figure 5.11: Message flow for validation of GpsFixRtk

5.2.2 Navigation state control system

The navigation state control system described in section [**?**] is implemented in the DUNE task Navigation. Figure 5.12 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 3.2.2 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 navigation state control system will change state to only relay on the external navigation system.

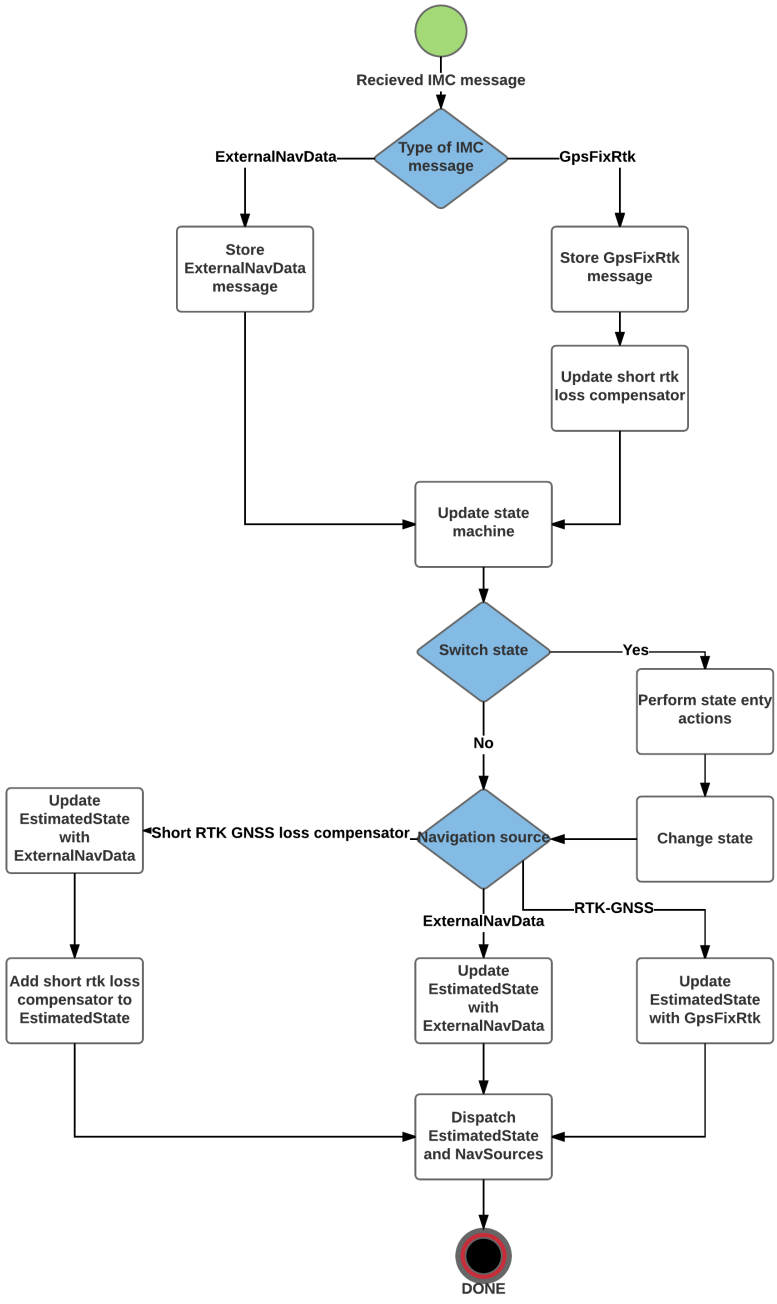


Figure 5.12: Flow chart of the navigation state control system

5.2.3 Mobile sensor unit

The mobile sensor unit is a stationary unit with the sole purpose of providing it's position to the rest of the Dune System to be used as a reference position for net placement in Neptus. The mobile sensor unit apply RTK-GNSS in the same manor as a UAV, however the navigation system 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.

5.2.4 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 color code to indicate which source is currently in use in addition to all sensor system that are available, as seen in figure 5.13, with color description given in table 5.4.

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

Table 5.4: Net approach parameters

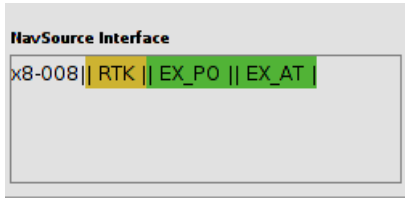


Figure 5.13: Navigation source interface

5.3 Summary

Chapter 6

Experimental field tests

The autonomous landing system has successfully been tested in the field, where the result from two subsequent days is presented. The landing plan was tested and verified at Agdenes airfield with a virtual net placed $26m$ above ground. The landing plan applied the control system given in appendix [**?**] in Fly By Wire A (FBWA) modus, with the navigation system which has been presented in this thesis. The result of the navigation system is presented in section 6.2.

6.1 Landing plan generation system

The landing plan generation system was test in the field at Agdenes airfield, where the results from two consecutive days are presented. The weather condition differed on those two day, where as the first had windspeeds between $8 - 9m/s$ from 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. The virtual net was placed above a runway at Agdenes, such that the landing path is similar to a landing path where a physical net is used. All landing plan was generated when the UAV was in a loiter manoeuvre, such that the plan could be reviewed and the correct controllers assigned to the plan.

