NTNU – Trondheim
Norwegian University of
Science and Technology

POSITION CONTROL FOR AUTOMATIC LANDING OF UAV IN A NET ON SHIP

Kjetil Hope Sørbø

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

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Abstract

Contents

Acronyms **ix**

1 Introduction **1**
- 1.1 Background 1
- 1.2 Literature review 2
- 1.3 Contributions 3
- 1.4 Outline 3

2 Basis and modelling **5**
- 2.1 UAV model 5
- 2.2 Landing path modelling 6
  - 2.2.1 Straight lines 7
  - 2.2.2 Dubins path 8

3 Path and Navigation **13**
- 3.1 Path system 13
  - 3.1.1 Landing Path 13
  - 3.1.2 Approach path 15
- 3.2 Navigation system 19
  - 3.2.1 Position estimation RTK-GPS 20
  - 3.2.2 Navigation state 22

4 Applied software and hardware **27**
- 4.1 LSTS toolchain 27
  - 4.1.1 IMC 27
  - 4.1.2 Dune 27
  - 4.1.3 Neptus 28
  - 4.1.4 Glued 28
- 4.2 RTKLIB 28
- 4.3 Pixhawk 30
- 4.4 Ardupilot 30
- 4.5 JSBsim 31

4.6 X8 and nest payload 31

5 Implementation 33
5.1 Landing path 34
5.2 Navigation system 35
5.2.1 RTK-GPS system 36
5.2.2 Operator interface 36

6 SIL test 39
6.1 Setup 39
6.2 Path planing SIL 39
6.3 Navigation SIL 39
6.3.1 Short rtk loss compensator 39

7 Experiment 41
7.1 Path 41
7.1.1 Landing path with pixhawk autopilot 41
7.2 Navigation 42
7.2.1 Formation position 42
7.2.2 Short loss compensator 42

References 45

Appendices

.1 Landing plan parameters 48
.1.1 Parameter used in experiment with ardupilot 48

A Rtklib Configuration 49

Acronyms

ENU East North Up.

GPS Global Positioning System.

IMC Inter-Module Communication.

NED North East Down.

RTK-GPS Real Time Kinematic GPS.

RTKLIB Real-Time Kinematic Library.

SIL Software In the Loop.

UAV Unmanned Aerial Vehicle.

Chapter 1

Introduction

1.1 Background

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 UAVs are mostly operated over land, however in the future this will include over sea as well. An UAV can provide an attractive alternative for many maritime operation where today manned aircraft or satellites is the only solution. In the maritime sector UAV can be used in iceberg management, monitoring of oil spills, search and rescue and maritime traffic monitoring. This will give some challenges which must be overcome. One of these challenges is that the UAV need to be able to perform a autonomous landing.

An important premise for successful and safe UAV operation, in particular at sea, is the provision of a robust system for safe landing of the UAV on a vessel 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 Literature review

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 autonomous landing system must first prove that it can successfully perform a landing on land before it can be tested at sea. The advantage with tests on land is that the net can be assumed stationary, however there are more environmental obstacles that might hinder the UAV.

This thesis focus on the navigation system and generation of landing path in the autonomous landing system. The navigation system apply RTK-GPS to provide high accuracy position estimation, which is needed to perform a autonomous landing. The landing path provide a flyable path from any initial position, where the length and direction of the virtual runway is determined by the operator. Through this work, the following contributions has been made:

- A landing path generator has been created, which guaranty a flyable path with controlled decent from any initial position 3.1.
- The landing path has been implemented in the DUNE runtime environment, which is capable to be used in both a stationary and moving net landing.
- A new Inter-Module Communication (IMC) message has been created to contain the landing path specifications.
- A net nest has been constructed to provide the GPS coordinates for the stationary net.
- A navigation state machine, which is used to control which position solution source should be used in the payload computer. The state machine will try to keep the RTK-GPS available as long as possible by adding the average difference to the position solution from the Pixhawk for a short duration of time until a new viable RTK-GPS message is received.
- A navigation source interface has been created to provide a visual indicator of which navigation system that is used in the DUNE environment.
- Physical experiments of the navigation system and landing path generator.

