UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Dept. of Systems Engineering and Automation

Design and Implementation of angle-of-attack control system using active disturbance rejection control algorithms for fixed-wing unmanned aerial vehicles.

Master's Thesis

Master's Degree in Automation and Industrial Computing

AUTHOR: Musoles Aguña, José Luis

Tutor: García-Nieto Rodríguez, Sergio

ACADEMIC YEAR: 2023/2024

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCOLA TÈCNICA SUPERIOR DE ENGINYERIA INDUSTRIAL

Máster en Automática e Informática Industrial

“Design and Implementation of Angle-of-Attack Control System Using Active Disturbance Rejection Control Algorithms for Fixed-Wing Unmanned Aerial Vehicles”

FINAL MASTER'S THESIS
Memory

Autor:
José Luis Musoles Aguña

Profesor:
Sergio García-Nieto Rodríguez

València; 2023 - 2024

Resumen

El Trabajo de Final de Master que se presenta a continuación cumple el objetivo de presentar, diseñar y validar un nuevo algoritmo de control, basado en la combinación de un controlador predictivo (MPC) con un controlador que rechaza perturbaciones (ADRC). Este nuevo diseño busca solucionar una problemática específica: reducir al máximo el modelado necesario para el diseño de controladores. Las leyes de control tradicionales se basan en tener un model preciso, bien para diseñarlos o predecir el comportamiento de la planta. Esta nueva ley de control busca reducir el trabajo de modelado del proceso a contoolar.

Con este objetivo en mente, se validará el nuevo controlador para governar el ángulo de ataque de una aeronave no tripulada. Para ello, se modelará toda la dinámica de la aeronave, se aplicarán los parámetros específicos de geometría, modelo propulsivo, aerodinámico... Mediante métodos semi-empíricos y datos experimentales, se ha conseguido toda esta información, haciendo posible conseguir un modelo no-lineal de la aeronave. A partir de ahí, se puede construir el proceso del ángulo de ataque.

Con el modelo lineal, se ha construido la nueva ley de control, junto a otros dos algoritmos a modo de comparación. Una vez diseñados y tuneados los controladores, se han validado frente al modelo no-lineal completo, primero en condiciones de vuelo en crucero, y después frente a perturbaciones de diversa tipología (estructural, propulsiva, de entradas...). Con estas pruebas, se ha podido comprobar la validez de la nueva ley de control, capaz de rechazar perturbaciones externas al no depender del modelo del proceso directmente.

Palabras Clave: Ecuaciones de *Bryan*, UAV, PID, MPC, ADRC, Ala fija, algoritmo de control, perturbaciones

Resum

El Treball de Final de Màster que es presenta a continuació complix l'objectiu de presentar, dissenyar i validar un nou algorisme de control, basat en la combinació d'un controlador predictiu (*MPC) amb un controlador que rebutja pertorbacions (ADRC). Este nou disseny busca solucionar una problemàtica específica: reduir al màxim el modelatge necessari per al disseny de controladors. Les lleis de control tradicionals es basen a tindre un model precís, bé per a dissenyar-los o predir el comportament de la planta. Esta nova llei de control busca reduir el treball de modelatge del procés a controlar.

Amb este objectiu en ment, es validarà el nou controlador per a governar l'angle d'atac d'una aeronau no tripulada. Per això, es modelarà tota la dinàmica de l'aeronau, s'aplicaran els paràmetres específics de geometria, model propulsiu, aerodinàmic... Mitjançant mètodes semi-empírics i dades experimentals, s'ha aconseguit tota esta informació, fent possible aconseguir un model no-lineal de l'aeronau. A partir d'ací, es pot construir el procés de l'angle d'atac.

Amb el model lineal, s'ha construït la nova llei de control, al costat d'altres dos algorismes a mode de comparació. Una vegada dissenyats i tunnejats els controladors, s'han validat enfront del model no-lineal complet, primer en condicions de vol en creuer, i després enfront de pertorbacions de diversa tipologia (estructural, propulsiva, d'entrades...). Amb estes proves, s'ha pogut comprovar la validesa de la nova llei de control, capaç de rebutjar pertorbacions externes al no dependre del model del procés directmente.

Paraules Clau: Ecuacions de *Bryan*, UAV, PID, MPC, ADRC, ala fixa, algorisme de control, perturbacions

Abstract

The Final Master's Thesis here presented meet with the goal of presenting, design and validate a new control algorithm, based on the combination of a predictive controller (MPC) with a disturbance rejector controller (ADRC). This new design has the objective of solving a very specific problematic: minimize the modeling necessary for the controllers design. Traditional control laws are based on having a precise model, either for the design phase or predict the process behavior. This new control law has the objective of reducing the modeling work of the controlled process.

With this goal in mind, the controller has been validated for governing the angle of attack of an unmanned aircraft vehicle. For this, it has been modeled all the aircraft dynamics and it has been applied the specific parameters of geometry, propeller model, aerodynamics... Through semi-empirics methods and experimental data, it has been achieved all of this information, making possible developing an aircraft non-linear model. From this point, it has been built the the angle of attack process.

With the process lineal model, it has been developing the new control law, apart from two traditional control algorithms as a form of comparison. Once designed and tuned the controllers, it has been validated with the complete non-linear model, first in cruise flight conditions and then with a diverse typology of perturbations (structural, propulsive, inputs...). With these tests, it has been tested the validity of the new control algorithm, capable of rejecting external perturbations as not depending directly from the model.

Keywords: *Bryan* equations, UAV, PID, MPC, ADRC, fixed-wing, control algorithm, perturbations.

Agradecimientos

Con la presentación de este proyecto final de máster, otra etapa de estudiante en la universidad se termina, aunque el aprendizaje es constante durante toda la vida. Y como toda etapa, hay una infinidad de personas por las que querría dar las gracias.

Primero de todo, la mayor sorpresa de este 2024, la mejor persona que he conocido nunca, el amor de mi vida, mi *smooshie muffin*. La persona más fantástica, inteligente, que escucha y que quiero con locura. Eres la niña de mis ojos y te amaré hasta el final de los tiempos. Muchas gracias por todos los momentos que llevamos y los que nos esperan.

En segundo, a mi familia, que con mi traslado a Madrid siguieron apoyándome en todas mis decisiones, desde que tengo uso de razón hasta el día de hoy. Como el trabajo de final de grado, son los principales responsables de la presentación de este trabajo, por su infinita paciencia y ayuda.

Además, agradezco a mi familia que escojo, a todos mis amigos que me han ayudado y me han soportado durante toda esta aventura. Primero, a Alejandra, la mejor compañera de piso que pudiera desear, y la mejor enfermera que conozco en Madrid, gracias por todos los chismes y los días del piseto. Al Ldo. Arturo, mi mejor amigo, mi hermano, al que le agradezco tantas cosas que no se ni por donde empezar. A mis compañeros de GMV, por acogerme en la empresa con los brazos abiertos y los planes que hemos podido hacer fuera de ella. Y como olvidarme de mis amigos de las “espaditas", mis grandes amigos del clan Spec-3: Mai, Pavlo, Alicia, Nuria, David, Laurine, Rafa, Esther, María, Lucía y Cris; y a nuestro gran entrenador, maestro y sensei, Alex; y todo Ludosport Valencia y Hispania, por introducirme en este gran deporte y hacerme sentir una gran pasión por él.

Y por último, pero no menos importante, a mi tutor Sergio, que me ha estado acompañando tanto en el trabajo final de grado y en este trabajo final de máster, y los trabajos que nos queden por delante. Por su paciencia y comprensión conmigo. Por la oportunidad que me brindó con el máster, las prácticas en el laboratorio en el departamento, y por abrirme la puerta a la investigación en este campo de la automática.

Contents

List of Figures

List of Tables

List of Symbols

Symbol	**Units**	**Description**
$\hat{X}_W$	-	X-axis of the wind reference frame
$\hat{Y}_W$	-	Y-axis of the wind reference frame
$\hat{Z}_W$	-	Z-axis of the wind reference frame
$\hat{X}_B$	-	X-axis of the body reference frame
$\hat{Y}_B$	-	Y-axis of the body reference frame
$\hat{Z}_B$	-	Z-axis of the body reference frame
$\hat{X}_E$	-	X-axis of the local reference frame
$\hat{Y}_E$	-	Y-axis of the local reference frame
$\hat{Z}_E$	-	Z-axis of the local reference frame
CoG	-	Center of gravity
u	m/s	Linear body velocity in X-axis
v	m/s	Linear body velocity in Y-axis
w	m/s	Linear body velocity in Z-axis
p	rad/s	Angular body velocity in $\hat{X}_B$
q	rad/s	Angular body velocity in $\hat{Y}_B$
r	rad/s	Angular body velocity in $\hat{Z}_B$
F_x	N	Total force in $\hat{X}_B$
F_y	N	Total force in $\hat{Y}_B$
F_z	N	Total force in $\hat{Z}_B$
L	$N \cdot m$	Angular momentum in $\hat{X}_B$
M	$N \cdot m$	Angular momentum in $\hat{Y}_B$
N	$N \cdot m$	Angular momentum in $\hat{Z}_B$
W	-	Wind reference frame
B	-	Body reference frame
N	-	Local reference frame
$[WB]$	-	Attitude matrix between wind and body frames
$[BN]$	-	Attitude matrix between body and inertial frames
α	rad	Angle of attack
β	rad	Slip angle
$\vec{F}$	N	Total force acting on body vector
$\dot{\vec{V}}$	m/s^2	Linear acceleration acting on body vector
m	kg	Aircraft mass
${}^N\omega^B$	rad/s	Angular velocity between body and inertial frame
${}^B\vec{F}_G$	N	Body force caused by the gravity
${}^B\vec{F}_A$	N	Body force caused by the aerodynamic force
${}^B\vec{F}_T$	N	Body force caused by the propeller thrust
g	m/s^2	Gravity vector
T_x	N	Component X of the propeller force
T_y	N	Component Y of the propeller force
T_z	N	Component Z of the propeller force
$T(\delta_P)$	N	Propeller force dependent on the throttle level
δ_P	-	Throttle level

ρ	kg/m^3	Air density
V	m/s	Flight airspeed
S_w	m^2	Wing reference surface
C_D	-	Aerodynamic drag coefficient
C_Y	-	Aerodynamic lateral force coefficient
C_L	-	Aerodynamic lift force coefficient
$R_z(\alpha)$	-	Rotation matrix around Z-axis with an angle of α
$R_y(\alpha)$	-	Rotation matrix around Y-axis with an angle of α
${}^{CoG}\dot{\vec{H}}$	$N \cdot m/s$	Angular momentum around the Center of Gravity
${}^{CoG}[I]$	kg/m^3	Body inertia matrix
I_{xx}	kg/m^3	Body inertia around the X-axis
I_{yy}	kg/m^3	Body inertia around the Y-axis
I_{zz}	kg/m^3	Body inertia around the Z-axis
I_{xy}	kg/m^3	Cross body inertia between X-axis and Y-axis
I_{xz}	kg/m^3	Cross body inertia between X-axis and Z-axis
I_{yz}	kg/m^3	Cross body inertia between Y-axis and Z-axis
${}^{B}\vec{M}_T$	$N \cdot m$	Moment around the CoG caused by the propeller force
${}^{B}\vec{M}_A$	$N \cdot m$	Moment around the CoG caused by the aerodynamic forces
$\vec{r}_{prop}$	m	Propeller arm distance vector from the CoG
L_{prop}	m	Propeller arm distance from the CoG
b_w	m	Reference aircraft wingspan
C_l	-	X-axis aerodynamic moment coefficient
C_m	-	Y-axis aerodynamic moment coefficient
C_n	-	Z-axis aerodynamic moment coefficient
ψ	rad	Yaw angle
θ	rad	Pitch angle
ϕ	rad	Euler angle correspondent to the roll angle
$\dot{\psi}$	rad/s	Yaw angle rate
$\dot{\theta}$	rad/s	Pitch angle rate
$\dot{\phi}$	rad/s	Roll angle rate
q_0	-	Real quaternion component
q_1	-	First imaginary quaternion component
q_2	-	Second imaginary quaternion component
q_3	-	Third imaginary quaternion component
$\dot{q}_0$	-	Real quaternion component rate
$\dot{q}_1$	-	First imaginary quaternion component rate
$\dot{q}_2$	-	Second imaginary quaternion component rate
$\dot{q}_3$	-	Third imaginary quaternion component rate
x	m	Aircraft X-axis position in local frame
y	m	Aircraft Y-axis position in local frame
z	m	Aircraft Z-axis position in local frame
$\dot{x}$	m/s	Local frame X-axis velocity
$\dot{y}$	m/s	Local frame Y-axis velocity
$\dot{z}$	m/s	Local frame Z-axis velocity
L_{elev}	m	Elevator arm distance from CoG
T	N	Engine thrust
P	W	Engine power

C_T	-	Engine thrust coefficient
C_P	-	Engine power coefficient
n	Hz	Propeller angular velocity
D	m	Blade diameter
J	-	Dimensionless velocity
$p_{C_{T_0}}$	-	First coefficient for propeller thrust coefficient linear model
$p_{C_{T_1}}$	-	Second coefficient for propeller thrust coefficient linear model
$p_{C_{T_2}}$	-	Third coefficient for propeller thrust coefficient linear model
$p_{C_{T_3}}$	-	Fourth coefficient for propeller thrust coefficient linear model
$p_{C_{P_0}}$	-	First coefficient for propeller power coefficient linear model
$p_{C_{P_1}}$	-	Second coefficient for propeller power coefficient linear model
$p_{C_{P_2}}$	-	Third coefficient for propeller power coefficient linear model
$p_{C_{P_3}}$	-	Fourth coefficient for propeller power coefficient linear model
$p_{C_{P_4}}$	-	Fifth coefficient for propeller power coefficient linear model
$p_{C_{P_5}}$	-	Sixth coefficient for propeller power coefficient linear model
$p_{C_{P_6}}$	-	Seventh coefficient for propeller power coefficient linear model
C_{D_0}	-	Zero lift aerodynamic drag coefficient
A_1	-	Linear lift aerodynamic drag coefficient
A_{Polar}	-	Polar aerodynamic drag coefficient
C_{Y_β}	-	Lateral force coefficient dependent on the sideslip angle
C_{Y_p}	-	Lateral force coefficient dependent on the roll rate
C_{Y_r}	-	Lateral force coefficient dependent on the yaw rate
$C_{Y_{\delta_A}}$	-	Lateral force coefficient dependent on the aileron deflection
δ_A	-	Aileron deflection
$C_{Y_{\delta_R}}$	-	Lateral force coefficient dependent on the rudder deflection
δ_R	-	Rudder deflection
C_{L_0}	-	Constant aerodynamic lift for the airfoil
C_{L_α}	-	Lift coefficient dependent on the angle of attack
$C_{L_{\dot\alpha}}$	-	Lift coefficient dependent on the angle of attack rate
C_{L_q}	-	Lift coefficient dependent on the pitch rate
$C_{L_{\delta_E}}$	-	Lift coefficient dependent on the elevator deflection
δ_E	-	Elevator deflection
$C_{L_{\delta_F}}$	-	Lift coefficient dependent on the flaps deflection
δ_F	-	Flaps deflection
C_{l_β}	-	Aerodynamic moment coefficient around the X-axis dependent on the sideslip
C_{l_p}	-	Aerodynamic moment coefficient around the X-axis dependent on the roll rate
C_{l_r}	-	Aerodynamic moment coefficient around the X-axis dependent on the yaw rate
$C_{l_{\delta_A}}$	-	Aerodynamic moment coefficient around the X-axis dependent on the ailerons
$C_{l_{\delta_R}}$	-	Aerodynamic moment coefficient around the X-axis dependent on the rudder
C_{m_0}	-	Constant aerodynamic moment coefficient around the Y-axis
C_{m_α}	-	Aerodynamic moment coefficient around the Y-axis dependent on the angle of atta
$C_{m_{\dot\alpha}}$	-	Aerodynamic moment coefficient around the Y-axis dependent on the angle of atta
C_{m_q}	-	Aerodynamic moment coefficient around the Y-axis dependent on the pitch rate
c_w	m	Mean aerodynamic chord
$C_{m_{\delta_E}}$	-	Aerodynamic moment coefficient around the Y-axis dependent on the elevator
$C_{m_{\delta_F}}$	-	Aerodynamic moment coefficient around the Y-axis dependent on the flaps
C_{n_β}	-	Aerodynamic moment coefficient around the Z-axis dependent on the sideslip

C_{n_p}	-	Aerodynamic moment coefficient around the Z-axis dependent on the roll rate
C_{n_r}	-	Aerodynamic moment coefficient around the Z-axis dependent on the yaw rate
$C_{n_{\delta_A}}$	-	Aerodynamic moment coefficient around the Z-axis dependent on the aileron
$C_{n_{\delta_R}}$	-	Aerodynamic moment coefficient around the Z-axis dependent on the rudder
$r(t)$	-	Reference signal
$e(t)$	-	Tracking error signal
$u(t)$	-	Control input signal
$y(t)$	-	Response signal
K_P	-	Proportional constant for the PID controller
K_I	-	Integral constant for the PID controller
K_D	-	Derivative constant for the PID controller
ξ	-	Flight bearing
$\boldsymbol{A}$	-	State space model matrix corresponding to the system dynamic
$\boldsymbol{B}$	-	State space model matrix corresponding to the sytem inputs
$\boldsymbol{C}$	-	State space model matrix corresponding to the system output
$\boldsymbol{D}$	-	State space model matrix corresponding to the system output from the inputs
$\vec{x}$	-	State space model state vector
$\vec{u}$	-	State space model input vector
ISE	-	Integral square error performance index
T_{sim}	-	Simulation time
T_s	-	Simulation sample time
$\theta_r(t)$	-	Reference pitch angle signal
δ	-	Response overshoot
T_{set}	-	Response settling time
p	-	Predictive controller prediction horizon
c	-	Predictive controller control horizon
Q_i	-	Predictive controller error tracking cost value
R_i	-	Predictive controller control input rate cost value
b_0	-	Critical nominal value gain
ω_o	-	Response bandwidth

1 Introduction

The aircraft industry has changed the idea of transportation, connecting far-apart regions in just a couple of hours. From the last century, up until the present, aircraft technology has exploded, from its invention in 1903 by the Wright brothers [3], up until the powerful jet-propelled aircraft used today for people and goods transportation. All of these advances have been possible with the rapid development in aircraft systems. One of the many fields of fast progress is the one related with the attitude control of the aircraft.

Since the early days of aviation, (conventional) aircraft control systems have trusted in the deflection of certain surfaces from their wings [13]. Through these defections, the aircraft is capable of turning with respect the its three principal axis. In general flight mechanics, these three axis take as origin the center of gravity and the X-axis pass through the aircraft nose, the Z-axis pointing down to the ground and the Y-axis pointing to the right wing:

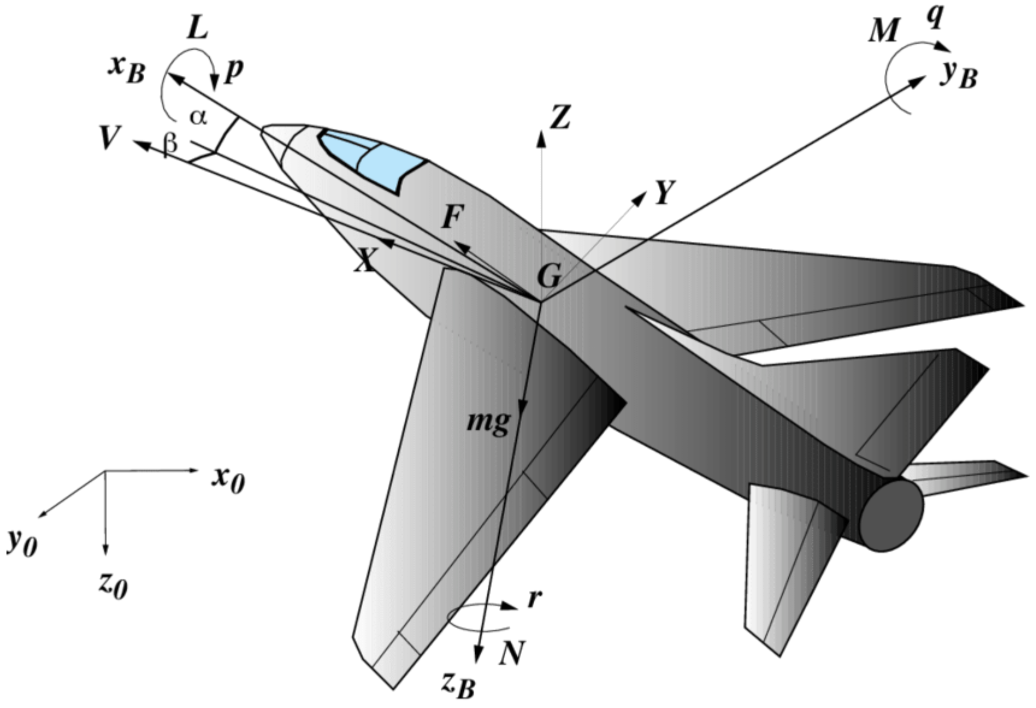


Figure 1: Body forces, moments and aircraft axis [36]

These control systems have had a very rapid development within the last century [34]. The early days of control systems carried out the pioneering work for the latter technologies to be developed, in order to meet with the demands of speed, maneuverability, maximum altitude. . . World War II and early cold war systems trusted in hydro-mechanics actuation in order to deflect the aircraft surfaces [34]. Nevertheless, as time passed, and the weight requirements were tightened, it was needed a new system for controlling them, hence the introduction of the fly-by-wire systems [34].

The difference between these two systems being that in the first one, the pilot control devices were connected directly to the control surfaces, via rods, levers, cables and pulleys, meaning that the pilot had direct control with the aircraft control surfaces [24]. In the second one, both in fly-by-wire and fly-by-light systems, the pilot control devices are connected to a flight computer, which is the one calculating the control actions and then send them out to the aircraft control systems [34] [24], meaning that there is a layer of abstraction between the pilot and the control surfaces. Both of them work via a flight computer, with the fly-by-wire systems running through conventional wiring inside the aircraft and the fly-by-light by using optical fibers.

For this objective, and with the introduction of the on-board flight computer, it is needed the precise control for the aircraft pitch angle. From figure (1), it's the angle between the aircraft X axis and the local inertial frame. This is required for maintaining cruise flight conditions (altitude, attitude, velocity, etc), as well as for ascents and descents operations [2]. In the past, pilots were capable of performing such tasks, but with more complex technology being developed for the aircraft, the fly-by-wire structure started to make his mark as the autopilot computer could perform these operations instead of the pilots [2].

Thus, the necessity of introducing control engineering inside the aircraft loop, as a way of connecting the pilot flight operation tasks and necessities into aircraft system inputs, so the onboard autopilot computer can calculate the required control surfaces deflections in order to achieve such operations. The controllers inside the autopilot computer are of paramount importance, as they are the responsible for the correct govern of the flight state variables, which are the ones responsible of a controller and nominal flight operation. There exists a lot of options for controllers to be included inside the autopilot, varying in complexity and demanding operative capabilities: PID controller [2], pole placement [1], intelligent control [38], predictive control, LQR, etc

The aircraft system is a high non-linear system, so the choosing of the controllers inside the autopilot will be relevant, concerning the amount of limitations they introduce to the system, as they could fail of not designed properly. That is why, during the controllers design and tuning phases, it is required an accurate mathematical model of the aircraft, in order to model every aspect fo the aircraft as accurately and precise as possible, meaning that the controller design and validation will not fail when connected to the real world.

This carries an evident problem, which is the actual model of the aircraft. There exists the general flight mechanic equations to model the flight of a rigid body, but how the forces and moments are calculated inside the aircraft come from semi-empirical methods and/or experimental data, so in some cases the model is not accessible or not accurate model due to lack of means, information or a combination of both. This results in having a partial model of the aircraft variable that want to be controlled, so some of the controllers will fail to meet the required demands as they need an accurate model of the variable to be controlled.

This project will tackle this specific problem, on how to design a controller with limited information on the controlled plant.

2 Objective and Scope of the Project

2.1 Objective of the Project

The project's main goal is to validate a new control algorithm, the *Modified Active Disturbance Rejection Predictive Control* (MADRPC), and compare its performance against more traditional control schemes. The main focus is to test how the MADRPC, which does not use the full plant for controlling, apart from two natural parameters, compares with more traditional forms of control, which use the full information from the system for the control process.

For this objective, it will be tested the MADRPC and the other controllers for governing the angle of attack of a flight platform, which corresponds to an unmanned aircraft, to compare their performances. They will be compared in nominal and non-nominal conditions, to check their robustness.

2.2 Scope of the Project

This project will consist of a series of milestones:

1. First, the development of the non-linear kinematics and dynamics for the flight platform to use, as it will be responsible for the correct simulation and where the tests will be performed.

2. From the non-linear model for the flight platform, it will be calculated the linear model, using the nominal flight conditions, to design and tune up the controllers.

3. Tune up the MADRPC, and the traditional control schemes (for this project they will be the PID and the MPC controllers), using the linear model calculated in the previous step.

4. With the controllers designed and tune up, validate the design using the complete non-linear system of the flight platform.

5. With the controllers design validated, they will be tested in a wide range of non-nominal conditions tests.

The completion of the project will come from a performance comparison of the MADRPC with the other traditional control schemes, both in cruise and non-cruise conditions, validating the new proposed controller.

3 Flight Platform

The main goal of this chapter is to develop the six-degrees-of-freedom dynamic model, in order to obtain the UAV's linear and angular velocities, the attitude and the navigation equations, so the adopted controller may be tested and validated. The airframe used will be the H200, an UAV developed by the Universidad Politècnica de Valencia technical team **HORUS UPV**.

