STATE UNIVERSITY OF SANTA CATARINA – UDESC
TECHNOLOGICAL SCIENCES CENTER – CCT
GRADUATE DEPARTMENT OF ELECTRICAL ENGINEERING – PPGEEL

DAYSE MARANHÃO CAVALCANTI

STUDY AND APPLICATION OF NEURAL NETWORK TECHNIQUES FOR DETECTING AND ADAPTING PREDICTIVE CONTROL LOOPS SUBJECT TO ABRUPT SYSTEM FAULTS

JOINVILLE
2021

DAYSE MARANHÃO CAVALCANTI

STUDY AND APPLICATION OF NEURAL NETWORK TECHNIQUES FOR DETECTING AND ADAPTING PREDICTIVE CONTROL LOOPS SUBJECT TO ABRUPT SYSTEM FAULTS

Thesis presented to the Graduate Department of Electrical Engineering at the Technological Sciences Center of the State University of Santa Catarina, in partial fulfillment of the requirements of the degree of Master of Electrical Engineering.

Advisor: Dr. Prof. Mariana Santos Matos Cavalca

JOINVILLE
2021

To generate the catalog of theses and
dissertations access the link:
https://www.udesc.br/bu/manuais/ficha

Cavalcanti, Dayse Maranhão
Study and application of neural network techniques for detecting and adapting predictive control loops subject to abrupt system faults / Dayse Maranhão Cavalcanti. - Joinville, 2021.
112 p. : il. ; 30 cm.

Orientadora: Mariana Santos Matos Cavalca.
.
Dissertação (Mestrado) - Universidade do Estado de Santa Catarina, Centro de Ciências Tecnológicas, Programa de Pós-Graduação em Engenharia Elétrica, Joinville, 2021.

1. Controle preditivo. 2. Rede neural artificial. 3. Tolerância a falhas. 4. Restrições. 5. Perturbações. I. Cavalca, Mariana Santos Matos. II. , . III. Universidade do Estado de Santa Catarina, Centro de Ciências Tecnológicas, Programa de Pós-Graduação em Engenharia Elétrica. IV. Título.

DAYSE MARANHÃO CAVALCANTI

STUDY AND APPLICATION OF NEURAL NETWORK TECHNIQUES FOR DETECTING AND ADAPTING PREDICTIVE CONTROL LOOPS SUBJECT TO ABRUPT SYSTEM FAULTS

Thesis presented to the Graduate Department of Electrical Engineering at the Technological Sciences Center of the State University of Santa Catarina, in partial fulfillment of the requirements of the degree of Master of Electrical Engineering.

Advisor: Dr. Prof. Mariana Santos Matos Cavalca

EXAMINING COMMITTEE:

Dr. Prof. Mariana Santos Matos Cavalca
State University of Santa Catarina

Members:

Dr. Prof. Gabriel Hermann Negri
Federal Institute of Santa Catarina

Dr. Prof. José de Oliveira
State University of Santa Catarina

Joinville, 31 May 2021

I dedicate this work to my family.

ACKNOWLEDGEMENTS

This work would not have been possible without the support and assistance from a great number of people. First of all, I would like to thank the Technological Sciences Center from the State University of Santa Catarina, in particular, the Graduate Department of Electrical Engineering, for allowing me this opportunity. Several professors, students, and staff members helped in different ways in the making of this dissertation thesis, especially my advisor, who is one of the most amazing people I have ever met and I will always remember her fondly. Also, I am very grateful for the sponsorship provided by FAPESC and PROMOP during this time. Last but not least, I thank my family and friends for supporting my research work and for forgiving me for missing some Sunday lunches.

“Here is another curse for you – may all your bacon burn.” (Calcifer)

ABSTRACT

The use of Model Predictive Control (MPC) on aircraft flight control consists on finding the optimal control sequence to follow a reference trajectory by predicting the aircraft movement in real time. To such extent, the main focus of this work is to influence the behavior of a didactic plant. Therefore, it is suggested the study of the system, that is, its structure, performance, restrictions, and requirements, obtaining the nominal and fault models on state-space representation, and design and implementation of a multi-model fault-tolerant controller. From the system model, the MPC predicts the process output over a time horizon or time window. That said, in the initial or zero time, it is performed an optimization process to find the best control signal that takes the output signal to the desired value in a small time period, saving the first value of the control law and observing the system response. Then, the beginning of the horizon moves to the corresponding time of that value, restarting the procedure. The fault tolerance, on the other hand, occurs through the diagnosis and consequent switching to the most appropriate predefined prediction model by an Artificial Neural Network (ANN). Two types of ANNs are studied: Multilayer Perceptron and Fully Connected Cascade. Which can recognize hidden patterns and correlations in raw data through algorithms. Also, a comparison with the classical one-model structure is discussed, and the obtained results show that the approach with multiple models is promising when it is possible to determine the dynamics models and perform a precise fault diagnosis.
Keywords: Model predictive control, artificial neural network, fault tolerance, constraints, disturbances.

RESUMO

O uso de Controle Peditivo (MPC) no controle de voo de aeronaves consiste em encontrar a sequência de controle ótima para seguir uma rota de referência em tempo real. Com isso em mente, o foco deste trabalho é influenciar o comportamento de uma planta didática. Portanto, propôs-se o estudo do sistema, isto é, sua estrutura, desempenho, restrições e requisitos, obtenção dos modelos em espaço de estados nominal e em falha, e o design e implementação de um controlador com múltiplos modelos tolerante a falhas. A partir do modelo do sistema, o MPC prevê a saída do processo dentro de um horizonte de tempo, ou janela de tempo. Dito isto, no tempo inicial ou zero, é realizado um processo de otimização no intuito de encontrar o melhor sinal de controle que leva a saída do sistema para o valor desejado em um curto período de tempo, salvando o primeiro valor da lei de controle e observando a resposta do sistema. Então, o começo do horizonte é movido para o tempo correspondente deste valor, reiniciando o procedimento. A tolerância a falhas, por outro lado, ocorre por meio do diagnóstico e consequente comutação para o modelo de de predição predefinido mais apropriado de acordo com a Rede Neural Artificial (ANN). Dois tipos de ANNs são estudadas: Perceptron Multicamadas (ou Perceptron de Múltiplas Camadas) e Cascata Totalmente Conectada. As quais são capazes de reconhecer padrões ocultos e correlações em dados brutos por meio de algoritmos. Além disso, é discutida uma comparação com a estrutura clássica de um único modelo, mostrando-se através dos resultados obtidos que a proposta de múltiplos modelos é promissora desde que seja possível determinar os modelos dinâmicos e garantir um diagnóstico de falhas preciso.

Palavras-chave: Controle preditivo, rede neural artificial, tolerância a falhas, restrições, perturbações.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

ANN	Artificial Neural Network
DCN	Deep Convolutional Network
DMC	Dynamic Matrix Control
DOF	Degrees of freedom
FCCNN	Fully Connected Cascade Neural Network
FFNN	Feed Forward Neural Network
FTC	Fault-Tolerant Control
GPC	Generalized Predictive Control
LQ	Linear Quadratic
MIMO	Multiple-input multiple-output
MHC	Moving Horizon Control
MLPNN	Multi-layer Perceptron Neural Network
MMPC	Multi-Model Predictive Control
MPC	Model predictive control
MPHC	Model Predictive Heuristic Control
NTM	Neutral Turing Machine
PID	Proportional-Integrative-Derivative
QDMC	Quadratic Dynamic Matrix Control
RBF	Radial Basis Network
RC	Remote control
RHC	Reciding Horizon Control
RNN	Recurrent Neural Network
SISO	Single-input single-output
SSMPC	State-Space Model Predictive
UAV	Unmanned Aerial Vehicles

LIST OF SYMBOLS

t	Time (s)
T_s	Settling time (s)
k	Sampling instant
f_s	Sampling frequency (s^{-1})
u	Input vector
x	State vector
y	Output vector
n	Number of states
p	Number of inputs
q	Number of outputs
A_c	Continuous-time state matrix
B_c	Continuous-time input matrix
C_c	Continuous-time output matrix
D_c	Continuous-time feedforward matrix
A_d	Discrete-time state matrix
B_d	Discrete-time input matrix
C_d	Discrete-time output matrix
D_d	Discrete-time feedforward matrix
A_ξ	Extended state matrix
B_ξ	Extended input matrix
C_ξ	Extended output matrix
D_ξ	Extended feedforward matrix
G	Dynamic matrix
F	Free system response
N	Prediction horizon
M	Control horizon
p_u	Input design parameter
p_y	Output design parameter
P_u	Input design parameter matrix
P_y	Output design parameter matrix

ref_p	Pitch angle reference (rad or degrees)
ref_r	Roll angle reference (rad or degrees)
ref_y	Yaw angle reference (rad or degrees)
R	Reference vector
S	Linear inequality constraints matrix
b	Linear inequality constraints vector
H	Quadratic objective term
Δu_{max}	Maximum input rate
Δu_{min}	Minimum input rate
u_{max}	Maximum input values
u_{min}	Minimum input values
y_{max}	Maximum output values
y_{min}	Minimum output values
χ	Input signal
Υ	Output signal
φ	Activation function
ω	Weight
θ	Pitch angle (rad or degrees)
ϕ	Roll angle (rad or degrees)
ψ	Yaw angle (rad or degrees)
V_{front}	Front rotor voltage (V)
V_{back}	Back rotor voltage (V)
V_{right}	Right rotor voltage (V)
V_{left}	Left rotor voltage (V)
D_y	Pitch damping (N.m.s/rad)
D_y	Yaw damping (N.m.s/rad)
K_{pp}	Pitch thrust gain (N.m/V)
K_{yy}	Yaw thrust gain (N.m/V)
K_{py}	Cross-torque from pitch voltage (N.m/V)
K_{yp}	Cross-torque from yaw voltage (N.m/V)
K_{sp}	Stiffness about the pitch axis (N.m/rad)

CONTENTS

1 INTRODUCTION

Although different civilizations might have had a great number of technological achievements throughout the years, a fair amount of the ancient work was done by man or animal power, and the theories proposed by some early-thinkers were not doable at the time (LANDELS, 1998). A brief look at the history of control systems shows that the earliest applications of feedback control were probably around 300 to 1 B.C. with the development of floating regulator mechanisms in Greece, which were used in Ktesibios' water clock and Philion's oil lamp (NISE, 2012). However, what is called early control only started to appear in industrial processes between the 18th and 19th centuries, like James Watt's flyball governor (OGATA, 2010).

Classical control theory appeared at the time of World War II by independent groups (DORF; BISHOP, 2011). Bode, Nyquist, and Black devised the foundation of feedback control and frequency domain at Bell Telephone Laboratories from 1927 to 1940, but, at the same time, the former Soviet Union dominated the field utilizing time-domain formulations following the lead of Lyapunov (DORF; BISHOP, 2011). Common classical controllers deal with relatively simple systems, traditionally single-input single-output (SISO), described by linear and time-invariant differential equations in the time domain, or in s-domain with the aid of Laplace transform and other transforming methods. A good example of classical control is the Proportional-Integrative-Derivative (PID) controller.

Modern control theory appeared sometime later. The earliest works are dated from the 1960s, with emphasis on Kalman's researches about linear state-space system theory and linear-quadratic (LQ) optimal control theory, and also the Kalman filter invention (BERNHARD; DESCHAMPS, 2019). Contrary to classic control, it uses mathematical concepts and computational techniques for the control of more complex systems, which may be dynamic and multiple-input multiple-output (MIMO) in the complex-s or the frequency domain represented by a set of differential equations (OGATA, 2010). Robust, adaptative, and nonlinear control fall under this category.

Model Predictive Control (MPC) is a class of modern control techniques that can be traced back to the industry in the late 1970s and the beginning of the 1980s (GARCÍA; PRETT; MORARI, 1989). However, it is said that the first generation of MPC started with Richalet et. al's Model Predictive Heuristic Control (MPHC) in 1978, based on the use of impulse response representation (CAMACHO; BORDONS, 2003). In 1980, Cuttler and Remaker proposed the Dynamic Matrix Control (DMC) at Shell Oil, which worked with the step response model, but its patent was granted to Prett et. al in 1982.

What is called the second generation of MPC covers the works of Garcia and Morchedi and Clarke et. al, respectively, the Quadratic Dynamic Matrix Control (QDMC) from 1986 and Generalized Predictive Control (GPC) from 1987 (MACIEJOWSKI, 2000). The third and fourth generations are from the 1990s and 2000s (QIN; BADGWELL, 2003). Over the years, growth has been seen in the number of academic and industrial researches about variants of MPC, also

expanding the scope of its applications from chemical plants to the most varied systems, since this method is known for handling multivariable cases, operational constraints and nonlinearities with ease (QIN; BADGWELL, 2003).

That being said, MPC is the class of controllers analyzed in this work by studying its advantages and effectiveness when applied to aerospace didactic systems. The implementation of MPC in drone control refers to path planning and flight control. Knowing that drones have been used in many different applications that require optimal performance, such as observation, inspection, and detection, the interest in the field of flight control arose, whose main goal is to operate the aircraft mechanisms in order to launch it, follow a given trajectory or direction of motion while satisfying the system operational and safety requirements.

Drones or Unmanned Aerial Vehicles (UAV) are described as distance-oriented or automated aircraft, that is, the pilot is not within the vehicle itself. They were developed as a military product, mainly used as aerial missile deployers and for enemy territory recognition or espionage. The earliest recorded use of UAVs dates back to 1849 when Austrian forces attacked the city of Venice using unmanned balloons carrying explosives, although most of them missed their target (VYAS, 2020).

The recreational remote control (RC) airplanes gained popularity around 1960 with the developments in transistor technology, which lowered the costs in production, creating a market niche and accelerating research in the area (VYAS, 2020). Nowadays, drones are widely used in surveillance, monitoring, identification, photography, and transport in military, recreational, and commercial operations, distinguished by their efficiency in performing tasks that require phenomena understanding and quantifying at any location (FOURTANÉ, 2018).

Given that it is a technology in development, hardware and software innovations emerge quickly and the details about its implementation are being improved constantly. Since it is essential that the system operates satisfactorily amidst faults, the object of study of this project, MPC stands out for its efficiency in multivariable processes with constraints and dead times, using a prediction horizon of future outputs obtained from the system dynamics model to calculate the next control action in real-time, keeping the variables within specified intervals (MATHWORKS, 2018). Thus, the design, implementation, and development of new models and refined control techniques seek to detect and diagnose risk situations, reducing the occurrence of damage to equipment and the threat to public and environmental safety.

As previously mentioned, an important research area is the aircraft flight control, in depth, to find a way to influence the physical system so that its behavior meets previously established performance criteria. Therefore, it is worth researching controlled environments, didactic plants, or even virtual systems, in order to better study the drone structure, performance, and requirements. The first solution is indoor labs. The second solution intends to represent the operation of the system in a simple and objective prototype, facilitating the analysis of some specific aspects. Finally, the third approach is about software simulations.

Products from specialized companies may have a high cost of acquisition and main-

tenance, while self-manufactured prototypes can be cheaper and have flexible configurations despite the possible drop in quality. In the end, it must be considered which option is the most suitable for each study. Nonetheless, the mathematical model that represents the system dynamics can be used for simulation to collect data about the system complexity while guaranteeing quick analysis and insights for hardware applications, since developing and testing code on software have low cost and low risk.

In this context, this work uses the concepts and representative models of AERO and 3DOF Hover, two flexible lab experiment platforms from Quanser, as control system processes for simulation purposes. The choice of these systems was due to the existence of a helicopter-type prototype in lab similar to the Quanser AERO, so that the simulations of these systems could be easily adjusted for practical tests.

1.1 GENERAL OBJECTIVE

The motivation for this work was the interest in drones, especially in the area of flight control, so that the problem that was wished to address in this research was the control of this type of aircraft in the event of faults, that is, ways of influencing the behavior of this type of system when its functioning does not correspond to the design.

Again, the use of predictive control on an aircraft flight control consists on finding the optimal control sequence to follow a reference trajectory by predicting the aircraft's movement, in most cases, in real-time. From the system model, the MPC predicts the process output over a time horizon by solving an online optimization problem. The predictive control variant that works with state-space models, the State-Space Model Predictive Control (SSMPC), deals with a time-domain approach where the input and output of the system are related by a set of first-order differential equations. This approach is the base for the Multi-Model Predictive Control (MMPC) proposed.

The fault tolerance is done through the diagnosis and consequent switching to the most appropriate predefined prediction model by an Artificial Neural Network (ANN). It was assumed that the faults occur only from the nominal model, that is, only one fault can occur at a time and it does not interfere with the other. Two neural network structures are studied, the Multilayer Perceptron (MLPNN) and the Fully Connected Cascade (FCCNN), and a comparison with the classical one-model structure is discussed. The obtained results show that the approach with multiple models is promising when it is possible to determine the dynamics models and perform a precise fault diagnosis.

The adoption of Quanser AERO serves the purpose of analyzing the performance of the ANN classification in conjunction with the MPC, so, the chosen simulation frequency was low and the considered faults were not very destructive. In the case of Quanser 3DOF Hover, its use provides a feasibility study, and therefore the adopted simulation frequency was higher and the faults were more destructive. If a very sudden system fault was applied in the previous system, it

would not be possible to control it. But if a minor fault like the one applied in Quanser AERO occurred on Quanser 3DOF Hover, the MPC alone would be able to correct the problem because of its intrinsic characteristic to accommodate actuator faults.

The simulation platform proposed in this project was developed using Python language in the Visual Studio Code editor, with some tests performed in MATLAB. The motive that most influenced the adoption of these programming languages is their continuous use over the years given the capacity of each of them in enabling easy communication between programmer and machine (TIOBE Software BV, 2020; GITHUB, 2020; CASS, 2019). Market pressure and risks prevent the industry from frequently updating the used software and hardware, while there might be inadequacy due to the complexity and peculiarities of a language in the academy. Moreover, the MATLAB environment is very complete and Python has several packages and modules that facilitate its use in the control field.

1.2 SPECIFIC OBJECTIVES

The specific objectives break down the general objective into smaller parts, addressing the many aspects of the problem statement and being seen as the means to achieve it. In this work, they are:

- Study of Quanser AERO and Quanser 3DOF Hover structures, performances, restrictions, and requirements, followed by obtaining of the state-space nominal and fault models;
- Design and implementation of an SSMPC for the Quanser 3DOF Hover system, evaluating the behavior of the closed-loop response in the nominal and fault states;
- Design and implementation of an SSMPC for the Quanser AERO system, evaluating the behavior of the closed-loop response in the nominal and fault states;
- Design and implementation of an MMPC for the Quanser AERO system considering an ideal fault classification algorithm;
- Development of an MLPNN fault identification algorithm for the Quanser AERO system;
- Development of an FCCNN fault identification algorithm for the Quanser AERO system;
- Implementation of an MMPC with the use of neural network fault identification algorithms for the Quanser AERO system.

Seeing that it was intended to understand and quantify the need for a better system representation to guarantee the controller optimal performance, non-feasibility was studied through experiments with the Quanser 3DOF Hover. Then, the strategy with multiple models was applied in a simpler system, in this case, the Quanser AERO.

1.3 TEXT STRUCTURE

This work is organized into seven chapters: Introduction, Model predictive control, Artificial neural networks, Didactic aerospace systems, Controller design, Test and validation, and Conclusions. These chapters are structured according to the description below:

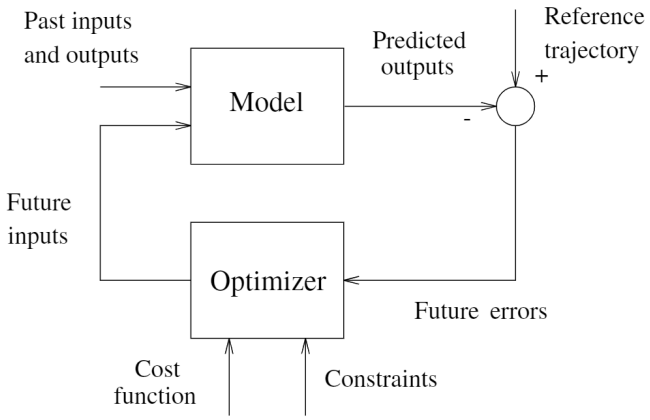
- Chapter 1: Brief contextualization about the control strategy and case study, followed by the introduction of the motivations and objectives of this work;
- Chapter 2: Discussion on the topic of MPC, including a review of its history, theory, mathematical deduction, and algorithm formulation, focusing on the state-space and multi-model strategies;
- Chapter 3: Discussion on the topic of ANN, explaining its use on fault diagnosis and fault tolerance, the central theme of this project, and providing an overview of the proposed ANN techniques;
- Chapter 4: Introduction of the studied aerospace didactic plants, that is, Quanser AERO and Quanser 3DOF Hover, also presenting their mathematical representations and open-loop analysis;
- Chapter 5: Controllers and ANNs step-by-step design;
- Chapter 6: Closed-loop simulation results, containing a detailed description of each proposed case study;
- Chapter 7: Conclusions and final considerations in addition to suggestions for continuing the research.

2 MODEL PREDICTIVE CONTROL

When analyzing a system, it is desired to find out the cause and effect relationships that exist between its input and output. Therefore, the purpose of a controller is to impose a behavior on the system by making its output signal achieve a certain value or follow a reference trajectory (MAYA; FABRIZIO, 2011). In an open-loop control system, the control action is detached from the other variables, while, in a closed-loop, or feedback control, it operates based on the error between the measured signals.

MPC is a set of feedback control methods with the ability to predict the future behavior of a system based on the system model and, consequently, the idea of choosing the control action as the solution of an optimization problem. Its operating principle can be seen in Figure 1. This strategy depends on the effectiveness with which the mathematical model represents the real behavior of the system and the cost function balance (ROSSITER, 2003).

Figure 1 – An MPC block diagram.



Source: Camacho and Bordons (2003).

As previously stated, some major applications take place in chemical plants and oil refineries, but MPC has also been widely used in other industries, like automotive, aerospace, and power electronics, thanks to its ability to handle MIMO systems and constraints. Maciejowski (2000) and Camacho and Bordons (2003) give a detailed description of various MPC strategies in their respective books. Zhang, Wu and Gao (2017) review recent developments in the use of SSMPC for advanced process operation. And MathWorks (2008) even presents a tutorial about the DMC.

This work is focused on the development of a fault-tolerant multi-model predictive control expanded from a SSMPC with constraint treatment, and, to that end, this chapter brings an overview about these control techniques. Details about fault-detection and diagnosis systems will be covered later on.

2.1 NOTIONS OF PREDICTIVE CONTROL

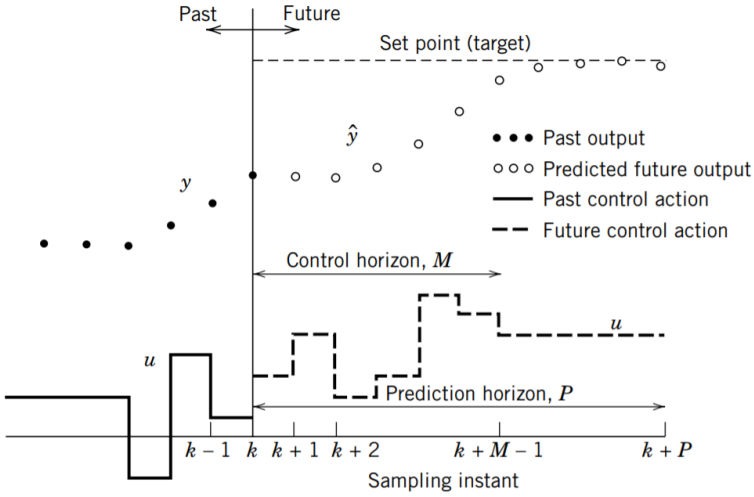
Predictive control, also referred to as Moving Horizon Control (MHC) or Receding Horizon Control (RHC), is widely used as an advanced digital control technique. Given that it is actually a class of control methods, there are many different approaches to it depending on the method used. Therefore, the main ideas, or principles, behind it can be summarized as (ORUKPE, 2012):

- Explicit use of the system model to predict the output;
- Moving, or receding, horizon;
- Numerically solving an optimization problem at each time step.

With the use of the process model, MPC can predict its output over a defined time horizon. To follow the reference trajectory efficiently, it is performed an optimization process at each time step in order to find the best control signal that takes the output to the desired value in a small period of time. To properly tune the control loop, the first value of the control law is saved and the system response is analyzed. Then, to restart the procedure, the beginning of the horizon moves to the corresponding time of that value, hence the name moving horizon (MACIEJOWSKI, 2000).

Since an optimization problem is solved at each time step, a faster and more powerful hardware with large memory is required to implement an MPC method when compared to classical control methods such as the PID. Its main advantages are the tuning simplicity, handling of MIMO systems, even when there are interactions between inputs and/or outputs, and constraints (MATHWORKS, 2018). With this in mind, the major elements of MPC are seen in Figure 2.

Figure 2 – A discrete MPC.



Source: Seborg et al. (2016).

The reference is the output value one wants to achieve. At each time step, the controller makes predictions about the future output and the optimization finds the best sequence of control actions to get to the reference. A big time step may prevent the system from identifying disturbances while a small one increases the computational cost of the algorithm, so, its size should be large enough to represent the relevant dynamics. The term receding refers to the fact that the calculation is repeated at each iteration (ROSSITER, 2003).

Both horizons are finite. The prediction horizon N represents how far it is intended to predict the behavior of the system, while the control horizon M represents how many control action increments are applied within the prediction horizon. Usually, the system response tends to be faster with higher values of N and M, thence, small values make the controller less aggressive considering that the cost function parameters are maintained (FAMA et al., 2005). However, predict the future too far ahead may be an unnecessary computational expense, since unforeseen events can make it required to eliminate part of what was calculated.

The prediction model corresponds to the representative mathematical model of the system, which may be, for example, the impulse response or state space, the first being mostly used in the industry and the second by the academic community (ORUKPE, 2012). Despite that, this approach is robust to modeling errors and allows for constraints treatment. Constraints are defined as physical or operational limitations of the process.

Finally, the cost or objective function is a performance evaluation tool for optimal control systems. Therefore, it penalizes the deviation of the predicted control actions from the reference trajectory through an equation that can be quadratic, linear, or non-linear. About the solver, it might be derivative, quadratic programming, or even bio-inspired (CAVALCA, 2019).

2.2 STATE-SPACE MODEL PREDICTIVE CONTROL

A classic formulation of predictive control is the SSMPC, where, as the name suggests, the mathematical model of the physical system is represented as a set of state-space equations. The formulation for the basic MIMO case can be seen on Maciejowski (2000), therefore, this section presents only the extended models used in the case study. For the algorithm to be applicable, it is necessary for the system to be controllable, or at least stabilizable, which means that it should be possible to make a system of equations with independent variables starting from any given point in time. But it can also be observable, so that the states are available for feedback.

2.2.1 State-space representation

A state-space representation is the description of a system as a set of differential equations that correlates input, output, and state variables, providing its dynamics, i.e., it is the overall of the possible configurations of the system (MAYA; FABRIZIO, 2011). The state of a system is defined as the series of information in the instant t of the state vector, in that way, the state space

is the n-dimensional space where the coordinate axes represent the vector components. The state-space model can be applied in linear, non-linear, time-variant, and time-invariant systems, and it is suitable for systems with single or multiple inputs and outputs being possible to include additional dynamics. For a continuous time-invariant linear system, it is written in the form from Equation (1).

$$\begin{cases} \dot{x}(t) = A_c x(t) + B_c u(t) \\ y(t) = C_c x(t) + D_c u(t) \end{cases} \tag{1}$$

where x is the state vector ($x(t) \in \mathbb{R}^n$), y is the output vector ($y(t) \in \mathbb{R}^q$), u is the input or control vector ($u(t) \in \mathbb{R}^p$), A_c is the state matrix ($A_c \in \mathbb{R}^{n \times n}$), B_c is the input matrix ($B_c \in \mathbb{R}^{n \times p}$), C_c is the output matrix ($C_c \in \mathbb{R}^{q \times n}$), and D_c is the feedforward matrix ($D_c \in \mathbb{R}^{q \times p}$). Therefore, n is the number of states, q is the number of outputs, and p is the number of inputs.

This technique can be used in both frequency and time domain, and, applying the Laplace transform under null conditions in a time-invariant linear system, it is possible to relate state space and transfer function (ROWELL, 2002). Another advantage of this method is the possibility to create new states to represent and control additional events considered interesting for the analysis as long as the variables' relationships are known.

2.2.2 Prediction model with input increments

The formulation in terms of input increments Δu is described in details by Camacho and Bordons (2003). Assuming a linearized discrete-time state-space system in the form presented in Equation (2) from Equation (1):

$$\begin{cases} x(k+1) = A_d x(k) + B_d u(k) \\ y(k) = C_d x(k) \end{cases} \tag{2}$$

and the extended model seen in Equation (3):

$$\Xi(k) = \begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix} \tag{3}$$

being,

$$\Delta u(k+1) = u(k) - u(k-1) \tag{4}$$

one arrives at Equation (5):

$$\begin{cases} \Xi(k+1) = A_\xi \Xi(k) + B_\xi \Delta u(k) \\ y(k) = C_\xi \Xi(k) \end{cases} \tag{5}$$

where,

$$A_\xi = \begin{bmatrix} A_d & B_d \\ O_{(p\times n)} & I_{(p\times p)} \end{bmatrix}, B_\xi = \begin{bmatrix} B_d \\ I_{(p\times p)} \end{bmatrix}, C_\xi = \begin{bmatrix} C & O_{(q\times p)} \end{bmatrix} \tag{6}$$

so that $A_d \in \mathbb{R}^{n\times n}$, $B_d \in \mathbb{R}^{n\times p}$, $C_d \in \mathbb{R}^{q\times n}$, $A_\xi \in \mathbb{R}^{(n+q)\times(n+q)}$, $B_\xi \in \mathbb{R}^{(n+q)\times p}$, and $C_\xi \in \mathbb{R}^{q\times(n+q)}$. Furthermore, I and O are, respectively, identity and zero matrices of corresponding order in subscript.