6.1.1 Day 1

First path

The first land plan created during the first day is shown in figure 6.1, which shows the desired path, net position and the actual path of the UAV. The desired heigh as well as the actual height is shown in figure 6.2. The straight line in the approach path is here chosen to cross the landing path, which lead the UAV into the crosswind on

the straight line between the circles. The effect of flying in the cross wind introduced oscillatory motion in the UAV, which the lateral control system was unable to remove. The oscillatory behaviour affect the path of the UAV when entering the final turning circle, where it overshoot the path. The overshot may cause the UAV to leave the line of sight of the pilot, which is a critical failure in a LOS flight operation. When entering the landing path the UAV continues to oscillate along the landing path, however the UAV was able at the time of net passing to have a cross track error within the acceptance criteria.

The longitudinal control system is able to follow it's reference, 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 does not reach the height of the net due to a smoothing filer of the desired height in the longitudinal control system. This results in the desired height only to converge to the desired path if the path height remain constant, which is not the case with a net impact angle greater then zero. However with a net impact angle equal to zero, the glide slope will be the sole part of the landing path which is used to decrease the height of the UAV.

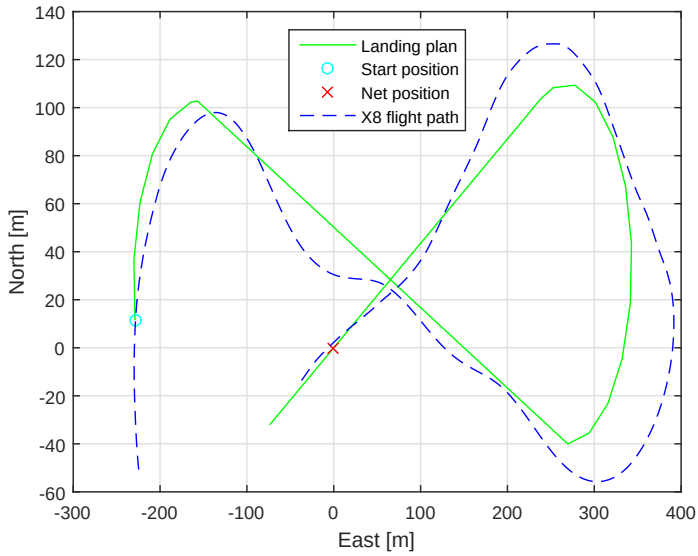


Figure 6.1: North-East plot of a landing plan

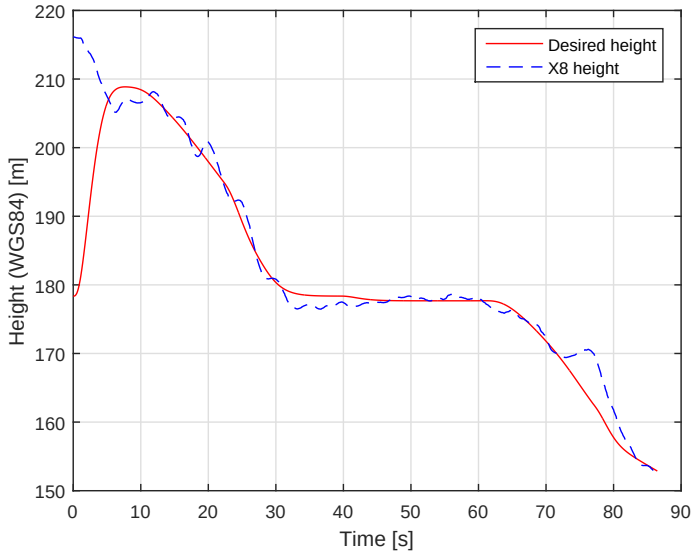


Figure 6.2: Height profile of landing plan with 3 deg net impact angle

Path with $\gamma_n = 0$

A new landing plan, where the desired height and UAV during the landing plan is shown in figure 6.3, was generated with the net impact angle $\gamma_n = 0$. 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.

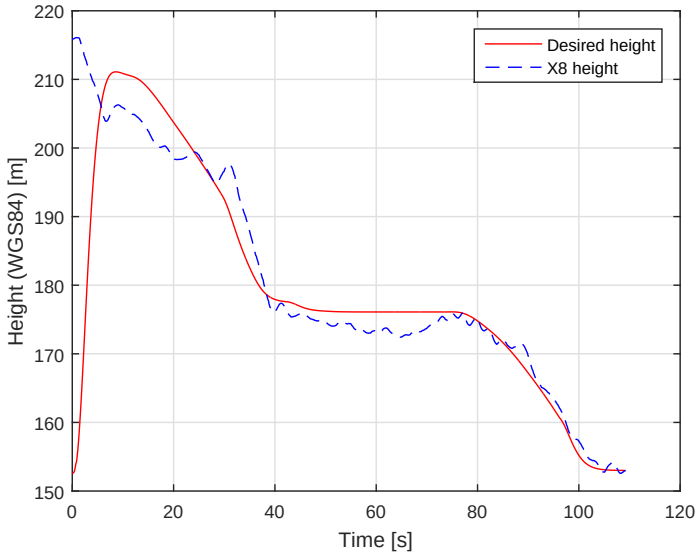


Figure 6.3: Height profile of landing plan with 0 deg net impact angle

Path with inverted rotation direction in the first circle

An attempt to reduce the oscilatoric motion of the UAV during the straight line part of the approach path, a new path was created with inverted rotation direction in the first circle, as shown in figure 6.4. The straight line part of the approach path moved the UAV to fly in the same direction as the wind, which resulted in less oscillatory motion. However the overshoot in the final circle is still present, which is due to a combination of the UAV turning up against the wind and the lateral controller not being design to follow a turning manoeuvre.