The H200 was born out of the necessity of designing a flying test bench for experimenting a hybrid propulsion system, combining LiPo batteries and a hydrogen power system. The main goal is to design, develop and manufacture an UAV capable of exceeding the current flight times limitations for these kind of flight systems, which typically spans 45 minutes [21]. Here is a design of the prototype:

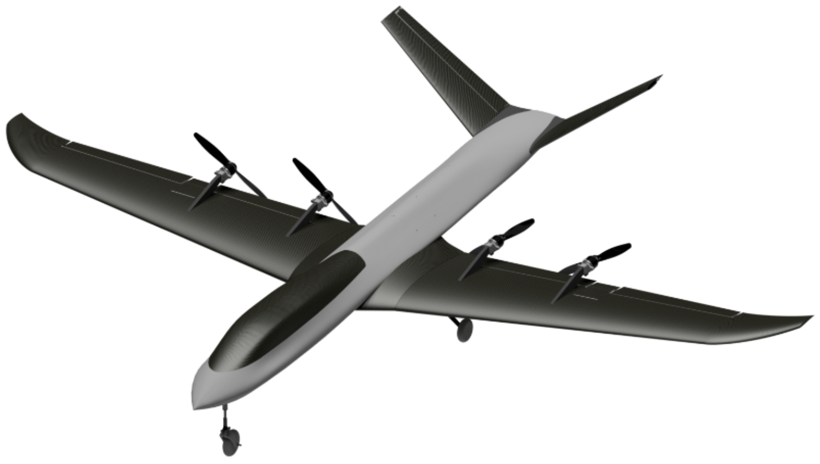


Figure 2: Render of the final design of the H200 prototype [21]

Among the main characteristics of the UAV are: it is designed to use a hydrogen cell as a power source (combined with LiPo batteries), it has a V-tail, making the rudder and the elevator inputs coupled, and count up with four propeller electric engines. All of these features will be analyzed in the following sections, and how they affect the UAV mathematical model and the control.

3.1 Non-Linear Dynamics Modeling

Before the control design, it is needed a non-linear description of the H200, both for simulation and calculating the linear model, so it can be used to design the controllers. For the modeling of the non-linear dynamics of the H200, it will be used the following geometric representation, which includes the reference frames used, the UAV's fundamental points, the velocities, the forces and moments acting on the H200:

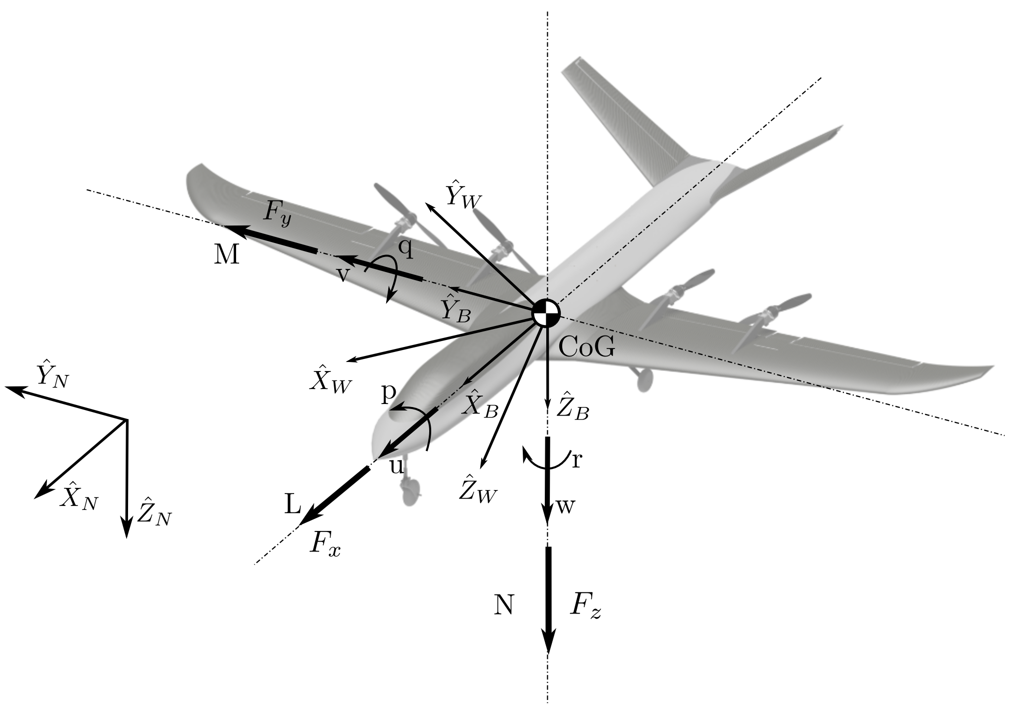


Figure 3: Diagram of reference frames, forces and moments

Here it may be seen the reference frames that are going to be used for the dynamic model:

- Wind Frame (W): $\{\hat{X}_W, \hat{Y}_W, \hat{Z}_W\}$: This frame is where the aerodynamic forces are applied. It will be related to the Body Frame through the attitude matrix $[WB]$, so it may be transformed the forces into the Body Frame for the dynamic model.
- Body Frame (B): $\{\hat{X}_B, \hat{Y}_B, \hat{Z}_B\}$: This frame is fixed to the airframe and will be used for the forces and moments calculation.
- Inertial or Local Navigation Frame (N): $\{\hat{X}_N, \hat{Y}_N, \hat{Z}_N\}$: This frame will be assumed with a flat Earth representation, where X_N points north, Y_N points east and Z_N points downwards, and the local origin is in an arbitrary position. In a latter section it will be discussed the election of this reference point of origin.

The conversions between reference frames are summed up in figure (4).

$$\begin{tikzcd}
\boxed{\text{Wind Frame (W)}} \arrow[r, leftrightarrow, "{[WB]}"'] & \boxed{\text{Body Frame (B)}} \arrow[r, leftrightarrow, "{[BN]}"'] & \boxed{\text{Inertial Frame (N)}}
\end{tikzcd}$$

Figure 4: Block diagram for the conversion between reference frames

The block diagram in figure (4) describes the existence of two fundamental conversion matrices, referred as *Attitude Matrices*. The first one, $[WB]$ describes the change between the wind axes and the body axes, while the second, $[BN]$, provides the conversion between the body frame and the local navigation frame. The first one is defined by the two aerodynamic angles, the angle of attack (α) and the slip angle (β), and the former is described via a wide range of options. For simulation purposes, it will be used a quaternion representation, although for results displaying, it will be used Euler angles description. During this chapter, it will be discussed how to develop these matrices.

The main objective is to obtain a series of differential equations that describes the dynamics and kinematics of the H200. All of these equations are referenced from [19].

3.1.1 Linear Dynamics Equations

The first set of equations to be calculated will be the set governing the UAV linear dynamics, responsible for the body linear velocities, ${}^{B}\vec{V} = [u, v, w]^T$. The first step is to take *Newton's Second Law*, stating that:

$$\sum \vec{F} = m\dot{\vec{V}} \Longrightarrow \sum {}^{B}\vec{F} = \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = m \; {}^{N}\frac{d}{dt}\left(\begin{bmatrix} u \\ v \\ w \end{bmatrix}\right) \tag{1}$$

where F_x, F_y and F_z are the three components of the total force applied in the UAV (gravitational, aerodynamic and thrust forces), and u, v and w are the components of the body velocity, all of them expressed in terms of the Body Frame.

The objective is to calculate the derivative of the velocity with respect to the Local Navigation Frame, so, using the *Transport Theorem*, it may be expressed as a function of the linear and angular velocities, both of them expressed from the Body Frame:

$$\dot{\vec{V}} = {}^{N}\frac{d}{dt}\left(\begin{bmatrix} u \\ v \\ w \end{bmatrix}\right) = {}^{B}\frac{d}{dt}\left(\begin{bmatrix} u \\ v \\ w \end{bmatrix}\right) + {}^{N}\vec{\omega}^{B} \times \vec{V} = \left(\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix}\right) + \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} \dot{u} + q\,w - r\,v \\ \dot{v} + r\,u - p\,w \\ \dot{w} + p\,v - q\,u \end{bmatrix} \tag{2}$$

Once the derivative of the body velocities have been developed with respect to the Local Navigation Frame, it has to be expanded the forces acting up on the UAV:

$$\sum {}^{B}\vec{F} = \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = {}^{B}\vec{F}_G + {}^{B}\vec{F}_A + {}^{B}\vec{F}_T \tag{3}$$

where each of the forces are:

- Gravitational force (${}^{B}\vec{F}_G$).
- Aerodynamic force (${}^{B}\vec{F}_A$).
- Engine thrust force (${}^{B}\vec{F}_T$).

Starting with the gravitational force, it simply may be calculated with the gravity vector, expressed in the Local Navigation Frame, and pre-multiplying it with the Attitude Matrix to express it in the Body Frame:

$${}^{B}\vec{F}_G = [BN] \begin{bmatrix} 0 \\ 0 \\ m\,g \end{bmatrix} \tag{4}$$

where m is the aircraft mass and g is the gravity acceleration. The Attitude Matrix will be discussed later. The following force to be calculated will be the thrust force, expressed in the Body Frame as:

$$^{B}\vec{F_T} = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} = \begin{bmatrix} T(\delta_P) \\ 0 \\ 0 \end{bmatrix} \tag{5}$$

As a matter of simplicity, the thrust force will only be considered in the $\hat{X}_B$ axis. Now, it is needed the thrust model, relating the throttle level with the actual thrust of the propeller. This engine model will be discussed in a later chapter.

The last force to develop is the aerodynamic force vector. As discussed in [19], these forces generally are calculated and expressed through the Wind Frame. It consist of the *Lift Force*, the *Drag Force* and the *Lateral Force*. They will be expressed in terms of dimensionless coefficients, that depend on the flight state variables and the inputs:

$$^{W}\vec{F_A} = \frac{1}{2}\,\rho\, V^2\, S_w \begin{bmatrix} -C_D \\ C_Y \\ -C_L \end{bmatrix} \tag{6}$$

where ρ is the air density, V is the airspeed, S_w is the wing reference surface, and C_D, C_Y and C_L are the drag, lateral force and lift coefficients, respectively. Their value will be discussed later on.

Then, with the expression of the aerodynamic force measured in the Wind Frame, it is needed to build the rotation matrix relating the Wind Frame and the Body Frame, $[WB]$, through a series of rotations with α and β:

1. First, starting from the Body Frame axes, rotate along the $\hat{Y}_B$ axis the angle of attack, α.
2. Then, from the new reference frame created, rotate along the new Z axis the sideslip angle, β

Mathematically, the rotation matrix to convert from Body Frame to Wind Frame is [53]:

$$\begin{aligned}
[WB] = R_z(\beta)\,R_y(\alpha) &= \begin{bmatrix} cos(\beta) & sin(\beta) & 0 \\ -sin(\beta) & cos(\beta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos(\alpha) & 0 & sin(\alpha) \\ -sin(\beta) & cos(\beta) & 0 \\ 0 & 0 & 1 \end{bmatrix} = \\
&= \begin{bmatrix} cos(\alpha)\,cos(\beta) & sin(\beta) & sin(\alpha)\,cos(\beta) \\ -cos(\alpha)\,sin(\beta) & cos(\beta) & -sin(\alpha)\,sin(\beta) \\ -sin(\alpha) & 0 & cos(\alpha) \end{bmatrix}
\end{aligned} \tag{7}$$

With this matrix calculated, the aerodynamic force expressed in the Body Frame will be:

$$^{B}\vec{F_A} = [BW]\,^{W}\vec{F_A} = [WB]^T\,^{W}\vec{F_A} \tag{8}$$

As a remark, this rotation matrix constitutes a DCM, and as such, its inverse is equal to its transpose. At this point, all the forces acting on the UAV's body have been calculated and expressed in the Body Frame:

$$\sum {}^{B}\vec{F} = {}^{B}\vec{F}_G + {}^{B}\vec{F}_T + {}^{B}\vec{F}_A = [BN]\begin{bmatrix}0\\0\\m\,g\end{bmatrix} + \begin{bmatrix}T(\delta_P)\\0\\0\end{bmatrix} + [WB]^T\left(\frac{1}{2}\rho\,V^2\,S_w\begin{bmatrix}-C_D\\C_Y\\-C_L\end{bmatrix}\right) \tag{9}$$

And putting together equations (2) and (9), the set of equations governing the linear dynamics are:

$$m\begin{bmatrix}\dot{u}+q\,w-r\,v\\\dot{v}+r\,u-p\,w\\\dot{w}+p\,v-q\,u\end{bmatrix} = [BN]\begin{bmatrix}0\\0\\m\,g\end{bmatrix} + \begin{bmatrix}T(\delta_P)\\0\\0\end{bmatrix} + [WB]^T\left(\frac{1}{2}\rho\,V^2\,S_w\begin{bmatrix}-C_D\\C_Y\\-C_L\end{bmatrix}\right) \tag{10}$$

3.1.2 Rotational Equations

The next step is deriving the H200's rotational equations, responsible of the body angular velocities, ${}^{B}\vec{\omega} = [p,q,r]^T$. Once again, as with the linear dynamics section, it is used *Newton's Second Law of Rotation*which states that:

$$\sum \vec{M} = {}^{CoG}\dot{\vec{M}} \Longrightarrow \sum {}^{B}\vec{M} = \begin{bmatrix}l\\m\\n\end{bmatrix} = {}^{N}\frac{d}{dt}\left({}^{CoG}\dot{\vec{H}}\right) \tag{11}$$

where l, m and n are the three components of the total angular momentum of the body, and ${}^{CoG}\vec{H}$ is the angular momentum measured from the center of gravity (CoG). By definition, the H200 is a rigid body, so the angular momentum will be expressed as:

$${}^{CoG}\vec{H} = {}^{CoG}[I]\;{}^{N}\vec{\omega}^{B} \tag{12}$$

where ${}^{CoG}[I]$ is the inertia matrix of the aircraft. This matrix is defined as:

$${}^{CoG}[I] = \begin{bmatrix}I_{xx} & -I_{xy} & -I_{xz}\\-I_{yx} & I_{yy} & -I_{yz}\\-I_{zx} & -I_{zy} & I_{zz}\end{bmatrix} \tag{13}$$

Now, it is going to be used the *Transport Theorem* once again to calculate the time derivative of the angular momentum measured on the Local Navigation Frame:

$${}^{CoG}\dot{\vec{H}} = {}^{B}\frac{d}{dt}\left({}^{CoG}\vec{H}\right) + {}^{N}\vec{\omega}^{B} \times {}^{CoG}\vec{H} \tag{14}$$

$$= \begin{bmatrix}I_{xx}\,\dot{p} - I_{xy}\,\dot{q} - I_{xz}\,\dot{r}\\-I_{yx}\,\dot{p} + I_{yy}\,\dot{q} - I_{yz}\,\dot{r}\\-I_{zx}\,\dot{p} - I_{zy}\,\dot{q} + I_{zz}\,\dot{r}\end{bmatrix} + \begin{bmatrix}p\\q\\r\end{bmatrix} \times \begin{bmatrix}I_{xx}\,p - I_{xy}\,q - I_{xz}\,r\\-I_{yx}\,p + I_{yy}\,q - I_{yz}\,r\\-I_{zx}\,p - I_{zy}\,q + I_{zz}\,r\end{bmatrix} \tag{15}$$

Now, with the time derivative of the body angular rates calculated, the moments acting on the H200 will consist of the engine thrust moment and the aerodynamic force, as the gravitational force does not generate a moment around the *CoG*:

$$\sum \vec{M} = \begin{bmatrix}l\\m\\n\end{bmatrix} = {}^{B}\vec{M}_T + {}^{B}\vec{M}_A \tag{16}$$

where each of the moments are:

- Thrust force moment (${}^{B}\vec{M}_T$).
- Aerodynamic moment (${}^{B}\vec{M}_A$).

The first one, the thrust force moment will be calculated as:

$$ {}^{B}\vec{M}_T = \vec{r}_{prop} \times {}^{B}\vec{F}_T = \begin{bmatrix} 0 \\ 0 \\ L_{prop} \end{bmatrix} \times \begin{bmatrix} T(\delta_P) \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ L_{prop}\, T(\delta_P) \\ 0 \end{bmatrix} \tag{17} $$

where L_{prop} is the vertical distance from the propeller to the CoG.

On the other hand, the moments caused from the aerodynamic forces will be expressed, as a matter of simplicity, in terms of a set of dimensionless coefficients, depending on the H200 geometry, the flight state variables and the inputs:

$$ {}^{W}\vec{M}_A = \frac{1}{2}\,\rho\, V^2\, S_w \begin{bmatrix} b_w\, C_l \\ c_w\, C_m \\ b_w\, C_n \end{bmatrix} \tag{18} $$

where ρ is the air density, V is the airspeed, S_w is the wing reference surface, b_w is the wingspan, c_w is the wing chord, and C_l, C_m and C_n are the coefficients for each of the moments. Again, these coefficients will be discussed in a later section.

In addition, as it happened with the aerodynamic forces, the moments are also expressed generally in terms of the Wind Frame, so it's needed to use again the matrix $[WB]$ to change from the Wind Frame to the Body Frame, being the aerodynamic moment expressed in the latter as:

$$ {}^{B}\vec{M}_A = [BW]\, {}^{W}\vec{M}_A = [WB]^T\, {}^{W}\vec{M}_A \tag{19} $$

Therefore, total sum of moments acting on the UAV will be:

$$ \sum \vec{M} = \begin{bmatrix} l \\ m \\ n \end{bmatrix} = {}^{B}\vec{M}_T + {}^{B}\vec{M}_A = \begin{bmatrix} 0 \\ L_{prop}\, T(\delta_P) \\ 0 \end{bmatrix} + [WB]^T \left(\frac{1}{2}\,\rho\, V^2\, S_w \begin{bmatrix} b_w\, C_l \\ c_w\, C_m \\ b_w\, C_n \end{bmatrix} \right) \tag{20} $$

Finally, joining equations (14) and (20), the differential equations governing the body rotation rate are:

$$ \begin{bmatrix} I_{xx}\,\dot{p} - I_{xy}\,\dot{q} - I_{xz}\,\dot{r} - q\,(I_{zx}\,p + I_{zy}\,q - I_{zz}\,r) + r\,(I_{yx}\,p - I_{yy}\,q + I_{yz}\,r) \\ I_{yy}\,\dot{q} - I_{yx}\,\dot{p} - I_{yz}\,\dot{r} + p\,(I_{zx}\,p + I_{zy}\,q - I_{zz}\,r) - r\,(I_{xy}\,q - I_{xx}\,p + I_{xz}\,r) \\ I_{zz}\,\dot{r} - I_{zy}\,\dot{q} - I_{zx}\,\dot{p} - p\,(I_{yz}\,p - I_{yy}\,q + I_{yz}\,r) + q\,(I_{xy}\,q - I_{xx}\,p + I_{xz}\,r) \end{bmatrix} $$
$$ = \begin{bmatrix} 0 \\ L_{prop}\, T(\delta_P) \\ 0 \end{bmatrix} + [WB]^T \left(\frac{1}{2}\,\rho\, V^2\, S_w \begin{bmatrix} b_w\, C_l \\ c_w\, C_m \\ b_w\, C_n \end{bmatrix} \right) \tag{21} $$

3.1.3 Attitude Coordinates Equations

Once the H200 dynamic equations have been solved, it has to be developed the kinematic equations, this means, calculating the time derivative of the Attitude Matrix, responsible of the UAV's orientation with respect of the Local Navigation Frame. Here, from all the possibilities there exist, it will be used Euler angles and quaternion representation, attending to output clarity and numerical reasons.

The first one is based on a rotation of each of the inertial axis by some angle in a specific order of rotation to achieve the final attitude orientation. It will be used the $[3-2-1]$ Euler angles, which the orientation will be defined as [53]:

1. Starting aligned with the Local Navigation Frame, the Z axis is rotated by the yaw angle (ψ).
2. Then, from the new frame, the Y axis is rotated by the pitch angle (θ).
3. And lastly, from the result frame, the X axis is rotated by the roll angle (ϕ).

This is the standard for of attitude coordinates for fixed wing aircraft, called *Roll-Pitch-Yaw* orientation. Taking this steps, and defining each of the rotation matrices, the Attitude Matrix using Euler angles is:

$$[BN] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\phi) & sin(\phi) \\ 0 & -sin(\phi) & cos(\phi) \end{bmatrix} \begin{bmatrix} cos(\theta) & 0 & -sin(\theta) \\ 0 & 1 & 0 \\ sin(\theta) & 0 & cos(\theta) \end{bmatrix} \begin{bmatrix} cos(\psi) & sin(\psi) & 0 \\ -sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} = \tag{22}$$

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

With the Attitude Matrix calculated in terms of Roll-Pitch-Yaw, the kinematic equation is defined as:

$$\begin{bmatrix} \dot{\psi} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \frac{1}{cos(\theta)} \begin{bmatrix} 0 & sin(\phi) & cos(\phi) \\ 0 & cos(\phi)\,cos(\theta) & -sin(\phi)\,cos(\theta) \\ cos(\theta) & sin(\phi)\,sin(\theta) & cos(\phi)\,sin(\theta) \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{23}$$

The main characteristics of this attitude representation are:

Advantages	**Disadvantages**
Suitable for angle output and control	Not numerically robust
Most commonly used orientation representation	Singularity at pitch angle = ± 90 deg
	Non-linear kinematic equation

Table 2: Characteristics for the Euler Angle orientation representation

In order to face the limitations for the Euler Angles, it is introduced the quaternion representation. Now, the Attitude Matrix have the following form:

$$[BN] = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2\,(q_1\,q_2 + q_0\,q_3) & 2\,(q_1\,q_3 - q_0\,q_2) \\ 2\,(q_1\,q_2 - q_0\,q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2\,(q_2\,q_3 + q_0\,q_1) \\ 2\,(q_1\,q_3 + q_0\,q_2) & 2\,(q_2\,q_3 - q_0\,q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \tag{24}$$

with its kinematic equation as:

$$\begin{bmatrix} \dot{q}_0 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix} \begin{bmatrix} 0 \\ p \\ q \\ r \end{bmatrix} \tag{25}$$

The main characteristics for the quaternion representation is:

Advantages	Disadvantages
Numerically stable	Not suitable for angle output and control
No singularities	
Linear kinematic equation	

Table 3: Characteristics for the quaternion orientation representation

With this, the main reason why two orientation representations have been chosen, in particular these, is because they complement each other nicely when it comes to flight simulation and control. Quaternion representation is generally used for simulating the UAV kinematics, as it is a linear equation with no singularities, and the Euler angles representation comes handy as it is intuitive, so it's generally used for attitude output and reference signal tracking.

3.1.4 Navigation Equations

The final set of equations are the flight navigation ones, with respect to the Local Navigation Frame. They can be calculated using:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = [BN]^T \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{26}$$

They can be easily calculated rotating the linear velocities expressed in the Body Frame to the Local Navigation Frame, so it has to be used the Attitude Matrix, in whichever form it is desired.

3.2 H200 Model Parameters

With the non-linear dynamic equations done, the next step is to particularize them with the aircraft parameters. All of them take as reference [21] and [56].

