B.A.Sc THESIS

Investigating the Control Approaches for eVTOLs and Designing a Non-linear Dynamic Inversion Controller for a Selected eVTOL Model

by

Elif Celik

Supervisor: Hugh H.T. Liu

April 2023

Division of Engineering Science
UNIVERSITY OF TORONTO

Abstract

This paper investigates the control approaches for Electric Vertical Take-off and Landing Aircraft (eVTOL), which is an aircraft type that uses electric power to take-off and land vertically. The design of such an aircraft presents various challenges, one of which is the choice of control systems that would achieve the required level of stability and safety. The trajectory of these airplanes can be highly non-linear, making it one of the main difficulties associated with the control systems design of eVTOLs. Although many strategies have been proposed for various eVTOL models, the selection of the appropriate control method still remains to be a topic of research, as a standard for the connection between the control systems and the models has not been established yet. In order to develop a modelling and control platform to incorporate eVTOL dynamics with a proper control design, this paper presents a detailed literature review, outlining the state-of-the-art control approaches for eVTOLs. Additionally, it presents a Non-linear Dynamic Inversion controller design for a selected eVTOL model. This controller design is implemented in Simulink and the performance of this control approach is evaluated using settling time and maximum overshoot as the criteria. It was shown that this control strategy provided reasonable results for the eVTOL configuration chosen. Overall, this paper acts a resource to engineers working on eVTOL designs, and provides a reliable database to investigate control systems.

Acknowledgements

This thesis research would not have been successfully completed without the supervision of my professor Hugh H. T. Liu from the Flight Systems and Control Lab at the Institute for Aerospace Studies University of Toronto. I would like to thank him for his knowledge and the advice he has given me throughout the school year. I would also like to thank the University of Toronto Engineering Science Division for providing the opportunity to work on a thesis project in my final year of undergraduate studies.

Contents

List of Symbols & Abbreviations

VTOL	Vertical Takeoff and Landing Vehicle
eVTOL	Electric Vertical Takeoff and Landing Vehicle
NDI	Non-linear Dynamic Inversion
INDI	Incremental Non-linear Dynamic Inversion
PID	Proportional, Integral & Derivative
PD	Proportional & Integral
g	Gravity
m	Mass of eVTOL
X	X Position
Y	Y Position
Z	Z Position
R	Rotation Matrix
ω	Speed of Each Rotor
K_p	Proportional Gain
K_i	Integral Gain
K_d	Derivative Gain
θ	Euler Angle - Pitch
ϕ	Euler Angle - Roll
ψ	Euler Angle - Yaw
$\vec{\Omega}$	Angular Velocity Vector in Body Frame
$\dot{\vec{\Omega}}$	Angular Acceleration Vector in Body Frame
p	Angular Rotation About x-axis

q	Angular Rotation About y-axis
r	Angular Rotation About z-axis
$\dot{p}$	Angular Rotation About x-axis
$\dot{q}$	Angular Rotation About y-axis
$\dot{r}$	Angular Rotation About z-axis
ρ	Density
k_T	Thrust Coefficient in Vertical Flight
c_τ	Rotor Torque Coefficient
k_d	Drag Coefficient in Vertical Flight
$\vec{M}$	Moments of eVTOL
$\vec{F}$	Forces of eVTOL
T	Thrust
α	Wing to Body Angle
L_{x1}	Length From y-axis to Closest Propeller, Parallel to x-axis
L_{x2}	Length From y-axis to Farthest Propeller, Parallel to x-axis
L_y	Length From x-axis to Propellers, Parallel to y-axis
ω_n	Natural Frequency
ξ	Damping Ratio
t_s	Settling Time
I_{xx}	Moment of Inertia Around x
I_{yy}	Moment of Inertia Around y
I_{zz}	Moment of Inertia Around z

List of Figures

List of Tables

1 Introduction

1.1 Background

This thesis research investigates various control approaches for an electric vertical take-off and landing aircraft (eVTOL), with the goal of designing a controller for a selected configuration.

eVTOLs use distributed electric power to take off, land and hover. Many different configurations of eVTOLs have been built and tested in the last decade. However, the control systems design of this electric vehicle has been a challenging task, especially due to the transition trajectory between hover and cruise being highly non-linear [1]. Furthermore, the suitable choice of a controller can vary based on the type of eVTOL model. Most commonly, the eVTOL types are classified as follows:

1. **Vectored Thrust**: These eVTOLs use the same propulsion system for both hover and cruise and can be in the form of a tilt-rotor or a tilt-wing [2]. Tilt-rotors tilt their rotors to transition between hover and forward flight, whereas the tilt-wings rotate their wings to rotate the whole wing since the motor is incorporated into the wings. The controller design for the vectored thrust configuration can be quite complex since the same propulsion system is used for both the forward and upward motion. However, this type of aircraft are quite efficient in the power consumption of the cruise phase [3].

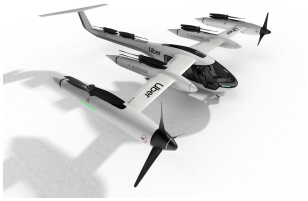
2. **Lift + Cruise**: These vehicles use separate propulsion systems for hover and cruise unlike the vectored thrust vehicles, and have a reduced design complexity as a result [2]. However, the cruise efficiency is lower compared to that of vectored thrust vehicles due to the introduction of additional drag by the rotors used in hovering [3].

3. **Wingless**: These are wingless multi-rotors that have a disk actuator surface, which makes them efficient in hover but less efficient in cruise [2].

(a) Lilium tilt-wing eVTOL

(b) Joby tilt-rotor eVTOL

Figure 1: Examples of Vectored Thrust eVTOLs [4], [5]



(a) Uber's eVTOL

(b) Voloctopter's eVTOL

Figure 2: Examples of Lift + Cruise (a) & Wingless (b) eVTOLs [6], [7]

All aircraft must have a control system that enables them to fly according to the desired flight trajectory and maintain stability in the face of unexpected perturbations. For a conventional aircraft, the control surfaces include ailerons, elevators, rudders. Due to the absence of a certified eVTOL, there does not exist a so-called "conventional" eVTOL at the moment. Although there are different types of eVTOLs, the design for each model can look highly different. Thus, there does not exist a conventional control approach for eVTOLs either. Various approaches to handle the control of eVTOLs have been suggested for these different configurations, such as gain scheduling, proportional integral derivative controllers (PID), linear quadratic regular controllers, non-linear dynamic inversion and more.

1.2 Goals & Objectives

While various control approaches have already been suggested, the eVTOL field is still in need of more techniques that could improve the performance of these planes. The objective of this thesis is to contribute to the development of eVTOLs through the creation of a modelling and control platform for an eVTOL configuration. This platform would provide useful information and act as a resource to other engineers who may desire to work on the same aircraft type, or build on the work presented in accordance to the specific requirements of their research/project.

In order to meet the goals of the thesis, a comprehensive state-of-the art literature review has been gathered which is included in Section 2 of this report. After analyses of various control techniques and eVTOLs, the model provided in the "eVTOL Design, Control System" report by Joris Holshijsen and Joost van der Weerd from the Delft University of Technology was chosen to be the configuration of interest. This is a separate lift + cruise eVTOL with fixed - wings and fixed - rotors, and it has 4 propellers on each wing. This eVTOL design also has ailerons to control the vehicle in cruise flight. [8]. A 3-D render of this design is shown in Figure 3. The proposed controller in the Delft report consists of 4 PID controllers for the vertical motion of the aircraft. However, aircraft are inherently non-linear systems,

meaning that a non-linear controller would most likely another useful and reliable alternative. Therefore, the goal of this thesis is to implement the Non-linear Dynamic Inversion control technique on the separate lift + cruise Delft model and conduct performance analysis.

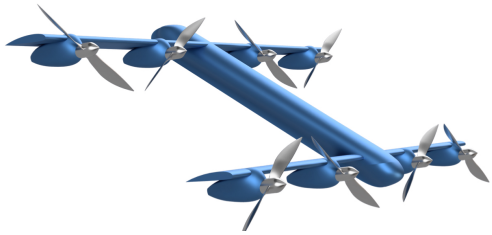


Figure 3: Delft Model - Selected Configuration [8]

In order to successfully implement the Non-linear Dynamic Inversion method to the selected separate lift + cruise aircraft, a series of steps will be taken. First, a comprehensive literature review is documented, which outlines the state-of-the art research for control approaches for eVTOLs. Next, a dynamic model of the selected eVTOL configuration is created in Simulink. This step is crucial since the plant dynamics is needed for the controller design. Next, the Non-linear Dynamic Inversion method is implemented for this dynamic model. Lastly, performance analysis is conducted and the results are compared to the ones presented in the Delft report.

2 Literature Review

2.1 Introduction

This literature review presents the current state-of-the art research for the control approaches for eVTOLs. In most cases, the control methods mentioned earlier can be used in conjunction with each other. This literature review describes these various control techniques in detail, and presents the reasonings for why the implementation of a Non-linear Dynamic Inversion controller would provide satisfactory results for the selected configuration.

2.2 Investigation of Control Approaches

2.2.1 Basics of Control Systems

The control system of an aircraft must ensure stability in all 6 degrees of freedom throughout all stages of flight. The translational degrees of freedom include moving along the x-axis (surging), the y-axis (heaving) and the z-axis (swaying). The rotational degrees of freedom include the roll, pitch and yaw directions. A simple control model includes pilot commands and/or cockpit controls, sensors, actuators and computers. A general control block for an aircraft is shown in Figure 4. The reference inputs from the pilot is fed into the loop, and the error is calculated using sensor measurements. Using the controller and the actuator, the orientation of the vehicle can be corrected to the desired trajectory.

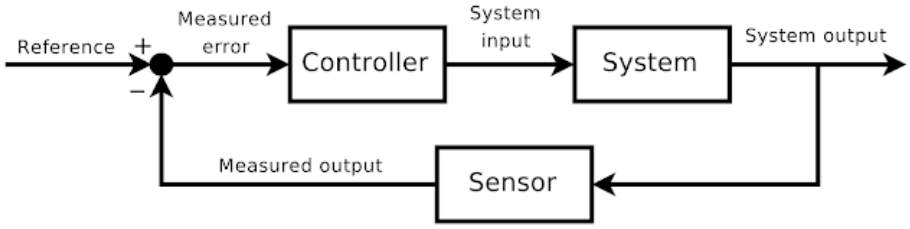


Figure 4: Simple Control Architecture for an Aircraft [9]

2.2.2 Control Approach 1: Gain Scheduling

Gain scheduling is a popular approach for control design that is used in many fields, including aerospace. It is commonly used for non-linear systems with varying dynamics. It is implemented when the controller gains are not sufficient enough to maintain the desired performance and stability. The idea behind the approach of gain scheduling is to "divide and conquer", where the non-linear system is decomposed into a set of linear models. Then, the non-linear model is converted to a Linear Parameter Varying (LPV) model, and each

linear model can be controlled as a function of scheduling parameters [10]. The selection of scheduling variables is important, since they reflect any changes to system dynamics [11]. Scheduling variables ensure that the controller gains are adjusted to the desired states. Figure 5 shows the addition of a gain scheduling variable to a simple control architecture.

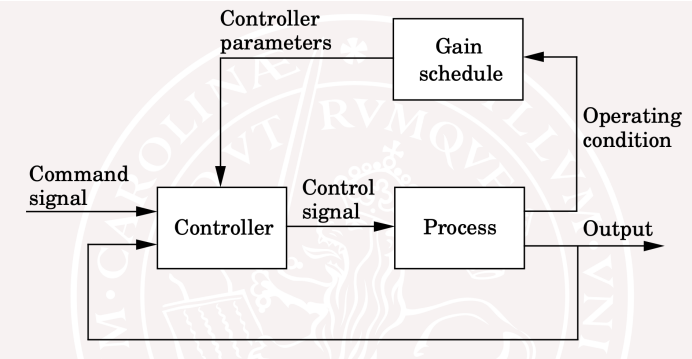


Figure 5: Control Architecture with Gain Scheduling [11]

Gain scheduling provides satisfactory performance when the scheduling variables are external parameters that do not vary fast, like the speed of a cruising airplane. This is because gain scheduling does not perform as well with scheduling variables that depend on fast-varying states of a system [12]. The use of gain scheduling with different types of controllers will be discussed in the following sections.

2.2.3 Control Approach 2: Proportional Integral Derivative (PID) Controller

As implied by its name, this type of controller uses proportional control combined with integral and derivative adjustments to compensate for the changes in the system. The implementation of PID controllers is relatively easy due to the tuning of three parameters only. This type of design approach can be used for a set of configurations by applying state errors to a PID controller. However, since PID controllers are linear controllers with rather straightforward implementation, they may not be the most suitable control approach (on their own) for the vectored thrust configuration where the system dynamics are decoupled, especially during the transition phase between hover and forward flight. However, for a separate lift + cruise eVTOL where the hover and cruise use different propulsion systems, PID controllers can be used due to their reduced complexity.

To understand how PID controllers can be used in the context of eVTOLs, the PID controller design proposed by the University of Delft for the selected eVTOL model is outlined below. This general method can be applied to various eVTOLs with different dynamics as well.

2.2.3.1 Vertical Flight: First, the linear forces and moments acting on the aircraft during vertical flight can be determined. The University of Delft report considers the gravitational force, the drag force, and the thrust force due to the motors of the eVTOL. The moments include the moments due to the thrust forces and the propellers. For simplification, it can be assumed that $\phi \approx 0°$, $\theta \approx 50°$, $\psi \approx 0°$, which represent the roll, pitch and yaw angles respectively. Using these forces and moments, the angular acceleration of the eVTOL in the vertical direction can be represented as follows:

$$\ddot{\phi} = \frac{u_1}{I_{xx}} + \frac{u_3 \tan(50°)}{I_{zz}} \tag{1}$$

$$\ddot{\theta} = \frac{u_2}{I_{yy}} \tag{2}$$

$$\ddot{\psi} = \frac{u_3}{I_{zz}\cos(50°)} \tag{3}$$

where u_1, u_2 and u_3 represent the virtual control inputs that will be used by the PID controllers, and I_{xx}, I_{yy} and I_{zz} are the moments of inertia around x, y and z respectively. Equations 1, 2 and 3 were developed under the assumption that the roll, pitch and yaw angles are approximately constant, and this linearization suggests that the angular velocities are zero.