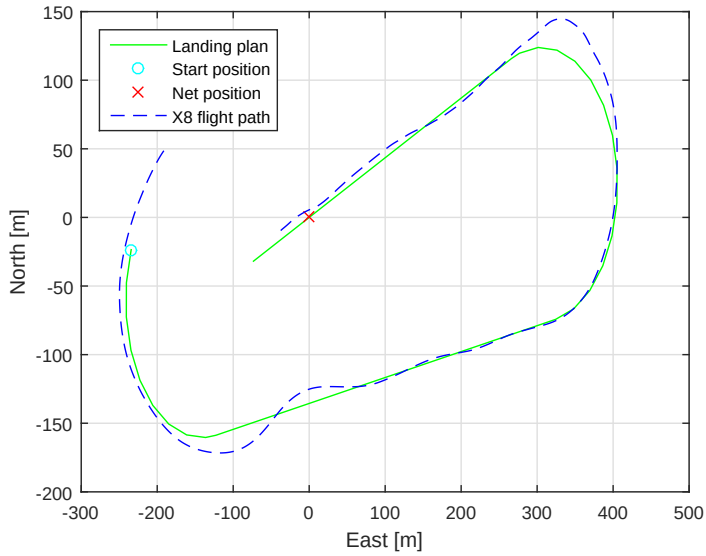


Figure 6.4: North-East plot where the straight line segment between the two circles give a path parallel to the wind

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 overshot at end of the final circle shows that the lateral controller is struggling to handle a turning circle.

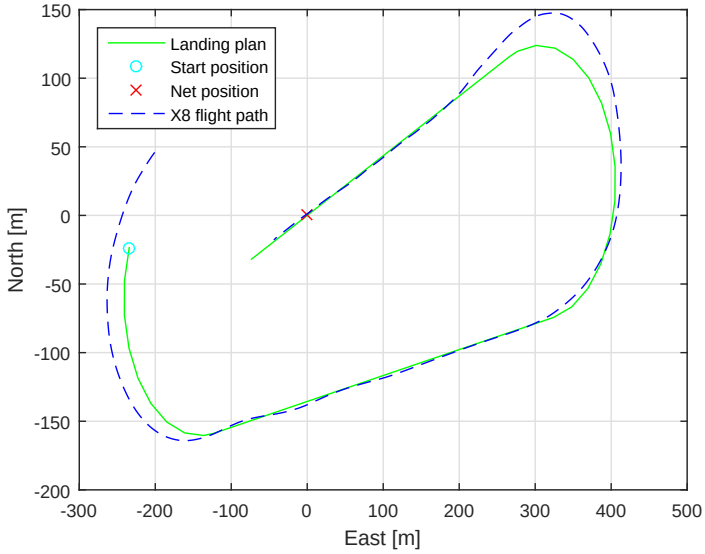


Figure 6.5: North-East plot where the lookahead distance of the lateral controller was reduced to increase performance when flying against the wind

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.

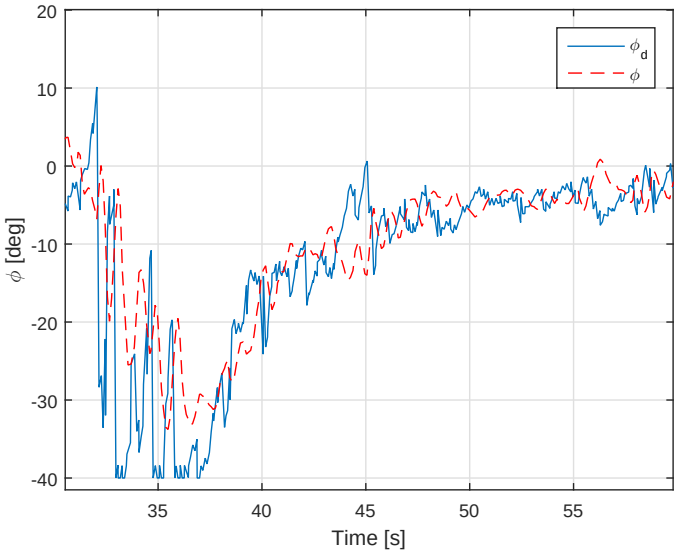


Figure 6.6: The desired roll and actual roll of the UAV

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 final turning circle. The average heigh and cross track error with respect to desired heigh and path respectfully for 11 landing plan performance, 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 decrease to allow increased certainty that the UAV is able to hit the stationary net. In table 6.2 the results of whether the UAV passed through the net or not, is present. Most of the alteration on the path and controllers 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 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. In addition the simulation results was obtain after the two experimental test day, where a different time constant was used in the smoothing filter in the longitudinal control system [Nevstad, 2016]. It is expected that this alteration would have increased the behaviour of longitudinal control system, however this has not been verified due to limitation in flight days.

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 a to high average height error by either passing over the net or in the upper part of the net.