3.2.1 Geometric Parameters

Geometric parameters			
Parameter	**Symbol**	**Value**	**Units**
Wingspan	b_w	2.95	m
Mean aerodynamic chord	c_w	0.3731	m
Wing surface	S_w	1.0162	m^2
Reference mass	m	15	kg
Elevator arm	L_{elev}	1.1	m
Propellers arm	L_{prop}	0	m
X-Axis inertia	I_{xx}	1.609	$\mathrm{kg/m}^3$
Y-Axis inertia	I_{yy}	2.773	$\mathrm{kg/m}^3$
Z-Axis inertia	I_{zz}	4.310	$\mathrm{kg/m}^3$
X-Y axis cross inertia	I_{xy}	$-4.804 \cdot 10^{-3}$	$\mathrm{kg/m}^3$
X-Z axis cross inertia	I_{xz}	$-8.232 \cdot 10^{-2}$	$\mathrm{kg/m}^3$
Y-Z axis cross inertia	I_{yz}	$-9.468 \cdot 10^{-4}$	$\mathrm{kg/m}^3$

Table 4: H200 geometric parameters [21]

3.2.2 Propulsion Model

Following with the propulsion model, taking into account that the aircraft is fueled by a hydrogen cell, direct methodologies cannot be applied. For this purpose, a Blade Element Theory (BET) code has been developed to study the power needs for certain flight conditions [21]. For the thrust and power model, they will be defined using two dimensionless coefficients (C_T and C_P), similarly as the aerodynamic model. So, the thrust and power will follow the next model:

$$T = C_T\,(\rho\, n^2\, D^4) \tag{27}$$
$$P = C_P\,(\rho\, n^3\, D^5) \tag{28}$$

where T and P_T are the engine thrust force and output power, respectively, n is the blade angular velocity (expressed in Hz) and D is the blade diameter. The objective is to develop an expression relating the throttle level (δ_P) with each of the variables (thrust and power).The first step in this process is to set the powerplant hardware. The H200 uses the *APC Propeller 13x6.5* [47] [56] blade and the *AT3520 Long Shaft 550KV* [55] [56]:

Figure 5: H200 powerplant hardware

So, the blade diameter will be:

$$D = 13\,\text{inches} = 0.3302\,\text{m} \tag{29}$$

The next step is to develop the actual thrust and power model for one motor. For this, it's going to be used experimental data done in the laboratory. First of all, it's going to be calculated the linear model relating blade angular velocity with the throttle level:

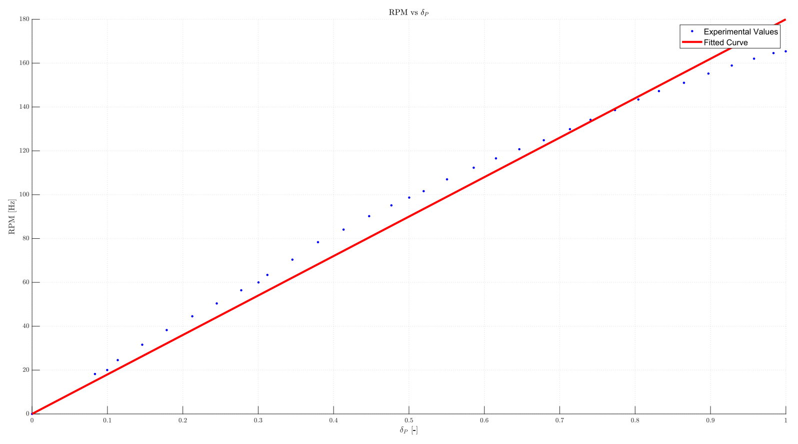


Figure 6: Linear model relating RPM with throttle level

By fitting a linear model to the experimental data, the blade angular velocity can now be expressed as:

$$n(\delta_P) = p_{n_1}\,\delta_P \qquad \text{[Units: Hz]} \tag{30}$$

The second step is to develop a polynomial model for the dimensionless coefficients of thrust and power, C_T and C_P respectively. For these, the experimental data will not be given directly against the throttle level. Instead, it will be used the non-dimensional induced velocity (J), expressed in the following form:

$$J(\delta_P) = \frac{V}{D\,n(\delta_P)} = \frac{V}{D\,(p_{n_1}\,\delta_P)} \tag{31}$$

As a result, the experimental data relating C_T and C_P with J is:

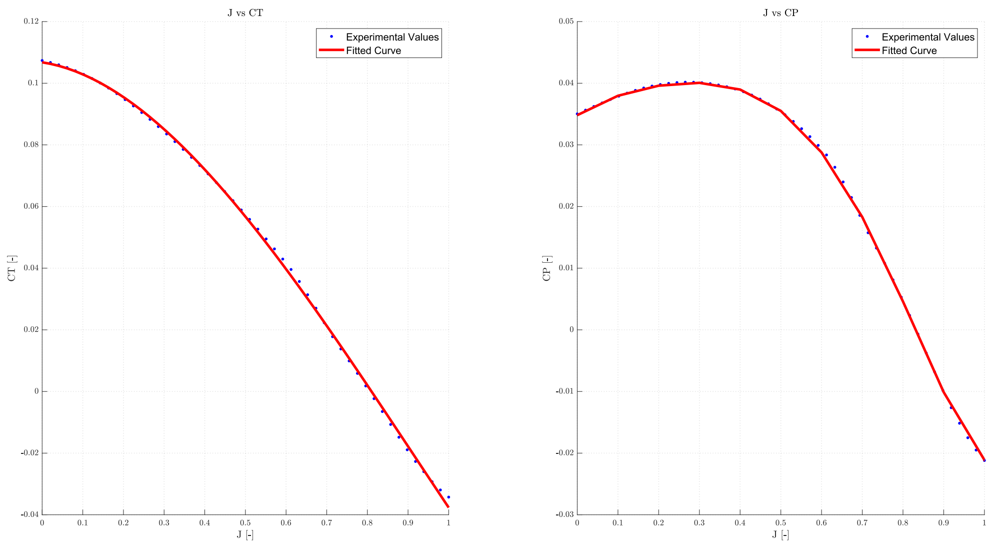


Figure 7: Linear model relating C_T and C_P with δ_P

Then, the dimensionless coefficients C_T and C_P will be expressed using a second-degree polynomial model, as:

$$C_T(J) = p_{C_{T_3}}\, J^3 + p_{C_{T_2}}\, J^2 + p_{C_{T_1}}\, J + p_{C_{T_0}} \tag{32}$$

$$C_P(J) = p_{C_{P_6}}\, J^6 + p_{C_{P_5}}\, J^5 + p_{C_{P_4}}\, J^4 + p_{C_{P_3}}\, J^3 + p_{C_{P_2}}\, J^2 + p_{C_{P_1}}\, J + p_{C_{P_0}} \tag{33}$$

At this point, combining (30), (31), (32) and (33) it is developed the complete model for the H200 propulsion system:

$$T(\delta_P) = C_T(\delta_P)\, \left(\rho\, D^4\, n^2(\delta_P)\right) \tag{34}$$

$$P(\delta_P) = C_P(\delta_P)\, \left(\rho\, D^5\, n^3(\delta_P)\right) \tag{35}$$

$$C_T(\delta_P) = p_{C_{T_3}}\, J^3(\delta_P) + p_{C_{T_2}}\, J^2(\delta_P) + p_{C_{T_1}}\, J(\delta_P) + p_{C_{T_0}} \tag{36}$$

$$\begin{aligned} C_P(\delta_P) &= p_{C_{P_6}}\, J^6(\delta_P) + p_{C_{P_5}}\, J^5(\delta_P) + p_{C_{P_4}}\, J^4(\delta_P) \\ &\quad + p_{C_{P_3}}\, J^3(\delta_P) + p_{C_{P_2}}\, J^2(\delta_P) + p_{C_{P_1}}\, J(\delta_P) + p_{C_{P_0}} \end{aligned} \tag{37}$$

$$J(\delta_P) = \frac{V}{D\, n(\delta_P)} \tag{38}$$

$$n(\delta_P) = p_{n_1}\, \delta_P \tag{39}$$

Finally, the H200 propulsive parameters are:

Propulsive model parameters							
D	0.3302	p_{n_1}	179.9970	$p_{C_{T_3}}$	0.07115	$p_{C_{P_6}}$	0.1446
				$p_{C_{T_2}}$	-0.1954	$p_{C_{P_5}}$	0.0126
				$p_{C_{T_1}}$	-0.02019	$p_{C_{P_4}}$	-0.4078
				$p_{C_{T_0}}$	0.1068	$p_{C_{P_3}}$	0.2859
						$p_{C_{P_2}}$	-0.1337
						$p_{C_{P_1}}$	0.0424
						$p_{C_{P_0}}$	0.03482

Table 5: H200 propulsive model parameters

As a remark, the only two variables left to be set, V and ρ correspond to the flight velocity and air density. They will be set depending on the flight instant, and will be set also for the nominal flight conditions in a future section.

3.2.3 Aerodynamic Coefficients

Finally, for the aerodynamic model of the H200, as commented previously, the aerodynamic force and moments will be expressed using dimensionless coefficients in the form of:

$$ {}^{W}\vec{F}_A = \frac{1}{2}\,\rho\,V^2\,S_w \begin{bmatrix} -C_D \\ C_Y \\ -C_L \end{bmatrix} \qquad {}^{W}\vec{M}_A = \frac{1}{2}\,\rho\,V^2\,S_w \begin{bmatrix} b_w\,C_l \\ c_w\,C_m \\ b_w\,C_n \end{bmatrix} \tag{40}$$

These coefficients will be expressed in terms of flight variables following a linear model, taking as reference the flight mechanics general aerodynamic model:

- Forces coefficients:

$$C_D = C_{D_0} + A_1\,C_L + A_{Polar}\,C_L^2 \tag{41}$$

$$C_Y = C_{Y_\beta}\,\beta + \frac{b_w}{2\,V}\left(C_{Y_p}\,p + C_{Y_r}\,r\right) + C_{Y_{\delta_A}}\,\delta_A + C_{Y_{\delta_R}}\,\delta_R \tag{42}$$

$$C_L = C_{L_0} + C_{L_\alpha}\,\alpha + \frac{c_w}{2\,V}\left(C_{L_{\dot\alpha}}\,\dot\alpha + C_{L_q}\,q\right) + C_{L_{\delta_E}}\,\delta_E + C_{L_{\delta_F}}\,\delta_F \tag{43}$$

- Moments coefficients:

$$C_l = C_{l_\beta}\,\beta + \frac{b_w}{2\,V}\left(C_{l_p}\,p + C_{l_r}\,r\right) + C_{l_{\delta_A}}\,\delta_A + C_{l_{\delta_R}}\,\delta_R \tag{44}$$

$$C_m = C_{m_0} + C_{m_\alpha}\,\alpha + \frac{c_w}{2\,V}\left(C_{m_{\dot\alpha}}\,\dot\alpha + C_{m_q}\,q\right) + \frac{L_{elev}}{c_w}\,C_{m_{\delta_E}}\,\delta_E + C_{m_{\delta_F}}\,\delta_F \tag{45}$$

$$C_n = C_{n_\beta}\,\beta + \frac{b_w}{2\,V}\left(C_{n_p}\,p + C_{n_r}\,r\right) + C_{n_{\delta_A}}\,\delta_A + C_{n_{\delta_R}}\,\delta_R \tag{46}$$

where α is the angle of attack, β is the sideslip angle, $\dot\alpha$ and $\dot\beta$ are the rate of change for the angle of attack and the sideslip angle respectively. Taking as reference [21] and [56], it has been experimented using various semi-empirical and numerical techniques (Panel method, CFD, etc) to calculate each of the coefficients:

Aerodynamic forces coefficients					
C_D		C_Y		C_L	
C_{D_0}	0.039	C_{Y_β}	-0.206777	C_{L_0}	0.308
A_1	0.007	C_{Y_p}	0.016541	C_{L_α}	5.140879
A_{Polar}	0.057	C_{Y_r}	0.126227	$C_{L_{\dot{\alpha}}}$	0.667
		$C_{Y_{\delta_A}}$	-0.000410	C_{L_q}	7.326102
		$C_{Y_{\delta_R}}$	-0.003172	$C_{L_{\delta_E}}$	0.007585
				$C_{L_{\delta_F}}$	0.013771

Aerodynamic moments coefficients					
C_l		C_m		C_n	
C_{l_β}	-0.105871	C_{m_0}	0.00835	C_{n_β}	0.031839
C_{l_p}	-0.476318	C_{m_α}	-0.507412	C_{n_p}	-0.046901
C_{l_r}	0.057	$C_{m_{\dot{\alpha}}}$	-2.1287	C_{n_r}	-0.06186
$C_{l_{\delta_A}}$	-0.003786	C_{m_q}	-7.275333	$C_{n_{\delta_A}}$	-0.000072
$C_{l_{\delta_R}}$	-0.000452	$C_{m_{\delta_E}}$	-0.0185	$C_{n_{\delta_R}}$	0.001064
		$C_{m_{\delta_F}}$	-0.001192		

Table 6: Linear aerodynamic model coefficients

4 State of the Art and Exploration of Alternatives

This chapter will serve as a review of all the state of the art for the different tools used in this work. This project present two main lines of work, where there exists a wide range of possibilities to tackle them:

1. First, the simulation of the H200 dynamic system: In order to validate and perform tests with the proposed controller and to compare it with the traditional control schemes.
2. Then, the control strategies itself: The control industry has exploded in the last century with the development of a wide range of control techniques to govern processes.

4.1 Simulation of the Dynamic Systems

The first line of work focuses on the problem for simulating the flight platform used to achieve the objectives described in this project: design and tuning up of the proposed new control algorithm, along with the tuning of the other two traditional control strategies, and test them against each other and compare their performance.

With this objective in mind, it is needed a piece of software where the non-linear dynamic equations of the chosen project's flight platform, the H200, which have been developed in section 3, are included and available to use for simulating the flight mission. This way, this software will be used as flight platform for calculating the linear model to tune up the controllers and then validating, simulating a flight maneuver with the complete real model.

It has to be remarked, that a validation of the model must be performed in order to check that the mathematical model actually reflects accurately the H200 behavior. Having stating this remark, there exists a wide range of tools, all with their advantages and disadvantages.

4.1.1 OpenModelica

OpenModelica is an environment used in industrial and academic applications, focused on modeling and simulation, which is based in the Modelica language [15]. It is an open-source tool, used for model-based simulations. This is, instead of simulating the signal flow during the simulation, like Simulink, OpenModelica simulates the real model, with a series of internal equations per model, and it is connected like the real model.

OpenModelica

Figure 8: OpenModelica offical logo [15]

This tool has been used in a wide range of projects, both academics and industrial-wise: for embedded applications [16] [30], network and distributed systems [52] [37], etc. For industrial applications, it is being used for analysis and optimization of hydraulic valves, verifying the correct performance of components, designing a steam generator, and much more [15].

In addition, it counts up with a graphical interface model designer, an interface for notebooks, (like Mathematica), a shell for interactive and debugging sessions. The way it works is translating the Modelica script code to C language for simulating it [15]. It is capable of modeling any physical system, either from official libraries or from the ones made by the community.

4.1.2 Python

The next tool to be reviewed is going to be Python. It is far more extended compared with OpenModelica in all types of usages and applications: computer vision, data science and much more. It is an open-source object-object oriented programming language which its main advantage is simplicity and flexibility [48].

Figure 9: Python official logo

The main features of the Python programming language are [48]:

- Easy to read and comprehend syntax: It is a programming language easy to develop, making it an outstanding candidate for prototyping.
- The large standard library, covering a wide range of common programming functions: networking, working with files, plotting, etc.
- It is easily extended, either via developing Python modules or via a compiled language (C or C++).
- It covers the basic data types: number, strings, list and dictionaries.
- The source code may be organized into modules and packages.

Python has been used in all areas of society, given its large community and flexibility and adaptability to every problem. From business (building a commercial cloud backup, developing timesheets, etc), data science, education, engineering (to be used with collaborative robots, etc) and so much more [48].

4.1.3 MATLAB

Last but not least, the final tool reviewed for implementing and simulating the flight platform model is MATLAB. MATLAB is one of the main programming languages used by engineers and researches to aid them in their tasks [42].

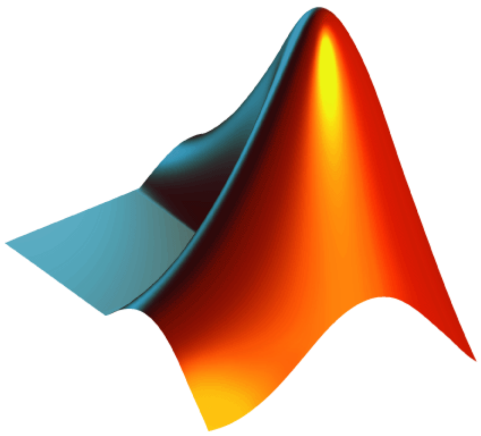


Figure 10: MATLAB official logo [42]

MATLAB stands for “MATrix LABoratory", specializing in numerical computing and matrix operations. Moreover, as like Python, the source code may be grouped and develop toolboxes. MATLAB count with a wide range of toolboxes for facing almost every problem in engineering: aerospace, signal treatment, networking, embedded systems, control engineering.

In fact, apart from the source script files that may be executed, it counts with a signal editor, called Simulink, for block diagram model simulation, for a signal flow simulation; or physical model simulation, through Simscape, for it to work like Open-Modelica.

In addition, as like Python, it counts with a wide range of official libraries or toolboxes, ready to use for the majority of engineering problems, and the community is quite hands-on working inside this programming language, developing community based toolboxes to tackle the more specific problems that may exist.

4.1.4 Chosen Tool for Flight Platform Simulation

Having reviewed some of the simulation tools that exist for the purpose of simulating the flight platform during the tests, it will be chosen **MATLAB** and **Simulink**. The reason behind this decision is that the combination of MATLAB script files with Simulink models offer the most complete, intuitive and easy to implement environment to build, design and simulate the flight platform and to include the controllers for this project.

In addition, along with MATLAB and Simulink, it will be used some of its toolboxes, as they offer the necessary functions to implement the flight platform and the controllers, without spending too much time developing the source code.

The advantages and disadvantages of MATLAB and Simulink are:

Advantages	**Disadvantages**
Wide range of toolboxes and libraries with the necessary functions to implement control schemes and aerospace systems	Not open source. Must pay for a license to use.
Intuitive and easy to implement environment for the flight platform	Might be slow to perform all the simulations, as the language is very high-level
One of the best programming languages to work with arrays and numerical operations	It will be a signal flow simulation, not simulating the actual physical elements

Table 7: Advantages and disadvantages of using MATLAB and Simulink for the project

4.2 Control Strategies

For the second line of work, it will be tackled the problem of which control scheme to be used in order to govern the angle of attack system from the flight platform. In the last century, the automatic control industry has exploded with a wide range of options for controlling systems. From the simple and robust PID, to the more complex predictive controllers, or the multivariate state-space formulation controllers, all with their own versions and their own sub-structures. Depending on the complexity of the system and the computer power available, there is an option for every problem.

4.2.1 PID Control

The first control algorithm to be reviewed will be the *Proportional-Integral-Derivative* control, referred as PID control. This control scheme is one of the most easy to implement and effective control strategies. Although the automatic control theory and industry has done nothing but grow during the last decades, 90% of industry's control problems are solved using the PID control [11].

One of its main advantages is its simplicity in design, tuning and implementation. In addition to this advantages, it has to be added its effectiveness and a satisfactory performance with digital systems [31] [31]. The PID control is a tool for feedback control, where the input to the controller is the tracking error signal, which is the difference between the reference and the response signals [39]. Specifically, this scheme is called *three-term* algorithm, as the controller output is a sum of three different actions [31]:

- A proportional action (P)
- An integral action (I)
- A derivative action (D)

Each of the different actions represent each of the operations done to the tracking error in order to calculate the controller output at a certain time instant t:

- The proportional action (P) takes the error value at instant t and multiplies it by a certain constant (K_P). It represents the *present* error [39].
- The integral action (I) takes the error integral up to the instant t and is multiplied by the K_I constant. This action represents the accumulated *past* error [39].

- Finally, the derivative action (D) takes the error derivative at the instant t, then to be multiplied by its respective constant K_D. It represents the forecast for the *future* error [39].

The physical meaning of these elements inside the PID controls is to produce an control output (input for the process) combining the past, present and future error signals in every step [39]. The basic structure of the PID control may be seen in figure (11).

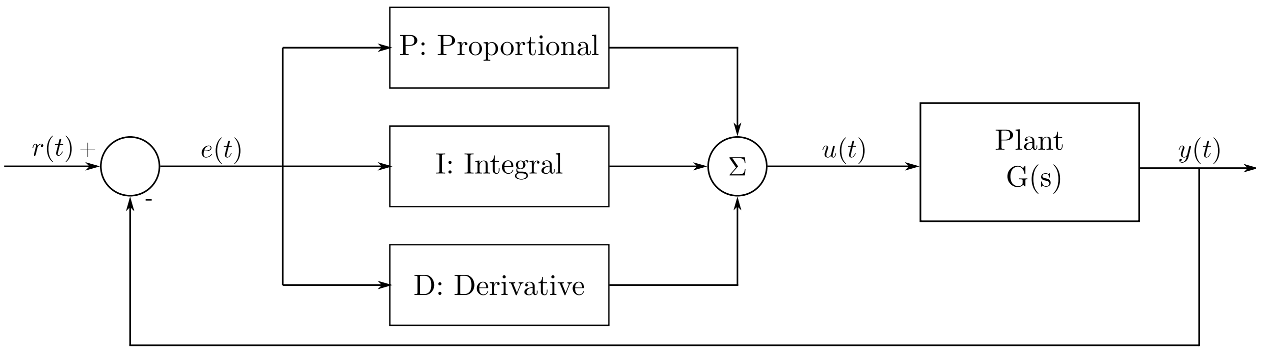


Figure 11: Structure of the PID control system

where it may be appreciated the tracking error signal ($e(t)$), which is the difference between the reference signal ($r(t)$) and the output signal ($y(t)$), is processed through the three PID actions and combined to calculate the controller output ($u(t)$) and input to the plant ($G(s)$), and then all this process starts again until the response is governed.

The PID controller calculates the output control action for the plant following the next equation:

$$u(t) = K_P\, e(t) + K_I \int_0^t e(\tau) d\tau + K_D\, \frac{de(t)}{dt} \quad \text{where} \quad e(t) = r(t) - y(t) \tag{47}$$

From equation (47), it may be seen how K_P, K_I and K_D controls the proportional, derivative and integral actions respectively. For the PID controller to give a successful response, the internal parameters must be tune up accordingly. Following figure (11), if the negative feedback formula is performed on the system, the closed loop transfer function will take the following structure:

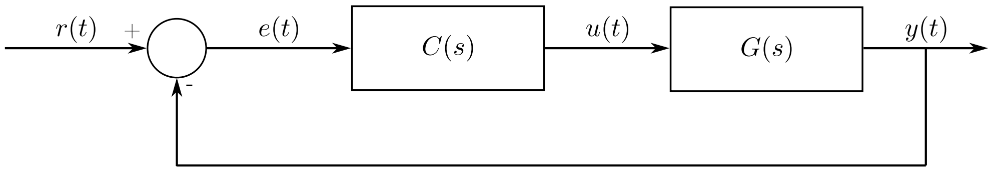


Figure 12: Negative feedback control structure

which will follow the next equation:

$$M(s) = \frac{Y(s)}{R(s)} = \frac{G(s)\, C(s)}{1 + C(s)\, G(s)} \tag{48}$$

where $C(s)$ is the controller block.

In order for the closed loop system for be stable, its poles must have negative real parts (if the system is continuous) or inside the unit circle in the imaginary plane (if the system is discrete). The poles of the system come from the denominator of the closed loop system, in equation (48). From this equation, it may be seen the denominator is directly dependent on the controller block, so the selection of K_P, K_I and K_D is of vital importance for the stability of the system, and thus, the process for tuning them.

The process for tuning starts with the modeling of the process as accurate as possible, then it has to be performed a linearization process, from a chosen equilibrium point. From here, the tuning process starts. Nowadays, there exists a wide range for tuning processes in order to design a correct set of control parameters: the manual trial and error method, the Root-Locus method [32] [46], the Ziegler-Nichols method [43] [58], optimization-based methods [44] [25] or model-based methods [23] [57].

Focused on the aerospace industry, PID controllers have a vast history in solving industry's main problems, one of them (and the main focus of this project), controlling the aircraft angle of attack. Historically, the PID controller has been one of the main control schemes for governing the aircraft's pitch angle, due to its simplicity, robustness and its good performance with the raise of digital systems and unmanned aircraft systems [45] [28] [54] [29].

4.2.2 State-Space Control

The next control framework to be reviewed is the state-space control. Unlike the PID controller, which is focused to single-input single-output systems, the state-space controller is capable of governing the dynamics of a multivariate system, as it is not described with a uniquely differential equation, but with a series of them.

With this characteristic in mind, the state-space system is not only bounded to one input and one output, but can track n inputs and m outputs. In addition, it gives internal insight of the state of the system in any given instant, which will be used by the design of controllers.

The system (in linear form) is generally described using the following matrix form:

$$\begin{aligned}\dot{\vec{x}}(t) &= \boldsymbol{A}\,\vec{x}(t) + \boldsymbol{B}\,\vec{u}(t)\\ \vec{y}(t) &= \boldsymbol{C}\,\vec{x}(t) + \boldsymbol{D}\,\vec{u}(t)\end{aligned} \quad \text{where} \quad \vec{x}(t_0) = \vec{x}_0 \tag{49}$$

where $\vec{x}(t)$ is called the state vector, holding the internal information about the system in any given instant:

$$\vec{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots x_n(t) \end{bmatrix} \tag{50}$$

and $\vec{u}(t)$ and $\vec{y}(t)$ hold the input and output vectors from the system. The state-space equations is not a control scheme in its own, but a framework to describe dynamic systems, as there exists a conversion between state-space systems and transfer functions, and vice versa. The most general control scheme for state-space systems is the state feedback controller, which follows the next structure:

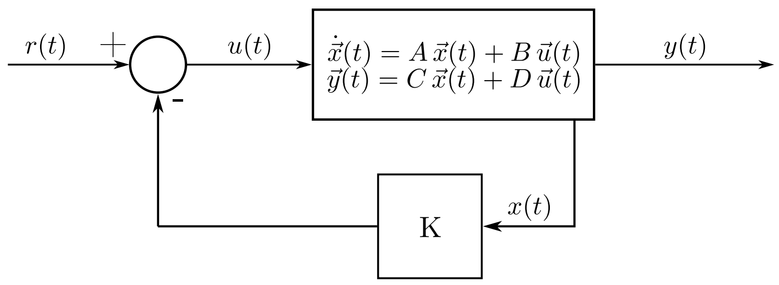

$r(t)$
+
-
$u(t)$
$\dot{\vec{x}}(t) = A\,\vec{x}(t) + B\,\vec{u}(t)$
$\vec{y}(t) = C\,\vec{x}(t) + D\,\vec{u}(t)$
$y(t)$
$x(t)$
K

Figure 13: State-space control structure

and the equation for calculating the control action is:

$$u(t) = -\boldsymbol{K}\,\vec{x}(t) + \vec{r}(t) \tag{51}$$

In equation (51), it may be seen the heart of state feedback control, which translates to a pure regulator. The control matrix K is multiplied by the system state vector $\vec{x}(t)$ and the difference is made with the reference signal $\vec{r}(t)$. This is the most general control scheme for state-space systems, but it's not limited to this structure in all applications. To study the stability of the system, it must be calculated the closed-loop system from figure (13), which follows the next equation:

$$\begin{cases} \dot{\vec{x}}(t) = \boldsymbol{A}\,\vec{x}(t) + \boldsymbol{B}\,(-\boldsymbol{K}\,\vec{x}(t) + \vec{r}(t)) \\ \vec{y}(t) = \boldsymbol{C}\,\vec{x}(t) + \boldsymbol{D}\,\vec{u}(t) \end{cases} \Longrightarrow \frac{Y(s)}{R(s)} = \frac{1}{|s\,I - (\boldsymbol{A} - \boldsymbol{B}\,\boldsymbol{K})|}\;[\boldsymbol{C}\,adj(s\,I - (\boldsymbol{A} - \boldsymbol{B}\,\boldsymbol{K}))\,\boldsymbol{B}] \tag{52}$$

From equation (52), as like the PID controller, the elements inside the control matrix K directly impacts the system stability, as the poles of the system will be given by the denominator of the system being equal to zero:

$$|s\,\boldsymbol{I} - (\boldsymbol{A} - \boldsymbol{B}\,\boldsymbol{K})| = 0 \tag{53}$$

As the same way as like the PID controller, this forms the system characteristic equation, and for it to be stable, the poles must have a complex real part (in continuous systems) or inside the complex unit circle (in discrete systems). To ensure its stability, the values of the control matrix must be tuning accordingly to the working requirements: settling time, overshoot, etc. With this objective in mind, there are a wide range of methods for choosing its values: pole placement [35], LQR [10] [33], integral control [5] [50], between others.

Reviewing the aerospace industry antecedents, state-space controller have been vastly used, along with the PID controllers, for solving industry's main problems, specially the pitch angle control problem. Unlike the PID controller, the state-space control is capable of a multivariable control, with multiple inputs and outputs, making this control-scheme really suitable for addressing aerospace problems [8] [6] [9] [7].

4.2.3 MPC Control

The next control scheme to be reviewed will be the *Model Predictive Control* (referred to as MPC). In the early stages of the control industry, an advanced algorithm was whatever scheme that deviates from the general PID structure. For this reason, 80% to 90% of industries' problems were solved using the PID control for safety, economical and quality reasons [27].

Nevertheless, a necessity arose between the industry and the academic field. It was needed a control scheme where the economical constraints could be included in the control structure. This is where the MPC algorithm comes in. The MPC (also known as receding or moving horizon control) uses a wide range of control algorithms where the control action calculation comes from the prediction of the output of the plant to be controlled [14] [27]. Generally, the plant to be controlled and predicted is described using the state-space framework, so the internal information can be accessed by the MPC algorithm, in order to predict and calculate the control action [59].