As for position control, the total forces acting on the aircraft are divided by the mass to obtain the linear accelerations as follows:

$$\begin{bmatrix} \ddot{X} \\ \ddot{Y} \\ \ddot{Z} \end{bmatrix} = \frac{1}{m} R^{-}1 \begin{bmatrix} 2k_T \cos(\alpha) z(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2) \\ 0 \\ 2k_T \sin(\alpha)(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \frac{1}{m} \begin{bmatrix} -k_d \dot{X} \\ -k_d \dot{Y} \\ -k_d \dot{Z} \end{bmatrix}, \tag{4}$$

where k_T represents the thrust coefficient in vertical flight, α is the wing to body angle, k_d is the drag coefficient in vertical flight, R is the rotation matrix to translate between inertial and body frames, and ω are the spin velocities of the propellers. By setting $u_4 = 2k_T(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2)$ (fourth virtual control input), the altitude acceleration can be simplified:

$$\ddot{Z} = \frac{(-\sin(\theta)\cos(\alpha) + \sin(\alpha)\cos(\theta))u_4 - k_d \dot{Z}}{m} - g \tag{5}$$

Using the formulas obtained for attitude and position control, the accelerations can be replaced by the formula for a PID controller and the following relationship can be obtained for the virtual control inputs:

$$u_1 = I_{xx}(-\frac{u_3 \tan(50°)}{I_{zz}} + K_{p,\phi} e_\phi + K_{d,\phi}\dot{\phi} + K_{i,\phi}\int e_\phi) \tag{6}$$

$$u_2 = I_{yy}(K_{p,\theta} e_\theta + K_{d,\theta}\dot{\theta} + K_{i,\theta}\int e_\theta) \tag{7}$$

$$u_3 = I_{zz}(K_{p,\psi} e_\psi + K_{d,\psi}\dot{\psi} + K_{i,\psi}\int e_\psi) \tag{8}$$

$$u_4 = \frac{m(g + K_{p,z} e_z + K_{d,z}\dot{Z} + K_{i,z}\int e_z)}{\sin(\alpha)\cos(50°) - \sin(50°)\cos(\alpha)}, \tag{9}$$

where K_p, K_d, K_i, are the proportional, differential, and integral control constants, and $e_{\phi,\theta,\psi,z}$ is the difference between the desired attitude/altitude and the current value [8].

2.2.3.2 Horizontal Flight: Horizontal flight includes two additional forces to vertical flight which are the lift and drag forces generated by the wings and the rotors. Similarly, the moments of interest would also include the moments due to these lift and drag forces.

The control equations for attitude and position can be derived in a similar fashion to vertical flight, and they can be designed such that control formulas including the PID controllers for u_1, u_2, u_3 same as the ones derived for vertical flight. For the altitude control however, the control input formula using the assumption $\phi \approx 0°$, $\theta \approx -40°$, $\psi \approx 0°$ is shown below [8]:

$$u_4 = \frac{m(g + K_{pz} e_z + K_{dz}\dot{Z} + K_{iz}\int e_z)}{\sin(\alpha)\cos(\theta - 40°) - \sin(-40°)\cos(\alpha)}, \tag{10}$$

2.2.3.3 Implementation of PID Controllers: The attitude and altitude control approaches derived above should be implemented. A controller consisting of two parts can be designed with the first part being the the four individual PID controllers and the second part being a function that calculates the rotor speeds to maintain the correct orientation. Figure 6 shows a sample implementation of the entire system for an eVTOL with PID controllers, and Figure 7 shows a sample implementation of a PID controller for u_1 [8].

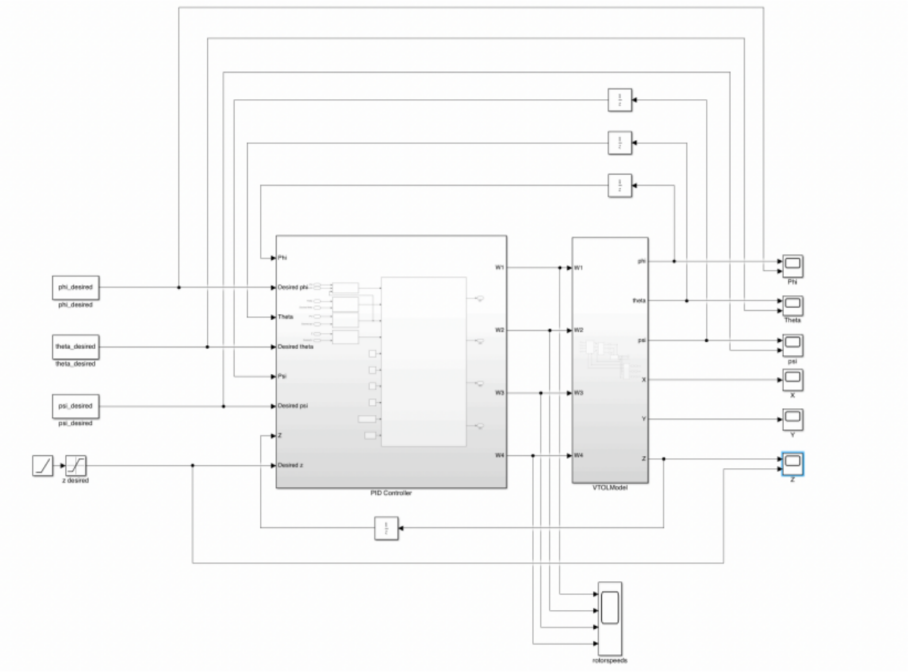


Figure 6: Overview of the Control System with PID Controllers [8]

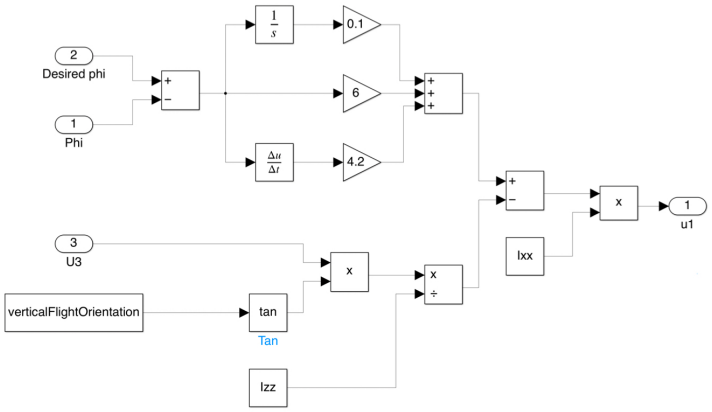


Figure 7: PID Controller Implementation [8]

2.2.4 Control Approach 3: Linear Quadratic Regulator (LQR) Controller

This method provides optimally controlled feedback gains via the solution of the Ricatti equation to allow for the closed-loop to be stable, and thus provide good performance. By using two matrices called Q and R that relate the cost of the states and the inputs, an optimal linear feedback matrix can be calculated [13]. LQR controllers are very robust, however they are usually limited to systems with linear dynamics, and the adjustment of Q and R matrices may present difficulties for the tuning of the controller.

The transition trajectory of an aircraft refers to the transition corridor between powered-lift flight to a wing-borne flight, that is composed of equilibrium and quasi equilibrium points related to certain wing-tilt angle and airspeed combinations. The trim points of this transition trajectory can be defined by $X_k = (\mathbf{x}_k^*, \mathbf{u}_k^*)$ where k = 1...N, and the nominal airspeed at the k^{th} trim location can be defined by V_k. A gain scheduling variable $\sigma(t)$ can be defined as a function of the nominal airspeed V_k and current speed of the eVTOL shown by $V(t)$ as follows:

$$\sigma(t) = \frac{V(t) - V_k}{V_{k+1} - V_k}, \; V_{k+1} > V(t) > V_k. \tag{11}$$

The control input vector $\mathbf{u}(t)$ can then be scheduled using the scheduling parameter $\sigma(t)$ as follows:

$$\mathbf{u}(t) = \begin{cases} \mathbf{u}_1(t), & V(t) \leq V_1 \\ (1-\sigma(t))\mathbf{u}_k(t) + \sigma(t)\mathbf{u}_{k+1}(t) & V_{k+1} > V_k \geq V_k \\ \mathbf{u}_N(t), & V(t) \geq V_k. \end{cases} \tag{12}$$

Then, at a given time t, the control input signal can be obtained by:

$$\mathbf{u}_k(t) = \mathbf{u}_k^*(t) + \Delta\mathbf{u}_k(t), \tag{13}$$

where $\Delta\mathbf{u}_k(t)$ is the LQR feedback control vector, and $\mathbf{u}_k^*(t)$ is the virtual trim control input. Furthermore, the non-linear dynamics of the chosen configuration system can be linearized, resulting in the following form:

$$\dot{\mathbf{e}}(t) = \mathbf{A}_k\mathbf{e}(t) + \mathbf{B}_k\Delta\mathbf{u}_k(t), \tag{14}$$

where $\mathbf{e}(t) = \mathbf{x}(t) - \mathbf{x}_k^*(t)$. Finally, for each linear system, the optimal feedback gain can be calculated as follows:

$$\Delta\mathbf{u}_k(t) = -\mathbf{K}_k\mathbf{e}(t) = -\mathbf{R}^-1_k\mathbf{B}_k^T\mathbf{P}_k(e)(t) \tag{15}$$

where $\mathbf{P}_k$ is the solution to the Riccati equation, and $\mathbf{Q}_k$ and $\mathbf{R}_k$ are the state error and control input cost matrices respectively [14]. The Ricatti equation is as follows:

$$\mathbf{0} = \mathbf{A}_k\mathbf{P}_k + \mathbf{P}_k\mathbf{A}_k^T - \mathbf{P}_k\mathbf{B}_k\mathbf{R}^-1_k\mathbf{B}^\mathbf{T}{}_k\mathbf{P}_k + \mathbf{Q}_k \tag{16}$$

2.2.5 Control Approach 4: Combination of Gain-Scheduled LQR and PID Controllers

It is possible to combine the LQR and PID approaches to enhance the stability of an aircraft. The combination of these two approaches serves as a more robust and stable approach for complex models like vectored thrust vehicles than using these methods separately. In order to successfully navigate through the transition trajectory, all controllers can be gain-scheduled. To integrate gain scheduling into the controller design, the gain scheduling parameter can be defined by the aircraft's wing incident angle. The steady state can be defined by the nominal wing incident angle γ, and the controller's function in accordance to the flight regime can be defined by γ. This suggests that γ should not change rapidly as this would cause the assumed flight regime to deviate from the actual flight regime.

A LQR controller has the ability the adjust the vehicle's orientation through the transition corridor, while the outer loops can be controlled by PID loops. The system's orientation can be be expressed as a linear state-space model based on the vehicle's steady state, where the model uses Euler angles and rotation rates to describe the orientation of the aircraft. The deviations of the eVTOL from the steady state during the transition corridor defined by $[M_1 M_2 M_3 M_4 W_1 W_2 W_3 W_4]$ can be inputted into this system as follows:

$$\dot{\mathbf{x}} = Ax + Bu, \tag{17}$$

where,

$$A = \begin{bmatrix} \frac{d\dot{\phi}}{dw_\phi} & \frac{d\dot{\phi}}{d\phi} & \frac{d\dot{\phi}}{dw_\theta} & \frac{d\dot{\phi}}{d\theta} & \frac{d\dot{\phi}}{dw_\psi} & \frac{d\dot{\phi}}{d\psi} \\ 1 & 0 & 0 & 0 & 0 & 0 \\ \frac{d\dot{\theta}}{dw_\phi} & \frac{d\dot{\theta}}{d\phi} & \frac{d\dot{\theta}}{dw_\theta} & \frac{d\dot{\theta}}{d\theta} & \frac{d\dot{\theta}}{dw_\psi} & \frac{d\dot{\theta}}{d\psi} \\ 0 & 0 & 1 & 0 & 0 & 0 \\ \frac{d\dot{\psi}}{dw_\phi} & \frac{d\dot{\psi}}{d\phi} & \frac{d\dot{\psi}}{dw_\theta} & \frac{d\dot{\psi}}{d\theta} & \frac{d\dot{\psi}}{dw_\psi} & \frac{d\dot{\psi}}{d\psi} \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{18}$$

and,

$$B = \begin{bmatrix} \frac{d\dot{\phi}}{dM_1} & \frac{d\dot{\phi}}{dM_2} & \frac{d\dot{\phi}}{dM_3} & \frac{d\dot{\phi}}{dM_4} & \frac{d\dot{\phi}}{dW_1} & \frac{d\dot{\phi}}{dW_2} & \frac{d\dot{\phi}}{dW_3} & \frac{d\dot{\phi}}{dW_4} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{d\dot{\theta}}{dM_1} & \frac{d\dot{\theta}}{dM_2} & \frac{d\dot{\theta}}{dM_3} & \frac{d\dot{\theta}}{dM_4} & \frac{d\dot{\theta}}{dW_1} & \frac{d\dot{\theta}}{dW_2} & \frac{d\dot{\theta}}{dW_3} & \frac{d\dot{\theta}}{dW_4} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{d\dot{\psi}}{dM_1} & \frac{d\dot{\psi}}{dM_2} & \frac{d\dot{\psi}}{dM_3} & \frac{d\dot{\psi}}{dM_4} & \frac{d\dot{\psi}}{dW_1} & \frac{d\dot{\psi}}{dW_2} & \frac{d\dot{\psi}}{dW_3} & \frac{d\dot{\psi}}{dW_4} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{19}$$

where dw_ϕ is the roll rotation rate, dw_θ is the pitch rotation rate and dw_ψ is the yaw rotation rate. Using the system shown by equation (17), the LQR controller returns an optimal control strategy as follows:

$$u = -Kx, \tag{20}$$

with the cost function

$$J = \int (x^T Q x + u^T R u) dt, \tag{21}$$

where Q and R are the state error and control input cost matrices respectively. Using this solution, a control strategy can be computed for each discrete point in the transition trajectory. The K matrix is a feedback matrix defined from a γ of 90° for hovering, to 1.745° for forward flight. K can be linearly interpolated to get a continuous feedback system.

Furthermore, in addition to the LQR controllers that track the orientation of the aircraft, outer loop PID controllers can be used to control the position and trajectory. The outer loop controllers can consist of an altitude and a positional controller. The former can use two gain scheduling PIDs to correct for the altitude due to throttle changes in hover. As the eVTOL starts to transition, the controller corrects for the altitude by changing the vehicle's pitch. The positional controller can be categorized into two flight segments: hover and forward flight. These controllers can switch based on if the eVTOL is hovering or γ is vertical. The hover controller obtains the desired velocity through a calculated roll and pitch attitude like a conventional multi-copter, and this desired attitude is rotated by the aircraft's yaw angle to be able to correct its position. These desired roll and pitch attitudes as well as its yaw heading are fed into the LQR controller to achieve the desired orientation. An overview of the controller systems is shown below in Figure 8 [13].