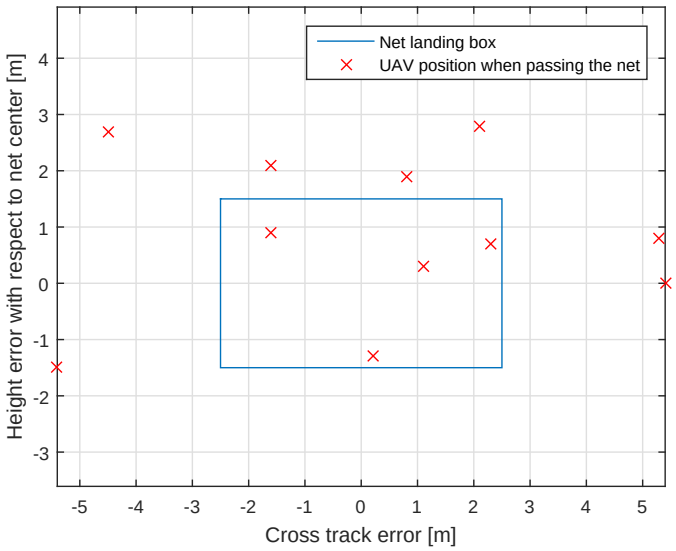


Figure 6.7: Position of UAV relative to the net center at the time of net passing

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. The flight is a Line Of Sight operation, meaning that the pilot must have the UAV within sight during the entire flight. 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 West. Landing attempts from the west has yet to tried, the reason being limited time and that would mean that the UAV would land in the same direction the wind is typically blowing. This would increase the ground speed of the UAV, which is undesired. However in the case where the wind is calm enough to be considered negligible, a autonomous landing from west would be a valid option.

6.1.2 Day 2

First path

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. The virtual net was place further to the west along the runway, and the length and angle of the glide slope was increase to $280m$ and $8\,\text{deg}$ respectfully. The full path specification for the first path is given in appendix C.3. Figure 6.8 show a North-East plot of the path created with the new specification, which show that the lateral path overshoots in both the start and finish turning circle. The overshot in the start circle is a result of the roll angle is not able to follow the desired roll angle fast enough, 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.

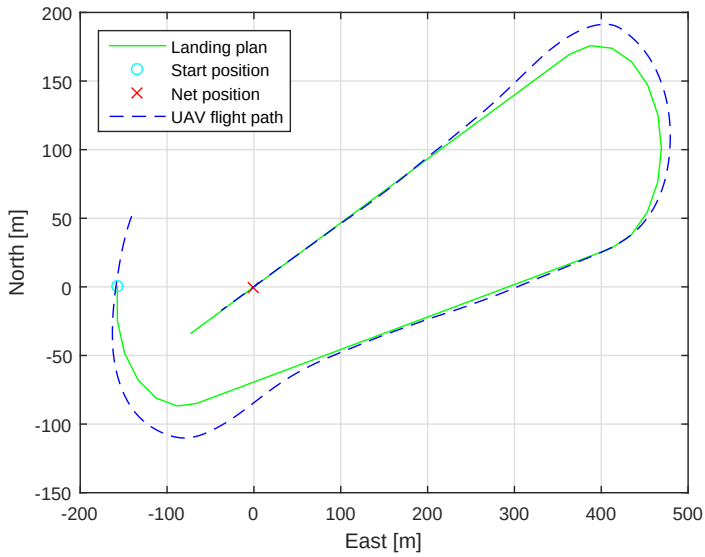


Figure 6.8: North-East plot

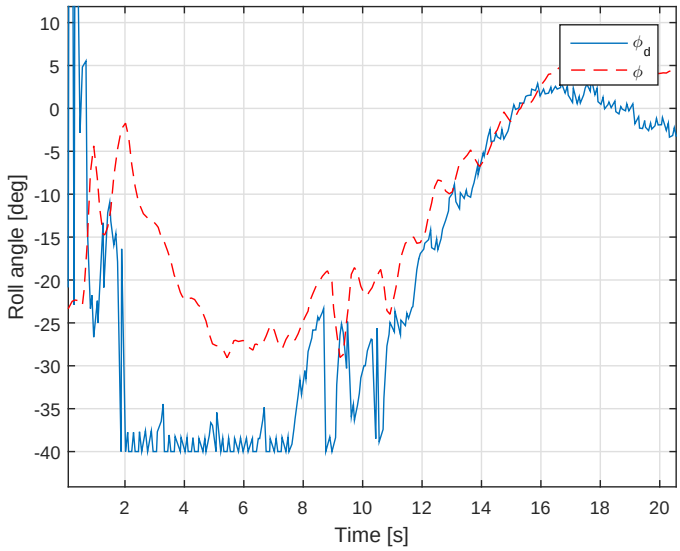


Figure 6.9: Roll

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 major reason for this behaviour is due to the failure 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.