The receding horizon characteristic of the controller may be seen in the following picture:

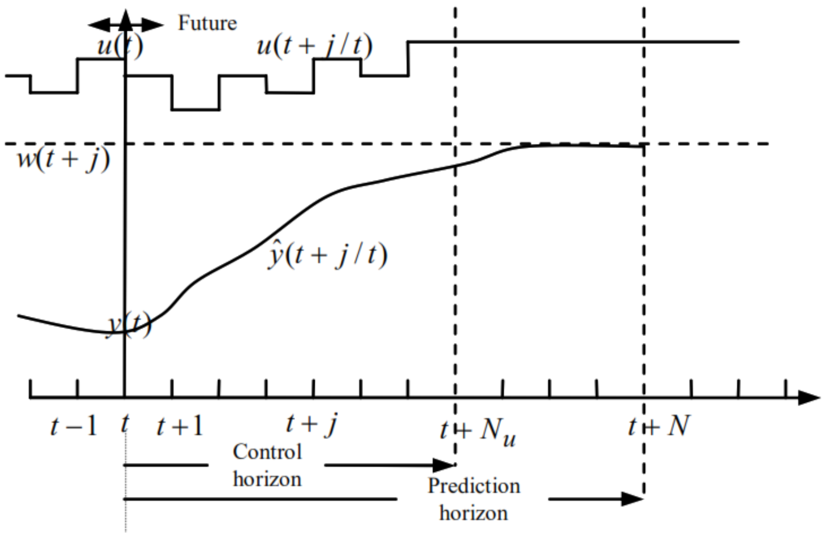


Figure 14: Receeding horizon structure of the MPC [27]

From figure (14), it may be seen the basic workings of the MPC controller: at each time instant j it is solved an optimization process minimizing a quadratic cost index. This cost index, J, follows the next equation:

$$\begin{aligned} J = &\left[(r_{t+1} - y_{t+1|j}) \dots (r_{t+p} - y_{t+p|t})\right] \begin{bmatrix} Q_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & Q_p \end{bmatrix} \left[(r_{t+1} - y_{t+1|j}) \dots (r_{t+p} - y_{t+p|j})\right]^T \\ &+ [\Delta u_t \dots \Delta u_{t+c-1}] \begin{bmatrix} R_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & R_c \end{bmatrix} [\Delta u_t \dots \Delta u_{t+c-1}]^T \end{aligned} \tag{54}$$

summarized in matrix notation as:

$$J = \sum_{j=1}^{p} Q_j \left[r_{t+j} - y_{t+j|t}\right]^2 + \sum_{j=1}^{c} R_j \left[\Delta u_{t+j-1|t}\right]^2 \tag{55}$$

which has to be bounded by the following constraints on the output signal and the control action signal:

$$\begin{array}{ll} \Delta u_{min} \leq \Delta u_{t+j|t} \leq \Delta u_{max} & \text{for} \quad j = 0, 1, \ldots, c-2, c-1 \\ u_{min} \leq u_{t+j|t} \leq u_{max} & \text{for} \quad j = 0, 1, \ldots, c-2, c-1 \\ y_{min} \leq y_{t+j|t} \leq y_{max} & \text{for} \quad j = 1, 2, \ldots, p-1, p \end{array} \tag{56}$$

where Δu_{min} and Δu_{max} are the lower and upper limits for the control action rate, u_{min} and u_{max} are the bounds for the control action signal, and y_{min} and y_{max} are the limits for the response signal. As a remark, the subindex $j|t$ means the prediction in time instant j from the conditions in instant t.

So the control process will take the following structure:

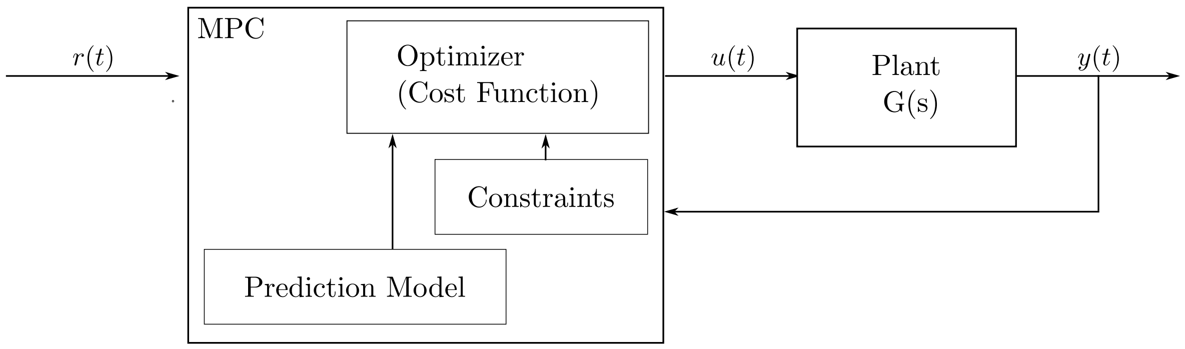


Figure 15: Structure of the MPC controller

From figure (15), and what is has been commented previously, the MPC controller performs an optimization process in any given time instant t, minimizing the cost function in equation (55), predicting the tracking error (the difference between the reference and the output signal) along a certain interval p, multiplied by the tracking error weight matrix Q and the control actions along a second horizon c, multiplied by the control effort weight matrix R, taking into account the constraints in equation (56). Once the optimization process has finished, and all the control action rates have been calculated, only the first one of them is applied, and the process starts again [14] [27].

The physical meaning of the cost function is to penalize large tracking errors and high control efforts, which are forecast in their respective “horizons". This is why it is needed a model as exact as possible, so the information given to the optimizer is precise and exact, so the controller will be able to calculate the most optimal control action. Then, with the optimization process finished, the control action at instant t is calculated by adding the last control action in the last instant with the optimized rate of control effort, which is the result from the optimization process. Then, the calculations start over for the next time instant, $t + 1$, giving the controller the negative feedback capabilities, as with the PID or the state-space control. This feature is what gives the controller its name, the *receding horizon*, where the horizons keep moving with the time instants.

With all of this, the control parameters to be tuned up inside the MPC controller are:

- p: The prediction horizon: How far the controller will predict the tracking error, through the future references and the prediction of the system output, via the internal prediction model. Higher values go for more robustness in the response, but can cause numerical instability.

- c: The control horizon: How far the controller will predict and optimize the rate of control outputs. Higher values will diminish the control effort, making the system dynamics slower but smoother.
- Q and R: The tracking error and rate control input weight: Govern the cost of the tracking error and the rate control input inside the cost function.

All of these parameters control the performance of the controller. Nevertheless, unlike the PID or the state-space control, the MPC has not an analytical solution for the closed loop system, so it can't be analyzed the impact of the control parameters inside the MPC *a priori*. For tuning the controller, it has to to be checked that the performance is acceptable *a posteriori*. Meaning that, the general way of tuning up the MPC controller is, with an accurate prediction model (which is the system plant) and the control objectives (which are the constraints expressed in equation (56)), tune and design the parameters inside the cost function (as expressed in equation (55)). The tuning of the MPC control parameters is given by iterating the closed-loop response until an acceptable performance is reached [49].

In order to tackle this iterative to process, for designing the control parameters inside the MPC controller, there exists a wide range of possibilities to perform this operating in the most orderly fashion possible. From guidelines in order to provide with steps to follow to choose the best parameters [4] [22], a semi-analytical method called controller matching [17], self-tuning methods [20] [18], and many more.

Focused on the aerospace industry, like the PID and the state-space control, the MPC controller has been adopted in various aerospace systems, included the control of an aircraft's pitch angle. Moreover, the MPC is able to adapt to the changing conditions of an aerospace system, modifying accordingly the controller constraints, as well as being well-suited for high non-linear systems [51] [26] [12].

4.2.4 MADRPC Control

The last controller to be reviewed will be a new control scheme, called the *Modified Active Disturbance Rejection Predictive Control* (referred as the MADRPC). This controller arises with the objective in solving a vital problem inside the control industry: reduce the necessary work involving the system modeling process [14]. All of the control schemes reviewed until this point rely on a precise model description for the control process:

- On the one hand, both the PID and the state feedback control need an accurate description of the system dynamics for designing the controller. Both of them need the linear model of the process (either in transfer function or state-space description) to tune up the controller parameters. An inaccurate system description may lead to a wrongful control parameters inside the controller, impacting the system performance.
- On the other hand, the MPC need the system model in state-space description in order to play the role of system prediction model, so the outputs may be calculated. An inaccurate model may lead to wrongful optimized control action rates, which may destabilize or worsen the response performance.

This controller, which is the one proposed in this project to be validated, solves this issue combining a receding horizon controller with an *Active Disturbance Rejection Controller* (referred as ADRC) [14].

The basic structure of the controller is as following:

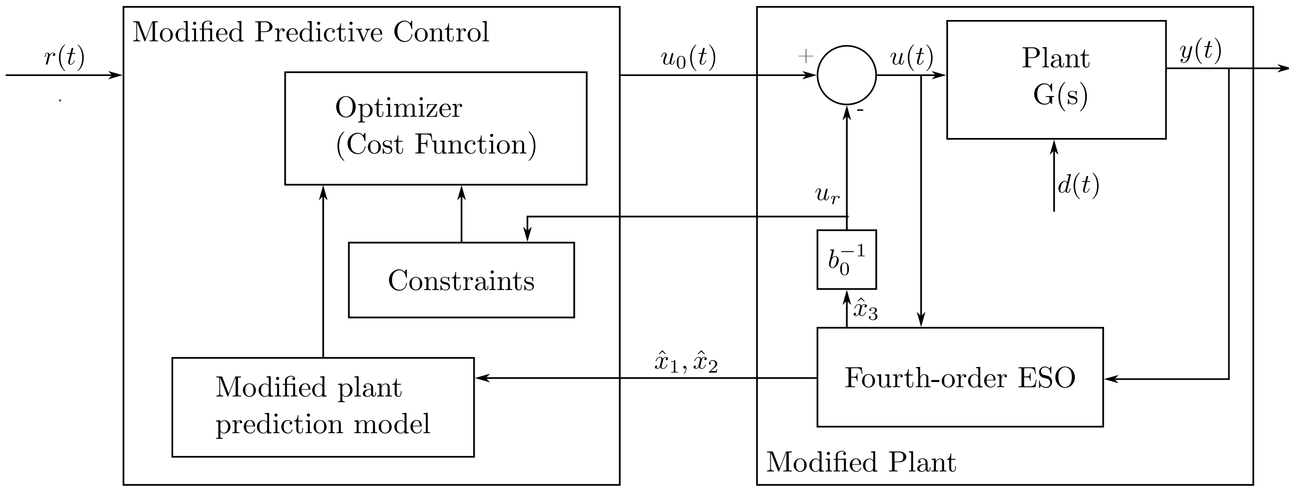


Figure 16: Structure of the MADRPC control scheme [14]

As it may be seen in figure (16), the structure is similar to the one presented with the MPC controller in figure (15). The key difference resides in the prediction model used inside the controller. As said previously, the MADRPC is designed to tackle a very specific problem: be a control scheme when a precise system description is not possible to obtain. The traditional control algorithms rely heavily on some sort of system modeling for them to work. Meanwhile, as see in figure (16), it is used a modified prediction model [14] [40].

The MADRPC combines the optimization process of the MPC with the rejection abilities and that it does not need a precise process model for the LADRC [14]. The optimization process, instead of using the full description of the system (as with the MPC), it used a modified first-order plus integrator system [14]. It is given by the following equation:

$$\ddot{y} = -\frac{1}{T}\dot{y} + \frac{K}{T}u + f \tag{57}$$

where y is the controlled variable, u is the control action, K is the static gain of the system, T is the system time constant, and f represents the total perturbation.

The main advantage this characteristic brings to the table is the reduced time needed for system identification, as the only variables that have to be calculated from the real plant are only two natural internal parameters, K and T, which will be introduced in equation (57). To account for the model mismatch, the total disturbance f is introduced, and using the LADRC controller, the real dynamics are enforced to the modified plant [14] [40].

With the LADRC responsible for the dynamics mismatch between the real model and the modified one, the MPC controller will calculate the optimized rate of control actions using the modified plant in equation (57). Then, the process of the receding horizon controller is the same one as reviewed in the previous section. An optimization process is started, using the prediction model in equation (57) and the cost index in equation (55), under the constraints given in equation (56). It has to be taken into account the disturbance in the modified system, accounting for the mismatch with the real plant, inside the constraints.

Once the predictive controller has finished and calculated the rate of control action to be input to the system, it has to be compensated with the disturbance calculated by the LADRC, making the relationship between the disturbance-free control action and the real control action as [14]:

$$u(t) = u_0(t) - \frac{\hat{x}_3}{b_0} \quad \text{where} \quad b_0 = \frac{K}{T} \tag{58}$$

where $u(t)$ is the real control action and $u_0(t)$ is the disturbance-free control action (is the output from the receding horizon controller using the modified plant). Then, the $\hat{x}_3$ represents the disturbance in the system, mainly from the mismatch between the real and the modified plants, which is introduced as a state inside the system. This way, a fourth-order *Extended State Observer* (ESO) will be used to estimate the system states: $\hat{x}_1$ and $\hat{x}_2$ will go back to the predictive controller, to be used inside for the prediction model, and $\hat{x}_3$ will be used to compensate the mismatch between the real model and the modified one.

With all of this, the control parameters to be tuned up inside the MADRPC controller are:

- K and T: The natural parameters to be used inside the modified prediction model, the first-order plus integrator model in equation (57). These values must be calculated using the response signal of the real plant and fitting the modified plant to the real response [14].
- ω_o: Is the ESO bandwidth, that must be tuned up for correctly estimating the modified system's states, necessary for successfully correcting the mismatch between the real model and the modified plant [14].
- p, c, Q and R: The inherited predictive controller parameters from the MPC. They must be tuned up correctly in order to calculate the optimized disturbance-free control action rates.

Unlike the control schemes reviewed in this section, the MADRPC is of very recent creating, not having yet any presence in the control industry. The first reference to it is given by [40], but this project will be the first one that will have a result related to the aerospace industry, as the MADRPC will be tested with the control of a pitch angle and validated against traditional control schemes.

4.2.5 Choosing for Control Strategies

For this project, it will be used the MADRPC controller, as the main objective of this work is to test the performance of a controller that uses no prior information of the system to control (apart from two natural parameters) and compare it with the traditional control schemes, which uses the complete system description.

The traditional control schemes that will be used alongside the MADRPC will be the PID and the MPC controllers. The reason behind this is the robustness and practical availability, as these two are for more used inside the control industry and the digital systems in general, and so, it makes more sense to use them to compare their performance with the MADRPC.

5 Design of the Control Schemes

Once reviewed the state of the art regarding control algorithms, from the more traditional control schemes (PID or MPC) to the newer ones (this work proposed scheme: the MADRPC), in the last chapter (4), it is now time to design those. This chapter aims to the design of three control algorithms to govern the H200 angle of attack. With this objective in mind, first of all it is needed the dynamic modeling for the H200, seen in chapter 3, where it was developed the non-linear dynamic system. The steps to be followed will be:

1. First of all, calculate the linear model for the H200, considering certain flight conditions (altitude and velocity). As a result, it will be derived the state-space modeling of the H200.
2. Secondly, from the state-space model, it will be extracted the transfer function relating the angle of attack (or pitch angle, as they will be considered the same as a matter of simplicity) and the elevator deflection.
3. Lastly, with the linear transfer function, it will be designed the three control algorithms (PID, MPC and MADRPC) to be used in nominal conditions. These control designs will be tested in the next chapter against the non-linear model and experimented on changing the nominal flight conditions.

As a remark, the linear model it is needed as these control algorithms are formulated as SISO controllers (Single-input, single-output) and the tuning algorithms used take as a starting point having the linear model of the process.

5.1 Linear Dynamics Modeling

So, as explained before, the first step is to calculate the H200 linear model, starting from the non-linear dynamic equations and extracting the desired transfer function.

5.1.1 H200 Linear Model

The starting point is to set the equilibrium flight conditions for the H200: the altitude and the velocity. In this project, the nominal flight conditions will be:

Cruise Flight	**Conditions**
Altitude	100 m
Velocity	21 m/s

Table 8: Nominal H200 flight conditions

Then, the next step is to define the state variables, the inputs and the outputs of the linear model. Starting from the state variables, they will be the body linear velocities (u, v, w), the body angular velocities (p, q, r) and the Euler angles: roll (ϕ), pitch (θ) and yaw (ψ). So, the state vector, represented by $\vec{x}$, will be:

$$\vec{x} = \begin{bmatrix} u \\ v \\ w \\ p \\ q \\ r \\ \phi \\ \theta \\ \psi \end{bmatrix} \tag{59}$$

This state vector holds the minimum range of variables, independent from each other, which contains the complete information of the system behavior in any given time.

On the other hand, the system input vector will correspond with the H200 physical inputs:

- Aileron deflection (δ_A)
- Elevator deflection (δ_E)
- Throttle level (δ_P)
- Rudder deflection (δ_R)
- Flaps deflection (δ_F)

And the system output vector will consist of the state variables, represented by the vector $\vec{x}$, the inertial position and the flight bearing. This last variable, represented by ξ is calculated using the following equation:

$$\xi = tan^{-1}\left(\frac{\dot{y}}{\dot{x}}\right) \tag{60}$$

So, the input and output vectors, represented as $\vec{u}$ and $\vec{y}$ respectively will be:

$$\vec{u} = \begin{bmatrix} \delta_A \\ \delta_E \\ \delta_P \\ \delta_R \\ \delta_F \end{bmatrix} \qquad \vec{y} = \begin{bmatrix} \vec{x} \\ x \\ y \\ z \\ \xi \end{bmatrix} \tag{61}$$

Then, with the election of the state, input and output vector is done, the linearization may proceed. The objective is to develop the H200 linear state-space model. The steps that are going to be followed are:

1. First, from the nominal flight conditions, calculate the equilibrium point of the system.
2. Then, with the equilibrium point calculated, obtain its linear state-space form, which will be expressed as:

$$\dot{\vec{x}} = \boldsymbol{A}\,\vec{x} + \boldsymbol{B}\,\vec{u} \tag{62}$$
$$\vec{y} = \boldsymbol{C}\,\vec{x} + \boldsymbol{D}\,\vec{u} \tag{63}$$

where $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ and $\boldsymbol{D}$ are the state space matrices, that describe the linear model around the operating point.

So, the equilibrium point of the H200, using the nominal flight conditions specified in table (8) and the non-lineal dynamic equations will be:

$$\vec{u}_0 = \begin{bmatrix} \delta_{A_0} \\ \delta_{E_0} \\ \delta_{P_0} \\ \delta_{R_0} \\ \delta_{F_0} \end{bmatrix} = \begin{bmatrix} 0 \\ 0.0220 \\ 0.5392 \\ 0 \\ 0 \end{bmatrix} \qquad \vec{x}_0 = \begin{bmatrix} u_0 \\ v_0 \\ w_0 \\ p_0 \\ q_0 \\ r_0 \\ \phi_0 \\ \theta_0 \\ \psi_0 \end{bmatrix} = \begin{bmatrix} 20.9785 \\ 0 \\ 0.9491 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0.0452 \\ 0 \end{bmatrix} \tag{64}$$

This operating point has been calculated setting the state variables derivatives to 0 and solving the non-linear equations to calculate them. Then, to extract the state-space matrices, it is needed to do the jacobian of the non-linear equations. For this, may the state variables non-linear equations expressed as:

$$\dot{\vec{x}} = \vec{f}(\vec{x}, \vec{u}) \tag{65}$$

And the output equations as:

$$\vec{y} = \vec{q}(\vec{x}, \vec{u}) \tag{66}$$

The state-space matrices will be calculated as:

$$\boldsymbol{A}_{ij} = \left.\frac{\partial f_i}{\delta x_j}\right|_{\vec{x}_0, \vec{u}_0} \qquad \boldsymbol{B}_{ij} = \left.\frac{\delta f_i}{\delta u_j}\right|_{\vec{x}_0, \vec{u}_0} \qquad \boldsymbol{C}_{ij} = \left.\frac{\delta q_i}{\delta x_j}\right|_{\vec{x}_0, \vec{u}_0} \qquad \boldsymbol{D}_{ij} = \left.\frac{\delta q_i}{\delta u_j}\right|_{\vec{x}_0, \vec{u}_0} = 0 \tag{67}$$

As a remark, the elements for matrix $\boldsymbol{D}$ will be 0 as the output does not depend on the input. With this, the state-space matrices will be:

$$\boldsymbol{A} = \begin{bmatrix} -0.0765 & 0 & 0.3614 & 0 & -0.9787 & 0 & 0 & -9.7966 & 0 \\ 0 & -0.2296 & 0 & 0.9702 & 0 & -20.8179 & 9.7966 & 0 & 0 \\ -0.7261 & 0 & -4.4907 & 0 & 19.6955 & 0 & 0 & -0.4432 & 0 \\ 0 & -2.5129 & 0 & -16.6757 & 0 & 5.2318 & 0 & 0 & 0 \\ 0.0637 & 0 & -0.7346 & 0 & -3.0135 & 0 & 0 & 0 & 0 \\ 0 & 0.2821 & 0 & -0.6131 & 0 & -0.8085 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0.0452 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.0010 & 0 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{B} = \begin{bmatrix} 0 & 0.0384 & 12.1186 & 0 & -0.0057 \\ 0.2226 & 0 & 0 & -0.5069 & 0 \\ 0 & 1.6632 & 0.0028 & 0 & -0.2488 \\ 56.6129 & 0 & 0 & -1.9864 & 0 \\ 0 & 24.1562 & 0.0176 & 0 & -0.0354 \\ 0.4019 & 0 & 0 & 1.7456 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{C} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0.9990 & 0 & 0.0452 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & -0.9491 & 0 & 21 \\ -0.0452 & 0 & 0.9990 & 0 & 0 & 0 & 0 & -21 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad \boldsymbol{D} = 0$$

5.1.2 Linear Model for the Pitch Angle

It has to be remembered that this state-space model represents the linear model for the complete H200 flight. There exist multi-variable controllers (state-space control, LQR, etc), but for the scope of this project, the controllers that will be designed are SISO (*Single Input, Single Output*). So, the last step is to extract the specific transfer function relating the angle of attack (or the pitch angle) with the elevator angle, which is generally the main control input used to govern this variable.

For this it has to be chosen the correct row and column corresponding to the pitch angle and the elevator angle. Once the state-space model has been simplified, it is used the following expression to calculate the transfer function:

$$G(s) = \frac{1}{|s\,\boldsymbol{I} - \boldsymbol{A}|}\,\left[\boldsymbol{C}\,adj(s\,\boldsymbol{I} - \boldsymbol{A})\,\boldsymbol{B}\right] \tag{68}$$

At the end, the transfer function that will be used in this project to design the controllers will be:

$$G_\theta(s) = \frac{\theta(s)}{\delta_E(s)} = \frac{24.16\,s^2 + 109.1\,s + 14.62}{s^4 + 7.581\,s^3 + 28.9\,s^2 + 3.581\,s + 8.014} \tag{69}$$

Now, once the transfer function has been calculated, the next step will be to design the nominal controllers.

5.2 Design of the Nominal Controllers

With the linear transfer function representing the pitch angle process, the next step is to design and tune up the controllers for the project, which will be used to compare their performances. Depending on the controller, there exist a wide range of methods to tune up their control parameters.

For this project, it will be used a common method among all three of them, which is a response optimization process. This method will focus on minimizing an error performance indicator among all three controllers, depending on their internal parameters. The performance indicator to be minimized will be the *Integral Square Error* (referred to as the ISE). The ISE follows the next equation:

$$ISE = \int_0^\infty e^2(t)dt \tag{70}$$

The process to be followed for the controllers tuning is:

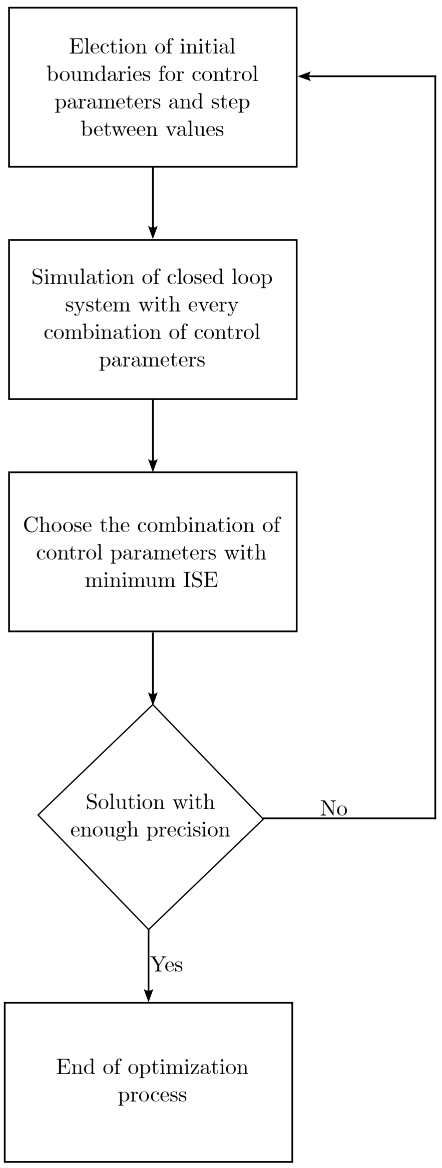

Election of initial boundaries for control parameters and step between values
Simulation of closed loop system with every combination of control parameters
Choose the combination of control parameters with minimum ISE
Solution with enough precision
No
Yes
End of optimization process

Figure 17: Optimization process for the controllers tuning

The steps that are described in figure (17) are:

1. The first step is to choose the boundaries for the control parameters. They consist of the minimum and maximum value and the step between these values. They have to be chosen reasonably not to increase the calculation time unnecessarily.
2. Then, the closed loop simulation is performed with every combination of the control parameters. The election of their boundaries and the step between them are of vital importance, as the calculation time of this step increases exponentially with the increase of number of control parameters to be tested.
3. Then, once the closed loop simulations are done, the control parameters combination which yielded the minimum value of ISE is chosen.
4. Then, if the solution has enough precision, the process ends. If not, the process repeats iteratively reducing the boundaries limits and the step between them, in order to choose a precise enough solution.