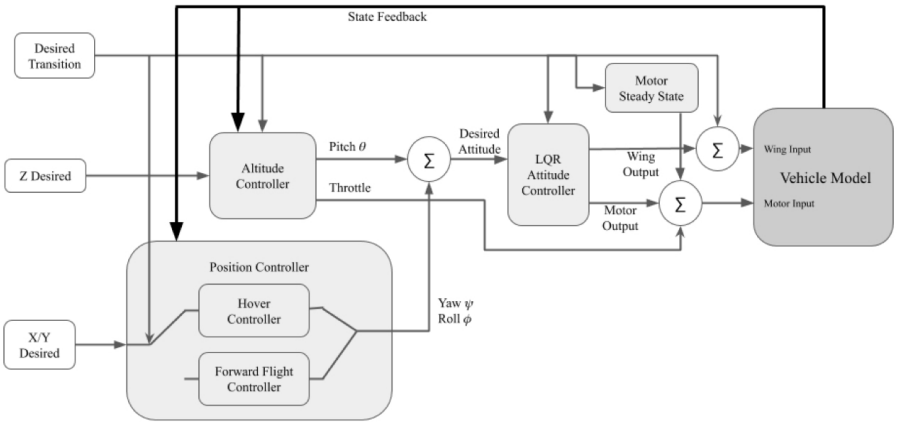


Figure 8: Overview of the Control Systems with PID and LQR Controllers [13]

2.2.6 Control Approach 5: Non-linear Dynamic Inversion

Non-linear dynamic inversion is a method that was first introduced by Krener and Brockett in the 1970s. This approach is based on the linearization of the model dynamics of an aircraft, which would then allow conventional linear controllers like PID controllers to stabilize the system. For this approach, one must know the exact system dynamics, otherwise the method can introduce instability and a loss of performance [15].

2.2.6.1 Single Input Single Output Model The concept of non-linear dynamic inversion can be understood by observing a single input single output (SISO) model first, in the form:

$$\dot{\mathbf{x}} = f(\mathbf{x}) + g(\mathbf{x})u, \tag{22}$$

where $\mathbf{x}$ is the state vector, u is the scalar control input vector, and $f(\mathbf{x})$ and $g(\mathbf{x})$ are non-linear functions. Representing equation 22 in companion form results in [16]:

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ . \\ . \\ . \\ x_{(n-1)} \\ x_n \end{bmatrix} = \begin{bmatrix} x_2 \\ . \\ . \\ . \\ x_n \\ f(\mathbf{x}) + g(\mathbf{x})u \end{bmatrix}. \tag{23}$$

Then, the virtual control input v, that is defined as:

$$v = \frac{dx_n}{dt} = f(\mathbf{x}) + g(\mathbf{x})u, \tag{24}$$

can be used to cancel the non-linearities of the system. The control input u can be determined by:

$$u = g^{-1}(\mathbf{x})[v - f(\mathbf{x})]. \tag{25}$$

Next step is to determine how to set the virtual input vector v. In order to make the resulting system exponentially stable, the following control law is used for v:

$$v = -k_0 x - k_1 \dot{x} - ... - k_{n-1} x^{(n-1)}, \tag{26}$$

where $k_0, k_1, ..., k_{n-1}$ are the proportional gains. Since it is also known that $v = \frac{dx_n}{dt}$, the whole system can be represented in a linear closed loop form as follows:

$$x^{(n)} + k_{n-1}x^{(n-1)} + ... + k_1\dot{x} + k_0 x = 0. \tag{27}$$

The stabilization of this system can be achieved through two loops: the **outer loop** is used for obtaining v using equation 26, and the **inner loop** is used for determining the corresponding value of u using equation 25.

For tasks with a desired tracking output $x_d(t)$, the error can be defined as $e = x - x_d$ and the control law becomes:

$$v = -k_0 e - k_1 \dot{e} - ... - k_{n-1}e^{(n-1)}. \tag{28}$$

Similar to equation 27, the closed loop version of equation 28 is given as:

$$e^{(n)} + k_{n-1}e^{(n-1)} + ... + k_1\dot{e} + k_0 e = 0. \tag{29}$$

The system described by equation 29 and the control law shown in equation 28 also follow a two loop structure where the inner loop performs the dynamic inversion to solve for u (equation 25), and the outer loop based computes v as shown in equation 29. The control architecture for this system is shown in Figure 9.

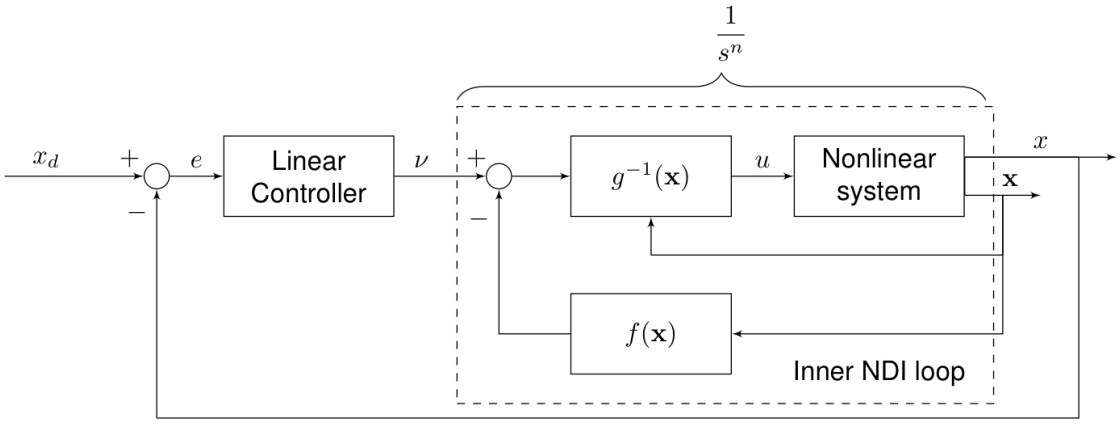


Figure 9: NDI Tracking Problem Control Architecture [15]

The explanation above of a SISO system was derived using the companion form. However, it can be difficult to represent a non-linear system in companion form, meaning there is may not be an explicit relation between the control inputs and the outputs. In this case, a

common practice is to use **input-output linearization**. To illustrate how the input-output linearization works, the following state-space representation of a SISO system can be used:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})u \tag{30}$$

$$y = h(\mathbf{x}), \tag{31}$$

where $\mathbf{x}$ is the state vector, u is the scalar control input vector, and $\mathbf{f}(\mathbf{x})$ and $\mathbf{g}(\mathbf{x})$ are non-linear vector fields, y is the scalar control output and $h(\mathbf{x})$ is a non-linear function. The method of input-output linearization has two important steps:

1. The derivative of the non-linear output y is computed iteratively until the control input u appears in it. This is done to find a relation between the output and the input.

   The differentiation of the output gives:

   $$\dot{y} = \frac{\partial h(\mathbf{x})}{\partial \mathbf{x}} \frac{d\mathbf{x}}{dt} = \nabla h(\mathbf{x})\dot{\mathbf{x}} = \nabla h(\mathbf{x})[\mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})u], \tag{32}$$

   where ∇ is the gradient operator with respect to $\mathbf{x}$.

   The **Lie derivative** $L_f h(\mathbf{x})$ is the gradient of a scalar function h($\mathbf{x}$) projected along a vector function:

   $$L_f h(\mathbf{x}) = \nabla h(\mathbf{x})\mathbf{f}(\mathbf{x}). \tag{33}$$

   Using the Lie derivative, equation 32 can be rewritten as:

   $$\dot{y} = L_f h(\mathbf{x}) + L_g h(\mathbf{x})u. \tag{34}$$

   As long as $L_g h(\mathbf{x})$ is not equal to 0, it is possible to solve for u assuming $\mathbf{v} = \dot{y}$:

   $$u = L_g h(\mathbf{x})^{-1}[v - L_f h(\mathbf{x})]. \tag{35}$$

   Moreover, higher order Lie derivatives can be obtained recursively. The i^{th} Lie derivative of h($\mathbf{x}$) is given by:

   $$L_f^i h(\mathbf{x}) = L_f[L_f^{(i-1)} h(\mathbf{x})] = \nabla[L_f^{(i-1)} h(\mathbf{x})]\mathbf{f(x)} \tag{36}$$

   $$L_g L_f h(\mathbf{x}) = [L_f h(\mathbf{x})]\mathbf{g(x)}. \tag{37}$$

Then, the generalized expression for u using Lie derivatives is as follows:

$$u = L_g L_f^{(r-1)} h(\mathbf{x})^{-1} [v - L_f^r h(\mathbf{x})], \tag{38}$$

where r denotes the **relative degree** r of the system (i.e $v = y^{(r)}$), meaning it represents the number of derivatives that need to be computed until a relation between the input and the output is obtained.

Another important term is the **order** of the system which is the number of state variables, represented by n. The system is linearizeable when r ≤ n. If r < n, the part of the input-output linearization becomes unobservable, and this is called the **internal dynamics**. In order for the controller to work properly, internal dynamics must be stable (bounded). Since the internal dynamics are often non-linear, it can be troublesome to determine the stabiliy of the unobservable part. For simplicity, making the assumption that the internal dynamics are linear, their eigenvalues are known as **zero dynamics** and correspond to the zeros of the transfer function of the control variables. n represents the number of poles in the system, and n - r is the number of zeros, meaning r is the number of excess poles. Thus, internal dynamics are stable if the zeros are located in the left half-plane [15].

2. Once the input-output relationship is obtained, a state local coordinate transformation must be defined that converts the system into companion form as shown above, such that a linear controller can be designed and placed in the outer loop for tracking desired performance.

   The Lie derivative can be used for this transformation that goes from the old state **x** to **z**. **z** is the so-called **linearizing state** that puts the system in the canonical form. Defining a function $\phi(\mathbf{x})$:

$$\mathbf{z} = \Phi(\mathbf{x}) = [\phi_1(\mathbf{x}), ..., \phi_r(\mathbf{x}), \phi_{r+1}(\mathbf{x}), ..., \phi_n(\mathbf{x})]^T, \tag{39}$$

   where the first r components can be obtained from the Lie derivatives as follows:

$$\phi_i = L_f^{i-1} h(\mathbf{x}), \tag{40}$$

   where i = 1, . . . r. The rest of the components transformation are the last n - r elements

and they are not a part of the internal dynamics. This means that they are not relevant as far as the input u is concerned. For $1 \leq i <\leq r$, the state transformation satisfies the following:

$$\dot{z}_i = \frac{\partial \phi_i(\mathbf{x})}{\partial \mathbf{x}}\dot{\mathbf{x}} = \frac{\partial}{\partial \mathbf{x}}[L_f^{i-1}h(\mathbf{x})]\mathbf{f}(\mathbf{x}) + \frac{\partial}{\partial \mathbf{x}}[L_f^{i-1}h(\mathbf{x})]\mathbf{g}(\mathbf{x})u = L_f^i h(\mathbf{x}) + L_g L_f^{i-1}h(\mathbf{x})u, \tag{41}$$

where $L_g L_f^{i-1}h(\mathbf{x})u \neq 0$ when i = r. Now, in order to form the canonical form, the following are set:

$$\dot{\mathbf{x}} = \Phi^{-1}(\mathbf{z}) \tag{42}$$

$$f(\mathbf{z}) = L_f^r h(\mathbf{x}) \tag{43}$$

$$g(\mathbf{z}) = L_g L_f^{r-1}h(\mathbf{x}), \tag{44}$$

and Equation 45 can be obtained. For components where i < r, equation 46 applies where i = 1, . . ., r-1.

$$\dot{z}_r = f(\mathbf{z}) + g(\mathbf{z})u \tag{45}$$

$$\dot{z}_i = L_f^i h(\mathbf{x}) = z_{i+1}. \tag{46}$$

As for the last n - r components that need to be determined, by choosing $L_g L_f^{i-1}h(\mathbf{x}) = 0$, the following equation can be obtained:

$$\dot{z}_i = L_f^i \phi(\mathbf{x}) = q_i(\mathbf{z}), \tag{47}$$

where i = r + 1,...,n. Since $q_i(\mathbf{z})$ does not depend on u, the states $z_{r+1}, ..., z_n$ are not controlled. Lastly, the system can be put into the final state-space representation as follows:

$$\frac{d}{dt}\begin{bmatrix} z_1 \\ . \\ . \\ . \\ z_{r-1} \\ z_r \\ z_{r+1} \\ . \\ . \\ . \\ z_n \end{bmatrix} = \begin{bmatrix} z_2 \\ . \\ . \\ . \\ z_r \\ f(\mathbf{z}) + g(\mathbf{z})u \\ q_{r+1}(\mathbf{z}) \\ . \\ . \\ . \\ q_n(\mathbf{z}) \end{bmatrix}. \tag{48}$$

As derived earlier, by introducing a virtual control input $v = z_1^{(r)}$ and designing a control law based on tracking for the linear system with e = y - y_d where y = z_1, one can solve for u to obtain the following expression:

$$u = g(\mathbf{z})^{-1}[v - f(\mathbf{z})]. \tag{49}$$

2.2.6.2 Multiple Input Multiple Output Model The derivations shown in Section 2.2.6.1 can be used in the context of a multiple input multiple output (MIMO) system. Such systems are of the form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x})\mathbf{u} \tag{50}$$

$$\mathbf{y} = \mathbf{h}(\mathbf{x}), \tag{51}$$

where **x** is the state vector of order n, **u** is the control input vector **y** is the control output vector, **f(x)** and **g(x)** are field vectors and **G(x)** is a square matrix that depends on the state vector (non-linearly in most cases). As before, the control output **y** has to be differentiated until the control input **u** appears in it. The derivative of a component of the control output vector is as follows:

$$\dot{y}_i = \frac{\partial h_i}{\partial \mathbf{x}}\frac{d\mathbf{x}}{dt} = \nabla h_i[\mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x})u] = \nabla h_i[\mathbf{f}(\mathbf{x}) + \mathbf{g}_1(\mathbf{x})u_1 + ... + \mathbf{g}_n(\mathbf{x})u_n], \tag{52}$$

where g_j, j = 1, . . . , n multiplied by the input u_j correspond to the columns of **G(x)**. As

before, the Lie derivative can be used in this expression:

$$\dot{y}_i = L_f h_i(\mathbf{x}) + \sum_{j=1}^{n} L_{gj} h_i(\mathbf{x}) u_j. \tag{53}$$

