Autopilot Design and Path Planning for a UAV

HENRIK GRANKVIST

FOI
Defence Research Agency
Defence and Security, Systems and Technology
SE-164 90 Stockholm

Phone: +46 8 555 030 00
Fax: +46 8 555 031 00

www.foi.se

FOI-R--2224--SE
ISSN 1650-1942

Scientific report
December 2006

Defence and Security, Systems and Technology

Henrik Grankvist

Autopilot Design and Path Planning for a UAV

FOI-R--2224--SE Scientific report
ISSN-1650-1942 December 2006

Defence and Security, Systems and Technology

Issuing organisation FOI – Swedish Defence Research Agency Defence and Security, Systems and Technology SE-164 90 STOCKHOLM	**Report number, ISRN** FOI-R--2224--SE	**Report type** Scientific report
	Research area code Operational Research, Modelling and Simulation	
	Month year December 2006	**Project no.** A64043
	Sub area code Unmanned Vehicles	
	Sub area code 2 Air Vehicle Technologies	
Author/s (editor/s) Henrik Grankvist	**Project manager** Ulrik Nilsson	
	Approved by Helena Bergman	
	Sponsoring agency FÖ	
	Scientifically and technically responsible Magnus Lindhé	
Report title Autopilot Design and Path Planning for a UAV		

Abstract

On the market there are many Unmanned Aerial Vehicle (UAV) airframes and many autopilot packages. This masters degree project aims at evaluating the combination of the SmartOne airframe and the MP2128^g autopilot and to develop path planning algorithms, i.e. search and track. To do this the UAV was modeled with analytical aerodynamics and then simulations were performed. This gave a set of control gains for the autopilot's stabilizing part as well as a reference for future flight testing and validation. To supplement a larger part of flight testing with simulation is a cost effective way of achieving a well tuned autopilot. The path planning algorithms were constructed, the search algorithm is a versatile and robust pattern that can be easily applied to an area. The tracking algorithm is a novel approach using target velocity estimates to perform close tracking.

Keywords

aerodata, UAV, simulation, path planning

Further bibliographic information	**Language** English
ISSN-1650-1942	**Pages** 44 p.
	Price acc. to pricelist

Utgivare FOI – Totalförsvarets forskningsinstitut Försvars- och säkerhetssystem 164 90 STOCKHOLM	**Rapportnummer, ISRN** FOI-R--2224--SE	**Klassificering** Vetenskaplig rapport
	Forskningsområde Operationsanalys, modellering och simulering	
	Månad år December 2006	**Projektnummer** A64043
	Delområde Obemannade farkoster	
	Delområde 2 Flygfarkostteknik	
Författare/redaktör Henrik Grankvist	**Projektledare** Ulrik Nilsson	
	Godkänd av Helena Bergman	
	Uppdragsgivare/kundbeteckning FÖ	
	Tekniskt och/eller vetenskapligt ansvarig Magnus Lindhé	
Rapportens titel Autopilotdesign och Banplanering för en UAV		

Sammanfattning

Det finns många kommersiella obemannade flygplan, UAVer, på markanden, det finns också många autopilotlösningar. Det här examensarbetet syftar till att utvärdera kombinationen av planet SmartOne och autopiloten MP2128^g samt att utveckla banplanneringsalgoritmer, nämligen sökning och bevakning/följning. För att åstadkomma detta konstruerades en simuleringsmodell av planet, med analytisk aerodynamik, och sedan genomfördes simuleringar. Detta gav en uppsättning av regulatorförstärkningar till autopilotens stabiliserande del samt en referens till framtida flygtestning och validering. Som ersättning till riktiga flygtester är simuleringar ett snabbt sätt att uppnå en bra inställning av autopiloten. Sökalgoritmen skapar en mångsidig och robust flygbana, som sedan enkelt kan appliceras på ett område. Bevakning/följningsalgoritmen är en ny lösning som använder ett estimat av målets hastighetsvektor för att åstadkomma bra målföljning.

Nyckelord aerodata, UAV, simulering, banplanering	
Övriga bibliografiska uppgifter	**Språk** Engelska
ISSN-1650-1942	**Antal sidor:** 44 s.
Distribution enligt missiv	**Pris: Enligt prislista**

Contents

Acknowledgments

I would like to thank my examiner Associate Professor Karl Henrik Johansson and my advisor PhD Student Magnus Lindhé at Automatic Control Lab at KTH. I would especially like to thank T.Lic Ulrik Nilsson, my advisor at FOI for all the help and support. Furthermore I would like to thank the people at FOI, in particular Peter, Petter, Jonas, Dag, John, Bengt, Leif, Thomas and Hugo for help big and small and to vice department head Martin Hagström for letting me do this project.

1 Introduction

UAV-Unmanned Aerial Vehicles have been around longer than one might think. The first UAV, the Hewitt-Sperry automatic aircraft, dates back to the first world war. However, it is only more recently that the UAVs on the market has reached a state of practical use. As with many other technical systems, the introduction of light weight structures and dependable, cheap computers made it possible to make simple UAVs work satisfactory. As will be apparent there is a world of difference between the UAV used in this project, compared with for example the X-45 UCAV, C is for Combat, that NASA is working on. X-45s are able to, in groups with distributed decision making, autonomously strike at a priori known as well as unknown targets, perform self diagnostics of crucial systems and act accordingly [17]. There are two ways to UAV designing, one, like X-45, is to take an advanced airframe and incorporate enough software to make the pilot obsolete. The other way, like SmartOne, is to take an RC model and automate functions such as stabilizing and navigation. The main difference, except for the price tag, is the level of autonomy. When discussing UAVs, one is generally interested in:

1. Level of Autonomy
2. Rules and Regulations
3. Capabilities.

To the aeronautical engineer it might seem odd that performance does not make the list. The current bottleneck in UAV development is the immaturity of the autonomy technology field [18]. Another obstacle is the regulations regarding flight in controlled airspace, where the UAVs are considered a hazard. Manned aircraft depends on the pilot to handle situations. The UAVs are currently lacking that amount of decision making and are therefore, by default, banned until they can be proven safe. A rather clever, but also enormously limiting way around this is to construct the UAV to limit damage if it should crash. That is to make it non lethal to humans and make the equivalent impulse as a bird strike to another aircraft. The SmartOne with the low weight and speed in combination with pushing propeller, rounded edges and soft material is considered safe.

In the UAV field, current research concerning path planning is the hot topic, the fore front involving multiple agents making distributed decisions at a high level of autonomy. The subcategory of search is focused on optimality to guarantee coverage or to minimize mission time. The area most interesting for this thesis is guaranteed area coverage in short time [11], [12],

[13] and [14]. Another field of interest is the topic of tracking and in the current research this often involves line of sight constraints, sensor limitations, threats and fuel efficiency [8], [9] and [16]. For this project none of the above was considered, to stay in the proximity of the target was defined as good enough. More on these topics in section 5.

The goal with this project is to assemble an autonomous unmanned aerial vehicle, UAV. The UAV is to incorporate simpler abilities, such as search of a predefined area and to track a target, mobile or stationary. The UAV is to be controlled by an autopilot using cross accelerations as control input. Furthermore, the UAV is to have an interface that makes it easy to test new or improved abilities in upcoming projects. In order to accomplish a cost effective testbed, off-the-shelf products and free software have a central role. The physical UAV is to be thoroughly modeled in order to make future simulations accurate.

SmartOne is the airframe used here which was originally developed by Swedish Agricultural University, SLU, as a vehicle for aerial photography. That is still one possible use, as the search algorithm coincides with an aerial photo sweep. Other possible uses are surveillance/monitoring of traffic, accident sites and protected sites but also of individual cars, boats etc.

A possible scenario for this UAV is: A person is lost at sea somewhere in the Stockholm archipelago, the coast guard quickly divides the most likely waters into a search area definition and launches the SmartOne. An operator monitors the photos relayed from the UAV, after some time he detects an orange spot in the water and designate it as a target for the UAV. The UAV switches to tracking mode and lingers around the target, continuously relaying images and position, until the coast guard arrives and picks the person up and recovers the plane.

The report is divided into three distinct parts. Chapters 3 through 5 are modeling and simulation, chapter 3 having more of an background character, whereas chapters 4 and 5 are the core of the project and contains the solutions to the problem definition. Sections 2, 6 and 8 are about the physical UAV, implementation aspects and future tests. The conclusions are found in section 7.

2 Description of SmartOne and Micropilot

The combination of MicroPilot MP2128^g [1] and SmartOne, see Figure 2.1, springs from the goal of trying cheap technology and off the shelf products. One of the main reasons for using this particular airframe is that it is type approved by Swedish aviation authorities, this is due to the low weight, low speed and soft rounded edges. The airframe is considered inherently safe for people and property. One of the reasons for using MicroPilot is the competitive prizing.

2.1 MicroPilot Performance, Computational Constraints

Micropilot MP2128^g is an off the shelf autopilot which stabilizes and navigates the aircraft. It is not built for more violent or advanced maneuvers such as rolling or looping. Among its sensors are a six degree of freedom Inertial Measurement Unit, IMU, airspeed sensor and a GPS receiver. The onboard packages are very light weight, and the autopilot board measures only ten by four centimeters. It has servos with servo mixing capabilities for handling the elevon configuration of SmartOne. The servos have an update rate of 50 Hz and the control system has a frequency of 30 Hz. Moreover, the PID regulators for stabilizing and navigation the aircraft allows gain scheduled for different speeds and also feed forward on some circuits, e.g aileron-elevator in a turn scenario.

All in all there are 16 PID loops to tune in order to achieve the desired/best performance. Software for programming, routes and autopilot performance testing is included, as are data log capabilities. The onboard calculating capacity is about five operations per second, however that rate is not constant since piloting takes priority.

One extension to this package is the XTENDER^{mp} software development kit, which allows the autopilot to be modified and also to enhance the possibilities with ground control systems, GCS.

2.2 SmartOne Performance, the Flight Envelope

SmartOne is a fairly small aircraft of the flying wing configuration, with a wing span of 1.2 meters, see Figure 2.1. It was developed by SLU to perform photo reconnaissance of forests and cash crop fields. It has a cruise speed of 12 m/s and an endurance of up to one hour. See Table 2.1 for specifics. The

UAV is fitted with an electric engine with a push propeller producing 260 W. The minimum turn radius is 3 m, for short time periods, and its maximum climb rate is 5 m/s.