Moreover, additionally to the ISE performance indicator, the response overshoot (δ) and the settling time (T_{set}) will be used as auxiliary performance indicators. As a remark, the controllers will be designed in discrete form, the PID at least, as the MPC and the MADRPC are inherently discrete controllers. The sample time for the tests, and thus the sample frequency will be:

$$T_s = 0.01\,\mathrm{s} \quad \Longleftrightarrow \quad f_s = 100\,\mathrm{Hz} \tag{71}$$

With the general considerations reviewed, it will be started the process of design and tuning of the controllers. For this, it will be used the transfer function calculated in equation (69):

$$G_\theta(s) = \frac{\theta(s)}{\delta_E(s)} = \frac{24.16\,s^2 + 109.1\,s + 14.62}{s^4 + 7.581\,s^3 + 28.9\,s^2 + 3.581\,s + 8.014} \tag{72}$$

which is, to remind, the linear model representing the aircraft pitch angle process in nominal cruise conditions (equilibrium point).

5.2.1 PID Controller

The first controller to be designed will be the PID control. As commented, this algorithm is one of the simplest and most effective control schemes, based on the feedback controller structure, as in figure (12).

Inside the PID controllers, there exists a wide range of possibilities for expressing them om a design-friendly way: continuous, discrete, ideal, parallel, etc. As said, this project will use the discrete ideal form, which is the one used in the majority of digital applications (and inside the chosen simulation platform: MATLAB and Simulink). The PID controller takes the following form:

$$\delta_E(t) = K_P + K_I\,T_s\,\frac{1}{z-1} + K_D\,\frac{1}{T_s}\,\frac{z-1}{z} \tag{73}$$

where T_s is the simulation sample time, set to 100 Hz as in equation (71), and z is the operator used when doing the discretization process of the controller.

With the PID structure chosen, the control parameters that will be tune up are:

- K_P: Is the gain responsible for the proportional action.
- K_I: Is the gain responsible for the integral action.
- K_D: Is the gain responsible for the derivative action.

They will be tune up to minimize the ISE performance indicator, as in equation (70). With all of this, it has been carried out tests for minimizing the ISE cost function, ensuring the best response against a step input. Following diagram (17), the steps carried out were:

1. Choose the initial boundaries for the K_P, K_I and K_D. The initial values were:
   - For K_P, the interval was $[1, 100]$, with steps of five between the values.
   - For K_I, the interval was $[1, 10]$, with steps of one between the values.
   - For K_D, the interval was $[1, 10]$, with steps of one between the values.

   These values were chosen attending to various sources when using a PID controller for the pitch angle control.

2. Then, it was carried out the linear closed loop simulations with every combination of K_P, K_I and K_D. The PID controller block is build using equation (73), filling the control parameters values with the corresponding ones of the combination. As a remark, as commented, the ISE is calculated in every one of them.

3. With the simulations all completed, the minimum combination of K_P, K_I and K_D were chosen, responsible for the minimum response ISE.

4. The process continued reducing the boundaries in both sides from the optimal solution, dividing the step size by 10, in order to calculate an accurate enough solution around its neighbors.

The results may be seen in figure (18):

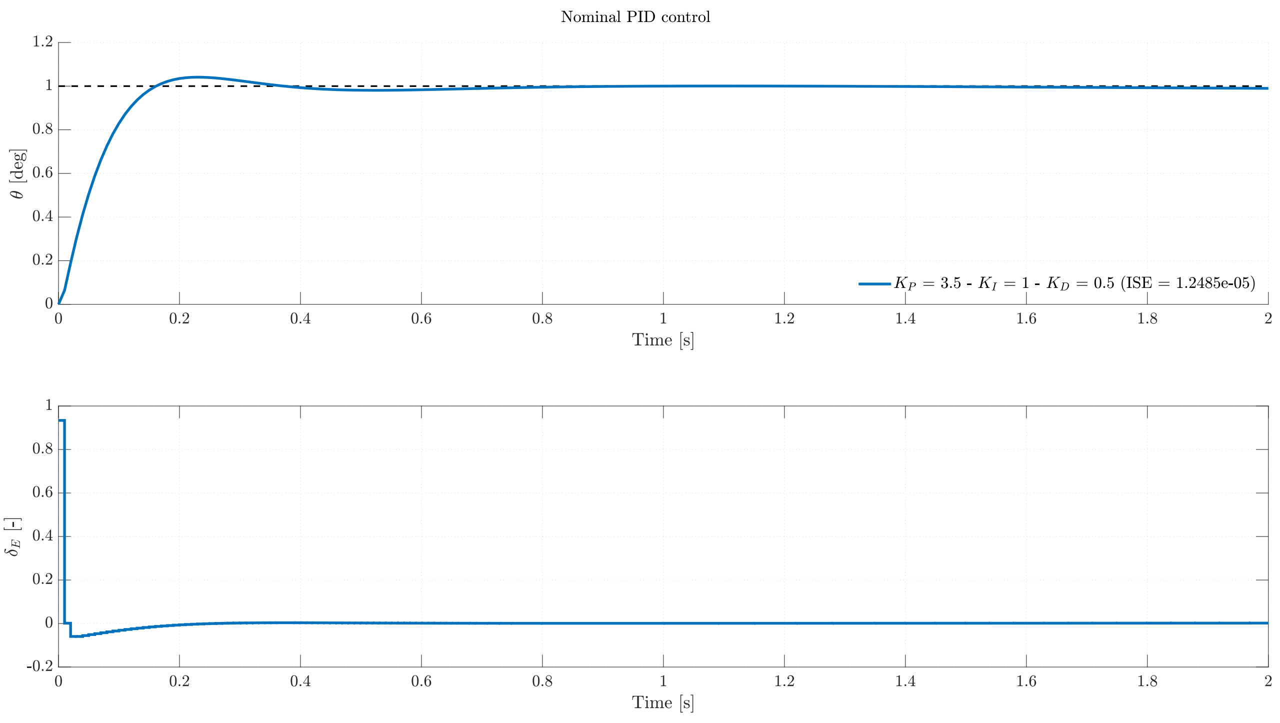


Figure 18: Pitch angle control with nominal PID controller

The optimal PID control parameters and its performance are:

K_P	K_I	K_D	ISE [-]	δ [%]	T_{set} [s]
3.5	1	0.5	$1.2795 \cdot 10^{-5}$	1.2485	0.8

Table 9: Nominal PID control parameters and performance

5.2.2 MPC Controller

The next controller to be designed will be the MPC control. As with the PID controller, the MPC also counts up with a wide range of structures in order to achieve different objectives. For this project, in order to have the same performance level as the PID controller, it will be used the *Offset-Free MPC* (referred as OF-MPC). Using this structure, it will be assure that the response signal has no steady error [14].

The first step in the process is, as the controller is based in state space description of the system, the pitch angle transfer function must be converted to discrete state space, using the equivalent transformation as in equation (52) from the transfer function in equation (69):

$$\begin{aligned} \vec{x}_{k+1} &= \boldsymbol{A}_\theta\, \vec{x}_k + \boldsymbol{B}_\theta\, \delta_{E_k} + \vec{\nu}_k \\ \theta_k &= \boldsymbol{C}_\theta\, \vec{x}_k \end{aligned} \tag{74}$$

where $\vec{x}$ is the state vector, $\vec{\nu}$ is the state disturbance (measured and not measured), θ_k is the pitch angle at instant k, δ_{E_k} is the elevator deflection at instant k and $\boldsymbol{A}_\theta$, $\boldsymbol{B}_\theta$ and $\boldsymbol{C}_\theta$ are the system state-space matrices [59].

Once the structure for the MPC controller is chosen, it will be reviewed the control parameters that will be tuned up using the optimization method:

- p: Is the prediction horizon, controlling how far the controller view and predict the tracking error of the system.
- c: Is the control horizon, controlling the amount of control actions will be optimized.
- Q and R: The tracking error and control action weights inside the cost index in equation (55)

The constraints for the optimization process are:

$$\begin{aligned} \Delta\delta_{E_{min}} &= -0.5 \qquad \delta_{E_{min}} = -1 \qquad \theta_{min} = -2 \\ \Delta\delta_{E_{max}} &= 0.5 \qquad \delta_{E_{max}} = 1 \qquad \theta_{max} = 2 \end{aligned} \tag{75}$$

these values are chosen following the physical limitations of the aircraft, both the pitch angle and the elevator deflection system.

With the specific system parameters all set up, the optimization process for tuning the OF-MPC parameters is started. Again, following the diagram presented in figure (17), the steps for the process were:

1. Choose the initial boundaries for p, c, Q and R. The initial values were:
   - For the prediction horizon, p, the interval was $[1, 100]$, with steps of five units.

- For the control horizon, c, the interval was $[1, 100]$, with steps of five units.
- For the tracking error and the control effort gains, the intervals were $[1, 10]$, with a step of one unit between the values.

Once again, these values were chosen following various references using the MPC for controlling the pitch angle.

2. Then, as with the PID control, it were carried out the linear closed loop simulations, with every combination of the control parameters. These values are then used inside the MPC block, filling the cost index in equation (55). As opposite to the PID controller, the MPC does not have a closed analytical form, as it is based on an optimization problem. Again, the ISE is calculated in every simulation performed.

3. With the simulations all completed, the combination of p, c, Q and R which yielded the minimum ISE is chosen.

4. Then process continued reducing the boundaries in both sides for the control parameters, around the optimal solution found, dividing the step size by 10. The boundaries reformulation attends the performance impact of every control parameter on the response, adapting them to the desired output behavior. As a consequence, it will be found the combination of control parameters precise enough to solve the control problem.

Once the process has been completed, it yielded the following result, as seen in figure (19):

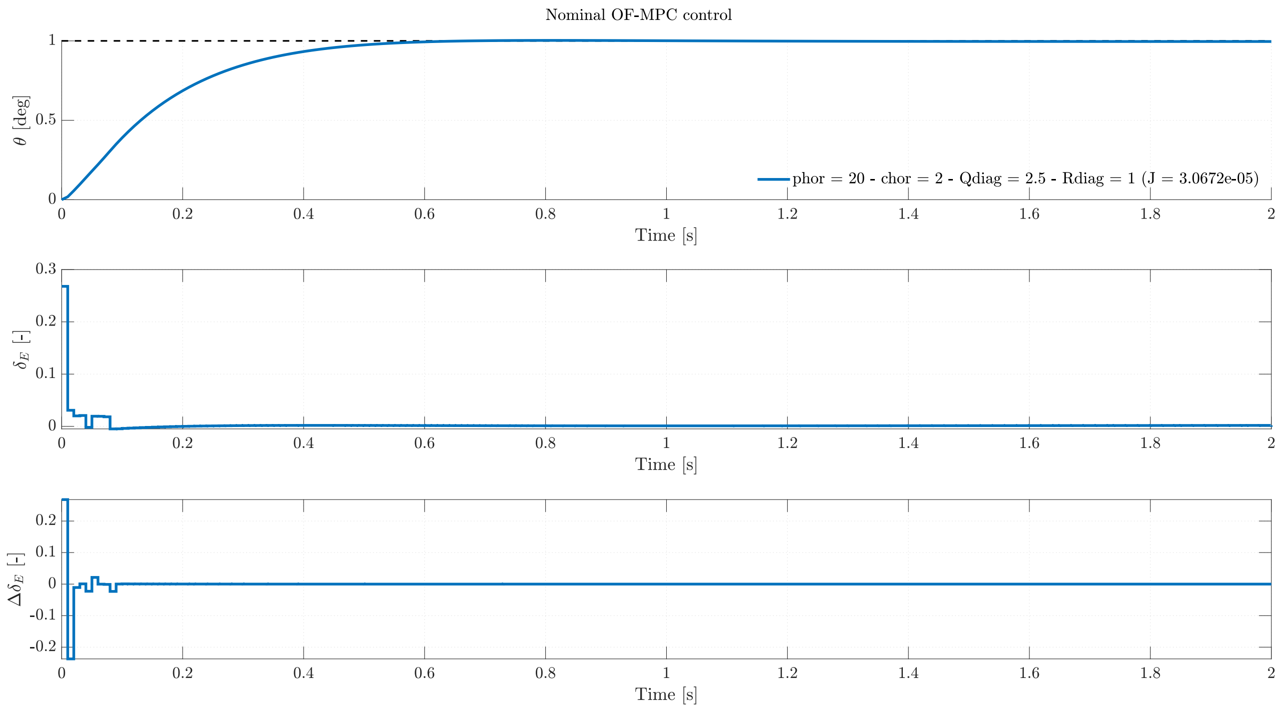


Figure 19: Pitch angle control with nominal OF-MPC controller

The optimal MPC control parameters and its performance are:

p	c	Q_i	R_i	ISE [-]	$\delta[\%]$	T_{set} [s]
20	2	2.5	1	$3.0672 \cdot 10^{-5}$	-	0.7

Table 10: Nominal MPC control parameters and performance

5.2.3 MADRPC Controller

Last but not least, it will be designed the MADRPC control, the controller proposed for this project to validate it for aerospace use, through the control of the H200 pitch angle. As it was previously commented in section 4, the MADRPC is designed to solve one specific problem: reducing the system modeling process. It is a controller that combines the predictive characteristics of a receding horizon controller with the disturbance rejection characteristics of the LADRC.

The main idea is to assume a modified plant describing the pitch angle, through a first-order plus integrator model, as expressed in equation (57):

$$\ddot{\theta} = -\frac{1}{T}\dot{\theta} + \frac{K}{T}\,\delta_E + f \tag{76}$$

where K is the static gain of the process, T is the plant time constant and f is the total perturbation inside the system. This modified plant is the one used inside the receding horizon, where the disturbance-free elevator deflections are calculated, and then the LADRC is responsible for enforcing the system real dynamics. In other words, it is responsible of calculating the total perturbation, f, and to be substracted from the modified system [14]. To be used inside the receding horizon control, like the pure MPC control, the system must be expressed in state-space framework:

$$\begin{gathered}\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & -\frac{1}{T}\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}0\\ \frac{K}{T}\end{bmatrix}\delta_E + \begin{bmatrix}0\\ 1\end{bmatrix} f\\ \theta = \begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}\end{gathered} \tag{77}$$

and with a discretization process applied, the modified system will be:

$$\begin{gathered}\vec{x}_{k+1} = \begin{bmatrix}1 & T\,(1-a)\\ 0 & a\end{bmatrix}\vec{x}_k + b_0\begin{bmatrix}T\cdot T_s - T^2(1-a)\\ T\,(1-a)\end{bmatrix}\delta_{E_k} + \begin{bmatrix}T\cdot T_s - T^2(1-a)\\ T\,(1-a)\end{bmatrix} f_k\\ \theta_k = \begin{bmatrix}1 & 0\end{bmatrix}\vec{x}_k\end{gathered} \tag{78}$$

Once the specific structure of the MADRPC has been defined, the the control parameters that will be tuned up using the optimization method are:

- b_0: Is the nominal value of the plant critical gain, calculated as $b_0 = \frac{K}{T}$.
- T: The apparent time constant for the modified plant. With this value and the value of b_0, it may be calculated the static gain of the system.
- ω_o: Desired bandwidth for the ESO inside the controller, in order to estimate the modified states of the system.
- p, c, Q and R: The control parameters inherited from the receding horizon controller, responsible of the disturbance-free control action rates.

The constraints for the optimization process, which is the same as the pure OF-MPC controller, are:

$$\begin{aligned}\Delta\delta_{E_{min}} &= -0.5 \qquad & \delta_{E_{min}} &= -1 \qquad & \theta_{min} &= -2\\ \Delta\delta_{E_{max}} &= 0.5 & \delta_{E_{max}} &= 1 & \theta_{max} &= 2\end{aligned} \tag{79}$$

It has to be remarked that the plant information from equation (69) is not used for the controller design. The only information from the model is to calculate the critical gain b_0, which is the ratio between the static gain K and the apparent time T. These two are two natural parameters from the plant, easily estimated from an input-output response analysis.

With the main information from the MADRPC reviewed, the optimization process for tuning the control parameters is started. Again, as with the PID and the MPC controllers, the objective is to reduce the ISE of the response. The steps followed were:

1. First, the initial boundaries for b_0, T, ω_o, p, c, Q and R are chosen. For b_0 and T, it is used a linear regression of the pitch angle process to a first order plus delay system, as in equation (57). This way, it will be calculated the initial values of this parameters. Then, in order to give a more accurate result, they will be given some boundaries around this initial value, so a more accurate value will be given. The initial boundaries were:

   - For b_0, the nominal critical gain value, the initial interval was $[25, 35]$, with a step of one unit between them.
   - For T, the apparent time constant, the initial interval was $[15, 25]$, with a step of one unit between them.

   And then, for the rest of control parameters:

   - For the observer bandwidth, ω_o, the interval was $[1, 50]$, with a step of five units between the values.
   - For the predictive control parameters, they are similar to the classic predictive controller commented in last chapter, with the intervals being: for the prediction and the control horizons, p and c, the intervals were $[1, 200]$ with a step size of 10 units between the values. This is, as the modified plant is a first-order system, the interval will be wider than the classic problem, in order to give more flexibility to the optimization process. Then for the tracking error and the control effort gains, Q and R the intervals were $[1, 100]$ with a step size of 10 units between them.

2. Then, it is carried out the linear closed loop simulations with every combination of the control parameters. As with the MPC controller, the MADRPC controller does not have an analytical equation, as it is based on an optimization problem. Each of the control parameters are used to fill out the values inside the modified plant and the cost function in equations (57) and (55) respectively.

3. With the simulations completed, the minimum combination of the control parameters are chosen attending to the one that resulted in the minimum ISE of the response.

4. The process continues, readjusting the control parameters boundaries to meet the desired performance, reducing the boundaries and the step size too. So a precise enough solution may be calculated that solves the control problem.

With the tuning process finished, the result can be seen in figure (20)).

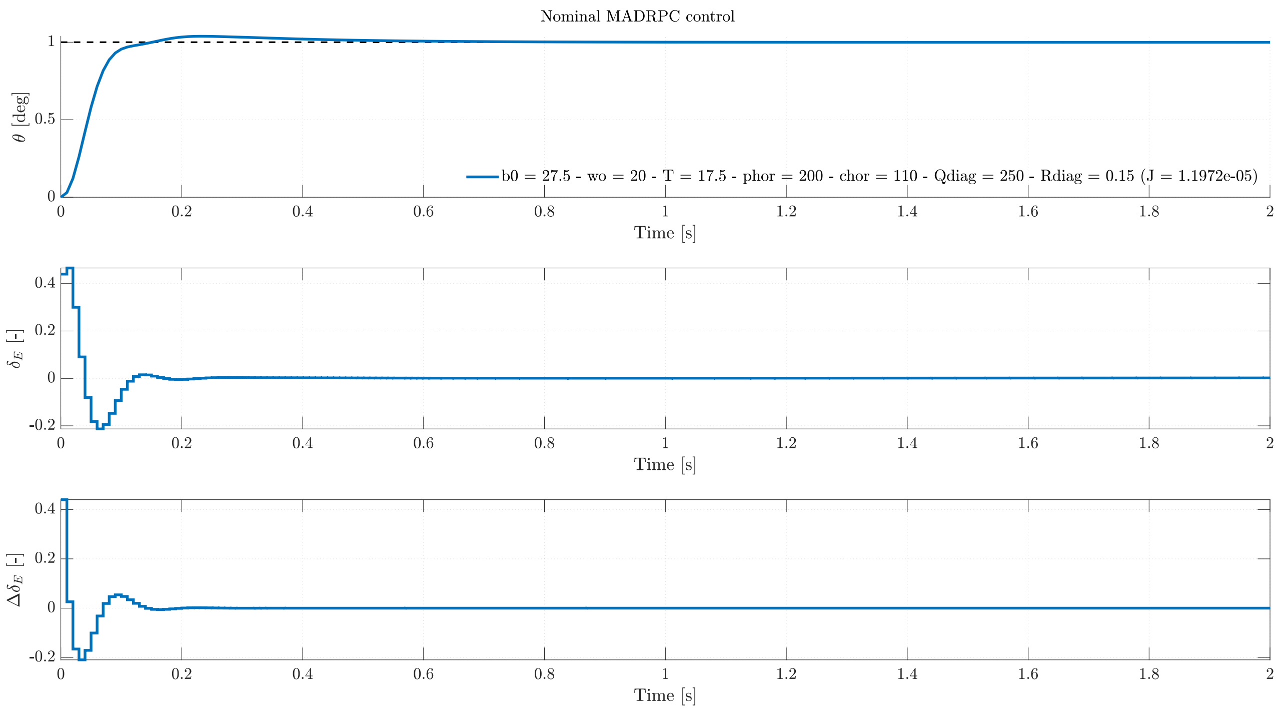


Figure 20: Pitch angle control with nominal MADRPC controller

The optimal MADRPC control parameters and its performance are:

b_0	ω_o	T	p	c	Q_i	R_i	ISE [-]	δ[%]	T_{set} [s]
27.5	17.5	20	200	110	250	0.15	$1.1972 \cdot 10^{-5}$	3.8216	0.6

Table 11: Nominal MADRPC control parameters and performance

Once the nominal controllers have been tuned up and tested against the pitch angle linear process, the next chapter will focus of testing these controllers against the H200 complete non-linear model in cruise flight conditions and against changes in them.

6 Visualization and Results Comparison

The last chapter will cover the validation of the controllers tuned in last section. For this, they will be tested against the complete non-linear H200 model, both in cruise flight conditions and with external perturbations. For this, it has been built a H200 Simulink model to run the simulations.

6.1 Implementation of H200 Non-Linear Model in Simulink

The first step is to develop the H200 non-linear dynamic equations, explained in section 3. For this objective, it is going to be used Simulink, a block-based MATLAB environment, where it is possible developing all type of models using blocks. For this project, the main block that is going to be used will be:

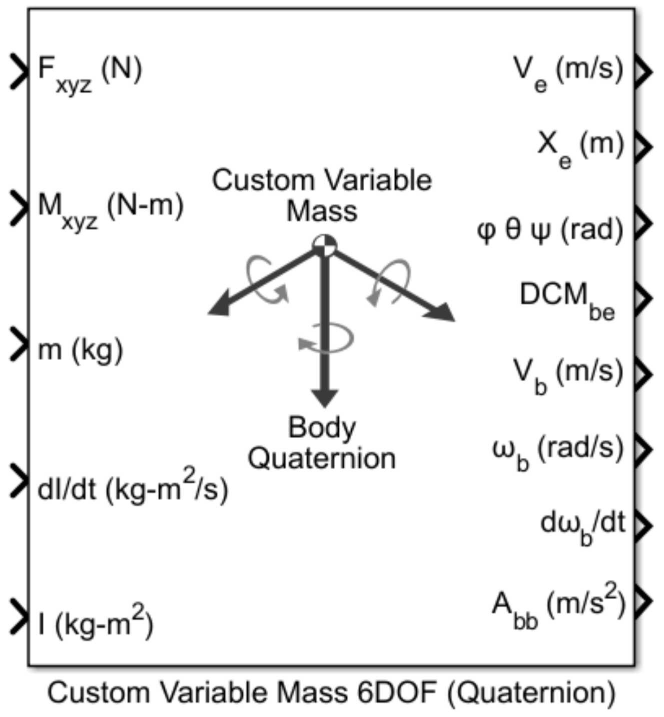


Figure 21: 6-DoF dynamic model block

This block is the cornerstone of the non-linear simulation, as it contains the complete rigid body dynamic equations developed in section (3). The inputs of this block are [41]:

- F_{xyz}, M_{xyz}: They represent the forces and moments around the center of gravity of the aircraft, with units N and $N \cdot m$ respectively.
- dm/dt, m: They represent the mass rate of change and the aircraft mass, expressed in kg/s and kg.
- dI/dt, I: They represent the inertial rate of change and the aircraft's inertia matrix, expressed in $kg \cdot m^2/s$ and $kg \cdot m^2$.
- V_{re}: It's the relative velocity at which the mass is expelled from the aircraft, expressed in m/s

And the outputs from the block are [41]:

- V_e, X_e: Aircraft velocity and position in flat Earth reference frame, expressed in m/s and m, respectively.
- ϕ, θ, ψ: They are the Euler rotation angles: roll, pitch and yaw, expressed in rad all of them.
- DCM_{be}: It represents the Direct-Cosine matrix, giving the transformation from flat-Earth representation to body frame.
- V_b, ω_b: They represent the aircraft's body linear and angular velocities, expressed in m/s and rad/s.
- $d\omega_b/dt$, A_{bb}, A_{be}: They represent the aircraft's angular acceleration, linear accelerations in body-fixed frame and linear accelerations expressed in the flat-Earth reference. They are expressed in rad/s^2 (angular acceleration) and m/s^2 (linear accelerations).

So, seeing the inputs from the block, it has to be developed the forces and moments acting on the H200's center of gravity in each of the steps. The rest of the inputs will be 0, as neither the aircraft's mass nor inertia changes in time, and there is no mass expelled from the body.

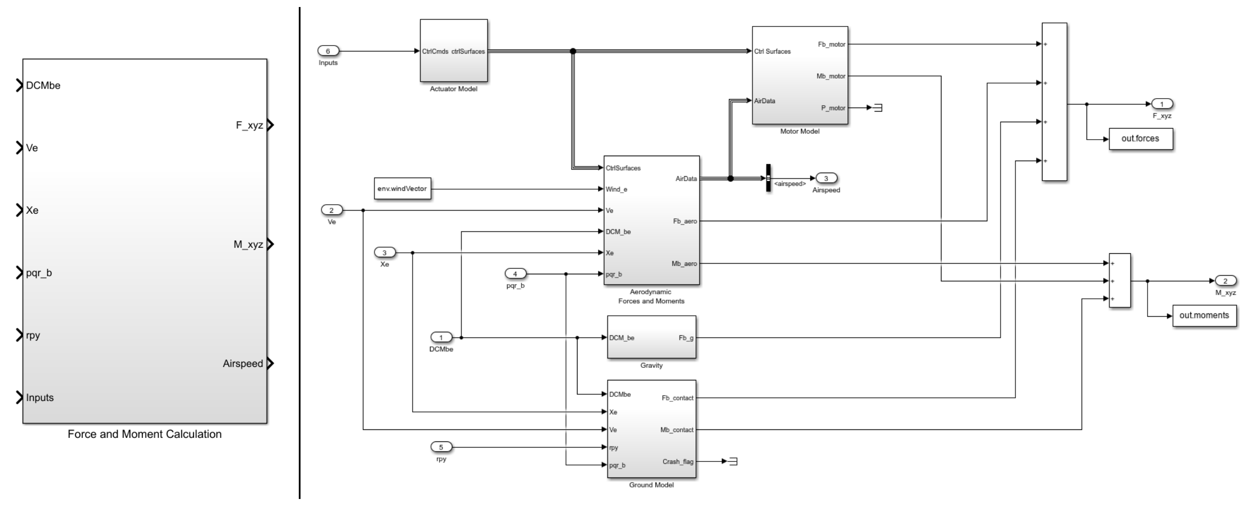


Figure 22: Forces and moments calculation block

In figure (22) it may be seen the subsystem used in Simulink for the calculation of aircraft's forces and moments around the center of gravity. The list of blocks inside are:

- *Actuator Model*: It represents the internal modeling for the H200's physical inputs.
- *Motor Model*: It represents the H200's propeller model, giving the force, moments and power from the engine.
- *Aerodynamic Forces and Moments*: This block has inside the H200's aerodynamic force and moments model.
- *Gravity*: It contains the gravity model for the simulation.
- *Ground model*: It contains the force and moments in the point of contact of the H200 with the ground.