1.4 Outline

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:

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

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

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

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

The kinetic equations is given as:

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = \mathbf{R}(\mathbf{\Theta})^{Body}_{NED} \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} - \frac{1}{2}\rho V_a^2 S \mathbf{R}(\alpha)^{Body}_{Stability} \begin{bmatrix} F_{Drag} \\ 0 \\ F_{Lift} \end{bmatrix} \tag{2.3a}$$

$$+ \frac{1}{2}\rho V_a^2 S \begin{bmatrix} 0 \\ C_y(\beta, p, r, \delta_a, \delta_r) \\ 0 \end{bmatrix} + \frac{1}{2}\rho S_{Prop} C_{Prop} \begin{bmatrix} (K_{Motor}\delta_t)^2 - V_a^2 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} L \\ M \\ N \end{bmatrix} = \frac{1}{2}\rho V_a^2 S \begin{bmatrix} C_L(\beta, p, r, \delta_a, \delta_r) \\ C_M(\alpha, q, \delta_e) \\ C_N(\beta, p, r, \delta_a, \delta_r) \end{bmatrix} + \begin{bmatrix} -k_{T_p}(K_\Omega \delta_t)^2 \\ 0 \\ 0 \end{bmatrix} \tag{2.3b}$$

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. $\mathbf{R}(\alpha)^{Body}_{Stability}$ 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.

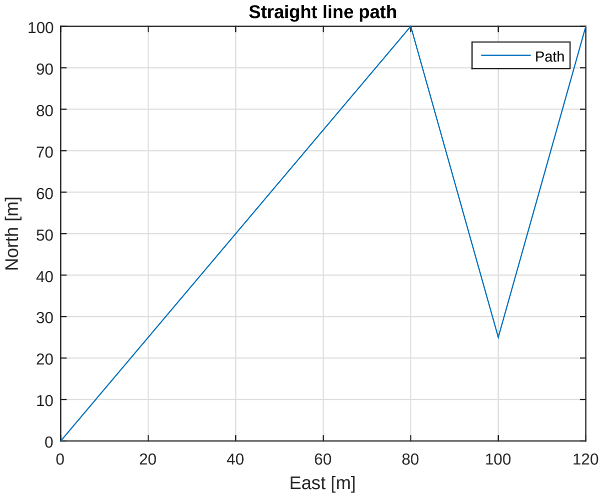


Figure 2.1: Straight line path

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.

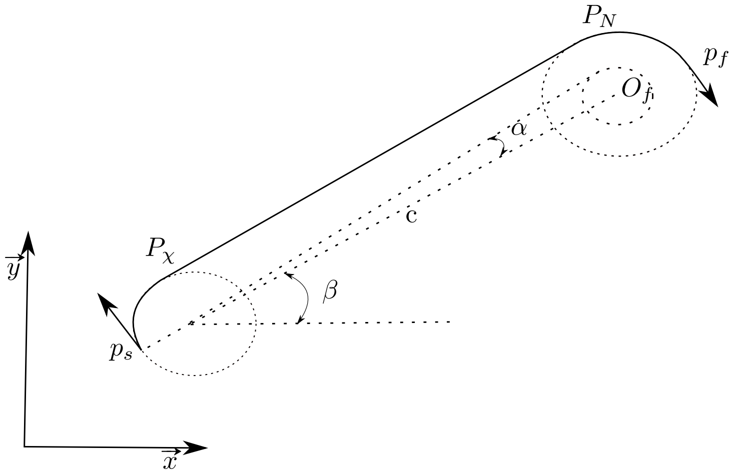


Figure 2.2: Dubins path

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 Path 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.

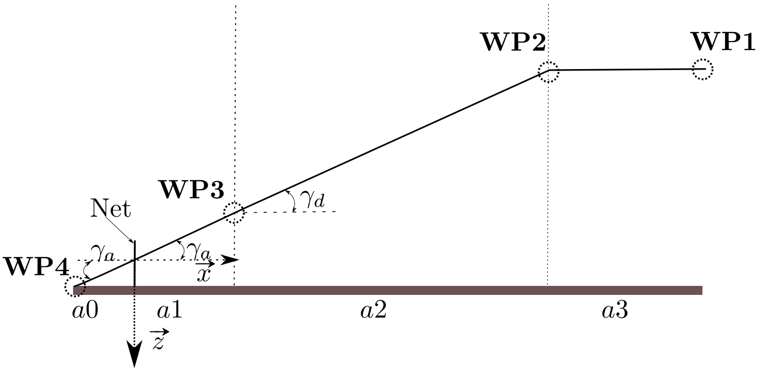


Figure 3.1: The landing path

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.