Hence Equation (7):

$$Y = G\Delta U + F_\xi \tag{7}$$

where, G is the dynamic matrix and F_ξ is the free response matrix. Given that F is described by:

$$F_\xi = \phi\Xi \tag{8}$$

the matrices G and ϕ are defined as:

$$G = \begin{bmatrix} C_\xi B_\xi & 0 & 0 & \cdots & 0 \\ C_\xi A_\xi B_\xi & C_\xi B_\xi & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C_\xi A_\xi^{N-1} B_\xi & C_\xi A_\xi^{N-2} B_\xi & C_\xi A_\xi^{N-3} B_\xi & \cdots & C_\xi A_\xi^{N-M} B_\xi \end{bmatrix}, \phi = \begin{bmatrix} C_\xi A_\xi \\ C_\xi A_\xi^2 \\ \vdots \\ C_\xi A_\xi^N \end{bmatrix} \tag{9}$$

However, it is worth noting that the discretization to be dealt with here is just the combination of a continuous system and converters and not the discrete plant.

2.2.3 Prediction model with integral control action

Here it is going to be explored the method in terms of difference-differential equations utilized by Negri (2014). Assuming the same linearized discrete-time state-space system from Equation (2) and the extended model as seen in Equation (10):

$$\Xi(k) = \begin{bmatrix} \Delta x(k) \\ y(k) \end{bmatrix} \tag{10}$$

being,

$$\Delta x(k+1) = A_d \Delta x(k) + B_d \Delta u(k) \tag{11}$$

one arrives at Equation (12):

$$\begin{cases} \Xi(k+1) = A_\xi \Xi(k) + B_\xi \Delta u(k) \\ y(k) = C_\xi \Xi(k) \end{cases} \tag{12}$$

where,

$$A_\xi = \begin{bmatrix} A_d & O_{(n\times q)} \\ C_d A_d & I_{(q\times q)} \end{bmatrix}, B_\xi = \begin{bmatrix} B_d \\ C_d B_d \end{bmatrix}, C_\xi = \begin{bmatrix} O_{(q\times n)} & I_{(q\times q)} \end{bmatrix} \tag{13}$$

so that $A_d \in \mathbb{R}^{n\times n}$, $B_d \in \mathbb{R}^{n\times p}$, $C_d \in \mathbb{R}^{q\times n}$, $A_\xi \in \mathbb{R}^{(n+q)\times(n+q)}$, $B_\xi \in \mathbb{R}^{(n+q)\times p}$, and $C_\xi \in \mathbb{R}^{q\times(n+q)}$. Furthermore, I and O are, respectively, identity and zero matrices of corresponding order in subscript.

From Equation (11), the expected behavior of Δx is:

$$\Delta x(k+i|k) = A_d^i \Delta x(k) + \sum_{j=1}^{i} A_d^{j-1} B_d \Delta u(k+i-j|k) \tag{14}$$

and, similarly:

$$\Xi(k+i|k) = A_\xi^i \Delta\Xi(k) + \sum_{j=1}^{i} A_\xi^{j-1} B_\xi \Delta u(k+i-j|k) \tag{15}$$

From Equation (12), one can expand y until the same instant:

$$y(k+i|k) = C_\xi A_\xi^i \Xi(k) + \sum_{j=1}^{i} C_\xi A_\xi^{j-1} B_\xi \Delta u(k+i-j|k) \tag{16}$$

Obtaining equivalent of Equation (7), where G is the dynamic matrix and F_ξ is the free response matrix. Given that F_ξ is described by Equation (8), the matrices G and ϕ are defined as proposed in Equation (9).

2.2.4 Optimization problem subject to constraints

The optimization problem considering the design parameter matrices Py and Pu as, respectively, $I_{(qN\times qN)}$ and $I_{(pM\times pM)}$, is to minimize Equation (17) subject to Equation (18).

$$J = (Y-R)^T P_y (Y-R) + \Delta U^T P_u \Delta U \tag{17}$$

$$Y = G\Delta U + F_\xi \tag{18}$$

Substituting Equation (18) in Equation (17):

$$J = \Delta U^T (G^T P_y G + P_u)\Delta U + 2\Delta U^T G^T P_y (F_\xi - R) + (F_\xi - R)^T P_y (F_\xi - R) \tag{19}$$

Deriving Equation (19) results in:

$$\frac{dJ}{d\Delta U} = 2(G^T P_y G + P_u)\Delta U + 2G^T P_y (F_\xi - R) \tag{20}$$

Considering ΔU_e stationary in Equation (20), it is obtained:

$$\Delta U_e = -(G^T P_y G + P_u)^{-1} G^T P_y (F_\xi - R) \tag{21}$$

Also from Equation (20), considering $\Delta U^* = \Delta U_e$ minimum point, one has:

$$\frac{d^2 J}{d\Delta U^2} = 2(G^T P_y G + P_u) > 0 \tag{22}$$

Therefore, if $P_u > 0$, then:

$$\Delta U^* = (G^T P_y G + P_u)^{-1} G^T P_y (R - F_\xi) \tag{23}$$

The scope of this work is to guarantee the solution to the optimization problem proposed in Equation (17) subject to Equation (18) in the existence of constraints. Emphasis is placed upon the use of a solver for quadratic objective functions with linear constraints, which finds a minimum for the problem specified as stated in Equation (24):

$$minJ(\Delta U) = \frac{1}{2}\Delta U^T H \Delta U + f^T \tag{24}$$

such that,

$$\begin{aligned} S\Delta U &\leq b, \\ S_{eq}\Delta U &= b_{eq} \\ lb \leq \Delta U &\leq ub \end{aligned} \tag{25}$$

A constraint is a systematic physical or operational limitation, that is, the optimization problem is aware of its existence. Three types of constraints will be considered here (CAMACHO; BORDONS, 2003):

- Slew rate limits of the actuator;
- Limits on the control action range;
- Limits on the output signals range.

Formulated in Equation(26) as:

$$\begin{bmatrix} I_{pM} \\ -I_{pM} \\ T_M^{Ip} \\ -T_M^{Ip} \\ G \\ -G \end{bmatrix} \Delta U \leq \begin{bmatrix} [\Delta u_{max}]_M \\ -[\Delta u_{min}]_M \\ [u_{max} - u(k-1)]_M \\ [u(k-1) - u_{min}]_M \\ [y_{max}]_N - F_\xi \\ F_\xi - [y_{min}]_N \end{bmatrix} \tag{26}$$

where,

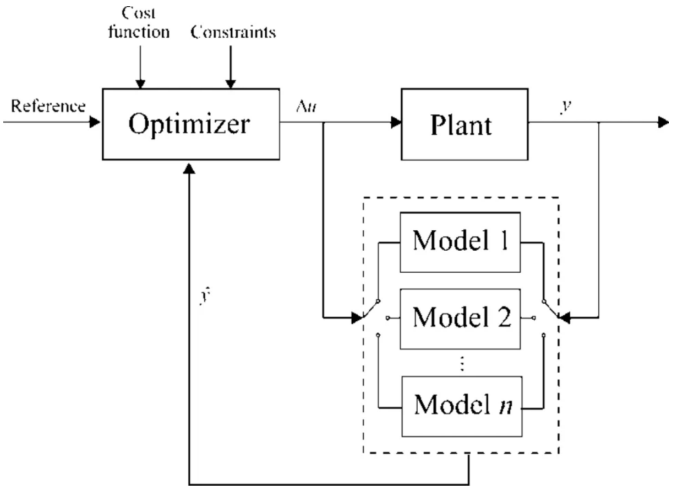
$$\begin{bmatrix} \Delta u_{1,min} \\ \Delta u_{2,min} \\ \vdots \\ \Delta u_{p,min} \end{bmatrix} \leq \Delta u(k) \leq \begin{bmatrix} \Delta u_{1,max} \\ \Delta u_{2,max} \\ \vdots \\ \Delta u_{p,max} \end{bmatrix}, \begin{bmatrix} u_{1,min} \\ u_{2,min} \\ \vdots \\ u_{p,min} \end{bmatrix} \leq u(k) \leq \begin{bmatrix} u_{1,max} \\ u_{2,max} \\ \vdots \\ u_{p,max} \end{bmatrix}, \begin{bmatrix} y_{1,min} \\ y_{2,min} \\ \vdots \\ y_{q,min} \end{bmatrix} \leq \Delta y(k) \leq \begin{bmatrix} y_{1,max} \\ y_{2,max} \\ \vdots \\ y_{q,max} \end{bmatrix} \tag{27}$$

also, I and T are, respectively, identity and toeplitz matrices of corresponding order.

2.3 MULTI-MODEL PREDICTIVE CONTROL

The MMPC, also known as Switched Model Predictive Control, is an adaptive method that relies on the assumption that the process can be represented by a finite number of models, and that a controller may be designed for each one of them (JING; SYAFIIE, 2020). The prediction model is chosen based on the comparison between the measured output signal and the predicted ones, and this information causes the valid model to be used in the following iteration of the feedback loop.

Figure 3 – An MMPC block diagram.



Source: Jing and Syafiie (2020).

There are many different design and implementation approaches to it. Some MMPC algorithms are considered robust due to an incremental realigned model, and others are adaptive since they rely on a model bank. This last configuration is built in a way that each linear model describes the system dynamic behavior over a specified operating condition (JING; SYAFIIE, 2020). Furthermore, there are a couple of methods to detect the system change, from state estimators to ANNs, which are trained to identify the system's nominal and fault states, and switching the prediction model when best suited.

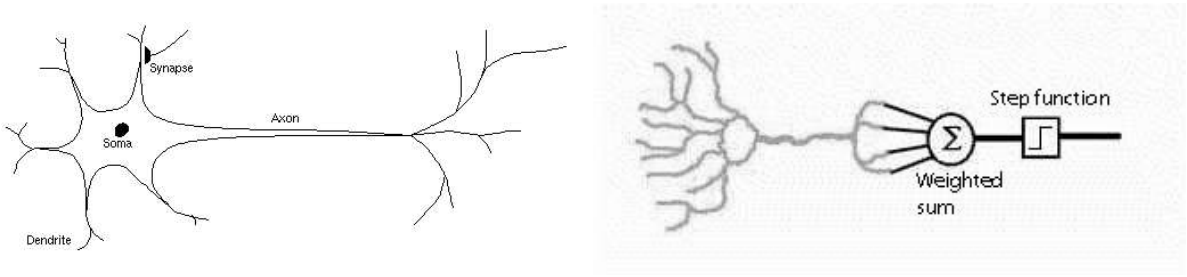
Several works about MMPC can be cited. Franco, Sacone and Parisini (2004) focus on nonlinear systems, with a finite set of discrete-time models representing the system dynamics

in different operating conditions. Hung et al. (2014) also works with nonlinear systems but use Fuzzy model to divide the system into several sub-linear systems. Di Palma and Magni (2004) propose the use of different models for each requested prediction. Zhang, Zhuang and Braatz (2016) deal with the issues of feasibility, stability, and robustness on the MMPC of a class of discrete-time switched linear systems with mode-dependent dwell time.

3 ARTIFICIAL NEURAL NETWORKS

An ANN is a computing model inspired by the human brain structure and behavior, loosely mimicking the thought process through a set of algorithms (KAVLAKOGLU, 2020). It is a subset of machine learning and artificial intelligence widely used in pattern recognition applications, which was first proposed in 1944 by Warren McCullough and Walter Pitts. However, the first trainable ANN was developed by Frank Rosenblatt in 1957, the Perceptron (HARDESTY, 2017). The perceptron itself is a type of Neuron, the basic unit of all ANNs, which is presented in Figure 4.

Figure 4 – Neurons: (a) Biological, and (b) Artificial.



(a) (b)

Source: Clabaugh, Myszewski and Pang (2000).

This technology has been in and out of focus due to machinery processing power, but some major works in the area include those related to object recognition, speech recognition, and time-series forecast (VAN VEEN, 2016). Nowadays, there are many different types of ANN, like Feed Forward (FFNN), Recurrent Neural Network (RNN) and Deep Convolutional Network (DCN), each with its own unique structure and applications. Nevertheless, some of these structures vary only in the activation function applied, the most common being Sigmoid, Tanh, ELU, and ReLU.

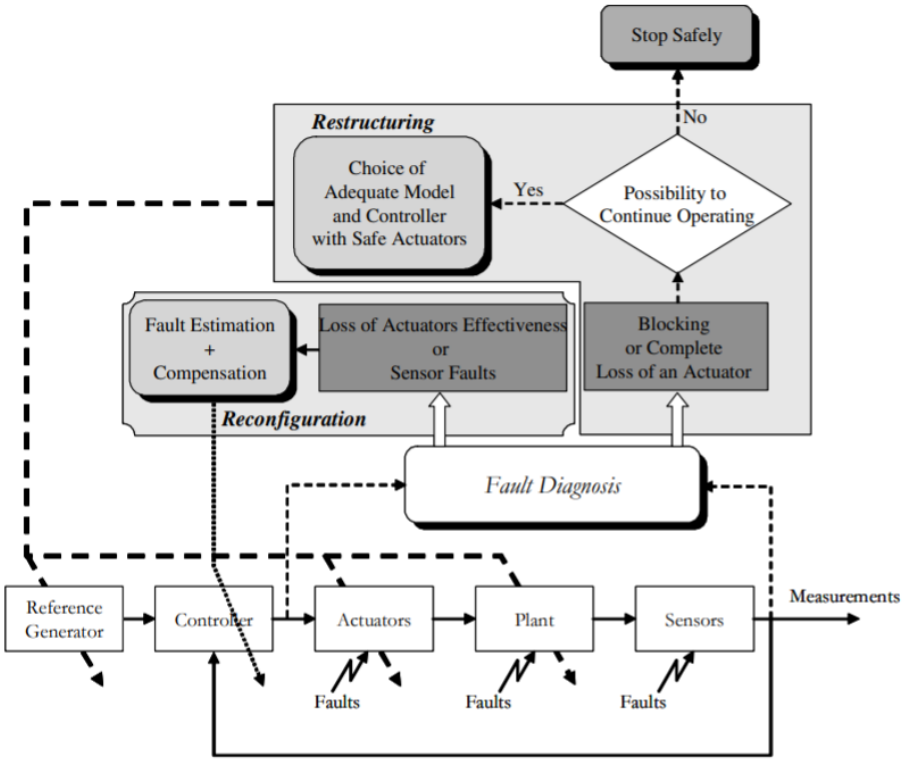
About its use combined with MPC, Piché et al. (1999) present a generic neural network-based technique for developing nonlinear dynamic models from empirical data and their use with predictive control. In Åkesson and Toivonen (2006), a neural network controller is designed by minimizing an MPC type cost function off-line for a set of training data, that is, the control law is represented by a neural network function approximator. Lastly, Popoff (2009) focus on neural predictive control for petrochemical processes, using a FFNN with error backpropagation.

In this work, it was developed an adaptative control to work with abrupt faults, specifically power losses. The adaptative approach does the adjustment according to the behavior of the process upon failure, unlike the robust, which focuses on maintaining satisfactory performance regardless of it. Two ANN approaches are studied, the MLPNN and FCCNN, with ReLU and Sigmoid activation functions, to identify system faults based on the process representative mathematical models. To that end, fault-detection and diagnosis are briefly introduced in the next section.

3.1 FAULT-DETECTION AND DIAGNOSIS SYSTEMS

A Fault-Tolerant Control (FTC) system is designed with the purpose of ensuring reliable operation in the event of faults, failures, or malfunctions. Since the process cannot always work perfectly, fault-tolerance enables the system to continue operating properly under non-ideal circumstances, despite not guaranteeing optimal performance (NOURA et al., 2009). Therefore, the fault-diagnosis scheme can be summarized in detection, isolation, analysis, and maintenance, in order to determine the type, causes, location, and extent of the problem (PEREIRA, 2019), which can be seen in Figure 5.

Figure 5 – An FTC scheme.



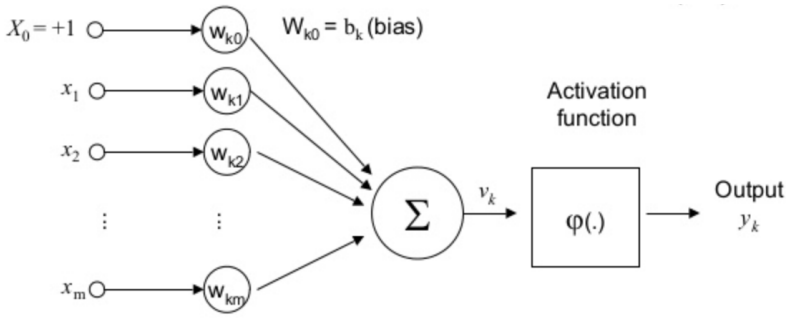
Source: Noura et al. (2009).

Some of the problems that can occur in a control process are disturbances, noises, and faults. The first ones are internal or external signals that affect the system output and are usually low frequency. Noises are similar to disturbances but are high-frequency signals. Lastly, faults are improper functioning or changes in the structure or parameters of the system about the nominal condition. Faults are classified concerning their form, time behavior, and extent. The form can be systematic or random, the time behavior can be abrupt, incipient or intermittent, and the extent can be local or global (ISERMANN, 2006).

3.2 OVERVIEW OF NEURAL NETWORKS

A basic neural network cell is connected to neurons via synaptic weights. The value of each cell is multiplied by its weight and the resulting values are added altogether with a bias, that is then passed through an activation function and the resulting value becomes the value of the cell (VAN VEEN, 2017). Meanwhile, a layer is a way of connecting neurons, consisting of either input, hidden, or output cells connected. It is important to remember that there are other types of cells like convolutional and recurrent, for example. As seen in Figure 6, the ANN main components are its inputs, weights, bias, and outputs.

Figure 6 – An ANN scheme.



Source: Data Science Academy (2019).

Usually, the ANN input is a vector or matrix that features values of a sample of external data with the necessary information to analyze the process. The output, on the other hand, needs to accomplish the ANN task, which may be something from binary classification to recognizing an object. The number of hidden neurons is defined more empirically, but a common starting point is an average value between the size of the input layer and the size of the output layer neurons, even though this number should be adjusted accordingly to the demands of the ANN later on (SHEELA; DEEPA, 2013).

In the search for the minimum error, a balance must be achieved between bias and variance. Bias is the error between the average model prediction and the expected result, while variance is the average fluctuation in the model prediction for the dataset (ANWAR, 2020). High bias implies high error on test and data train due to an overly-simplified model or underfitting. High variance implies high error on the test but low on the train due to an overly complex model or overfitting. It is common practice to initialize bias and weights randomly.

Equation 28 presents its algebraic formulation according to the scheme from Figure 6:

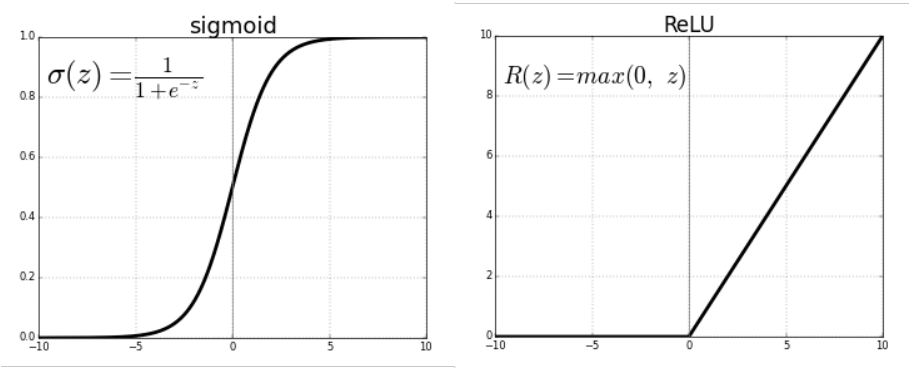
$$\begin{aligned} \upsilon_k &= \sum_{j-1}^{n} \omega_{kj} \chi_j \\ \Upsilon_k &= \varphi(\upsilon_k) \end{aligned} \tag{28}$$

where χ are the input signals, ω are the weights, Υ are the output signals, and φ is the activation function (DATA SCIENCE ACADEMY, 2019). Each input is multiplied by its respective

synaptic weight, which has the effect of inhibition or excitation on the activation of input signals. This sum generates the activation potential or υ. The bias is a polarization parameter that is included in order to increase the degree of freedom of this sum, allowing a neuron to return non-null output even if all its inputs are null. The activation function regulates the output of the network and is responsible for limiting the sum to a certain interval, generating the final output of the neuron.

Figure 7 shows the two main activation functions used in this work, that is, sigmoid and ReLU. Sigmoid is a non-linear function that transforms the input values into the range between 0 and 1, and ReLU, also a non-linear function, is defined as the positive part of its argument. Both are used to classify the output as a "yes" or "no", still, ReLU is aggregated to approximate a function, while sigmoid is mostly applied in cases where it is necessary to predict a probability (SHARMA, 2017).

Figure 7 – Sigmoid and ReLU activation functions.



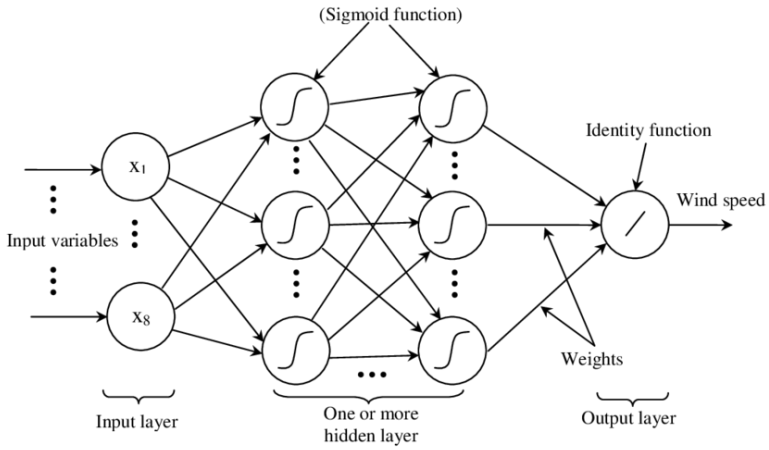
Source: Sharma (2017).

There are many metrics for evaluating classification models, but this work focus on the accuracy (CHOLLET et al., 2015). Accuracy refers to the number of times the predictions were correct, that is, equal labels. It is worth mentioning that the loss is calculated on training and validation checking how well the model is doing for these sets. The purpose of the loss function is to calculate the quantity that the model should try to minimize during training.

3.3 MULTILAYER PERCEPTRON NEURAL NETWORK

MLPNN is a type of FFNN that feeds the information from the input layer to the output layer not forming a cycle. The input layer receives the data or input signal, then passes the information to the hidden layers that perform the computational mechanism, and, in the end, the output layer returns the response. However, the model can also be trained through backpropagation, an algorithm for computing the gradient of the loss function with respect to the weights of the ANN, improving its accuracy (DATA SCIENCE ACADEMY, 2019). This configuration is the most common when dealing with ANNs, being widely used for solving nonlinear problems, and it can be seen in Figure 8

Figure 8 – An MLPNN scheme.



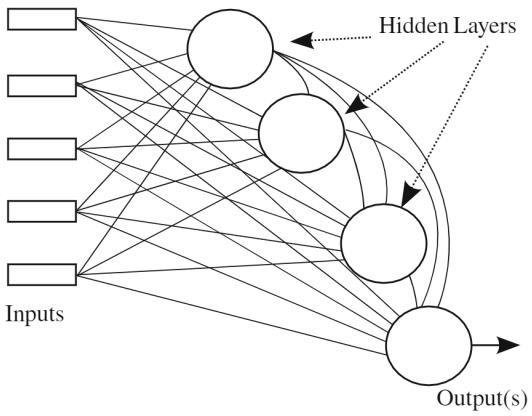
Source: Palaniappan (2018).

Some recent developments in MLPNN are seen on Delashmit and Manry (2021). Araujo et al. (2010) uses it for flow prediction, forecasting days ahead of the average and maximum daily flow of a river in a small forest headwaters. Sonawane and Patil (2015) presents a prediction system for heart disease with great accuracy. Furthermore, an introduction to this strategy with TensorFlow's API is done by Li (2019).

3.4 FULLY CONNECTED CASCADE NEURAL NETWORK

In the FCCNN, all hidden layers have only one neuron and all neurons are connected to all subsequent layers, which supposedly reduces the number of cells and computational expense when compared to the MLPNN algorithm (NEGRI, 2020). This type of ANN is widely used for data classification, but it does not work well in image recognition and classification.

Figure 9 – An FCCNN scheme.



Source: Negri et al. (2017).

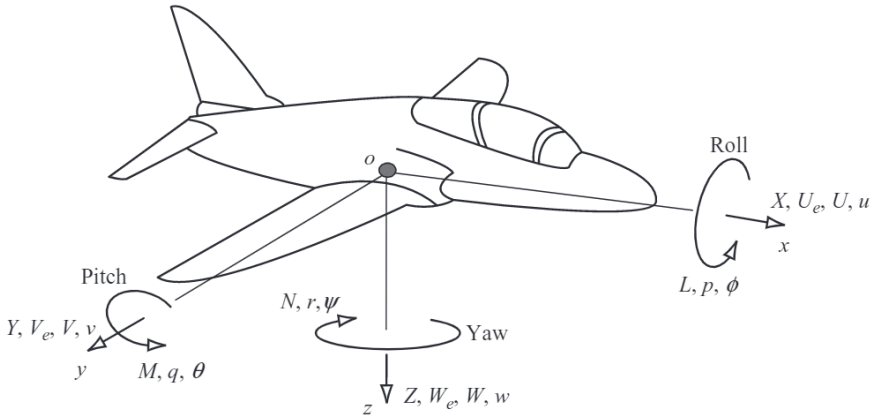
About the works with FCCNN, Cheng (2017) derive and describe in detail an efficient backpropagation algorithm for computing the gradient for this class of ANNs. A fast cascade neural network is proposed by Qiao et al. (2015), whose performance is evaluated on artificial and real-world benchmark problems. Hussain, Mokhtar and Howe (2013) study aircraft sensor and actuator failures, combining a neuron by neuron algorithm and this strategy.

4 DIDACTIC AEROSPACE SYSTEMS

As mentioned earlier, this dissertation thesis focuses on simulations only, mostly due to the limited time and resources available. To eventually design and implement an FTC, it is necessary to obtain the system nominal and fault model representations, that is, to describe the causal relationship between the input and the output of the system considering a variety of situations, which may be external disturbances or output measurement noises. For such study, it is proposed the use of AERO and 3DOF Hover, two lab solutions developed by Quanser for the study of modern flight systems theory and application. It is worth mentioning that Quanser offers a variety of didactic plants that can be used for teaching and research, focusing on control, robotics, and mechatronics (QUANSER, 2017b).

The analysis of the system mostly includes its structure, performance, restrictions, and requirements, i.e., the development of models that faithfully portray the system and its singularities. The validation of such models is usually done through simulation and bench testing. That way, it is possible to compare the system's actual performance with the standards in order to determine the need for changes and corrections to make the system behavior meet the project specifications. The system of axes and notations from Figure 10 are used to represent aircraft generalized movement, but this is specific to each case depending on the prototype and working conditions (COOK, 2011).

Figure 10 – Aircraft moving axes.



Source: Cook (2011).

Several works explain in detail the modeling and some approaches of MPC for aircraft flight control. In Matos (2008), an MPC with fault accommodation for a quadcopter is presented, using the multiple-model strategy. The application of MPC in helicopters with three degrees of freedom (DOF) is seen on Maia (2008), Pascoal (2010), and Afonso (2012), with the first two focusing on robust MPC and the third dealing with tolerant MPC for actuator failures. Casara (2015) controls a helicopter with two DOF with PI and LQR techniques, while Nascimento (2016) continues the studies with the same plant but applies an SSMPC.

4.1 QUANSER AERO

The Quanser AERO, seen in Figure 11, is a dual-rotor helicopter system basically composed of a central bar with a rotor at each end, best described as a reconfigurable aerospace experiment since it can operate in conventional or tandem (half-quadrotor) helicopter mode, depending on the position of the tail rotor (QUANSER, 2017b). The main or front rotor is set up horizontal to the ground and is mainly responsible for the pitch movement, while the back or tail rotor is vertically laid out and mostly influences the yaw movement. Since it reduces the torque generated by the main rotor on the yaw axis, helping to stabilize it, the tail rotor also acts as an anti-torque rotor (QUANSER, 2017b).

Figure 11 – Quanser AERO didactic plant.

Source: Quanser (2017b).

The system physical specifications, provided by the company, can be seen in Table 1.

Table 1 – Quanser AERO specifications from a post-production unit.