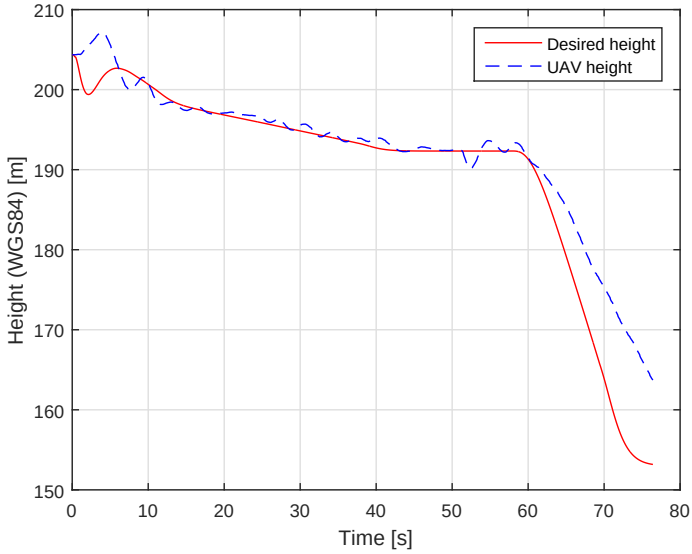


Figure 6.10: Height

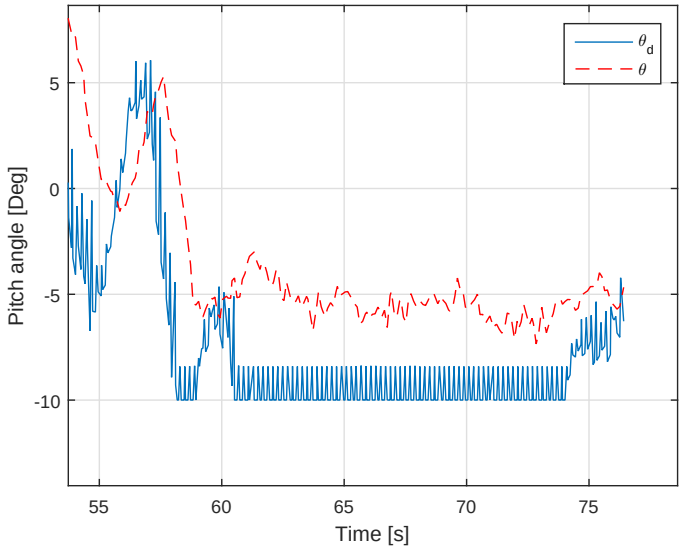


Figure 6.11: Pitch

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.

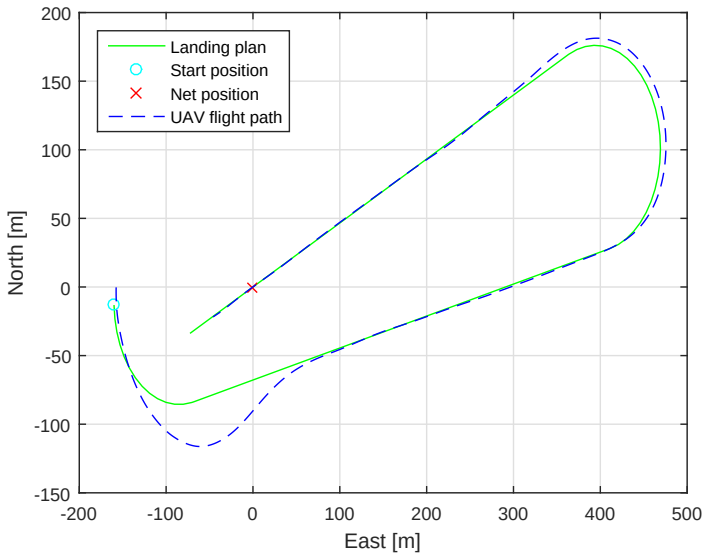


Figure 6.12: North-East plot where the distance between each arc segments has been reduced from $25m$ to $10m$

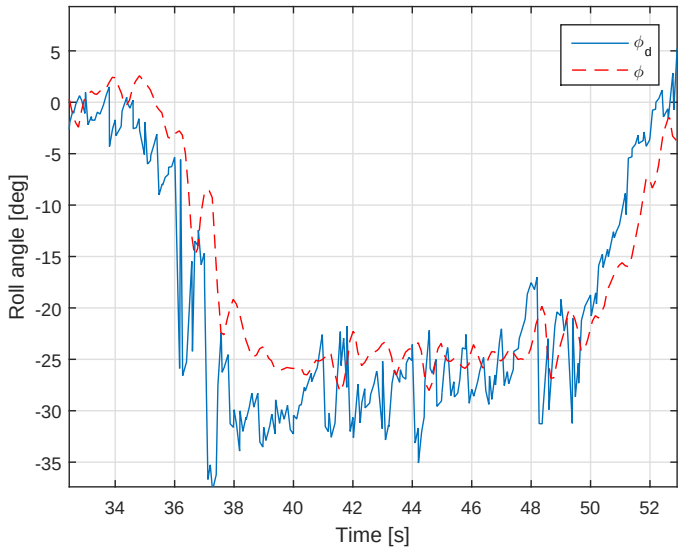


Figure 6.13: Roll and desired roll in the final turning circle

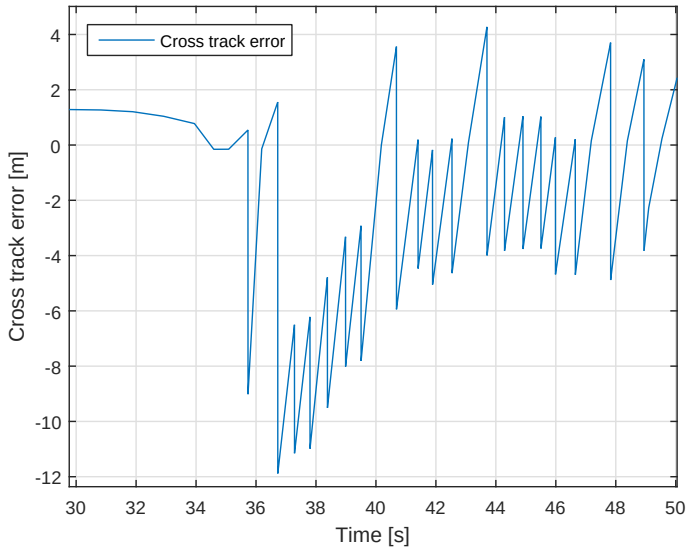


Figure 6.14: Cross track error in the final turning circle

Summary day 2

The second day had ideal weather condition, which allowed for testing of the autonomous landing system in ideal condition. 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 glide slope angle during the flight test is reflected in table 6.3, which contain the results of whether the UAV passed through the net or not. Attempts with glide slope angle greater then 6 deg resulted in the UAV to miss the net, however this performance could be increased with a better the use of the same time constant in the smoothing filter in the longitudinal control system and better tuning of the low level pitch controllers. With the current longitudinal control system the maximum glide slope angle that the UAV is able to follow during a flight 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 could be increased to $120m$ by moving the net further west over. 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, which contain the performance from 8 landing plan missions. In addition the performance from the longitudinal control system remain similar to the first day. This show that the longitudinal control system

is less affected by the wind, 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 hight average height error 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

