Design of a model predictive controller to control UAVs

A.J.M. Raemaekers

DCT 2007.141

Traineeship report

Coach(es):	dr. R. Hill (RMIT) prof. dr. L. Wang (RMIT)
Supervisor:	ass. prof. dr. C. Bil (RMIT) prof. dr. H. Nijmeijer (TU/e)

Technische Universiteit Eindhoven
Department Mechanical Engineering
Dynamics and Control Group

Royal Melbourne Institute of Technology
Aerospace Design Technology
Sir Lawrence Wackett Centre

Eindhoven, December, 2007

Abstract

In this report a continuous time model predictive controller is applied to the control of an Unmanned Aerial Vehicle (UAV). The purpose of this work, is to see if it is possible to design a good tracking controller that is capable of simulating a highly nonlinear aircraft system.

For the simulations a MATLAB Simulink model of the UAV 'Ariel' is used. The nonlinear Simulink model covers all the aircraft dynamics. The model consists of five inputs, twelve states and twelve outputs. This model is linearized at several different operating conditions, for use with the model predictive controller.

The model predictive controller is capable of simulating in real time, making it possible to use this in on-line applications. The Ariel model is linearized to a linear time invariant state space system. The state space model is then extended. This means the outputs of the original model are included in the state vector of the extended model. With this extended state space model, it can be shown that the selected outputs can be tracked by the controller. The controller uses an algorithm which reduces the required computational power compared to classic model predictive control.

The possibility to apply constraints on the controller input, the derivative of the controller input, the states and the outputs of the system, make this model predictive controller a nonlinear system. Because in real life constraints are almost always present on any dynamical system, the presented model predictive controller has a broad application scope.

The simulations show that the controller performs well, after tuning of the parameters. Applying the constraints puts some limits on the possible reference trajectories, but shows that the system is very capable of handling these controls. Also, simulation of failure of one of the controls is possible, simply by setting the maximum and minimum values to zero. The UAV controller is able to use the other three controls to compensate for the lack of the fourth, keeping the aircraft at steady flight.

The application of model predictive control to aircraft control is relatively new. This report shows that using a model predictive controller works well for the simulations done in this report. This method of MPC can be used not only on aircraft systems, but in general on all types of dynamic systems, with the possibility of applying constraints.

For future work, a switching controller can be designed to use the different linearized models at different operating conditions. Wrapping another controller around the model predictive controller presented here, should make it possible to let the aircraft follow a set of way-points, thus not requiring a pre-defined reference trajectory to be able to fly fully autonomously.

Contents

Chapter 1

Introduction

In recent years, model predictive control (MPC) has been an upcoming control technique in on-line applications due to the ever increasing computer power of modern computers. MPC is mostly used to control large time scale factory processes, but in recent years the application has moved to dynamic processes with smaller time scales.

Most existing MPC implementations use a discrete-time form; the continuous-time counterpart was developed much later. Continuous time modeling gives the advantage that future predictions can be calculated in an analytical form. This reduces the required computational effort because there are less parameters that need to be optimized.

The response time of the control signal in the continuous time MPC case is a bit slower compared to the discrete time case [1]. This means the performance is slightly worse, because it takes more time before the system is stabilized. However, continuous time MPC requires less optimization steps to calculate the control signal. It also makes the tuning of the controller more intuitive, and the calculated control signal smoother. One of the latest developments in continuous time MPC is the use of orthonormal functions to describe the control signal [1]. In this report the application of this method to the control of a Unmanned Aerial Vehicle (UAV) will be discussed.

In recent years, unmanned aircraft have become more and more popular. The military has great interest in these machines, for amongst other things, battlefield surveillance, reconnaissance and targeting. They can also be used for civilian applications such as reconnaissance, crop dusting, observations, police surveillance, TV broadcasting, photography etc. The fact that they are unmanned make UAVs particularly suitable for going into inhospitable or hazardous environments. In general, UAVs have the potential to complement traditional manned surveillance and monitoring operations or, for some missions, can be a cost effective alternative.

In the beginning of the last century the research was mainly focused on auto pilot systems, but this knowledge was restricted for the public for military security. Controllers designed before the development of classical control theory, were simple altitude and heading holds. But with the up-come of classical control theory and the introduction of the jet engine more and more controllers were applied to the aircraft to assist the pilot in flying the aircraft. Therefore, control of aircraft has been studied for a long time, ever since the development of modern computers in the 1950's.

To control UAVs, different methods have been studied over the years, with different results. In this report a continuous time model predictive controller with orthonormal functions will be designed to control a UAV. The UAV 'Ariel' used in this report was modeled by Crump [2] with the MATLAB Simulink software.

This Simulink model is linearized at different operating points. The linearized model is used in the model predictive controller. The controller calculates the system controls, needed to track the selected outputs of the Ariel aircraft model. The model predictive controller makes it possible to add hard constraints to the states, controls and outputs of the linear state space model. It is also possible to simulate failure of one or more of the controls by setting the hard constraints to zero.

MPC has been applied to UAV control before [3], but this method differs from the previous work done, because, by using orthonormal functions to approximate the control signal and calculate the UAV response, the required computation effort decreases, making it possible to use this version of MPC to control the Ariel model in an on-line environment.

This report is organized as follows; in the second chapter the background of UAV and flight control is described. The third chapter gives the reader some insight in flight dynamics and the equations of motion describing the states of an aircraft. It also describes the linearization of the (non linear) equations of motion and how the steady state operating conditions are determined. The fourth chapter will familiarize the reader with model predictive control, and deals with the use of orthonormal functions to describe the control trajectory in more detail. The fifth chapter contains the simulations done to show how the implementation of the model predictive controller is able to control the Ariel UAV and also how it is able to deal with constraints on and failure of one or more of the control inputs. Finally in chapter six the conclusions and recommendations are dealt with.

Chapter 2

UAV and flight background

2.1 UAV development

In the last few decades the use of Unmanned Aerial Vehicles (UAVs), has become more and more popular. Due to the more advanced technology available nowadays, the emphasis has moved from Remotely Piloted Vehicles (RPVs) to UAVs, which have a form of autonomy. The autonomous capabilities of UAVs have improved over the last years due to improvements in (control) technology and improvements of computational power.

A brief review of the last 20 years of UAV activity is given in [4]. Another review of UAV activity can be found in [5], where several specific aircraft platforms are discussed in reasonable depth. All reports indicate the increasing prominence which UAVs are taking in present day armed forces. These documents only discuss the developments in Western countries, whilst development within the Middle East, the Soviet bloc and Asia is also well underway.

The majority of UAV development has been military based. Most well known are The Predator (a medium sized tactical UAV), developed by General Atomics and produced in 1997 and the Northrop Grumman Global Hawk, which has been developed as a high altitude, large area surveillance aircraft. But there have been several civilian forays into the field as well. The two most successful civilian UAV are the Aerosonde aircraft, used for meteorological surveying and the Yamaha Aero-Robot, an unmanned helicopter used for crop-dusting.

A study of the market in Australia for both civilian and military uses is given in [6, 7]. They conclude that Australia has both a very large potential market for UAV applications and also a mature capability for advanced aircraft design, which could be applied very successfully to a 'small' design project such as a UAV. The development of a multi-purpose Australian UAV is discussed in [8], addressing some of the issues that must be considered in design for the Australian environment.

2.2 Aircraft control history

The introduction of the jet aircraft resulted in more complicated aircraft dynamics. Underdamped high frequency modes which became increasingly difficult for human pilots to control became evident. The so called short period longitudinal mode and the dutch roll lateral mode fall in this category. Thus, in these jet aircraft, artificial damping had to be provided to allow satisfactory handling qualities. These artificial damping systems were developed using the emerging classical design techniques which are mainly based on working with single input single output (SISO) systems and using tools such as Bode plots, Nyquist plots and Evans Root Locus techniques. These techniques all use the frequency domain as a design region and are widely accepted, utilizing phase margins and gain margins to illustrate robustness. These classical techniques proved extremely successful for the aircraft of the time; however, they have some well known limitations. The main design techniques are all based upon linear time invariant (LTI) SISO systems and with increasingly complicated aircraft and control objectives, these techniques became increasingly difficult to apply. In 1960, Kalman published three major papers that were the beginning of the modern control era.

These papers contained detailed development of the optimal control of systems, through the use of a Linear Quadratic Regulator (LQR) and the development of an optimal filtering technique, commonly known today as the Kalman filter. The main point of Kalman's work is that it is a time domain approach that uses linear algebra, so systems with multiple inputs and multiple outputs (MIMO) can be handled with no extra effort. He also introduced the concept of the internal state, implying that one is concerned with the internal system behavior as well as the system input/output behavior. Kalman formalized the concept of 'control optimality' by minimizing a quadratic energy function, which provides a unique solution with guaranteed performance. However, only because of the development of digital computer technology, it was possible to solve his matrix equations, which are very time intensive and difficult to solve by hand.

One method that has found a large amount of acceptance in the control industry is an extension of both linear quadratic regulators and Kalman filters to a technique known as Linear Quadratic Gaussian with Loop Transfer Recovery (LQG/LTR). This method utilizes both the linear quadratic regulator (LQR) which requires full state feedback and the Kalman filter to provide a state estimate based upon system outputs. This method does not have the guaranteed properties of each individual system on its own, so the enhancement, Loop Transfer Recovery (LTR), attempts to automatically synthesize an appropriate Kalman filter given a regulator or vice-versa. This technique will generally almost fully recover the desired properties of the full state feedback regulator. Examples of the use of this technique are given in [9].

These new control techniques have two problems compared to classical techniques. The first problem is the lack of engineering intuition in control design, with all problems simply solved through the solution of matrix equations, whilst the second more important problem is that of robustness. The guaranteed performance of modern control designs could quite easily be lost when unmodeled dynamics, disturbances and measurement noise are introduced. Also, the well known phase and gain margin concepts can not be easily applied to the multivariable systems that modern control is so suited for. These problems lead to the introduction of robust modern control theory, and more recently, with computing power still increasing every day, the application of model predictive control (MPC). MPC has been known and used since the 1970's and has been successfully applied to systems with large time constants, e.g. in the control of process industry, because MPC requires a lot of computational power. However in recent years the application of MPC to systems with small time constants has increased and new applications arise every day.

Chapter 3

Flight dynamics

In this report a continuous time model predictive controller is designed with orthonormal functions to control a UAV model. M. Crump has developed an accurate MATLAB Simulink model of the UAV Ariel [2], which is an unmanned aircraft developed by the UAV group in the Department of Aeronautical Engineering at the University of Sydney. A picture of this aircraft is shown in figure 3.1. The majority of data on this aircraft is available in [10].

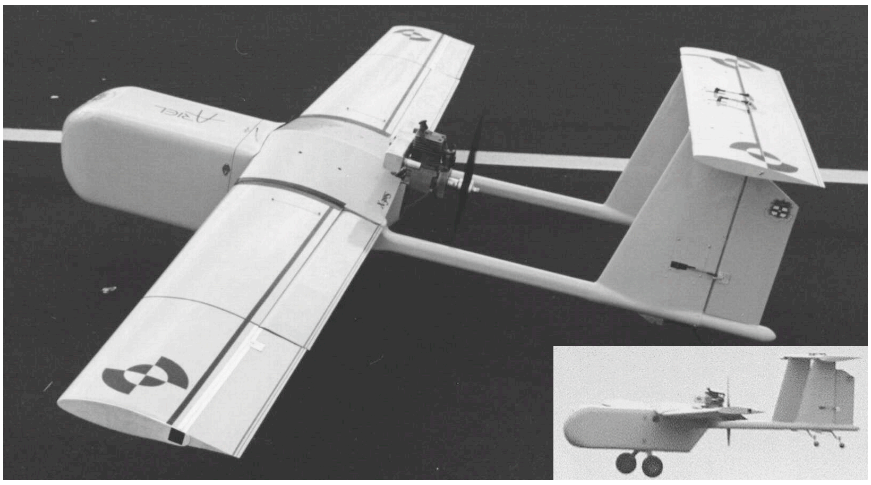


Figure 3.1: Ariel UAV

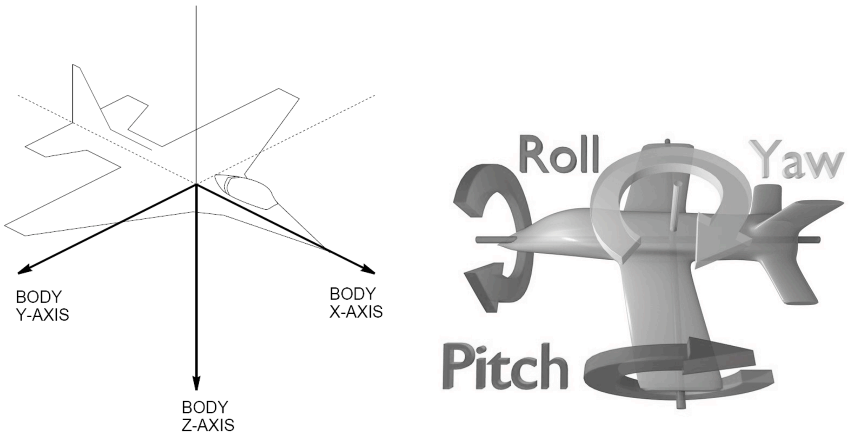
3.1 Coordinate system

The movement of an airplane is described by the equations of motion with respect to the body frame coordinate system. In general there are three coordinate systems that are important when describing the motion of an airplane [11].
1. Body-fixed coordinate system
2. Earth-fixed coordinate system
3. Atmosphere-fixed coordinate system

The body-fixed frame can be expressed in terms of the Earth-fixed frame by the use of Euler angles (ϕ, θ and ψ).