Device specification	
Base width	17.8cm
Base height	17.8cm
Base depth	7cm
Device height	35.6cm
Device length	51cm
Device mass	3.6kg
Propeller diameter	12.7cm
Yaw angle range	360°
Pitch angle range	124° (±62°)
Recommended output voltage range	$\pm 18V$
Maximum output voltage range	$\pm 24V$

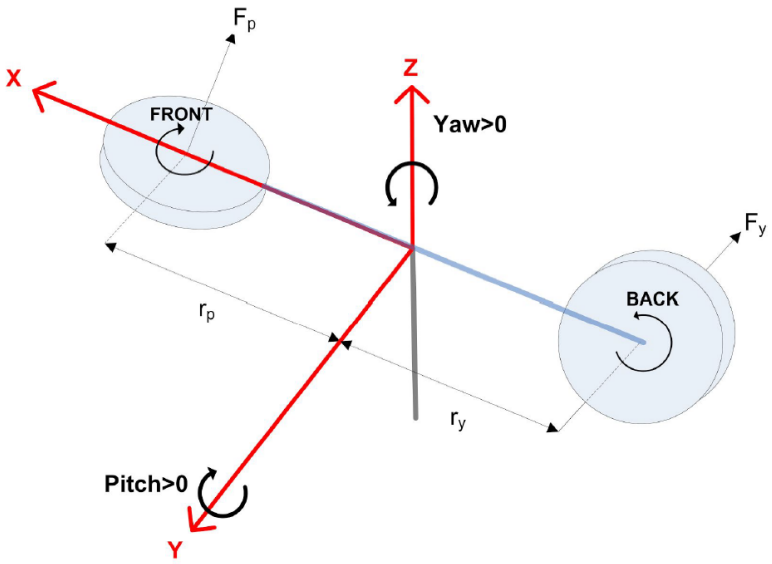
Source: Adapted from Quanser (2017b).

4.1.1 System representation

The system mathematical representation describes the equations that rule the process based on the relationship between its inputs and outputs signals. Similar systems were modeled in detail by Tastemirov, Lecchini-Visintini and Morales-Viviescas (2017) and Oliveira Jr (2018), but here it is going to be followed the process presented by Quanser (2017b). Quanser AERO is a MIMO system with two DOF, the voltage from the two rotors are its two inputs and the pitch and yaw angle are its outputs.

The pitch angle is considered to be zero when the Quanser AERO is horizontal and parallel to the ground, increasing positively when the front rotor moves upwards and the body rotates counterclockwise on the Y axis, while the yaw angle increases positively when the body rotates counterclockwise on the Z axis (QUANSER, 2017b). In the free body diagram from Figure 12, the system is seen as a bar and two cylinders.

Figure 12 – Simple free body diagram of Quanser AERO.



Source: Quanser (2017b).

The equations of motion and torque for each axis are defined, as well as the total moment of inertia. Applying the Laplace transform to the equations of motion and assuming zero initial conditions, the transfer functions that relate the inputs and outputs are obtained. Given the format of state-space equations, it can be defined the states, outputs, and control variables of the system, to then derive the matrices from the equations of motion.

The state, output, and input vectors are defined as follow:

$$x^T(t) = [\theta(t), \psi(t), \dot{\theta}(t), \dot{\psi}(t)] \tag{29}$$

$$y^T(t) = [\theta(t), \psi(t)] \tag{30}$$

$$u^T(t) = [V_p(t), V_y(t)] \tag{31}$$

where θ is the pitch angle, ψ is the yaw angle, V_p is the motor voltage applied to the pitch rotor, and V_y is the motor voltage applied to the yaw rotor.

The state-space matrices are:

$$A_c = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{K_{sp}}{J_p} & 0 & -\frac{D_p}{J_p} & 0 \\ 0 & 0 & 0 & -\frac{D_y}{J_y} \end{bmatrix} \tag{32}$$

$$B_c = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{K_{pp}}{J_p} & \frac{K_{py}}{J_p} \\ \frac{K_{yp}}{J_y} & \frac{K_{yy}}{J_y} \end{bmatrix} \tag{33}$$

$$C_c = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{34}$$

$$D_c = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{35}$$

whose variables, and their respective values, are seen in Table 2.

Table 2 – Quanser AERO parameters from a post-production unit.

Symbol	Description	Value
Dp	Pitch damping	0.0071116 N.m.s/rad
Dy	Yaw damping	0.0220 N.m.s/rad
Kpp	Pitch thrust gain	0.0011 N.m/V
Kyy	Yaw thrust gain	0.0022 N.m/V
Kpy	Pitch cross-torque from yaw voltage	0.0021 N.m/V
Kyp	Yaw cross-torque from pitch voltage	-0.0027 N.m/V
Ksp	Stiffness about the pitch axis	0.037463 N.m/rad
Jp	Moment of inertia about pitch axis	0.0218869 kg.m^2
Jy	Moment of inertia about yaw axis	0.0219921 kg.m^2

Source: Adapted from Quanser (2017b).

The damping coefficients are obtained through the free response of the system for each axis. Then, the analysis of the equations of motion for a DOF is made, the Laplace transform is

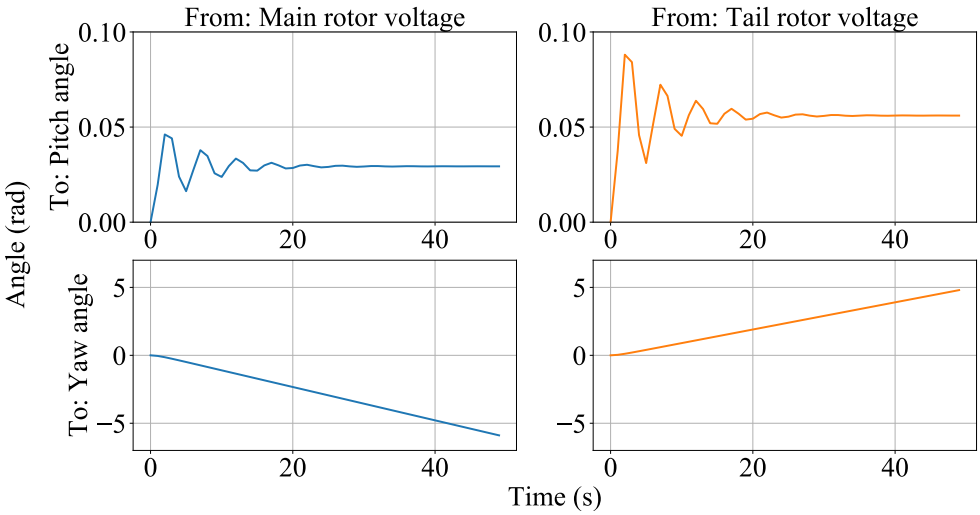
applied and null initial conditions are assumed. Finally, the desired parameters are isolated. The same is done for the impulse parameters, however, the cross torque ones need to be evaluated considering both axes unlocked. Again, one must find the equations of motion in terms of angular velocity and isolate the desired parameters.

4.1.2 Model analysis

Three representative models of the functioning of the Quanser AERO were analyzed. The first model replicates the nominal mode of the system, that is, without fault and deterministic. The second and third models, on the other hand, are characterized by 50% power loss faults in the main and tail rotor, respectively. It is important to note that it is not possible to cover all of the system characteristics and functionalities with a simplified model, but it is unnecessary to test every single condition in most study cases.

The first model studied represents the nominal state. Figure 13 shows its open-loop step response. It is analyzed one rotor switched on at a time, therefore, the subfigures on the first column represent the situation where the main rotor has a positive voltage, while, it is the tail rotor that has a positive voltage in the subfigures on the second column. Also, the pitch angle response is seen on the first row and the yaw angle response is seen on the second row.

Figure 13 – Open-loop step response of Quanser AERO.

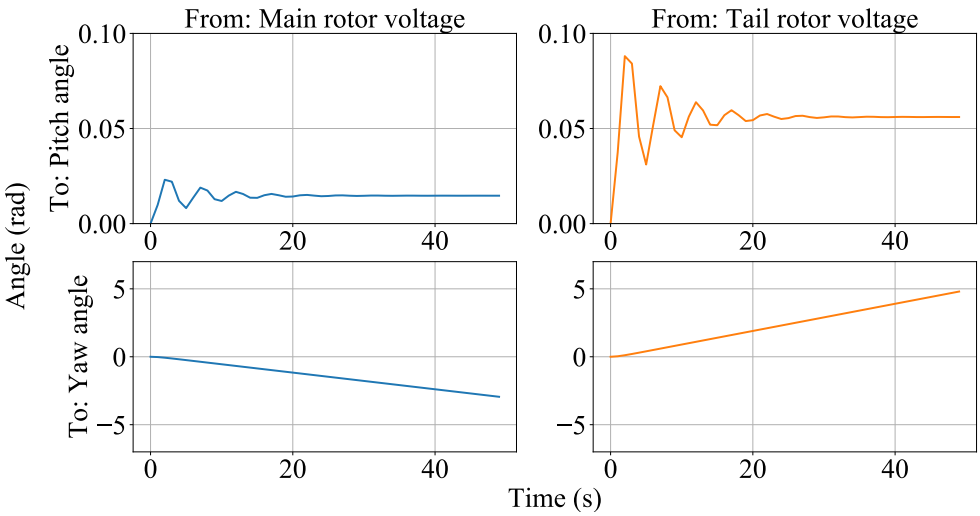


Source: Author (2021).

In this case, it can be observed that the pitch angle response is underdamped with settling time around 30s when any rotor voltage is positive. In the subfigure on the top left, the pitch angle maximum value stays below 0.05rad, but it goes to almost 0.1rad in the subfigure on the top right. On the other hand, the yaw angle goes from 0rad to -5rad in an interval of 40s in the subfigure on the bottom left, but it goes from 0rad to almost 5rad in the subfigure on the bottom right. Nonetheless, it can be said that the yaw angle value decreases when the main rotor voltage is positive and increases when the tail rotor voltage is positive.

The second model considers a fault of 50% power loss in the main rotor and its step response can be seen in Figure 14.

Figure 14 – Open-loop step response of Quanser AERO with main rotor fault.

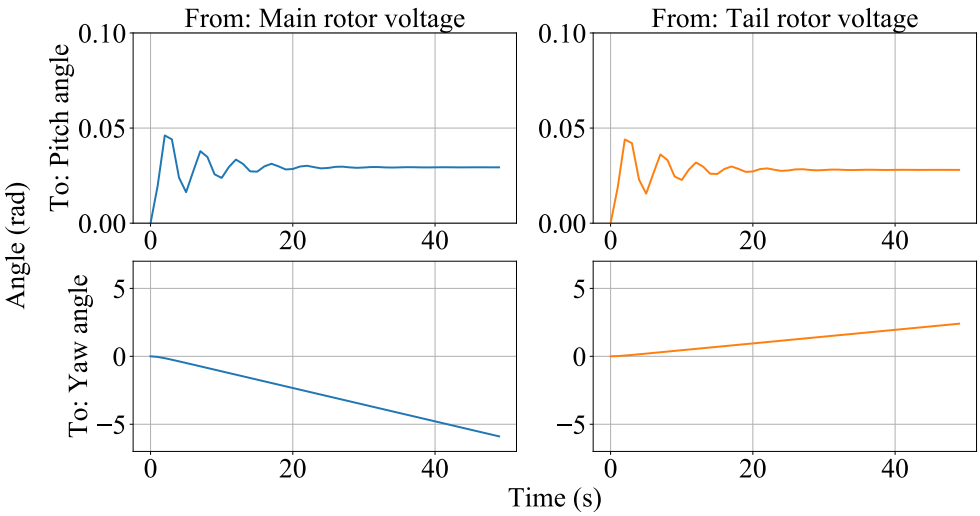


Source: Author (2021).

Here, as expected, the main rotor voltage has half the impact on the pitch and yaw angle responses, but the waveform of both graphs remained. That is, the values of the angles in both subfigures on the left are half of the values found in the no-fault situation seen above, respectively, a peak value around 0.025rad for the pitch angle and a drop from 0rad to 2.5rad for the yaw angle in the same 40s range.

The third model considers a fault of 50% power loss in the tail rotor and its step response can be seen in Figure 15.

Figure 15 – Open-loop step response of Quanser AERO with tail rotor fault.



Source: Author (2021).

Here, as expected, the tail rotor voltage has half the impact on the pitch and yaw angle responses, but the waveform of both graphs remained. That is, the values of the angles in both subfigures on the right are half of the values found in the nominal situation seen above, respectively, a peak value around 0.05rad for the pitch angle and a rise from 0rad to 2.5rad for the yaw angle in the same 40s range.

Analyzing all three step responses, the settling time (T_s) and simulation frequency (f_s) are defined as, respectively, 30s and $10s^{-1}$ for both systems. That said, the sample time is calculated as $\frac{T_s}{40}$.

4.2 QUANSER 3-DOF HOVER

Meanwhile, the Quanser 3DOF Hover, seen in Figure 16, is a quadcopter system consisting of a planar round frame with four propellers driven by motors (QUANSER, 2017a). When a positive voltage is applied to a rotor, a thrusting force is generated causing an elevation of the powertrain. In a controlled environment, with the thrust of the four propellers approximately equal, the total torque in the system is balanced since two of the propellers are counter-rotating (QUANSER, 2017a).

Figure 16 – Quanser 3DOF Hover didactic plant.

Source: Quanser (2017a).

The system physical specification, provided by the company, can be seen in Table 3.

Table 3 – Quanser 3DOF Hover specifications from a pre-production unit.

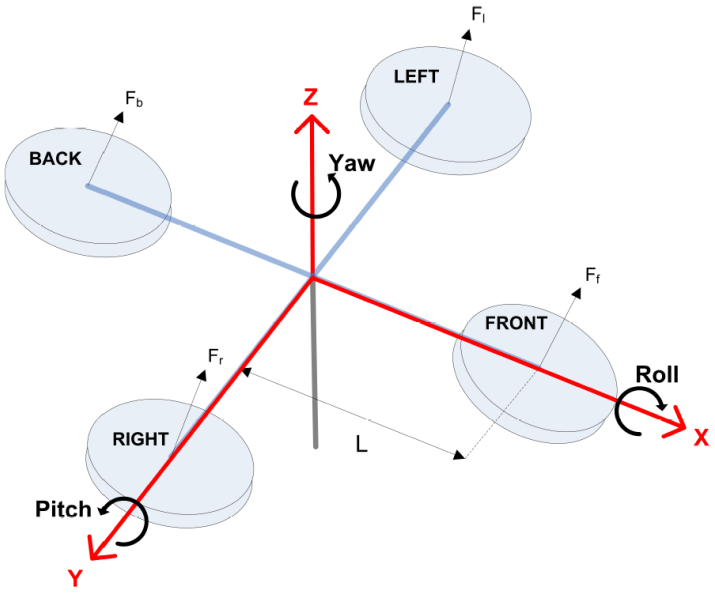
Device specifiation	
Base width	17.5cm
Base length	17.5cm
Device height	45cm
Device mass	3.46kg
Quadcopter length	48cm
Quadcopter mass	1.39kg
Propeller diameter	20.3cm
Propeller pitch	15.2cm
Yaw angle range	360°
Pitch angle range	75° (±37.5°)
Maximum output voltage range	24V

Source: Adapted from Quanser (2017a).

4.2.1 System representation

About the system mathematical representation, similar systems were modeled in detail by Ijuim (2019) and Alves (2012), but here it is going to be followed the process presented by Quanser (2017a). Quanser 3DOF Hover is a MIMO system with three DOF, the voltage from the four rotors are its four inputs and the yaw, pitch, and roll angles are its outputs. The body rotates about pitch and yaw axes. In the free body diagram from Figure 17, the system is seen as two bars and four cylinders.

Figure 17 – Simple free body diagram of Quanser 3DOF Hover.



Source: Quanser (2017a).

Lagrange's method is used to obtain the dynamic model of the system. But, the dynamics

for the pitch and roll axis can be described by a general equation that relates a differential thrust-force and the distance of the propeller motor and the pivot on the axis, which is rewritten in terms of voltage. As for the motion around the yaw axis, it is caused by the difference in torques exerted by the propellers.

The state, output, and input vectors are defined as follow:

$$x^T(t) = [\psi(t), \theta(t), \phi(t), \dot{\psi}(t), \dot{\theta}(t), \dot{\phi}(t)] \tag{36}$$

$$y^T(t) = [\psi(t), \theta(t), \phi(t)] \tag{37}$$

$$u^T(t) = [V_f(t), V_b(t), V_r(t), V_l(t)] \tag{38}$$

where ψ is the yaw angle, θ is the pitch angle, ϕ is the roll angle, V_f is the motor voltage applied to the front rotor, and V_b is the motor voltage applied to the back rotor, V_r is the motor voltage applied to the right rotor, and V_l is the motor voltage applied to the left rotor.

The state-space matrices are:

$$A_c = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{39}$$

$$B_c = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{K_t}{J_y} & -\frac{K_t}{J_y} & \frac{K_t}{J_y} & \frac{K_t}{J_y} \\ \frac{LK_f}{J_p} & -\frac{LK_f}{J_p} & 0 & 0 \\ 0 & 0 & \frac{LK_f}{J_r} & -\frac{LK_f}{J_r} \end{bmatrix} \tag{40}$$

$$C_c = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{41}$$

$$D_c = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{42}$$

whose variables, with their respective values, are seen in Table 4.

Table 4 – Quanser 3DOF Hover parameters from a pre-production unit.

Symbol	Description	Value
Kt	Propeller force-thrust constant	0.0036 N.m/V
Kf	Propeller torque thrust constant	0.1188 N.m/V
L	Distance from propellers to the central axis	0.197 m
Jp	Moment of inertia about pitch axis	0.110 kg.m^2
Jy	Moment of inertia about yaw axis	0.0552 kg.m^2
Jr	Moment of inertia about roll axis	0.0552 kg.m^2

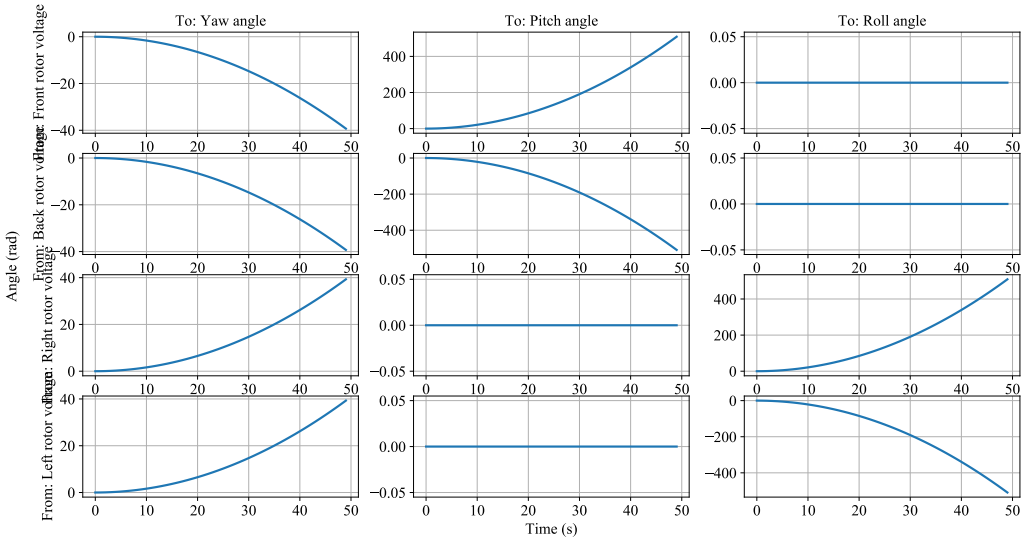
Source: Adapted from Quanser (2017b).

4.2.2 Model analysis

Again, three representative models of the functioning of the Quanser 3DOF Hover were analyzed. The first model replicates the nominal mode of the system, that is, without faults and deterministic. The second and third models, on the other hand, are characterized by 90% power loss faults in the front and back rotor, respectively.

The first model studied represents the nominal state. Figure 18 shows its open-loop step response. It is analyzed one rotor switched on at a time, therefore, the subfigures on the first row represent the situation where the front rotor has a positive voltage, the subfigures on the second row represent the situation where the back rotor has a positive voltage, the subfigures on the third row represent the situation where the right rotor has a positive voltage, and the subfigures on the fourth row represent the situation where the left rotor has a positive voltage. Also, the yaw angle response is seen on the first column, the pitch angle response is seen on the second column, and the roll angle response is seen on the third column.

Figure 18 – Open-loop step response of Quanser 3DOF Hover.

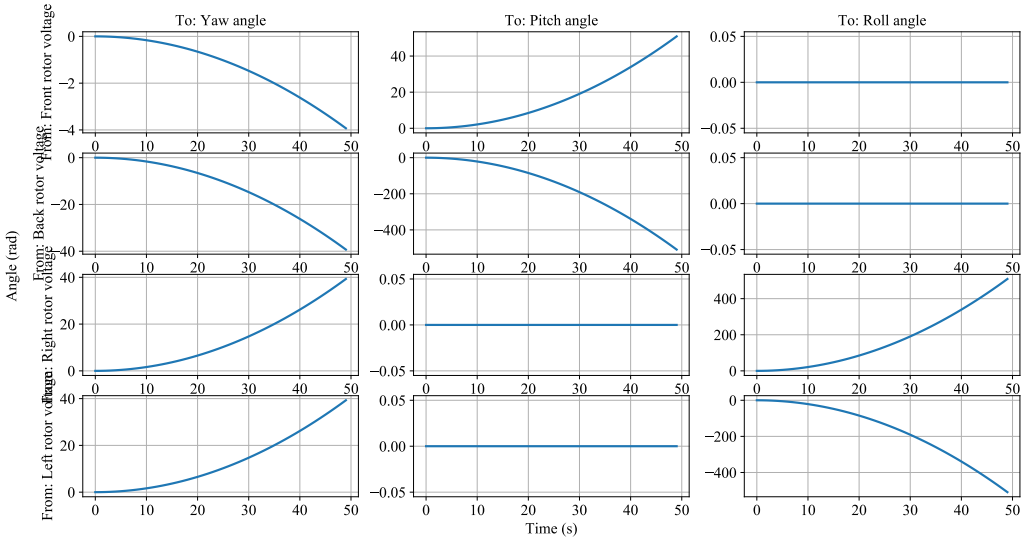


Source: Author (2021).

It is observed that the frontal rotor has a negative influence over the yaw angle and a positive one on the pitch, not acting on the roll. The back rotor has a negative influence over the yaw and pitch angles but does not act on the roll. The right rotor has a positive influence over the yaw and roll angles but does not act on the pitch. Finally, the left rotor has a positive influence over the yaw angle and a negative one on the roll, not acting on the pitch.

The second model considers a fault of 90% power loss in the front rotor and its step response can be seen in Figure 19.

Figure 19 – Open-loop step response of Quanser 3DOF Hover with front rotor fault.

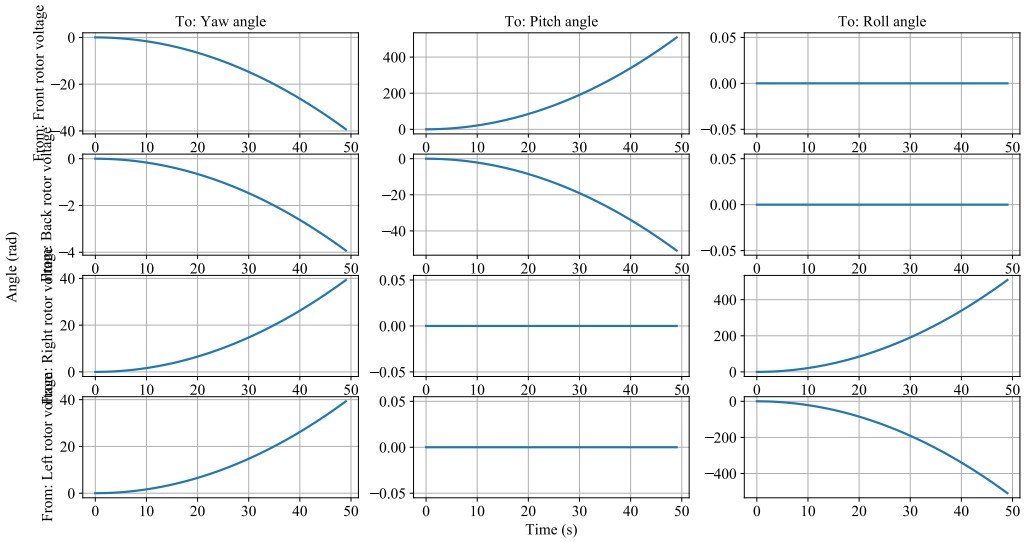


Source: Author (2021).

The influence of the front rotor was drastically reduced over the yaw and pitch angles, but it is still null around the roll. In the same 50s range, the yaw angle range dropped from 0rad to -40rad in the nominal case and from 0rad to -4rad in the fault-state. While the pitch angle range rose from 0rad to 500rad in the nominal case and from 0rad to 50rad in the fault-state, also in the same 50s range.

The third model considers a fault of 90% power loss in the back rotor and its step response can be seen in Figure 20.

Figure 20 – Open-loop step response of Quanser 3DOF Hover with back rotor fault.



Source: Author (2021).

Similar to the last case, the influence of the back rotor was drastically reduced over the yaw and pitch angles, but it is still null around the roll. In the same 50s range, the yaw angle range dropped from 0rad to -40rad in the nominal case and from 0rad to -4rad in the fault-state. While the pitch angle range dropped from 0rad to -500rad in the nominal case and from 0rad to -50rad in the fault-state, also in the same 50s range.

Analyzing all three step responses, the control sample time is defined as 50ms and process model emulation frequency as $100s^{-1}$.

5 CONTROLLER DESIGN

In this chapter, it will be shown the design process of an SSMPC for each system. The main controller design considers the linearized state-space model of the Quanser AERO, which was previously presented. In order to avoid the repetition of information given the project similarities, a more simplified analysis of the Quanser 3DOF Hover controller design is shown. In addition, it is discussed the implementation of the ANNs as fault-detection strategies. The same data set, number of hidden layers, and epochs were considered in the tests for a better performance comparison regarding the two proposed formulations.

The tuning of the MPC was based on the Quanser AERO system without faults. The goal is to find the best parameters to provide a response signal without steady state error and overshoot. To that end, tests are made varying the values of the prediction horizon from 3 to 15, the control horizon from 2 to 10, output parameter from 0.001 to 10, and control parameter from 0.001 to 10. Also, the physical system constraints are studied and the operational constraints are chosen.

The same process is done for the Quanser 3DOF Hover system, but varying the values of the prediction horizon from 50 to 70, the control horizon from 5 to 15, output parameter from 0.01 to 10, and control parameter from 0.001 to 0.1. It is noteworthy that tests were performed with other values for both systems, but the intervals described here were considered the most relevant. Also, the function qp (quadratic programming) from CVXOPT, a free software package for convex optimization based on the Python programming language, was used to solve the cases with constraints.

On account that they are MIMO cases, there is coupling between the systems' inputs and outputs, so that they interfere with each other. Also, since the systems are said to be time-invariant, the relationship between the inputs and outputs is independent of time. A good response, in this case, will be without oscillatory behavior, that does not saturate the control action, and that is suitable for the controller based on the nominal model given the operational constraints.

The ANNs' training uses Quanser AERO's nominal and fault models open-loop responses considering a varying input signal. The input array arrangement takes into account valuable data to represent the dynamics of the linearized system, thus, it is necessary to save the data from a certain number of previous simulation steps in addition to the current one. In summary, this array is composed of control action signals and the respective system output signals. In turn, the output array of the ANN is composed of 0s and 1s, indicating whether the input in question refers to the nominal or fault state, respectively. It is worth stressing that each ANN is trained to identify the nominal system and one type of fault, i.e., four ANN are trained in total.

The data is divided into the training set, validation set, and test set in a ratio of 7:2:1, in other words, 70% of the data was used for training, 20% for validation, and 10% for the final performance tests. The input vector was composed of batches of u and y from the last 10 time steps. Bias and weights are initialized randomly. The choice of the number of layers takes into

account the characteristics of the data that are being processed, thus, tests are made varying this value from 3 to 10. The number of hidden neurons can be defined empirically, but it was decided that the smallest number would be 4 and the largest would be 64. The number of epochs can also be defined empirically, being opted for 100, 300, and 500. Again, tests were performed with other values, but the intervals described here contain the most relevant information for this study.

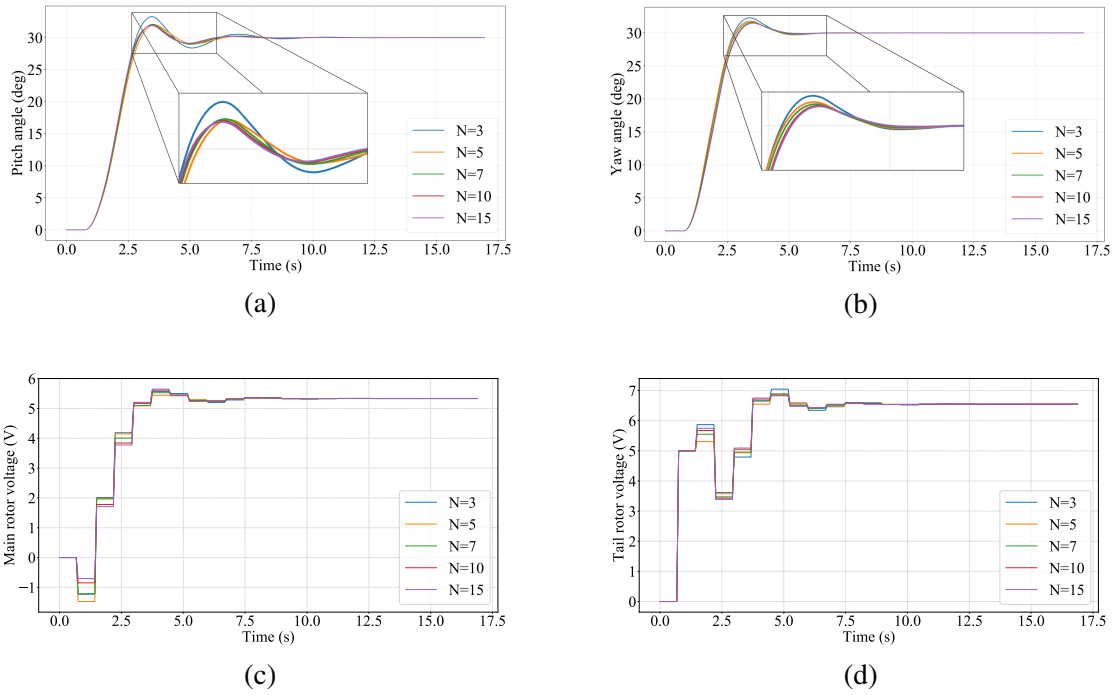
5.1 QUANSER AERO STATE-SPACE MODEL PREDICTIVE CONTROL TUNING

The Quanser AERO nominal-state closed-loop simulations in this section illustrate the system behavior for a variety of parameter combinations. As mentioned, the influence of the prediction horizon, control horizon, output parameter, and control parameter will be evaluated. The obtained results are presented and discussed in the subsequent items considering the prediction model shown in Equation (10).