6.1.1 Actuator Model

Inside the *Actuator Model* block, it is represented the physical model for the actuators. They will be represented as second order actuators, with a certain natural frequency and damping ratio (calculated through experimentation):

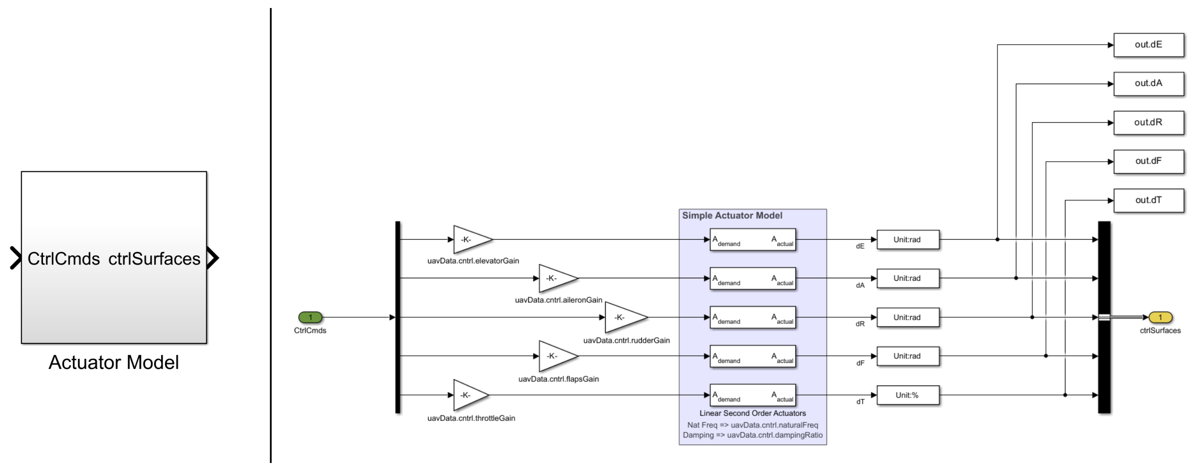

CtrlCmds
ctrlSurfaces
Actuator Model
CtrlCmds
-K-
uavData.cntrl.elevatorGain
uavData.cntrl.aileronGain
uavData.cntrl.rudderGain
uavData.cntrl.flapsGain
uavData.cntrl.throttleGain
Simple Actuator Model
Linear Second Order Actuators
Nat Freq => uavData.cntrl.naturalFreq
Damping => uavData.cntrl.dampingRatio
dE
dA
dR
dF
dT
Unit:rad
Unit:%
out.dE
out.dA
out.dR
out.dF
out.dT
ctrlSurfaces

Figure 23: Actuator model Simulink block

Also, it will be included here the specific inputs gains for each of them, so the external inputs will be expressed in the range of $[-1, 1]$ and then transformed to have physical meaning.

6.1.2 Motor Model

The next block to be reviewed will be *Motor Model*:

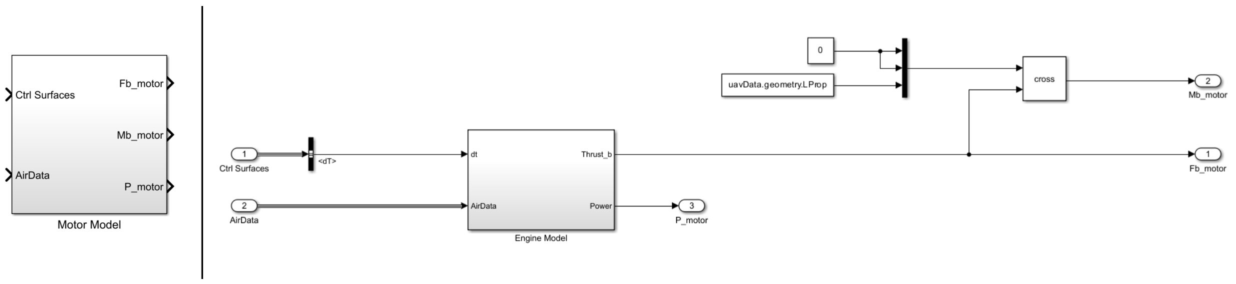


Figure 24: Motor model Simulink block

Here it is calculated the engine's force, moments and power. For the force and moments, calculated in *Engine Model*, it will be used equations (34 - 39). For the moments, it will be calculated through the cross product between the engine's force and the propeller distance vector to the H200's center of gravity.

6.1.3 Aerodynamic Forces and Moments

The next block to be reviewed will be the *Aerodynamic Forces and Moments* block. Its purpose is to calculate the aerodynamic forces and moments in any given instant for the H200:

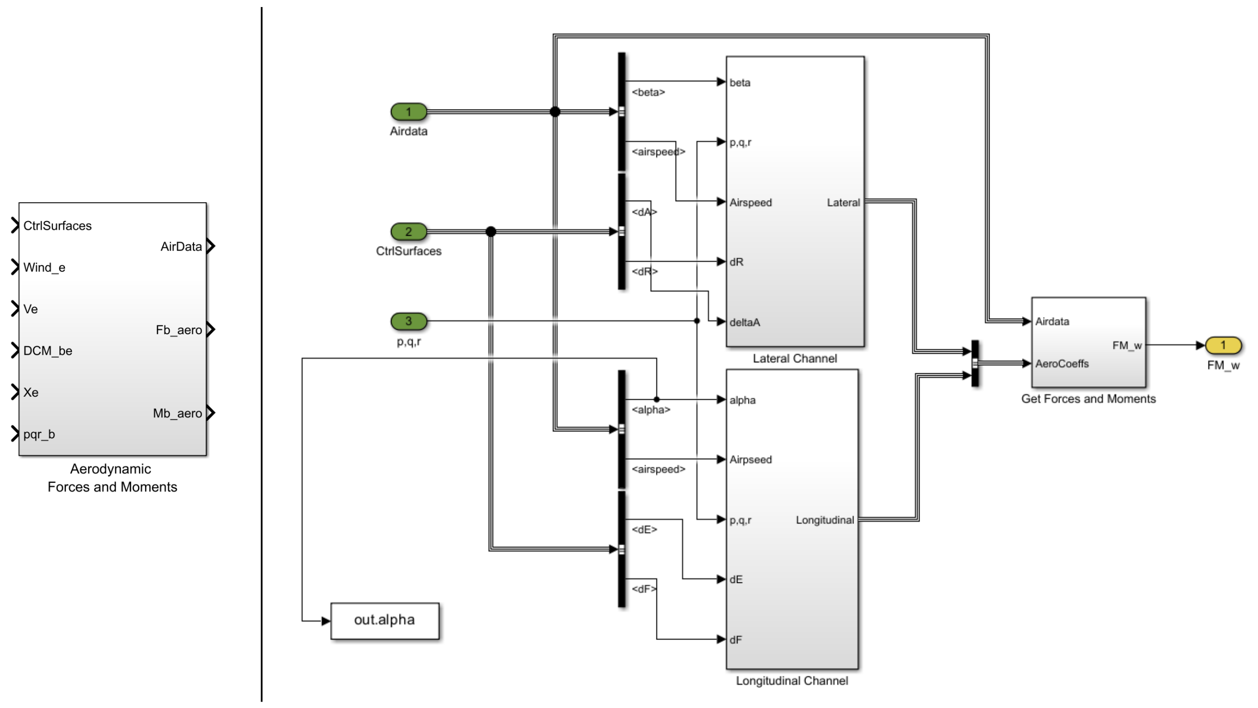


Figure 25: Aerodynamic forces and moments Simulink block

Here, it is separated the longitudinal from the lateral channel in the calculation of forces and moments, corresponding to equations (40), (41) and (44).

6.1.4 Gravity

The next block will be the *Gravity* block, responsible for the gravity model used during the simulation. In this project, it will be used a constant gravity model, as the cruise flight altitude is 100 meters, so the gravity acceleration will not decrease in a significant manner, as in equation (4):

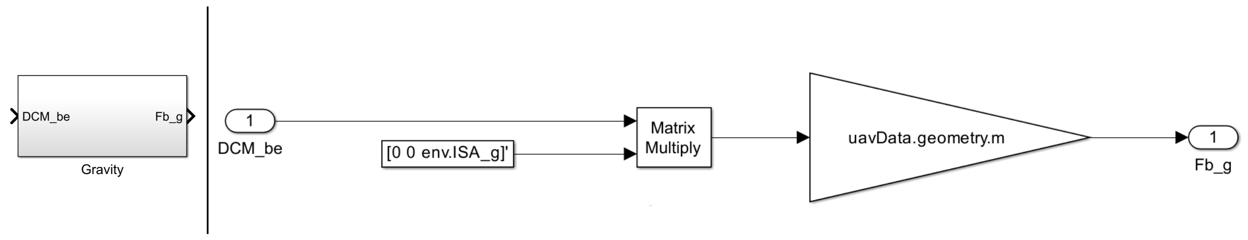
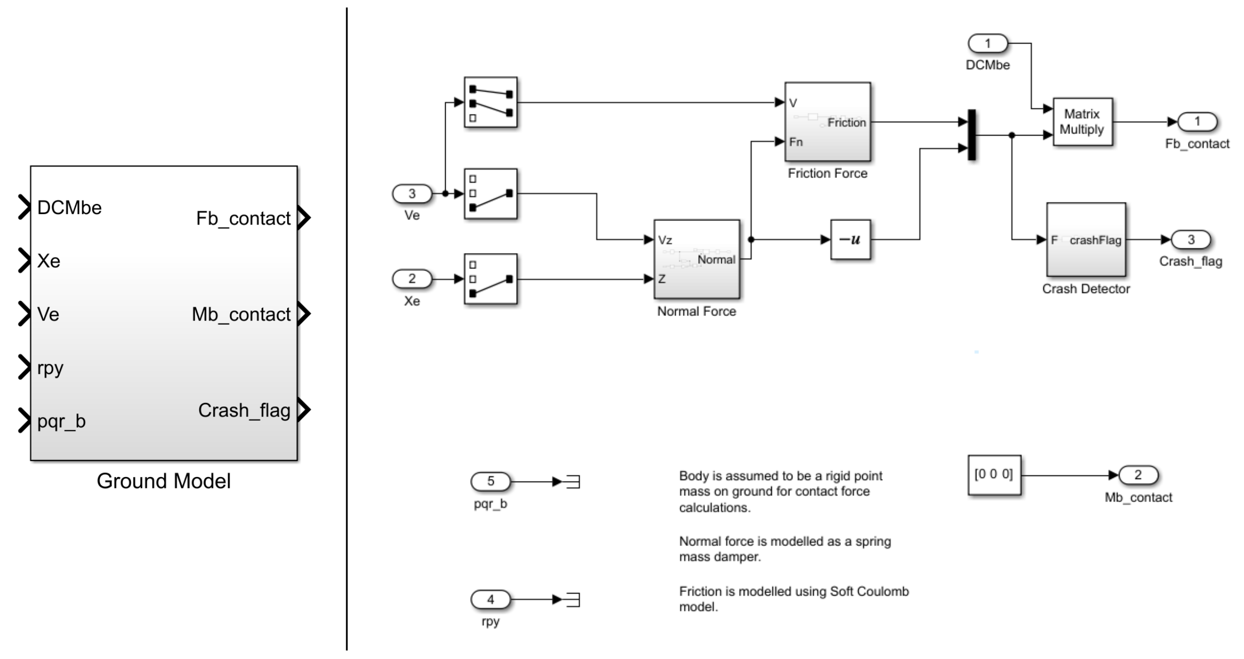


Figure 26: Gravity Simulink block

Here it is needed the DCM_{be} as an inputs, as the gravity vector is expressed in flat-Earth reference framce, and it is needed to be expressed in H200's body frame.

6.1.5 Ground Model

Last, but not least, it will be reviewed the *Ground Model* block. It will be assumed that the H200 is a rigid body, concentrated in a single point to simplify the contact force calculations:



Body is assumed to be a rigid point mass on ground for contact force calculations.

Normal force is modelled as a spring mass damper.

Friction is modelled using Soft Coulomb model.

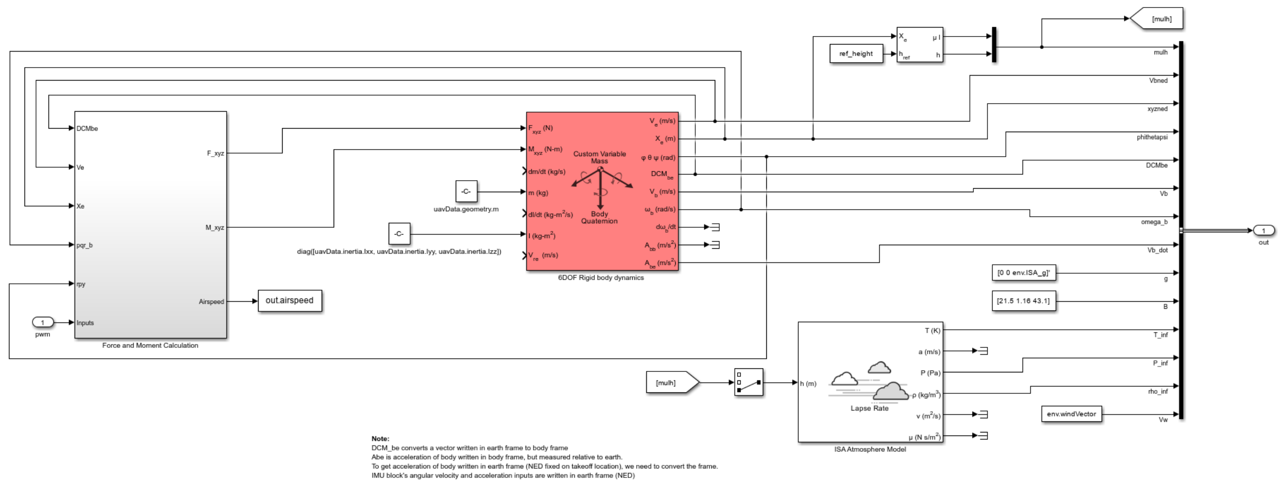
Figure 27: Ground model Simulink block

The normal force with the ground will be calculated using a spring-damper system, the friction with the ground using a soft Coulomb model, and this block will raise a flag in case the H200 crashes with the ground.

6.1.6 Complete H200 Simulink Model

With all of this blocks inside the *Force and Moment Calculation* block, it is possible to calculate the H200's forces and moments around its center of gravity. Then, it will be fed up as inputs to the 6-DoF Simulink block, showed in figure (21). Then, it will be logged all the outputs for better flight representation.

The complete H200 Simulink dynamic model will be:



Note:
DCM_be converts a vector written in earth frame to body frame
Abe is acceleration of body written in body frame, but measured relative to earth.
To get acceleration of body written in earth frame (NED fixed on takeoff location), we need to convert the frame.
IMU block's angular velocity and acceleration inputs are written in earth frame (NED)

Figure 28: Complete H200 Simulink non-linear model

6.2 Validation of the Nominal Controllers

With the H200's non-linear Simulink model has been built, the next step is the controllers validation from section (5. This subsection will be divided in two main areas of work:

- Validation of the nominal controllers in flight cruise conditions: The simulations will be run using the three controllers as designed and tuned in section (5), in order to validate their performance in cruise flight.
- Test the controllers in non-flight cruise conditions: Tests will be performed with the three controllers, but introducing some external perturbations which will modify the flight conditions of the H200. The external perturbations will consist in a wide variety of conditions (structural, control inputs, etc)

Before starting with the description of the experiments, the simulation has to be set up. First of all, the initial conditions for the H200 simulation will be defined using the next table:

Simulation Initial Conditions

Initial condition	Symbol	Value	Units
Initial geodetic position	$[lat,\, lon,\, h]$	$[39.4974, -0.6268, 100]$	$[deg, deg, m]$
Initial attitude	$[\phi,\, \theta,\, \psi]$	$[0, \theta_{trim}, 0]$	$[deg, deg, deg]$
Initial airspeed	V	V_{Test}	m/s
Initial angular rates	$[p,\, q,\, r]$	$[0, 0, 0]$	$[rad/s, rad/s, rad/s]$
Initial mass	m	m_{Test}	kg

Table 12: Non-linear simulation initial conditions

The H200 will initially be in stable flight, meaning that it will navigate in trim conditions, in a given airspeed and altitude. As it may be seen in table (12), there are some parameters which are not given numerically, as it will depend on the test to be performed:

- θ_{trim}: It's the trim pitch angle value. As commented, the initial pitch angle for the H200 will be the one that results in a stable flight.
- V_{Test}: It's the airspeed at which the H200 will fly over. For most of the test, it will be the one given in table (8), but for two specific tests, it will be changed to check the controller performance against a change in flight conditions.
- m_{Test}: It's the H200 mass. Once again, for most of the tests it will be the one specified in section (3), when reviewed the H200 geometric parameters. However, for two tests it will be changed so the controllers performance may be assessed if there exists a change in mass during the flight.

For the test results, they will be divided in two parts. On the one hand, they will be given by the simulation output plots, which give graphical information about the controller performance during the flight. They will consist of:

- Controller plot: It will be used to show the main controller information: reference signal and response signal, input control action signal and rate of control action (only for the MPC and the MADRPC).

- Aircraft geodetic position: It will be shown the inertial position of the H200, given in latitude, longitude and altitude.
- Flat-Earth position: It will be shown the inertial position of the H200, in a NED frame, taking the origin of coordinates the initial position given in table (12).
- 3D navigation plot: It will be shown the inertial position of the H200, given in latitude, longitude and altitude in a 3D environment, so it can be checked the complete flight trajectory.
- Attitude plot: It will be shown the attitude position of the H200, given in the Euler-Angles representation (roll, pitch and yaw).
- Body velocities from the local frame: It will be shown the body velocities the H200 experiences, represented in the local frame (flat-Earth representation).
- Body velocities from the body frame: It will be shown the body velocities the H200 experiences, represented in the body frame.
- Body rotations: Showing the body angular velocities the H200 experiences.
- H200 inputs: Showing the H200 physical inputs: elevator deflection (δ_E), aileron deflection (δ_A), rudder deflection (δ_R), throttle level (δ_P) and flaps deflection (δ_F).

And on the other hand, it will be used some key performance indicators, in order to give an objective and numerical conclusion about the controllers performance. In this project, it will be used:

- ITAE (Integral Time Absolute Error). This indicator is used generally to test errors which last in time, as greater times make the indicator increase. The formula for the ITAE is:

$$ITAE = \int_0^\infty t\,|e(t)|dt \tag{80}$$

- ISE (Integral Square Error). This indicator has been used for testing the performance of the controllers during the design phase using the linear plant. It is used against high errors during the simulation, as the square operation contribute to higher values. The formula for the ISE is:

$$ISE = \int_0^\infty e^2(t)dt \tag{81}$$

- IAE (Integral Absolute Error). This indicator will be used supporting the ISE, in order to greatly eliminate small errors, as squaring them lowers the ISE. The formula for the IAE is:

$$IAE = \int_0^\infty |e(t)|dt \tag{82}$$

- MSE (Mean Square Error). Gives the mean value of the ISE along all the simulation, giving more importance to greater errors appearing during the test. It is one of the main used indicators when evaluating performance, as it gives a more conservative measure of it. The formula for the MSE is:

$$MSE = \frac{1}{t}\int_0^{\infty} e^2(t)dt \tag{83}$$

For the controller implementation in Simulink, the same structure will be followed for all the controllers:

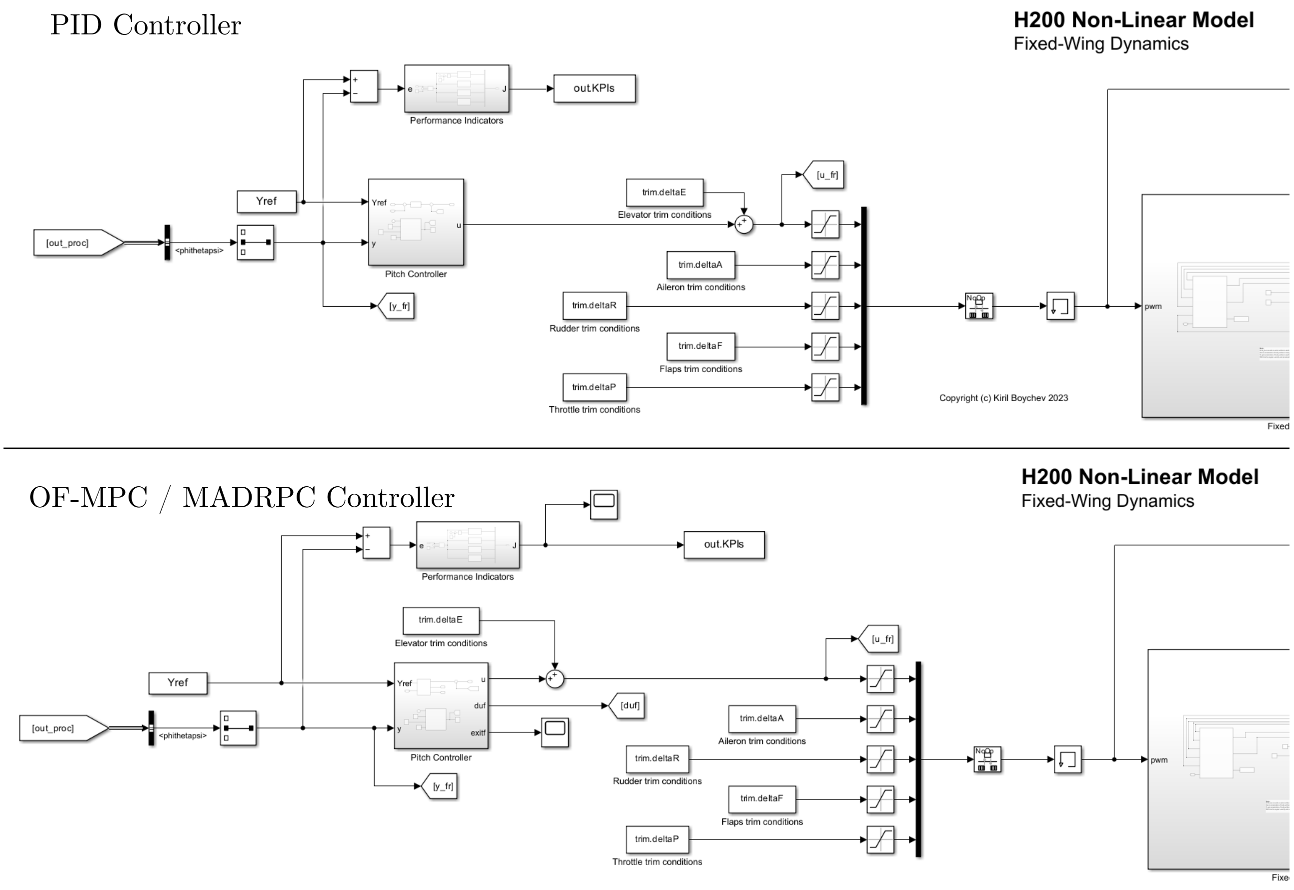


Figure 29: PID, OF-MPC and MADRPC controller structure inside the H200 non-linear model

As it can be seen in figure (29), the top part corresponds with the PID controller structure, while the down one correspond to the OF-MPC and MADRPC controller structure. The inputs will be the pitch angle reference signal and the pitch angle response in any instant during the simulation, so the tracking error will be calculated inside and the control action may be computed. The only difference is that the outputs for the PID controllers will be the response signal (along to the reference trajectory) and the control action signal, whereas the OF-MPC and the MADRPC will output additionally the rate of control action signal.

On the other hand, the rest of the H200 inputs will stay in their respective trim conditions value, so it is assured a stable flight while the pitch angle is modified. In other words, it will be assumed that somehow the control inputs will remain the same, with their values equal to their correspondent trim values.