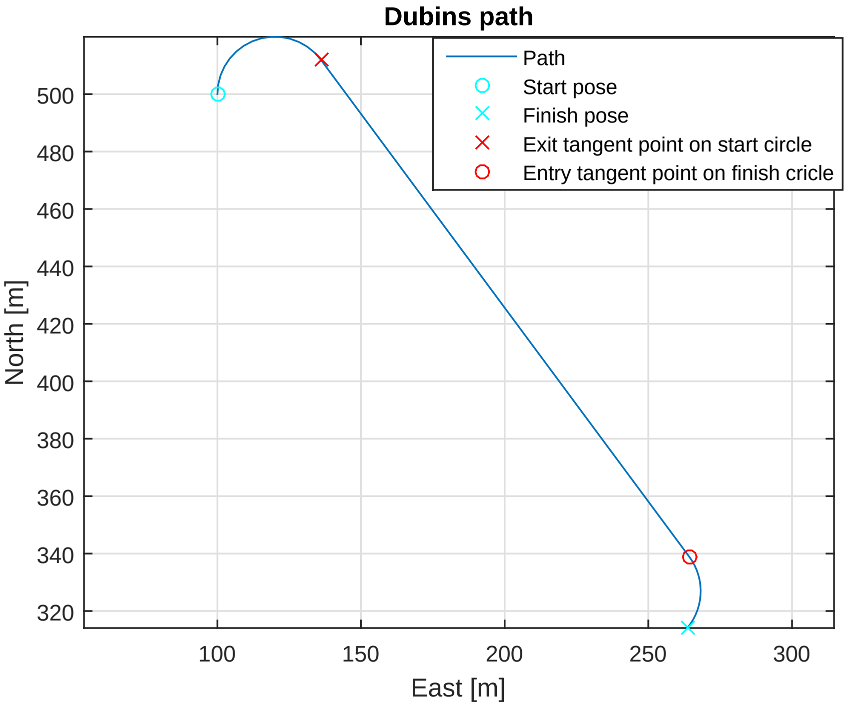


Figure 3.2: Lateral Dubins path

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:

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

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

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:

$$\Delta z = z_{\mathbf{WP1}} - z(i) \tag{3.10}$$

where $Z_{\mathbf{WP1}}$ 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.

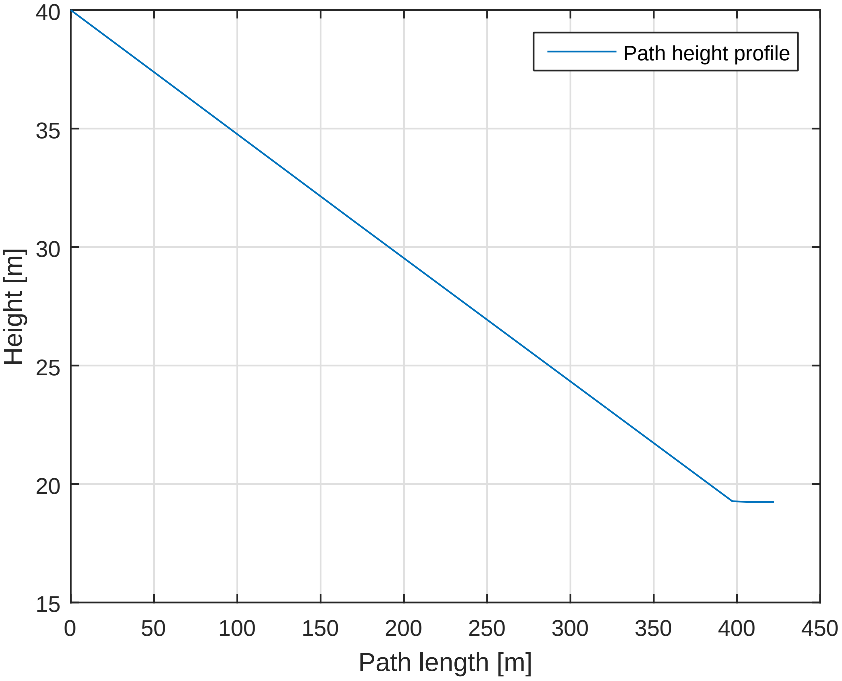


Figure 3.3: Height profile of the landing path

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.