5.1.1 Prediction and control horizons

To assess the effects of the value of the prediction horizon, the other parameters are fixed. So, the output parameter is defined as $\rho_y = 1$, the control parameter as $\rho_u = 0.001$, and the control horizon as $M = 3$. The physical restrictions of the system were also taken into account. As seen in Figure 21, for a 30° reference signal for both outputs, 5 values of prediction horizon were tested, $N = 3, 5, 7, 10$, and 15.

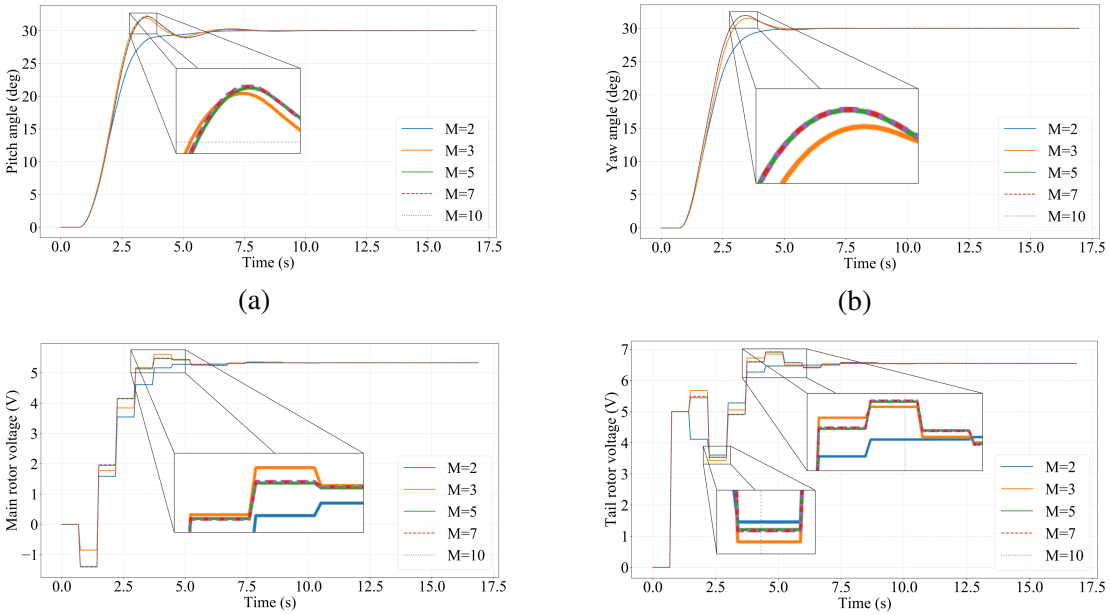
Figure 21 – SSMPC of Quanser AERO varying the prediction horizon N (values indicated in the legend): (a) Pitch angle, (b) Yaw angle, (c) Main rotor voltage, and (d) Tail rotor voltage.



Source: Author (2021).

Similarly, in order to assess the effects of the value of the control horizon, the other parameters are fixed. So, the output parameter is defined as $\rho_y = 1$, the control parameter as $\rho_u = 0.001$, and the prediction horizon as $N = 10$. The physical restrictions of the system were also taken into account. As seen in Figure 22, for a 30° reference signal for both outputs, 5 values of control horizon were tested, $M = 2, 3, 5, 7$, and 10.

Figure 22 – SSMPC of Quanser AERO varying the control horizon M (values indicated in the legend): (a) Pitch angle, (b) Yaw angle, (c) Main rotor voltage, and (d) Tail rotor voltage.



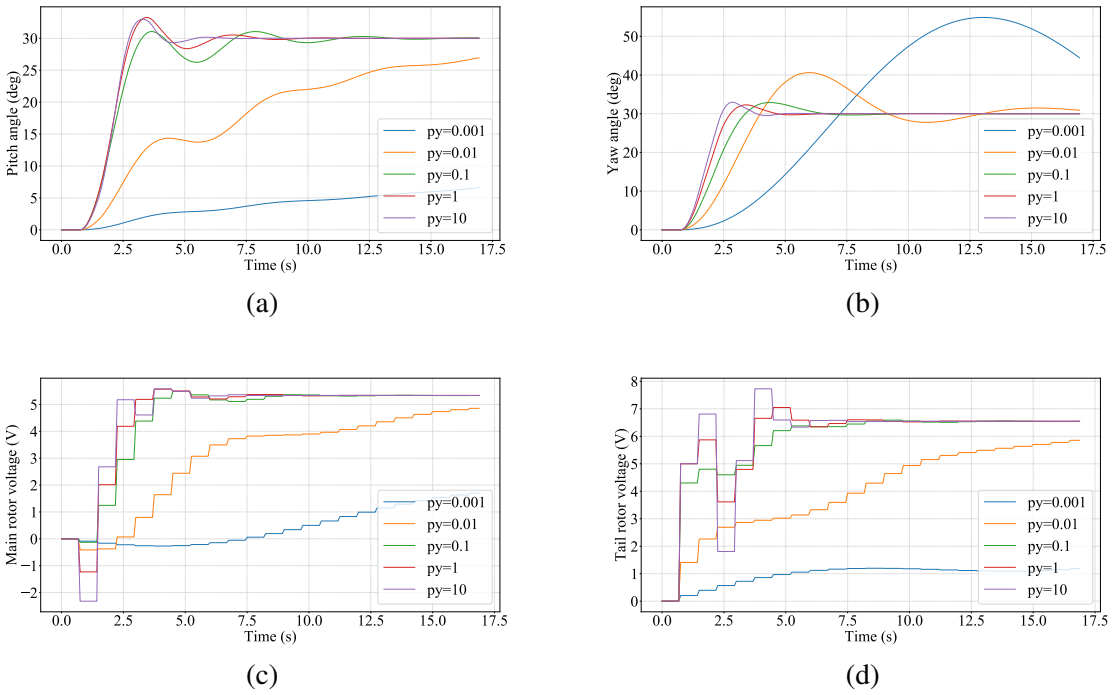
Source: Author (2021).

Theoretically, the higher the value of the horizons, the faster the response converges to the reference. Based on the results seen in Figure 21, $N = 3$ presents a good balance between response speed and damping, not being observed enough improvements to justify the use of a higher value and possibly compromise the computational cost. As for the results seen in Figure 22, it is observed that the system behavior gets better when the M value tends to N, so, it was opted for $M = 3$.

5.1.2 Design parameters

To assess the effects of the value of the output parameter, the other parameters are fixed. So, the control parameter is defined as $\rho_u = 0.001$, the prediction horizon as $N = 3$, and the control horizon as $M = 3$. The physical restrictions of the system were also taken into account. As seen in Figure 23, for a 30° reference signal for both outputs, 5 values of the output parameter were tested, $\rho_y = 0.001, 0.01, 0.1, 1$, and 10.

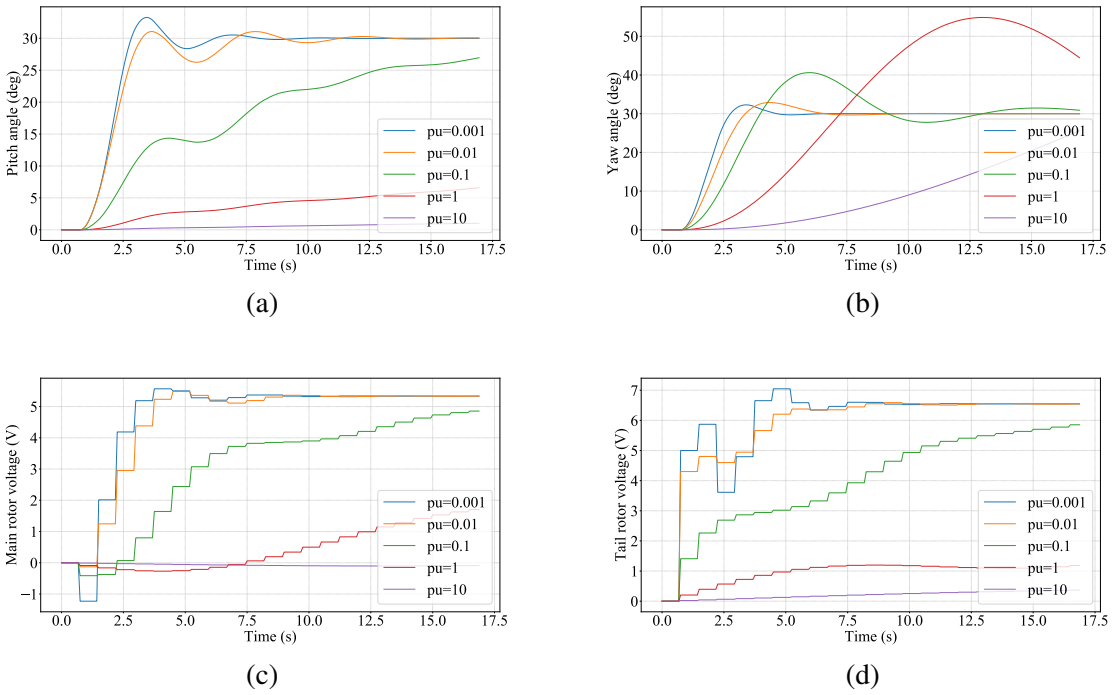
Figure 23 – SSMPC of Quanser AERO varying the output parameter ρ_y (values indicated in the legend): (a) Pitch angle, (b) Yaw angle, (c) Main rotor voltage, and (d) Tail rotor voltage.



Source: Author (2021).

Similarly, in order to assess the effects of the value of the control parameter, the other parameters are fixed. So, the output parameter is defined as $\rho_y = 1$, the prediction horizon as $N = 3$, and the control horizon as $M = 3$. The physical restrictions of the system were also taken into account. As seen in Figure 24, for a 30° reference signal for both outputs, 5 values of the control parameter were tested, $\rho_u = 0.001, 0.01, 0.1, 1$, and 10.

Figure 24 – SSMPC of Quanser AERO varying the control parameter ρ_u (values indicated in the legend): (a) Pitch angle, (b) Yaw angle, (c) Main rotor voltage, and (d) Tail rotor voltage.



Source: Author (2021).

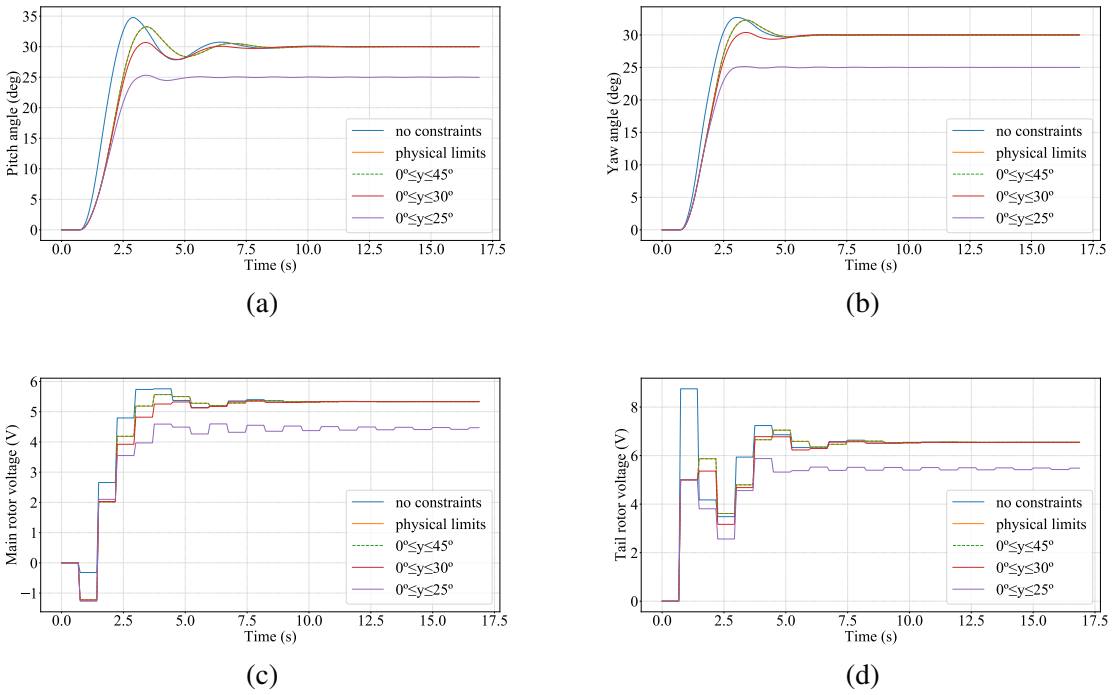
The decrease in the ρy causes the value of y to vary more, that is, it makes the system output more oscillatory, while the increase in the value of ρu increases the penalty on the control action, which makes the u to move slowly. Based on the results presented in Figure 23 and 24, it is seen a good performance for $\rho_y = 1$ and $\rho_u = 0.001$, in such a way, these values were chosen.

5.1.3 Constraints handling

Given the system's physical and operational limits, it was opted to apply a constraint treatment. Initially, the limits of pitch and yaw angles are, respectively, from $0°$ to $124°$ and from $0°$ to $360°$, the voltage limits for the main and tail rotors are both from $-18V$ to $+18V$, and the limits of the rate of voltage variation were chosen arbitrarily from $-5V$ to $+5V$. In addition, the controller parameters were kept at $N = 3$, $M = 3$, $\rho_y = 1$, and $\rho_u = 0.001$.

Some tests were carried out to better understand the behavior of the system under each constraint, emphasizing that the constraints will be constant throughout the prediction horizon. Figure 25 brings the weighting of the output signal constraint, that is, pitch and yaw angles, for $y_{max} = 45°, 30°$ and $25°$, for a reference signal of $30°$ for both outputs.

Figure 25 – SSMPC of Quanser AERO varying the $y(k)$ constraint (values indicated in the legend): (a) Pitch angle, (b) Yaw angle, (c) Main rotor voltage, and (d) Tail rotor voltage.

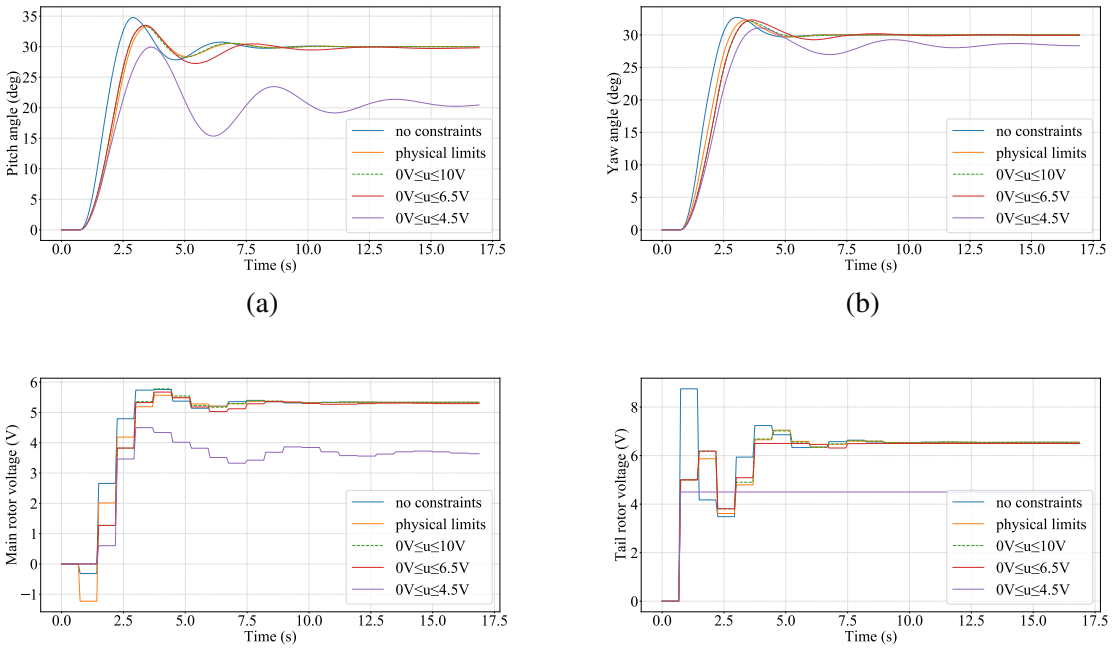


Source: Author (2021).

It is observed that the limits in $y(k)$ ensure that the output values are never too high or too low, also possibly avoiding an initial overshoot.

Figure 26 brings the weighting of the input signal constraint, that is, main and tail rotor voltage, for $0V < u(k) < 10V$, $0V < u(k) < 6.5V$ and $0V < u(k) < 4.5V$, for a reference signal of 30° for both outputs.

Figure 26 – SSMPC of Quanser AERO varying the $u(k)$ constraint (values indicated in the legend): (a) Pitch angle, (b) Yaw angle, (c) Main rotor voltage, and (d) Tail rotor voltage.

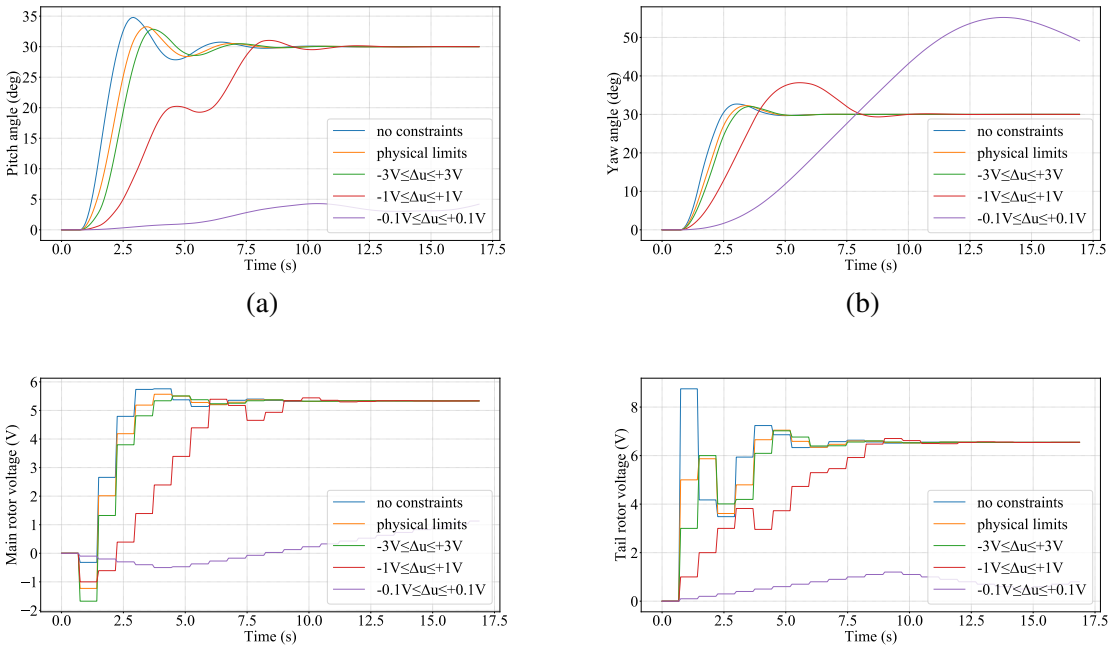


Source: Author (2021).

It is observed that upper limits imposed in $u(k)$ may prevent the output from reaching the reference, keeping them around the highest possible value.

Lastly, Figure 27 brings the weighting of the rate of change of the input signal constraint for $-3V < \Delta u(k) < +3V$, $-1V < \Delta u(k) < +1V$ and $-0.1V < \Delta u(k) < +0.1V$, for a reference signal of 30° for both outputs.

Figure 27 – SSMPC of Quanser AERO varying the $\Delta u(k)$ constraint (values indicated in the legend): (a) Pitch angle, (b) Yaw angle, (c) Main rotor voltage, and (d) Tail rotor voltage.



Source: Author (2021).

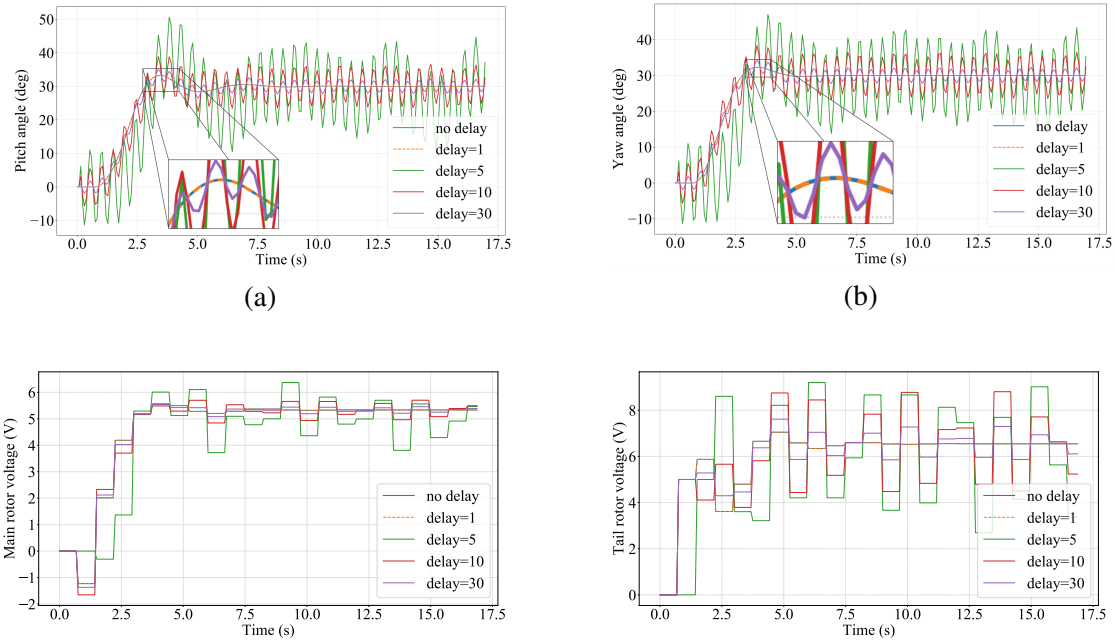
It is observed that $\Delta u(k)$ constraints affect the initial overshoot and the settling time since this variable corresponds directly to the speed at which the control signal goes up or down.

In brief, it is noted that when the constraints are not rigid, the system oscillates a lot around the reference and takes time to converge to it, and, in the more restrictive cases, the system chooses not to reach the reference in order to respect a constraint. In general, the constraints on the control action will be respected if they are specified with the requirements of the system in mind, that is, as long as they are feasible. Output constraints, on the other hand, may not be satisfied if there is a model mismatch or if they are too restrictive. Besides, it is also possible to include other constraints or cause the system to transgress any of them in an emergency situation.

5.1.4 Sensitivity to noise and delays

One of the specifications of the control project is to update the control loop at a frequency 10 times lower than the plant emulation frequency, however, it is still interesting to study the effects of a possible communication delay. For this, the controller parameters were kept at $N = 3$, $M = 3$, $\rho_y = 1$, and $\rho_u = 0.001$. The system physical constraints were also taken into account. As seen on Figure 28, for a reference signal of 30° for both outputs, different delays in terms of simulation steps were introduced.

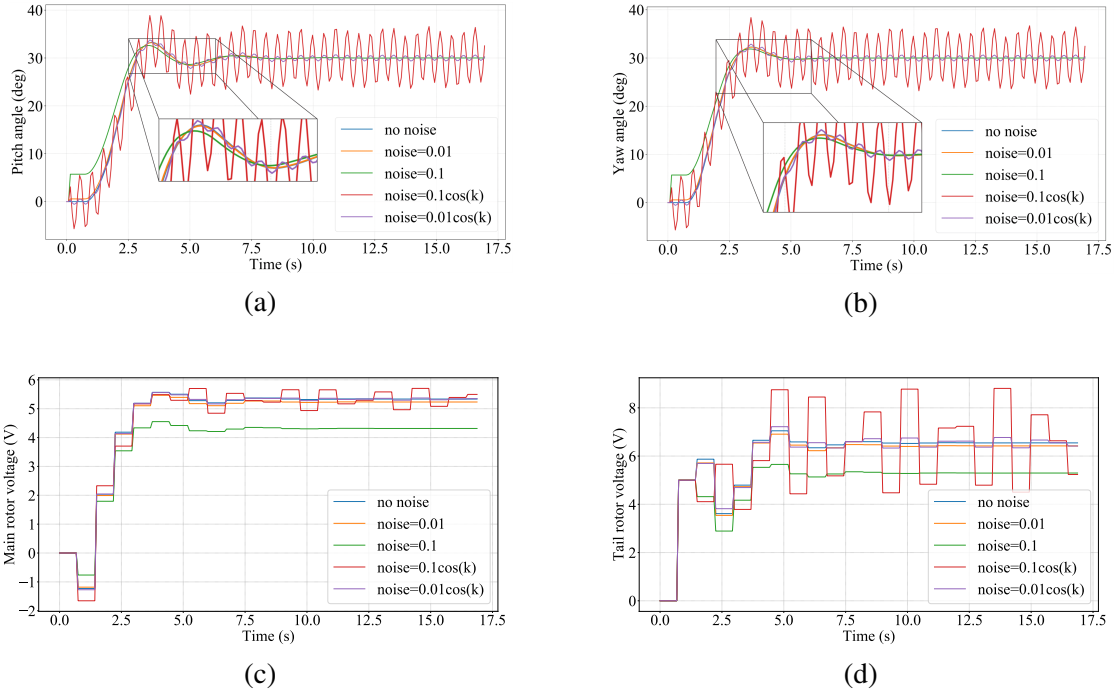
Figure 28 – Effect of adding a delay (values indicated in the legend) to the control action in the SSMPC of Quanser AERO: (a) Pitch angle, (b) Yaw angle, (c) Main rotor voltage, and (d) Tail rotor voltage.



Source: Author (2021).

Similarly, the use of non-ideal sensors can compromise the accuracy of the data and the return time of the information in a real situation. With that in mind, it was considered important to analyze the system response while adding noises to the voltage measurement of the motors. As seen in Figure 29, for a reference signal of 30° for both outputs, different noises were introduced.

Figure 29 – Effect of adding measurement noise (values indicated in the legend) to the readings of the output sensors in the SSMPC of Quanser AERO: (a) Pitch angle, (b) Yaw angle, (c) Main rotor voltage, and (d) Tail rotor voltage.



Source: Author (2021).

In both cases, it was observed that the system managed to stay around the desired operation point, however, the response oscillates around it as the problems increase.

Although these situations were not applied in the final simulation, delays and noises are real problems. Some works go deeper into this theme and it seems to be common the use of filters and state estimators for these cases, such as moving average filter or Kalman filter (CAVALCA, 2019; HE et al., 2017; ESTRADA-SÁNCHEZ; VELASCO-VILLA; RODRIGUEZ, 2017).

5.2 QUANSER 3DOF HOVER STATE-SPACE MODEL PREDICTIVE CONTROL TUNING

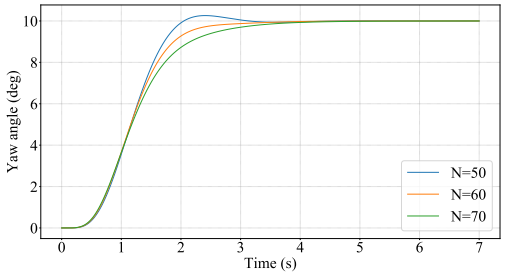
The Quanser 3DOF Hover nominal-state closed-loop simulations in this section illustrate the system behavior for a variety of parameter combinations. As mentioned, the influence of the prediction horizon, control horizon, output parameter, and control parameter will be evaluated. The obtained results are presented and discussed in the subsequent items considering the prediction model shown in Equation (3).

5.2.1 Prediction and control horizons

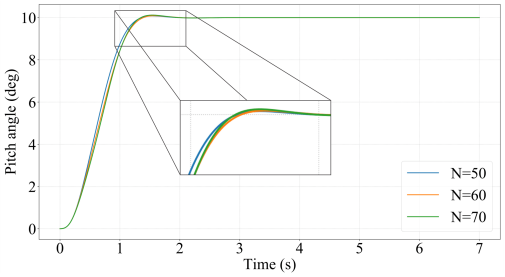
In order to assess the effects of the value of the prediction horizon, the other parameters are fixed. So, the output parameter is defined as $\rho_y = 1$, the control parameter as $\rho_u = 0.01$, and the control horizon as $M = 10$. The physical restrictions of the system were also taken into

account. As seen in Figures 30 and 31, for a reference signal of 10° for both outputs, 3 values of prediction horizon were tested, $N = 50, 60$, and 70.