In the single input single output case, it was shown that there may not be an explicit relation between the input and the output, implying $L_{gj} h_i(\mathbf{x}) u_j = 0$ where j = 1, . . ., n. Using the the relative degree, equation 53 can be re-written as:

$$y_i^{(ri)} = L_f^{(ri)} h_i(\mathbf{x}) + \sum_{j=1}^{n} L_{gj} L_f^{(ri-1)} h_i(\mathbf{x}) u_j, \tag{54}$$

where r_i is the relative degree of the i^{th} output component. Also, the total relative degree of the MIMO system is given by $r = \sum_{j=1}^{n} r_j$. As mentioned above, in the case that the relative degree r is less than the total order of the system, unless the internal dynamics are bounded, stability and effective control cannot be guaranteed. In order to solve for the control input **u**, the derivatives of the outputs that have dependence on the control input can be put in a matrix form as follows:

$$\begin{bmatrix} y_1^{(r_1)} \\ . \\ . \\ . \\ y_n^{(r_n)} \end{bmatrix} = \mathbf{a(x)} + \mathbf{B(x)u}, \tag{55}$$

where **a(x)** and **B(x)** contain the Lie derivatives as follows:

$$\mathbf{a(x)} = \begin{bmatrix} L_f^{r_1} h_1(\mathbf{x}) \\ . \\ . \\ . \\ L_f^{r_n} h_n(\mathbf{x}) \end{bmatrix}, \tag{56}$$

and

$$\mathbf{B}(\mathbf{x}) = \begin{bmatrix} L_{g1}L_f^{(r1-1)}h_1(\mathbf{x}). & . & . & L_{gn}L_f^{(r1-1)}h_1(\mathbf{x}) \\ . & . & . & . \\ & & & \\ . & . & . & . \\ L_{g1}L_f^{(rn-1)}h_n(\mathbf{x}). & . & . & L_{gn}L_f^{(rn-1)}h_n(\mathbf{x}) \end{bmatrix}. \tag{57}$$

Lastly, in order to solve for the control input **u**, similar to the SISO system, a virtual control vector $v = [v_1, ..., v_n]^T$ can be substituted for the derivatives of the outputs, which results in the following expression for the control input:

$$\mathbf{u} = \mathbf{B}(\mathbf{x})^{-1}[v - \mathbf{a}(\mathbf{x})]. \tag{58}$$

Thus, this concludes the theoretical derivation of the non-linear dynamic inversion based controller for multiple input multiple output systems. The controller architecture would be very similar to the diagram shown in Figure 9, but it would be for a system with multiple inputs and outputs [15].

2.2.7 Other Control Approaches

Some other approaches that are used in controller designs of VTOLs include **sliding mode controllers**, **fuzzy logic controllers** and **adaptive controllers**. The sliding mode controller is a non-linear control method that applies a discontinuous switching control signal which slides the system to the desired state. Fuzzy logic controllers use distinct controllers for different flight regimes. They discretize the transition trajectory between hover and forward flight and use a discrete controller for each section. Then, the outputs of each of these controllers are mixed to attain the desired trajectory. Lastly, an adaptive controller can be used for systems that have varying parameters by regulating the control parameters to compensate for the changing conditions. Adaptive control is useful because unlike conventional controllers, it can adapt to changing control parameters [13].

2.3 Analysis of Control Approaches

All control approaches presented in this literature review are used in the aerospace industry quite often. They all have shown to provide excellent stability for various applications. For this thesis, the selected configuration is a separate lift + cruise vehicle and the proposed controller for this design included PID controllers. Since PID controllers already have been

designed for this model, they are not the control approach that will be used for the purposes of this research. In addition, since PID controllers are linear controllers and the dynamics of the eVTOL are non-linear, it would most certainly be an appropriate alternative to use a non-linear controller.

Gain scheduling is a method that is most commonly used in combination with other controllers to boost their performance such as LQR controllers. LQR controllers are highly optimal and known to provide a good stability. They also provide a more optimal energy compared to PID and fuzzy logic controllers [16]. However, there are difficulties associated with implementing these types of controllers. For instance, obtaining an analytical solution to the Ricatti equation is difficult unless the system dynamics are simple, which would not be the case for the selected eVTOL model for this thesis. This is reasonable as LQR controllers are known to work better with linear plant dynamics. Lastly, while sliding mode controllers are good non-linear controllers, they can have disadvantages such as singularity, meaning that it would be difficult for the error to converge to 0.

Aircraft are inherently non-linear systems, including the one that is selected for this report. Therefore, it would be appropriate to use a non-linear controller like Non-linear Dynamic Inversion for this analysis. The biggest challenge associated with NDI controllers is the requirement to know the exact system dynamics. The system dynamics are included in the Delft report where the eVTOL was selected from, therefore, all the necessary information exists to implement a Non-linear Dynamic Inversion. Additionally, NDI controllers have well defined error dynamics and allow for exact tracking of a reference model. Lastly, the implementation of this type of controller would not go beyond the scope of this thesis project, unlike methods like adaptive control.

3 eVTOL Dynamics and Implementation of the NDI Controller

As mentioned in Section 1.2, the separate lift + cruise type Delft model was chosen as the configuration of interest with the goals of creating an alternative control architecture for the model. This new control approach proposed makes use of the Non-linear Dynamic Inversion method.

3.1 The Separate Lift + Cruise Delft Model Specifications

This model is a fixed-wing, fixed-rotor eVTOL with two wings as shown in Figure 3. The wings are not located at the same height, meaning that they are at an angle with respect to the body of the aircraft. This was done to reduce the impact the front propellers and wing would have on the back propellers.

This eVTOL includes features for auto-stabilizing when hovering and cruising as follows:

1. **Ailerons:** They are located on the front wing to generate forces that would enforce a rolling moment. The aileron deflection angle enables one to change the lift forces of the aircraft. The aileron deflection angle is limited to be in the range of -20 to +20 degrees. Using the aileron deflection angle, one can estimate the lift coefficient.

2. **Remote Piloting:** This eVTOL is designed to be remotely controlled to maximize cargo transportation per trip and reduce the operating costs. Additionally, since the European Union Aviation Safety Agency (EASA) has already approved regulations for Unmanned Aerial Vehicles (UAV), this project is chosen to be remotely controlled.

3. **Communication:**The Line of Sight (LOS) communication type is used for this eVTOL. LOS can provide coverage up to a few hundred kilometers, which is sufficient for the Delft model since it has a maximum range of 125 kilometers [8].

3.2 Design of the NDI Controller

3.2.1 eVTOL Dynamics

All 6-degrees of freedom are considered in the design of this controller. These first three degrees of freedom include the $\vec{x}$ position of the centre of gravity of the eVTOL in the inertial frame. It is assumed that this frame is a flat Earth model with a gravitational constant of 9.8

$\frac{m}{s^2}$. The other 3 degrees of freedom are the Euler angles that are used for the orientation of the model, which are roll, pitch and yaw.

There are two reference frames used in the derivation of the dynamics. These are the inertial and body frames which are represented by $\vec{x}$ and $\vec{x}_B$ respectively. The Delft report provides a diagram of the body frame shown from the side and top views, which can be seen in Figure 10.

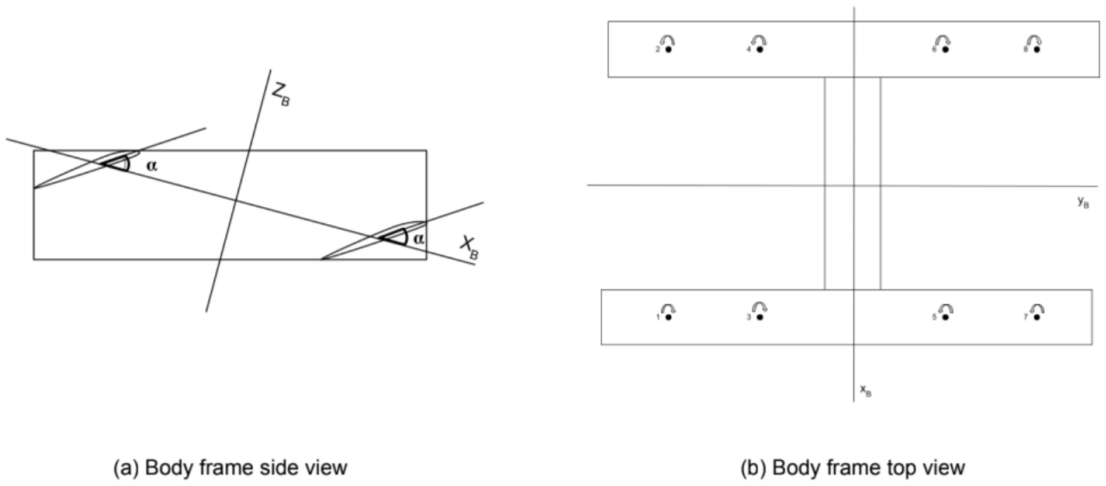


Figure 10: Body Frame From Different Views [8]

It can be seen from Figure 10 that since the wings are not centred vertically, the body frame is tilted and the wing to body angle α is 40°. The origin of the body frame is the centre of mass of the eVTOL configuration.

The Delft report proposes a design with PID controllers for the vertical motion only, which is why the horizontal equations of motion will not be considered in this report. The forces that will be considered are the gravitational force acting on the centre of gravity ($\vec{F}_g$), the drag force ($\vec{F}_d$) and lastly the thrust force due to the motors ($\vec{F}_t$). The thrust force is in the body frame and the other two forces are in the inertial frame. Therefore, the thrust force must be expressed in the inertial frame. This is possible by using a rotation matrix **R** that translates from the body frame to the inertial frame using Euler angles. This matrix looks as follows:

$$\mathbf{R} = \begin{bmatrix} c(\theta)c(\psi) & c(\theta)s(\psi) & -s(\theta) \\ -c(\phi)s(\psi) + s(\phi)s(\theta)c(\psi) & c(\phi)c(\psi) + s(\phi)s(\theta)s(\psi) & s(\phi)c(\theta) \\ s(\phi)s(\psi) + c(\phi)s(\theta)c(\psi) & -s(\phi)c(\psi) + c(\phi)s(\theta)s(\psi) & c(\phi)c(\theta) \end{bmatrix}, \tag{59}$$

where s represents sine and c represents cosine. Using this rotation matrix, the total vertical forces can be obtained by:

$$\vec{F} = \vec{F}_g + \mathbf{R}^{-1}\vec{F}_t + \vec{F}_d. \tag{60}$$

The gravitational force acts in the downward z direction, which is defined as negative:

$$\vec{F}_g = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix}, \tag{61}$$

where m is the mass and g is the gravitational constant. The drag force is given by:

$$\vec{F}_d = \begin{bmatrix} -k_d v_x \\ -k_d v_y \\ -k_d v_z \end{bmatrix}, \tag{62}$$

where k_d is the known drag coefficient in vertical flight and v_x, v_y and v_z are the eVTOL velocity in x, y and z directions respectively. As shown in equation 62, the drag has a linear relation with the velocity due to low air speeds. Lastly, the thrust force due to the motors are determined as follows:

$$\vec{F}_t = \begin{bmatrix} \sum_{n=1}^{8} T_{x,i} \\ 0 \\ \sum_{n=1}^{8} T_{z,i} \end{bmatrix} = \begin{bmatrix} \sum_{n=1}^{8} k_T \cos(\alpha)\omega_i^2 \\ 0 \\ \sum_{n=1}^{8} k_T \sin(\alpha)\omega_i^2 \end{bmatrix}, \tag{63}$$

where k_T is the thrust coefficient in vertical flight, and ωs are the rotor speed of each motor. As shown in Figure 3, there is a total of 8 motors. For the derivation of the thrust force in the body frame, the diagram shown in Figure 11 was used.

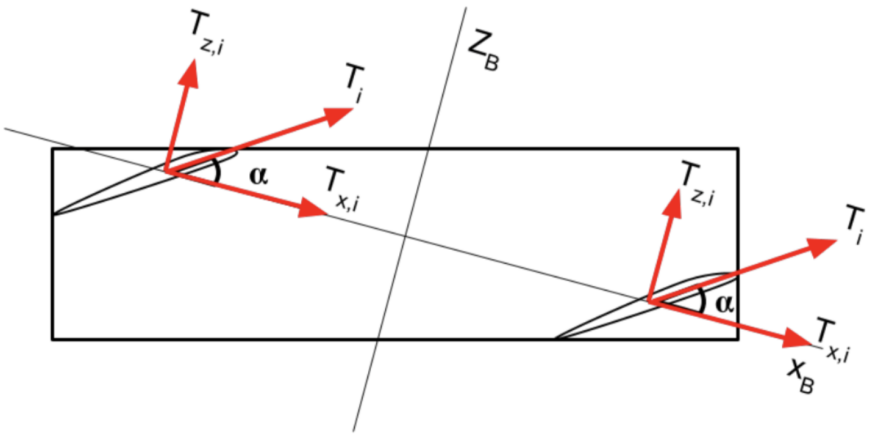


Figure 11: Thrust Force In Body Frame of eVTOL [8]

Next, rotational dynamics are determined. The total moment $\vec{M}$ consists of moments due to the thrust forces acting on the eVTOL ($\vec{M}_T$) and the moment created by the propellers that spin ($\vec{M}_\tau$):

$$\vec{M} = \vec{M}_T + \vec{M}_\tau. \tag{64}$$

Forces and torques in the body frame of the vertical flight are shown in Figure 12. The distances shown in this figure are used in the calculations of the moments.