Notation	Value	Property
S_{ref}	0.33 m^2	Reference area
c_{ref}	0.276 m	Reference chord
b_{ref}	1.2 m	Reference span
x_{aero}	-0.201 m	Distance to aerodynamic center
x_{cg}	-0.18 m	Distance to center of gravity
m	0.9 kg	Mass
I_{xx}	0.0306 kgm^2	Moment of inertia
I_{yy}	0.0180 kgm^2	Moment of inertia
I_{zz}	0.0251 kgm^2	Moment of inertia
P	260.0 W	Power
V_{max}	22.0 m/s	Max speed
V_{min}	9.0 m/s	Stall speed
δ_{max}	20.0°	Max deflection down
δ_{min}	20.0°	Max deflection up

Table 2.1: SmartOne properties

Figure 2.1: SmartOne on Rye

2.3 System Overview

The complete UAV system including ground station and complementary equipment is described in this section, see Figure 2.2 for a schematic of the UAV.

The Main power supply is a Lithium-Polymere battery 11.1 V and 2200 mAh which supplies the servo board, the servos, the engine and the engine control. The auxiliary battery, also lithium-polymere battery, 7.4 V and 1800 mAh supplies the MP2128^g, the radio modem and the RC-receiver. The inputs to the MP2128^g comes from the pitot tube and from the GPS antenna, as well as the built in 6-DoF IMU. The output is directed via the servo board to the engine control and the control surface servos. No payload or extra equipment were integrated here. The RC signal is directed through the MP2128^{g}to the servo board, when the Pilot In Control (PIC) mode is active.

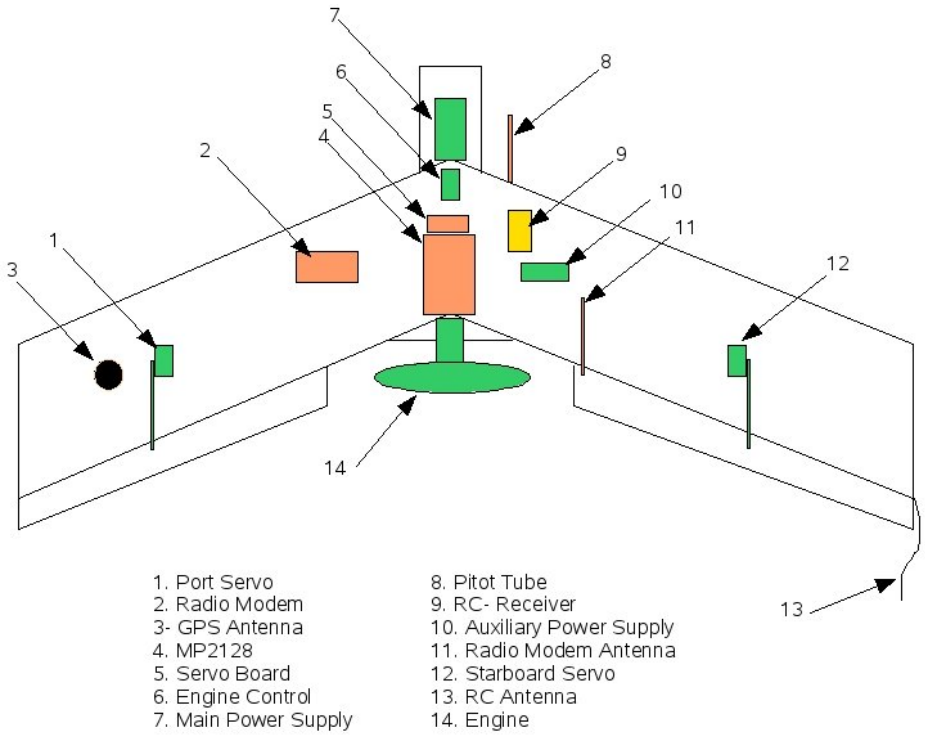

Figure 2.2: Schematic view of the UAV Main Components

3 Modeling the Aircraft and the Autopilot

In this section the simulation model is presented along with methods to obtain necessary data. The goal with the model and the following simulations is to achieve control parameters for MP2128^g and to get an estimation of aircraft performance. An aircraft is a full six degrees of freedom system, since it has no spatial constraints. Such a system is usually described by rigid body dynamics and most of the forces acting on the aircraft are aerodynamic. In the interaction between the structure and the aerodynamics, aeroelastic phenomena arise. The structural dynamics of a flexible aircraft are however neglected here. An additional assumption is that the earth is flat and stationary. The calculating unit of the autopilot work in discrete time domain, but the implications thereof are also omitted as of little consequence. The software used to build the model is Ptolemey II [19]. It is a java based simulation freeware developed at Berkley university. The appearance is somewhat similar to that of Matlab-Simulink [20] and the choice of Ptolemey II over Matlab/Simulink was a monetary matter. The model is built of function/operation boxes and connected with wires that transmit inputs and outputs. Ptolemey II can operate in different modes, such as descrete event, continous time or petri nets, the one used here was continuos time. One pro of Ptolemey II is that it is very easy to set up a model, one can picture the different boxes of a physical system, such as engine and servos, and then implement them. Another pro is that Ptolemey II can handle a full nonlinear model. The cons are the poor help section, and the limited number of predefined functions, although that is redeemed by the fact that Matlab can be called by Ptolemey II.

3.1 Equations of Motion

Here is a brief explanation to the equations that operate in the dynamics box. The dynamics of an aircraft are described by the Newton-Euler equations of motion for a rigid body. The rate of change of the linear and angular momentum is equal to the total external force and moment around the mass center

$$F = ma, \tag{3.1}$$
$$M = I\omega, \tag{3.2}$$

where F is the force, a is the linear acceleration, m is mass, M is the moment, I is the moment of inertia and finally ω is the angular velocity. Aerospace applications use a number of different coordinate systems, for example stability axis and wind axis. The above equation is most conveniently given in

body fixed, cartesian coordinates, see Figure 3.1 for definition. The vectors $\boldsymbol{v}_B = (u, v, w)^T$ and $\boldsymbol{\omega}_B = (p, q, r)^T$ are the linear and angular velocities respectively, in the xyz order, and the positive directions are defined by the right hand rule.

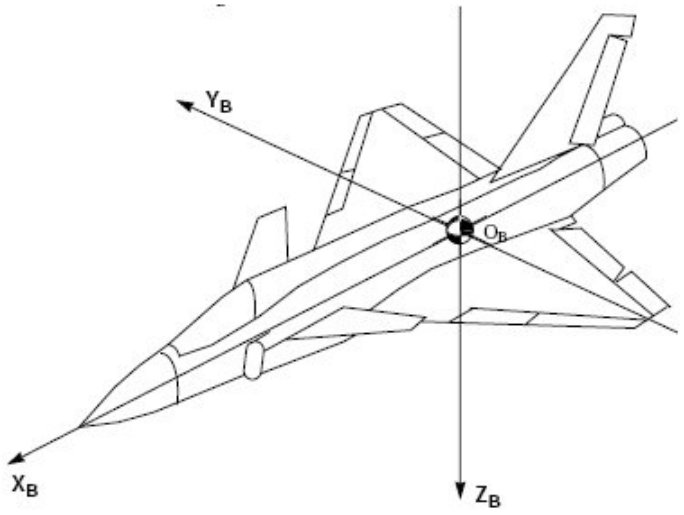

Figure 3.1: Definition of Body Axis coordinate system [22]

Here it is assumed that the origin is at the center of mass and mass and inertia are constant [4]. That is a fair estimation since no payload is going to be dropped and no fuel consumed. The plane is considered to be symmetric in the xz-plane, i.e. the I_{xy} and I_{yz} elements are zero, moreover it is assumed that I_{xz} is small in comparison to the diagonal elements. Thus

$$\boldsymbol{I} = \begin{pmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{pmatrix}$$

describes the inertia of the aircraft. Now (3.1) and (3.2) can be rewritten in vector form as,

$$\begin{aligned} \frac{1}{m}\boldsymbol{F}_B &= \dot{\boldsymbol{v}}_B + \boldsymbol{\omega_B} \times \boldsymbol{v}_B - \boldsymbol{g}, & (3.3) \\ \boldsymbol{M}_B &= \boldsymbol{I}\dot{\boldsymbol{\omega}}_B + \boldsymbol{\omega}_B \times \boldsymbol{I}\boldsymbol{\omega}_B, & (3.4) \end{aligned}$$

where $\boldsymbol{F}_B$ are the forces, $\dot{\boldsymbol{v}}_{\boldsymbol{B}}$ is the linear acceleration and $\boldsymbol{M}_B$ are the moments, $\dot{\boldsymbol{\omega}}_B$ is the angular acceleration and $\boldsymbol{I}$ is the matrix of inertia. The angle of attack α and sideslip β are defined as the deviation of the body x-coordinate axis from the velocity vector $\boldsymbol{v}_B$

$$\begin{aligned} u &= V\cos\alpha\cos\beta, \\ v &= V\sin\beta, \\ w &= V\sin\alpha\cos\beta, \end{aligned} \qquad (3.5)$$

where the absolute velocity V is

$$V = \sqrt{u^2 + v^2 + w^2}. \tag{3.6}$$

The total external force F is composed of aerodynamic forces, thrust and gravity. Thrust is usually incorporated with the aerodynamic forces in the vector $\boldsymbol{F}_B$, since they have constant directions with respect to each other. The gravity vector $\boldsymbol{F}_g$ is defined by

$$\boldsymbol{F}_g = [-g\sin\theta, g\cos\theta\sin\phi, g\cos\theta\cos\phi] \tag{3.7}$$

where θ and ϕ are the pitch and bank Euler angles respectively. Now let $\boldsymbol{F}_B$ = $(X, Y, Z)^T$ be the aerodynamic forces in the body frame, and similar let $\boldsymbol{M}_B$ = $(L, M, N)^T$ be the moments. Thus one can rewrite (3.3) and (3.4) to a system of equations that can be oriented arbitrary in relation to the earth.

3.2 Aerodynamic Model

In this section the aerodynamic coefficients and their build-up are explained. The aerodynamic data set was supplied by the Aerodynamicists at FOI. In order to find the aerodynamic coefficients without doing experiments, the computer program Athena Vortex Lattice, AVL, was used. AVL is a free software [21] based on the extended vortex lattice method. This means that only inviscid fluid mechanics are used, i.e. friction and stall can not be predicted. The things that AVL do calculate are aerodynamic forces and moments, with their derivatives with respect to angles, rotations and controls. Furthermore Trefftz plane analysis is used for the induced drag. The friction drag was estimated using similar aircrafts as guidelines, there are also a few empirical equations to approach this subject. The aerodynamics department being a highly experienced group, the confidence in the accuracy of their data is high.

3.2.1 Aerodynamic Coefficients

The forces are given on dimensionless coefficient form, for instance for C_L the lift coefficient according to

$$C_L = \frac{L}{q_\infty S_{ref}}, \tag{3.8}$$

where L is the lifting force, q_∞ is dynamic pressure and S_{ref} is the projected wing area including wing box.