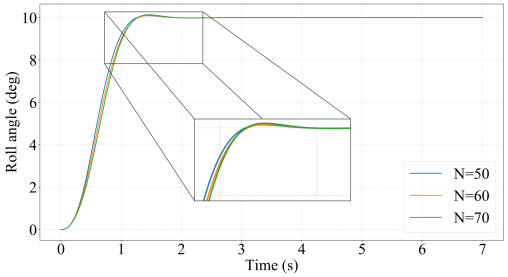
Figure 30 – SSMPC of Quanser 3DOF Hover varying the prediction horizon N (values indicated in the legend): (a) Yaw angle, (b) Pitch angle, and (c) Roll angle.



(a)



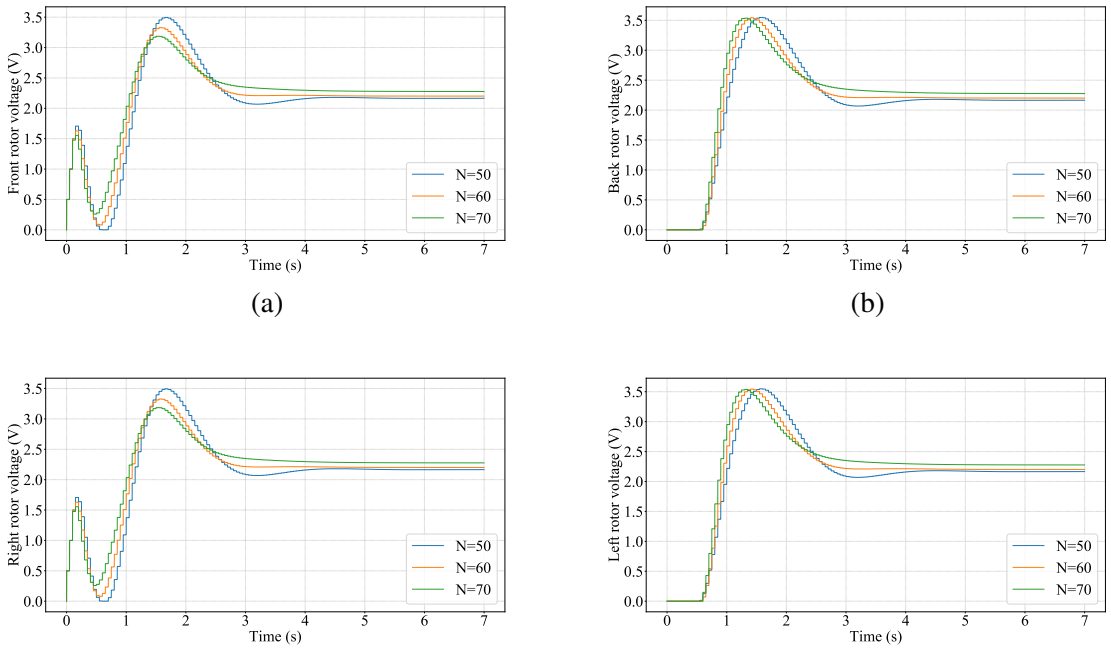
(b)



(c)

Source: Author (2021).

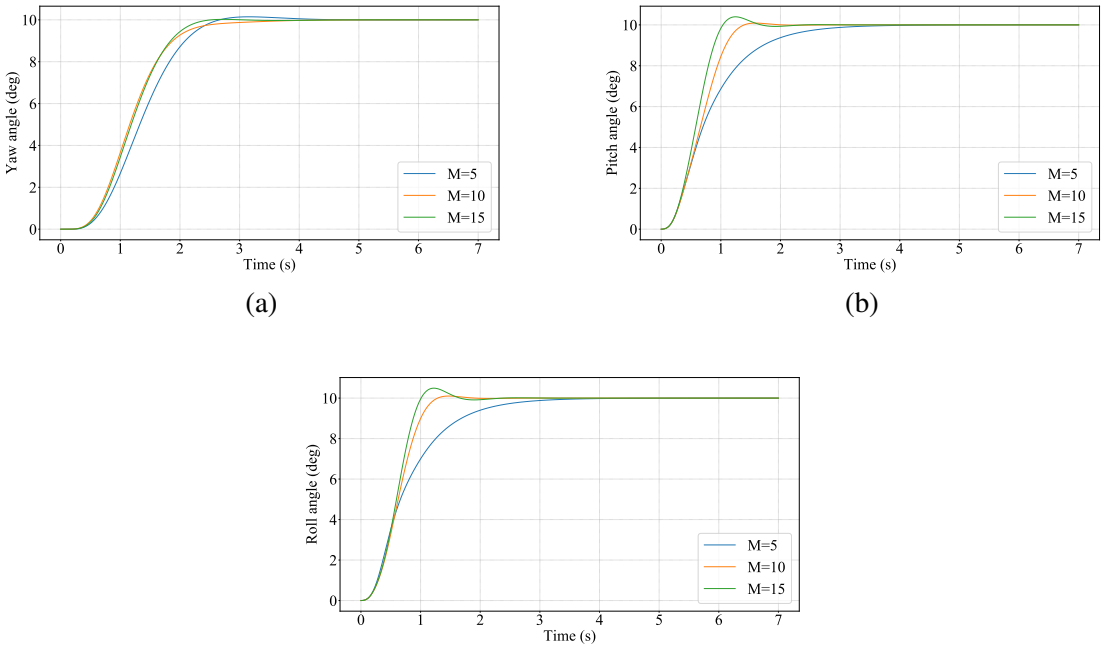
Figure 31 – SSMPC of Quanser 3DOF Hover varying the prediction horizon N (values indicated in the legend): (a) Front rotor voltage, (b) Back rotor voltage, (c) Right rotor voltage, and (d) Left rotor voltage.



Source: Author (2021).

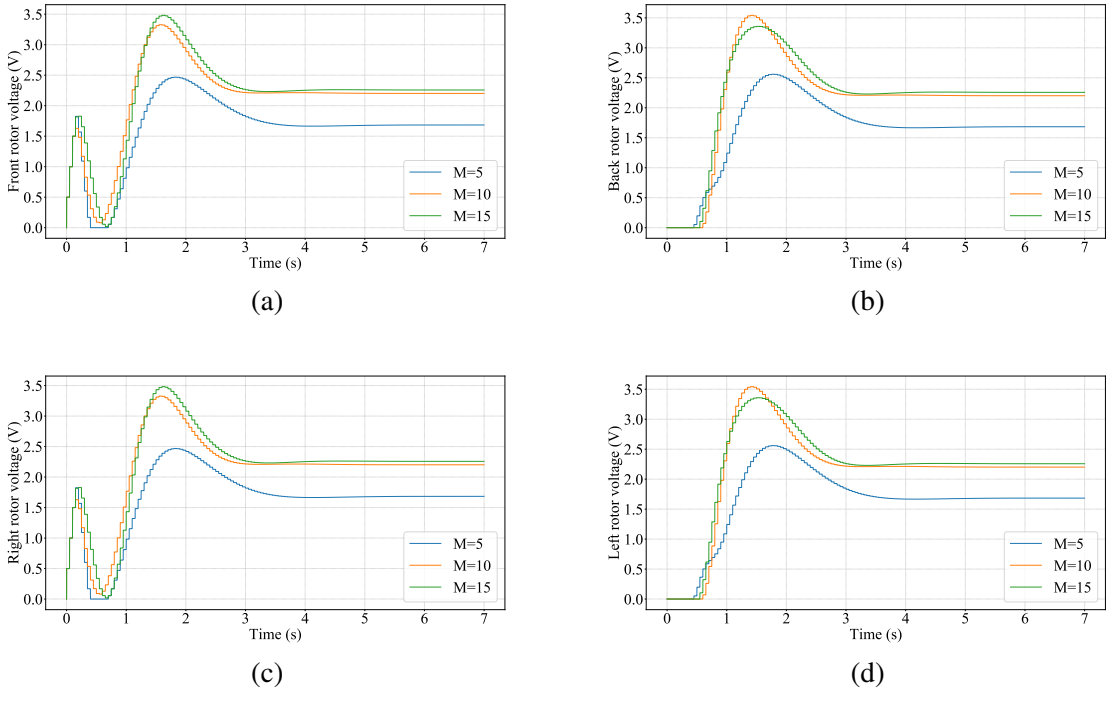
Similarly, in order to assess the effects of the value of the control horizon, the other parameters are fixed. So, the output parameter is defined as $\rho_y = 1$, the control parameter as $\rho_u = 0.01$, and the prediction horizon as $N = 60$. The physical restrictions of the system were also taken into account. As seen in Figures 32 and 33, for a reference signal of 10° for both outputs, 3 values of control horizon were tested, $M = 5, 10$, and 15.

Figure 32 – SSMPC of Quanser 3DOF Hover varying the prediction horizon M (values indicated in the legend): (a) Yaw angle, (b) Pitch angle, and (c) Roll angle.



Source: Author (2021).

Figure 33 – SSMPC of Quanser 3DOF Hover varying the prediction horizon M (values indicated in the legend): (a) Front rotor voltage, (b) Back rotor voltage, (c) Right rotor voltage, and (d) Left rotor voltage.



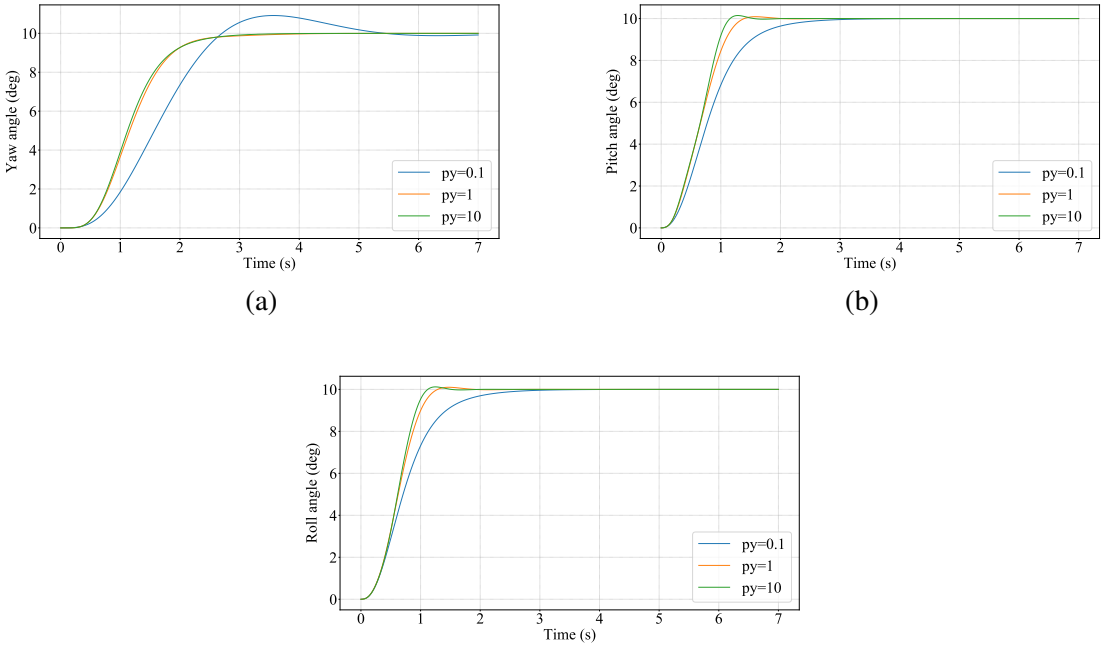
Source: Author (2021).

Based on the results seen above, $N = 60$ and $M = 10$ present a good balance between response speed and damping. These values were chosen since the computational expense did not justify the use of a higher value.

5.2.2 Design parameters

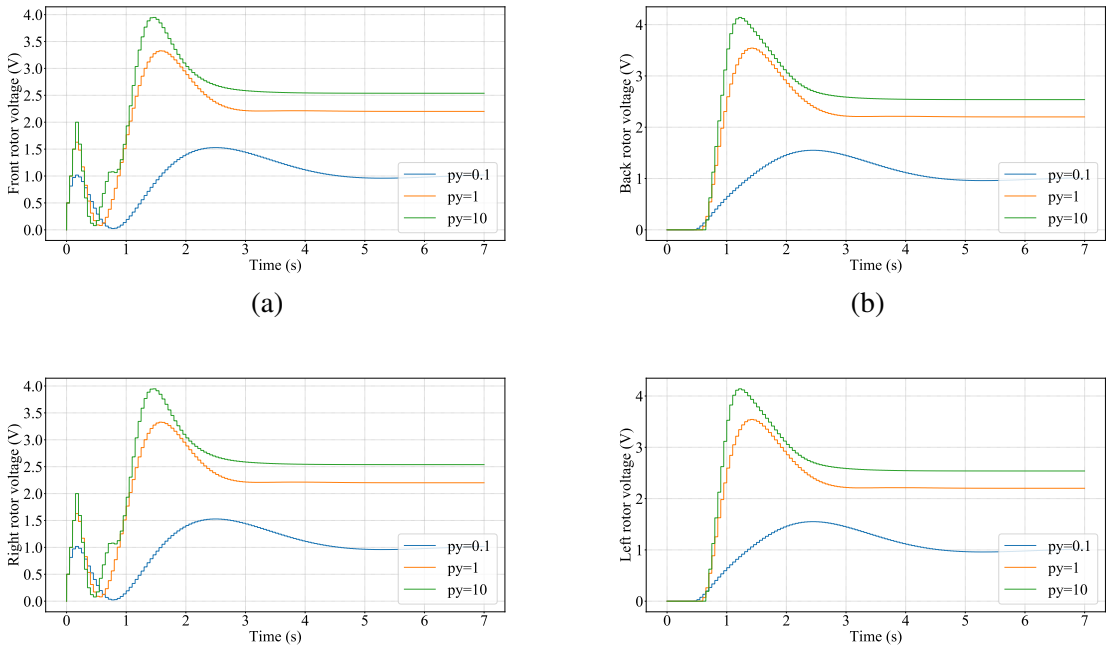
To assess the effects of the value of the output parameter, the other parameters are fixed. So, the control parameter is defined as $\rho_u = 0.01$, the prediction horizon as $N = 60$, and the control horizon as $M = 10$. The physical restrictions of the system were also taken into account. As seen in Figures 34 and 35, for a reference signal of 10° for both outputs, 3 values of the output parameter were tested, $\rho_y = 0.01, 0.1$, and 1.

Figure 34 – SSMPC of Quanser 3DOF Hover varying the prediction horizon ρ_y (values indicated in the legend): (a) Yaw angle, (b) Pitch angle, and (c) Roll angle.



Source: Author (2021).

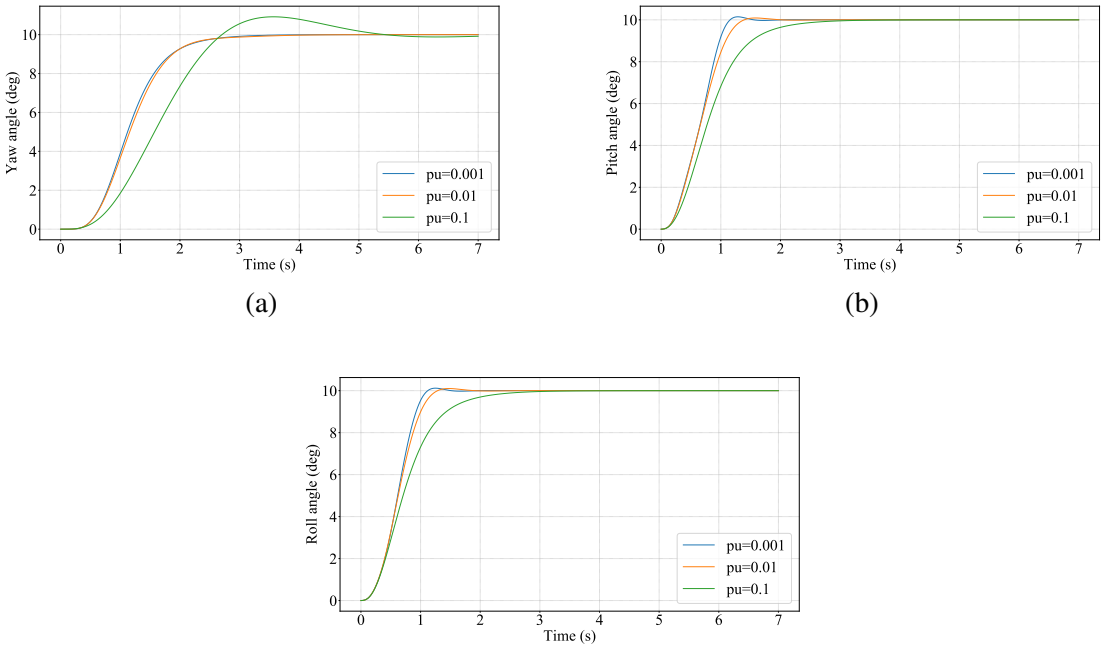
Figure 35 – SSMPC of Quanser 3DOF Hover varying the prediction horizon ρ_y (values indicated in the legend): (a) Front rotor voltage, (b) Back rotor voltage, (c) Right rotor voltage, and (d) Left rotor voltage.



Source: Author (2021).

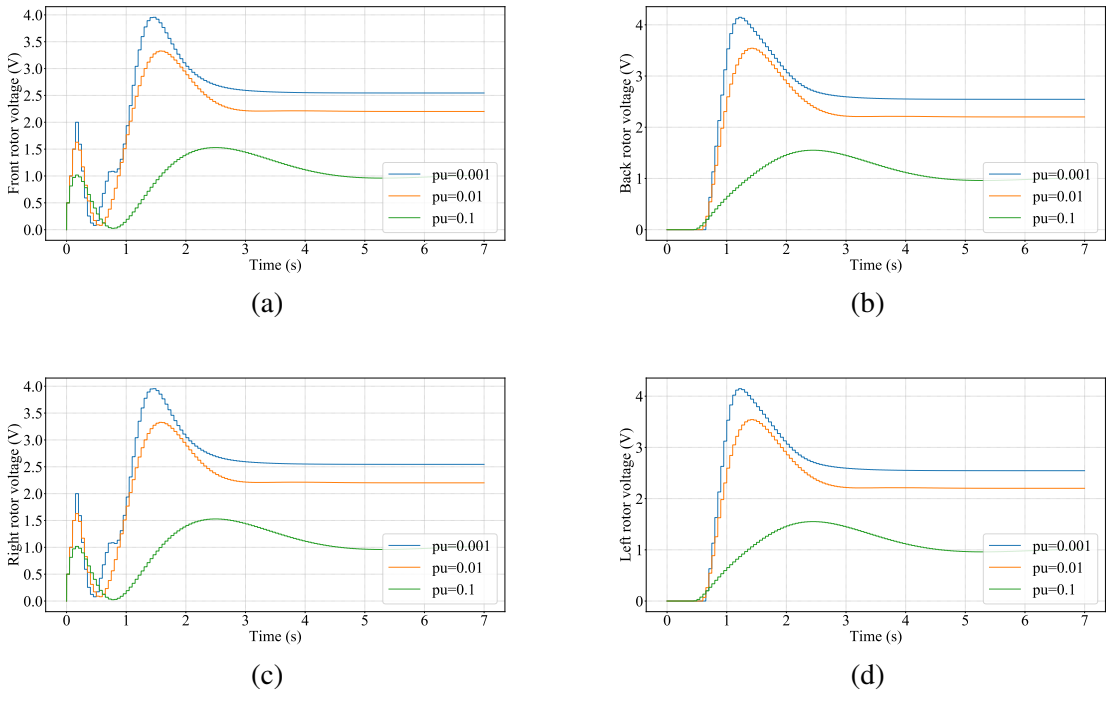
Similarly, in order to assess the effects of the value of the control parameter, the other parameters are fixed. So, the output parameter is defined as $\rho_y = 1$, the prediction horizon as $N = 60$, and the control horizon as $M = 10$. The physical restrictions of the system were also taken into account. As seen in Figures 36 and 37, for a reference signal of 10° for both outputs, 3 values of the control parameter were tested, $\rho_u = 0.001, 0.01$, and 0.1.

Figure 36 – SSMPC of Quanser 3DOF Hover varying the prediction horizon ρ_u (values indicated in the legend): (a) Yaw angle, (b) Pitch angle, and (c) Roll angle.



Source: Author (2021).

Figure 37 – SSMPC of Quanser 3DOF Hover varying the prediction horizon ρ_u (values indicated in the legend): (a) Front rotor voltage, (b) Back rotor voltage, (c) Right rotor voltage, and (d) Left rotor voltage.



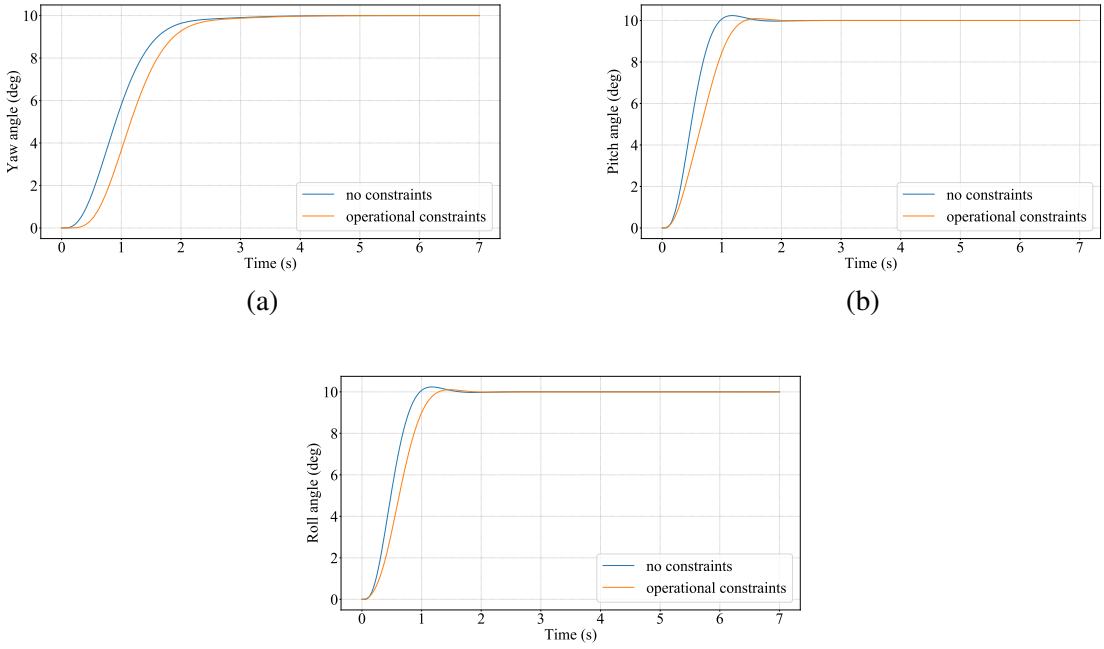
Source: Author (2021).

Based on the results presented above, it is seen a good performance for $\rho_y = 1$ and $\rho_u = 0.01$. These values were chosen since the computational expense did not justify the use of a higher value.

5.2.3 Constraints handling

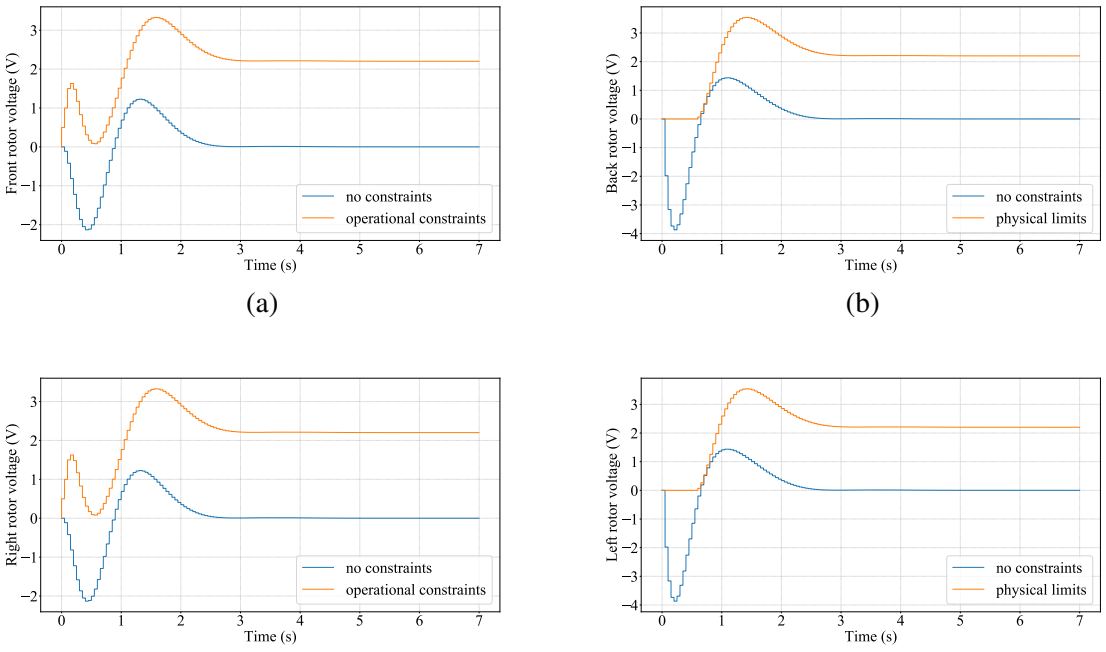
Given the system's physical and operational limits, it was opted to apply a constraint treatment. Initially, the limits of yaw, pitch, and roll angles are, respectively, from $-360°$ to $360°$, from $-30°$ to $30°$ and from $-30°$ to $30°$, the voltage limits for rotors are both from $0V$ to $+12V$, and the limits of the rate of voltage variation were chosen arbitrarily from $-0.5V$ to $+0.5V$. In addition, the controller parameters were kept at $N = 60$, $M = 10$, $\rho_y = 1$, and $\rho_u = 0.01$. The results can be seen in Figures 38 and 39.

Figure 38 – SSMPC of Quanser 3DOF Hover with and without constraint treatment: (a) Yaw angle, (b) Pitch angle, and (c) Roll angle.



Source: Author (2021).

Figure 39 – SSMPC of Quanser 3DOF Hover with and without constraint treatment: (a) Front rotor voltage, (b) Back rotor voltage, (c) Right rotor voltage, and (d) Left rotor voltage.



Source: Author (2021).

The system was able to work under the specified constraints. With the constraints handling, there was a decrease in the initial overshoot in the pitch and roll angles, but the accommodation time was maintained. The voltage graphs have also changed considerably to adhere to the condition of imposing only positive voltage values.

5.3 MULTILAYER PERCEPTRON NEURAL NETWORK TRAINING FOR QUANSER AERO ROTOR FAULTS

The analysis and validation of the MLPNN take into account the system nominal and fault models, being designed one ANN for the identification of each fault. Several tests were carried out to verify the accuracy in relation to the number of inputs, layers, neurons, and epochs, knowing that too little information and processing can lead to underfitting, but too much can lead to overfitting.

Table 5 provides a summary of the simulation results that seeks to find the best parameters combination for the Quanser AERO's main rotor fault identification.

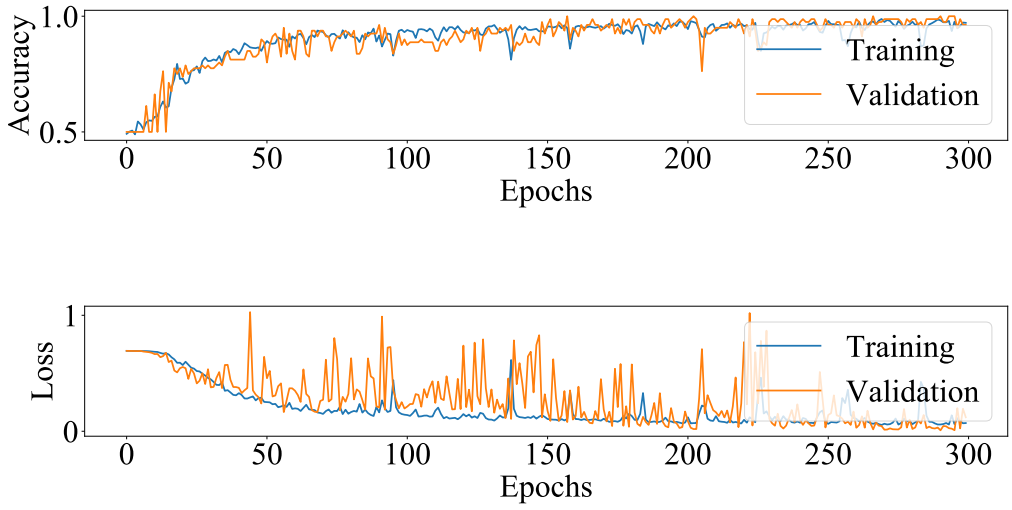
Table 5 – MLPNN accuracy for Quanser AERO's main rotor fault identification.

Hidden layers	Epochs	Accuracy (%)
3	100	50
3	300	94
3	500	91
5	100	80
5	300	95
5	500	81
7	100	80
7	300	96
7	500	84
10	100	80
10	300	87
10	500	50

Source: Author (2021).

It can be noted that the accuracy of the ANN was better in the tests with 300 epochs, with a slightly better response for the case with 7 hidden layers. Accordingly, Figure 40 provides a graphical representation of the performance of the final ANN, facilitating the observation and evaluation of its performance.

Figure 40 – Final MLPNN loss and accuracy for Quanser AERO's main rotor fault identification.



Source: Author (2021).

Then, the same process is done for the Quanser AERO's tail rotor fault identification. Table 6 brings a summary of the results.