3.1.1 Body-fixed coordinate system

This coordinate system's origin is located at the aircraft center of gravity. The x-axis points forward along some axis of the fuselage in the aircraft's plane of symmetry. The y-axis is normal to the plane of symmetry pointing in the direction of the right wing. The z-axis points downward, completing the right-handed Cartesian system. See also figure 3.2(a). Rotations about the x, y and z-axis are called roll (ϕ), pitch (θ) and yaw (ψ) respectively. See also figure 3.2(b). The Body-fixed coordinate system is also called the ABC frame.



(a) Body CS

(b) Rotations

Figure 3.2: Coordinate system

3.1.2 Earth-fixed coordinate system

The Earth-fixed coordinate system points in the directions North, East and Down (NED frame). The x, y-plane is normal to the local gravitational vector with the x-axis pointing north and the y-axis pointing east. The z-axis points down. For navigation on a small piece of the Earth, the curvature of the Earth can be neglected for simplification. The NED frame is then fixed and does not rotate. This simplification is also called the 'flat Earth' approximation.

3.1.3 Atmosphere-fixed coordinate system

The atmosphere-fixed coordinate system is defined such that all three axes are always parallel to those of the Earth-fixed coordinate system. However, the atmospheric coordinate system moves at a constant velocity relative to the Earth-fixed coordinate system. The atmosphere coordinate system is only used when deriving the equations of motion. It is also used in wind tunnel testing of aircraft models.

3.2 Equations of motion

When deriving the equations of motion, the aircraft is considered as a rigid body. To switch between the ABC and NED coordinate frame, a matrix multiplication is used, consisting of three rotations:
1. Rotation about the z-axis, positive nose right (yaw ψ)
2. Rotation about the new y-axis, positive nose up (pitch θ)

3. Rotation about the new x-axis, positive right wing down (bank ϕ)

The yaw, pitch and bank angles are known as the Euler angles. The transformation matrix B is given as:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_{ABC} = B \begin{bmatrix} x \\ y \\ z \end{bmatrix}_{NED} \tag{3.1}$$

with

$$B = \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ -\cos\phi\sin\psi + \sin\phi\sin\theta\cos\psi & \cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi & \sin\phi\cos\theta \\ \sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi & -\sin\phi\cos\psi + \cos\phi\sin\theta\sin\psi & \cos\phi\cos\theta \end{bmatrix}$$

The inertia properties of the rigid airplane body can be defined by:

$$\begin{array}{llll} c_1 = \frac{(J_y - J_z)J_z - J_{xz}^2}{\tau} & c_2 = \frac{(J_x - J_y + J_z)J_{xz}}{\tau} & c_3 = \frac{J_z}{\tau} & c_4 = \frac{J_{xz}}{\tau} \\ c_5 = \frac{J_z - J_x}{J_y} & c_6 = \frac{J_{xz}}{J_y} & c_7 = \frac{1}{J_y} & c_8 = \frac{J_x(J_x - J_y) + J_{xz}^2}{\tau} \\ c_9 = \frac{J_x}{\tau} & \tau = J_x J_z - J_{xz}^2 & & \end{array} \tag{3.2}$$

And the aerodynamic and propulsive forces and torques will be split into the body axes:

$$F_B = \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} \qquad T_B = \begin{bmatrix} L \\ M \\ N \end{bmatrix} \tag{3.3}$$

With these definitions, the equations of motion of the aircraft can be described by:

$$\begin{aligned} \dot{U} &= RV - QW - g\sin\theta + F_x/m \\ \dot{V} &= -RU + PW + g\sin\phi\cos\theta + F_y/m \\ \dot{W} &= QU - PV + g\cos\phi\cos\theta + F_z/m \\ \dot{\phi} &= P + \tan\theta(Q\sin\theta - R\cos\theta) \\ \dot{\theta} &= Q\cos\theta - R\sin\phi \\ \dot{\psi} &= (Q\sin\phi + R\cos\phi)/\cos\theta \\ \dot{P} &= (c_1 R + c_2 P)Q + c_3 L + c_4 N \\ \dot{Q} &= c_5 PR - c_6(P^2 - R^2) + c_7 M \\ \dot{R} &= (c_8 P - c_2 R)Q + c_4 L + c_9 N \\ \dot{p_N} &= U\cos\theta\cos\psi + V(\sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi) + W(\sin\phi\sin\theta\cos\psi + \sin\phi\sin\psi) \\ \dot{p_E} &= U\cos\theta\sin\psi + V(\sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi) + W(\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi) \\ \dot{p_D} &= -U\sin\theta + V\sin\phi\cos\theta + W\cos\phi\cos\theta \end{aligned} \tag{3.4}$$

With the definition of the state vector as:

$$X = \begin{bmatrix} U & V & W & \phi & \theta & \psi & P & Q & R & p_N & p_E & p_D \end{bmatrix}^T \tag{3.5}$$

with

- U, V and W the velocity with respect to the body axes x, y and z respectively
- ϕ, θ and ψ the three Euler angles
- P, Q and R the angular rates
- p_N, p_E and p_D the positions with respect to the NED frame.

The model is linearized using the so called 'flat Earth' approximation. In this approximation it is assumed that the Earth is flat, and this can be used when navigation is only required over a small area of the Earth.

3.2.1 Aircraft controls

The Ariel UAV can be controlled by four deflective surfaces and the throttle, which are shown in figure 3.3. The controls and their effect on the aircraft are:
1. Throttle: Accelerates/deccelerates the aircraft
2. Elevator: Change the pitch (θ) of the aircraft
3. Rudder: Change the yaw (ψ) of the aircraft.
4. Aileron: Change the banking angle (ϕ) of the aircraft.
5. Flaps: Increase the lift of the wings. Used for take-off and landing.

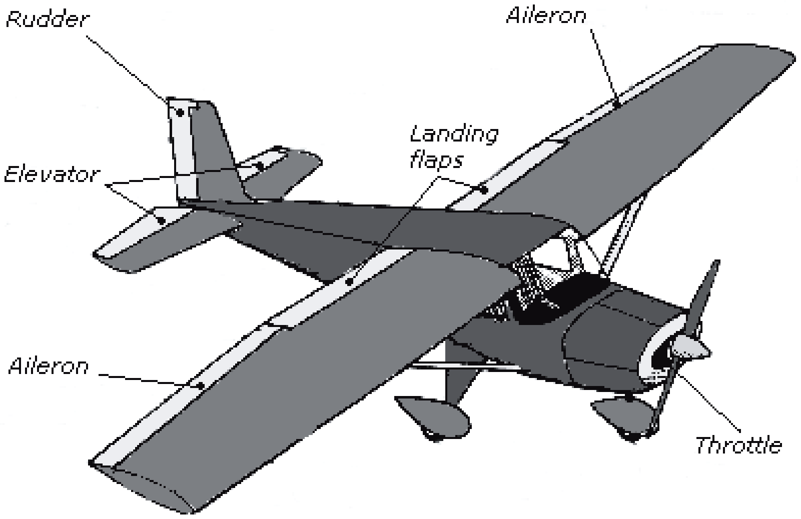


Figure 3.3: Aircraft controls

3.3 Aircraft trim

Aircraft trim is the definition of the aircraft flying at steady state. This means that all the aircraft velocity variables are fixed at a constant value. The aircraft is 'tuned' in a way that it conducts a steady flight and no torque is applied to the airplane's center of gravity. This implies that the aircraft acceleration components are all zero. This definition leads to:

$$\dot{U}, \dot{V}, \dot{W}, \dot{P}, \dot{Q}, \dot{R} \equiv 0, \quad U > 0 \tag{3.6}$$

Steady wings-level flight and steady turning flight is allowed using the 'flat Earth' simplification discussed in section 3.1.2. Ignoring the effects of changing density with altitude, steady wings-level climb and steady climbing turn are also trim conditions. For each of the different trim conditions extra constraints need to be applied. For instance, for a steady wings level flight, the trim condition also requires:

$$\dot{\phi}, \dot{\theta}, \dot{\psi}, P, Q, R, \phi \equiv 0 \tag{3.7}$$

With these constraints an optimum trim condition can be found using an optimization algorithm. In general, a good objective function to optimize is to minimize the sum of the square of the aircraft body accelerations. In the work of Crump [2] an optimization algorithm has been provided and is used to obtain trim conditions for the state, input and output variables for different operating conditions.

The Ariel model is linearized around different trim conditions. Each trim condition is a steady state equilibrium, therefore, the state space system obtained for different operating conditions is stable. In the simulations, the deviations around the steady states of the different state space models have to be added to the values of the trim conditions to provide the 'real' values that are used as input for the non-linear Ariel model.

3.4 Ariel model

The Ariel model is built in the MATLAB Simulink environment. The model uses lookup tables to find values for the aerodynamic and propulsive forces and aircraft parameters for different operating conditions, acquired by windtunnel testing. The 6-degree of freedom dynamic model is complemented by modeling the engine, sensor and actuator dynamics. More details about the Ariel Simulink model can be found in [2]. The model has five inputs; throttle, elevators, ailerons, rudder and flaps, represented as $\begin{bmatrix} \delta_T & \delta_e & \delta_a & \delta_r & \delta_f \end{bmatrix}$ respectively. The values and rates of these inputs are limited. The states of the equations of motion (3.5) are the internal states of the model. The measured outputs are:

$$Y = \begin{bmatrix} n_x & n_y & n_z & V_T & \alpha & \beta & P & Q & R & \phi & \theta & \psi \end{bmatrix}^T \tag{3.8}$$

with

- n_x, n_y and n_z the accelerations in x, y and z direction with respect to the ABC-frame
- V_T the total velocity, $V_T = \sqrt{U^2 + V^2 + W^2}$
- α the angle of attack
- β the sideslip angle
- P, Q and R the angular rates
- ϕ, θ and ψ the Euler angles

3.5 Linearizing the equations of motion

A general nonlinear system can be written in the following form:

$$\dot{x}(t) = f\left(x(t), u(t), t\right) \tag{3.9}$$

$$y(t) = g\left(x(t), u(t), t\right) \tag{3.10}$$

In these equations, $x(t)$ represents the n-states of the model, $u(t)$ represents the m-inputs of the model, and $y(t)$ represents the p-outputs of the model.
A linearized model of this system is valid in a small region around the operating point $t = t_0$, $x(t_0) = x_0$, $u(t_0) = u_0$, and $y(t_0) = g(x_0, u_0, t_0) = y_0$. Operating points can be found using the trim algorithm. Subtracting the operating point values from the states, inputs, and outputs defines a set of variables centered about the operating point:

$$\begin{aligned} \delta x(t) &= x(t) - x_0 \\ \delta y(t) &= y(t) - y_0 \\ \delta u(t) &= u(t) - u_0 \end{aligned} \tag{3.11}$$

The linearized model can be written in terms of these new variables and is usually valid when these variables are small, i.e. when the departure from the operating point is small:

$$\begin{aligned} \delta\dot{x}(t) &= A(t)\delta x(t) + B(t)\delta u(t) + \tfrac{\partial f}{\partial t}(x_0, u_0, t) \\ \delta y(t) &= C(t)\delta x(t) + D(t)\delta u(t) + \tfrac{\partial g}{\partial t}(x_0, u_0, t) \end{aligned} \tag{3.12}$$

The state-space matrices $A(t)$, $B(t)$, $C(t)$, and $D(t)$ of this linearized model represent the Jacobian matrices of the system, which are defined as follows:

$$\begin{aligned} A(t) &= \begin{bmatrix} \nabla_x f_1(x_0, u_0, t) \\ \vdots \\ \nabla_x f_n(x_0, u_0, t) \end{bmatrix} & B(t) &= \begin{bmatrix} \nabla_u f_1(x_0, u_0, t) \\ \vdots \\ \nabla_u f_n(x_0, u_0, t) \end{bmatrix} \\ C(t) &= \begin{bmatrix} \nabla_x g_1(x_0, u_0, t) \\ \vdots \\ \nabla_x g_p(x_0, u_0, t) \end{bmatrix} & D(t) &= \begin{bmatrix} \nabla_u g_1(x_0, u_0, t) \\ \vdots \\ \nabla_u g_p(x_0, u_0, t) \end{bmatrix} \end{aligned} \tag{3.13}$$

where

$$\nabla_x \boldsymbol{f}(x_0, u_0, t) = \begin{bmatrix} \frac{\partial \boldsymbol{f}(x_0,u_0,t)}{\partial x_1(t)} & \frac{\partial \boldsymbol{f}(x_0,u_0,t)}{\partial x_2(t)} & \dots & \frac{\partial \boldsymbol{f}(x_0,u_0,t)}{\partial x_n(t)} \end{bmatrix} \tag{3.14}$$

The obtained matrices $A(t)$, $B(t)$, $C(t)$, and $D(t)$ are all time dependent. The matrix $A(t)$ is specified in appendix A. When the aircraft trim condition for steady wings-level flight (specified in section 3.3) is applied, the system matrix further simplifies to:

$$A = \begin{bmatrix} 0 & 0 & -0.07 & 0 & -9.81 & 0 & 0 & -2.05 & 0.01 & 0 & 0 & 0 \\ 0 & 0 & 0 & 9.81 & 0 & 0 & 2.05 & 0 & -30 & 0 & 0 & 0 \\ 0.07 & 0 & 0 & 0 & 0 & 0 & -0.01 & 30 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.07 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.07c_2 & 0 & 0.07c_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.07c_8 & 0 & -0.07c_2 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & -0.02 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -2.05 & 0 & 30 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0.01 & -30 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.15}$$

The other matrices $B(t)$, $C(t)$ and $D(t)$ will also become time invariant. Because all the accelerations are zero and the external forces don't change over time, $\frac{\partial f}{\partial t}(x_0, u_0, t) \equiv 0$ and $\frac{\partial g}{\partial t}(x_0, u_0, t) \equiv 0$ and these terms can thus be neglected. In the steady wings-level flight case, the state space system becomes a linear time invariant (LTI) state space system.