Similar operations are done for the moments but with an additional reference length as well as the reference area. Wing chord is used for longitudinal moment and the wing span for lateral moment. For pitching moment coefficient C_m

$$C_m = \frac{M}{q_\infty c_{ref} S_{ref}}. \tag{3.9}$$

Except for the force and moment coefficients the stability derivatives are of great interest, i.e. how different forces and moments change with respect to

angular velocity and change of geometry, e.g a control surface deflection. To exemplify, the lift dependency on α is shown, the notation is $C_{L\alpha}$ and it is defined by

$$C_{L\alpha} = \frac{\partial}{\partial \alpha}\left(\frac{L}{q_\infty S_{ref}}\right). \tag{3.10}$$

See Table 3.1 for the data set of SmartOne used in the model. The definition for control surface deflections state that elevator deflection is positive with trailing edge deflected down, and aileron deflection is positive when the surface on the right wing is deflected downwards. These deflections are the ones creating negative moments around respective axis.

Notation	Value	Property
$C_{D,0}$	0.02	Zero Lift Drag
k	0.11	Efficiency factor
$C_{L,0}$	0.071	Lift at zero AoA
$C_{L\alpha}$	3.224	Lift due to AoA (rad^{-1})
$C_{L\delta_e}$	0.7377	Lift due to elevator (/rad)
$C_{C\beta}$	-0.1555	Sideforce due to sideslip (rad^{-1})
$C_{l\delta_a}$	-0.1874	Roll moment due to aileron (rad^{-1})
$C_{l\beta}$	-0.1638	Roll moment due to sideslip (rad^{-1})
C_{lp}	-0.3387	Roll damping due to roll rate (s/rad)
C_{m0}	-0.0071	Pith moment at zero AoA
$C_{m\alpha}$	-0.1399	Moment due to AoA (rad^{-1})
$C_{m\delta_e}$	-0.5347	Moment due to elevator (rad^{-1})
C_{mq}	-1.2713	Pitch damping due to pitch rate (s/rad)
$C_{n\beta}$	0.0544	Yawing moment due to sideslip (rad^{-1})
C_{np}	-0.1004	Yaw damping due to roll rate (s/rad)
C_{nr}	-0.0260	Yaw damping due to yaw rate (s/rad)
$C_{n\delta_a}$	-0.0111	Yawing moment due to aileron (rad^{-1})

Table 3.1: SmartOne Aerodynamics

3.2.2 Build-up of Body Axis Coefficients

Here the force coefficients in the aerodynamic block are assembled by their stability axis affiliation, namely drag, lift and side force and the moments roll, pitch and yaw. The stability axis system is the body axis system rotated around the y-axis so that the velocity vector is parallel to the x-axis. The build-up of stability axis force and moment coefficients are,

$$\begin{aligned}
C_L &= C_{L,0} + C_{L\alpha}\alpha + C_{L\delta_e}\delta_e, \\
C_C &= C_{C\beta}\beta, \\
C_D &= C_{D,0} + kC_L^2, \\
C_l &= C_{l\delta_a}\delta_a + C_{l\beta}\beta + \tfrac{b_{ref}}{2V}(C_{lp}p + C_{lr}r), \\
C_m &= C_{m,0} + C_{m\alpha}\alpha + C_{m\delta_e}\delta_e + \tfrac{c_{ref}}{2V}(C_{mq}q), \\
C_n &= C_{n\beta}\beta + C_{n\delta_a}\delta_a + \tfrac{b_{ref}}{2V}(C_{np}p + C_{nr}r),
\end{aligned} \tag{3.11}$$

where $b_{ref}/2V$ and $c_{ref}/2V$ are factors that normalize the damping contributions. To obtain the body axis force coefficients on use the aerodynamic angle α to transform the stability axis coefficients,

$$\begin{aligned} C_X &= -C_D \cos\alpha + C_L \sin\alpha, \\ C_Y &= C_C, \\ C_Z &= -C_L \cos\alpha - C_D \sin\alpha. \end{aligned} \tag{3.12}$$

$$\boldsymbol{F}_B = \begin{pmatrix} C_X S_{ref} q_\infty + T \\ C_Y S_{ref} q_\infty \\ C_Z S_{ref} q_\infty \end{pmatrix}. \tag{3.13}$$

Here the engine is incorporated as thrust, T, aligned with the x_B-axis. The leverage arm $x_{cg} - x_{aero}$ from mass center and the aerodynamic reference point are calculated and multiplied by the vertical force coefficient, C_Z. The same can be done for y and z axis, but those contributions are very small and are therefore neglected here. Similarly the moment coefficients are transformed by,

$$\begin{aligned} C_L &= C_l \cos\alpha - C_n \sin\alpha, \\ C_M &= C_m, \\ C_N &= C_l \sin\alpha + C_n \cos\alpha. \end{aligned} \tag{3.14}$$

Now the moments can be assembled,

$$\boldsymbol{M}_B = \begin{pmatrix} C_L S_{ref} b_{ref} q_\infty \\ C_M S_{ref} c_{ref} q_\infty + C_Z (x_{cg} - x_{aero}) S_{ref} q_\infty \\ C_N S_{ref} b_{ref} q_\infty \end{pmatrix}. \tag{3.15}$$

3.3 The Autopilot

In this section the MP2128^g is modeled and the structure used can be found in the MP2128^g manual [1]. The channel set for level flight was chosen as:

Throttle from airspeed,
Pitch from altitude,
Elevator from pitch,
Aileron from roll.

The overall structure is shown in Figure 3.2. The individual circuits are PID circuits, however since Ptolemey II do not have a derivative function an approximation was implemented. Also added is an anti windup on the integral parts. The inputs are the airspeed, altitude and bank angle errors respectively. The outputs are sent via the actuator dynamics box, where filters add first order dynamics, i.e. the control surface motions are not instantaneous. Thereafter the control surface deflections are sent to the aerodynamics block and the thrust is sent directly to the force and moment assembly box.

3.4 The Complete Model

The model is basically built by the components mentioned before. As one can see there are autopilot, aerodynamic and dynamic boxes, with some help

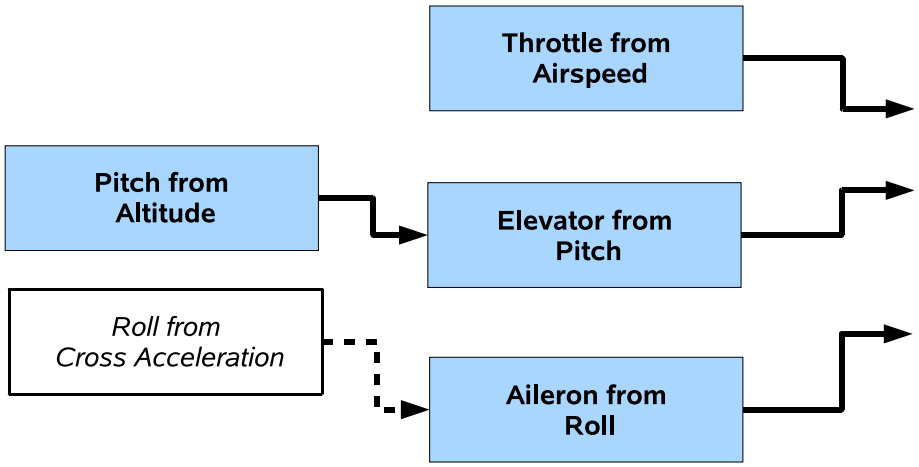

Figure 3.2: The autopilot structure, the white box is the intended input from the path planning algorithms

functions around them, see Figure 3.3 for a schematic overview. The accuracy of the simulation model is determined to a great extent by the parameters of the model, how well the inertia and areodynamics match reality for example. Once the model is done it can be used to simulate different flight conditions and maneuvers to tune the autopilot parameters.

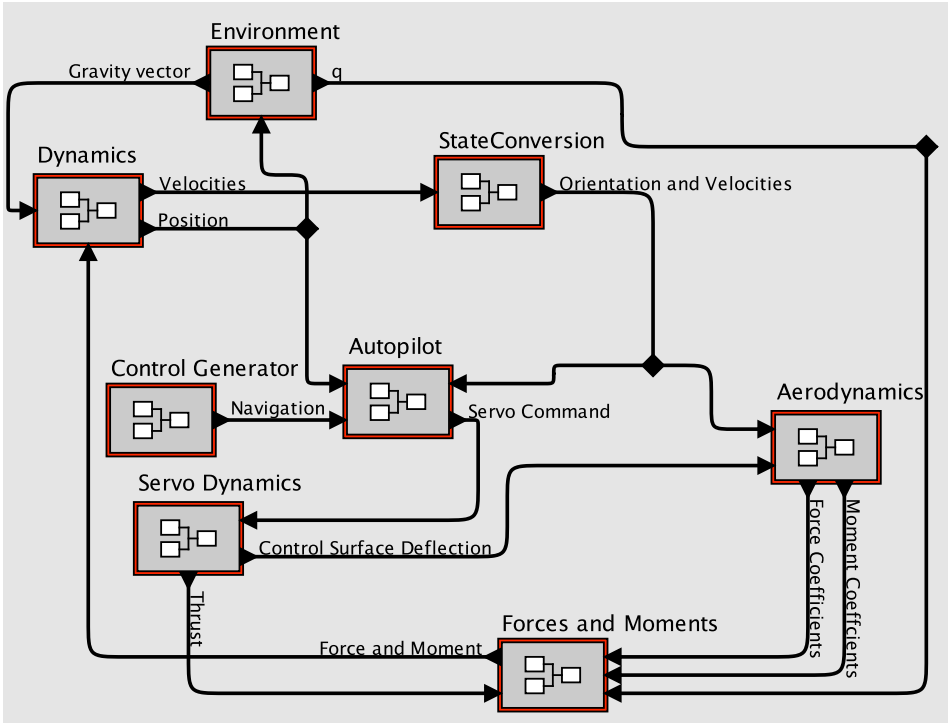

Figure 3.3: Simulation Model

4 Simulation of the UAV

In this section the simulation of the model is described, that is the stabilizing part of the autopilot and the dynamical characteristics of the UAV. All figures presented here are based on the final gains, which can be found in Table 4.1. There are several reasons for doing simulations. It is a good way to tune the controllers in the autopilot and it gives a good estimate on the capabilities of the UAV. Moreover it is a way to verify qualitatively that the aerodata set is reasonable. No other control design tools were used, simulations and tuning were iterated until the performance was acceptable from a flight mechanic point of view, e.g α was kept small. Two sets of stabilizing gains were developed. One with high gains for fast error suppression and one with low gains using less control power. This was done to have a backup in case the parameters from one of them would work poorly in MP2128^g. The higher gain configuration is named SmartOne V1 or for short V1, the low gain version is named SmartOne V2. In order to tune the autopilot, the model was started with the desired states equal to the initial states,

$$h_{Des} = h, \tag{4.1}$$
$$V_{Des} = V, \tag{4.2}$$
$$\phi_{Des} = \phi, \tag{4.3}$$

that is altitude, airspeed and bank angle respectively. For a set of control parameters the initial disturbance was damped out. The disturbances come from the fact that the model do not start in a trimmed condition. The response for step input, impulse input and 3-2-1-1 input was investigated for altitude, airspeed and bank angle. The 3-2-1-1 input is a square wave input, and defined as full positive control for three time units, full negative control for two time units and then followed by two one unit pulses plus and minus, to end up back at neutral. This input is quite common for parameter testing in the aeronautical field [15]. The square wave form contains a rich spectrum of frequencies and the bang-bang appearance excites the airframe, moreover it is easy to implement. The demands on the time domain responses were, fast rise time, small overshoot and a minimum of oscillations. That is to say a well behaved aircraft. Good enough performance was determined by trade-offs between interacting channels and system limitations.