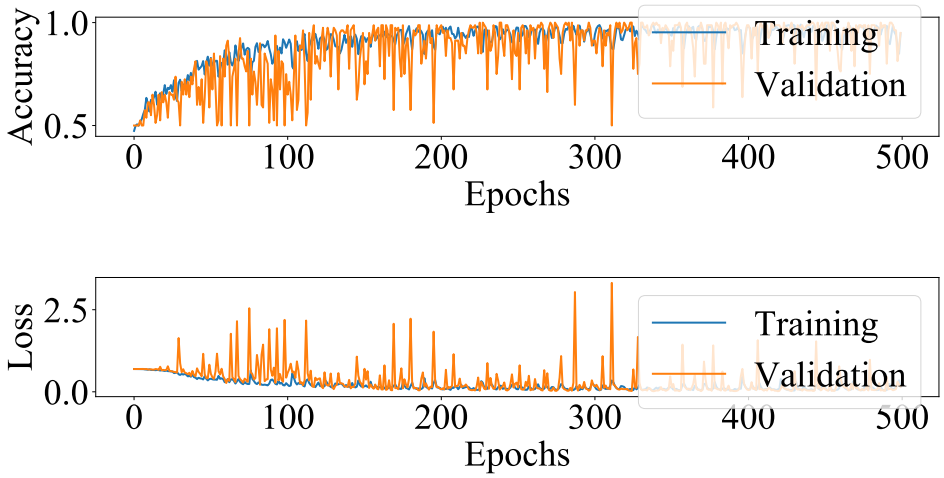
Table 6 – MLPNN accuracy for Quanser AERO's tail rotor fault identification.

Hidden layers	Epochs	Accuracy (%)
3	100	71
3	300	86
3	500	90
5	100	70
5	300	76
5	500	80
7	100	57
7	300	93
7	500	94
10	100	74
10	300	84
10	500	90

Source: Author (2021).

In this case, it can be noted that the accuracy of the ANN was better in the tests with 500 epochs, with a slightly better response for the case with 7 hidden layers. Figure 41 provides a graphical representation of the performance of the final ANN, facilitating the observation and evaluation of its performance.

Figure 41 – Final MLPNN loss and accuracy for Quanser AERO's tail rotor fault identification.



Source: Author (2021).

5.4 FULLY CONNECTED CASCADE NEURAL NETWORK TRAINING FOR QUANSER AERO ROTOR FAULTS

Similar to the MLPNN, the FCCNN aims to identify the occurrence of faults. Several tests were carried out to verify the accuracy in relation to the number of inputs, layers, neurons,

and epochs.

Table 7 provides a summary of the simulation results that seeks to find the best parameters combination for the Quanser AERO's main rotor fault identification.

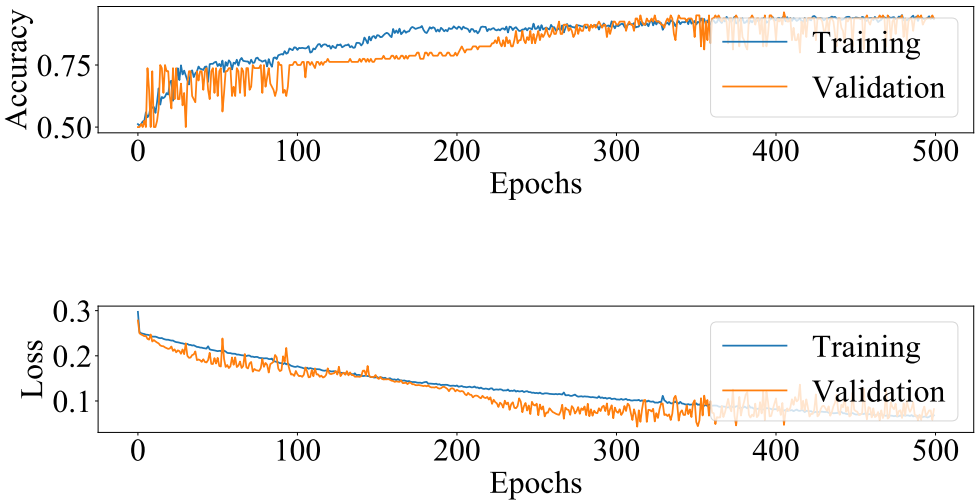
Table 7 – FCCNN accuracy for Quanser AERO's main rotor fault identification.

Hidden layers	Epochs	Accuracy (%)
3	100	65
3	300	74
3	500	80
5	100	71
5	300	81
5	500	86
7	100	72
7	300	83
7	500	85
10	100	80
10	300	92
10	500	93

Source: Author (2021).

It can be noted that the accuracy of the ANN increases with the increase in the number of hidden layers and epochs, achieving a good response with 10 hidden layers and 500 epochs. Figure 9 provides a graphical representation of the performance of the final ANN, facilitating the observation and evaluation of its performance.

Figure 42 – Final FCCNN loss and accuracy for Quanser AERO's main rotor fault identification.



Source: Author (2021).

Then, the same process is done for the Quanser AERO's tail rotor fault identification. Table 8 brings a summary of the results.

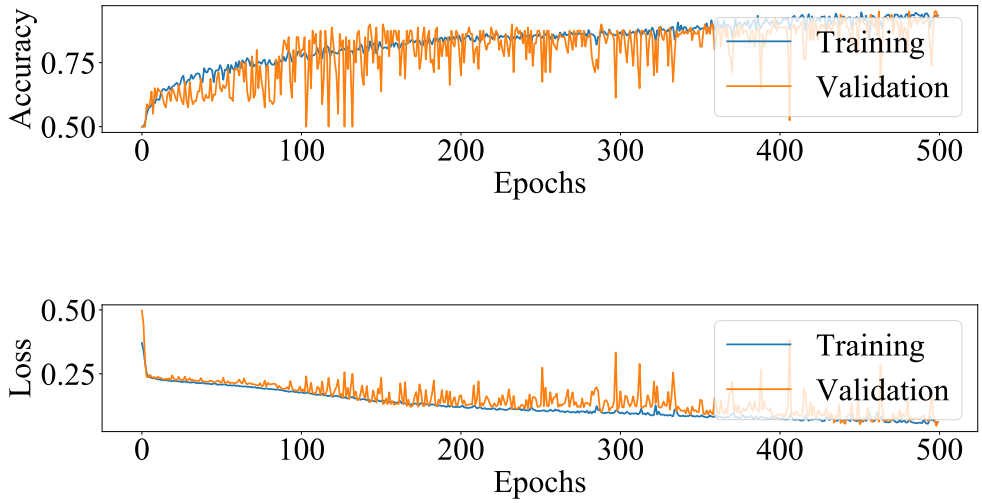
Table 8 – FCCNN accuracy for Quanser AERO's tail rotor fault identification.

Hidden layers	Epochs	Accuracy (%)
3	100	71
3	300	79
3	500	80
5	100	72
5	300	82
5	500	87
7	100	76
7	300	82
7	500	90
10	100	82
10	300	90
10	500	92

Source: Author (2021).

Again, it can be noted that the accuracy of the ANN increases with the increase in the number of hidden layers and epochs, achieving a good response with 10 hidden layers and 500 epochs. Figure 43 provides a graphical representation of the performance of the final ANN, facilitating the observation and evaluation of its performance.

Figure 43 – Final FCCNN loss and accuracy for Quanser AERO's tail rotor fault identification.



Source: Author (2021).

6 TEST AND VALIDATION

Closed-loop simulations were performed with the purpose of evaluating the behavior of the proposed MPC algorithm. Initially, nominal and fault systems responses are analyzed in order to distinguish their dynamical properties. Then, it is studied the issue of feasibility, which serves as a starting point in assessing the need for the testing of the control strategy with multiple models. The application of the proposed controller is done in a variety of situations. Lastly, the fault identification algorithm is applied to the controller and the obtained results are discussed.

To better visualize the performance of the algorithm, it is used a confusion or error matrix basic formulation, which contains a set of four different combinations of expected and actual values. From Table 9, a true positive (TP) means the predicted positive is true, a true negative (TN) means the predicted negative is true, a false positive (FP) is a type 1 error that means the predicted positive is false, and the false negative (FN) is a type 2 error that means the predicted negative is false (NERKHEDE, 2018).

Table 9 – Confusion matrix.

		Actual values	
		Positive	Negative
Fault detection	Positive	True positive (TP)	False positive (FP)
	Negative	False negative (FN)	True negative (TN)

Source: Adapted from Nerkhede (2018).

In this context, TP represents a fault that was identified correctly, TN represents the system that is operating in the nominal mode and no faults have been identified, FP represents the system that is operating in the nominal mode but a fault was identified, and FN means that there is a system fault but it wasn't identified. It is reinforced that the Quanser AERO faults are 50% power losses in the main and tail rotors, while the Quanser 3DOF Hover faults are 90% power losses in the front and back rotors.

6.1 QUANSER 3DOF HOVER STATE-SPACE MODEL PREDICTIVE CONTROL ANALYSIS

The study of the Quanser 3DOF Hover type system took place in order to assess the effects of non-feasibility, to that end, it is proposed the analysis of the MPC designed specifically for the nominal case without fault tolerance. This controller was not designed to deal with the proposed faults and there is no mechanism to detect the occurrence of them, even though they are severe. The controller parameters are presented in Table 10 and the system operational constraints were considered.

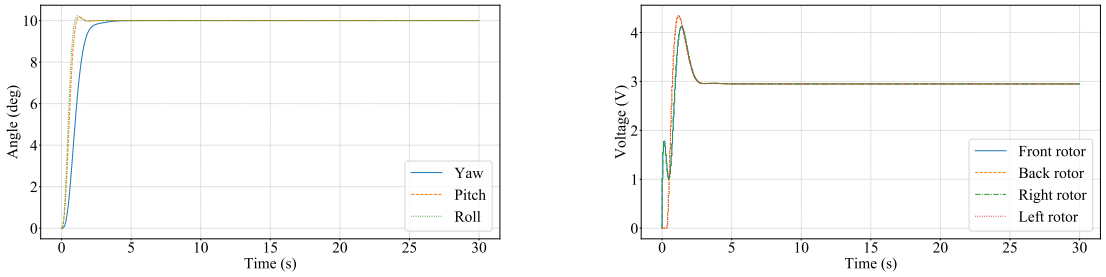
Table 10 – Quanser 3DOF Hover SSMPC parameters.

Parameter	Value
N	60
M	10
ρ_y	1
ρ_u	0.01

Source: Author (2021).

Figure 44 shows the system response for a reference signal of 10° for both outputs. This response has little overshoot and reaches steady state quickly. Also, the behavior of the rotors is similar when the same reference is imposed for all angles.

Figure 44 – SSMPC of Quanser 3DOF Hover with nominal emulation model for $ref_y = ref_p = ref_r = 10°$: (a) Angle, and (b) Voltage.

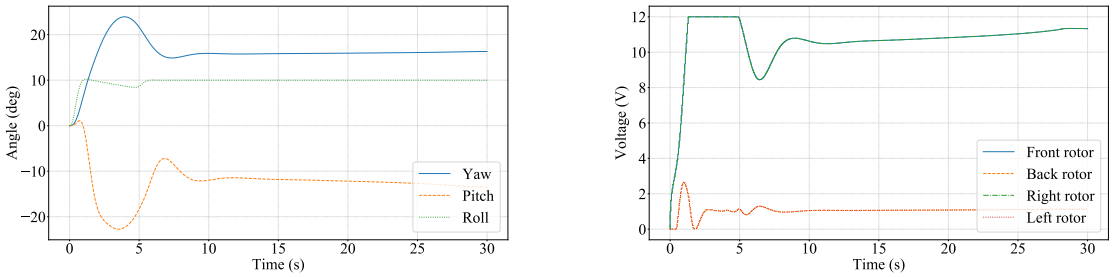


(a) (b)

Source: Author (2021).

In Figure 45, it is observed that the controller designed for the nominal system was not able to control the system with front rotor fault. It was necessary to relax the constraints for the pitch and roll angles from 15° to 30°, but, although these have not been violated, the system output does not follow the established reference.

Figure 45 – SSMPC of Quanser 3DOF Hover with front rotor fault emulation model for $ref_y = ref_p = ref_r = 10°$: (a) Angle, and (b) Voltage.

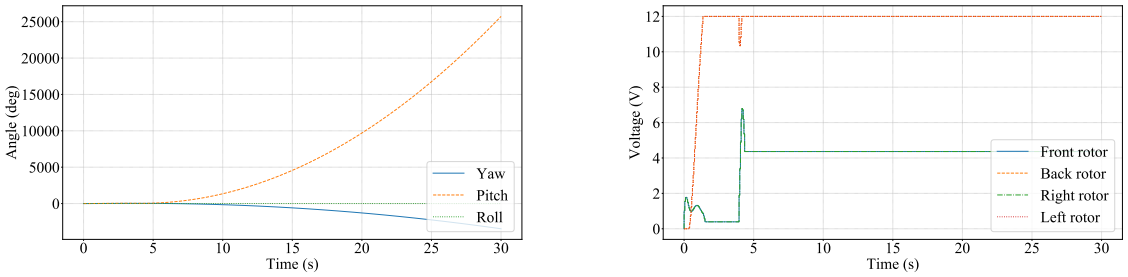


(a) (b)

Source: Author (2021).

In the case of the back rotor fault, seen in Figure 46, the optimization problem becomes non-feasible and all operational restrictions are violated in addition to the system not reaching the reference. This happens even when imposing the physical constraints in the simulation, as observed in Figure 47.

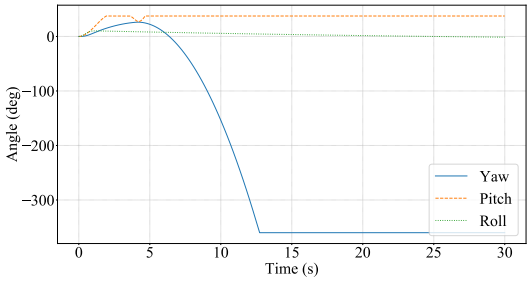
Figure 46 – SSMPC of Quanser 3DOF Hover with back rotor fault emulation model for $ref_y = ref_p = ref_r = 10°$: (a) Angle, and (b) Voltage.



(a) (b)

Source: Author (2021).

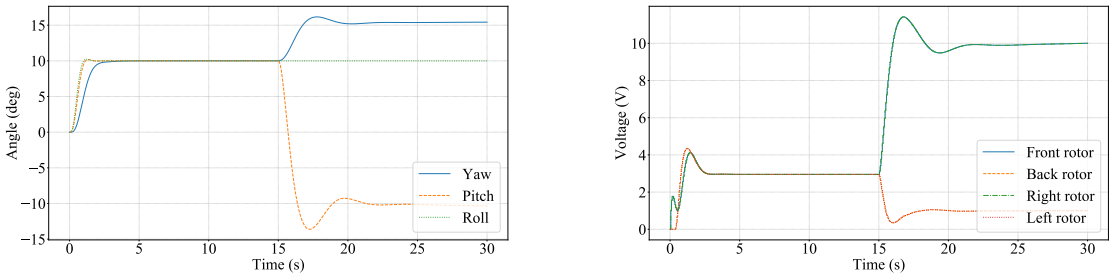
Figure 47 – SSMPC of Quanser 3DOF Hover with back rotor fault emulation model but forcing the simulation of physical constraints.



Source: Author (2021).

Analyzing the occurrence of faults during the simulation, it is observed that, even if the system reaches the permanent regime given the use of the correct prediction model in relation to the system emulation model, once the emulation model changes, the system response is lost. Figure 48 shows the response to the occurrence of front rotor fault in the middle of the simulation time, while Figure 49 shows the response to the back rotor fault in the same situation.

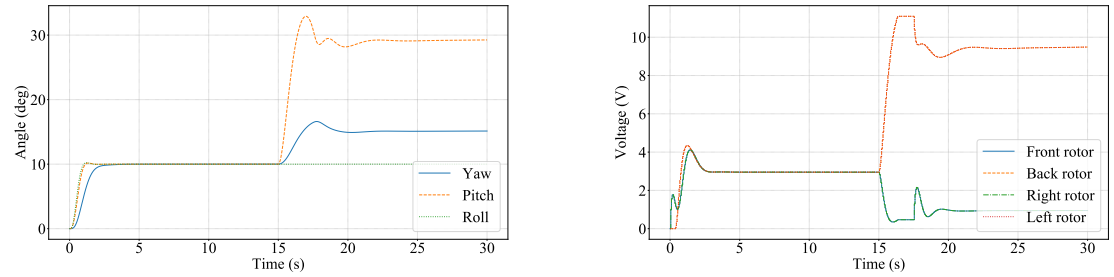
Figure 48 – SSMPC of Quanser 3DOF Hover with the occurrence of front rotor fault for $ref_y = ref_p = ref_r = 10°$: (a) Angle, and (b) Voltage.



(a) (b)

Source: Author (2021).

Figure 49 – SSMPC of Quanser 3DOF Hover with the occurrence of back rotor fault for $ref_y = ref_p = ref_r = 10°$: (a) Angle, and (b) Voltage.



(a) (b)

Source: Author (2021).

From this analysis it can be estimated that the updating of the prediction model will help with the feasibility problem, therefore, it is interesting to have a system that detects the occurrence of faults and changes the prediction model accordingly. With this, it is expected to respect the safety constraints and to be able to carry out the appropriate measures, such as, for example, the system shutdown.

6.2 QUANSER AERO STATE-SPACE MODEL PREDICTIVE CONTROL ANALYSIS

After verifying the need to address non-feasibility through the analysis of the Quanser 3DOF Hover, it is proposed the study of the Quanser AERO, which is a simpler system, but whose results can be generalized for other drone systems. The controller parameters are presented in Table 11, and the system operational constraints were considered.

Table 11 – Quanser AERO SSMPC parameters.

Parameter	Value
N	3
M	3
ρ_y	1
ρ_u	0.001

Source: Author (2021).

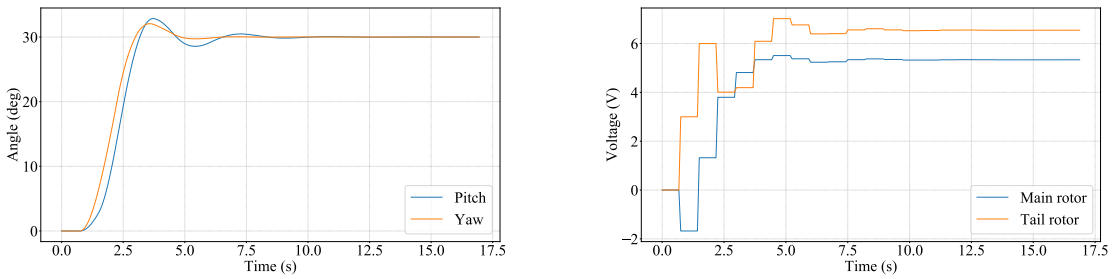
6.2.1 Nominal controller

Initially, it is repeated the analysis proposed in the previous section but for the new system, that is, the use of the nominal SSMPC in the occurrence of faults. Then, controllers were designed specifically for the fault models. Finally, it was analyzed the cases of correct and incorrect fault identification.

6.2.1.1 True negative or system without faults

The nominal system was thoroughly studied. Followed by the analysis of the system in fault states without updating the prediction model and constraints. Figure 50 shows the closed-loop simulation results for a reference signal of 30° for both outputs.

Figure 50 – SSMPC of Quanser AERO with nominal emulation model for $ref_p = ref_y = 30°$: (a) Angle, and (b) Voltage.



(a) (b)

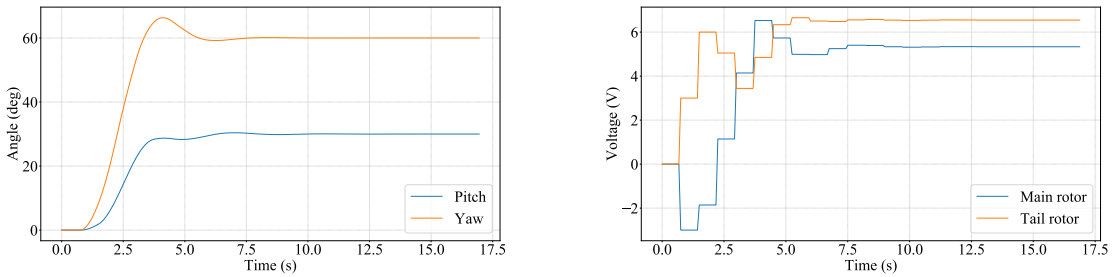
Source: Author (2021).

In sequence, several different reference signals were tested, being observed that the system has some difficulty in reaching it when the desired value of the pitch and yaw angles differs considerably. This was expected due to the linearization process in the mathematical modeling.

Figure 51 shows the closed-loop simulation results for a reference signal of 30° for the pitch angle and 60° for the yaw angle, while Figure 52 shows the results for a reference signal of 30° for the pitch angle and 180° for the yaw angle. Comparing these results, it can be seen that

the system reaches the reference in half of the time in the first response. Also, in both cases, the output signal does not come close to reaching the established constraints.

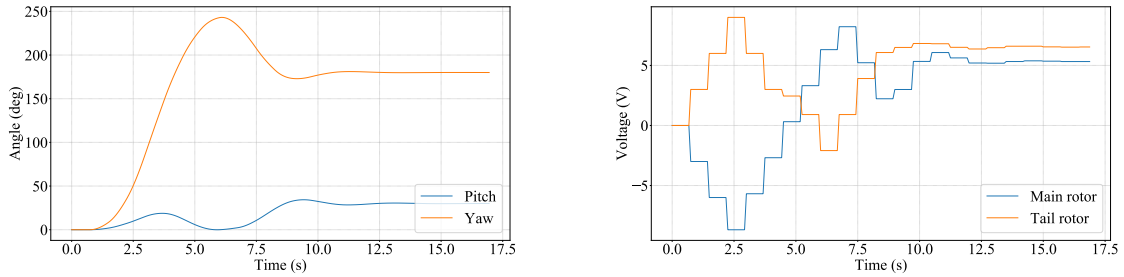
Figure 51 – SSMPC of Quanser AERO with nominal emulation model for $ref_p = 30°$ and $ref_y = 60°$: (a) Angle, and (b) Voltage.



(a) (b)

Source: Author (2021).

Figure 52 – SSMPC of Quanser AERO with nominal emulation model for $ref_p = 30°$ and $ref_y = 180°$: (a) Angle, and (b) Voltage.

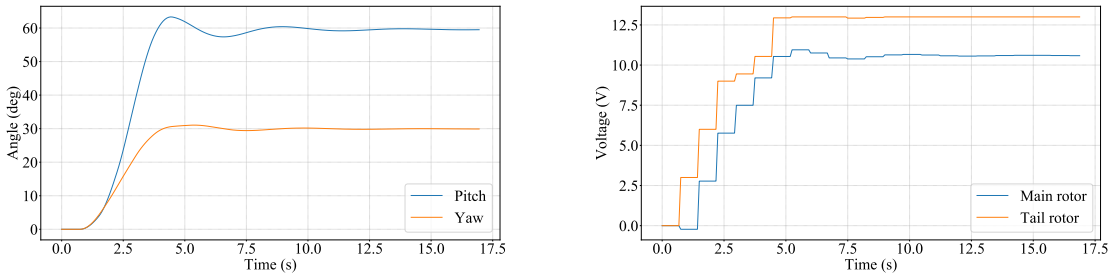


(a) (b)

Source: Author (2021).

Figure 53 shows the closed-loop simulation results for a reference signal of 60° for the pitch angle and 30° for the yaw angle, while Figure 54 shows the results for a reference signal of 90° for the pitch angle and 30° for the yaw angle. Comparing these results, it can be seen that the second response cannot reach the reference due to the output constraints.

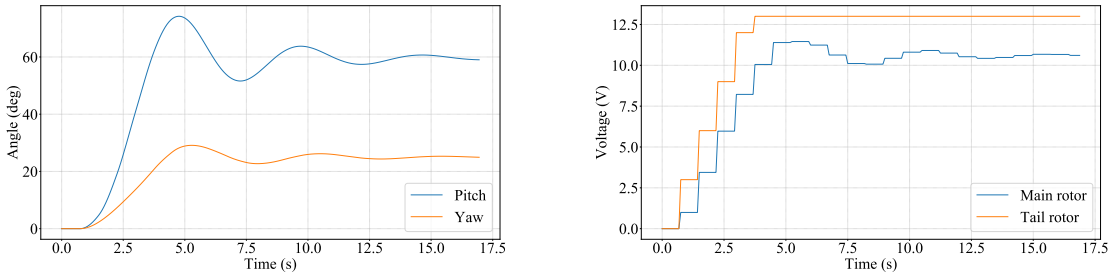
Figure 53 – SSMPC of Quanser AERO with nominal emulation model for $ref_p = 60°$ and $ref_y = 30°$: (a) Angle, and (b) Voltage.



(a) (b)

Source: Author (2021).

Figure 54 – SSMPC of Quanser AERO with nominal emulation model for $ref_p = 90°$ and $ref_y = 30°$: (a) Angle, and (b) Voltage.

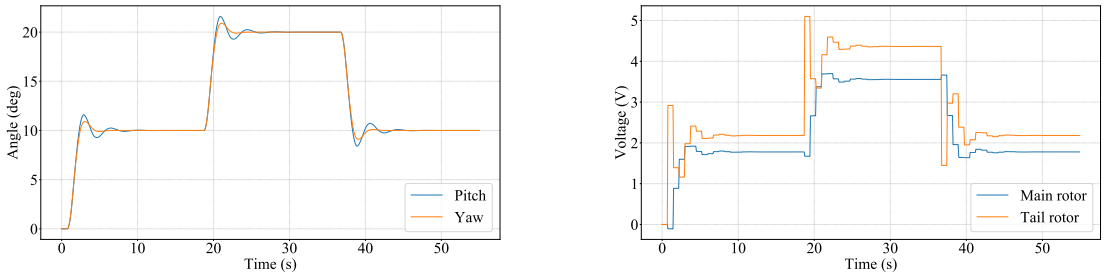


(a) (b)

Source: Author (2021).

Lastly, for two equal variable reference signals going from 10° to 20°, and then going back to 10°, the behavior of the system remains similar to the first case. That is, small overshoot and quickly reaching steady state.

Figure 55 – SSMPC of Quanser AERO with nominal emulation model for a variable reference signal: (a) Angle, and (b) Voltage.



(a) (b)

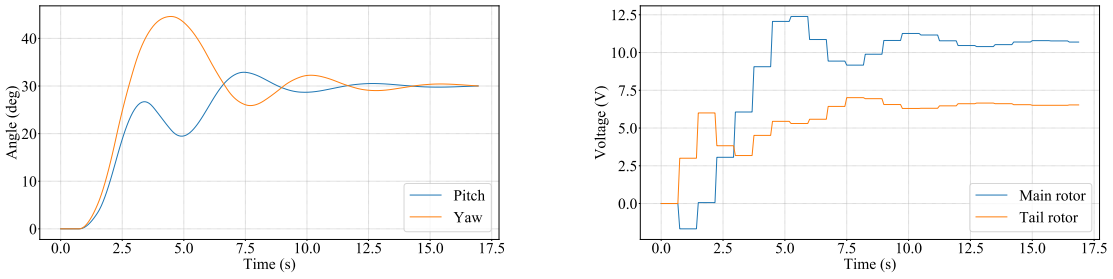
Source: Author (2021).

6.2.1.2 *False negative or system with unidentified faults*

The fault systems were analyzed while maintaining the nominal controller to verify the need to design a separate controller for each case, as foreseen by the studies of the other system. This case is defined as a false negative since the controller used was not designed for such an application, which is the same as saying that there was an error in identifying the fault.

Figures 56 and 57 show that, for a reference of 30°, the nominal controller was able to accommodate both faults after double the time observed in the system with no faults.

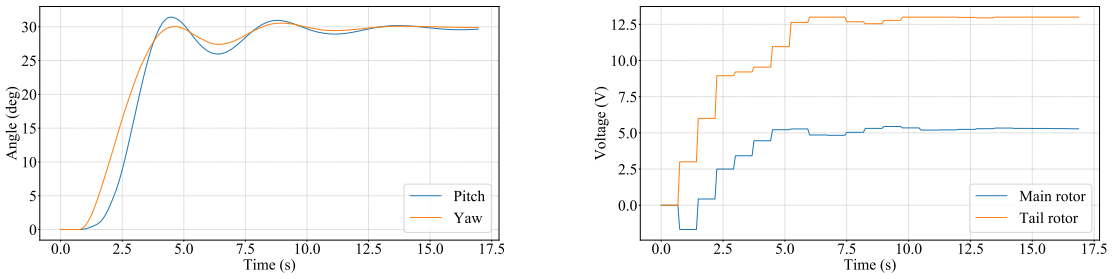
Figure 56 – SSMPC of Quanser AERO with main rotor fault emulation model and nominal control model for $ref_p = ref_y = 30°$: (a) Angle, and (b) Voltage.



(a) (b)

Source: Author (2021).

Figure 57 – SSMPC of Quanser AERO with tail rotor fault emulation model and nominal control model for $ref_p = ref_y = 30°$: (a) Angle, and (b) Voltage.

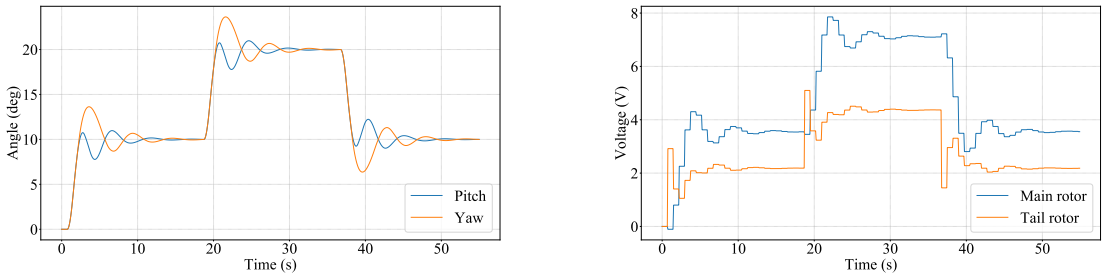


(a) (b)

Source: Author (2021).