The last thing to review before the testing is the reference signal that will be used. The structure for the reference pitch angle will be:

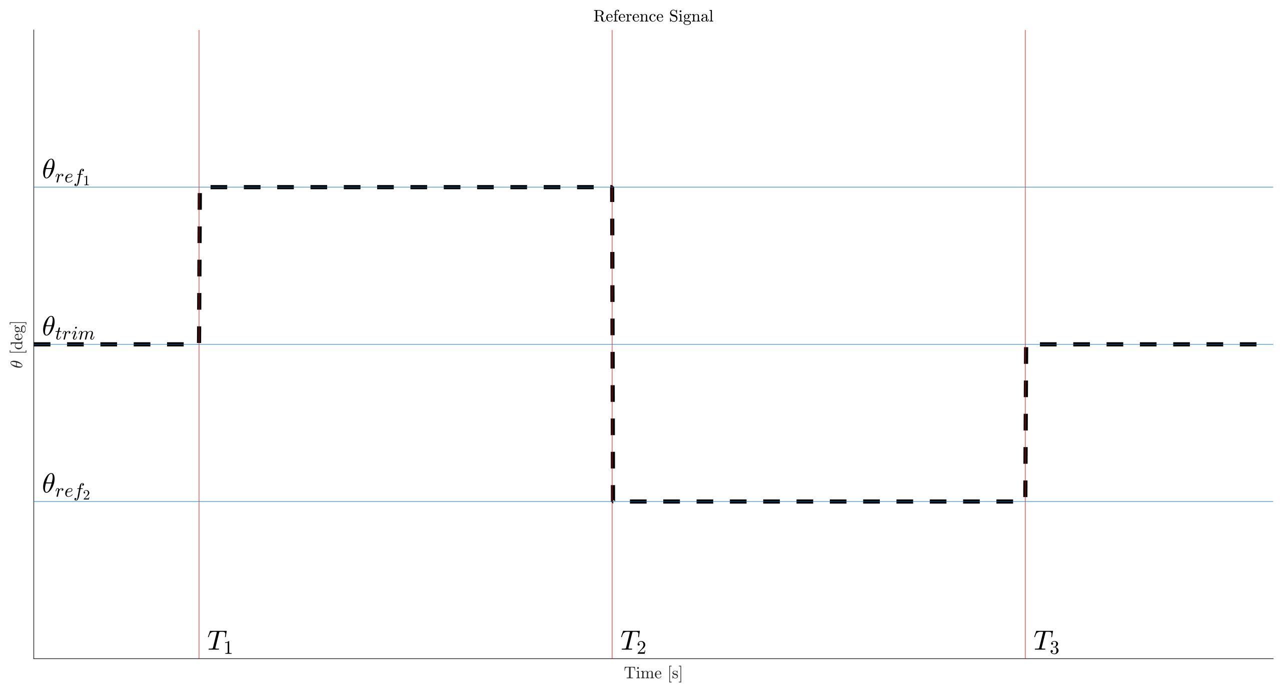


Figure 30: Reference signal for controller validation simulations

As it may be seen in figure (30), it is divided in four sections:

1. From start of the simulation until T_1: The H200 will start in stable flight conditions, before starting the maneuvers for the controllers validation. Meaning, that the initial pitch angle for the H200 will be the one for trim conditions in the specific test ($\theta_{trim}s$).
2. Between T_1 and T_2: The controller will be demanded to go to a pitch angle above the trim pitch angle ($\theta_{ref_1} > \theta_{trim}$).
3. Between T_2 and T_3: The controller will be demanded to go below the trimming pitch angle ($\theta_{ref_2} < \theta_{trim}$).
4. From T_3 until the end of the simulation: The controller will be demanded to return to trim conditions until the simulation ends.

The structure for the reference signal has been chosen in order to, first of all, visualize the transition periods when the reference pitch angle changes, making the controller go above and below its equilibrium point.

For the numerical values in figure (30), which are T_1, T_2, T_3, θ_{trim}, θ_{ref_1} and θ_{ref_2}:

- The temporal parameters (T_1, T_2 and T_3) will be the same in all test to be performed, as they will ensure a proper separation between the different pitch angle reference values to follow. And also consistency between all the tests will be assured.

- The pitch angles to be followed (θ_{trim}, θ_1 and θ_2) will be chosen depending on the test to be run, as the tests will or will not follow cruise flight conditions. As a consequence, the equilibrium pitch angle (θ_{trim}) will change depending on them. For θ_{ref_1} and θ_{ref_2}, they will be chosen following the conditions commented previously:

$$\theta_{ref_1} > \theta_{trim} > \theta_{ref_2} \tag{84}$$

With all of this reviewed, the controllers validation tests may be started.

6.2.1 Validation of Controllers in Cruise Flight Conditions

To begin with, the first test to be performed will be the controllers validation in cruise flight conditions using the complete H200 non-linear model. The objective is to check that the controllers tuning phase using the linear process, done in section (5), meet the specifications with the complete non-linear model (implemented in Simulink). The numerical values for the reference signal parameters in figure (30) for this test will be:

Symbol	Value	Units
T_1	2	s
T_2	7	s
T_3	12	s
θ_{trim}	2.5905	deg
θ_{ref_1}	3	deg
θ_{ref_2}	2	deg

Table 13: Reference signal parameters for controller validation

Now, having reviewed all of this, it is now the moment to validate the controllers with the H200 non-linear model. The results have been:

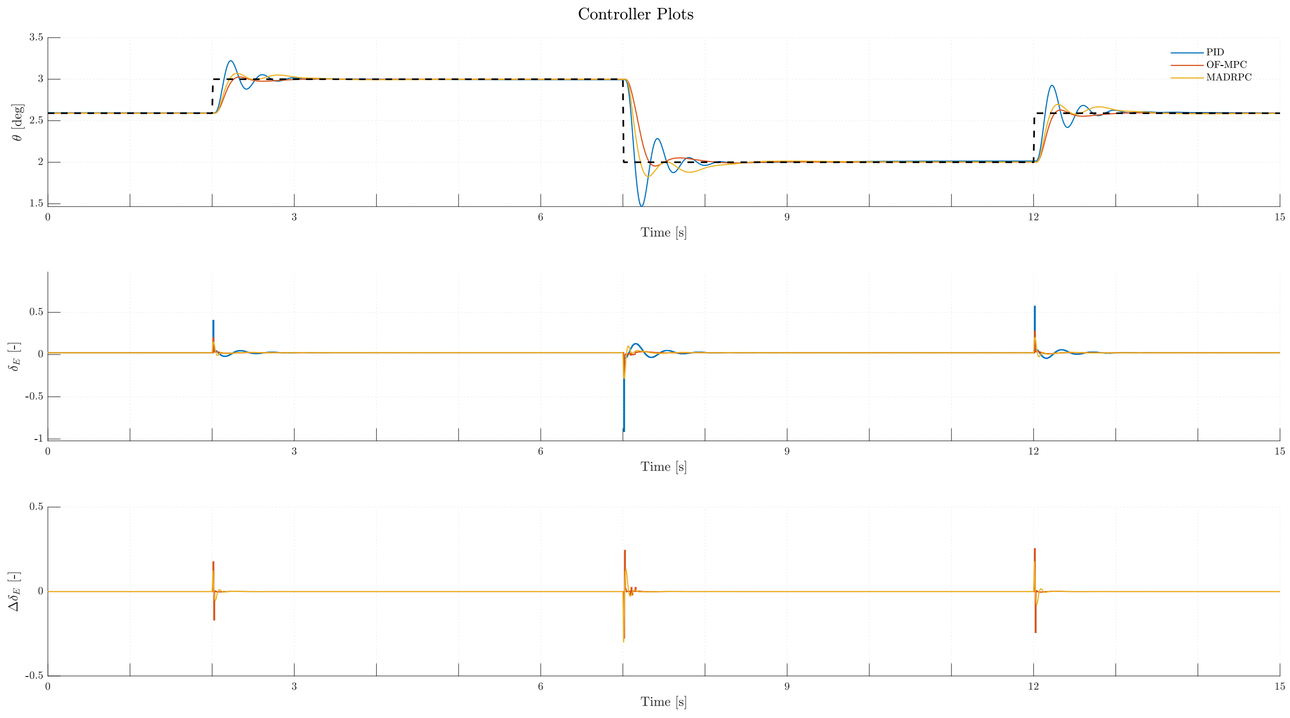


Figure 31: Controller plots for test with cruise conditions

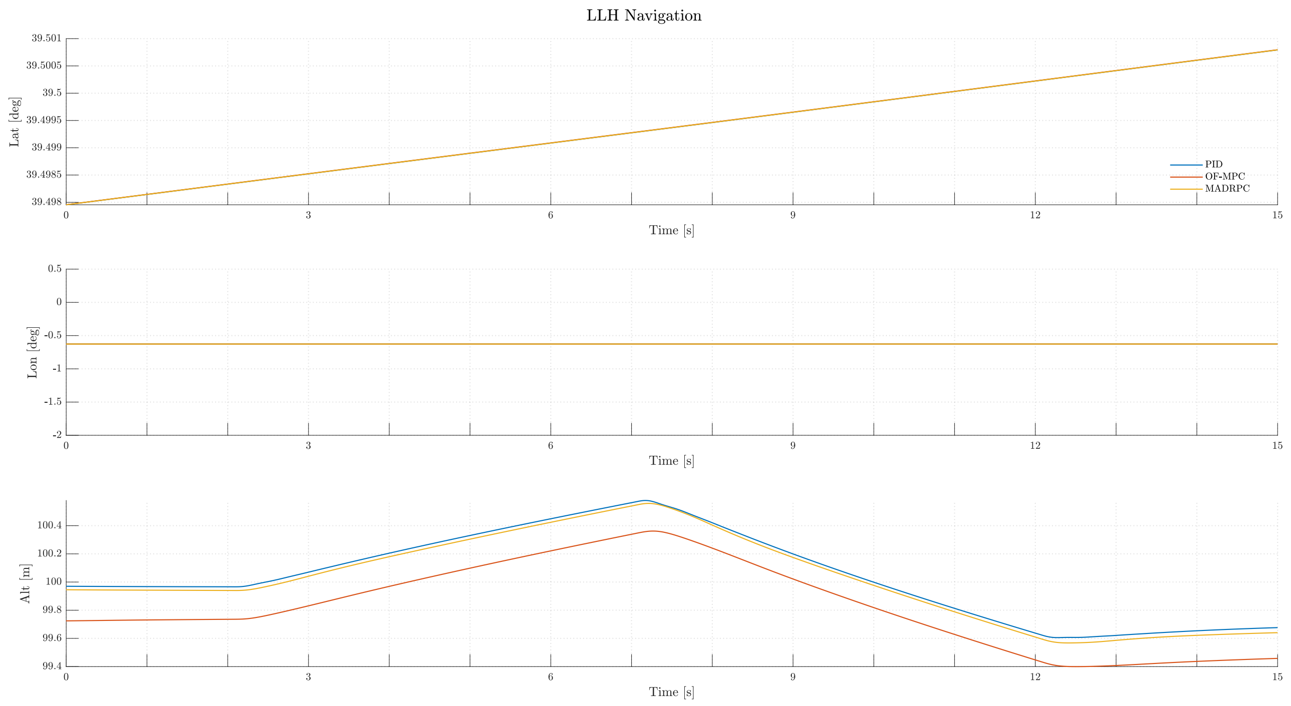


Figure 32: H200 LLH navigation during test with cruise conditions

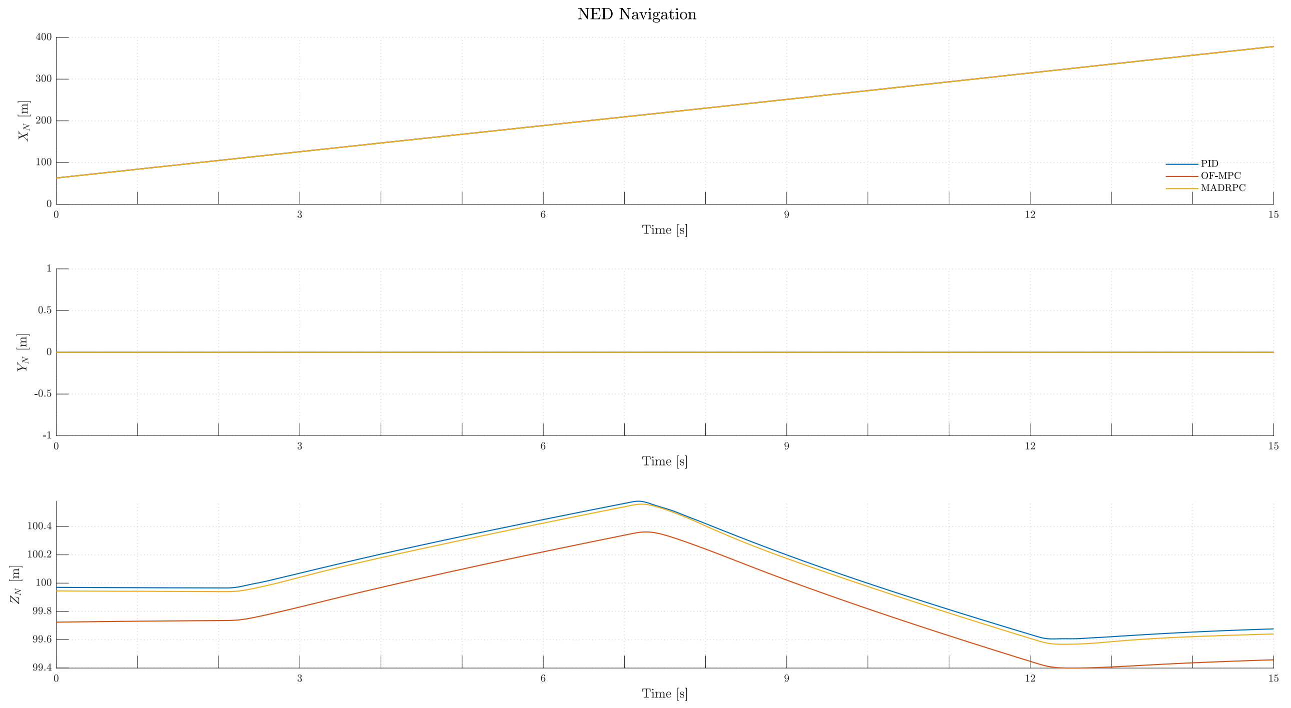


Figure 33: H200 NED navigation during test with cruise conditions

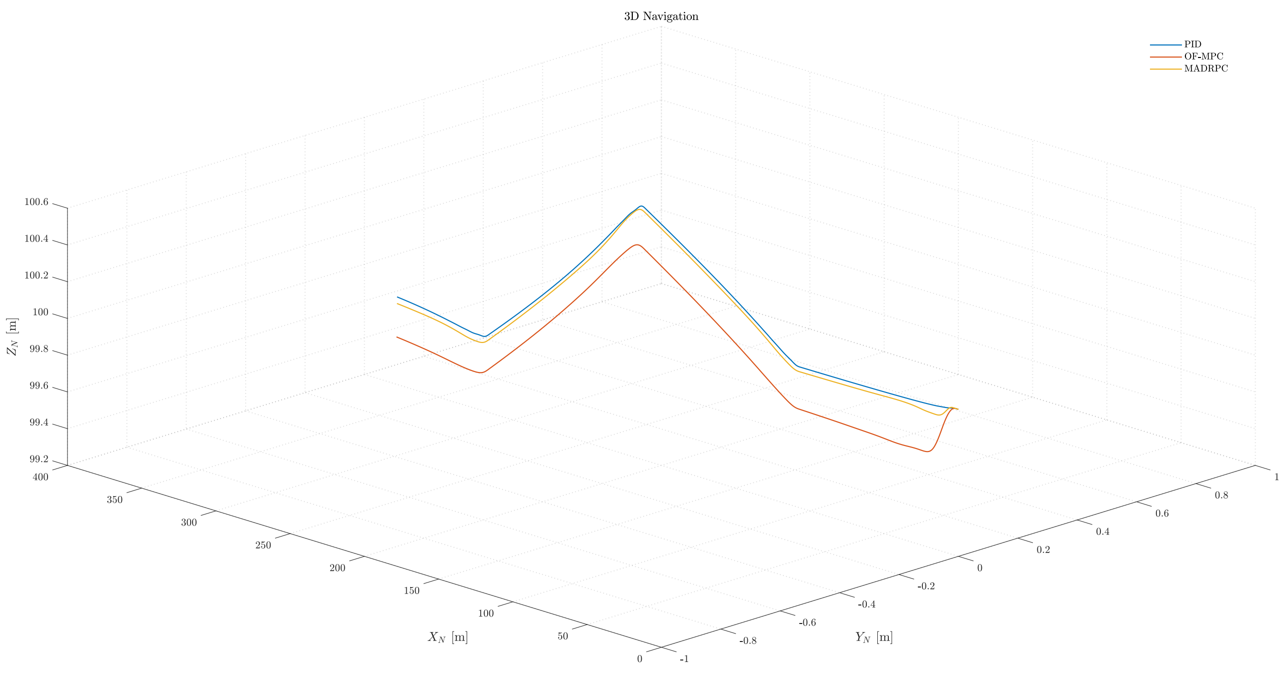


Figure 34: H200 3D navigation during test with cruise conditions

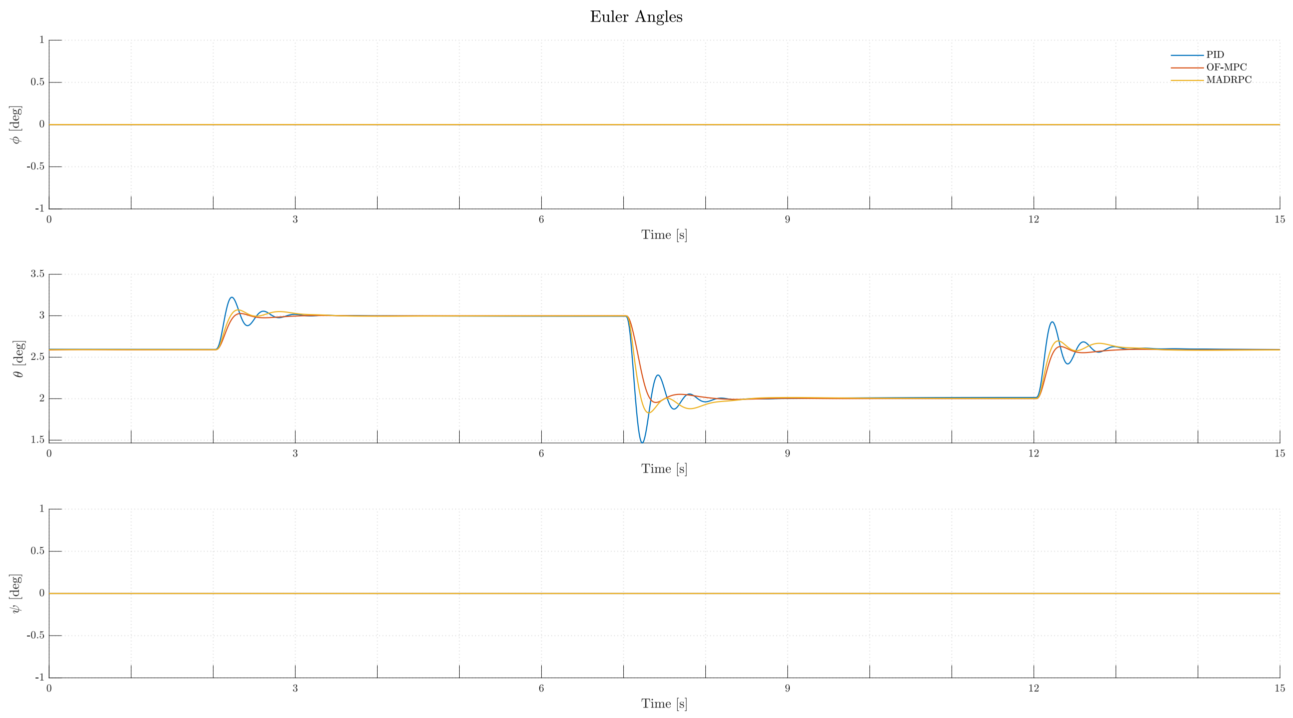


Figure 35: H200 Euler angles during test with cruise conditions

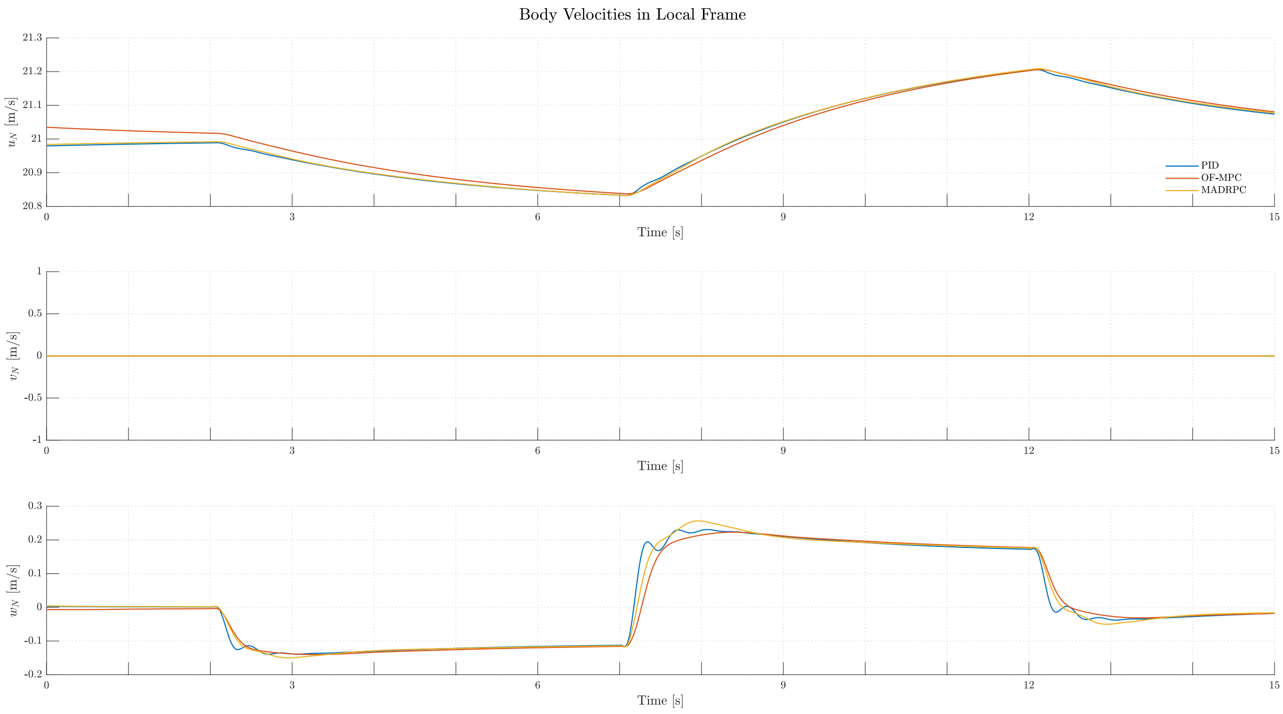


Figure 36: H200 body velocities expressed in local frame during test with cruise conditions

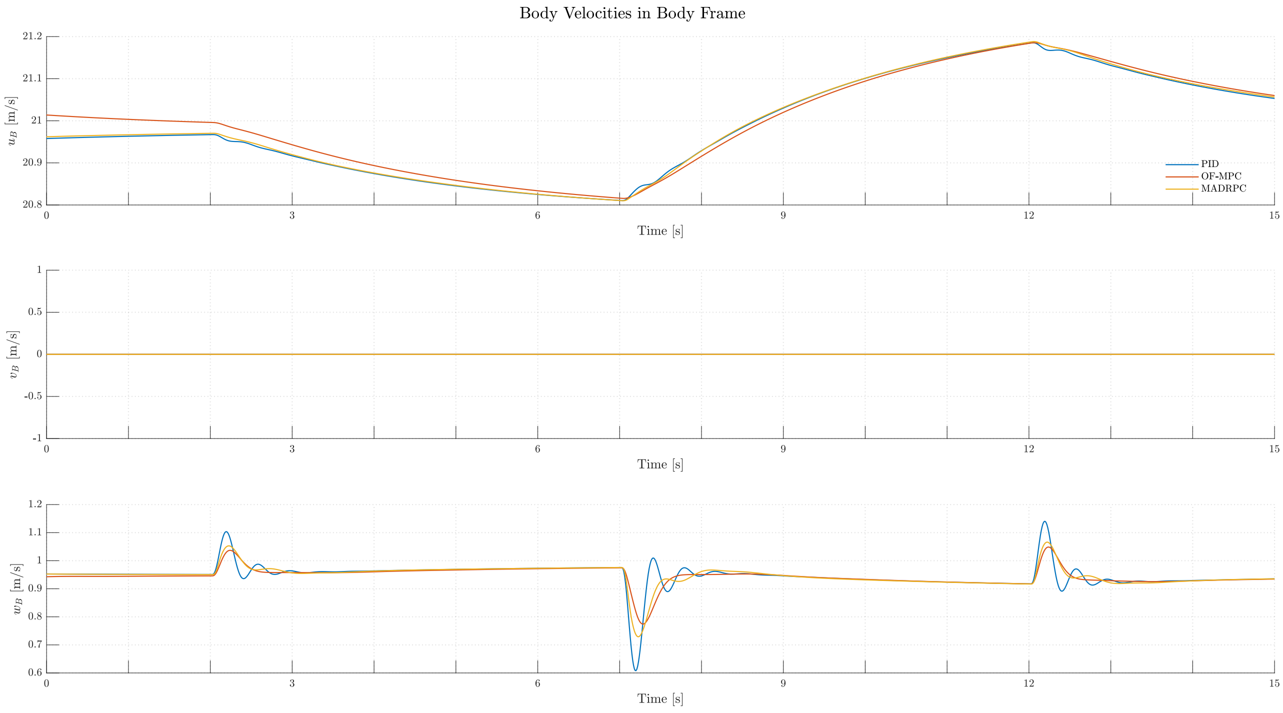


Figure 37: H200 body velocities expressed in body frame during test with cruise conditions