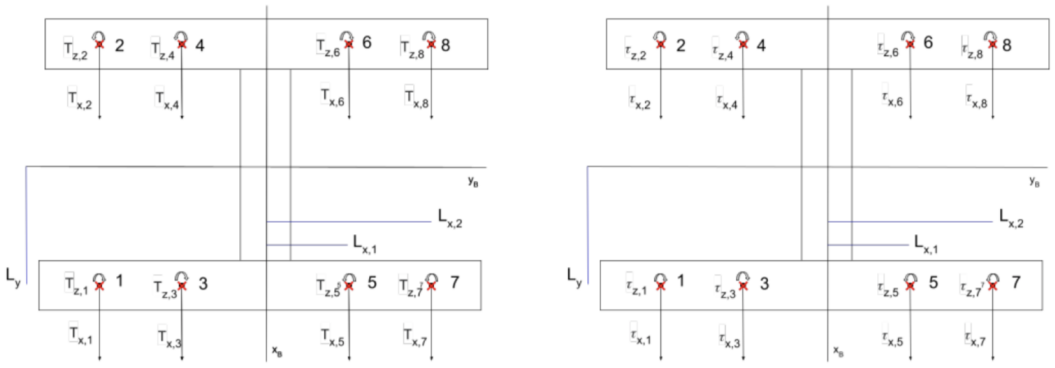


Figure 12: Forces and Moments In Body Frame of eVTOL [8]

The moment due to the thrust force is given by:

$$\vec{M}_T = \begin{bmatrix} L_{x2}(T_{z,1} + T_{z,2} - T_{z,7} - T_{z,8}) + L_{x1}(T_{z,3} + T_{z,4} - T_{z,5} - T_{z,6}) \\ L_y(T_{z,1} - T_{z,2} + T_{z,3} - T_{z,4} + T_{z,5} - T_{z,6} + T_{z,7} - T_{z,8}) \\ L_{x2}(T_{x,1} - T_{x,2} - T_{x,7} + T_{x,8}) + L_{x1}(T_{x,3} - T_{x,4} + T_{x,5} + T_{x,6}) \end{bmatrix}, \tag{65}$$

where L_{x1} is the length from the y-axis closes propeller, L_{x2} is the length from the y-axis farthest propeller and L_y is the length from x-axis to propellers as shown in Figure 12.

The moment due to the propellers force is given in Equation 66.

$$\vec{M}_\tau = \begin{bmatrix} c_\tau \cos(\alpha)(-\omega_1^2 + \omega_2^2 - \omega_3^2 + \omega_4^2 + \omega_5^2 - \omega_6^2 + \omega_7^2 - \omega_8^2) \\ 0 \\ c_\tau \sin(\alpha)(-\omega_1^2 + \omega_2^2 - \omega_3^2 + \omega_4^2 + \omega_5^2 - \omega_6^2 + \omega_7^2 - \omega_8^2) \end{bmatrix}, \tag{66}$$

where c_τ is the rotor torque coefficients. Note that motors 1, 3, 6 and 8 rotate clockwise while the rests rotates counterclockwise. The counterclockwise direction is denoted by a positive sign and the clockwise rotation is denoted by a negative sign. This completes the required dynamics for the controller design and implementation [8].

3.2.2 Implementation of the Controller

Using the Non-linear Dynamic Inversion theory explained in section 2.2.6 as well as the dynamics equations from 3.2.1, the inner and outer loops of the NDI controller can be assembled.

3.2.2.1 Inner Loop This is where the the dynamic inversion occurs, and the control input vector **u** is determined. In order to achieve this, a state vector **x** must be defined. It is important to control the attitude to stabilize the eVTOL. For this reason, the angular accelerations of the body which are $\dot{p}, \dot{q}$ and $\dot{r}$ need to be controlled. In addition, in order to maintain the desired position of the aircraft, the vertical acceleration in the z direction can be tracked, which is denoted by $\ddot{Z}$. Therefore, the derivative of the inner loop state vector **x** is given by:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \\ \ddot{Z} \end{bmatrix}. \tag{67}$$

In order to determine the angular accelerations, one can use:

$$\dot{\vec{\Omega}} = I^{-1}(\vec{M} - \vec{\Omega} x (I\vec{\Omega})), \tag{68}$$

where I is the inertia matrix and $\vec{\Omega}$ is a vector containing the angular velocities. For this case, the eVTOL assumed to be symmetric around the x and y axes of the body frame, such that the inertia matrix is diagonal:

$$I = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ O & 0 & I_{zz} \end{bmatrix}. \tag{69}$$

Using equation 68, the angular accelerations can be determined and three control variables u_1, u_2 and u_3 can be defined as follows:

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{u_1 + (I_{yy} - I_{zz})qr}{I_{xx}} \\ \frac{u_2 + (I_{zz} - I_{xx})pr}{I_{yy}} \\ \frac{u_3 + (I_{xx} - I_{yy})pq}{I_{zz}} \end{bmatrix} \tag{70}$$

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} \cos(\alpha)(k_T(-\omega_1^2 - \omega_2^2 + \omega_3^2 + \omega_4^2)(L_{x2} + L_{x1}) + c_\tau(-\omega_1^2 + \omega_2^2 + \omega_3^2 - \omega_4^2)) \\ 2k_T \cos(\alpha)(-\omega_1^2 + \omega_2^2 + \omega_3^2 - \omega_4^2) \\ \sin(\alpha)(k_T(-\omega_1^2 + \omega_2^2 - \omega_3^2 + \omega_4^2)(L_{x2} + L_{x1}) + c_\tau(-\omega_1^2 + \omega_2^2 + \omega_3^2 - \omega_4^2)) \end{bmatrix} \tag{71}$$

where $p, qandr$ are the angular velocities in the body frame. As for the last component of the state vector **x**, the vertical force equation 60 and Newton's second law $F = ma$ can be used to solve for the acceleration in the z direction. The accelerations and the last control input vector component u_4 are given by:

$$\begin{bmatrix} \ddot{X} \\ \ddot{Y} \\ \ddot{Z} \end{bmatrix} = \frac{u_4}{m}\mathbf{R}^{-1} \begin{bmatrix} \cos(\alpha) \\ 0 \\ \sin(\alpha) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \begin{bmatrix} -k_d\dot{X} \\ -k_d\dot{Y} \\ -k_d\dot{Z} \end{bmatrix} \tag{72}$$

$$u_4 = 2k_T(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2). \tag{73}$$

Therefore, $\ddot{Z}$ can be represented by the control input component u_4:

$$\ddot{Z} = \frac{(-\sin(\theta)\cos(\alpha) + \sin(\alpha)\cos(\theta))u_4 - k_d\dot{Z}}{m} - g. \tag{74}$$

Then, the derivative of state vector $\mathbf{x}$ can be represented by:

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \\ \ddot{Z} \end{bmatrix} = \begin{bmatrix} \frac{u_1 + (I_{yy} - I_{zz})qr}{I_{xx}} \\ \frac{u_2 + (I_{zz} - I_{xx})pr}{I_{yy}} \\ \frac{u_3 + (I_{xx} - I_{yy})pq}{I_{zz}} \\ \frac{(-\sin(\theta)\cos(\alpha) + \sin(\alpha)\cos(\theta))u_4 - k_d\dot{Z}}{m} - g \end{bmatrix}. \tag{75}$$

Rearranging:

$$\underbrace{\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \\ \ddot{Z} \end{bmatrix}}_{\dot{\mathbf{x}}} = \underbrace{\begin{bmatrix} \frac{1}{I_{xx}} & 0 & 0 & 0 \\ 0 & \frac{1}{I_{yy}} & 0 & 0 \\ 0 & 0 & \frac{1}{I_{zz}} & 0 \\ 0 & 0 & 0 & \frac{(-\sin(\theta)\cos(\alpha)) + \sin(\alpha)\cos(\theta))}{m} \end{bmatrix}}_{\mathbf{G}(\mathbf{x})} \underbrace{\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix}}_{\mathbf{u}} + \underbrace{\begin{bmatrix} \frac{(I_{yy} - I_{zz})\Omega_y\Omega_z}{I_{xx}} \\ \frac{(I_{zz} - I_{xx})\Omega_x\Omega_z}{I_{yy}} \\ \frac{(I_{xx} - I_{yy})\Omega_x\Omega_y}{I_{zz}} \\ \frac{-k_d\dot{Z}}{m} - g \end{bmatrix}}_{\mathbf{f}(\mathbf{x})} \tag{76}$$

The control input vector $\mathbf{u}$ can be solved for by:

$$\mathbf{u} = \mathbf{G}(\mathbf{x})^{-1}(\dot{\mathbf{x}}_d - \mathbf{f}(\mathbf{x})), \tag{77}$$

where $\dot{\mathbf{x}}_d$ is the desired derivative of the x vector, which is equivalent to the virtual control input vector v, as described in Section 2.2.6.

3.2.2.2 Outer Loop This is where the virtual control input vector v is determined. Since the dynamics are inverted in the inner loop to linearize the system, the outer loop can be controlled using a linear controller. The system output vector $\mathbf{y}$ can be defined as:

$$\mathbf{y} = \begin{bmatrix} \phi \\ \theta \\ \psi \\ Z \end{bmatrix}. \tag{78}$$

The variables chosen for $\mathbf{y}$ include the roll, pitch yaw angles, as well as the position in

the z direction. A second order controller is needed for the roll, pitch and yaw angles, since they have to be differentiated twice to determine the angular accelerations. Thus, the following control laws can be used for the construction of the first three virtual control input components v:

$$\dot{p}_d = \varepsilon(\phi_d - \phi) + \beta p \tag{79}$$

$$\dot{q}_d = \varepsilon(\theta_d - \theta) + \beta q \tag{80}$$

$$\dot{r}_d = \varepsilon(\psi_d - \psi) + \beta r, \tag{81}$$

where ε and β are adjustable parameters for the controller, ϕ_d, θ_d and ψ_d are the desired roll, desired pitch and desired yaw values respectively. The relationship between the angular velocities and Euler angles is given by:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \frac{\sin(\phi)}{\cos(\theta)} & \frac{\cos(\phi)}{\cos\theta} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{82}$$

Making the assumption that the Euler angles are small, one can assume $\dot{\phi} \approx p$, $\dot{\theta} \approx q$ and $\dot{\psi} \approx r$, meaning $\ddot{\phi} \approx \dot{p}$, $\ddot{\theta} \approx \dot{q}$ and $\ddot{\psi} \approx \dot{r}$ Using these assumptions, equations 79, 80 and 81 can be rearranged as:

$$\ddot{\phi} = \varepsilon(\phi_d - \phi) + \beta\dot{\phi} \tag{83}$$

$$\ddot{\theta} = \varepsilon(\theta_d - \theta) + \beta\dot{\theta} \tag{84}$$

$$\ddot{\psi} = \varepsilon(\psi_d - \psi) + \beta\dot{\psi} \tag{85}$$

Using Laplace transform on equations 83, 84 and 85:

$$\phi s^2 = \varepsilon(\phi_d - \phi) + \beta\phi s \tag{86}$$

$$\theta s^2 = \varepsilon(\theta_d - \theta) + \beta\theta s \tag{87}$$

$$\psi s^2 = \varepsilon(\psi_d - \psi) + \beta\psi s, \tag{88}$$

and rearranging:

$$\frac{\phi}{\phi_d} = \frac{\varepsilon}{s^2 - \beta s + \varepsilon} \tag{89}$$

$$\frac{\theta}{\theta_d} = \frac{\varepsilon}{s^2 - \beta s + \varepsilon} \tag{90}$$

$$\frac{\psi}{\psi_d} = \frac{\varepsilon}{s^2 - \beta s + \varepsilon} \tag{91}$$

Equations 89, 90 and 91 essentially correspond to second-order low-pass filters in the form:

$$\frac{x}{x_d} = \frac{\omega_n^2}{s^2 - 2\xi\omega_n^2 s + \omega_n^2} \tag{92}$$

$$\varepsilon = \omega_n^2, \beta = -2\xi\omega_n^2, \tag{93}$$

where ω_n corresponds to the natural frequency of the filter, and ξ corresponds to the damping ratio. Low-pass filters are useful as there may be high frequency components in the control response that might alter the plant dynamics. Low-pass filters are used to filter these signals to prevent any undesired behaviour occurring in the plant.

Settling time refers to the time it takes for an output to fully converge within an error band, and the damping ratio describes the number of oscillations in a system that can decay. A study was done on second order systems where various damping ratios between 0.1 and 1 were tested using a unit step with a natural frequency 1 radians per second. It was found that although a big damping ratio increases the settling time, it decreases the overshoot and the oscillatory behaviours. The study suggested an initial damping ratio value of 0.9 to minimize the overshoot. The same study tested various settling time values at a damping ratio value of 0.9, and since small settling times are desired, a settling time of 1 second was suggested as an initial estimate. The graphs generated in this study are included in Appendix A .

The relationship between the settling time, damping ratio, and natural frequency is given by:

$$t_s = \frac{4}{\xi\omega_n}. \tag{94}$$

The initial values chosen for the controller are:

$$\xi = 0.9, t_s = 1s, \varepsilon \approx 19.75 s^{-2}, \beta \approx -8 s^{-1}. \tag{95}$$

These values are chosen since they were shown to work for similar quadrotor designs [15]. They will be reevaluated and changed when the controller is built, in the case that desired response is not achieved. Next, controllers should be designed for the component of the output vector **y**, which is Z. The proposed controller for this component is a PD controller, as shown below:

$$\ddot{Z}_d = K_p(Z_d - Z) + (\dot{Z_d} - \dot{Z})K_d \tag{96}$$

where K_p and K_d are the proportional and derivative gain parameters respectively.

The four controllers that are derived for the virtual control input vector v can be combined as follows:

$$v = \begin{bmatrix} \dot{p}_d \\ \dot{q}_d \\ \dot{r}_d \\ \ddot{Z}_d \end{bmatrix} = \underbrace{\begin{bmatrix} \varepsilon & 0 & 0 & 0 \\ 0 & \varepsilon & 0 & 0 \\ 0 & 0 & \varepsilon & 0 \\ 0 & 0 & 0 & K \end{bmatrix}}_{\mathbf{K}_1} \begin{bmatrix} \phi_d - \phi \\ \theta_d - \theta \\ \psi_d - \psi \\ (Z_d - Z) + (\dot{Z_d} - \dot{Z}) \end{bmatrix} + \underbrace{\begin{bmatrix} \beta & 0 & 0 & 0 \\ 0 & \beta & 0 & 0 \\ 0 & 0 & \beta & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{K}_2} \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \tag{97}$$

Note that a general gain variable K was used to represent the proportional and derivative gain parameters in this equation for simplicity. Equation 97 concludes the derivation of the outer loop [15].

3.2.2.3 Overview of the Controller Using the dynamics equations from 3.2.1 as well as the inner and outer loop setup from 3.2.2.1 and 3.2.2.2, a block diagram for the controller is assembled, which is shown in Figure 13.