Figures 58 and 59 show that, for a variable reference signal, the nominal controller response was too oscillatory when compared to the system with no faults.

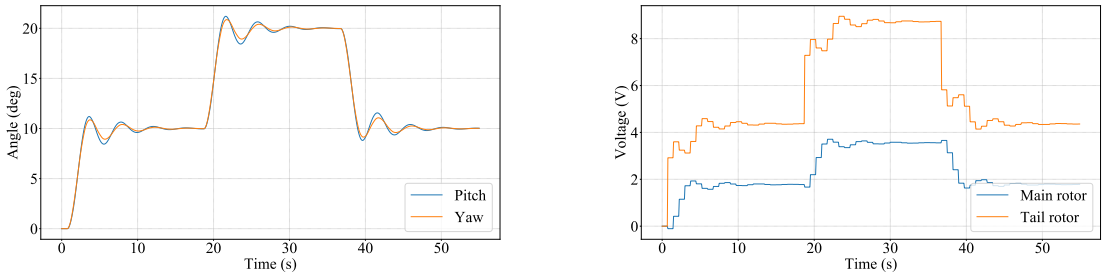
Figure 58 – SSMPC of Quanser AERO with main rotor fault emulation model and nominal control model for a variable reference signal: (a) Angle, and (b) Voltage.



(a) (b)

Source: Author (2021).

Figure 59 – SSMPC of Quanser AERO with tail rotor fault emulation model and nominal control model for a variable reference signal: (a) Angle, and (b) Voltage.



(a) (b)

Source: Author (2021).

Through these responses, it is observed that simulations for the front and back rotor fault follow the predicted behavior.

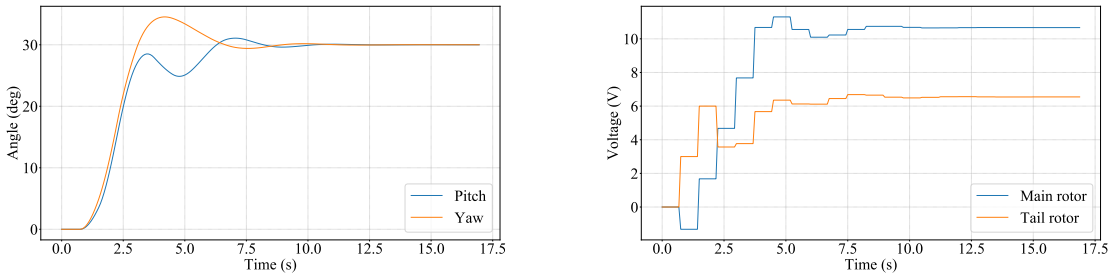
6.2.2 Fault-state controller

Given the indispensability of the controller design specifically for rotor faults, it was decided to make a controller based on each model of the system.

6.2.2.1 True positive or system with correct fault identification

This analysis corresponds to the faults' correct identification and the consequent change of the prediction model, that is, a true positive case.

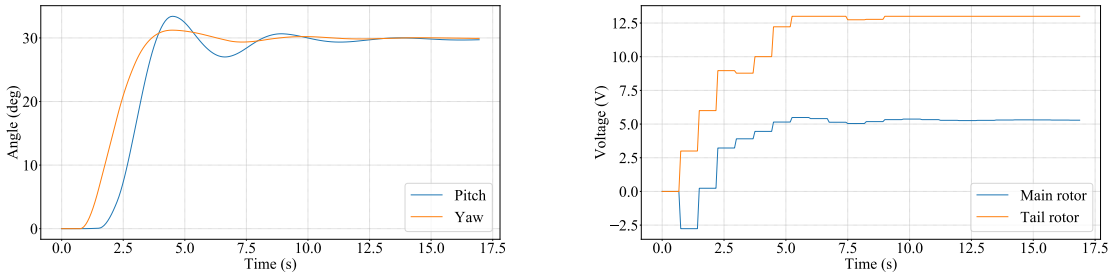
Figure 60 – SSMPC of Quanser AERO with main rotor fault emulation and control model for $ref_p = ref_y = 30°$: (a) Angle, and (b) Voltage.



(a) (b)

Source: Author (2021).

Figure 61 – SSMPC of Quanser AERO with tail rotor fault emulation and control model for $ref_p = ref_y = 30°$: (a) Angle, and (b) Voltage.



(a) (b)

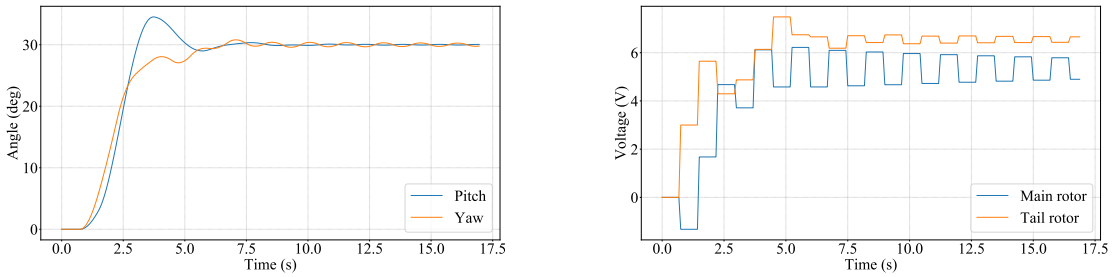
Source: Author (2021).

Figures 60 and 61 show that the system converges faster and with a smaller overshoot in both cases, also, the response is less oscillatory.

6.2.2.2 *False positive or system with incorrect fault identification*

Another situation is the incorrect detection of faults when the system is operating in the nominal mode, this would cause the controller to be updated for one of the fault models while the emulation of the plant would occur with the nominal model. This case is defined as a false positive.

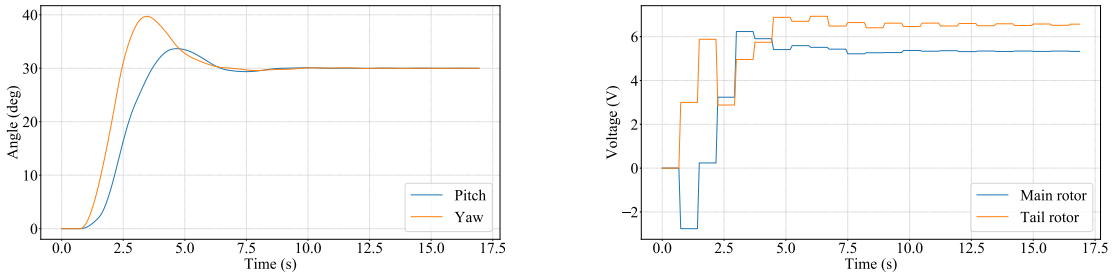
Figure 62 – SSMPC of Quanser AERO with nominal emulation model and main rotor fault control model for $ref_p = ref_y = 30°$: (a) Angle, and (b) Voltage.



(a) (b)

Source: Author (2021).

Figure 63 – SSMPC of Quanser AERO with nominal emulation model and tail rotor fault control model for $ref_p = ref_y = 30°$: (a) Angle, and (b) Voltage



(a) (b)

Source: Author (2021).

It is observed that the use of controllers designed for fault cases does not perform well when applied to the nominal system. The response is oscillating around the reference in Figure 62 and is below the reference in Figure 63.

6.3 QUANSER AERO MULTI-MODEL PREDICTIVE CONTROL ANALYSIS

Since the purpose of this work is to study the behavior of the Quanser AERO in the occurrence of faults online, it is necessary to extend this analysis to the situation in which the change of the representative model of the system occurs during the simulation. For such, the three system models are applied.

6.3.1 Classification without the neural network

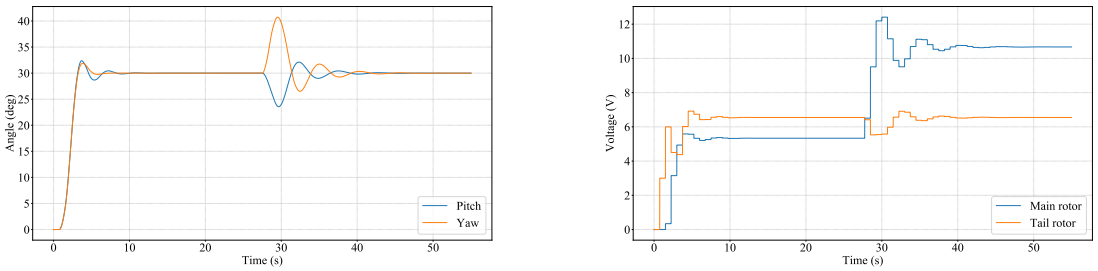
Again, three situations are considered in the following tests and both occur instantly. The first situation refers to the ideal identification of the system models corresponding to each state, i.e., the ANN used has 100% accuracy and is capable of identifying the faults correctly every

time. The second situation refers to when the identification doesn't occur at all. Lastly, the third situation refers to when the identification is always wrong.

6.3.1.1 False negative or system with unidentified faults

To represent the situation of a false negative, the system emulation model was changed in the middle of the simulation, but the nominal model was maintained for the controller.

Figure 64 – MMPC of Quanser AERO with the occurrence of main rotor fault and using only the nominal control model for $ref_p = ref_y = 30°$: (a) Angle, and (b) Voltage.

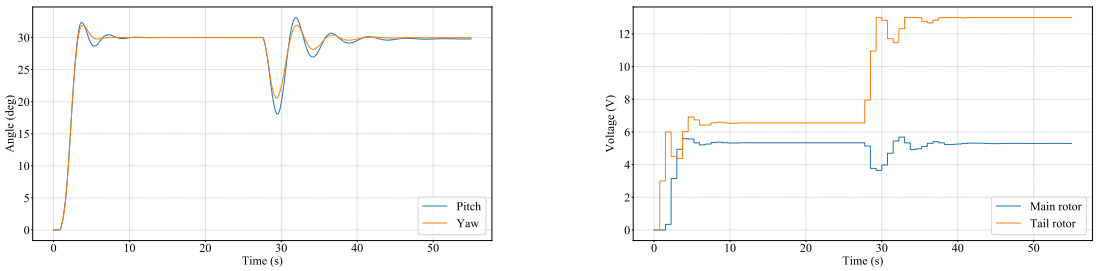


(a) (b)

Source: Author (2021).

From Figure 64, it is observed that simulations for the front rotor fault follow the predicted behavior, that is, the system is able to reach the permanent regime and to follow the reference signal even with the change of model.

Figure 65 – MMPC of Quanser AERO with the occurrence of back rotor fault and using only the nominal control model for $ref_p = ref_y = 30°$: (a) Angle, and (b) Voltage.

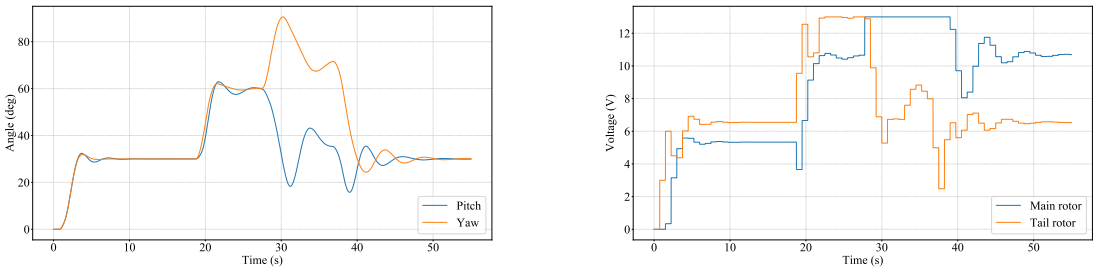


(a) (b)

Source: Author (2021).

From Figure 65, it is observed that simulations for the back rotor fault also follow the predicted behavior. Despite the response oscillating around the reference, the control is able to react correctly after changing the model.

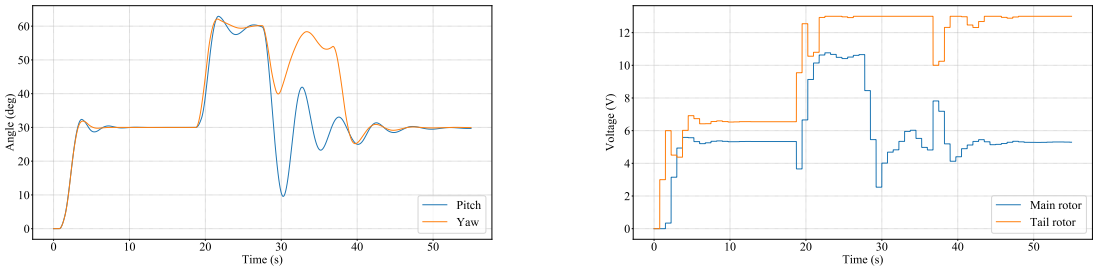
Figure 66 – MMPC of Quanser AERO with the occurrence of front rotor fault and using only the nominal control model for a variable reference signal: (a) Angle, and (b) Voltage.



(a) (b)

Source: Author (2021).

Figure 67 – MMPC of Quanser AERO with the occurrence of back rotor fault and using only the nominal control model for a variable reference signal: (a) Angle, and (b) Voltage.



(a) (b)

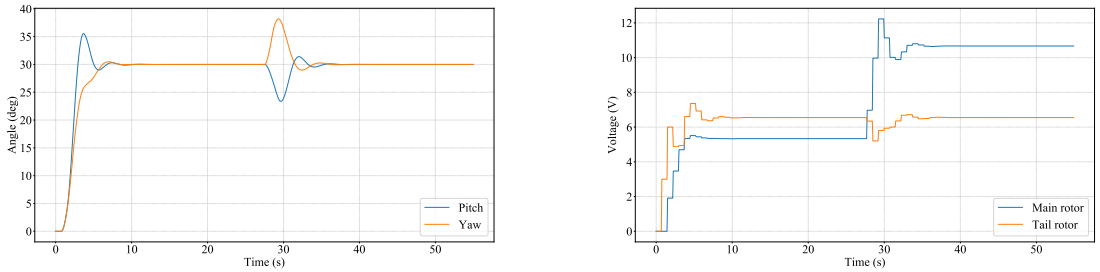
Source: Author (2021).

Figures 66 and 67 show what happens when the reference signals go from 30° to 60° and then go back to 30°.

6.3.1.2 False positive or system with incorrect fault identification

To represent the situation of a false positive, the system emulation model was changed in the middle of the simulation, but the fault model was maintained for the controller.

Figure 68 – MMPC of Quanser AERO with the occurrence of main rotor fault and using only the fault control model for $ref_p = ref_y = 30°$: (a) Angle, and (b) Voltage.

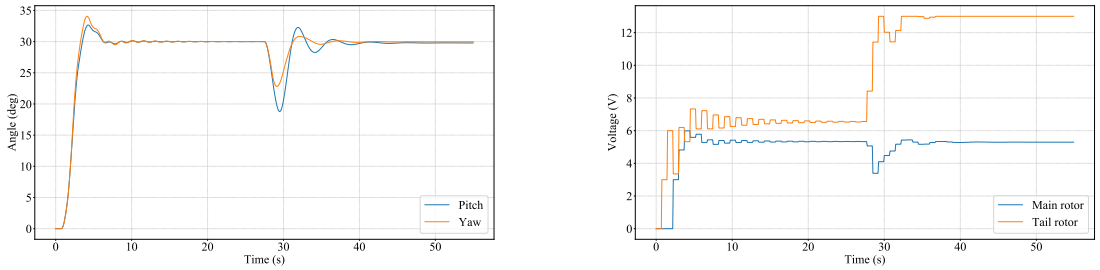


(a) (b)

Source: Author (2021).

From Figures 68, it is observed an initial overshoot, but the system reaches steady state.

Figure 69 – MMPC of Quanser AERO with the occurrence of tail rotor fault and using only the fault control model for $ref_p = ref_y = 30°$

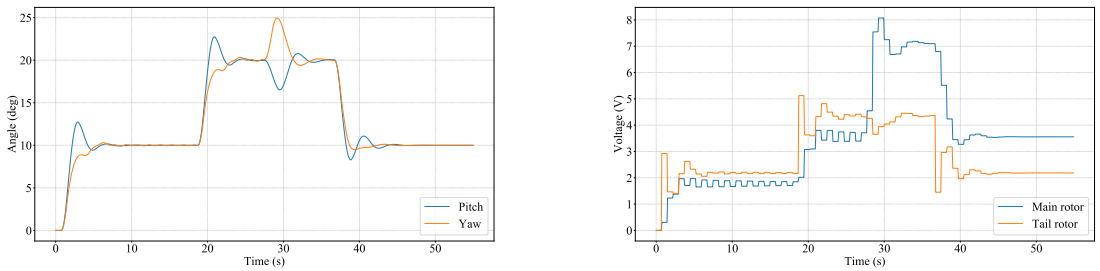


(a) (b)

Source: Author (2021).

From Figure 69, it is observed an oscillatory response, but the system reaches steady state.

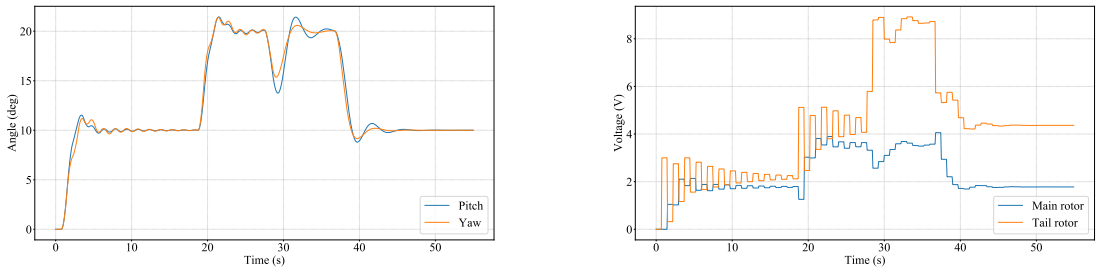
Figure 70 – MMPC of Quanser AERO with the occurrence of main rotor fault and using only the fault control model for a variable reference signal: (a) Angle, and (b) Voltage.



(a) (b)

Source: Author (2021).

Figure 71 – MMPC of Quanser AERO with the occurrence of tail rotor fault and using only the fault control model for a variable reference signal: (a) Angle, and (b) Voltage.



(a) (b)

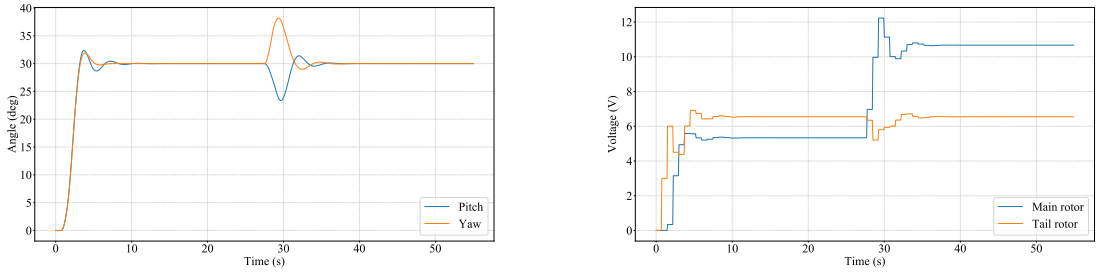
Source: Author (2021).

Figures 70 and 71 show what happens when the reference signals go from 30° to 60° and then go back to 30°.

6.3.1.3 True positive or system with correct fault identification

For the analysis of a true positive case, the emulation and control models are the same, that is, the nominal system controller is only used for the nominal case, while the fault system controllers are used for the fault cases.

Figure 72 – MMPC of Quanser AERO with the occurrence of main rotor fault and using the correct control model for $ref_p = ref_y = 30°$.

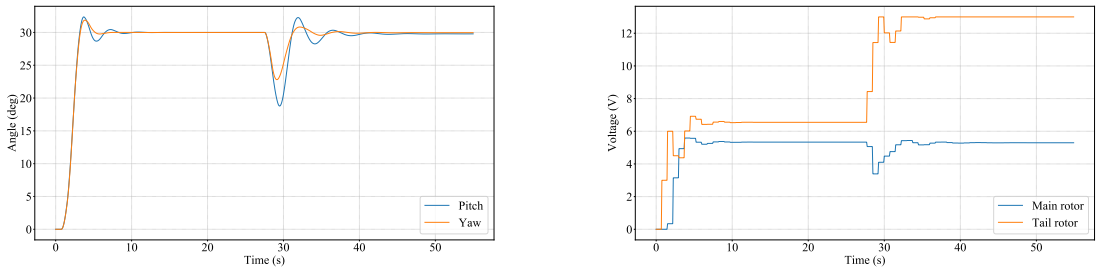


(a) (b)

Source: Author (2021).

Observing Figure 72, the controller designed for the fault state showed the best response.

Figure 73 – MMPC of Quanser AERO with the occurrence of tail rotor fault and using the correct control model for $ref_p = ref_y = 30°$.

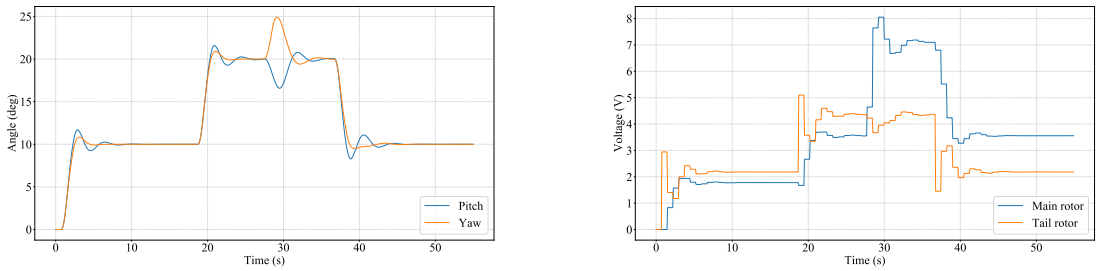


(a) (b)

Source: Author (2021).

The same can be seen in Figure 73, that is, the controller designed for the fault state showed the best response.

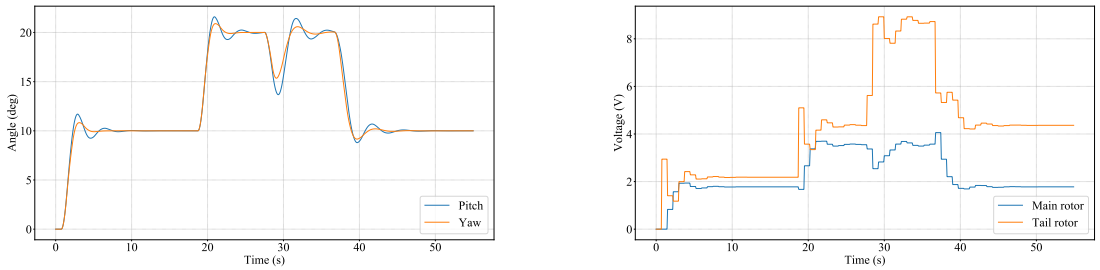
Figure 74 – MMPC of Quanser AERO with the occurrence of main rotor fault and using the correct control model for a variable reference signal: (a) Angle, and (b) Voltage.



(a) (b)

Source: Author (2021).

Figure 75 – MMPC of Quanser AERO with the occurrence of tail rotor fault and using the correct control model for a variable reference signal: (a) Angle, and (b) Voltage.



(a) (b)

Source: Author (2021).

Figures 74 and 75 show what happens when the reference signals go from 30° to 60° and then go back to 30°.

6.3.2 Classification with the neural network

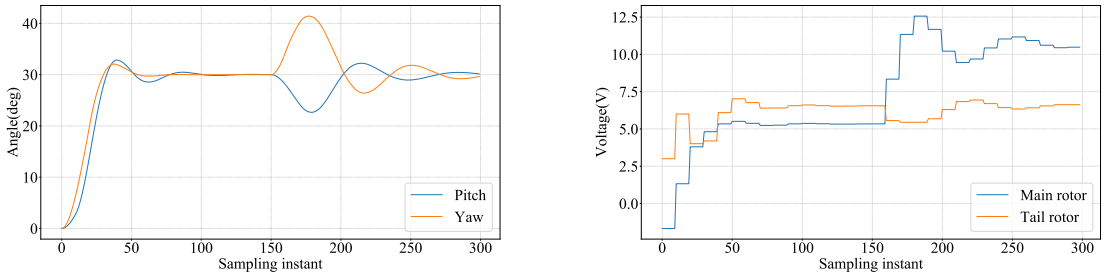
Finally, it is going to be considered that the identification of the system model corresponding to each situation is done by one of the designed ANNs, i.e., that the ANN used does not have 100% accuracy, being unable to identify the faults correctly every time.

6.3.2.1 Fault identification with multilayer perceptron neural network

The MMPC algorithm is merged with the MLPNN. It was defined that the system nominal model would be used for the controller calculations in the first 100 simulation steps of the control loop since there is still not enough data for a good prediction by the ANN.

Figure 76 presents the results where the emulation model is changed from the nominal to the one for the main rotor fault in the middle of the simulation, but the references and constraints are kept. After the fault identification, the transition is made to the MPC corresponding to the model of the system in the fault state.

Figure 76 – MMPC of Quanser AERO with main rotor fault and MLPNN classification: (a) Angle, and (b) Voltage.

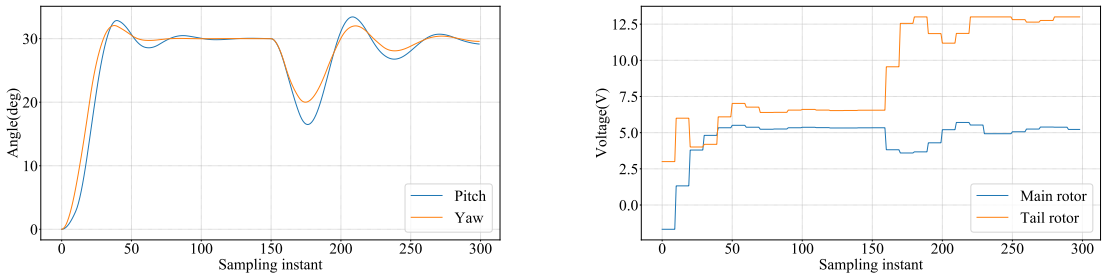


(a) (b)

Source: Author (2021).

Figure 77 presents the results where the emulation model is changed from the nominal to the one for the tail rotor fault in the middle of the simulation.

Figure 77 – MMPC of Quanser AERO with tail rotor fault and MLPNN classification: (a) Angle, and (b) Voltage.



(a) (b)

Source: Author (2021).

Despite its greater accuracy when compared to the FCCNN, the MLPNN could not correctly identify the rotor faults. This might have been caused by overfitting, which would be as if the ANN had just memorized the training set without generalizing, remembering that the training used the open-loop data from the discrete linearized system.

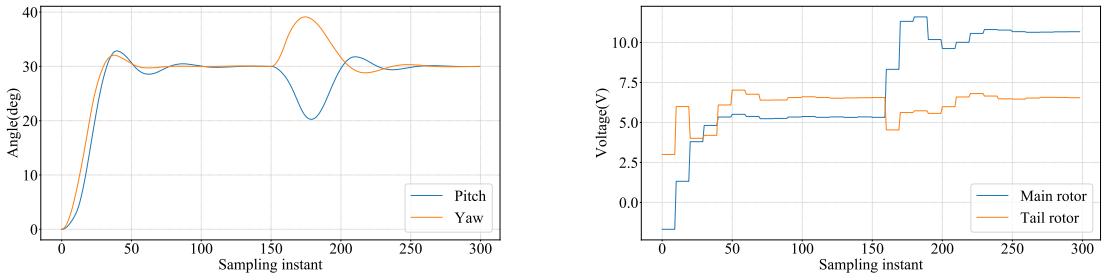
Some tests were carried out with ANNs with lower accuracy, around 80%, and the results obtained were a little closer to the desired one. Therefore, a solution to this problem would be the best treatment of the input data of the ANN, choosing those that best represent the system's behavior. Or yet, to change the shape of the ANN, reducing the number of hidden layers and neurons, for example.

6.3.2.2 Fault identification with fully connected cascade neural network

Similarly, the MMPC algorithm is merged with the FCCNN. Again, it was defined that the system nominal model would be used for the controller calculations in the first 100 simulation steps of the control loop since there is still not enough data for a good prediction by the ANN.

Figure 78 presents the results where the emulation model is changed from the nominal to the one for the main rotor fault in the middle of the simulation, but the references and constraints are kept. After the fault identification, the transition is made to the MPC corresponding to the model of the system in the fault state.

Figure 78 – MMPC of Quanser AERO with main rotor fault and FCCNN classification: (a) Angle, and (b) Voltage.

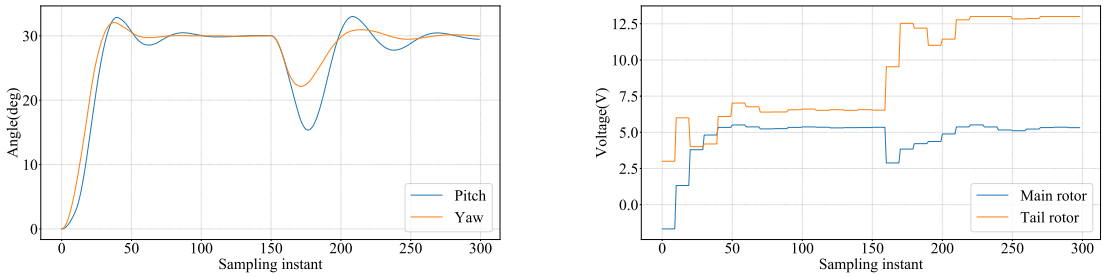


(a) (b)

Source: Author (2021).