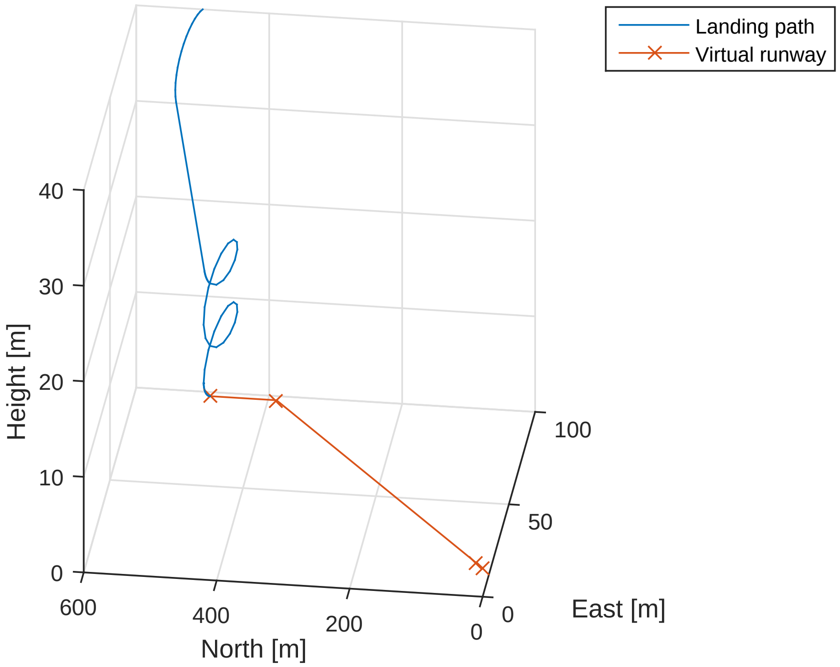


Figure 3.4: Approach path connected to the landing path

3.2 Navigation system

The navigation system apply RTK-GPS for position and velocity measurement, which provide higher position accuracy then a standalone GPS. 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.

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.

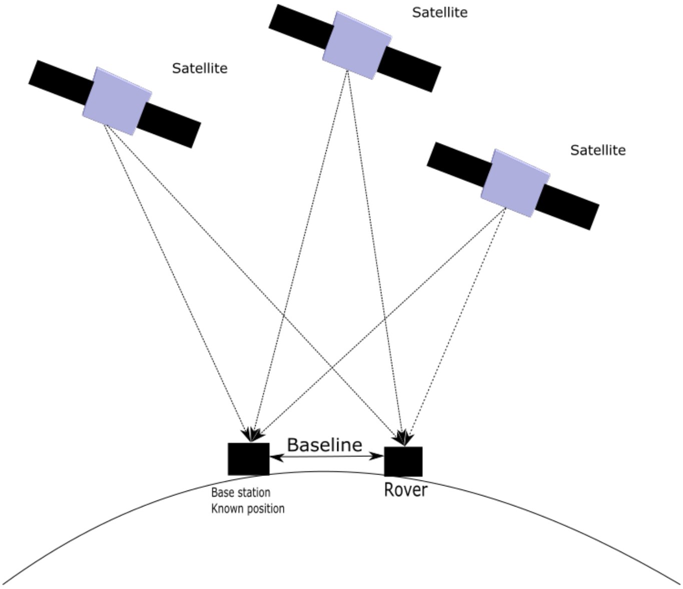


Figure 3.5: Concept figure of Real Time Kinematic GPS (RTK-GPS)

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.