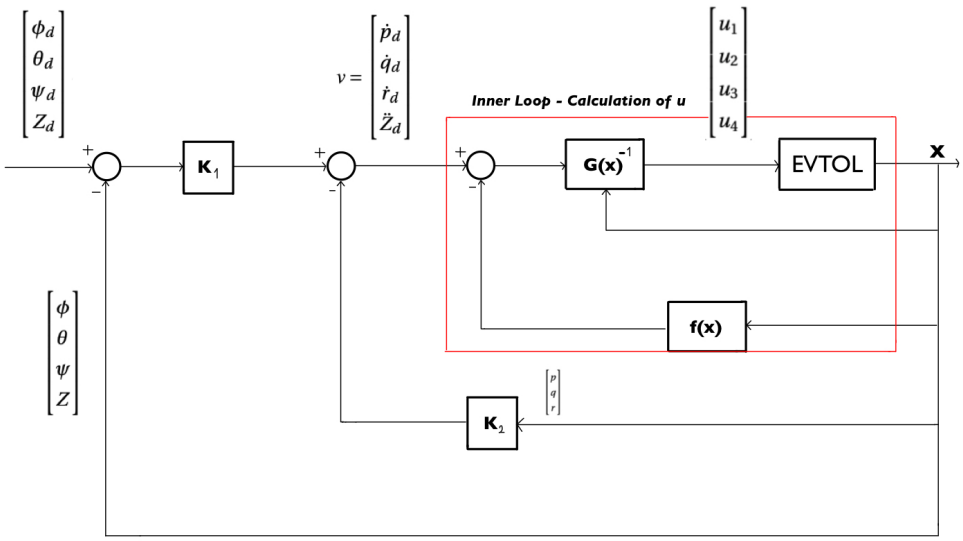


Figure 13: NDI Controller Architecture for eVTOL

Figure 13 is implemented in Simulink as shown in Figure 14. In this implementation, the rotor speeds block as well as the eVTOL dynamics blocks were created using the work presented in the Delft report.

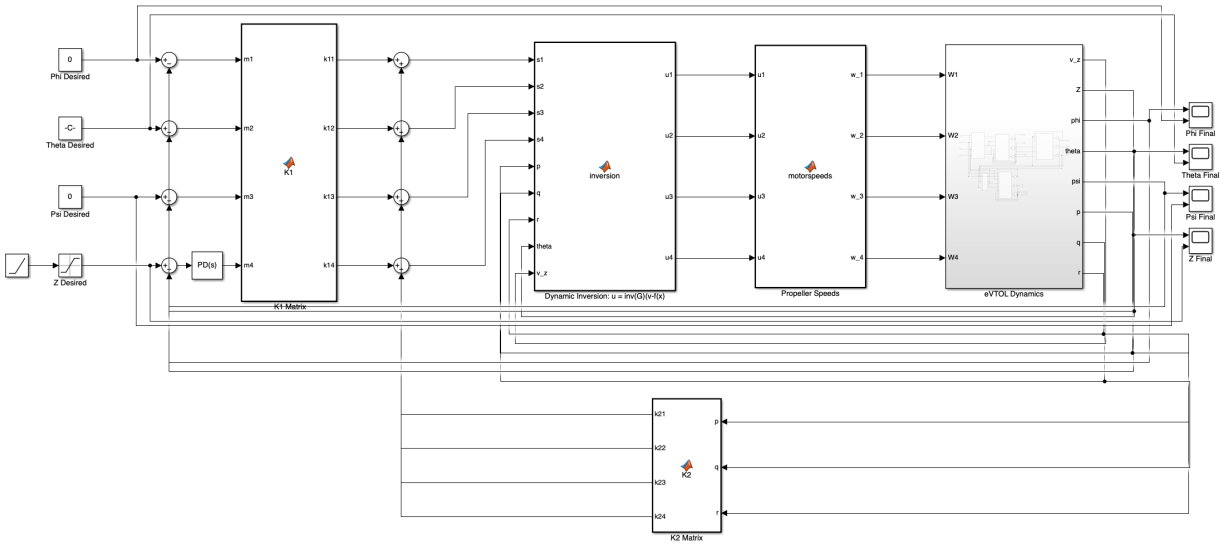


Figure 14: Simulink Implementation of the NDI Controller

The eVTOL subsytem block shown in Figure 14 looks as follows:

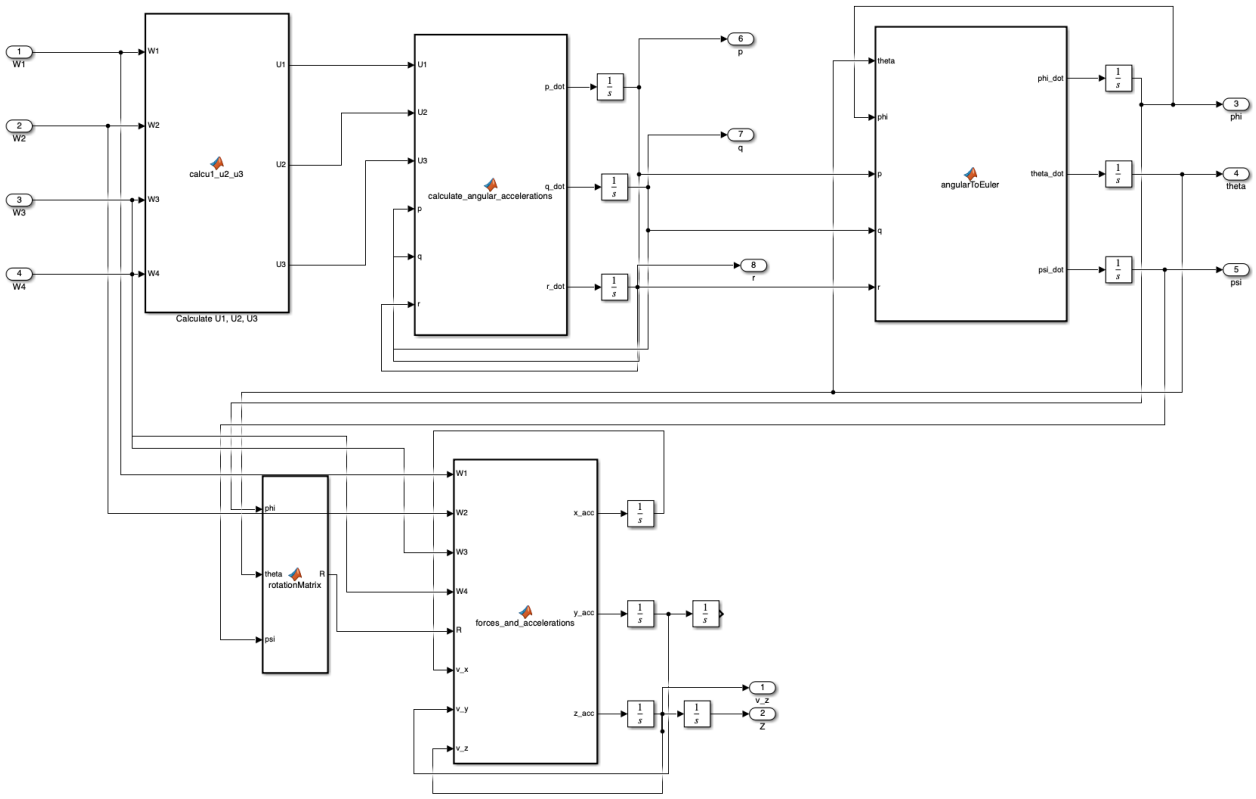


Figure 15: Simulink Implementation of the eVTOL dynamics

Here, the major difference between this implementation and the Delft report implementation is that, the Delft report makes assumptions to linearize the system dynamics. The system dynamics were not linearized for the Non-linear Dynamic Inversion controller.

4 Results of Analysis

4.1 Testing

In order to test the proposed NDI controller, performance tests were run. The tests were selected to be similar to the tests run in the Delft report. To test the roll, pitch and yaw angles, they were initialized at an offset of 20 degrees from their desired positions. In order to test the position in the Z direction, it was initialized at 0 meters (ground) and the desired position was 20 meters. Table 1 summarizes the desired orientation and position of these parameters, as well as the initialized values.

Parameter	Starting Value	Desired Value
Roll Angle	0.349 radians	0 radians
Pitch Angle	-1.22 radians	-0.8727 radians
Yaw Angle	0.349 radians	0 radians
Z Position	0 meters	20 meters

Table 1: Desired and Starting Values for Testing

These tests were run in two ways. First, all the parameters were tested simultaneously, meaning they all were initialized at an offset from the desired values at the same time (Type 1). For the second type of test, the initialization for the Euler angles with an offset was done one by one, to see if this impacted the performance of the controller (Type 2).

4.2 Tuning the PD Controller

Since a proportional-derivative control law is used for the Z position, the proportional and derivative gains were tuned in order to achieve the desired performance. While tuning the PD controller, a few factors were considered. First, since the inner and the outer loops of the NDI controller are linked through the virtual control input which has the same components of $\dot{x}$. Using a Laplace transform, the relation between these variables is simply $\frac{1}{s}$. The inner loop was replaced with an integrator in Simulink temporarily to ensure that exact system dynamics were being implemented. In order to choose the gains, the trial and error method suggested in [17] was used. This method suggests that the proportional gain should be increase until an maximum overshoot of 10 percent achieved. Then, the derivative gain should be increased until the maximum overshoot is approximately eliminated. Lastly, an integral term can be used if needed in order to control the steady state error. In this case,

an integral term was not used as there was no steady state error for the Z term. Table 2 summarizes the gains chosen after implementing this trial and error method.

Parameter	Value
K_p	4.5
K_d	2.3

Table 2: PD Controller Gains

As for the last step of the tuning process, the integrator term that represented the inner loop was replaced with the system dynamics as shown in Figure 15. It was confirmed that the gains from Table 2 resulted in the same performance with this implementation as well. Thus, this check also validated the implementation of the controller.

4.3 Results and Validation

Tests as outlined in 4.1 were simulated in Simulink. The first set of tests included the initialization of the Euler angles and the Z position at an offset simultaneously. The following figures show the results of these simulations. Here, the red curve is the response of the controller, and the dotted black line is the desired value.