4.1 Airspeed Control and Performance

In order to test the airspeed control performance a step response test was carried out, see Figure 4.1. The top plot is the step response, the performance

is fairly quick with an acceptable overshoot. As one can appreciate, the rise time here corresponds to the excess power of the engine, a step from 16 to 20 do not have the same response. One also have to bear in mind that the airspeed channel has to be able to handle disturbances from other channels, e.g altitude.

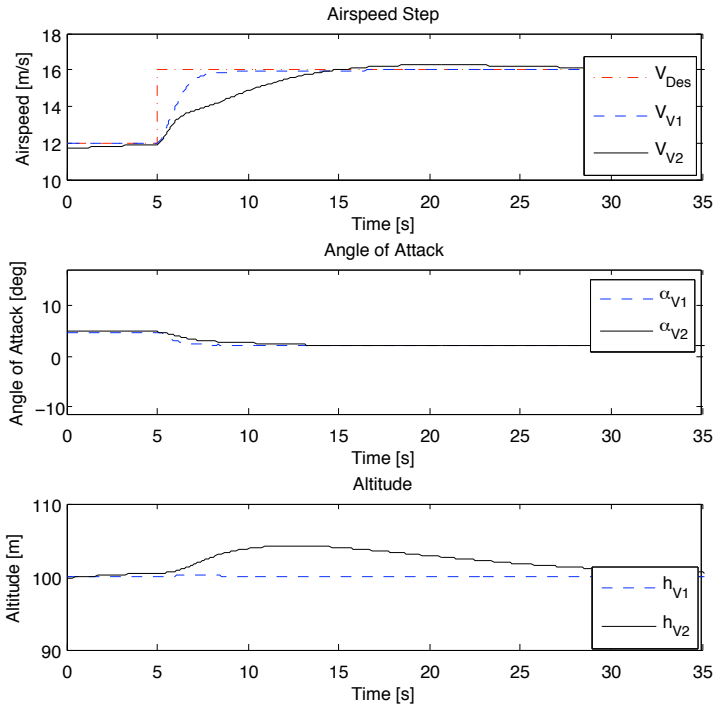

Figure 4.1: Airspeed Control and Performance

4.2 Altitude Control

Since altitude and pitch angle both are controlled by the elevator, the two channels are tested simultaneously. First is a step response, see Figure 4.2. The altitude follows the desired altitude nicely. One limiting factor here is, again, the engine power available, i.e. rise time has a very real, physical limitation. The other channels, airspeed and aerodynamic angles both shows acceptable reactions to this step. The oscillations in α for V1 peaks at 15° which might seem high, this is probably due to a too small pitch damping factor, which would also explain the high frequency. In order to test longitu-

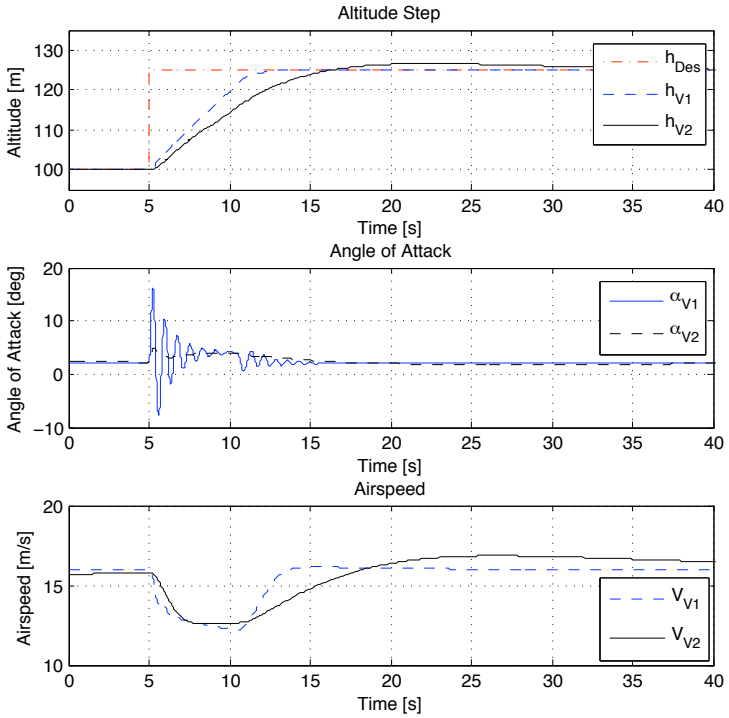

Figure 4.2: Altitude Step Response

dinal stability, the aircraft was subjected to a 3-2-1-1 input. The input is given as $h_{Des} = \pm 100m$ via the whole longitudinal control chain involving pitch from altitude and elevator from pitch, rather than a direct control surface command. The point with this investigation is to try to excite the longitudinal modes, short period and/or phugoid. It is done to see how the aircraft steadies itself after a brusque command sequence. In this case, see Figure 4.3, it is clear that the aircraft and the autopilot works satisfactorily. Checking back to altitude and airspeed control there are some disturbances, however an altitude error of 5 m is of small consequence, neither is the resulting speed loss, see Figure 4.4. The one output here that is of most interest is α which is within reasonable values for both configurations.

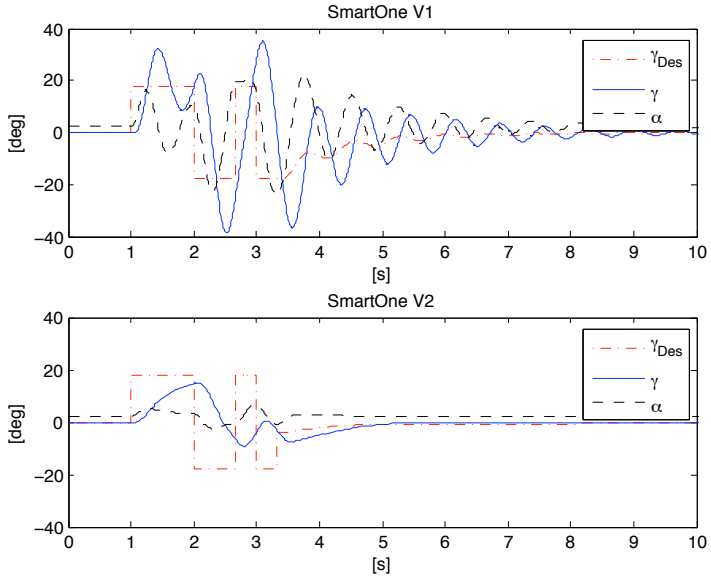

Figure 4.3: Aircraft Response to a 3-2-1-1 input in Pitch

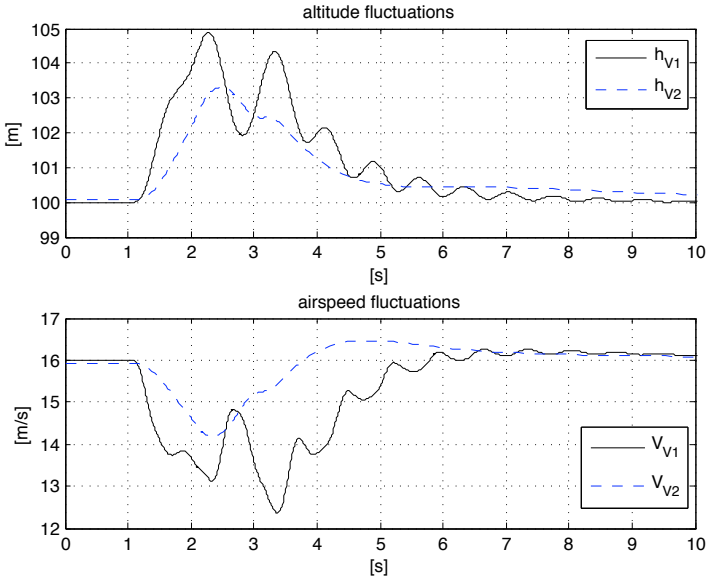

Figure 4.4: Aircraft Response to a 3-2-1-1 input in Pitch

4.3 Bank Angle Control and Turn Performance

Here the turning capabilities are tested along with the lateral stability. A pulse response was performed, see Figure 4.5. For this particular aircraft there is no rudder, which could be a problem, but as one can see β is small throughout the maneuver. This indicates that the aircraft is weathercock stable. There is an overshoot of approximately 20% for V1, a little less for V2, and some significant oscillating, but since the settling time is short, the behavior is acceptable. By examining the turn radius for a given bank angle, while also checking altitude and airspeed tracking, a smallest turning radius for sustained turn can be found. No specific demands were given, so it was decided that a bank angle of 15° is the limit. Of course, tighter turns are possible, for SmartOne, but the penalties in other channels are mounting. It is also possible that the MP2128^g might not be able to turn tighter, it is built for stabilizing the aircraft. For reference, the theoretical minimum turn radius [5], with load factor $n = 1.5$ and airspeed $V_\infty = 16$ m/s, gives a turning radius,

$$R = \frac{V_\infty^2}{g\sqrt{n^2-1}} = 23.3\ m \tag{4.4}$$

and a bank angle,

$$\phi = \arccos\left(\frac{1}{n}\right) = 48.2^\circ \tag{4.5}$$

The bank angle used as input here is 14.3°, see Figure 4.6 for resulting performance. This bank angle gives a turning radius of 100 m and it takes 10.15 seconds to perform a 90° turn, i.e. the duration of the pulse, see Figure 4.7 for ground track.

The lateral stability was also put to the test by a 3-2-1-1 input in desired roll to excite a possible dutch roll. As is shown in Figure 4.8, oscillations are quickly damped. In Figure 4.9 the resulting airspeed and altitudes are shown. The influence in both channels for both configurations are small.