As mentioned before in section 3.3 there are four different trim conditions. The model only simplifies to a LTI state space system when the trim condition for steady wings-level flight or steady wings-level climb is applied. When the other trim conditions, steady turning flight and steady climbing turn, are used the state space system will remain time dependent.
In this research, only the LTI state space model is used. Using the time varying model in the model predictive controller will require more computations, and therefore makes it not (yet) possible to use this for real time control.

The general equations of motion and the aircraft specific propulsive and aerodynamic force models are combined in the Simulink model of the Ariel UAV together with the sensor and actuator dynamics. To obtain the Linear Time Invariant (LTI) model, a numerical perturbation algorithm is used to linearize the system. This involves introducing small perturbations to the nonlinear model and measuring the responses to these perturbations. Both the perturbations and the responses are used to create the matrices of the linear state-space model. In appendix B an example of a state space model is given linearized around a trim condition.

Chapter 4

Model Predictive Control

In the work of Wang [1] it is shown how to create a continuous time model predictive controller using orthonormal functions. The big advantage of this method is that it reduces the required computing power. First, general model predictive control will be discussed. Secondly, the use of orthonormal functions to design a continuous time model predictive controller will be discussed. In this section a model predictive controller is presented which can be used to control all general types of state space models, as long as they satisfy certain conditions (see 4.2.3). This method reduces the required computing power, thus giving it significant advantages when used in an on-line environment. In chapter 5 the application and results of this design method to control the Ariel model will be described.

4.1 Model Predictive Control

Model predictive control is an approach to controller design that involves on-line optimization calculations. The online optimization problem takes account of system dynamics, constraints and control objectives. Several authors have published reviews of MPC theoretical issues, including the papers of García et al. [12], Ricker [13], Morari and Lee [14], Muske and Rawlings [15] and Rawlings et al. [16]. The main reason for the wide-scale adoption by industry of model predictive control is its ability to handle hard constraints on controls and states that arise in most applications. Conventional model predictive control requires the solution of an open-loop optimal control problem, in which the decision variable is a sequence of control actions, at each sample time. Current control action is set equal to the first term of the optimal control sequence.

4.1.1 Receding horizon concept

The receding horizon idea is the basic concept behind MPC [17]. In most cases, model predictive control uses a discrete time setting. At time t the predicted plant output is calculated using the internal plant model. This is done for the prediction horizon p. Based on the plants predicted output at time $t + p$, a control effort is computed to reduce the system error, thus following the reference trajectory. The first step $(t + 1)$ of the calculated control input is executed. Then the outputs are measured again $(y + 1)$, and based on this output, the new predicted plant states and outputs are calculated for the new prediction horizon $t + 1 + p$. This way, the prediction horizon keeps shifting, every time p seconds ahead of the current time t. This principle is illustrated in figure 4.1.

In this research a continuous time MPC is used. This means that a continuous time model is used to calculate the system behavior and the desired control trajectory. The execution of the algorithm does use a discrete time setting in the sense that for a predefined time step the control effort is executed before a new optimization step is made.

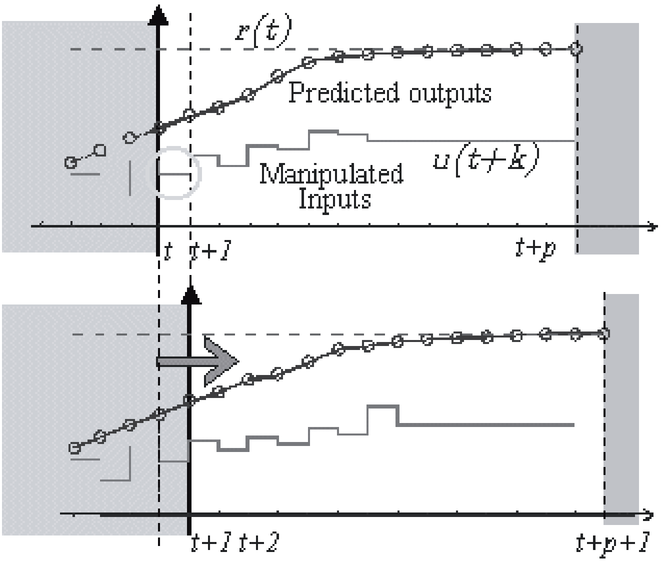


Figure 4.1: Receding horizon control

4.1.2 Optimal control formulation

The plant behavior can be described by:

$$\frac{dx}{dt} = f(x, u, t) \tag{4.1}$$

The model outputs are calculated up to time p. The control objective is to minimize a cost function, which has the form:

$$V(x, u, t) = \int_0^p l(x(t), u(t), t)dt + F(x(p)) \tag{4.2}$$

where $l(x(t), u(t), t)$ is some function, satisfying $l(x(t), u(t), t) \geq 0$, $F(x(p))$ is the terminal state weighting (at the prediction horizon p) and the control input is constrained to be in some set $u(t)\epsilon U$. This is a very general formulation, and just about every sensible problem can be represented in this form by using suitable functions f, F and l.

To solve the minimization of the cost function, the following partial differential equation needs to be solved:

$$\frac{\partial}{\partial t}V^0(x, t) = min_{u\epsilon U} H(x, u, \frac{\partial}{\partial x}V^0(x, t)) \tag{4.3}$$

Where $H(x, u, \lambda) = l(x, u) + \lambda f(x, u)$, with the boundary condition $V(x, p) = F(x(p))$. This equation is called the Hamilton-Jacobian-Bellman equation and in most cases it is virtually impossible to solve this partial differential equation. It is necessary to make some assumptions to get solutions in practice. One way is to assume the plant is linear, so that $f(x, u, t)$ has the special form:

$$\dot{x} = A(t)x(t) + B(t)u(t) \tag{4.4}$$

and that the functions l and F are quadratic functions:

$$\begin{aligned} l(x(t), u(t), t) &= x^T(t)Q(t)x(t) + u^T(t)R(t)u(t) \qquad (4.5)\\ F(x(p)) &= x^T(p)Sx(p) \qquad (4.6) \end{aligned}$$

where $Q(t)$, $R(t)$ and S are square, positive-definite matrices. In that case the Hamilton-Jacobi-Bellman equation simplifies to an ordinary differential equation, the Ricatti equation. A candidate function to solve the HJB equation is to maintain a quadratic form of the cost function at all time t. Assume that:

$$V^0(x, t) = x^T(t)P(t)x(t), \qquad P(t) = P^T(t) \tag{4.7}$$

Then, the HJB equation is simplified to the following Riccati equation:

$$\begin{aligned} -\dot{P}(t) &= P(t)A(t) + A(t)^T P(t) + Q(t) - P(t)B(t)R^{-1}(t)B(t)^T P(t) \qquad (4.8)\\ P(p) &= S \qquad (4.9) \end{aligned}$$

This can be solved and leads to a linear, time-varying feedback law:

$$u(t) = F(t)x(t) \tag{4.10}$$

where $F(t)$ is the 'state feedback' matrix, which depends on the solution of the Ricatti equation.

$$F(t) = R^{-1}(t)B(t)^T P(t) \tag{4.11}$$

4.2 Continuous time model predictive control design using orthonormal functions

4.2.1 Introduction

Most of the MPC techniques developed over the last years have focussed on discrete time systems. In recent years the interest in continuous time model predictive control has grown. In general, continuous time MPC is more complicated than its discrete time counterpart. This is because in the continuous time case, the future plant behavior is calculated through convolution instead of direct iteration, which is computationally more demanding. An emulator was proposed to obtain the predictions through a Taylor series expansion in [18, 19, 20].

One of the crucial steps widely used in the derivation of discrete time MPC is to capture the projected control signal over a prediction horizon using a moving average structure. To make the structure finite dimensional, long term increments of the future control signal are assumed to go to zero. This technique can not be naturally adopted in continuous time MPC design where the corresponding components are the derivatives of the control signal over the prediction horizon. These can not be assumed to be zero without affecting closed-loop performance.
Based on a state space model, Gawthrop [21] proposed to use a set of exponential

functions to capture the control trajectory with the parameters related to the location of the closed-loop poles. In his method the number of exponential functions matches the number of state variables.

The design technique presented here can be used as an alternative to classical receding horizon control without the need to solve the matrix differential Ricatti equation. Future prediction is calculated in an analytical form. The control trajectory is calculated using a pre-chosen set of orthonormal basis functions. This technique is designed for continuous time, but it can be readily extended to discrete time systems and it is expected to be applicable to non-linear systems as well [1]. The first derivative of the control signal can be approximated within the prediction horizon using orthonormal basis functions [22]. The accuracy of the approximation, when using orthonormal functions, is dependent on the choice of certain scaling factors. For example, when using a Laguerre network, the pole location is important. To simplify the performance specification procedure, Laguerre orthonormal functions are used in this design. The scaling factor is made a performance tuning parameter. This parameter plays an important role in the response speed of the closed-loop system.

The novelty of this approach lies in the fact that the problem of finding the optimal control signal is converted into one of finding a set of coefficients for the Laguerre model. This approach effectively reduces the number of parameters required in the operation, thus offering a substantial advantage when used in an on-line environment.

4.2.2 Description of control trajectory

It is known that an arbitrary function $f(t)$ has a formal expansion analogous to a Fourier expansion [23]. Based on this technique, an arbitrary function $f(t)$ is expressed in terms of a series expansion as

$$f(t) = \sum_{i=1}^{\infty} \xi_i l_i(t) \tag{4.12}$$

where ξ_i, $i = 1, 2, \ldots$ are the coefficients and $l_i(t)$, $i = 1, 2, \ldots$ are orthogonal functions satisfying

$$\int_0^{\infty} l_i(t)^2 dt = 1 \qquad \int_0^{\infty} l_i(t) l_j(t) dt = 0 \qquad i \neq j \tag{4.13}$$

Further, assuming that $f(t)$ is a piece-wise continuous function satisfying

$$\int_0^{\infty} f(t)^2 dt < \infty \tag{4.14}$$

then for any $\epsilon > 0$ and $0 \leq t \leq \infty$, there exists an integer N such that for all $k \geq N$

$$\int_0^{\infty} (f(t) - \sum_{i=1}^{k} \xi_i l_i(t))^2 dt \leq \epsilon \tag{4.15}$$

In summary, a truncated expansion $\sum_{i=1}^{N} \xi_i l_i(t)$ is used to closely approximate $f(t)$.

One set of orthonormal functions frequently used is the set of Laguerre functions.

Laguerre functions are appealing to engineers because the Laplace transform of $l_i(t)$ has a particularly simple form:

$$\int_0^\infty l_i(t)e^{-st}dt = \sqrt{2p}\frac{(s-p)^{i-1}}{(s+p)^i} \tag{4.16}$$

where p is positive and often called scaling factor. From (4.16), a differential equation can be derived, which is satisfied by the Laguerre functions. Specifically, let $L(t) = [l_1(t) \ \ l_2(t) \ \ \dots \ \ l_N(t)]^T$ and $L(0) = \sqrt{2p}[1 \ \ 1 \ \ \dots \ \ 1]^T$. Then the Laguerre functions satisfy the differential equation:

$$\dot{L}(t) = A_p L(t) \tag{4.17}$$

where

$$A_p = \begin{bmatrix} -p & 0 & \dots & 0 \\ -2p & -p & \dots & 0 \\ \vdots & & \ddots & \vdots \\ -2p & \dots & -2p & -p \end{bmatrix} \tag{4.18}$$

The solution of the differential equation (4.17) leads to a representation of the Laguerre functions in terms of a matrix exponential function:

$$L(t) = e^{A_p t}L(0) \tag{4.19}$$

For a linear time invariant system, when the closed loop system is stable, the control signal for a set-point change or a load disturbance rejection exponentially converges to a constant after the transient response period. Using this in the receding horizon control design, the future control signal, derived in section 4.1.2 responds in a time invariant fashion for each moving horizon window. Thus the derivative of the control signal $\dot{u}(t)$ converges to zero, when it is assumed that the closed loop system is stable within each moving horizon window $t_i \leq t \leq t_i + T_p$. It is seen that:

$$\int_{t_i}^{t_i+T_p} \dot{u}(t)^2 dt < \infty \tag{4.20}$$

By applying the technique used by Wiener and Lee [22], the derivative of the control signal can be described by using a Laguerre function based series expansion as:

$$\dot{u}(t) = \sum_{i=1}^{N} \xi_i l_i(t) = L(t)^T \eta \tag{4.21}$$

where $\eta = [\xi_1 \ \ \xi_2 \ \ \dots \ \ \xi_N]^T$ is the vector of coefficients of the orthogonal functions $l_i(t)$ which express the series expansion of $\dot{u}(t)$.

Remarks:

- Assume that a moving time window is taken as $t_i \leq t \leq t_i + T_p$, and within this window the control signal is denoted by $u(t)$ and its derivative by $\dot{u}(t)$. In general, the control signal in a stable closed-loop system does not converge to zero as $t \rightarrow \infty$ if there exists a constant external signal such as a set-point signal, step input disturbance or step output disturbance.
  Suppose that for $t_i \leq t \leq t_i + T_p$, the projected control signal $u(t)$ converges exponentially to a constant, consequently leading to the exponential convergence of the derivative, $\dot{u}(t)$, to zero. As a result the orthonormal functions

can not be used to describe the control signal with the expectation of convergence of the expansion in a finite number of terms. However, the derivative of the control signal does preserves the property that $\int_0^\infty \dot{u}(t)^2 dt \leq \infty$.
In summary, if there is an external constant input signal to the closed-loop control signal, the suitable candidate variable which can be modelled by a set of orthonormal functions is the derivative of the control signal, not the control signal itself. This is a realistic situation as setpoint change and disturbance rejection are the primary roles in which a feedback control system plays.