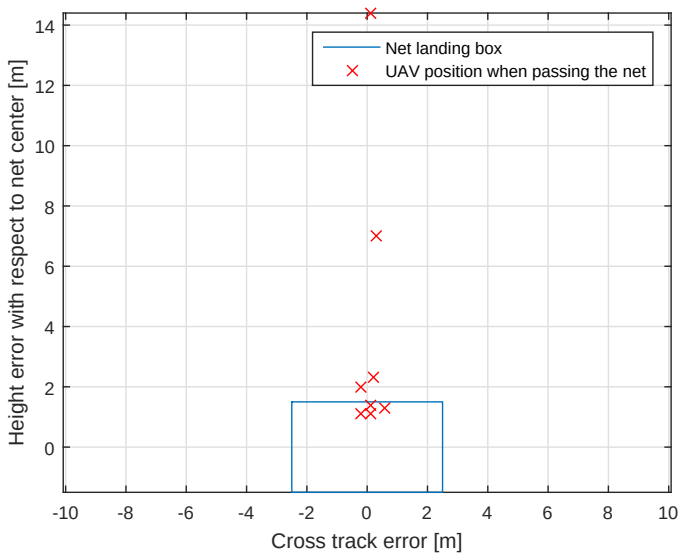


Figure 6.15: Position of the UAV relative to the net center at the time of net passing

6.2 Navigation

6.2.1 RTK-GNSS performance

The RTK-GNSS system was used during both the test days, however due to similar performance only the 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-GPS 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

6.2.2 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.

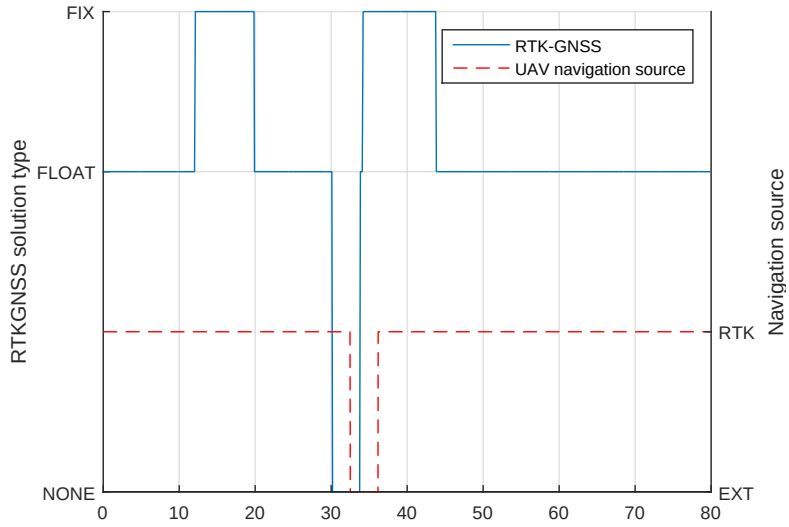


Figure 6.16: State of RTK-GNSS system and UAV navigation source.

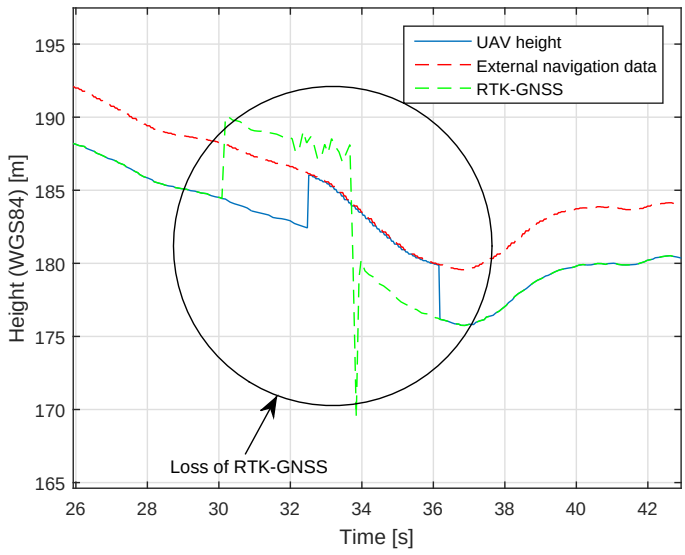


Figure 6.17: Loss of RTK-GPS triggers the short loss compensator such that the UAV keeps the RTK-GPS position solution longer