4.4 Finalized Control Gain

A step in altitude affects airspeed to some degree which depends on the maximum allowed flight path angle γ_{max}. If γ_{max} is too big, i.e. a steep climb, the airspeed will drop and similarly in a dive the airspeed will rise. Given the step responses the PID parameters could be tuned with respect to overshoot, rise time, settling time and static error. By tuning the PID parameters one can shape the response, taking care so that the performance of affected channels are not substantially degraded. After doing the necessary tradeoffs the parameters used are found in Table 4.1.

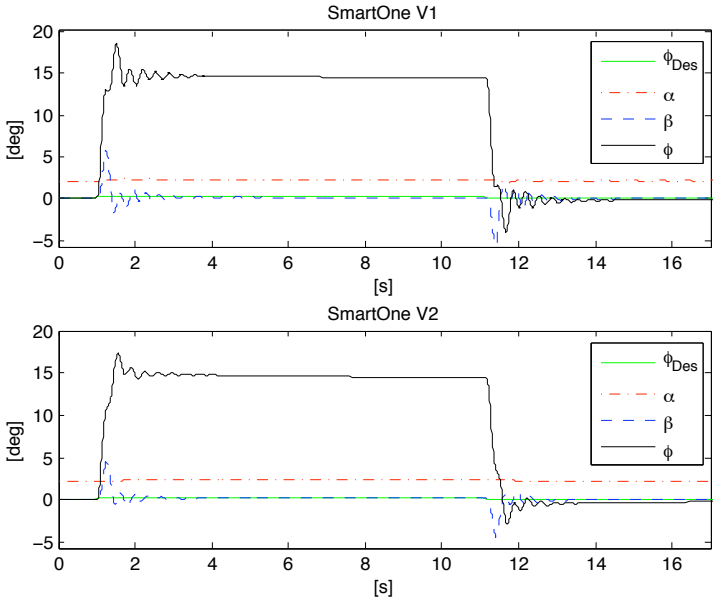

Figure 4.5: Bank Angle pulse Response

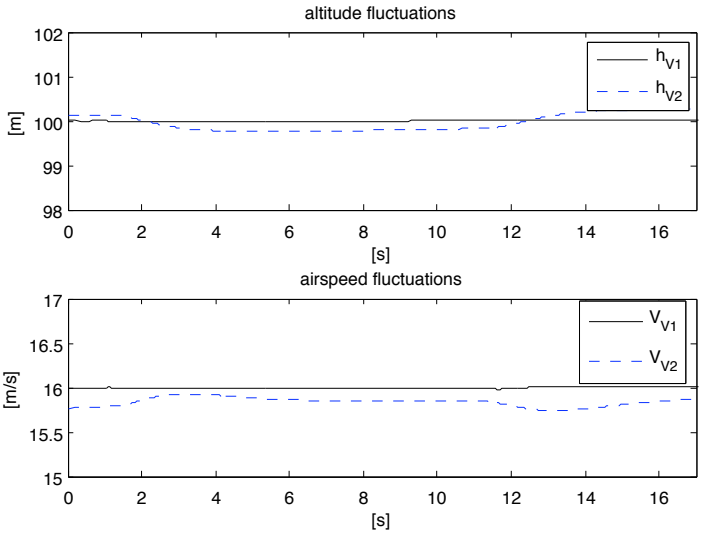

Figure 4.6: Turn Performance

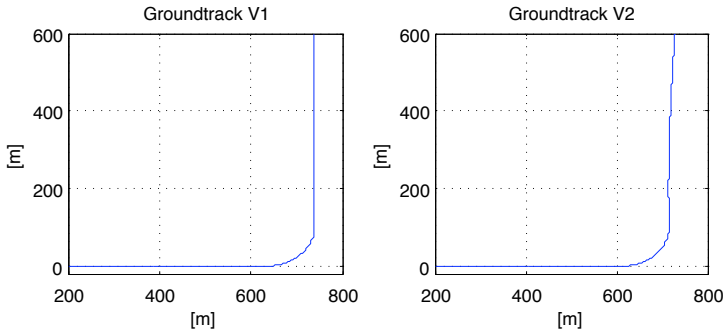

Figure 4.7: Ground Track

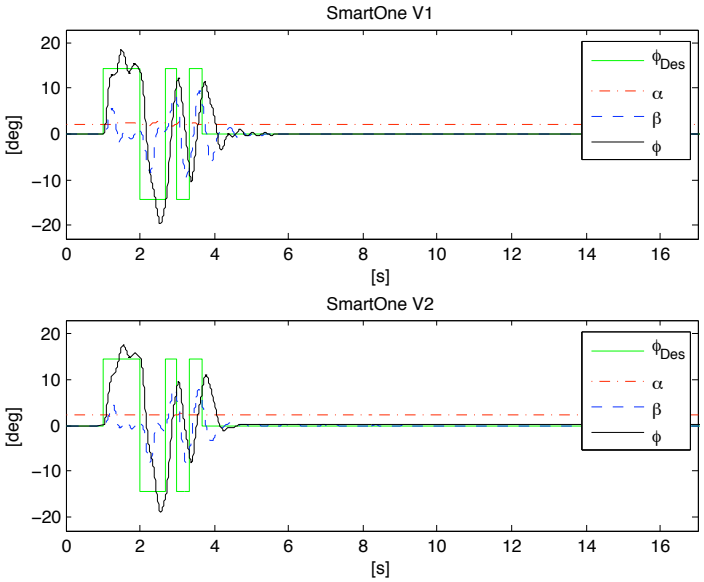

Figure 4.8: Aircraft Response to a Bank Angle 3-2-1-1 input

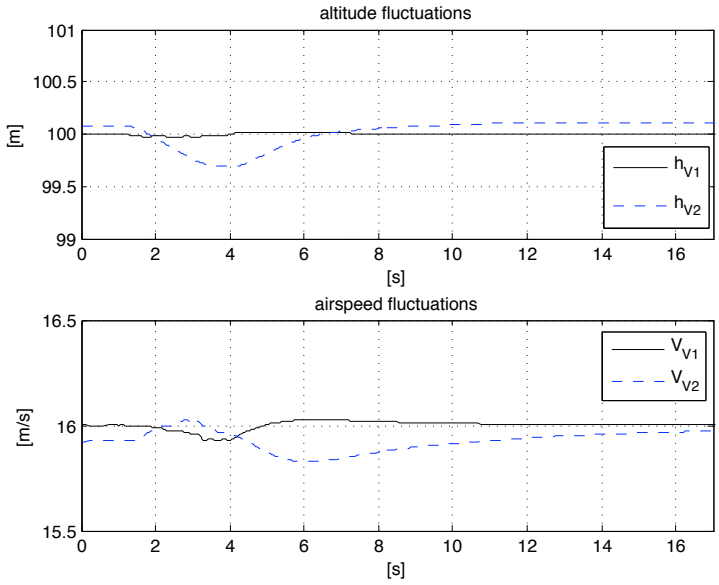

Figure 4.9: Aircraft Response to a Bank Angle 3-2-1-1 input

	Gain	V1	V2
Throttle from Airspeed Error			
Proportional	k_p	30	10
Integral	k_i	7.5	0.5
Derivative	k_d	0	0
Pitch from Altitude Error			
Proportional	k_p	-0.08	-0.02
Integral	k_i	0	0
Derivative	k_d	0	-0.01
Elevator from Pitch Angle Error			
Proportional	k_p	0.1875	0.094
Integral	k_i	0.4	0.01
Derivative	k_d	0.375	0.0094
Aileron from Bank Angle Error			
Proportional	k_p	-0.37	-0.38
Integral	k_i	-0.1	-0.28
Derivative	k_d	-0.56	-0.1

Table 4.1: Control Parameters

5 Path Planning

Path planning is the part of the autopilot that handles the navigation and consequently the mission, i.e the algorithms that determine where the UAV is going and also when. The path planning part can furthermore contain different parts, focused on different tasks, such as search. The mission statement for this thesis called for search and target tracking/following path planning capabilities. There are different approaches to construct these, one can program algorithms for the different tasks into the onboard autopilot or one can use a ground station and a radio link. The algorithm described here is meant to be programed into the onboard computer.

5.1 Search Algorithms

Here we shall treat patterns that basically covers areas, this means that the search can also mean photo reconnaissance or similar tasks. As stated in section 1 the research concerning search algorithms is a hot topic, however the ability asked for here may not be at the front of research. The approach proposed by Ablavsky et. al [11], [12], [13], is to calculate a map consisting of isochronal contours. Based on target mobility, terrain type and time elapsed since last time the location of the target was known, a map with probable locations can be created. That map is then divided into search zones of some simple shape, e.g elliptical. Here the scenario is looking for mobile Scud launchers, i.e firing position is known and also the local road network and speed of the trucks in terrain and on roads. For each subregion there exists an optimal search pattern i.e raster, zamboni or box-spiral. The near optimal global solution can then be found as a sequence of visits of the subregions where the entry/exit points are used as variables. The pattern most efficient for search is the raster pattern [12], which is the pattern used in the search algorithm here. But here there is no need for pre specified shapes. However, the transit between two or more areas here claim no optimality. One can also incorporate uncertainty models, as Bertuccelli argues, the probabilities are not exactly known to a mission planner in beforehand. The Uncertain Probability Map (UMP) take in account delayed information, erroneous information and the propagation thereof in time [14]. One more issue discussed is: *is flown over equal to detected?* For this algorithm an ideal sensor is used and the probability of detection is considered to be one. There are more similar types of optimization based algorithms and they all have one thing in common, they require a lot of computations.

5.1.1 Search Problem Statement

The definition of the search problem is:

> To search an area, so that a down looking sensor e.g a camera completely covers an area of arbitrary shape. The area may be divided into subareas.

There are no demands on time or path length optimality. The range of the sensor can be considered to be twice the turning radius.

5.1.2 Search Pattern

In this section the search problem is solved. The main theme is to apply a raster pattern, by construction it will cover the whole area and no two sweeps will intersect. This section is divided in two parts, first are the demands on area definition and starting points. Second is a description of the workings of the algorithm.

Too begin, here are the definitions of a few terms that are used here:

Sweep, the part of the flight where the UAV is going straight and can employ a downward facing sensor.
Sweep direction, the heading of the UAV while at a sweep.
Raster, a pattern built by a number of sweeps and 180° turns.
Raster progression direction, the direction in which the raster is going, i.e. perpendicular to the sweeps.
Origin of raster, the point the sweep is progressing from, the minimum point.
Break line, a line perpendicular to the raster progression, that intersects the corner point furthest away from the origin of the raster, the maximum point.

Demands on the Area Definition

This algorithm only works for convex polygons, or at any rate the subalgorithm that determines if a boundary is passed or not. Any area can be divided into polygons of arbitrary convex shape, this condition presents no limitation on the search area definition. Any number of polygons can be combined to fully describe the desired search area. The area is defined by its corner points.