- Model predictive control is an open-loop optimisation strategy unless the state variable $x(t)$ is measurable. The actual feedback is introduced through the use of an observer to estimate the state variables (see section 4.2.4). Also, by introducing the controller, the open loop properties of the system are affected. Assuming that the closed loop system is stable is therefore questionable.

4.2.3 Predicted plant output

The plant to be controlled is a multivariable state space system with r inputs $u(t)$ and q outputs $y(t)$. Furthermore, it is assumed that the plant suffers from external disturbances $\omega(t)$ (e.g. changing wind, temperature change of the air, changes of the air density) and the measurements of the outputs are disturbed with $\mu(t)$:

$$\begin{aligned} \dot{x}_m(t) &= A_m x_m(t) + B_m u(t) + \omega(t) \\ y(t) &= C_m x_m(t) + D_m u(t) + \mu(t) \end{aligned} \tag{4.22}$$

where $x_m(t)$ is the state vector of dimension n. It is assumed that

$$\begin{aligned} E\left\{\frac{d\omega(t)}{dt}\right\} &= 0 \\ E\left\{\frac{d\mu(t)}{dt}\right\} &= 0 \end{aligned} \tag{4.23}$$

and

$$\begin{aligned} E\left\{\frac{d\omega(t)}{dt}\frac{d\omega(\tau)^T}{d\tau}\right\} &= W_\omega \delta(t-\tau) \\ E\left\{\frac{d\mu(t)}{dt}\frac{d\mu(\tau)^T}{d\tau}\right\} &= R_\mu \delta(t-\tau) \end{aligned} \tag{4.24}$$

where $E\{\}$ denotes expectation and $\delta(.)$ is the Dirac function. Here it is assumed that $\omega(t)$ and $\mu(t)$ are continuous time white noise processes.

Introduce the variable

$$\dot{z}(t) = \ddot{x}_m(t) = \frac{d}{dt}\Bigg(A_m x_m(t) + B_m x_m(t) + \omega(t)\Bigg) \tag{4.25}$$

The state space system (4.22) can now be written in an augmented form:

$$\begin{aligned} \dot{X}(t) &= AX(t) + B\dot{u}(t) + \begin{bmatrix} \dot{\omega}(t) \\ \dot{\mu}(t) \end{bmatrix} \\ y(t) &= CX(t) \end{aligned} \tag{4.26}$$

where

$$X(t) = \begin{bmatrix} z(t) \\ y(t) \end{bmatrix} \quad A = \begin{bmatrix} A_m & 0 \\ C_m & 0 \end{bmatrix} \quad B = \begin{bmatrix} B_m \\ D_m \end{bmatrix} \quad C = [0 \;\; I]$$

I is the $q \times q$ unit matrix. Note that the augmented state space description (4.26) takes the first derivative of the control signal as its input and its output remains the same.
Because the white noise has a zero average, the expected effect of the random disturbances ($\omega(t)$, $\mu(t)$) in the future prediction is assumed to be zero. The measurement noise $\omega(t)$ and the disturbances $\mu(t)$ are not amplified, because an observer is used to estimate the state variables so that explicit differentiation is avoided. Because of this, the noise terms are neglected in this analytical approach, and it is not necessary to calculate the derivative of $\omega(t)$ and $\mu(t)$ in equation (4.26). Simulations show that the presence of measurement noise and disturbances do not influence the performance of the controller [1].
It can be shown that if the augmented state space system is observable and controllable, it is possible to track the outputs of the original system using a model predictive controller [26]. Of course, this only holds when there are no constraints active on the system.

It is assumed that at the current time t_i, the state variable $X(t_i)$ is available. Then at the future time $t_i + \tau$, $\tau > 0$, the predicted state variable $X(t_i + \tau)$ is described by the following equation:

$$\begin{aligned} X(t_i+\tau) &= e^{A\tau}X(t_i) + \int_{t_i}^{t_i+\tau} e^{A(t_i+\tau-\beta)}B\dot{u}(\beta)d\beta && (4.27) \\ &= e^{A\tau}X(t_i) + \int_{0}^{\tau} e^{A(\tau-\gamma)}B\dot{u}(t_i+\gamma)d\gamma && (4.28) \\ & && (4.29) \end{aligned}$$

where the expected effect of the random disturbances ($\dot{\omega}(t)$, $\dot{\mu}(t)$) in the future prediction is zero. The projected control trajectory $\dot{u}(t)$ will be parameterized using the orthonormal functions described in section 4.2.2.

The control signal can be written as:

$$\dot{u}(t) = [\dot{u}_1(t) \quad \dot{u}_2(t) \quad \ldots \quad \dot{u}_r(t)]^T$$

and the input matrix B as:

$$B = [B_1 \quad B_2 \quad \ldots \quad B_r]$$

where B_i is the i-th column of the B matrix. The i-th control signal $\dot{u}_i(t)$ ($i = 1, 2 \ldots r$) is expressed as:

$$\dot{u}_i(t) \cong L_i(t)^T \eta_i$$

where $L_i(t)^T = \begin{bmatrix} l_1^i(t) & l_2^i(t) & \ldots & l_{N_i}^i(t) \end{bmatrix}$ and $\eta_i = \begin{bmatrix} \xi_1^i & \xi_2^i & \ldots & \xi_{N_i}^i \end{bmatrix}^T$. p_i and N_i are pre-chosen. Then the predicted future state at time $t_i + \tau$, $X(t_i + \tau)$, is

$$\begin{aligned} & X(t_i+\tau) = e^{A\tau}X(t_i) \\ + & \int_0^{\tau} e^{A(\tau-\gamma)} \begin{bmatrix} B_1L_1(\gamma)^T & B_2L_2(\gamma)^T & ... & B_rL_r(\gamma)^T \end{bmatrix} \eta d\gamma \qquad (4.30) \end{aligned}$$

where the coefficient vector $\eta^T = \begin{bmatrix} \eta_1^T & \eta_2^T & \ldots & \eta_r^T \end{bmatrix}^T$ has dimension $\sum_{i=1}^r N_i$. Accordingly, the plant output prediction can be represented by

$$y(t_i + \tau) = CX(t_i + \tau) \qquad (4.31)$$

The major computation load in evaluating the prediction comes from the convolution operation in (4.30), which requires the solutions of $(n + q) \times \sum_{i=1}^r N_i$ integral equations. For a given τ, the integral equations can be solved numerically using finite sum approximation. However, such a numerical approximation is unnecessary and analytical solutions can be found for the convolution integral.

The convolution integral corresponding to the i-th input is defined as:

$$I_{int}(\tau)^i = \int_0^\tau e^{A(\tau-\gamma)} B_i \;\; L_i(\gamma)^T d\gamma \tag{4.32}$$

where $I_{int}(\tau)^i$ is a matrix with dimensionality of $(n+q) \times N_i$. Substituting this in (4.30) shows that the prediction of the future trajectory can be expressed in terms of $I_{int}(\tau)^i$ with $1 \leq i \leq r$. The matrix $I_{int}(\tau)^i$ can be expressed as:

$$AI_{int}(\tau) - I_{int}(\tau)A_p^T = -BL(\tau)^T + e^{A\tau}BL(0)^T \tag{4.33}$$

where $L(\tau)^T$, $L(0)^T$ and A_p are defined by equations (4.17), (4.18) and (4.19). Upon obtaining the matrices $I_{int}(\tau)^i$ for $i = 1, 2, \ldots, r$, as outlined above, the prediction $X(t_i+\tau)$ can be determined. With this the predicted plant output can be calculated using (4.31).

4.2.4 Optimal control strategy

To compute the optimal control a cost function has to be optimized. Suppose that a set of future setpoints $r(t_i+\tau) = \left[\begin{array}{cccc} r_1(t_i+\tau) & r_2(t_i+\tau) & \ldots & r_q(t_i+\tau) \end{array} \right]$, $0 \leq \tau \leq T_p$ are available, where T_p is the prediction horizon. The common objective of model predictive control is to find the control law that will drive the predicted plant output $y(t_i+\tau)$ as close as possible, in a least squares sense, to the future trajectory of the setpoint $r(t_i+\tau)$. The equivalent cost function can be defined as:

$$J = \int_0^{T_p} [r(t_i+\tau)-y(t_i+\tau)]^T Q [r(t_i+\tau)-y(t_i+\tau)] d\tau + \int_0^{T_p} \dot{u}(\tau)^T R \dot{u}(\tau) d\tau \tag{4.34}$$

where Q and R are symmetric matrices with $Q > 0$ and $R \geq 0$. To simplify the tuning process, Q is set to the unit matrix I and R to zero. In [1] it is shown that the performance of the MPC is predominantly determined by tuning the values of p and N and tuning of Q and R is therefore not necessary. The tuning of the MPC will be done by changing the parameters p (poles of the Laguerre functions) and N (number of orthonormal functions).

The cost function can be written as a function of η instead of $y(t_i+\tau)$. Assuming further that the reference trajectory does not change within the prediction horizon T_p, the quadratic cost function can be written as:

$$J = \eta^T \Pi \eta - 2\eta^T \{\Psi_1 r(t_i) - \Psi_2 X(t_i)\} + \int_0^{T_p} w^T(t_i+\tau) Q w(t_i+\tau) d\tau \tag{4.35}$$

where

$$\Pi = \int_0^{T_p} \phi(\tau) Q \phi(\tau)^T d\tau + \bar{R}$$

$$\Psi_1 = \int_0^{T_p} \phi(\tau) Q d\tau \;\; ; \;\; \Psi_2 = \int_0^{T_p} \phi(\tau) Q C e^{A\tau} d\tau$$

and

$$\bar{R} = diag\left(\lambda_i \mathbf{I}_{N_i \times N_i}\right)$$

where λ_i are the eigenvalues of extended system matrix A and $\mathbf{I}_{N_i \times N_i}$ is the $N_i \times N_i$ unity matrix.

The minimum of (4.35), without hard constraints on the variables, is then given by the least squares solution:

$$\eta = \Pi^{-1}\{\Psi_1 r(t_i) - \Psi_2 X(t_i)\} \tag{4.36}$$

The derivative of the control input $\dot{u}(t)$ can now be computed by:

$$\dot{u}(t_i) = \begin{bmatrix} L_1(0)^T & 0 & \dots & 0 \\ 0 & L_2(0)^T & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & L_r(0)^T \end{bmatrix} \Pi^{-1}\{\Psi_1 r(t_i) - \Psi_2 X(t_i)\} \tag{4.37}$$

Integration gives the control law:

$$u(t) = \int_0^t \dot{u}\,(\tau)\,d\tau \tag{4.38}$$

The cost function can be expanded by adding weighting to the terminal state [1], to guarantee closed-loop stability for continuous time model predictive control [18].

Because the cost function (4.35) is a quadratic function, it is relatively easy to put hard constraints on the output-, the control- and the first derivative of the control variables. The optimization of the cost function, subject to a number of inequality constraints is a standard quadratic programming problem. To form a set of inequality constraints for the output and control signals, discretization of the trajectories is necessary.
The bounds on the derivative of the control can be defined as:

$$\dot{u}_{low}(t_i+\tau_i) \le \begin{bmatrix} L_1(\tau_i)^T & 0 & \dots & 0 \\ 0 & L_2(\tau_i)^T & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & L_r(\tau_i)^T \end{bmatrix} \eta \le \dot{u}_{high}(t_i+\tau_i) \tag{4.39}$$

where τ_i, $i = 0, 1, \dots, K$, denotes the set of future time instants at which we wish to impose the limits on $\dot{u}$. (4.39) yields a set of linear inequality equations. Since $L_k(\tau)$, $k = 1, 2, \dots, r$ are the exponential functions, which guarantees the exponential decay of $\dot{u}(t_i + \tau)$, it is necessary to apply the constraints only to the initial stage of the prediction horizon. This feature could potentially reduce the number of constraints required.