Figure 79 presents the results where the emulation model is changed from the nominal to the one for the tail rotor fault in the middle of the simulation.

Figure 79 – MMPC of Quanser AERO with tail rotor fault and FCCNN classification: (a) Angle, and (b) Voltage.



(a) (b)

Source: Author (2021).

The final FCCNN correctly identified both faults most of the time, which was verified through the classification flag. The obtained response was quite similar to that of the ideal case, achieving much better results when compared to the MLPNN despite being less accurate than it.

7 CONCLUSIONS

This work proposed a fault-tolerant controller for didactic aerospace systems, combining the strategies of MMPC and ANNs. The fault treatment is done by changing the prediction model since different representative models are used for the nominal and fault states, characterizing this control strategy as adaptive. The faults considered are mainly abrupt power losses, also, there isn't fault overlap given that each fault occurs from the nominal state.

Assuming that the possible faults are known beforehand and can be represented through a set of equations, either through mathematical modeling or system identification, the obtained results show that the use of a model that represents the particularities of each mode of operation can improve control performance. Furthermore, from the analysis of the forced responses of the system, it is possible to train a neural network capable of identifying these different models, so that the controller can identify the occurrence of a fault and change the prediction model online.

Concluding, this strategy proved to be promising, and since this case study is a linear and time-invariant MIMO system, the code is generic enough to be used in future projects of the same nature, as long as the system is represented in state space. In addition, the proposed control strategy can be applied in non-linear systems as long as the necessary modifications are made.

7.1 CONTRIBUTIONS

The main contributions of this work can be summarized by:

- Analysis of the use of the nominal state-space model predictive controller in the fault systems;
- Analysis of the cases of correct and incorrect fault identification with consequent state-space model predictive controller switch;
- Implementation of multi-model predictive control algorithm with multilayer perceptron artificial neural network;
- Implementation of multi-model predictive control algorithm with fully connected cascade artificial neural network.

7.2 FUTURE WORKS

In order to continue this work, it is proposed:

- Simulate the nonlinear system model;
- Address unknown power losses;
- Address other types of faults;
- Address other types of constraints;
- Apply other types of neural networks;
- Perform experimental tests.

BIBLIOGRAPHY

AFONSO, R. J. M. **Controle preditivo tolerante a falhas de atuador**. Master thesis (Master) — Instituto Tecnológico de Aeronáutica, São José dos Campos, 2012. Cited on page 41.

ÅKESSON, B. M.; TOIVONEN, H. T. A neural network model predictive controller. **Journal of Process Control**, v. 16, n. 9, p. 937–946, 2006. ISSN 0959-1524. Cited on page 35.

ALVES, A. S. C. **Estudo e aplicação de técnicas de controle embarcadas para estabilização de vôo de quadricópteros**. PhD thesis (Doctorate) — Universidade Federal de Juiz de Fora, Juiz de Fora, 2012. Cited on page 48.

ANWAR, A. **Cheat Sheets for Machine Learning Interview Topics**. 2020. <https://medium.com/swlh/cheat-sheets-for-machine-learning-interview-topics-51c2bc2bab4f>. Accessed: 22-0-2021. Cited on page 37.

ARAUJO, P. et al. Multilayer perceptron neural network for flow prediction. **Journal of environmental monitoring : JEM**, v. 13, p. 35–41, 11 2010. Cited on page 39.

BERNHARD, P.; DESCHAMPS, M. Kalman 1960: The birth of modern system theory. **Mathematical Population Studies**, Routledge, v. 26, n. 3, p. 123–145, 2019. Cited on page 21.

CAMACHO, E. F.; BORDONS, C. **Model Predictive Control**. 2. ed. [S.l.]: Springer, 2003. ISBN 1852336943. Cited 4 times on pages 21, 26, 29, and 32.

CASARA, V. L. **Helicóptero 2-DOF: Desenvolvimento e controle das malhas de arfagem e guinada por técnicas PI e LQR**. Bachelor's Thesis — Universidade do Estado de Santa Catarina, Joinville, 2015. Cited on page 41.

CASS, S. **The Top Programming Languages 2019**. 2019. <https://spectrum.ieee.org/computing/software/the-top-programming-languages-2019>. Accessed: 04-09-2020. Cited on page 24.

CAVALCA, M. S. M. **Lecture notes in Model Predictive Control**. Joinville: Universidade do Estado de Santa Catarina, 2019. Cited 2 times on pages 28 and 63.

CHENG, Y. Backpropagation for fully connected cascade networks. **Neural Processing Letters**, v. 46, p. 293–311, 08 2017. Cited on page 40.

CHOLLET, F. et al. **Keras**. 2015. <https://keras.io/>. Accessed: 22-03-2021. Cited on page 38.

CLABAUGH, C.; MYSZEWSKI, D.; PANG, J. **Neural Networks**. 2000. <https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/index.html>. Accessed: 10-07-2021. Cited on page 35.

COOK, M. V. **Flight Dynamics Principles: A Linear Systems Approach to Aircraft Stability and Control**. [S.l.]: Elsevier Science, 2011. (Elsevier aerospace engineering series). ISBN 9780080550367. Cited on page 41.

DATA SCIENCE ACADEMY. **Deep Learning Book**. [S.l.: s.n.], 2019. Https://deeplearningbook.com.br/. Accessed: 21-03-2021. Cited 2 times on pages 37 and 38.

DELASHMIT, W.; MANRY, M. Recent developments in multilayer perceptron neural networks. 05 2021. Cited on page 39.

DI PALMA, F.; MAGNI, L. A multi-model structure for model predictive control. **Annual Reviews in Control**, v. 28, n. 1, p. 47–52, 2004. ISSN 1367-5788. Cited on page 34.

DORF, R. C.; BISHOP, R. H. **Sistemas de controle modernos**. 11. ed. Rio de Janeiro: LTC, 2011. ISBN 9788521617143. Cited on page 21.

ESTRADA-SÁNCHEZ, I.; VELASCO-VILLA, M.; RODRIGUEZ, H. Prediction-based control for nonlinear systems with input delay. **Mathematical Problems in Engineering**, v. 2017, p. 1–11, 10 2017. Cited on page 63.

FAMA, R. C. et al. Predictive control of a magnetic levitation system with explicit treatment of operational constraints. In: . [S.l.: s.n.], 2005. Cited on page 28.

FOURTANÉ, S. **Drones for Search-And-Rescue, Delivery Services Take-off**. 2018. <https://interestingengineering.com/drones-for-search-and-rescue-delivery-services-take-off>. Accessed: 26-04-2021. Cited on page 22.

FRANCO, E.; SACONE, S.; PARISINI, T. Stable multi-model switching control of a class of nonlinear systems. In: . [S.l.: s.n.], 2004. v. 2, p. 1873 – 1878 vol.2. Cited on page 33.

GARCÍA, C. E.; PRETT, D. M.; MORARI, M. Model predictive control: Theory and practice—a survey. **Automatica**, v. 25, n. 3, p. 335–348, 1989. ISSN 0005-1098. Cited on page 21.

GITHUB. **The 2020 State of the Octoverse**. 2020. <https://octoverse.github.com/>. Accessed: 04-09-2020. Cited on page 24.

HARDESTY, L. **Explained: Neural networks**. 2017. <https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414>. Accessed: 21-03-2021. Cited on page 35.

HE, T. et al. Moving average filter-based model predictive control for electric vehicles bidirectional chargers. In: **2017 20th International Conference on Electrical Machines and Systems (ICEMS)**. [S.l.: s.n.], 2017. p. 1–6. Cited on page 63.

HUNG, N. T. et al. Design of multi model predictive control for nonlinear process plant. In: **2014 5th International Conference on Intelligent and Advanced Systems (ICIAS)**. [S.l.: s.n.], 2014. p. 1–6. Cited on page 34.

HUSSAIN, S.; MOKHTAR, M.; HOWE, J. Aircraft sensor estimation for fault tolerant flight control system using fully connected cascade neural network. In: . [S.l.: s.n.], 2013. p. 1–8. ISBN 978-1-4673-6128-6. Cited on page 40.

IJUIM, F. K. **Controle robusto de quadrirotor**. Master thesis (Master) — Universidade do Estado de Santa Catarina, Joinville, 2019. Cited on page 48.

ISERMANN, R. **Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance**. [S.l.]: Springer Berlin Heidelberg, 2006. ISBN 9783540303688. Cited on page 36.

JING, C. J.; SYAFIIE, S. Multi-model generalised predictive control for intravenous anaesthesia under inter-individual variability. **Journal of Clinical Monitoring and Computing**, 2020. Cited on page 33.

KAVLAKOGLU, E. **AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What's the Difference?** 2020. <https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks>. Accessed: 21-03-2021. Cited on page 35.

LANDELS, J. G. **Engineering in the Ancient World**. [S.l.]: Constable, 1998. ISBN 9780094788909. Cited on page 21.

LI, L. **Introduction to Multilayer Neural Networks with TensorFlow's Keras API**. 2019. <https://towardsdatascience.com/introduction-to-multilayer-neural-networks-with-tensorflows-keras-api-abf4f813959>. Accessed: 10-05-2021. Cited on page 39.

MACIEJOWSKI, J. M. **Predictive Control with Constraints**. 1. ed. [S.l.]: Prentice Hall, 2000. ISBN 0201398230. Cited 4 times on pages 21, 26, 27, and 28.

MAIA, M. H. **Controle preditivo robusto de um helicóptero com três graus de liberdade sujeito a perturbações externas**. Master thesis (Master) — Instituto Tecnológico de Aeronáutica, São José dos Campos, 2008. Cited on page 41.

MATHWORKS. **Dynamic Matrix Control Tutorial**. 2008. <https://www.mathworks.com/matlabcentral/mlc-downloads/downloads/submissions/19479/versions/1/previews/html/dmctutorial.html>. Accessed: 15-05-2021. Cited on page 26.

MATHWORKS. **Understanding Model Predictive Control**. 2018. <https://www.mathworks.com/videos/series/understanding-model-predictive-control.html>. Accessed: 15-05-2021. Cited 2 times on pages 22 and 27.

MATOS, M. S. **Controle preditivo com múltiplos modelos para a acomodação de falhas**. Master thesis (Master) — Instituto Tecnológico de Aeronáutica, São José dos Campos, 2008. Cited on page 41.

MAYA, P. A.; FABRIZIO, L. **Controle essencial**. São Paulo: Pearson, 2011. ISBN 9788576057000. Cited 2 times on pages 26 and 28.

NASCIMENTO, G. **Aplicação de controle preditivo nas malhas de arfagem e guinada da planta didática Helicóptero 2-DOF**. Bachelor's Thesis — Universidade do Estado de Santa Catarina, Joinville, 2016. Cited on page 41.

NEGRI, G. H. **Aplicação de métodos de controle preditivo baseado em modelo em um motor de corrente contínua sem escovas**. Bachelor's Thesis — Universidade do Estado de Santa Catarina, Joinville, 2014. Cited on page 30.

NEGRI, G. H. **Algoritmo de controle de caminhada para robôs bípedes utilizando geração de referência por modelo passivo**. PhD thesis (Doctorate) — Universidade do Estado de Santa Catarina, Joinville, 2020. Cited on page 39.

NEGRI, G. H. et al. Evaluation of nonlinear model-based predictive control approaches using derivative-free optimization and fcc neural networks. **Journal of Control Automation and Electrical Systems**, v. 28, 06 2017. Cited on page 39.

NERKHEDE, S. **Understanding Confusion Matrix**. 2018. <https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62>. Accessed: 10-05-2021. Cited on page 76.

NISE, N. S. **Engenharia de sistemas de controle**. 6. ed. Rio de Janeiro: LTC, 2012. ISBN 9788521621355. Cited on page 21.

NOURA, H. et al. **Fault-tolerant Control Systems: Design and Practical Applications**. [S.l.]: Springer London, 2009. (Advances in Industrial Control). ISBN 9781848826533. Cited on page 36.

OGATA, K. **Engenharia de controle moderno**. 5. ed. São Paulo: Pearson, 2010. ISBN 9788576058106. Cited on page 21.

OLIVEIRA Jr, J. G. de. **Model Predictive Control applied to a 2-DOF Helicopter**. Master thesis (Master) — Universidade de São Paulo, São Paulo, 2018. Cited on page 43.

ORUKPE, P. E. Model predictive control fundamentals. **Nigerian Journal of Technology**, v. 31, n. 2, p. 139–148, 2012. Cited 2 times on pages 27 and 28.

PALANIAPPAN, V. **Multilayer Perceptron**. 2018. <https://medium.com/engineer-quant/multilayer-perceptron-4453615c4337>. Accessed: 27-04-2021. Cited on page 39.

PASCOAL, R. M. **Controle preditivo robusto para um helicóptero com três graus de liberdade**. Master thesis (Master) — Instituto Tecnológico de Aeronáutica, São José dos Campos, 2010. Cited on page 41.

PEREIRA, M. **Estudo de técnicas de diagnóstico e controle tolerante a falhas aplicado a robôs manipuladores**. Bachelor's Thesis — Universidade do Estado de Santa Catarina, Joinville, 2019. Cited on page 36.

PICHÉ, S. et al. Neural network based model predictive control. In: **Proceedings of the 12th International Conference on Neural Information Processing Systems**. Cambridge, MA, USA: MIT Press, 1999. (NIPS'99), p. 1029–1035. Cited on page 35.

POPOFF, L. H. G. **Controle preditivo neural aplicado à processos petroquímicos**. Master thesis (Master) — Universidade Federal do Rio Grande do Norte, Natal, 2009. Cited on page 35.

QIAO, J. et al. Constructive algorithm for fully connected cascade feedforward neural networks. **Neurocomputing**, v. 182, 12 2015. Cited on page 40.

QIN, S. J.; BADGWELL, T. A. A survey of industrial model predictive control technology. **Control Engineering Practice**, v. 11, n. 7, p. 733 – 764, 2003. ISSN 0967-0661. Cited 2 times on pages 21 and 22.

QUANSER. **3 DOF Hover**. 2017. <https://www.quanser.com/products/3-dof-hover/>. Accessed: 04-04-2020. Cited 2 times on pages 47 and 48.

QUANSER. **Quanser AERO**. 2017. <https://www.quanser.com/products/quanser-aero/>. Accessed: 23-10-2019. Cited 5 times on pages 41, 42, 43, 44, and 50.

ROSSITER, J. **Model-Based Predictive Control: A Practical Approach**. [S.l.: s.n.], 2003. ISBN 9781315272610. Cited 2 times on pages 26 and 28.

ROWELL, D. **State-Space Representation of LTI Systems**. 2002. <http://web.mit.edu/2.14/www/Handouts/StateSpace.pdf>. Accessed: 07-10-2020. Cited on page 29.

SEBORG, D. E. et al. **Process Dynamics and Control**. 4. ed. [S.l.]: Wiley, 2016. ISBN 9781119298489. Cited on page 27.

SHARMA, S. **Activation Functions in Neural Networks**. 2017. <https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6>. Accessed: 22-04-2021. Cited on page 38.

SHEELA, K Gnana; DEEPA, Subramaniam N. Review on methods to fix number of hidden neurons in neural networks. **Mathematical Problems in Engineering**, Hindawi, v. 2013, 2013. Cited on page 37.

SONAWANE, J.; PATIL, D. R. Prediction of heart disease using multilayer perceptron neural network. **2014 International Conference on Information Communication and Embedded Systems, ICICES 2014**, 02 2015. Cited on page 39.

TASTEMIROV, A.; LECCHINI-VISINTINI, A.; MORALES-VIVIESCAS, R. M. Complete dynamic model of the twin rotor mimo system (trms) with experimental validation. **Control Engineering Practice**, v. 66, p. 89–98, 2017. ISSN 0967-0661. Cited on page 43.

TIOBE Software BV. **TIOBE Index**. 2020. <https://www.tiobe.com/tiobe-index/>. Accessed: 04-09-2020. Cited on page 24.

VAN VEEN, F. **The neural network zoo**. 2016. <https://www.asimovinstitute.org/neural-network-zoo/>. Accessed: 21-03-2021. Cited on page 35.

VAN VEEN, F. **Neural network zoo prequel: Cells and layers**. 2017. <https://www.asimovinstitute.org/author/fjodorvanveen/>. Accessed: 21-03-2021. Cited on page 37.

VYAS, K. **A Brief History of Drones: The Remote Controlled Unmanned Aerial Vehicles**. 2020. <https://interestingengineering.com/a-brief-history-of-drones-the-remote-controlled-unmanned-aerial-vehicles-uavs>. Accessed: 01-07-2020. Cited on page 22.

ZHANG, L.; ZHUANG, S.; BRAATZ, R. D. Switched model predictive control of switched linear systems: Feasibility, stability and robustness. **Automatica**, v. 67, p. 8–21, 2016. ISSN 0005-1098. Cited on page 34.

ZHANG, R.; WU, S.; GAO, F. State space model predictive control for advanced process operation: A review of recent development, new results, and insight. **Industrial & Engineering Chemistry Research**, v. 56, n. 18, p. 5360–5394, 2017. Cited on page 26.

APPENDIX A – MODEL PREDICTIVE CONTROL ALGORITHM

This appendix contains parts of the source code referring to the implementation of the predictive controller.

A.1 PREDICTION MODEL WITH INPUT INCREMENTS

A.1.1 Control parameters

```
def ParametersMPC(Ad,Bd,Cd,Dd,n,p,q,N,M,py,pu):

    ## matrices
    Aksi = np.block([[Ad,Bd],[np.zeros((p,n)),np.identity(p)]])
    Bksi = np.block([[Bd],[np.identity(p)]])
    Cksi = np.block([Cd,np.zeros((q,p))])

    ## G
    Gaux = Cksi@Bksi
    for i in range(1,N):
        Gaux = np.block([[Gaux],[Cksi@matrix_power(Aksi,i)@Bksi]])
    # end for
    G = Gaux
    if M > 1:
        for i in range(1,M):
            G = np.block([G,np.block([[np.zeros((q*i,p))],[Gaux[:((N*q)-(q*i)),:]]])])
        # end for
    # end if

    ## phi
    phi = Cksi@Aksi
    for i in range(2,N+1):
        phi = np.block([[phi],[Cksi@matrix_power(Aksi,i)]])
    # end for

    ## Py
    Py = py*np.identity(q*N)

    ## Pu (!= 0)
    Pu = pu*np.identity(p*M)
```

```
    ## Kmpc
    kaux = np.linalg.inv((G.transpose()@Py@G) + Pu)@G.transpose()@Py
    kmpc = kaux[:p,:]

    return G,phi,Py,Pu,kmpc
```

A.1.2 Quadratic programming parameters

```
def ParametersQP(n,p,q,N,M,G,Py,Pu):

    ## Gn
    Gn = Py@G

    ## Hqp
    Hqp = 2*((G.transpose()@Py@G) + Pu)

    ## I
    IpM = np.identity(p*M)

    ## T
    Ip1 = np.matlib.repmat(np.identity(p),M,1)
    TM = Ip1
    if M > 1: # TMp
        for i in range(1,M):
            TM = np.block([TM,np.block([[np.zeros((p*i,p))],[Ip1[:((M*p)-(p*i)),:]]])])
        # end for
    # end if

    ## Aqp = S
    Aqp = np.vstack((IpM,-IpM,TM,-TM,G,-G))

    return Gn,Hqp,Aqp
```

A.1.3 Control loop of the unconstrained predictive controller

```
if ke==0 or ke%fsim==0:

    ## count kc
    kc = kc + 1
```

```
    ## ksi
    ksi = np.block([x[ke],u2[ke−1]])
    ksi = ksi.reshape(n+p,1) # column vector

    # free system response
    f = phi@ksi

    # delta u
    du = kmpc@(r−f)
    du = du.reshape(1,p) # row vector

    # u
    if kc==1:
        u1[kc] = du
    else:
        u1[kc] = u1[kc−1] + du
    # end if−else
```

A.1.4 Control loop of the constrained predictive controller

```
if ke==1 or ke%fsim==0:

    ## count kc
    kc = kc + 1

    ## ksi
    ksi = np.block([x[ke],u2[ke−1]])
    ksi = ksi.reshape(n+p,1) # column vector

    ## free system response
    f = phi@ksi

    ### quadratic programming

    ## fqp
    fqp = 2∗Gn.transpose()@(f−r)

    try:
```

```
        ## bqp
        if kc == 1:
            bqp = np.vstack((np.matlib.repmat(dumax,M,1),-np.matlib.repmat(dumin,M,1),
                ↪ np.matlib.repmat((umax-np.zeros((p,1))),M,1),np.matlib.repmat((np.
                ↪ zeros((p,1))-umin),M,1),(np.matlib.repmat(ymax,N,1)-f),(f-np.matlib.
                ↪ repmat(ymin,N,1))))
        else:
            bqp = np.vstack((np.matlib.repmat(dumax,M,1),-np.matlib.repmat(dumin,M,1),
                ↪ np.matlib.repmat((umax-u1[kc-1].reshape((p,1))),M,1),np.matlib.
                ↪ repmat((u1[kc-1].reshape((p,1))-umin),M,1),(np.matlib.repmat(ymax,N
                ↪ ,1)-f),(f-np.matlib.repmat(ymin,N,1))))

        ## solve opt problem
        sol = solvers.qp(matrix(Hqp),matrix(fqp),matrix(Aqp),matrix(bqp))
        if sol['status'] == 'unknown':
            raise ValueError()

        ## delta u
        du = np.asarray(sol['x'])
        du = du[:p].reshape(1,p)

    except ValueError: # no feasible solution
        du = np.zeros((1,p))
    # end try-except

    ## u
    if kc == 1:
        u1[kc] = du
    else:
        u1[kc] = u1[kc-1] + du
    # end if-else
```

A.1.5 System emulation

```
x[ke+1] = Ae@x[ke] + Be@u1[kc]
y[ke+1] = Ce@x[ke+1]
u2[ke] = u1[kc]
```

A.2 PREDICTION MODEL WITH INTEGRAL CONTROL ACTION

A.2.1 Control parameters

```
def ParametersMPC(Ad,Bd,Cd,Dd,n,p,q,N,M,py,pu):

    ## add integral action
    Aksi = np.block([[Ad,np.zeros((n,q))],[Cd@Ad,np.identity(q)]])
    Bksi = np.block([[Bd],[Cd@Bd]])
    Cksi = np.block([np.zeros((q,n)),np.identity(q)])

    ## G
    Gaux = Cksi@Bksi
    for i in range(1,N):
        Gaux = np.block([[Gaux],[Cksi@matrix_power(Aksi,i)@Bksi]])
    # end for
    G = Gaux
    if M > 1:
        for i in range(1,M):
            G = np.block([G,np.block([[np.zeros((q*i,p))],[Gaux[:((N*q)-(q*i)),:]]])])
        # end for
    # end if

    # phi
    phi = Cksi@Aksi
    for i in range(2,N+1):
        phi = np.block([[phi],[Cksi@matrix_power(Aksi,i)]])
    # end for

    ## Py
    Py = py*np.identity(q*N)

    ## Pu (!= 0)
    Pu = pu*np.identity(p*M)

    ## Kmpc
    kaux = np.linalg.inv((G.transpose()@Py@G) + Pu)@G.transpose()@Py
    kmpc = kaux[:p,:]

    return G,phi,Py,Pu,kmpc
```

A.2.2 Quadratic programming parameters

```
def ParametersQP(n,p,q,N,M,G,Py,Pu):

    ## Gn
    Gn = Py@G

    ## Hqp
    Hqp = 2*((G.transpose()@Py@G) + Pu)

    ## I
    IpM = np.identity(p*M)

    ## T
    Ip1 = np.matlib.repmat(np.identity(p),M,1)
    TM = Ip1
    if M > 1: # TMp
        for i in range(1,M):
            TM = np.block([TM,np.block([[np.zeros((p*i,p))],[Ip1[:((M*p)-(p*i)),:]]])])
        # end for
    # end if

    ## Aqp = S
    Aqp = np.vstack((IpM,-IpM,TM,-TM,G,-G))

    return Gn,Hqp,Aqp
```

A.2.3 Control loop of the unconstrained predictive controller

```
if ke==0 or ke%fsim==0:

    ## count kc
    kc = kc + 1

    ## ksi
    dx = x[ke] - xpast # update dx
    xpast = x[ke] # save past x
```

```
    ksi = np.block([dx,y[ke]]) # ksi
    ksi = ksi.reshape(q+n,1) # column vector

    # free system response
    f = phi@ksi

    # delta u
    du = kmpc@(r-f)
    du = du.reshape(1,p) # row vector

    # u
    if kc==1:
        u1[kc] = du
    else:
        u1[kc] = u1[kc-1] + du
    # end if-else
```

A.2.4 Control loop of the constrained predictive controller

```
### loop de controle
if ke==0 or ke%fsim==0:

    ## count kc
    kc = kc + 1

    ## ksi
    dx = x[ke] - xpast # update dx
    xpast = x[ke] # save past x
    ksi = np.block([dx,y[ke]]) # ksi
    ksi = ksi.reshape(q+n,1) # column vector

    ## free system response
    f = phi@ksi

    ### quadratic programming

    ## fqp
    fqp = 2*Gn.transpose()@(f-r)
```

```
try:
    ## bqp
    if kc == 1:
        bqp = np.vstack((np.matlib.repmat(dumax,M,1),−np.matlib.repmat(dumin,M,1),
            ↪ np.matlib.repmat((umax−np.zeros((p,1))),M,1),np.matlib.repmat((np.
            ↪ zeros((p,1))−umin),M,1),(np.matlib.repmat(ymax,N,1)−f),(f−np.matlib.
            ↪ repmat(ymin,N,1))))
    else:
        bqp = np.vstack((np.matlib.repmat(dumax,M,1),−np.matlib.repmat(dumin,M,1),
            ↪ np.matlib.repmat((umax−u1[kc−1].reshape((p,1))),M,1),np.matlib.
            ↪ repmat((u1[kc−1].reshape((p,1))−umin),M,1),(np.matlib.repmat(ymax,N
            ↪ ,1)−f),(f−np.matlib.repmat(ymin,N,1))))

    ## solve opt problem
    sol = solvers.qp(matrix(Hqp),matrix(fqp),matrix(Aqp),matrix(bqp))
    if sol['status'] == 'unknown':
        raise ValueError()

    ## delta u
    du = np.asarray(sol['x'])
    du = du[:p].reshape(1,p)

except ValueError: # no feasible solution
    du = np.zeros((1,p))
# end try−except

## u
if kc == 1:
    u1[kc] = du
else:
    u1[kc] = u1[kc−1] + du
```

APPENDIX B – ARTIFICIAL NEURAL NETWORK ALGORITHM

This appendix contains parts of the source code referring to the implementation of the neural networks.

B.1 MULTILAYER PERCEPTRON NEURAL NETWORK TRAINING

```
def MLP(input,x_train,y_train,x_evaluate,y_evaluate):

    # build model
    visible = Input(shape=(input,)) # first layer = input layer
    hidden1 = Dense(8,activation='relu')(visible)
    hidden2 = Dense(4,activation='relu')(hidden1)
    hidden3 = Dense(4,activation='relu') (hidden2)
    output = Dense(1,activation='sigmoid')(hidden3) # last layer = output layer

    model = Model(inputs=visible, outputs=output)

    # train ann
    model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy']) #
        ↪ default

    history = model.fit(x_train,y_train,epochs=100,validation_data=(x_evaluate,y_evaluate))
        ↪ # default

    return model, history
```

B.2 FULLY CONNECTED CASCADE NEURAL NETWORK TRAINING

```
def FCC(input,x_train,y_train,x_evaluate,y_evaluate):

    nr_input_units = input
    nr_output_units = 1
    nr_of_neurons = 1
    nr_of_layers = 3

    hidden_activation = 'relu'
    output_activation = 'sigmoid'
```

```
    nr_epochs = 100
    batch_size = 32

    visible = Input(shape=(nr_input_units,))
    hidden1 = Dense(nr_of_neurons, activation=hidden_activation)(visible)
    fork = [visible, hidden1]

    for idx in range(nr_of_layers):
        merge = concatenate(fork)
        hidden = Dense(nr_of_neurons, activation=hidden_activation)(merge)
        fork.append(hidden)
    # end for

    output = Dense(nr_output_units, activation=output_activation)(merge)

    model = Model(inputs=visible, outputs=output)

    model.compile(optimizer='adam', loss='mean_squared_error', metrics=['accuracy'])

    history = model.fit(x_train, y_train, epochs=nr_epochs, batch_size=batch_size, verbose
        ↪ =1,validation_data=(x_evaluate,y_evaluate))

    return model, history
```

B.3 MODEL VALIDATION

```
def Evaluate(model,history,x_evaluate,y_evaluate):

    # print loss and accuracy values
    val_loss,val_acc = model.evaluate(x_evaluate,y_evaluate)
    print(val_loss,val_acc)

    # plot loss and accuracy
    plt.figure(1)

    # summarize history for accuracy
    plt.subplot(211)
    plt.plot(history.history['accuracy'])
    plt.plot(history.history['val_accuracy'])
```

```
#plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epochs')
plt.legend(['Training', 'Validation'], loc='lower␣right')

# summarize history for loss
plt.subplot(212)
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
#plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epochs')
plt.legend(['Training', 'Validation'], loc='upper␣right')

plt.tight_layout()

plt.show()
```