The Algorithm in Brief

To successfully start the algorithm one needs the approach direction to be equal to the desired sweep direction. The approach direction dictates the sweep direction as $V_{Des} = V_{Approach}$, this is one of the more important parameters to pick, long sweeps are effective, short sweeps less so. The switch between approach and search can be at any point in or outside the polygon. Moreover, the approach have to intersect with the polygon no more than the sensor range from the point/line furthest away, in the negative sweep progression direction to ensure sensor coverage. The origin of the raster should

be the point, or any point on the line, furthest away from the first sweep, in the negative sweep progression direction.

Here it is assumed that the sensor is rendered more or less useless during a hard turn. Therefore the turn is performed outside of the polygon and measures have been taken to ensure that the hard turn is done when the UAV reenters the polygon. This is accomplished by using two shadow points, at a distance perpendicular to the velocity of the UAV, one on each side. When all three are outside, the reentry point is on the boarder.

The break condition is simply that the UAV and the shadow points, are past the break line.

To proceed to the next polygon the transition is made by adding the new approach to the *flyto* function. In Figure 5.1 the raster is shown over one polygon, also some of the terms used here are illustrated.

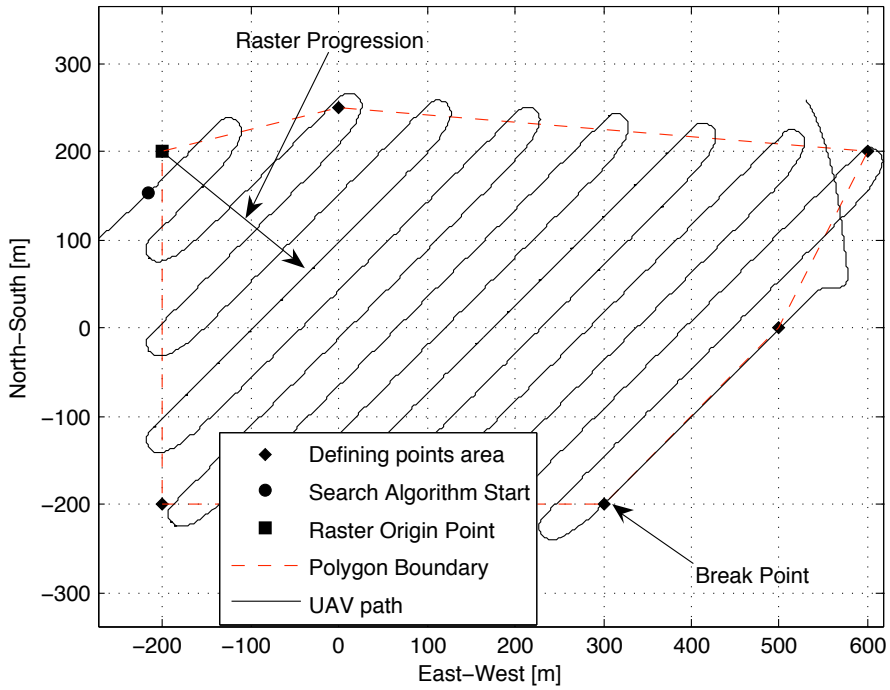

Figure 5.1: Raster Pattern on Arbitrary Polygon

Algorithm Governing Parameters

The governing parameters can be found in Table 5.1, those that have *calculated in code* as initial value are automatic, that is to say they do not need a preset value. First the boundaries are defined, then a condition telling the inside from the outside initiate the $insideArea$ boolean. This is done by the matrix inequality,

$$\boldsymbol{if} \quad \boldsymbol{A} \cdot r_{UAV} \leq \boldsymbol{b} \quad \Rightarrow insideArea = \boldsymbol{true}. \tag{5.1}$$

Where r_{UAV} is the current UAV position, $\boldsymbol{A}$ and $\boldsymbol{b}$ are calculated as

$$\begin{aligned} p_i &= p(x_i, y_i), && (5.2)\\ \Delta p_i &= p_i - p_{i+1}, && (5.3)\\ A_i &= [\Delta p_{y,i}\ \Delta p_{x,i}], && (5.4)\\ b_i &= \Delta p_y \cdot p_i(x) - \Delta p_x \cdot p_i(y). && (5.5) \end{aligned}$$

Where p_i and Δp_i are the points and lines defining the boundaries of the polygon. Next an additional condition is calculated, the scalar product of desired and actual velocity,

$$flyStraight = \frac{V_{UAV}}{|V_{UAV}|} \cdot \frac{V_{Des}}{|V_{Des}|}. \quad (5.6)$$

This expression shows how much the UAV has turned. The variable fs is then compared to n, which can be any number from -1 to 1, the purpose is to smooth the exit of the turn. The value used here is 0.8 which makes the UAV exit turn mode when it have turned about 140°. In order to find the most remote point and compare it with the current position, the following scheme is applied,

$$\begin{aligned} BP &= \max_i [\boldsymbol{V}\boldsymbol{p}^T(p_i - S)] && (5.7)\\ \boldsymbol{V}\boldsymbol{p}^T((x,y) - S) &\geq BP \Rightarrow \textit{break}. && (5.8) \end{aligned}$$

$Parameter$	Type	Designation	Initial Value
$insideArea$	boolean	Inside Area	calculated in code
$allowTurn$	boolean	Allow Turn	1
$flyStraight$	reference	Fly Straight	calculated in code
V_{Des}	vector	Desired Direction of Flight	V_{UAV}
z	vector	Direction of Turn, left/right	[0 0 -1]
Vp	vector	Raster Progression Vector	calculated in code
S	coordinate	Raster Origin	-200,200 (here)
r_{UAV}	coordinate	UAV Current Position	calculated in code
r_{sup}	coordinate	Shadow Point Position	calculated in code

Table 5.1: Algorithm Governing Parameters

Pseudo Code

Now for a description of the algorithm itself. The states, Position Velocity and Acceleration are calculated every 0.2 seconds, this is done in one of the functions Turn or Straight. Turn is an open loop controller, it returns a desired bank angle to the autopilot, the straight function on the other hand is a closed loop, it is built to minimize course deviation and works with smaller bank angles. All ***if*** statements are visited in each time step and executed according to the conditions. The CALCULATE and UPDATE commands are executed in every time step.

INITIATE $Position$
INITIATE $Velocity$
INITIATE $Acceleration$
INITIATE $allowTurn = 0$
INITIATE $z = [0\ 0\ -1]$
INITIATE $V\_Des\ =\ V\_UAV$

while $Vp^T((x, y) - S) < BP$ **do**

if $\boldsymbol{A} \cdot r_{UAV} \leq \boldsymbol{b}$ AND $allowTurn = 0$ **then**
SET $V\_Des = -V\_Des$
SET $allowTurn = 1$
SET $z = -z$
end if

if $allowTurn = 1$ AND $\frac{V_{UAV}}{|V_{UAV}|} \cdot \frac{V_{Des}}{|V_{Des}|} \leq n$ **then**

Turn

else if $allowTurn = 0$ OR $\frac{V_{UAV}}{|V_{UAV}|} \cdot \frac{V_{Des}}{|V_{Des}|} \geq n$ **then**

Straight

end if

UPDATE Position Speed Acceleration

if $\boldsymbol{A} \cdot r_{UAV} \leq \boldsymbol{b}$ AND $fl\frac{V_{UAV}}{|V_{UAV}|} \cdot \frac{V_{Des}}{|V_{Des}|} \geq n$ **then**
$allowTurn = 0$
end if

end while

Properties and Performance

In order to employ this routine effectively one has to realize when it is not searching, in the turn part, outside the area, and in transitions between areas. What one wants to do is fly as long sweeps as possible and minimize the transitions between subareas. This means that there are cases when it would be fruitful to cover larger areas than asked for, if that maximize the search time divided by total time of all k possible pattern combinations.

$$\max_k \left(\frac{T_{search}}{T_{total}} \right) \tag{5.9}$$

The parameter one have for one polygon is the sweep direction, for example for a football field, flying lengthwise or across, the number of turns would differ by a factor 2. For this example the turning radius is 13.9 m, but

since the exit from the turn is softened, the sensor radius have to be half the sweep width 16.9 m which is still less than the specified two turn radiuses, 27.8 m.

5.2 Tracking Algorithms

The ability to track a target/object once located is of great interest in the UAV field, for surveillance as well as for strike or rescue missions. Once the target is identified the autopilot needs to employ some algorithm to track rather than to search. The performance, type of and positioning of the onboard sensors determine how the UAV has to maneuver in order to keep a target in focus. A gimbaled camera have fewer requirements on UAV orientation than a fixed camera. Other parameters are the sensor range and maximum target distance.

When it comes to tracking a known target there are methods very straight forward, e.g. applying a sinusoidal path to correct for speed difference [8] and a rose curve or circle if the target stops. Another method is to aim the tracking UAV on a trajectory tangent to some proximity circle of the target [9], this method apply the same algorithm independent of target speed, i.e even immobile. More advanced applications take in threats and no-fly areas to the planning of the path and employ receding horizon like algorithms [10]. Since none of these suited the purposes of this project, a new algorithm was devised. A potential field of parabolic shape centered on the target. This, also simple, but effective algorithm keeps the UAV in the proximity of the target at all times. Moreover, nothing more than the position of the target is needed to track it.

5.2.1 Tracking Problem Statement

The tracking problem is defined by:

> ”To track a moving target, i.e. to keep the UAV in proximity of the target. Furthermore the algorithm must be able to handle a target speed range from zero to the max airspeed of the UAV.”

The proximity is considered to be approximately 50 m. Furthermore it is assumed that by being in the proximity of the target, the target position is at all times known.