Constraints on the control signal, the output variables and the system states can be defined as follows:

$$\begin{aligned} u_{low}(t_i+\tau_i) \;&\le\; \begin{bmatrix} \int_0^{\tau_i} L_1(\gamma)^T d\gamma & 0 & \dots & 0 \\ 0 & \int_0^{\tau_i} L_2(\gamma)^T d\gamma & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \int_0^{\tau_i} L_r(\gamma)^T d\gamma \end{bmatrix} \eta + u(t_i - \Delta t) \\ &\le\; u_{high}(t_i+\tau_i) \end{aligned} \tag{4.40}$$

where $u(t_i - \Delta t)$ is the previous control signal. Simple computation shows that, with p and N suitably chosen,

$$\int_0^{\tau_i} L_k(\gamma)^T d\gamma = \left[A_p^{-1}(e^{A_p \tau_i} - I)L(0)\right]^T \tag{4.41}$$

where A_p is defined by equation (4.18). Thus, for a given set of time instants τ_i, equation (4.40) yields a set of linear inequality constraints on the control signal.

$$X_{low} \leq e^{A\tau_i} X(t_i) + \left[\begin{array}{cccc} I_{int}(\tau_i)^1 & I_{int}(\tau_i)^2 & \dots & I_{int}(\tau_i)^r \end{array} \right] \eta \leq X_{high} \tag{4.42}$$

and

$$y_{low} \leq Ce^{A\tau_i} X(t_i) + C \left[\begin{array}{cccc} I_{int}(\tau_i)^1 & I_{int}(\tau_i)^2 & \dots & I_{int}(\tau_i)^r \end{array} \right] \eta \leq y_{high} \tag{4.43}$$

The whole procedure described above assumes that the states at time t_i are known. In general it is not possible to measure the state of a plant. The standard approach is the use of an observer to estimate te state variables $X(t_i)$ [24, 25]. The observer equation that is needed for the implementation of the continuous time MPC is given by:

$$\dot{\hat{X}}(t) = A\hat{X}(t) + B\dot{u}(t) + J_{ob}(y(t) - C\hat{X}(t)) \tag{4.44}$$

Here $\hat{X}(t)$ is the estimate of $X(t)$ and J_{ob} is the observer gain. $\dot{u}(t)$ is determined from the optimal solution of the model predictive control strategy, either using the analytical solution or a quadratic programming approach. The observer can be designed using a continuous time steady-state Kalman filter or using other techniques such as pole assignment. Here, J_{ob} is calculated recursively using a Kalman filter. The iterative solution of the Riccati equation is not required in real-time. The observer gain is calculated off-line. Provided that the system model (A, B, C, D) is completely observable, in theory, J_{ob} can be chosen such that the error, $\tilde{X}(t) = X(t) - \hat{X}(t)$, decays exponentially at any desired rate. However, in practice, the observer gain J_{ob} is often limited by the existence of measurement noise.

Chapter 5

Simulations and results

5.1 Model selection

From the original twelve outputs:
$Y = \begin{bmatrix} n_x & n_y & n_z & V_T & \alpha & \beta & P & Q & R & \phi & \theta & \psi \end{bmatrix}^T$ of the Ariel model, up to 4 outputs can be selected for tracking, to satisfy the condition that the extended state space system must be both observable and controllable to be able to track the selected outputs [26].

Several linearized models are obtained by linearizing at different trim conditions for steady wings level flight. In table 5.1 a list of the different operating points is shown. The exact trim conditions can be found in appendix D. Different altitudes (100, 1000 and 5000 m) and velocities (30 and 55 m/s) are selected.

The rank of the controllability matrix is smaller than the rank of matrix A for the higher velocity. This is because the constraints on the aircraft are active in the nonlinear Ariel model. At this trim condition the throttle is at its maximum, thus being unable to increase the velocity. This results in a bad controllability condition, implying that tracking of the selected outputs is not possible, but simulations show that it is still possible to operate at this condition, as long as no increase in throttle is required.

Altitude $[m]$	Velocity $[m/s]$	Rank A	*obsv*	*ctrb*
100	30	10	13	12
100	55	10	13	5
1000	30	10	14	12
1000	55	10	13	5
5000	30	10	14	14
5000	55	10	13	8

Table 5.1: Observability and controllability of different linearized models

Simply put, by introducing the extended state space model, the selected tracking outputs become part of the state space system. If the extended system is observable and controllable, then all the (extended) states can be controlled. This means that the selected outputs can be controlled. Given the fact that there are four control inputs, it seems logical that it is possible to control a maximum of four outputs.

If more then 4 outputs are selected for the extended state space system, the observability and controllability conditions are no longer satisfied and the outputs and states of the original system can no longer be controlled and/or tracked. Therefore, the maximum number of outputs that can be selected for this particular system is equal to four. Other systems may be able to include more or less outputs, depending on the specific system dynamics.

All the simulations are done by using one linearized time invariant model. For better performance and more reliable results, a switching controller can be designed, which switches between different linearized models when the operating conditions of the aircraft change. Switching between different models can cause instability, and care should be taken when a switching controller is applied that the overall stability of the system is maintained. However, to show that this control method works, this does not necessarily needs to be implemented. As can be seen in section 5.4.1, only the magnitudes of the state space system differ for the different operating conditions, the general dynamic behavior remains the same.

For the four outputs that are tracked, different reference signals are defined. The LTI model obtained by linearizing the Ariel model at 1000 m altitude and 30 m/s velocity is used in all the simulations. This model is specified in appendix B and operates at the following equilibrium point:

$$x_0 = \begin{bmatrix} U \\ V \\ W \\ \phi \\ \theta \\ \psi \\ P \\ Q \\ R \\ p_N \\ p_E \\ p_D \end{bmatrix} = \begin{bmatrix} 30 \\ 0.012 \\ 2.052 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0.068 \\ 0 \\ 0 \\ 0 \\ -1000 \end{bmatrix} \quad y_0 = \begin{bmatrix} n_x \\ n_y \\ n_z \\ V_T \\ \alpha \\ \beta \\ P \\ Q \\ R \\ \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} 0.068 \\ 0 \\ 0.998 \\ 30.07 \\ 0.068 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0.068 \\ 0 \end{bmatrix} \quad u_0 = \begin{bmatrix} \delta_T \\ \delta_e \\ \delta_a \\ \delta_r \\ \delta_f \end{bmatrix} = \begin{bmatrix} 0.238 \\ 2.790 \\ 0.328 \\ 0.061 \\ 1 \end{bmatrix} \tag{5.1}$$

The reference signal is the change from the steady state operating conditions. After the simulations, the values of the steady state operating conditions are added to the system states, outputs and controls to obtain the 'real', physical values that can be fed into the nonlinear system. These 'real' values are the values shown in all the simulation plots.

5.2 Output selection

One of the four outputs that needs to be selected is the yaw angle ψ. This is a property of this particular system and is needed, because otherwise the system needs to be reduced by another state, ψ. Without being able to control the yaw angle of the aircraft, it is not possible to keep the aircraft from drifting off course.

By doing different simulations with different selected outputs for tracking, it can be seen that not all output variables give the desired system performance. Compare two different selections; in figure 5.1 the outputs n_x, n_y, n_z and ψ have been chosen. In figure 5.2 the simulation results are shown when the outputs V_T, ϕ, θ and ψ are chosen. To compare the performance, both times a step is put on the yaw angle. This way it is possible to compare the response of the system and the resulting actions of the four system inputs for the two selections.

Although the tracking in both scenarios is good, e.g. the outputs all follow the given setpoint (note that n_y will also converge over time), the control effort and the effect on the internal states differ. In appendix C the difference between the control efforts and the states of the system can be seen. The control effort when choosing as reference output the three angles and the velocity for tracking, seems to give a more 'natural' aircraft control than the other selection of outputs. The step op the yaw angle is mainly taken care of by the rudder, which seems a logical control effort.

By selecting the accelerations as output, the step on the yaw angle results in a much smaller effort of the rudder. The throttle and elevator also take hold of a big part of the total control effort. This seems a much less natural way of controlling an aircraft. Also, the response is much faster when the angles and velocity are selected as output.

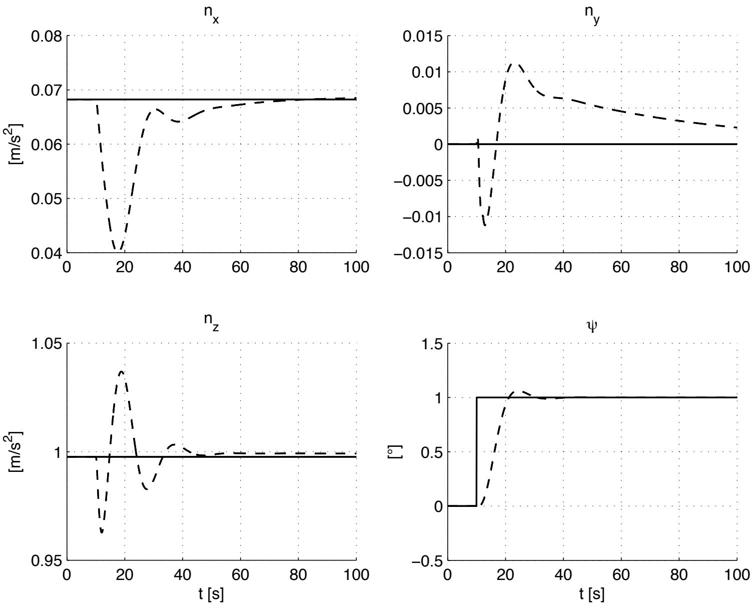


Figure 5.1: Tracking of the outputs n_x, n_y, n_z and ψ (solid: reference, dashed: control signal)

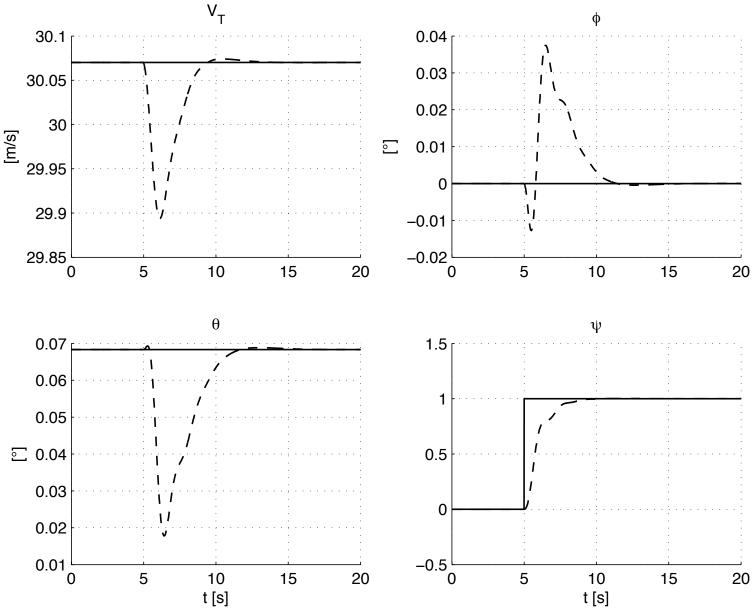


Figure 5.2: Tracking of the outputs V_T, ϕ, θ and ψ (solid: reference, dashed: control signal)

After comparing several possible output combinations, the following configurations came out best:

Configuration	Outputs
1	$V_T \quad \phi \quad \theta \quad \psi$
2	$n_y \quad V_T \quad \theta \quad \psi$
3	$V_T \quad P \quad \theta \quad \psi$

Table 5.2: Selected outputs

The choice of the outputs seems logical. It is important to keep track of the velocity V_T to keep the aircraft flying. The yaw angle ψ or the acceleration in y-direction n_y seems a good choice to keep the aircraft on the right course. And the pitch angle θ is a good choice to control the altitude of the aircraft. In the following simulations, the outputs of configuration 1 have been chosen as controlled output variables.

5.3 Tuning

The behavior of the MPC can be changed by adjusting the location of the poles p, the order of the control trajectory N, the prediction horizon T_p, the weighting matrices Q and R and the weighting matrix for the system inputs:

- parameter T_p : Prediction horizon;
- parameter p: Pole location of the Laguerre model for the projected control signal; also the key parameter for control horizon;

- parameter N: Number of terms used in the Laguerre model to capture the projected control signal;
- weighting matrices: Q and R. As mentioned in section 4.2.4, $Q = I$ and $R = 0$ to simplify the tuning procedures;
- weighting matrix W: When constraints are applied the efforts of the controls need to be scaled.

Prediction horizon T_p: For a stable system, the prediction horizon is recommended to be chosen approximately equal to the open loop process settling time, i.e. the time that the response reaches 99 percent of its final value. For unstable and integrating processes, T_p is recommended to be chosen depending on the value of p, approximately equal to $10/p$, i.e. larger than the time that the projected control signal effectively reaches a constant [1].

Pole location p: p is a tuning parameter for the closed-loop response speed. Larger p results in larger changes in the projected control signal in the initial period, and thus faster closed-loop response speed. The most effective tuning procedure for p is to start near the location of the plant dominant pole, and to then subsequently speed up (or slow down) the closed-loop response by increasing (or decreasing) p to achieve the required performance. The upper performance limit is reached when increasing p no longer changes the closed-loop response speed. It was found that an upper limit was reached for $|p| > 0.1$.

Parameter N: The parameter N stands for the number of terms that will be used in capturing the projected control signal. As N increases, the degrees of freedom in describing the control trajectory increases. The control signal tends to be more aggressive when N increases. It was found that up to $N = 9$ the speed of the response still increases quite significantly. Increasing N further still gives a little bit better results, but it is not really worth the extra computational effort for the small increase in response.

Weighting matrix W: When there are constraints active on the control signal, these need to be scaled in such a was that the relative action of the controls is equally weighted. The constraints on the Ariel model are shown in table 5.3.

Value	min	max	rate min	rate max
$\delta_{throttle}$ [% of max]	1	100	-200	200
$\delta_{elevator}$ [°]	-32	16	-250	250
$\delta_{aileron}$ [°]	-16	16	-250	250
δ_{rudder} [°]	-16	16	-250	250

Table 5.3: Input constraints

These are weighted and scaled as follows:

$$W = (\delta_{e,max} - \delta_{e,min}) \cdot \begin{bmatrix} \delta_{t,max} - \delta_{t,min} & 0 & 0 & 0 \\ 0 & \delta_{e,max} - \delta_{e,min} & 0 & 0 \\ 0 & 0 & \delta_{a,max} - \delta_{a,min} & 0 \\ 0 & 0 & 0 & \delta_{r,max} - \delta_{r,min} \end{bmatrix}^{-1} \tag{5.2}$$

5.4 Simulations

5.4.1 Simulation without constraints

Several scenarios are simulated to show that the controller will always stabilize the system, if no constraints are applied. Here simulations are shown when 5 m/s steps are put on the velocity until it reaches 55 m/s. As can be seen in figure 5.3, the reference signal is very well tracked.