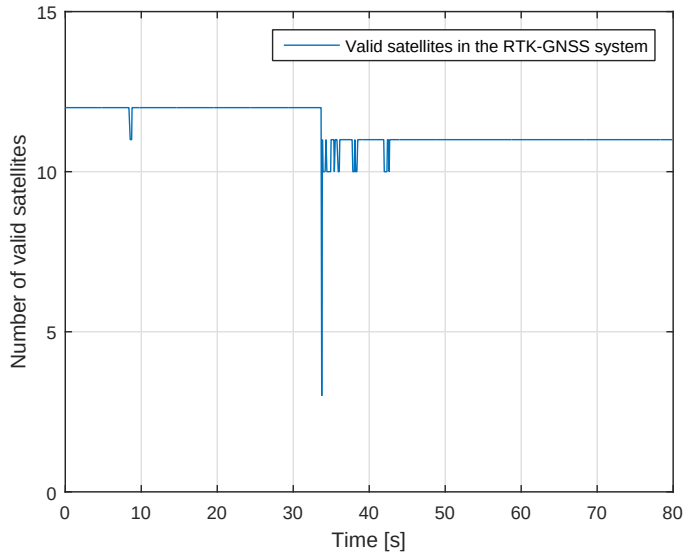


Figure 6.18: Number of valid satellites during a flight mission

6.2.3 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 then the short loss compensator presented in this thesis.

The short loss compensator was able to compensate the solution from the external navigation system, such that a large jump in UAV position was avoided for a short time. 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.

6.3 Summary

This chapter has presented the results of a X8 fixed wing UAV performing autonomous landing in a virtual net, in addition to the performance of the navigation system. The two days where the autonomous landing system was tested had different weather condition, which allowed for identification of strength and weaknesses with the system. The longitudinal control system had a stable performing, 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. Compeering to the results obtain in the SIL experiment 5.1.4 where the low level controllers are fined tuned, it's observed that the performance can increase with better tuning of the low level pitch controller. Better performance from the low level pitch controller can allow for increase in the glide slope angle, which would result in a shorter glide slope and increase operational control over the UAV. The lateral control system function completely different depending on the wind condition. The lateral controller require different tuning parameter depending on the wind speed and in which direction is affecting the UAV. When flying against the wind a short lookahead distance is needed in order for the lateral control system is stay on the desired path, however when flying with the wind the lookahead distance should increase in order for the UAV to stay on the desired path. 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. The lateral control system was unable to completely follow the path along the turning circles without overshooting. A solution which would solve this problem would be a lateral control system design for turning manoeuvres. The problem with this approach would be that a landing path would require two set of lateral control system, which would result in a over complicated system. The best approach would be to improve the current lateral control system to better perform turning manoeuvre, which together with the proposed solution for the lookahead distance would result in a robust lateral control system.

The landing plan has proven that it's flexible when it comes to path alteration. The path can be constructed to suit the needs of the operator, and can be use to increase the performance from the control system. In order for the longitudinal control system to reach a desired height which corresponds to the net center height, the net impact angle must be set to zero. Unfortunately this means that the glide slope will be the only part of the landing path where the height is decreased, however from a control perspective this means that the UAV have longer time to converge to the correct height. In the turning circles the distance between each arc segment can

be adjusted in order to achieve better performance from the lateral control system. However for this to have an affect the time of arrival factor must be kept between $1-2$ seconds, in order to prevent the path control system from switching point to early. A arc segment distance of $10m$ was found to ensure that the lateral control system kept a high roll angle through the turning circle.

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. This strain the operation boundaries in which the UAV operates, since a landing path could move the UAV out of the line of sight of the pilot. An autonomous landing from the east is possible with the current system, however further testing is needed in order to determine the operation weather limitations of the system. In addition a landing attempt from the west could be performed with a lower start height, which then would require less usage of the runway.

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, however an estimate would be that the current limitation could be doubled.

Chapter 7

Conclusion and recommendation for further work

This chapter presents the main findings and conclusions from the thesis, as well as several topics for further work

7.1 Conclusion

The development of a autonomous landing system for fixed wing UAV has resulted in a system capable to perform a successful autonomous landing. The system has successfully been implemented and tested in the field, from which valuable insight operation information has been gathered. The main finding from the operational study has been that a autonomous landing from the East would require the use of the entire airfield at Agdenes, thought would still push the limitation of the operational area in which the UAV can fly. Further testing of the autonomous landing system is needed in order to create operational specification for autonomous landing in both direction on Agdenes airfield.

The landing path generator includes the versatility needed for a autonomous landing operation. The path can be customized to for-fill the need of the user, which was proven when it was used in a concept study of a fixed wing UAV landing system where the net was suspended between multirotor UAVs.

The testing of the lateral control system provided satisfying results during calm wind conditions, however this was not the case when flying in windy weather condition. Tuning of the lookahead distance to the lateral control system increased the performance when flying against the wind, however it decreased the performance when flying with the wind. The lateral control system caused the UAV to overshot in the turning circles along the approach path, which is due to the control system used

was design for straight line path following. The control system must be improved in order to prevent overshot when following the turning circles, which is necessary in a autonomous landing attempt from the East at Agdenes with a stationary net.

The testing of the longitudinal control system proved that the current glide slope angle of the longitudinal control system is 6 deg, however better tuning of the low level pitch controller could increase this limitation. The performance of the longitudinal control system stayed consistent both the field test days.

The navigation system has successfully been integrate with RTK-GNSS, which works with satisfying performance. The navigation system now has the accuracy need in order to perform a autonomous landing. The robustness of the RTK-GNSS has been increased by the introduction of the short RTK-GNSS compensator, all though further study is needed to determine the time limitation of the compensator.

The mobile sensor unit has successfully been created an tested in the field where it was used as a reference position in Neptus. The full potential of the mobile sensor unit has not been reach, since the only interaction with the unit is through view the position in Neptus.