5.2.2 Potential Field Method

This method is based on a potential field, shaped as a parabolic, centered on the target. The control error grows with the square of the distance to the target, times a proportionality constant. In order to improve performance of the algorithm a similar penalty on velocity difference is incorporated. The more the velocity differs, the more control input in the direction of the discrepancy. This is then used to calculate a turning acceleration, $\boldsymbol{a}_{turn}$. A safety measure is the ”amplified instability” i.e. the equilibrium point when the target is straight behind the aircraft. This prevents the UAV from getting to far

ahead of the target. The amplified instability conditions the UAV to perform a maximum acceleration turn if the target bearing angle is too great, here that is set to $175°$. This part is then inserted in the turning acceleration algorithm. Simulation showed that the performance can be enhanced if the air speed of the tracking UAV is only slightly greater than that of the target. In the simulations it became apparent that a too great speed excess would force the UAV to make a circle, during that time the target got a big lead. Thus a condition on speed difference is used. What happens is that if the target turns, the UAV can use the speed surplus to quickly intersect. One needs to calculate the two accelerations, parallel and perpendicular to $\boldsymbol{V}$ separately. If not, the aircraft attempts to loop when passing straight over the target. The target bearing $\boldsymbol{s}$ and estimate of target speed $\boldsymbol{V}_T$ are given by,

$$\boldsymbol{s} = (x,y)_{T,n} - (x,y)_{UAV,n}, \tag{5.10}$$

$$\boldsymbol{V}_T = \frac{(x,y)_{T,n} - (x,y)_{T,n-1}}{\Delta t}, \tag{5.11}$$

where (x, y) is the position, subscript T denotes target and subscript UAV the UAV, also note that n denotes present time and Δt is the time resolution. Calculating the control error, with constants k_D and k_V, the force equation yields the desired acceleration,

$$\boldsymbol{F} = \boldsymbol{s}||\boldsymbol{s}||_2 k_D + k_V(\boldsymbol{V}_T - \boldsymbol{v}_{0,UAV}) \tag{5.12}$$

$$\hat{\boldsymbol{a}}_{turn} = \boldsymbol{F}\frac{1}{m}. \tag{5.13}$$

where $\boldsymbol{v}_{0,UAV}$ is the initial UAV velocity vector. Taking the cross acceleration part, the perpendicular projection, and normalize it so that the acceleration do not exceed the maximum turn rate.

$$\tilde{\boldsymbol{a}}_{turn} = \frac{\boldsymbol{v}_{0,UAV} \times (\boldsymbol{v}_{0,UAV} \times \hat{\boldsymbol{a}}_{turn})}{||\boldsymbol{v}_{0,uav}||_2^2} \tag{5.14}$$

$$\boldsymbol{a}_{turn} = \tilde{\boldsymbol{a}}_{turn}\frac{\min(a_{max,turn}, ||\tilde{\boldsymbol{a}}_{turn}||_2)}{||\tilde{\boldsymbol{a}}_{turn}||_2} \tag{5.15}$$

The condition of amplified instability is simply calculated as the dot product of target bearing $\boldsymbol{s}$ and $\boldsymbol{v}_{0,UAV}$, which instigates a turn if the target is located in a cone of $5°$ behind the UAV.

When it comes to speed adaptation, the UAV should only be a few percent faster than the target in order to be able to intersect, the desired speed surplus denoted V_+. It might also be desirable if the UAV is at cruise speed if the target stops. Using the estimate of the target speed to calculate the difference, ΔV, from the airspeed of the UAV. One more effect of the speed surplus is the amount of twisting, greater surplus gives more twisting, small surplus less so. This might be a consideration in military applications. Another consideration might be to keep the throttle setting fixed while solely compensating speed difference with twisting, this could be fuel conserving.

$$\Delta V = ||V_{T,n}|| - ||\boldsymbol{v}_{0,UAV}|| + V_+ \tag{5.16}$$

$$\boldsymbol{a}_{long} = \frac{\boldsymbol{v}_{0,UAV}}{||\boldsymbol{v}_{0,UAV}||}a_{max,long}\Delta V \tag{5.17}$$

Thus the output acceleration vector can be assembled,

$$\boldsymbol{a}_{total} = \boldsymbol{a}_{long} + \boldsymbol{a}_{turn} \tag{5.18}$$

one now have a suitable input for the autopilot, where the input to rudder/ailerons and throttle are easily distinguishable and one can easily transform turning acceleration to a desired bank angle.

The ability to accurately track a target depends on the speed ratio, and on maximum acceleration. The above algorithm can accurately track targets from zero to about the UAVs own speed. See Figure 5.2 for a simulation, here UAV $\boldsymbol{v}_{max}$ is 20 m/s and maximum acceleration, $\boldsymbol{a}_{max}$ is 10 m/s^2. The target speed is 13 m/s and the desired speed surplus is 1 m/s. After 200 seconds the target stops while the UAV keep going for an additional 100 seconds. Estimated minimum turn radius is 20 m and maximum distance to target is 20 m, the shape of the orbit around the stopping point of the target is arbitrary depending on speed, position and direction when the target stops. The target is described in Appendix A.

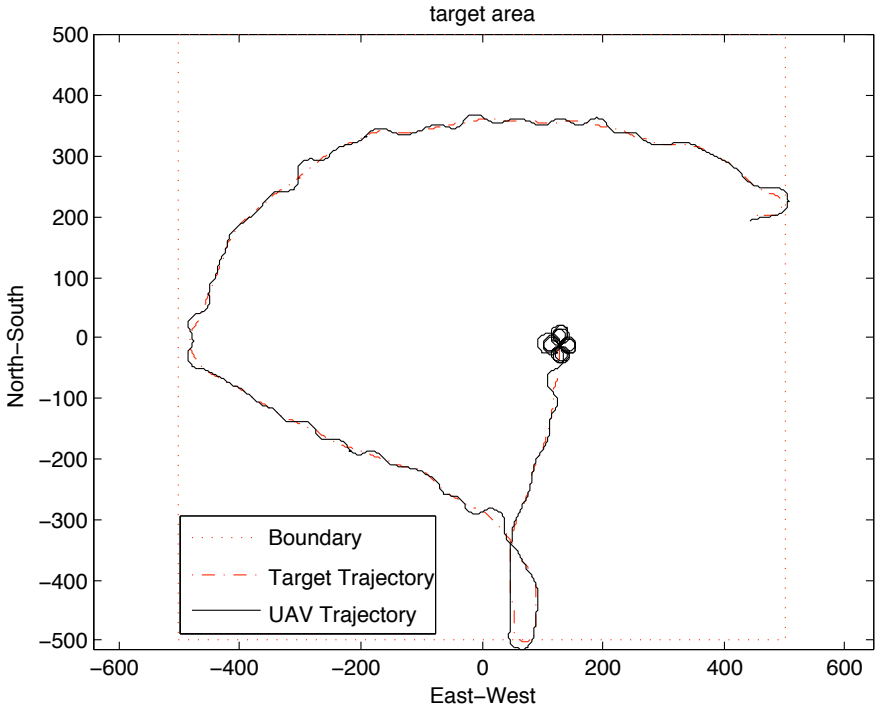

Figure 5.2: Potential Field Method Tracking the Target

6 Implementation

In this section the implementation aspects are covered with a focus on the software part and not so much on the physical construction of the plane and the autopilot. The focus is on setting the system parameters of the control loops of the MP2128^g, and also some fundamental parameters of the UAV, such as take off speed and servo action limits.

6.1 Converting the Gain

In order to take the gains from the simulations in Ptolemey II and implement them into MP2128^g, one have to convert them. This is because MP2128^g uses a system gain, which is built up by integers, i.e. decimal numbers have to be recalculated. This can be done by following the conversion equation,

$$K_{MP} = \frac{K_{PT}}{d} \cdot \frac{Output\ unit}{Input\ unit} \cdot \left(\frac{1}{f}\right). \tag{6.1}$$

Where K_{MP} is the system gain used by MP2128^g, K_{PT} is the corresponding gain that has been calculated in section 4. The gain is multiplied by the divisor d, the divisors can be found in Table 6.1. Last is a unit normalization, units can also be found in Table 6.1. The divisor is a number that is programmed in the MP2128^g. The I-part is dependent on the sampling frequency (f), and have to be divided by the update frequency, see Table 6.1. See Table 6.2 for the converted gains of the two different models.

6.2 Configuring the UAV

The input to the stabilizing part of the autopilot is given in a vrs file. It is a file that is created in the software Horizon^{MP} and contains such data as the control parameters from above, but also airframe data such as minimum airspeed and control surface configuration. Two sets were created, the only difference being the control parameters. The input to the navigation part is aptly called a fly file. This file contains all data needed for the flight, such as take off sequence, flight path, altitude and error traps. Several of these files were written in order to test the different inputs from section 4, but also simple figures, e.g squares, to test basic airworthiness and performance.

Channel	Multiplier	d	Input Units	Output Units
Throttle from	k_P	32	m/s · 0.3048	Fine servo
Airspeed	k_I	32	m/s · 0.3048	Fine servo
(5 Hz)	k_D	32	-	Fine servo
Pitch from	k_P	1024	m · 26.2467	Radians ·1024
Altitude	k_I	32768	m · 26.2467	Radians·1024
(5 Hz)	k_D	1024	m/s · 0.3048	Radians·1024
Elevator from	k_P	512	Radians · 1024	Fine servo
Pitch	k_I	16348	Radians · 1024	Fine servo
(30 Hz)	k_D	2048	Radians · 21504	Fine servo
Aileron from	k_P	4096	Radians · 1024	Fine servo
Roll	k_I	32768	Radians · 1024	Fine servo
(30 Hz)	k_D	256	Radians · 21504	Fine servo

Table 6.1: Control System Units

	Gain	V1	V1
Throttle from Airspeed Error			
Proportional	k_p	412813	137605
Integral	k_i	3441	918
Derivative	k_d	-	-
Pitch from Altitude Error			
Proportional	k_p	-3196	-799
Integral	k_i	0	-63921
Derivative	k_d	0	0
Elevator from Pitch Angle Error			
Proportional	k_p	11702	5857
Integral	k_i	23	23
Derivative	k_d	1193	113
Aileron from Bank Angle Error			
Proportional	k_p	-185237	-190244
Integral	k_i	-445	-445
Derivative	k_d	-838	-418

Table 6.2: Control Parameters System Units

6.3 Pre Flight Test

To make sure that the MP2128^g, the radio modem and the RC receiver did not interfere with each other, a test of radio control reach was performed. The method was to carry the UAV away and see how far one got before signal was lost. The results were acceptable with fair reception at one hundred meters. The GPS was tested at the roof of the FOI building, see Figure 6.1. The original antenna, the copper plate with the black bulge on it in Figure 6.1, failed to deal with the cold start, but by attaching another antenna the cold start was overcome. Once that was done the onboard antenna was sufficient to maintain a GPS lock. During these trials an independent GPS receiver

detected eight satellites with good reception. When that GPS receiver was connected to SmartOne's antenna only two satellites were detected and with poor signal strength. The problem was relieved by reposition the antenna to the wing. This solution gave a sufficient number of satellites, almost as many as was detectable with the hand held set.

Figure 6.1: The GPS trials at the FOI building in Kista

7 Conclusions

The first main goal in this thesis was to accurately and thoroughly model the SmatOne UAV and the MP2128^g autopilot. This has been done and the simulations show that the operation of the model is ok, i.e. all responses are the expected. This means that the flight mechanics and the controls part of the model is correctly structured. For the aerodynamic data set the conclusion is not as easy to draw, as it is never easy to verify aerodynamics. It is worthwhile to point out that neither Computational Fluid Dynamics, CFD, using Navier-Stokes equations nor wind tunnel testing give an unambiguous and correct answer and it is very hard and expensive to measure on a live plane. However, it is the consensus in the project group that all data is fairly accurate, the most doubt concerns friction drag, $C_{D,0}$ and the damping factors. As for tuning autopilot parameters and transfer to MP2128^g that have not been tested, however, comparing the parameters from section 4 with the parameters used by the people at SLU (who have done test flights with the same setup), all parameters match in magnitude.