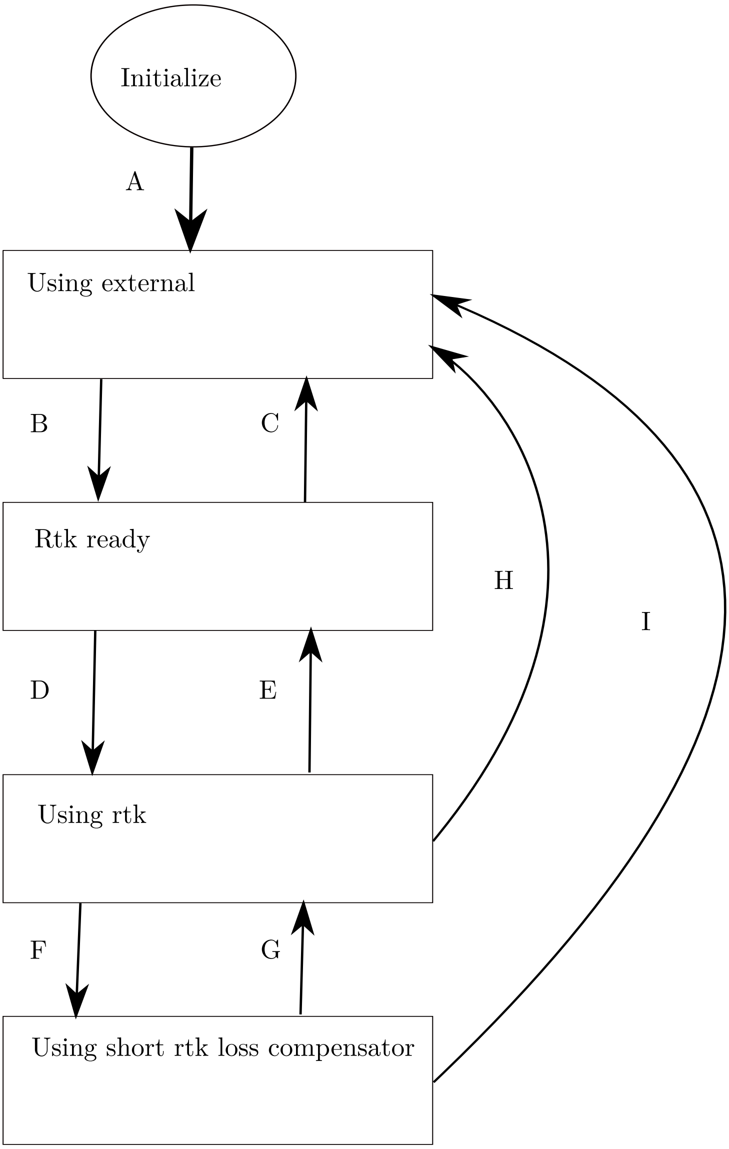
3.2.2 Navigation state

When applying RTK-GPS the navigation system must have a backup positioning system in the case of RTK-GPS drop out. The backup consist of the positioning solution from the pixhawk, which is proven reliable. The switching between the two system is managed by a state machine in the navigation system, where using the pixhawk and RTK-GPS is two separately states. The state machine consists of five states, which is given in table 3.2. The state machine is design such that the user can chose if the system should apply the RTK-GPS system, or continue using the pixhawk, which is the standard positioning system. 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.

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 3.2: States in the navigation system with description

The state machine is shown in figure 3.6, with edge description given in table 3.3. The state transition is triggered on time-out and configuration alteration events. The RTK-GPS solution type must by a stable Fix before the navigation system change state into the "Rtk ready" state. This is to prevent the RTK-GPS to imminently be available again after a drop out, which during a critical operation has caused a abort.


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	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 3.3: Description of the edges used in the state machine

Short loss of RTKGPS

The navigation system must be able to continuing supplying high accurate position solution even thought the RTK-GPS position is momentarily unavailable. This is to prevent large discontinuity in the position estimate, which will happen when the navigation system changes from RTK-GPS to the external navigation system. However it has been observed that the difference between the RTK-GPS and the external navigation system is almost constant. Therefore a solution to handle short loss of RTK-GPS is present where the average difference of the N latest position solution from the RTK-GPS and the external navigation system as:

$$\text{offset} = \frac{1}{N}\sum_{n=0}^{N}(\text{RTKGPS}(n) - \text{External}(n)) \tag{3.12}$$

where $n \in [0, N-1]$ and the RTK-GPS solution is displaced into the External nav system. The short loss compensator prolong 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.

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 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.