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.

Further increase the robustness of the navigation system capable to keep the RTK-GNSS position accuracy for a prolonged duration then current system

A lateral control system capable with high performance in both turning manoeuvres and straight line path following in the present of wind disturbance. The current lateral control system can be improved by letting the lookahead distance parameter becoming a variable dependent on the ground speed with respect to the desired airspeed.

A longitudinal control system capable of quickly decent over a short distance in a controlled manor, which uses the high dynamic capability in the fixed wing UAV.

Expand the functionality of the mobile sensor unit 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.

Dong Il You, Yeun Deuk Jung, Sung Wook Cho, Hee Min Shin, Sang Hyup Lee, and David Hyunchul Shim. A guidance and control law design for precision automatic take-off and landing of fixed-wing uavs. In *AIAA Guidance, Navigation, and Control Conference*, pages 1–19, 2012.

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 generation API consist of the IMC LandingPlanGeneration, where the field with description is given in table

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: Table containing the landing plan generation 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. The control system is separated into two groups, high level and low level control. The low level control is the control loops for the actuators, which are only controlled by ardupilot. The high level controllers depend on the the configuration of Ardupilot, which is listed in table B.1.

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 s

Mode	**Description**
Guide	Ardupilot set-point guidance and control system
FBWB	DUNE lateral controller with desired height controlled in a set-point controller in Ardupilot
FBWA	DUNE lateral and longitudianl controller, where control input is sent directly to the low level controllers in Ardupilot

Table B.1: Guidance and control modes in autopilot viable for the landing system

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

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

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

B.2 Longitudinal controller

The longitudinal controller is based on the paper [You et al., 2012]

Need to refer to the height smoothing filter with advantage and disadvantage.

Appendix C

Landing plan parameters

C.1 Path parameter used in the SIL simulation

C.2 Start path parameter used the first flight day

C.3 Start path parameter used the second flight day

Parameter	**Value**
Net latitude	63.6281111085521 deg
Net longitude	9.724609316783464 deg
Net height	$150m$
Net height offset	$-3m$ (NED frame)
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

Parameter	**Value**
Net latitude	63.62852832784504 deg
Net longitude	9.726611753451145 deg
Net height	$150m$
Net height offset	$-3m$ (NED frame)
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

Parameter	**Value**
Net latitude	63.62831245876848 deg
Net longitude	9.72534155984453 deg
Net height	$150m$
Net height offset	$-3m$ (NED frame)
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

Appendix D

Navigation performance results

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 RKT-GNSS system the first day during the executing of the landing plans

Appendix E

Approach path in a multirotor recovery system

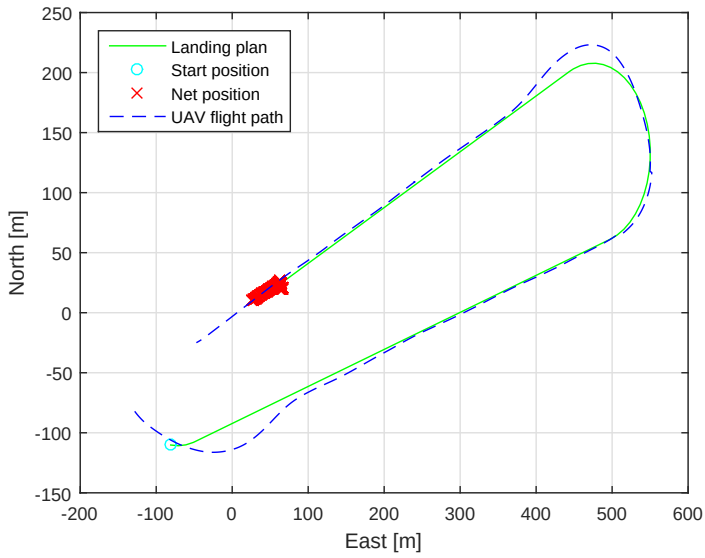


Figure E.1: North-East plot shown a cooperative multicopter UAV net recovery plan

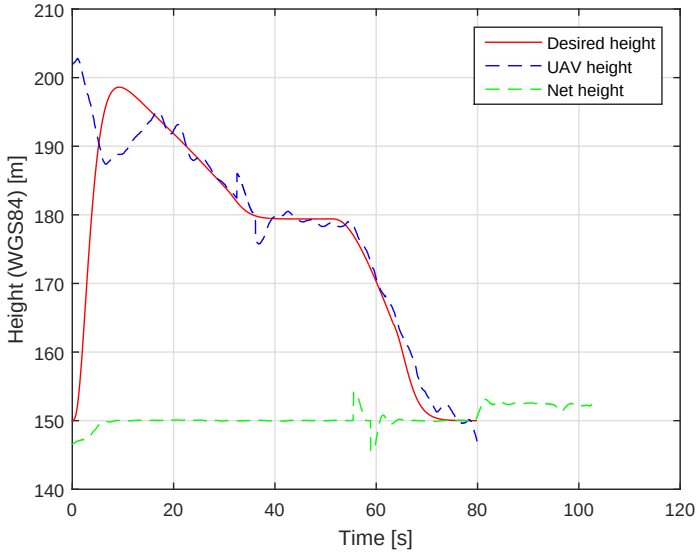


Figure E.2: Height profile of the fixed wing UAV and the suspended net

Appendix F

Rtklib Configuration

Configuration of RTKLIB,rtkrcv is given bellow.

```
# 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
```