What can be seen, is that the value of input δ_t goes to 114%, which is higher then the value at trim condition with a velocity of 55 m/s. With increased velocity, the pitch angle θ should be reduced to keep the aircraft at steady flight. Without decreasing the pitch angle the aircraft starts to climb, which seems a logical result. The difference in throttle input, pitch angle and elevator input between the two linearized models both operating at 55 m/s can be seen in table 5.4.
This difference originates from the difference of the atmospheric forces at different velocities, which are implemented in the nonlinear Simulink model by using lookup tables. The data in these tables was obtained by doing wind tunnel testing of the Ariel aircraft by Crump [2]. The linearized model does not take this into account and this is where the difference originates from.

Linearized model	Operating velocity	δ_t	θ	δ_e
30 m/s	55 m/s	114	0.07	3.9
55 m/s	55 m/s	100	0.00	3.3

Table 5.4: Difference in operating values for different models

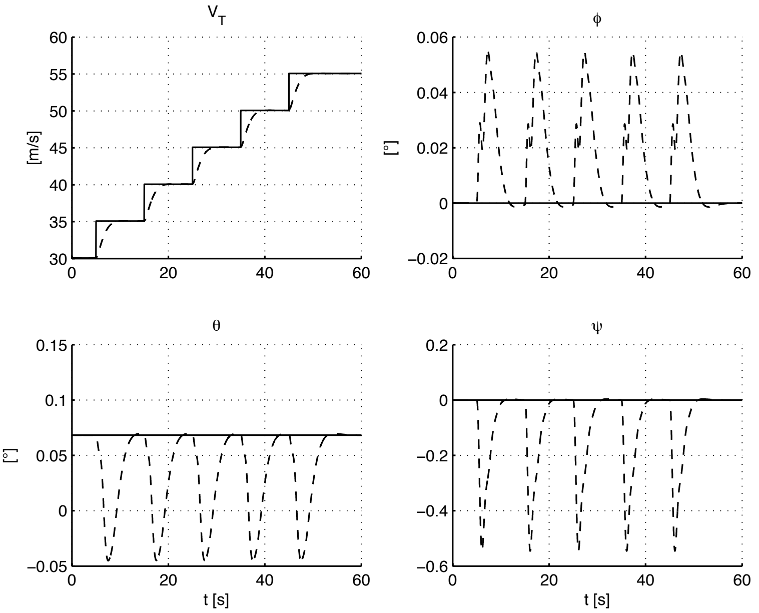


Figure 5.3: System outputs for simulation of steps on the velocity (solid: reference, dashed: control signal)

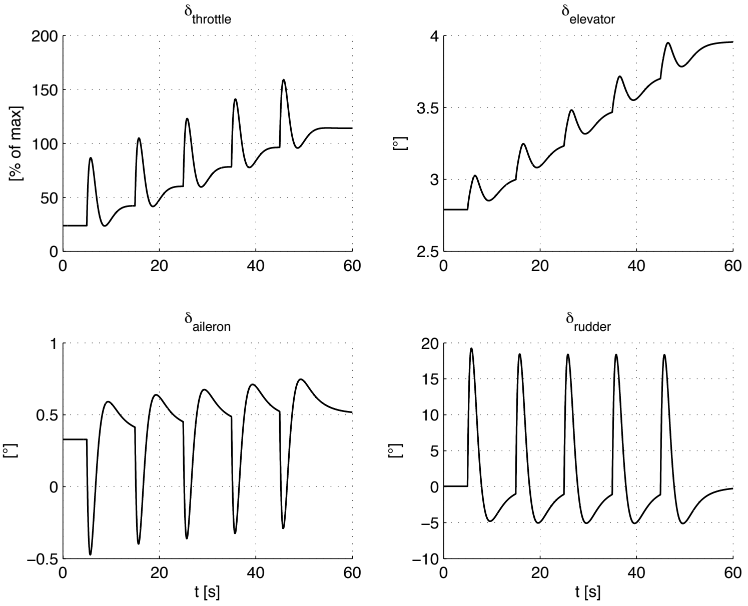


Figure 5.4: System inputs for simulation of steps on the velocity

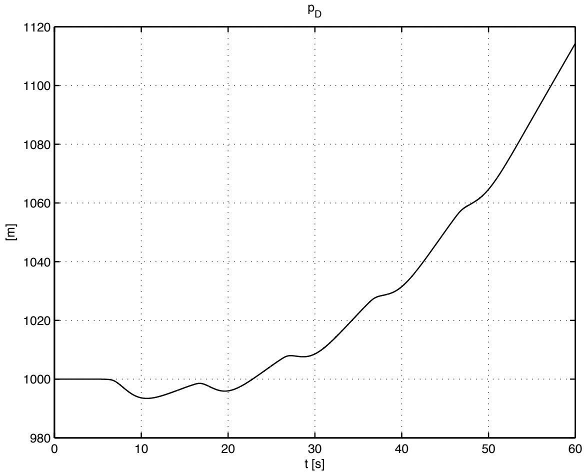


Figure 5.5: Aircraft altitude for simulation of steps on the velocity

5.4.2 Simulation with constraints

In reality, the aircraft controls are constrained. There are constraints on the maximum and minimum values of the controls and on the control rates. The constraints are listed in table 5.5. To make the effort of all the controls the same, the controls are weighted so that the domain of each control input is valued equally.

Value	min	max	rate min	rate max
$\delta_{throttle}$ $[\% of max]$	1	100	-200	200
$\delta_{elevator}$ [°]	-32	16	-250	250
$\delta_{aileron}$ [°]	-16	16	-250	250
δ_{rudder} [°]	-16	16	-250	250

Table 5.5: Input constraints

To see if the controller is capable of following a mission profile, two simulations are made, one increasing the altitude of the aircraft, typically representing the climb to cruise altitude. A second simulation is done decreasing the altitude of the aircraft, e.g. for reconnaissance of the surroundings. The reference trajectory is, as mentioned in section 5.1, superimposed to the steady state operating point of the LTI state space model.

The reference and output trajectory, the controls and the altitude of the simulation are shown in figures 5.6, 5.7 and 5.8 respectively, for the first simulation, where the altitude of the aircraft is increased. The aircraft starts at 1000 m altitude at 30 m/s, begins to climb with a 8° climbing angle, flying 25 m/s and continues a steady flight at 2500 m altitude at 45 m/s.

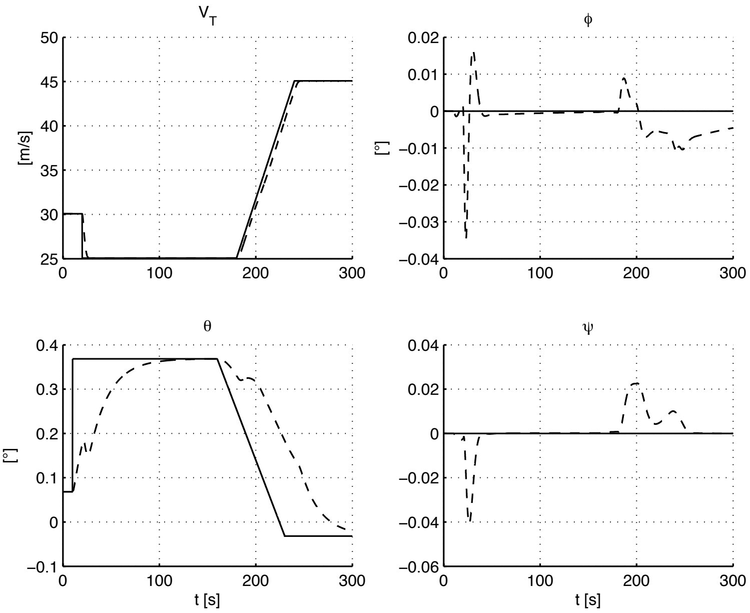


Figure 5.6: Aircraft climb; tracked outputs (solid: reference, dashed: control signal)

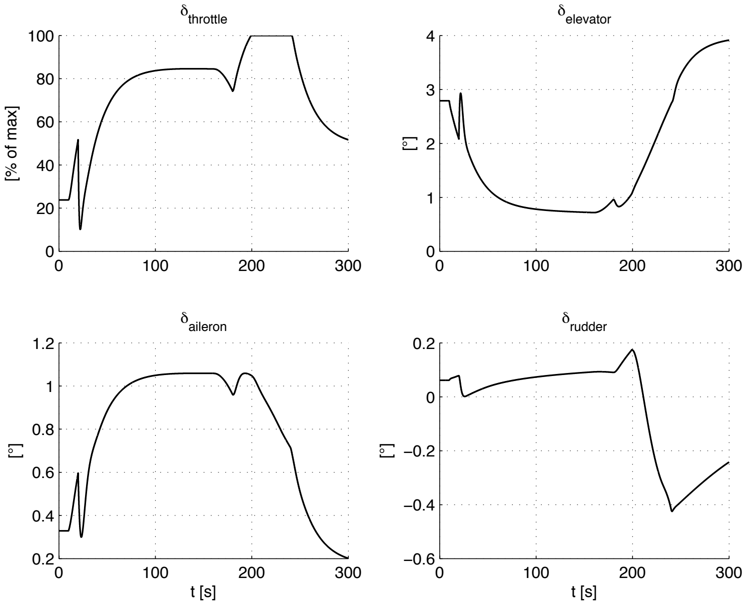


Figure 5.7: Aircraft climb; constrained system inputs

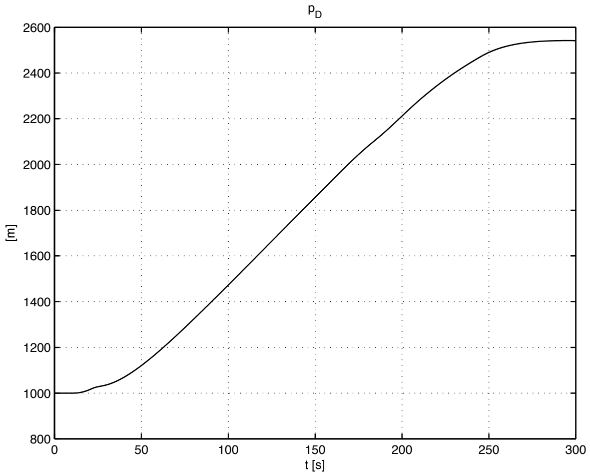


Figure 5.8: Aircraft climb; aircraft altitude

In the second simulation, the altitude is decreased from 1000 m to 200 m. The reference and output trajectory, the controls and the altitude are shown in figures 5.9, 5.10 and 5.11 respectively. The velocity increases during the descent to 45 m/s, and goes back to 30 m/s at the steady levelled flight.

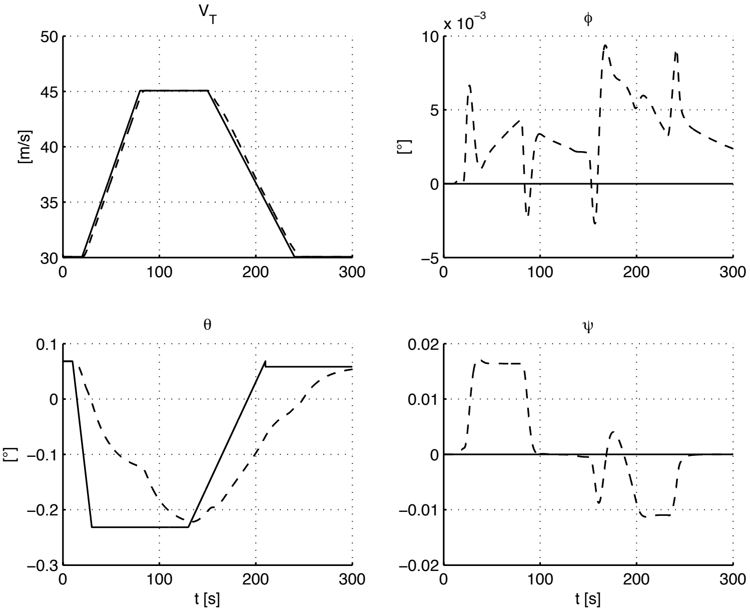


Figure 5.9: Aircraft descent; tracked outputs (solid: reference, dashed: control signal)

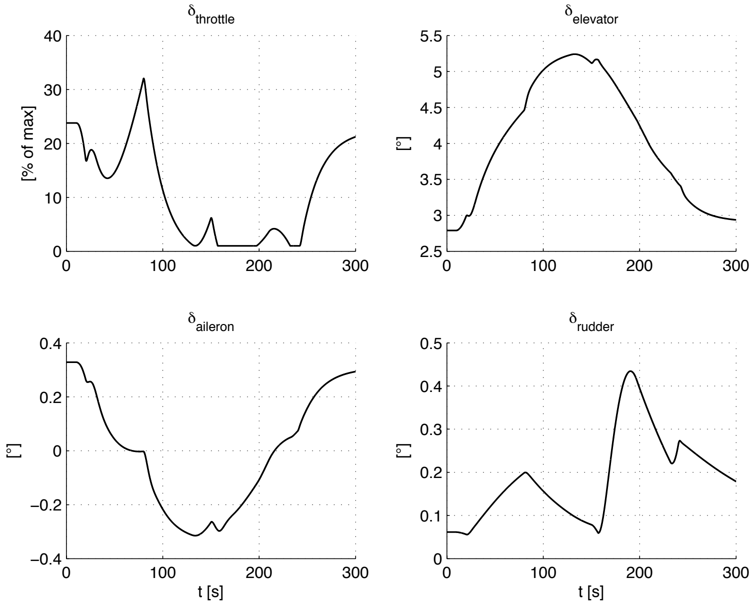


Figure 5.10: Aircraft descent; constrained system inputs

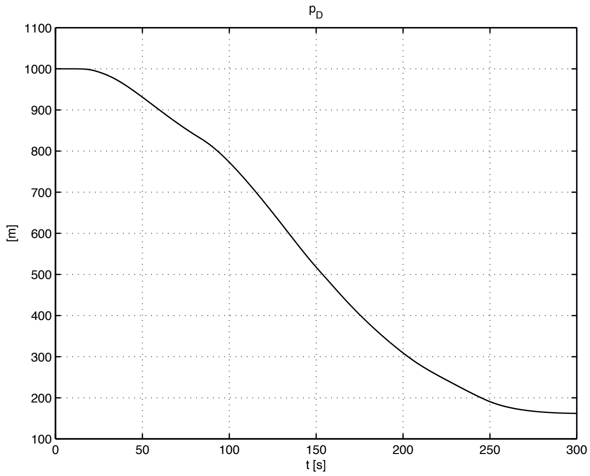


Figure 5.11: Aircraft descent, aircraft altitude

Note that the controls hit the constraints in both situations. In the first simulation, the throttle is fully open (100%), to reach the desired velocity as soon as possible. In the second simulation the throttle is completely closed (1%) when the aircraft starts the descent, and goes down to zero several times after that to prevent the velocity from increasing too much, while the elevator is effecting a change to steady flight.