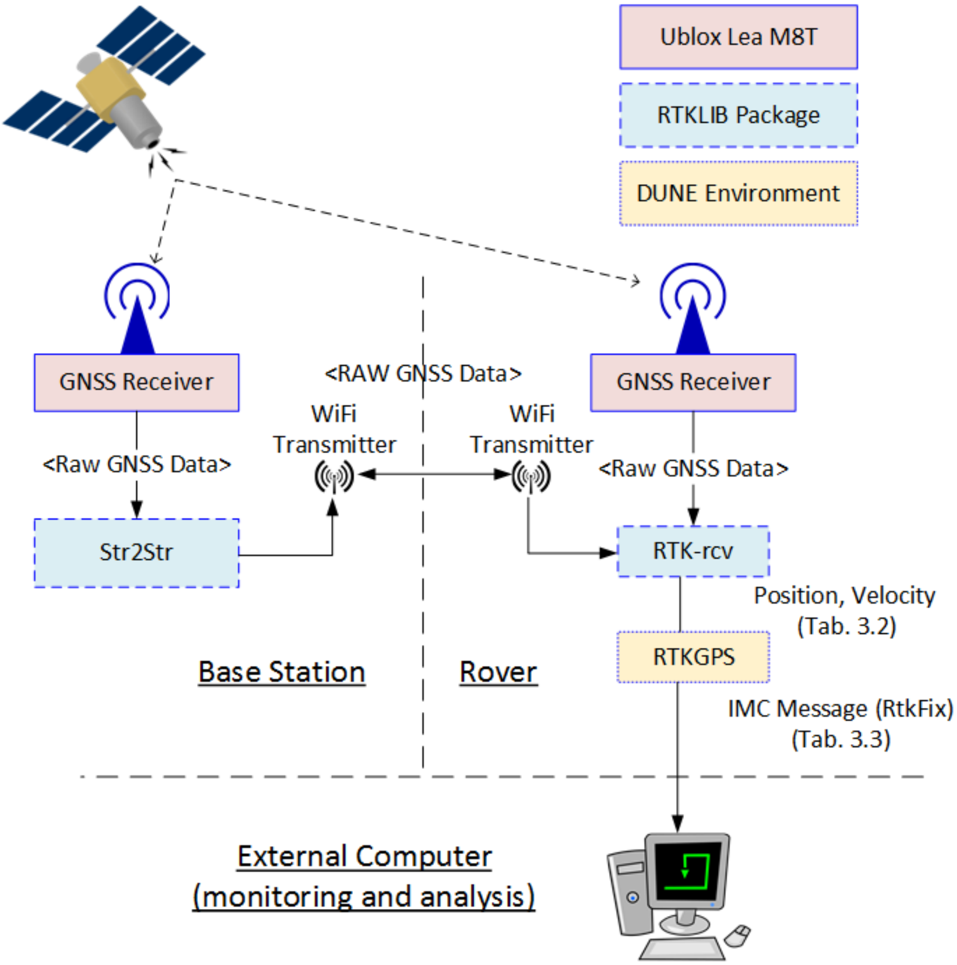

Ublox Lea M8T
RTKLIB Package
DUNE Environment
GNSS Receiver
<Raw GNSS Data>
Str2Str
<RAW GNSS Data>
WiFi Transmitter
WiFi Transmitter
GNSS Receiver
<Raw GNSS Data>
RTK-rcv
Position, Velocity (Tab. 3.2)
RTKGPS
IMC Message (RtkFix) (Tab. 3.3)
Base Station
Rover
External Computer (monitoring and analysis)

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.

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

This chapter contain the technical specification of the navigation system and landing path system, in addition to the payload installed in the X8 and both nests. The main focus of this thesis is the implementation of the landing path system and the navigation system, including user interface for both system in Neptus. A simplified model of the DUNE system is shown in figure 5.1, where the task LandingPlan creates the landing path, RTKGPS dispatch the RTK-GPS solution, Navigation decide the navigation source, Path Control creates desired control input to the low level controllers and Ardupilot which communicates with the Pixhawk.