Simple and effective Path Planning was one of the main goals with this work. The Search Algorithm proposed in section 5 fulfills the demands stated and furthermore the demand on sensor range has been diminished, from 2 times the turning radius to half the sweep width. Here that corresponds to 16.75 m instead of 28 m. The goal of searching an arbitrarily shaped area is also met, since any area can be divided into polygons. There were no demands on optimality in mission time or flight path length, nevertheless the algorithm is very effective, no part of the area is covered twice, this is due to the construction of the pattern. One can probably always find a raster that is more effective for a given polygon, as one can probably also always find a division of an area into better subareas. It is one conclusion that Ablavsky and his team found that ellipses are a better shape because there are only two parameters that govern the shape and thereby the search pattern. An arbitrary polygon have arbitrarily many such parameters. Other benefits with the algorithm here are that it is simple to define an area, it is simple to combine sub areas, the demand on input for the UAV to be able to carry out the search is GPS position updates at some rather low frequency, 5 hz was simulated here because that is what MP2128^g offers, it is likely that one can lower that even further, but it is unclear if such slow updates are necessary.

The other algorithm was the tracking algorithm, given that the proximity sensor do not lose the lock on the target, the algorithm will always work. Of course the UAV is dependent on some small speed surplus, simulations were done with speed surplus from 0 to 100%, and it was found that a too big

surplus would on occasion force the UAV to turn 360° and whilst doing so the range to the target could exceed the proposed 50 m. On the other hand, a small surplus may put the UAV close to and at an almost fixed position relative to the target, which in a military application might not be desirable. The input needed by the UAV here is the target's and the UAV's own position at present time and last time instance. When combining flying to the target and flying parallel to it one get a turn to the same direction as the target, i.e. one will not pass over it in the wrong direction. This ensures good tracking for the speed range basically using twisting maneuvers to keep close.

On the point of having an easy to operate interface, that did not happen since the software from Micropilot Corporation did not work as promised.

For the ultimate goal of flying the UAV, that failed on a technicality. The UAV is ready for take off but the necessary piloting permit is not in place.

8 Future Work

Due to a number of unforeseen events, e.g. lack of software for the plug-ins, lack of flight permission for the tests and not to mention a number of breakdowns and other minor setbacks, the time ran out and no tests could be performed nor any plug-ins written. The project deadline came before the flight permission, which put the last nail in the coffin. However it is the firm belief of the author and the rest of the project staff that, given more time, the goals could have been fulfilled.

8.1 Micropilot Plug-ins

To demonstrate the abilities of the UAV, a maneuver was planned to be implemented with the aid of a plug-in. The plug-in was to be written in C++ and then loaded into the MP2128^g via the software development kit $XTENDER^{mp}$. It was decided to constrain the programming to a simple maneuver, namely a 180° turn at a given position. This should serve as a demonstrator of how many different applications could be constructed and implemented. The plug-ins work according to this rather schematic description. At start up, MP2128^g checks for plug-ins and the event parameters of the plug-in. The event parameter works like a boolean variable, true or false, it could for example be a geographic condition, as was the case here. The MP2128^g calls on the plug-in at some frequency to be determined, in sync with MP2128^g's own frequencies, and the plug-in returns a number, 1,2, or 3. Where 1 means discard result, do not call again, 2 means use result if applicable and 3 means discard result but call again. Moreover, the plug-in returns some output of the code it is built from, for example an elevator angle. To summarize: if the MP2128^g receives a 1 and an elevator angle of, for example 10°, the MP2128^g discards the elevator angle and don't call the plug-in again. If MP2128^g should receive a 2 and elevator angle of 10° it would override its own navigation solution and set the elevator to 10°. In the third case i.e. MP2128^g receives a 3 and some parameter, it would simply implement its own solution, and then call the plug-in again.

8.2 Flight Test

The purpose of Flight Test is to verify the work of section 3 and section 4. The flight using plug-ins also serves as a proof of concept, that cheap of the shelf products can be made into versatile UAVs. Here the first fly file is to be flown, it is a square of 100 m. The results should show if the UAV can handle

itself and fly 5 laps around the square at, near, constant altitude and speed, the turn performance will also be demonstrated. Both autopilot settings will be tried. After the initial test, step and impulse is to be tested, in the three channels; airspeed, altitude and bank angle. The inputs are the same as in section 4. Here one will be able to isolate performance per channel, and also compare the responses to those from the simulations. To check for the lateral and longitudinal modes the 3-2-1-1 inputs are to excite the UAV, then one can see how well and how fast it returns to level, unaccelerated flight, the critical number here is ”time to half”, that is the time it takes for the amplitude to half. The last item on the planned test session is the MP-plugin.

Bibliography

[1] MicroPilot Corporation. Micropilot Documentation and Manuals (available upon purchase). *Micropilot, 2005 .*

[2] B. Etkin & L.D. Reid. Dynamics of Flight, Stability and Control. *John Wiley & Sons, 1996.*

[3] R.C. Nelson. Flight Stability and Automatic Control. *WCB/McGraw-Hill, 1998.*

[4] B.L. Stevens & F.L. Lewis. Aircraft Control and Simulation. *John Wiley & Sons, 1992.*

[5] J.D. Anderson. Aircraft performance and Design. *WCB/McGraw-Hill, 1999.*

[6] T. Glad & L. Ljung. Reglerteknik. *Studentlitteratur 1989.*

[7] G. Platanitis & S. Shakarayev. Integration of an Autopilot for a Micro Air Vehcle. *AIAA 2005.*

[8] J. Lee, R. Huang, A. Vaughn, X. Xiao, JK. Hedrick, M. Zennaro & R. Sengupta Strategies of Path-Planning for a UAV to Track a Ground Vehicle. *AINS Conference 2003.*

[9] F. Rafi, S. Khan, K. Shafiq & M. Shah. Autonomous Target Following by Unmanned Aerial Vehicles. *Defence and Security Symposium, Orlando Florida. SPIE 2006.*

[10] U. Zengin & A. Dogan. Dynamic Target Pursuit by UAVs in Probabilistic Threat Exposure Map. *AIAA 3rd Unmanned Unlimited Technical Conference, Chicago 20-23 september 2004.*

[11] V. Ablavsky, S. Holden & M. Snorrason. Efficient Pursuit of a Moving Target via Spatial Constraints Exploitation. *In AIAA Guidance, Navigation and Control Conference and Exhibit, Montreal Canada AIAA 6-9 August 2001.*

[12] V. Ablavsky, & M. Snorrason. Optimal Search for a Moving Target: A Geometric Approach. *In AIAA Guidance, Navigation and Control Conference and Exhibit, Denver Colorado AIAA 14-17 August 2000.*

[13] V. Ablavsky, D. Stauch & M. Snorrason. Search Path Opitmization for UAVs using Stochastic Sampling with Abstract Pattern Descriptors. *In AIAA Guidance, Navigation and Control Conference and Exhibit, Austin Texas AIAA 11-14 August 2003.*

[14] L.F. Bertuccelli. & J.P. How. Robust UAV Search for Environments with Imprecise Probabillity Maps. *In IEEE Conference on Decision and Control 2005 and 2005 European Control Conference CDC-ECC'05 Seville Spain p5680-5685.*

[15] Eugene A. Morelli. Advances in Experiment Design for High Performance Aircraft. *NATO Symposium on System Identification for Integrated Aircraft development and Flight Testing Madrid Spain 5-7 May 1998.*

[16] P. Ögren. A. Backlund T. Harryson, L. Kristensson & P. Stensson. Autonomous UCAV Strike Missions using Behavior Control Lyapunov Functions. *In AIAA Guidance, Navigation and Control Conference and Exhibit, Keystone Colorado AIAA 21-24 August 2006.*

[17] Michael. S. Francis J-UCAS Update. *Military Technology issue 12 p 16-22.*

[18] Wikipedia.org Available Unmanned Aerial Vehicle. *http://en.wikipedia.org/wiki/UAV available 061214.*

[19] Proffesor E. Lee et al Ptolemey II 5.0. *http://ptolemey.eecs.berkeley.edu/index.html available 061214.*

[20] MatLab. Mathworks Inc *www.mathworks.com available 061214.*

[21] Mark Drela & Harold Youngren. AVL *http://web.mit.edu/drela/Public/web/avl/ GNU General Public License available 061214 .*

[22] Hans Backström. Generic Aerodata Model *SAAB Military Aircraft 1996 www.foi.se/admire available 061214.*

A Description of the Target

The target used herein is a mobile target with a stochastic pattern of movement, its speed ranges from zero to 75% of the UAV max speed. There is also a constraint on maximum acceleration. The movement of the target is described by,

$$\begin{aligned} \boldsymbol{a_0} &= F/m(\boldsymbol{v_0} \times [0\ 0\ 1]^T) && \text{(A.1)} \\ \boldsymbol{a_B} &= 1/m((\boldsymbol{v_0} \times \boldsymbol{F_L}) \times \boldsymbol{v_0}) && \text{(A.2)} \\ \boldsymbol{a} &= \boldsymbol{a_0} + \boldsymbol{a_B} && \text{(A.3)} \\ \boldsymbol{v} &= \boldsymbol{v_0} + \boldsymbol{a}\Delta t && \text{(A.4)} \\ \boldsymbol{x_{TP,i}} &= \boldsymbol{x_{TP,i-1}} + \boldsymbol{v}\Delta t && \text{(A.5)} \end{aligned}$$

where F is a random normal distributed force input and the initial speed $\boldsymbol{v}_0$ likewise is vector of random magnitude and direction. The step size Δt, is the time resolution for the target dynamics and i is present time. The construction of (A.1) gives the target random pushes from its sides, forcing a turn. To confine the target in a specified area a restoring force F_L, is applied normal to and at line 25 m inside the boundaries.

FOI is an assignment-based authority under the Ministry of Defence. The core activities are research, method and technology development, as well as studies for the use of defence and security. The organization employs around 1350 people of whom around 950 are researchers. This makes FOI the largest research institute in Sweden. FOI provides its customers with leading expertise in a large number of fields such as security-policy studies and analyses in defence and security, assessment of different types of threats, systems for control and management of crises, protection against and management of hazardous substances, IT-security an the potential of new sensors.

FOI
Swedish Defence Research Agency
SE-164 90 STOCKHOLM

Tel: +46 8 5550 3000
Fax: +46 8 5550 3100

www.foi.se