Defining a good trajectory by hand is quite difficult. An extra controller can be designed to translate for instance a list of way points intro a trajectory for the outputs that the MPC tracks. However, this does not lie within the scope of this research.

5.4.3 Simulation of failure

Another interesting feature of the model predictive controller with constraints applied is the ability to simulate some kind of failure of one of the controls. To simulate one of the controls becoming 'stuck' for instance, the minimum and maximum on the input constraint is set to zero. This capability makes the model predictive control scheme useful for fault-tolerant control applications, within certain limits. The limitation is that the control can only be 'stuck' at it's equilibrium, e.g. $u = u_{fixed} = u_0$. Using a different value for u_{fixed} requires a state space model which is linearized at a different operating point, which takes into account the fact that one of the controls has a different numerical value.

In this simulation the aileron is assumed to be stuck at its initial value, and this means the controller cannot change the banking angle by means of adjusting the ailerons any more. The simulation shows that the airplane is still capable of flying. Although the maneuverability of the airplane is limited, it is still able to fly in a stable, steady flight, allowing the UAV to return home for instance. The simulation has been made with and without the failure of the ailerons. The difference in the tracking of the reference signal and the control inputs can be seen in figures 5.12 and 5.13.

Although the response is not as good without the aileron working, the aircraft is still able to follow the reference trajectory. The control signal of the throttle and the rudder changes to compensate for the failed ailerons.

It will also be interesting to investigate how a random value for one of the controls affects the aircraft model. This can simulate one of the controls being 'free' to move, e.g. when a controller cable has failed. The movement of the control surface will not be completely random however, and more research needs to be done to investigate the 'free' movement of one of the control surfaces.

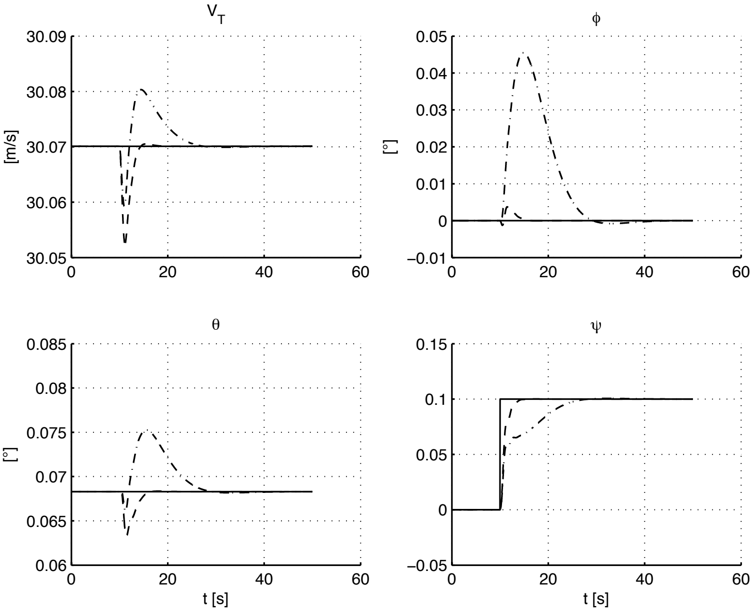


Figure 5.12: Tracked outputs (solid:reference, dashed: without failure, dash-dotted: with failure)

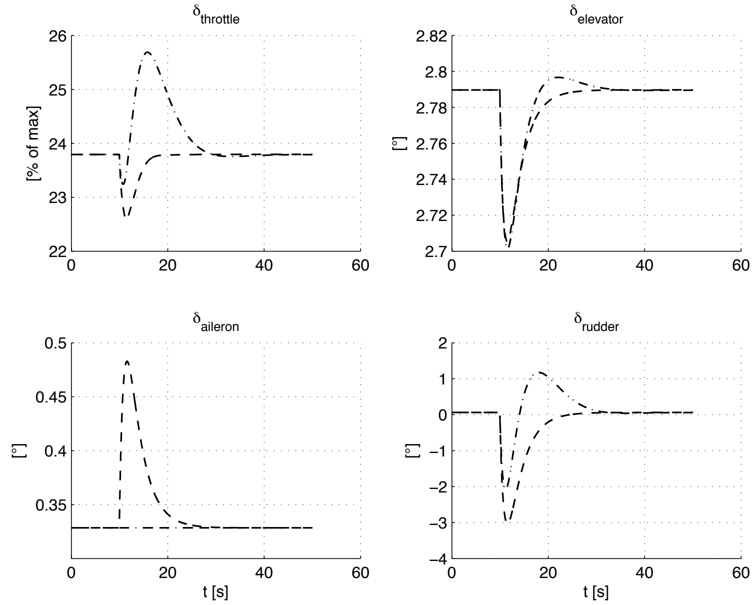


Figure 5.13: System inputs (dashed: without failure, dash-dotted: with failure)

Chapter 6

Conclusions and recommendations

6.1 Conclusions

In this report a continuous time model predictive controller is built, which uses orthonormal functions to describe the control signal. A multi-input-multi-output linear time invariant state space system is derived from a nonlinear MATLAB Simulink model of an unmanned aerial vehicle. Constraints are applied to the controls and it is shown that the model predictive controller is also capable of fault tolerant control within certain limits.

Control of a UAV by means of a model predictive controller works well. It is shown that by using the MPC discussed in this report, it is possible to track a given reference signal if no constraints are active. The continuous time MPC which uses orthonormal functions to describe the control signal has not been used before to control a complex 4-input-4-output system like the aircraft model presented here.

Adding constraints on the controls, inputs and outputs of the aircraft model is easy. Simulations show that the application of 'hard' constraints works very well. The controller will never override the applied constraints. It should be noted that when the constraints are active, the aircraft model is no longer able to follow any given reference signal. There are limits to what the aircraft model is still capable of. If the constraints are hit for a long period of time, the controller tries to reach its objective by changing the other controls. For example, by setting a big step on the velocity, the controller will not just raise the throttle to its maximum and wait patiently for the aircraft model to reach the desired velocity, but it will pitch the aircraft in a downwards motion to speed up the acceleration process. This is a logical result from the optimization algorithm, but in general this behavior is not desired.

The work presented in this report shows that the continuous time MPC using orthonormal functions to describe the control signal, gives good performance when applied to the linear time invariant state space system derived from operating at the steady wings-level flight trim condition. To use this controller for the entire flight scope of the aircraft, a MPC capable of handling time-varying state space systems needs to be designed to handle the trim conditions which result in time-varying state space systems.

6.2 Recommendations

To obtain a good trajectory following solution, a second (MPC) controller can be wrapped around the one presented here. This second controller can translate for instance a given set of way-points, into an optimized North-East-Down and velocity vector, varying over time. This trajectory must then be translated into the tracked output variables of the system. The here presented MPC will assure that the aircraft model will smoothly follow the given trajectory and will also make the UAV robust to failure of any of the aircraft controls, within certain limits.

Another possible refinement is to make a switching controller that switches between different linearized models for different operating conditions. The linearized models are typically not valid for the whole mission scope, and switching between state space models is necessary to assure good and reliable performance. It should be noted that a switching controller can cause instability of the whole system. Therefore care should be taken when implementing such a controller to assure stability of the aircraft model. A switching controller can be implemented within the MATLAB Simulink environment.

Finally, to make trajectory following easier to implement, the Simulink Aerospace toolbox can be used to translate between a reference trajectory of North-East-Down coordinates and the Euler angles to the selected reference model outputs. It is also possible to link this to the FlightGear flight simulation software to make a visual presentation of the simulation.

Appendix A

Time dependent A matrix

($c = cos$, $s = sin$, $tn = tan$)

$$A(t) = \begin{bmatrix} 0 & R & -Q & 0 & -9.81c\theta & 0 & 0 & -W & V & 0 & 0 & 0 \\ -R & 0 & P & 9.81c\phi c\theta & -9.81s\phi s\theta & 0 & W & 0 & -U & 0 & 0 & 0 \\ Q & -P & 0 & -9.81s\phi c\theta & -9.81c\phi s\theta & 0 & -V & U & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & p_1 & 0 & 1 & tn\theta s\theta & -tn\theta c\theta & 0 & 0 & 0 \\ 0 & 0 & 0 & -Rc\phi & -Qs\theta & 0 & 0 & c\theta & -s\phi & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{Qc\phi - Rs\phi}{c\theta} & \frac{Qs\phi + Rc\phi}{c\theta^2 s\theta} & 0 & 0 & \frac{s\phi}{c\theta} & \frac{c\phi}{c\theta} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & c_2Q & c_1R + c_2P & c_1Q & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & c_5R - 2c_6P & 0 & c_5P + 2c_6R & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & c_8Q & c_8P - c_2R & -c_2Q & 0 & 0 & 0 \\ c\theta c\psi & p_{10} & p_{11} & p_2 & p_3 & p_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ c\theta s\psi & p_{12} & p_{13} & p_5 & p_6 & p_7 & 0 & 0 & 0 & 0 & 0 & 0 \\ -s\theta & s\phi c\theta & c\phi c\theta & p_8 & p_9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{A.1}$$

with:

$$\begin{aligned} p_1 &= (1 + tn\theta^2)Qs\theta - Rc\theta + t\theta Qc\theta + Rs\theta \\ p_2 &= V(c\phi s\theta c\psi + s\phi s\psi) + W(c\phi s\theta c\psi + c\phi s\psi) \\ p_3 &= -Us\theta c\psi + Vs\phi c\theta c\psi) + Ws\phi c\theta c\psi \\ p_4 &= -Uc\theta s\psi + V(-s\phi s\theta s\psi - c\phi c\psi) + W(-s\phi s\theta s\psi + s\phi c\psi) \\ p_5 &= Vc\phi s\theta s\psi + W(-s\phi s\theta s\psi - c\phi c\psi) \\ p_6 &= -Us\theta s\psi + V(s\phi c\theta s\psi - s\theta c\psi) + Wc\phi c\theta s\psi \\ p_7 &= Uc\theta c\psi + V(s\phi s\theta c\psi - c\theta s\psi) + W(c\phi s\theta c\psi + s\phi s\psi) \\ p_8 &= Vc\phi c\theta - Ws\phi c\theta \\ p_9 &= -Uc\theta - Vs\phi s\theta - Wc\phi s\theta \\ p_{10} &= s\phi s\theta c\psi - c\phi s\psi \\ p_{11} &= s\phi s\theta c\psi + s\phi s\psi \\ p_{12} &= s\phi s\theta s\psi + c\theta c\psi \\ p_{13} &= c\phi s\theta s\psi - s\phi c\psi \end{aligned}$$

Appendix B

LTI Ariel model

The time invariant Ariel model linearized at trim condition

$$x_0 = \begin{bmatrix} U \\ V \\ W \\ \phi \\ \theta \\ \psi \\ P \\ Q \\ R \\ p_N \\ p_E \\ p_D \end{bmatrix} = \begin{bmatrix} 30 \\ 0.012 \\ 2.052 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0.068 \\ 0 \\ 0 \\ 0 \\ -1000 \end{bmatrix} \quad y_0 = \begin{bmatrix} n_x \\ n_y \\ n_z \\ V_T \\ \alpha \\ \beta \\ P \\ Q \\ R \\ \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} 0.068 \\ 0 \\ 0.998 \\ 30.07 \\ 0.068 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0.068 \\ 0 \end{bmatrix} \quad u_0 = \begin{bmatrix} \delta_T \\ \delta_e \\ \delta_a \\ \delta_r \\ \delta_f \end{bmatrix} = \begin{bmatrix} 0.238 \\ 2.790 \\ 0.328 \\ 0.061 \\ 1 \end{bmatrix}$$

$$A_m = \begin{bmatrix} -0.075 & -0.001 & 0.408 & 0 & -2.056 & 0.012 & 0 & -9.787 & 0 & 0 & 0 & 0 \\ 0.000 & -0.529 & 0 & 2.478 & 0 & -29.208 & 9.787 & 0 & 0 & 0 & 0 & 0 \\ -0.431 & -0.021 & -3.243 & -0.014 & 28.637 & -0.001 & 0 & -0.669 & 0 & 0 & 0 & -0.001 \\ -0.030 & -0.156 & 0.018 & -7.681 & -0.089 & 0.615 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.037 & 0.019 & -0.161 & 0.001 & -2.967 & -0.525 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.004 & 0.504 & 0.006 & -2.001 & -0.349 & -1.702 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0.068 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.002 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.998 & 0 & 0.068 & 0 & 0 & 0 & 0.001 & 0 & -0.012 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & -2.052 & 0 & 30.07 & 0 & 0 & 0 \\ -0.068 & 0 & 0.998 & 0 & 0 & 0 & 0.012 & -30.07 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$B_m = \begin{bmatrix} 3.651 & 0.012 & 0 & 0 & 0.009 \\ 0 & 0 & -0.003 & 0.119 & 0 \\ 0.011 & -0.215 & 0 & 0 & -2.890 \\ 1.146 & 0.002 & -1.183 & 0.033 & 0 \\ -1.032 & -0.473 & 0 & 0 & 0.640 \\ 0.128 & 0 & -0.126 & -0.299 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$C_m = \begin{bmatrix} -0.0080 & 0 & 0.042 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -0.054 & 0 & 0.043 & 0 & 0.081 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.044 & 0.002 & 0.331 & 0 & 0.139 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.997 & 0 & 0.068 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.002 & 0 & 0.033 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.033 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

$$D_m = \begin{bmatrix} 0.372 & 0.001 & 0 & 0 & 0.001 \\ 0 & 0 & 0 & 0.012 & 0 \\ -0.001 & 0.0223 & 0 & 0 & 0.295 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Appendix C