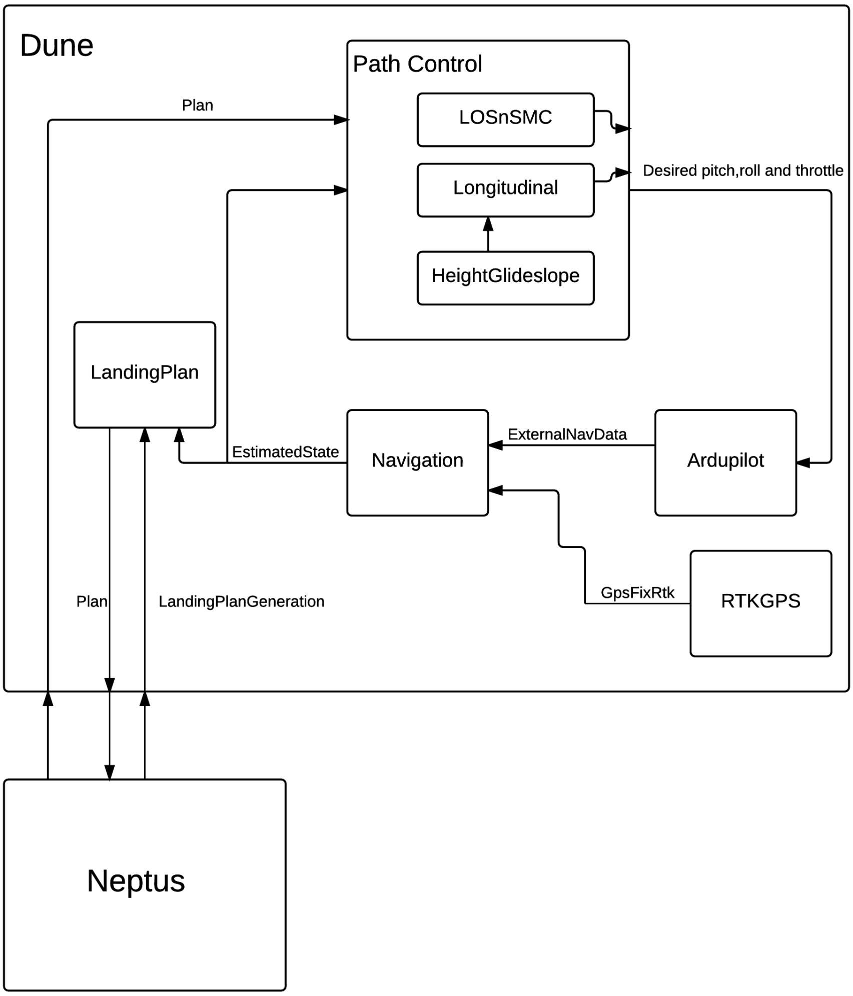

Dune
Path Control
Plan
LOSnSMC
Longitudinal
Desired pitch,roll and throttle
HeightGlideslope
LandingPlan
EstimatedState
Navigation
ExternalNavData
Ardupilot
GpsFixRtk
RTKGPS
Plan
LandingPlanGeneration
Neptus

Figure 5.1: A simplified figure of the Dune auto land system

5.1 Landing path

The landing path system is design to start generation of a landing path when the IMC message LandingPlanGeneration message is dispatched in the DUNE system. The IMC message was created to ease the configuration of the landing path, and includes option on how the path should be created. Among the parametrisation options there is the possibility to manually decide the rotation direction of both the start and finish circle in the approach path. This option allow the user to create a

Parameter name	**Action**
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

desired path, which not necessary is the shortest path. The behaviour parameters in the LandingPlanGeneration message is given in table 5.1. In addition the user can specify that the UAV should wait in a loiter manoeuvre at the end of the approach path before continuing towards the landing path. This is useful when performing dynamical landing where the position of the landing point is not fixed. In the case of performing a dynamic landing the land path system will only create the approach path, and not the landing path. This was found out to be a preferable solution since a dynamical landing require a feedback loop to correct the desired path, which is currently not included in the landing path system. A solution for performing a dynamical landing is currently research by fellow Master students where the multi-copters is used to catch the UAV, where this landing system is used to create a approach path to ready the UAV for a dynamic landing.

From Neptus the plug-in LandmapLayer, which is an altered version of Neptus plug-in developed in thesis [Frølich, 2015]. Alteration in the plug-in include new parameters, the inclusion of the IMC message LandingPlangeneration and the abillity to manually write the global position coordinates of the net.

5.2 Navigation system

The navigation system is control by a state machine 3.2.2, which is used to control the content of the output IMC messages EstimatedState and NavSources. Depending on which state the navigation system is in 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-GPS solution is dispatched from the DUNE task RTKGPS, however before the message is accepted by the Navigation task the message must include a valid base station position. The base station position is not included in the output message from RTKlib, which demand the base station position to be calculated locally at the base station as a standalone GNSS receiver. For this purpose the DUNE task BasestationFix is used to lock the current position of the base station, which result in the base station position being transmitted to the RTKGPS task. The navigation system require to now the reference position of the base station in order to use the RTK-GPS solution. However the base station position is currently not part of the output message from rtkrcv. This is resolved by allowing the base station to calculate it's own position as a standalone GPS. The GPS position is transmitted to a local Dune task on the base station, where the operator can decide when the base station can be considered as fixed. When the base station is considered fixed the position is sent to the X8, where it's included in the RTK-GPS solution message.

Nest system

A nest system is a stationary unit with the sole purpose of providing it's position to the rest of the Dune System. As part of the navigation system the base station is defined as a nest, where the GPS position is sent to the RTK-GPS system when fixed. An other nest has been created to obtain the GPS position of the stationary net. The net nest is configured as a rover in RTK-GPS configuration, such that the position relative to the base station is in the same frame as the X8.

5.2.2 Operator interface

The state of the navigation system is monitored though a interface in Neptus. The interface indicate which source the Dune system is using for state information. The interfaced apply a color code to indicate which source is currently in use in addition to all sensor system that are available, as seen in table 5.2.

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

Table 5.2: Net approach parameters

Chapter 6

SIL test

The system was tested through a Software In the Loop (SIL) simulation. In a SIL simulation the same code that runs in the hardware is tested against a simulator which behave similarly to the real system.

6.1 Setup

When performing a SIL test the Dune system is locally connected to ardupilot, which is communicating with JSBsim instead of the real UAV. The simulation is performed with a mathematical model of the X8 developed in the Master thesis [ref master thesis]. The purpose with the SIL test is to test the same software which will run in the X8 during a experimental test, and is used to verified that the system works. However since it's a simulated environment the results can only be considered as ideal results.

6.2 Path planing SIL

The landing path is tested in both guided and FBWA.

6.3 Navigation SIL

The navigation system was tested by creating a RTK-GPS simulator. This allowed for testing of the navigation interface, including the fixing of the base station position.