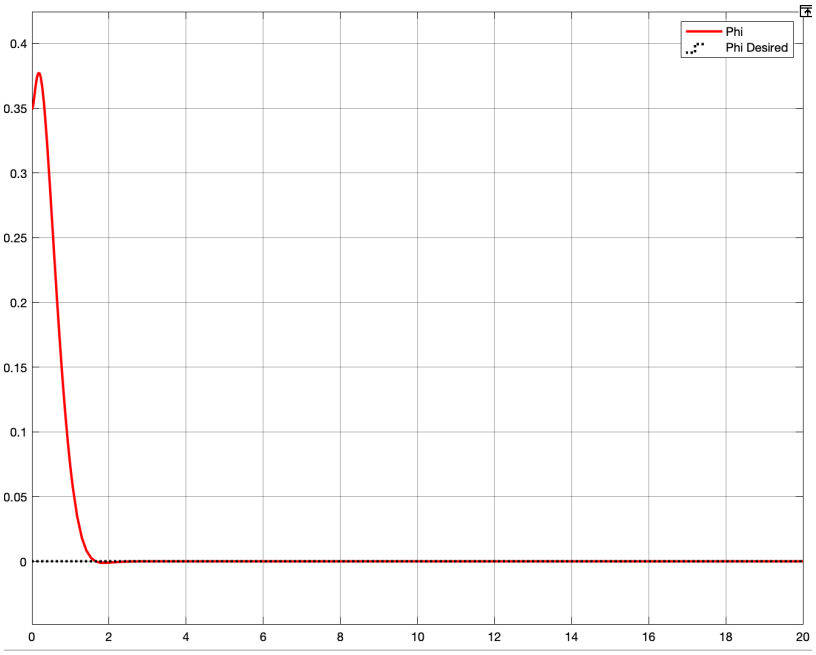


Figure 16: Roll Response, Type 1 Test

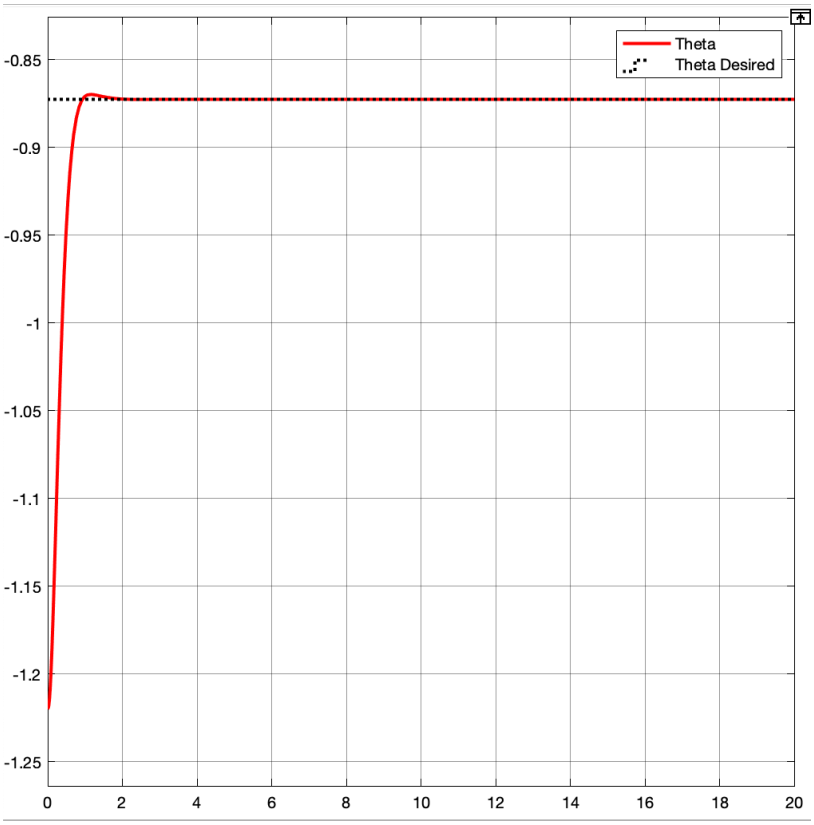


Figure 17: Pitch Response, Type 1 Test

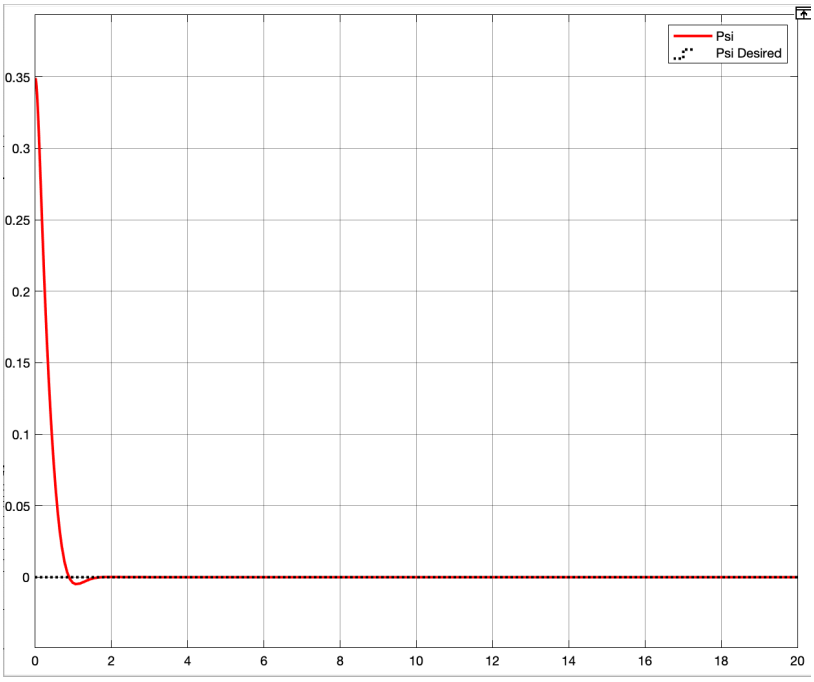


Figure 18: Yaw Response, Type 1 Test

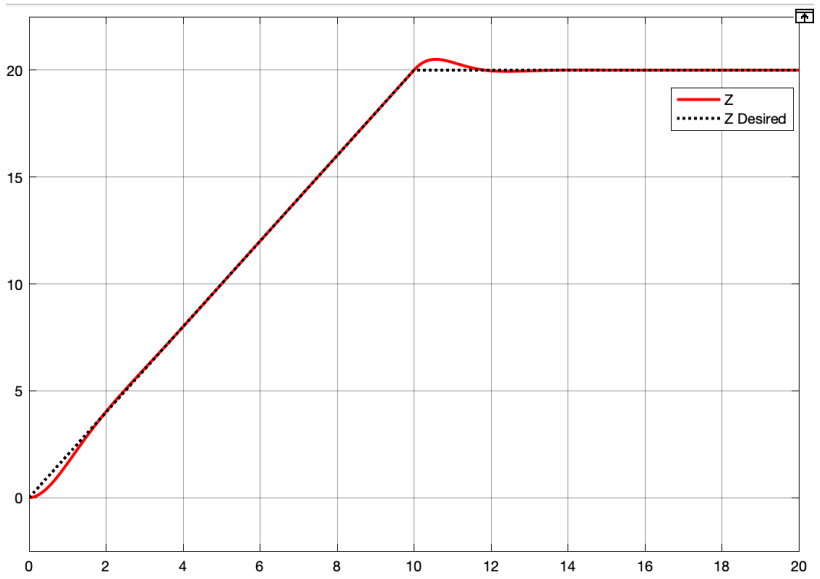


Figure 19: Z Response

As shown in figures 16-19, each simulation successfully converges to the desired orientation and/or position with a small settling time. In order to validate the performance of the controller, the settling times and overshoots for the Euler angles, and the maximum deviation for the Z position were considered. These results were compared to the results from the Delft report as shown in Tables 3 - 6. The purpose of this comparison is to **verify** that the results of the NDI controller are reasonable, since the PID controller from the Delft report is the only resource with proven, good performance for this specific eVTOL type.

Using the Delft report results as a reference for performance validation of the NDI controller, it can be observed that the NDI controller in fact has excellent performance. The maximum deviation for the Z is quite small, and the Euler angles have relatively small overshoots and settling times.

Controller	Roll Angle
PID (Delft)	Overshoot: 11% , Settling Time: 5.5 s
NDI	Preshoot: 9.4%, Undershoot: 0.55%, Settling Time: 1.3 s

Table 3: Roll Angle Performance Criteria, Type 1

Controller	Pitch Angle
PID (Delft)	Overshoot: 11% , Settling Time: 6.5 s
NDI	Overshoot: 1.3%, Settling Time: 1.1 s

Table 4: Pitch Angle Performance Criteria, Type 1

Controller	Yaw Angle
PID (Delft)	Overshoot: 8% , Settling Time: 4.0 s
NDI	Overshoot: 1.5%, Settling Time: 0.7 s

Table 5: Yaw Angle Performance Criteria, Type 1

Controller	Z Position
PID (Delft)	Maximum Deviation: 2 meters
NDI	Maximum Deviation: 0.25 meters

Table 6: Z Position Performance Criteria

For the next set of tests, the initializations for the Euler angles were done one by one. For example, when testing the roll angle, the initialization was made according to starting value from Table 1, and the rest of the angles were initialized at the desired values. This test was done only for the Euler angles, as the Z component is not coupled with the angles, thus the results for the Z position would not change. Figures 20 - 22 show the results for the roll, pitch and yaw response for the type 2 test. Tables 7 - 9 show the performance criteria for the plots shown in figures 20 - 22.

Comparing type 2 Euler Angle test results with type 1 test results, a few observations can be made. The type 1 test has a preshoot for the roll response, which disappears in the case of the type 2 test. In addition, the settling time and the maximum overshoot are lower in the type 2 test compared to the type 1 test. Similarly for the pitch response, the type 2 results in a smaller settling time and a maximum overshoot than the type 1 test. Lastly, for the yaw response, although the settling time is decreased in the type 2 test, the maximum overshoot is increased. In addition, in terms of computational run-time, there was no difference between the two sets of tests that were done. Although both type of tests result in excellent results, it is evident that the type 2 test is able to generate smaller settling times and maximum overshoots in most cases.

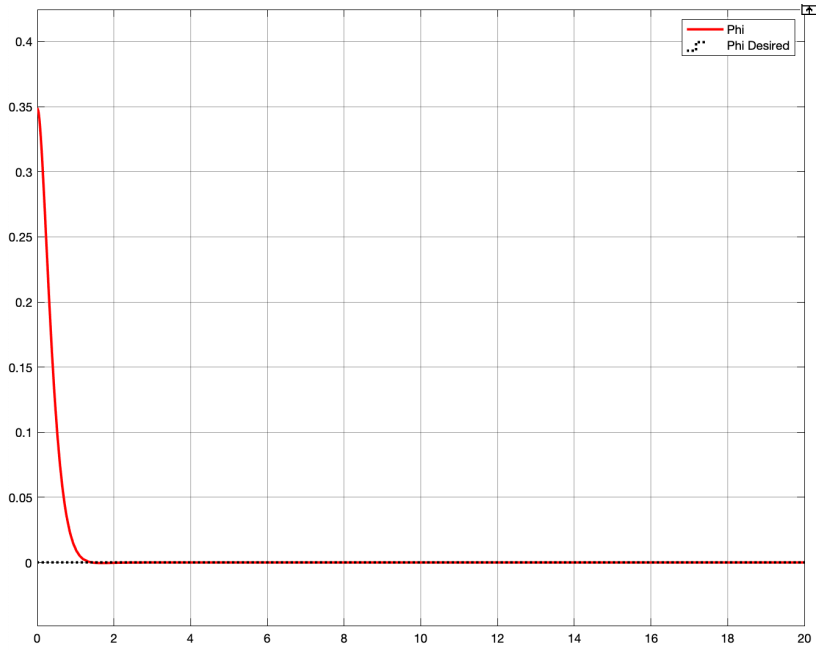


Figure 20: Roll Response, Type 2 Test

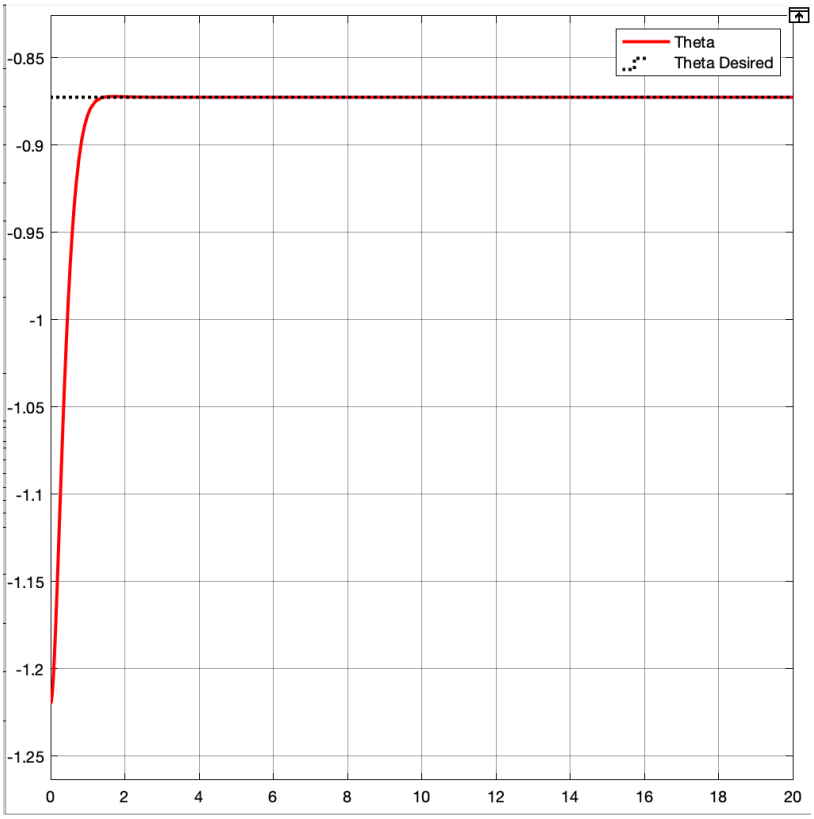


Figure 21: Pitch Response, Type 2 Test

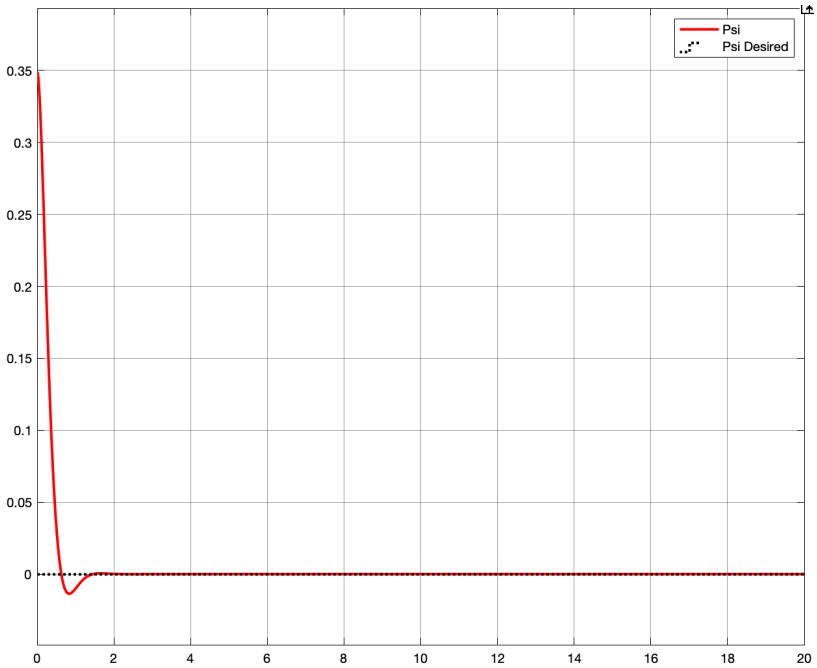


Figure 22: Yaw Response, Type 2 Test

Controller	Roll Angle
PID (Delft)	Overshoot: 11% , Settling Time: 5.5 s
NDI	Preshoot: 0%, Undershoot: 0.5%, Settling Time: 0.7 s

Table 7: Roll Angle Performance Criteria, Type 2

Controller	Pitch Angle
PID (Delft)	Overshoot: 11% , Settling Time: 6.5 s
NDI	Overshoot: 0.5%, Settling Time: 0.9 s

Table 8: Pitch Angle Performance Criteria, Type 2

Controller	Yaw Angle
PID (Delft)	Overshoot: 8% , Settling Time: 4.0 s
NDI	Overshoot: 3.6%, Settling Time: 0.6 s

Table 9: Yaw Angle Performance Criteria, Type 2

5 Conclusions and Future Work

5.1 Conclusion

Since the concept of eVTOLs have first emerged in 2009, this technology is relatively new compared to other technologies in the aviation industry. As a result, there does not exist much literature for eVTOLs. One of the particularly challenging aspects of eVTOL design is the control systems design for these airplanes. With this thesis, the purpose of creating a modelling and control platform using a selected eVTOL model and proposing alternative control systems was achieved. This platform shows how different control systems can be applicable to a specific eVTOL type, and it can be used by other engineers as a resource for their specific design problems.

In addition to the detailed, state-of-the-art literature review provided, it was shown that the control approach chosen, Non-linear Dynamic Inversion can provide adequate results when the system dynamics are known. Although this control approach has its limitations just like any other control approach, it can be applicable to many eVTOL types and provide good performance. This research contributes to the eVTOL industry by providing a reliable research database to investigate control systems. This, in return, will help speed up the overall design and certification of eVTOLs.

5.2 Future Work

The following points outline the potential tasks to be completed in the future:

1. This paper only proposes a Non-linear Dynamic Inversion controller for the vertical motion of the selected eVTOL. An NDI controller should also be designed for horizontal flight, in order to consider all flight stages for this eVTOL.

2. The eVTOL dynamics equations obtained from the Delft report included a simplied version of the aerodynamic forces. Using the non-simplied version of these equations can affect the performance of the system. Another version of the system dynamics can be created with these more complex equations, and the performance of the NDI controller can be re-evaluated. This may require the PD controller to be re-tuned, and the adjustable parameters for the outer loop may be re-selected.

3. Another version of Non-linear Dynamic Inversion called Incremental Non-linear Dynamic Inversion (INDI) could be investigated. One of the biggest limitations of

NDI is that it requires the exact knowledge of the system dynamics. INDI does not have this requirement, and it is known to be more robust than NDI in some cases.

4. The proposed NDI controller can be tested on a different eVTOL model with more complicated dynamics, such as the transition trajectory of a Vectored Thrust eVTOL.

References

[1] M. Pavel, "Understanding the control characteristics of electric vertical take-off and landing (evtol) aircraft for urban air mobility", *Aerospace Science and Technology*, vol. 125, p. 107 143, Sep. 2021. DOI: 10.1016/j.ast.2021.107143.