Controls and states scenario 1 and 2

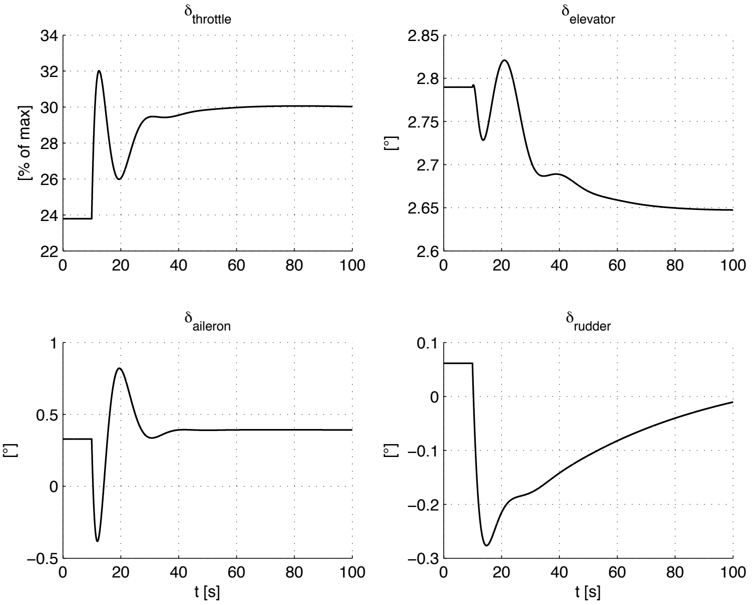


Figure C.1: Control efforts for outputs n_x, n_y, n_z and ψ

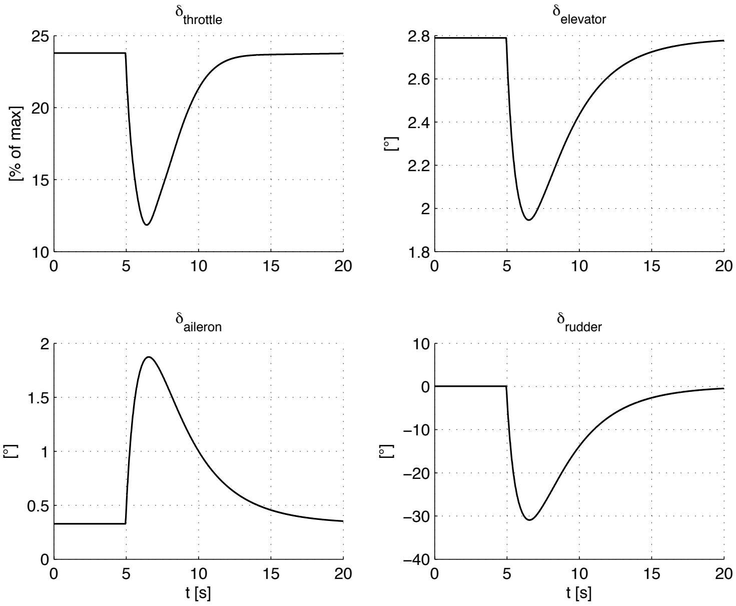


Figure C.2: Control efforts for outputs V_T, ϕ, θ and ψ

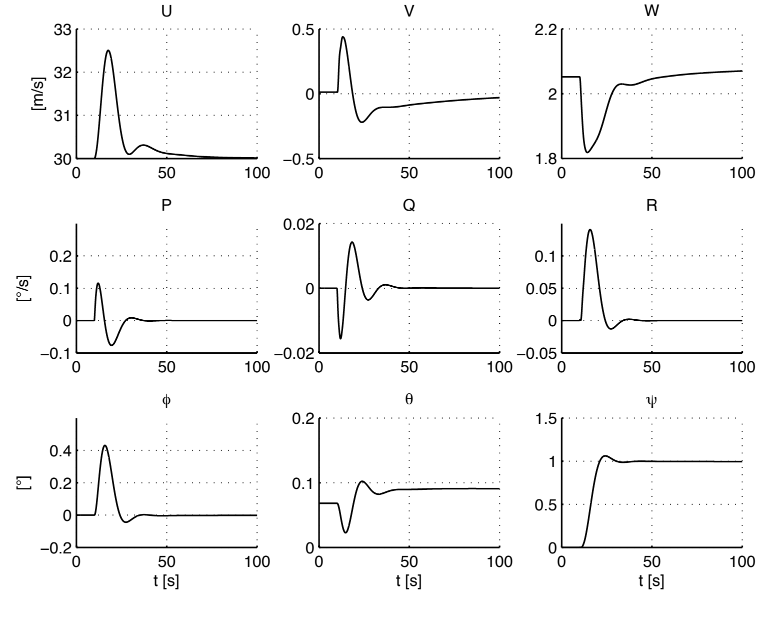


Figure C.3: States for outputs n_x, n_y, n_z and ψ

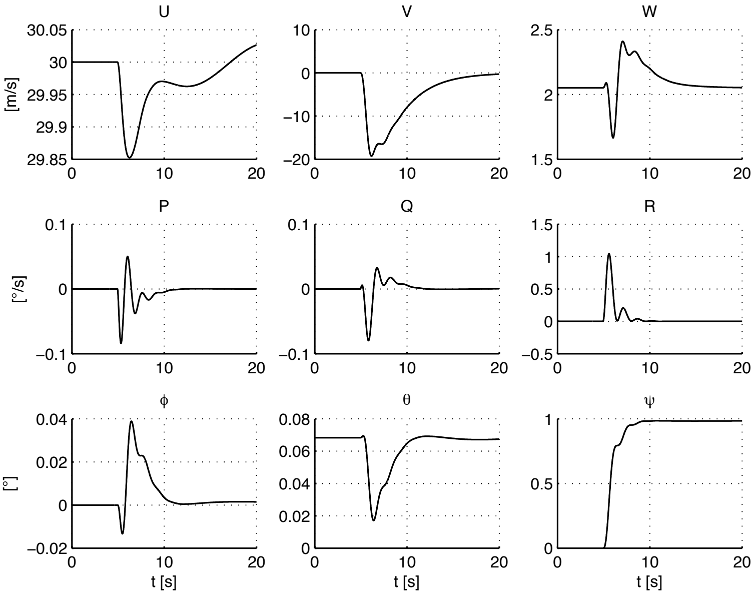
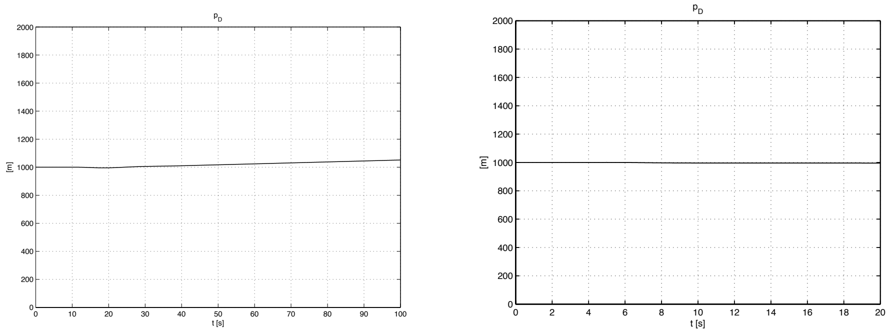


Figure C.4: States for outputs V_T, ϕ, θ and ψ



(a) scenario 1

(b) scenario 2

Figure C.5: State p_D

Appendix D

Trim conditions

Operating point		outputs				controls			
alt $[m]$	vel $[m/s]$	V_T $[m/s]$	ϕ $[rad]$	θ $[rad]$	ψ $[rad]$	δ_t [%]	δ_e $[rad]$	δ_a $[rad]$	δ_r $[rad]$
100	30	30.05	0	0.06	0	23	2.90	0.31	0.05
1000	30	30.07	0	0.07	0	24	2.79	0.33	0.06
5000	30	30.24	0	0.13	0	32	1.84	0.47	0.11
100	55	54.51	0	0	0	100	0	0	0
1000	55	54.55	0	0	0	100	3.36	0.24	0.01
5000	55	54.68	0	0.01	0	100	3.33	0.24	0.02

Table D.1: Operating conditions

Bibliography

[1] L. Wang
Continuous time model predictive control design using orthonormal functions
International Journal of Control, Vol. 74(16), 1588-1600, 2001.

[2] M. R. Crump
The dynamics and control of catapult launching unmanned air vehicles from moving platforms
Technical report, Royal Melbourne Institute of Technology, 2002.

[3] D. H. Shim, H. J. Kim and S. Sastry
Decentralized Nonlinear Model Predictive Control of Multiple Flying Robots in Dynamic Environments
IEEE Conference on Decision and Control, December 2003.

[4] R. Howard and I. Kaminer
Survey of unmanned vehicles
In Proceedings of the American Control Conference Vol. 5: 2950-2953, 1995.

[5] R. Moore
Unmanned air vehicles - a prospectus
Aerospace America, Vol. 27: 26-28, February, 1989.

[6] K. Wong
Unmanned aerial vehicles (UAVs) - are they ready this time? are we?
Technical report, RAES Sydney Branch, 1997.

[7] K. Wong and C. Bil
Uav's over Australia- market and capabilities
In Bristol RPV/UAV Systems Conference, 1998.

[8] L. Thompson, S. Abanteriba, and C. Bil
A multipurpose autonomous flight vehicle system
Technical report, RMIT Wackett Centre, 1995.

[9] D. L. Juul, M. McDermott, E. L. Nelson, D. M. Barnett, and G. N. Williams
Submersible control using the linear quadratic gaussian with loop transfer recovery method
In IEEE Sympsium on Autonomous Underwater Vehicle Technology, pp. 417-425, 1994.

[10] D. Newman and K. Wong
Six Degree of Freedom Flight Dynamic and Performance Simulation of a Remotely-Piloted Vehicle
Technical report, The University of Sydney, 1993.

[11] Warren F. Phillips
Mechanics of flight
Hoboken, New York: Wiley, 2004.

[12] C. E. García, D. M. Prett, M. Morari
Model predictive control: Theory and practice–a survey
Automatica, Vol. 25(3): 335-348, 1989.

[13] N. L. Ricker
Model predictive control: State of the art
In Y. Arkun W. H. Ray (eds), Chemical Process Control–CPC IV, Fourth International Conference on Chemical Process Control, Elsevier, Amsterdam, pp. 271-296, 1991.

[14] M. Morari, J. H. Lee
Model predictive control: The good, the bad, and the ugly
In Y. Arkun W. H. Ray (eds), Chemical Process Control–CPC IV, Fourth International Conference on Chemical Process Control, Elsevier, Amsterdam, pp. 419-444, 1991.

[15] K. R. Muske, J. B. Rawlings
Model predictive control with linear models
AIChE J. Vol. 39(2): 262-287, 1993.

[16] J. B. Rawlings, E. S. Meadows, K. R. Muske
Nonlinear model predictive control: A tutorial and survey
ADCHEM '94 Proceedings, Kyoto, Japan, 1994.

[17] M. J. Maciejowski
Predictive Control: with constraints
Pearson Education Limited, England: Harlow, 2001.

[18] H. Demircioglu, D. W. Clarke
CGPC with guaranteed stability properties
IEEE Proceedings, Pt. D. Vol. 139, 371-380, 1992.

[19] H. Demircioglu, P. J. Gawthrop
Continuous time generalized predictive control
Automatica. Vol. 27, 55-74, 1991.

[20] H. Demircioglu, P. J. Gawthrop
Multivariable continuous time generalized predictive control
Automatica. Vol. 28, 697-713, 1992

[21] P. J. Gawthrop
Linear predictive pole-placement control: practical issues
39th IEEE Cenference on Decision and Control. Sydney, Australia, pp. 160-165, 2000.

[22] Y. W. Lee
Statistical Theory of Communication
New York: Wiley, 1960.

[23] C. R. Wylie
Advanced Engineering Mathematics
New York: McGraw-Hill, 1960.

[24] T. Kailath
Linear systems
Englewood Cliffs, New Jersey: Prentice-Hall, 1980.

[25] G.C. Goodwin, S. Graebe, M. Salgado
Control System Design
Englewood Cliffs, New Jersey: Prentice-Hall, 2000.

[26] L. Wang
Model Predictive Control System Design and Implementation using MATLAB
In preparation. Publisher: Springer-Verlag, 2007.