6.3.1 Short rtk loss compensator

The short rtk loss compensator was tested by adding a bias in the position solution from the simulated GpsFixRtk message.

Chapter 7

Experiment

7.1 Path

7.1.1 Landing path with pixhawk autopilot

A landing path was create with the goal of testing the concept of the landing path system. The test was perform with the pixhawk autopilot, and navigation system. Thus high performance from the X8 is not expected. The path was create in cooperation with the operator and pilot, which dictated the maximum size of the landing path.

The height where the UAV held after the approach path was to low, and during a real landing would cause the UAV to crash. Thus either the length of the landing path or the glide slope angle has to increase. The performance of the pixhawk autopilot was not adequate to successfully hit the net.

The wind condition was measured to be $5m/s$ from the north, which caused the UAV to have to perform the landing with a cross wind.

The path was created with manually deciding the rotation directions, due to the start position and where the operator wanted the UAV to circle down to the correct height. In this case the shortest path would bring the UAV close to a no-fly zone

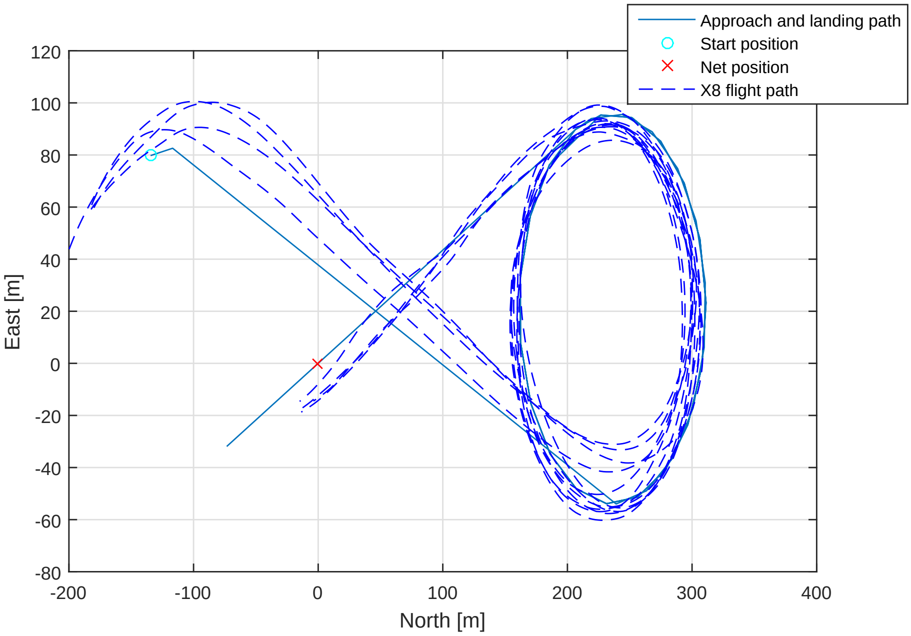


Figure 7.1: Four lateral flight paths towards the net position

7.2 Navigation

The navigation system was tested in both a X8 and in a multicopter system.

7.2.1 Formation position

The accuracy of the RTK-GPS position system was tested with two mutlicopter system, with the goal of measuring the distance between them. During stationary conditions it was determined that RTK-GPS was highly accurate, and is required when performing a formation flight. The same principle is used in determining the net position, where the net nest apply RTK-GPS to determine it's own position. However currently the position has to manually be written to Neptus, which must change when performing a ship landing.

7.2.2 Short loss compensator

Include result when short loss was not in use. Compare to result when it's in use.

The short loss compensator bring the position solution from the pixhawk close enough to the RTK-GPS solution such that the navigation system does not lose its position. However a highly aggressive control system that attempt to keep the error at centimeter level is expected to react to the discontinues introduced from the compensator. All though currently it has not been concluded that the compensator cause problems for the control system, since failure has happen during demanding wind condition.

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.

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.

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.

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.

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

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

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

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

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

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

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

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

Parameter	**Value**
Net latitude	$64,62848193735199\,\mathrm{deg}$
Net longitude	$9,726324254172361\,\mathrm{deg}$
Net height	$150m$
Net height offset	$-3m$ (NED frame)
Net heading	$66.5\,\mathrm{deg}$
Distance behind net	$80m$
Final approach length	$15m$
Final approach angle	$3\,\mathrm{deg}$
Glide slope length	$200m$
Glide slope angle	$3\,\mathrm{deg}$
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

.1 Landing plan parameters

.1.1 Parameter used in experiment with ardupilot

Appendix A

Rtklib Configuration

Configuration of RTKLIB,rtkrcv is given bellow.