[2] A. Bacchini and E. Cestino, "Electric vtol configurations comparison", *Aerospace*, vol. 6, no. 3, 2019, ISSN: 2226-4310. DOI: 10.3390/aerospace6030026. [Online]. Available: https://www.mdpi.com/2226-4310/6/3/26.

[3] G. Palaia, K. Abu Salem, V. Cipolla, V. Binante, and D. Zanetti, "A conceptual design methodology for e-vtol aircraft for urban air mobility", *Applied Sciences*, vol. 11, no. 22, 2021, ISSN: 2076-3417. DOI: 10.3390/app112210815. [Online]. Available: https://www.mdpi.com/2076-3417/11/22/10815.

[4] *Lilium jet*, https://lilium.com/jet, Accessed: 2023-01-30.

[5] *Joby aviation*, https://www.jobyaviation.com/, Accessed: 2023-01-30.

[6] *Uber elevate*, https://evtol.news/uber-elevate-ecrm-002/, Accessed: 2023-01-30.

[7] *Volocopter*, https://www.volocopter.com/, Accessed: 2023-01-30.

[8] J. Holshuijsen Joris. van der Weerd, *eVTOL design, Control System*. Delft University of Technology, 2021.

[9] *Fly-by-wire flight control system*, https://www.aircraftnerds.com/2019/12/fly-by-wire-flight-control-system.html, Accessed: 2023-01-30.

[10] D. Leith and W. Leithead, "Survey of gain-scheduling analysis and design", *Int. J. Control*, vol. 73, pp. 1001–1025, Jan. 2000. DOI: 10.1080/002071700411304.

[11] A. Bernhardsson, *Gain scheduling*, Department of Automatic Control LTH, Accessed: 2023-01-30.

[12] *Gain scheduling basics*, https://www.mathworks.com/help/control/ug/gain-scheduled-control-systems.html, Accessed: 2023-01-30.

[13] T. D., *Robust Control of an eVTOL Through Transition With a Gain Scheduling LQR Controller*. University of Maryland, 2020.

[14] L. A. Sobiesak, H. Beaudette, F. Bloc-Teasdale, *et al.*, "Modelling and control of transition flight of an evtol tandem tilt wing aircraft", *EUCASS*, 2019.

[15] R. F. E. Almedia, *Incremental Nonlinear Dynamic Inversion applied to Quadrotor UAV Control*. Tecnico Lisboa, 2017.

[16] *Nonlinear dynamic inversion*, http://www.aerostudents.com/courses/advanced-flight-control/nonlinearDynamicInversion.pdf, Accessed: 2023-01-30.

[17] G. van der Zalm, "Tuning of pid-type controllers", *Technische Universiteit Eindhoven*, 2004.

A Appendix A

A.1 Damping Ratio

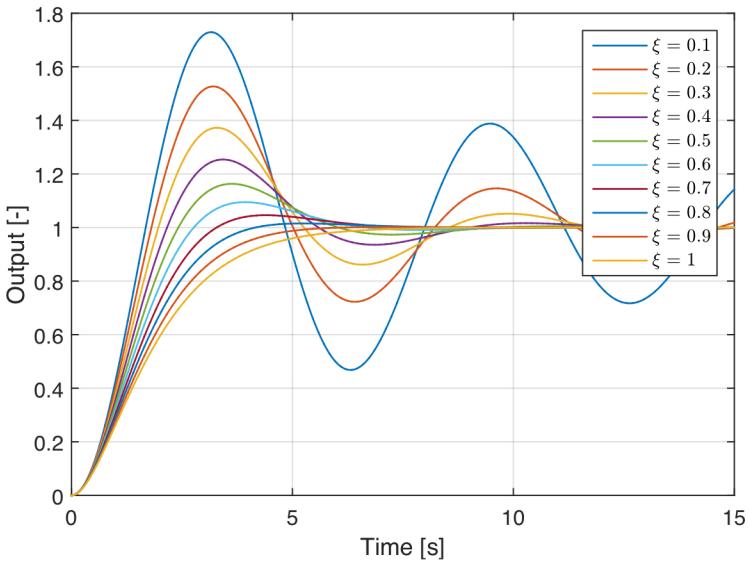


Figure 23: Time Response of Second Order System to Unit Step [15]

A.2 Settling Time

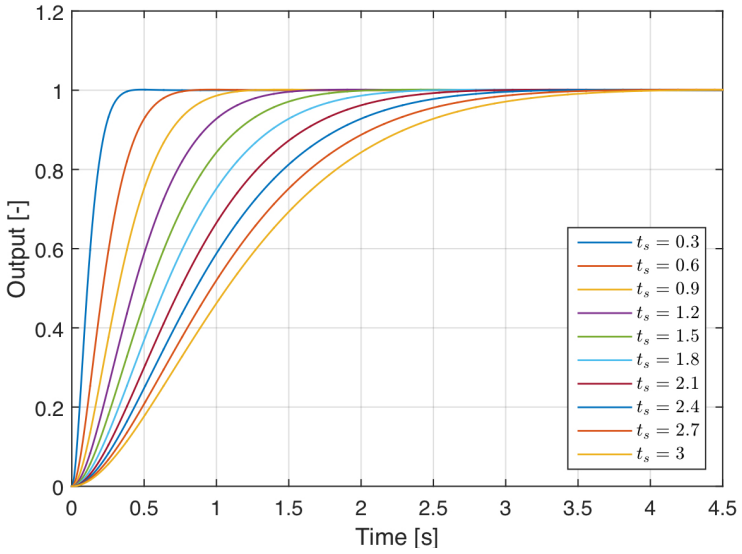


Figure 24: Time Response of Second Order System to Unit Step [15]

B Appendix B

B.1 Controller Blocks (Simulink)

```
% K1
function [k11, k12, k13, k14] = K1(m1, m2, m3, m4)

alpha = 19.75;
K = [alpha 0 0 0; 0 alpha 0 0; 0 0 alpha 0; 0 0 0 0]*[m1; m2; m3; 0];

K(4) = K(4) + m4;

k11 = K(1);
k12 = K(2);
k13 = K(3);
k14 = K(4);

end

% K2
function [k21, k22, k23, k24] = K2(p, q, r)

beta = -8;

K = [beta 0 0; 0 beta 0; 0 0 beta; 0 0 0]*[p; q; r];
k21 = K(1);
k22 = K(2);
k23 = K(3);
k24 = K(4);
end
```

Figure 25: K1 and K2 Blocks Embedded Matlab Code

```
% inversion
function [u1, u2, u3, u4] = inversion(s1, s2, s3, s4, p, q, r,theta, v_z)

% Now, creating G(x) and f(x) and then performing the inversion

Ixx = 3.6125*10^03;
Iyy = 3.5328*10^03;
Izz = 8.0078*10^03;
wing2bodyAngle = 0.6981;
m = 2500;
Kd = 1;
g = 9.81;

G_x = [1/Ixx 0 0 0; 0 1/Iyy 0 0; 0 0 1/Izz 0; 0 0 0 (-sin(theta)*cos(wing2bodyAngle) + cos(theta)*sin(wing2bodyAngle))/m];

f_x = [((Iyy-Izz)*q*r)/Ixx; ((Izz-Ixx)*p*r)/Iyy; ((Ixx-Iyy)*p*q)/Izz; ((-Kd*v_z)/m -g)];

s = [s1; s2; s3; s4]; % Components of v

difference = s - f_x;
u = G_x\difference;
u1 = u(1);
u2 = u(2);
u3 = u(3);
u4 = u(4);

end
```

Figure 26: Inversion Block Embedded Matlab Code

```
% motor speeds
% This Code is Directly Taken From [8]
function [w_1, w_2, w_3, w_4] = motorspeeds(u1, u2, u3, u4)
    wing2bodyAngle = wrapToPi(deg2rad(40));
    Ly = 2.5; %meters, distance from wing to C.O.G. parallel to x
    Lx1 = 1.6; %meters, distance to first propeller parallel to y
    Lx2 = 3.2; %meters, distance to second propeller parallel to y
    mw1 = 220; %kg, mass of front wing
    mw2 = 220; %kg, mass of back wing
    m = 2500; %kg, total mass of VTOL
    [m1, m2, m3, m4, m5, m6, m7, m8] = deal(34.5); %kg, mass of propeller + motor combination
    Ixx = Ly^2*(m2 + m4 + m6 + m8 + mw1 +mw2);
    Iyy = Lx1^2*(m3 + m4 + m5 + m6) + (Lx1+Lx2)^2*(m1 + m2 + m7 +m8);


    Izz = ((Lx1+Lx2)^2 + Ly^2)*(m1 + m2 + m7 + m8) + (Lx1^2 + Ly^2)*(m3 + m4 + m5 + m6) + Ly^2*(mw1+mw2);

    Kd = 1;
    Kmt = 0.002; %relation between rpm^2 and thurst
    Kmtorque = 0.0003; %relation between rpm^2 and torque

    X1 = (Lx1 + Lx2)*Kmt*cos(wing2bodyAngle);
    Y1 = 2*Ly*Kmt*cos(wing2bodyAngle);
    Z1 = (Lx1 + Lx2)*Kmt*sin(wing2bodyAngle);
    torques_thrust = [-X1 -X1 X1 X1; -Y1 Y1 -Y1 Y1; -Z1 Z1 Z1 -Z1];
    X2 = Kmtorque*cos(wing2bodyAngle);
    Y2 = 0;
    Z2 = Kmtorque*sin(wing2bodyAngle);
    torques_torque_rotor = [-X2 X2 X2 -X2; -Y2 Y2 Y2 -Y2; -Z2 Z2 Z2 -Z2];

    Atorque = torques_thrust + torques_torque_rotor;
    Athrust = [2*Kmt 2*Kmt 2*Kmt 2*Kmt];
    A = [Atorque; Athrust];
    u = [u1; u2; u3; u4];
    rpms = A\u;
    W1 = rpms(1);
    W2 = rpms(2);
    W3 = rpms(3);
    W4 = rpms(4);
```

Figure 27: Motor Speeds Block Embedded Matlab Code 1

```
    if W1 < 0
        W1 = 0;
    elseif W1 > 3600^2
        W1 = 3600^2;
    end

    if W2 < 0
        W2 = 0;
    elseif W2 > 3600^2
        W2 =3600^2;
    end

    if W3 < 0
        W3 = 0;
    elseif W3 > 3600^2
        W3 = 3600^2;
    end

    if W4 < 0
        W4 = 0;
    elseif W4 > 3600^2
        W4 = 3600^2;
    end

    w_1 = sqrt(W1);
    w_2 = sqrt(W2);
    w_3 = sqrt(W3);
    w_4 = sqrt(W4);

end
```

Figure 28: Motor Speeds Block Embedded Matlab Code 2

B.2 eVTOL Dynamics Blocks (Simulink)

eVTOL Dynamics

```
%calcul_u2_u3
% Taken from [8], modified
function [U1, U2, U3]= calcul_u2_u3(W1, W2, W3, W4)

wing2bodyAngle = 0.6981;
Kmt = 0.002; %relation between rpm^2 and thurst
Kmtorque = 0.0003; %relation between rpm^2 and torque

a1 = 1.6;
a2 = 3.2;
b = 2.5;

% Calculate  U.
U1 = Kmt*cos(wing2bodyAngle)*(a2*(-W1^2 - W2^2 + W3^2 + W4^2) + a1*(-W1^2 - W2^2 + W3^2 + W4^2)) + cos(wing2bodyAngle)*Kmtorque*(-W1^2 + W2^2 + W3^2 - W4^2);
U2 = 2*Kmt*b*cos(wing2bodyAngle)*(-W1^2 + W2^2 - W3^2 + W4^2);
U3 = Kmt*sin(wing2bodyAngle)*(a2*(-W1^2 + W2^2 + W3^2 - W4^2) + a1*(-W1^2 + W2^2 +  W3^2 - W4^2)) + sin(wing2bodyAngle)*Kmtorque*(-W1^2 + W2^2 + W3^2 - W4^2);

end
```

Figure 29: U1 U2 U3 Calculation Block Embedded Matlab Code

```
% angularToEuler
% Taken from [8], modified
function [phi_dot, theta_dot, psi_dot] = angularToEuler(theta, phi, p, q, r)
    angEuler = [1 sin(phi)*tan(theta) cos(phi)*tan(theta); 0 cos(phi) -sin(phi); 0 sin(phi)/cos(theta) cos(phi)/cos(theta)];
    ang = angEuler*[p; q; r];
    phi_dot = ang(1);
    theta_dot = ang(2);
    psi_dot = ang(3);
end
```

Figure 30: Angular to Euler Block Embedded Matlab Code

```
%rotationMatrix
% Taken from [8]
function R = rotationMatrix(phi, theta, psi)
    R = [cos(theta)*cos(psi)  (-cos(phi)*sin(psi) + sin(phi)*sin(theta)*cos(psi)) (sin(phi)*sin(psi) + cos(phi)*sin(theta)*cos(psi));
         cos(theta)*sin(psi) (cos(phi)*cos(psi) + sin(phi)*sin(theta)*sin(psi))  (-sin(phi)*cos(psi) + cos(phi)*sin(theta)*sin(psi));
         -sin(theta)  sin(phi)*cos(theta)  cos(phi)*cos(theta)];
end
```

Figure 31: Rotation Matrix Block Embedded Matlab Code

```
%forces_and_accelerations
% Taken from [8],
function [x_acc, y_acc, z_acc]= forces_and_accelerations(W1, W2, W3, W4, R, v_x, v_y, v_z)

wing2bodyAngle = 0.6981;
Kmt = 0.002; %relation between rpm^2 and thurst
m = 2500;
g = 9.81;
Kd = 1;

% Calculate the Total Force
Fxb = 2*Kmt*(W1^2 + W2^2 + W3^2 + W4^2)*cos(wing2bodyAngle); %in hover
Fzb = 2*Kmt*(W1^2 + W2^2 + W3^2 + W4^2)*sin(wing2bodyAngle); %in hover
F_drag = [-Kd*v_x; -Kd*v_y; -Kd*v_z];
F_thrust_b = [Fxb; 0; Fzb];
F_g = [0; 0; -m*g];

F_world_thrust = R*F_thrust_b;
F_total = F_g + F_world_thrust + F_drag;

Fx = F_total(1);
Fy = F_total(2);
Fz = F_total(3);


x_acc = Fx/m;
y_acc = Fy/m;
z_acc = Fz/m;

end
```

Figure 32: Forces and Accelerations Block Embedded Matlab Code