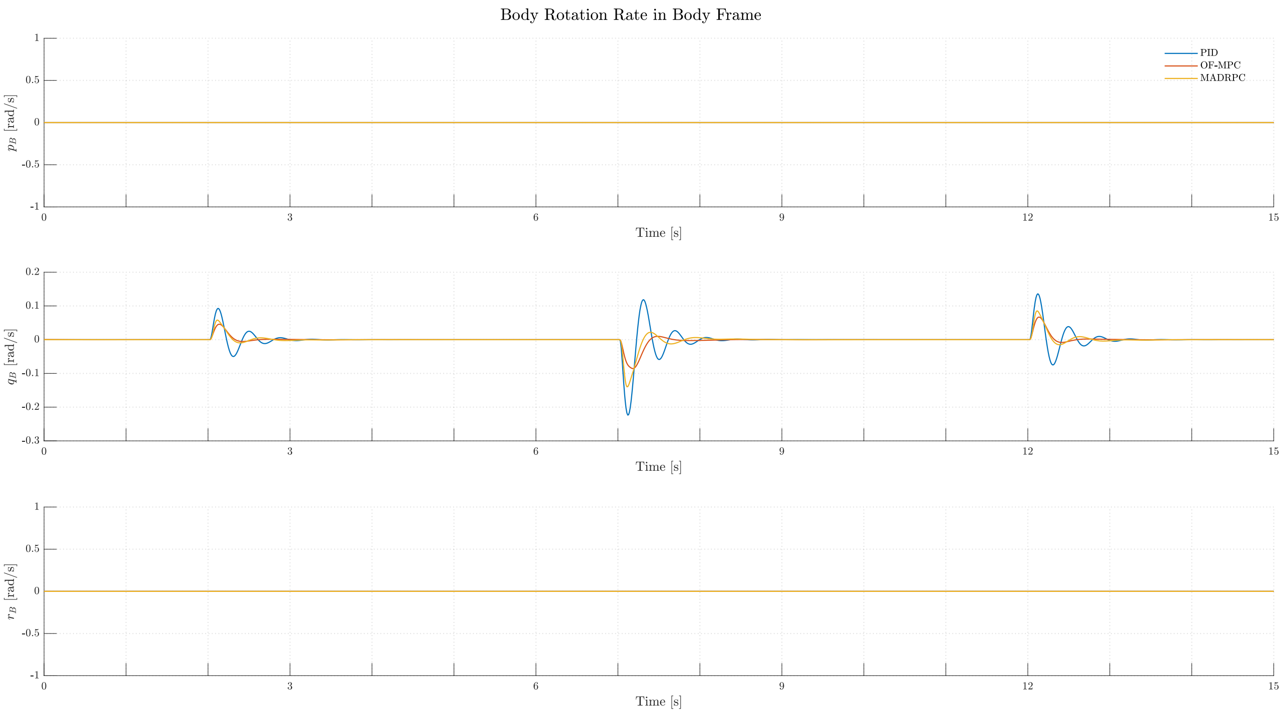


Figure 38: H200 body rotations during test with cruise conditions

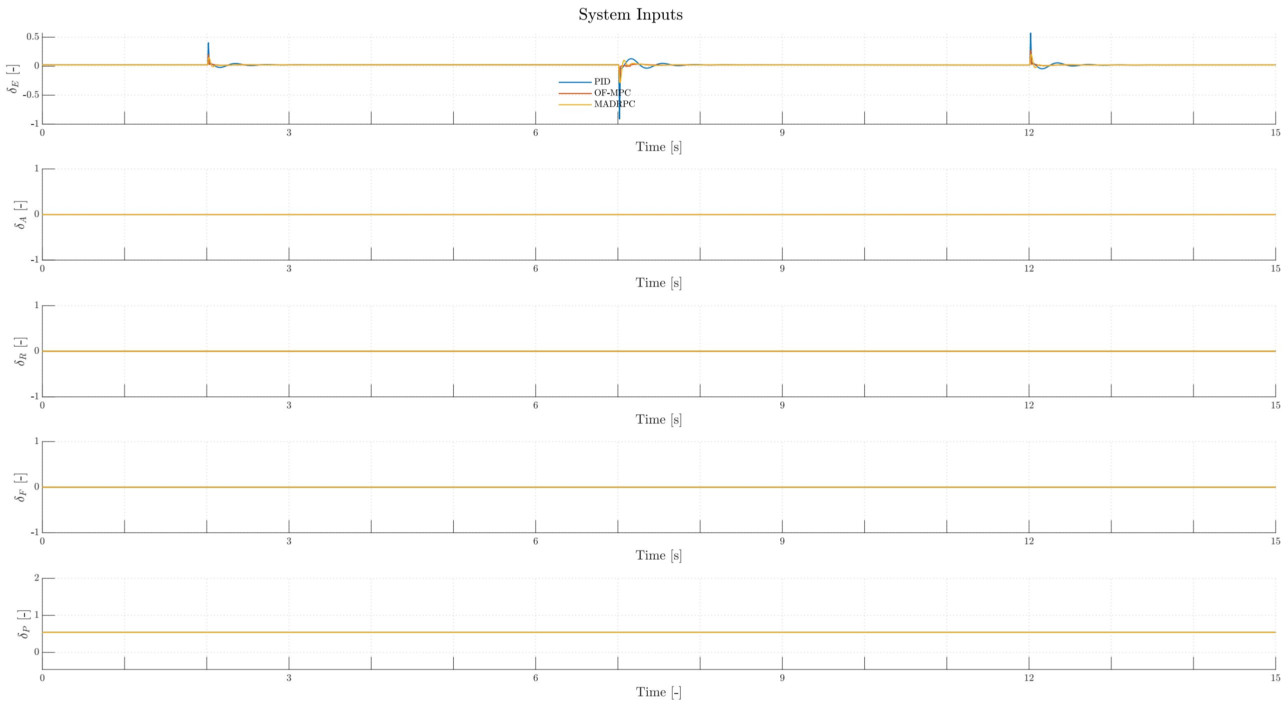


Figure 39: H200 system inputs during test with cruise conditions

And the performance indicators have been:

	ITAE	ISE	IAE	MSE
PID	7.2998	0.0054968	0.92144	0.00036645
OF-MPC	5.2179	0.0060641	0.68505	0.00040427
MADRPC	6.3005	0.0050746	0.79551	0.00033831

Table 14: Controller performance indicators for test with cruise conditions

From figures (31) to (39), the H200 starts in stable flight conditions. Then, from T_1, the pitch angle is demanded to go above and below its equilibrium point. As appreciated from the plots, each of the controllers is capable of governing the H200's pitch angle, successfully performing a stable flight during all the validation test. From table (14), the best performance is given by the OF-MPC followed by the MADRPC and then the PID, although their results are in the same order of magnitude. Meaning that, in nominal conditions, the three of them are capable of a successful stable flight.

It has to be remarked the fact that the MADRPC is capable of controlling the angle of attack without any prior information of the process (apart from two natural plant parameters). The PID and the OF-MPC needs the pitch angle model information, either for design purposes or predictive capabilities, respectively. The MADRPC, having inside a modified assumed model for the pitch angle, and a disturbance rejection to compensate the error between the real model and the modified one, it is capable of stable flight only with two natural parameters from the plant.

From this test, using cruise flight conditions, it may be seen that the PID controller is the one demanding the most from the elevator deflection, as it is "locked" in the present instant, compensating for the instantaneous value of the tracking error signal. The OF-MPC and the MADRPC, being both of them predictive controllers, the control actions are more relaxed, as they find the most optimal solution given a series of constraints that ensure a smooth response and control action signal.

With the successful results obtained in this test, the controllers are validated using cruise flight conditions, making them suitable for governing the H200 in a normal flight mission. Now, taking these results as a starting point, the controllers will be faced against a series of tests with non-cruise conditions, in order to check their performance outside their working point.

6.2.2 Validation in Non-Cruise Flight Conditions

As commented, with the three control algorithms already validated, it is time for testing them in non-cruise conditions. The main objective of these tests is to validate the MADRPC controller against the traditional control schemes: the PID and the OF-MPC control, as the pitch angle transfer function, in equation (69), will be modified when the flight conditions are changed. Both the PID and the OF-MPC rely on having an accurate description of the process, for tuning the controllers (PID) and making predictions to optimize the control actions (OF-MPC). The MADRPC does not need a precise model in advance, as it has an internal assumed model, which needs two natural parameters from the process (and not a complete description of it) and it uses a disturbance rejection controller to compensate the difference between models.

With this objective in mind, a series of tests will be defined, depending on the perturbations being introduced inside the system:

- Flight condition perturbations: They are going to focus on changing some of the flight conditions the H200 will fly. They are relevant as the aircraft generally don't stay in cruise flight conditions during all its mission. The tests will focus on changing mainly the flight airspeed, as the altitude (more in low-altitude flights) do not significantly affect the results:
  - **Test A.1**: Cruise altitude, with less airspeed.
  - **Test A.2**: Cruise altitude, with greater airspeed.
- Structural perturbations: These tests are going to focus on modifying the aircraft inner parameters, which play a big role inside the H200 non-linear model. As commented before, this parameters are the result from either experimental data or semi-empirical methods. These tests will focus in changing the aerodynamic model, the propeller model and the aircraft flight mass, within the operative maximum and minimum ranges:
  - **Test B.1**: Underestimation of the aerodynamic coefficients. Inside the model used for aerodynamic forces and moments, it will be scaled down the coefficients directly related to the pitch angle and the elevator deflection.
  - **Test B.2**: Overestimation of the aerodynamic coefficients. It will be like **Test B.1**, but in this case, the coefficients will be scaled up. Through these tests, it will be checked the controllers performance inside the aerodynamic model acceptable range.
  - **Test B.3**: Underestimation of the propeller coefficients. The propeller model developed for the H200 has been calculated using experimental data, and numerical fitting to them. This test will scale down the coefficients for the thrust coefficient inside the model, as the thrust against the propeller angular velocity is reliable with manufacturer data.
  - **Test B.4**: Overestimation of the propeller coefficients. It will be like **Test B.3**, but in this case, the coefficients will be scaled up. Once again, through these tests, it will be checked the controllers performance against the propeller model operative range.
  - **Test B.5**: Flying with a higher mass than the cruise conditions. The mass specified in section 3 it's the one assumed for this specific configuration of the H200. This test will check the performance against a higher payload than designed.
  - **Test B.6**: Flying with a less mass than the cruise conditions. The H200 is capable of dropping its payload at any moment during its flight mission, so the controllers must be able to perform while on a mass below from the designed mass in nominal condition.
- Input perturbations: These tests are going to focus on altering the control input signal during the simulations. Either via introducing wind gusts during the flight mission or perturbations in the elevator deflection signal:
  - **Test C.1**: Wind gust perturbation sustained over time. The first test will introduce a wind gust that will be maintained during all the flight mission at some specific time.
  - **Test C.2**: Instantaneous wind gust perturbation. This test will introduce a wind gust over a specific time and then remove it, in order to check the instantaneous perturbation rejection of the controllers.

- **Test C.3**: Elevator deflection perturbation sustained over time. This test will introduce a elevator deflection perturbation sustained during al the flight mission at some specific time.
- **Test C.4**: Instantaneous elevator deflection perturbation. This test will introduce an elevator deflection error over a specific time and then remove. Once again, in order to check the instantaneous perturbation rejection of the controllers.

The reference pitch angle to be used during the simulation will be the one used for the validation of controllers in cruise flight, given in figure (30), although the pitch angles to be followed during the simulation will change, depending on the test running. But the reference signal structure will be the same along all tests.

6.2.2.1 Flight Conditions Perturbations

This family of tests will focus on changing the flight conditions the H200 will fly over the simulation. The objective of these tests is to check the performance of the controllers around the operative limits of its flight envelope. They will check that, as the H200 will not fly always in flight cruise conditions, the controllers operation is not worsened by changing the operating point of the aircraft.

As it has already been specified, the change in flight conditions will come from a modification in the cruise airspeed. The reason behind this is that a change in the flight altitude will not affect significantly the flight performance, specially at low altitude. So the tests will only focus on changing at what speed the aircraft will be traveling.

As a consequence, changing the flight airspeed will change the equilibrium point, which are, at the same time, the trimming values for a cruise flight in the modified conditions. This will affect the initial conditions for the simulations, as the airspeed and the trim pitch angle are the initial conditions for the H200, and the numerical values of the reference signal, figure (30), will be modified for accounting to this change, although the structure will be the same.

6.2.2.1.1 Test A.1: Cruise Altitude, Lower Cruise Airspeed

The first test will focus on decreasing the cruise airspeed during the H200 flight mission. The modified flight conditions for **Test A.1** will be:

Flight Conditions	
Altitude	100 m
Velocity	15 m/s

Table 15: Flight conditions for **Test A.1**

As previously said, a change in the flight conditions will result in a modification for the trimming values. For this, the value for the trim pitch angle will be:

$$\theta_{trim} = 8.3114\,\text{deg} \tag{85}$$

And the numerical values for the reference signal for **Test A.1** will be:

Symbol	Value	Units
T_1	2	s
T_2	7	s
T_3	12	s
θ_{trim}	8.3114	deg
θ_{ref_1}	10	deg
θ_{ref_2}	7	deg

Table 16: Reference signal parameters for **Test A.1**

The result plots of **Test A.1** can be seen from figure (40) to (48).

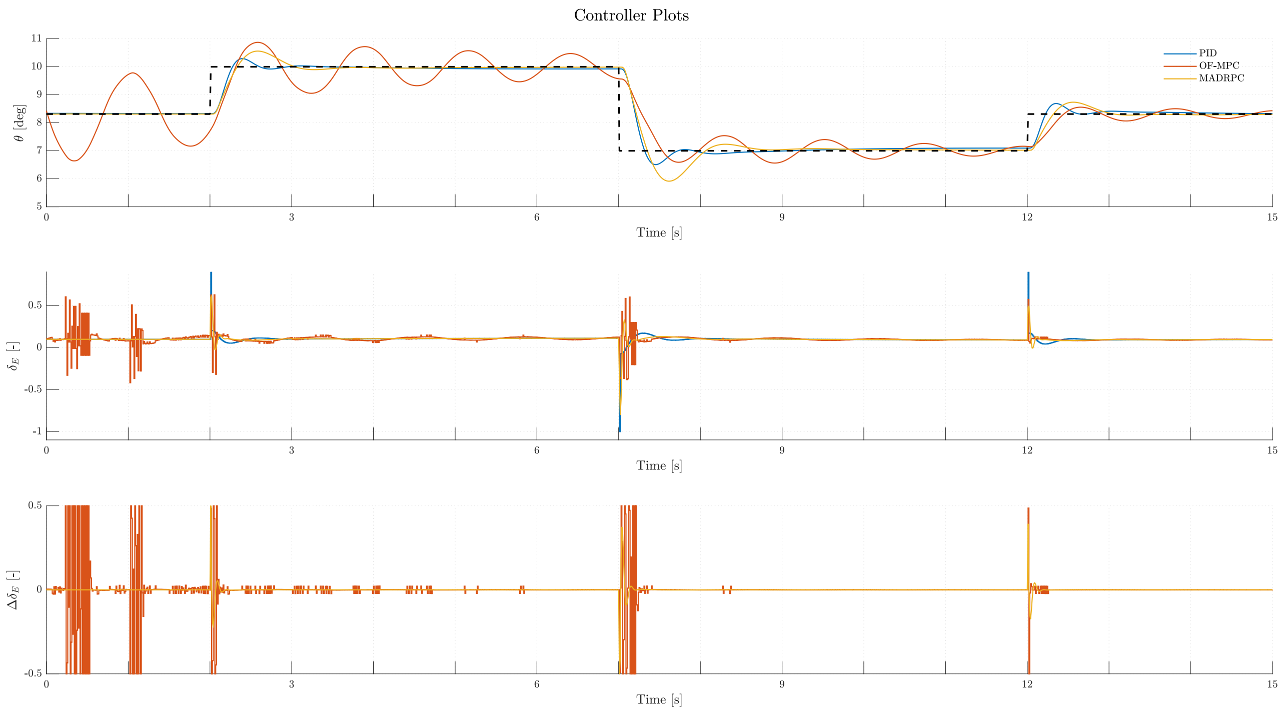


Figure 40: Controllers plots during **Test A.1**

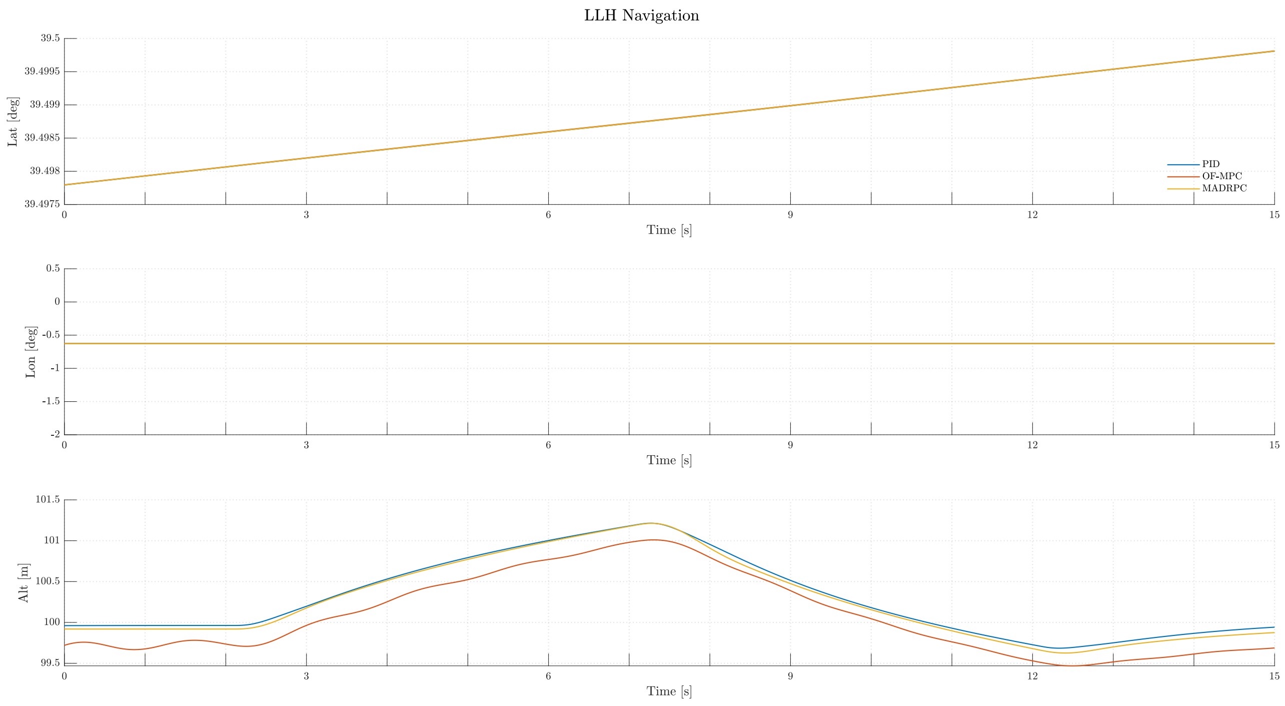


Figure 41: H200 LLH navigation during **Test A.1**

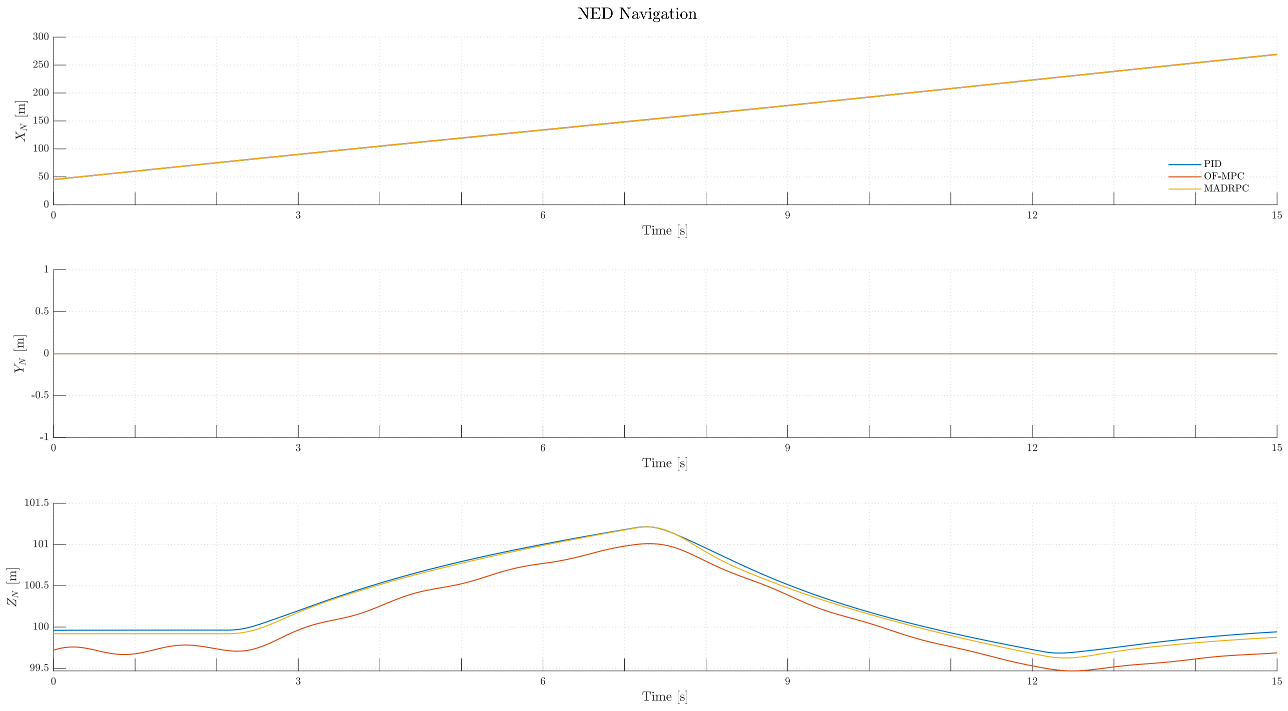


Figure 42: H200 NED navigation during **Test A.1**

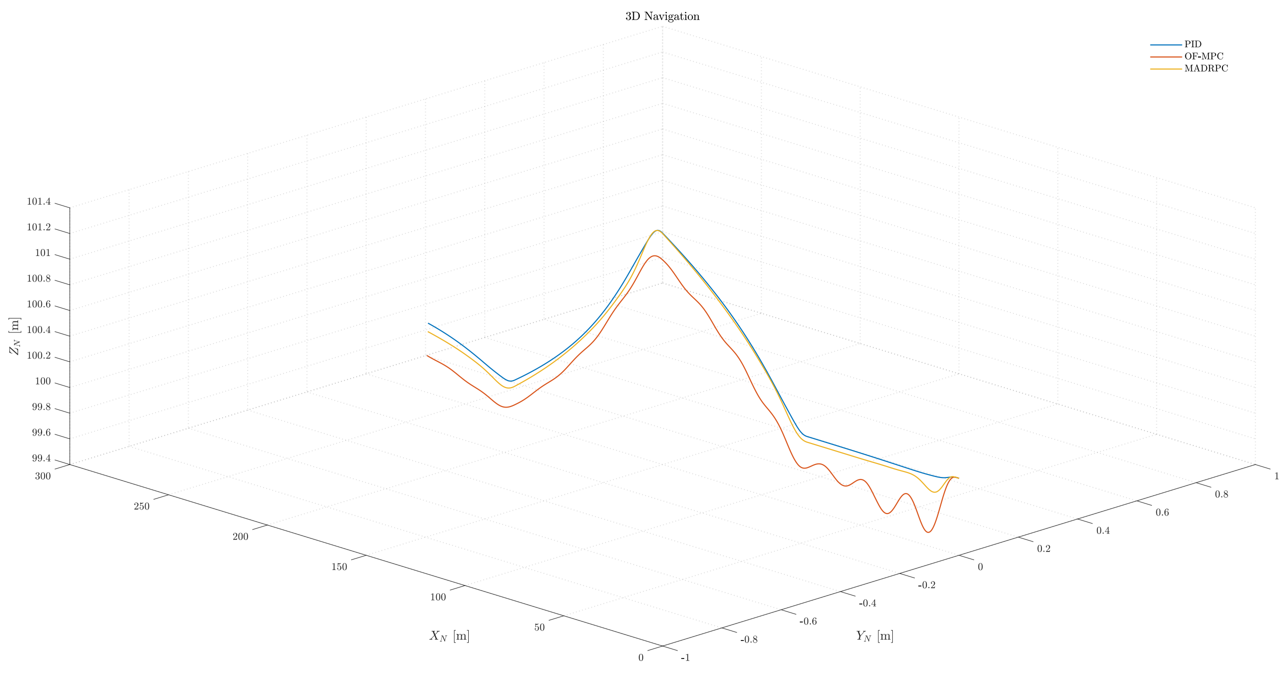


Figure 43: H200 3D navigation during **Test A.1**

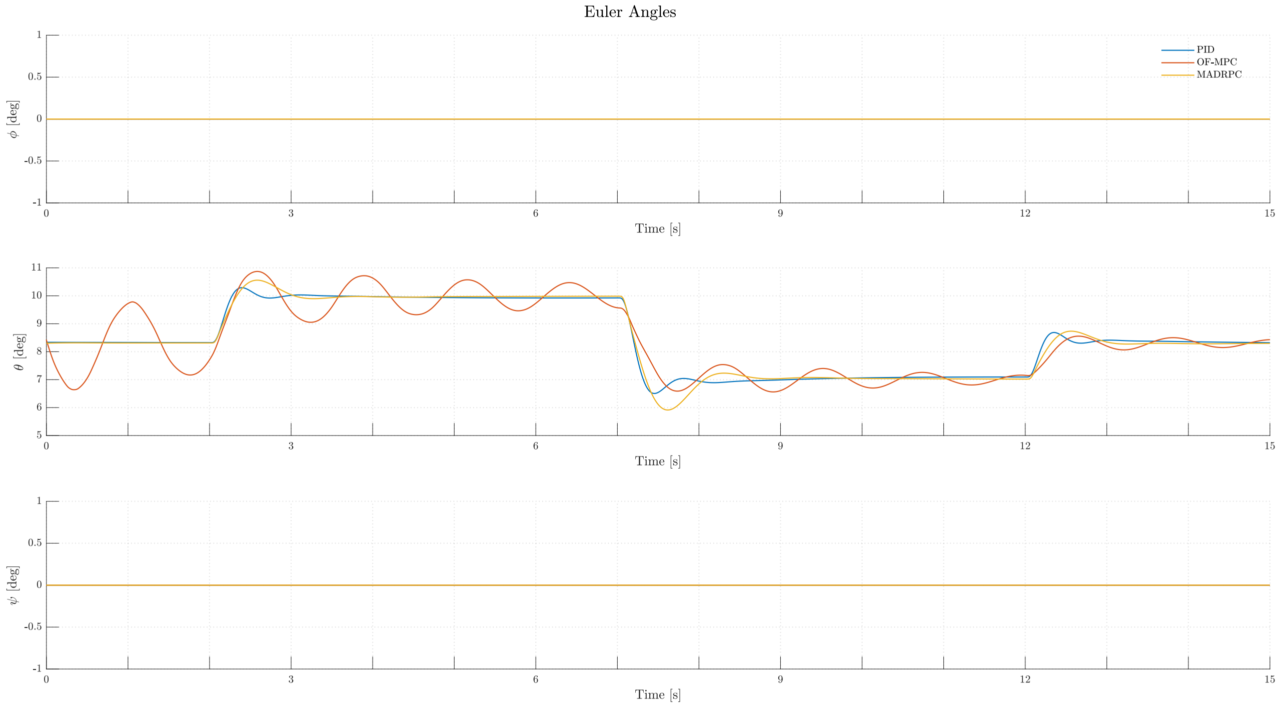


Figure 44: H200 Euler angles during **Test A.1**

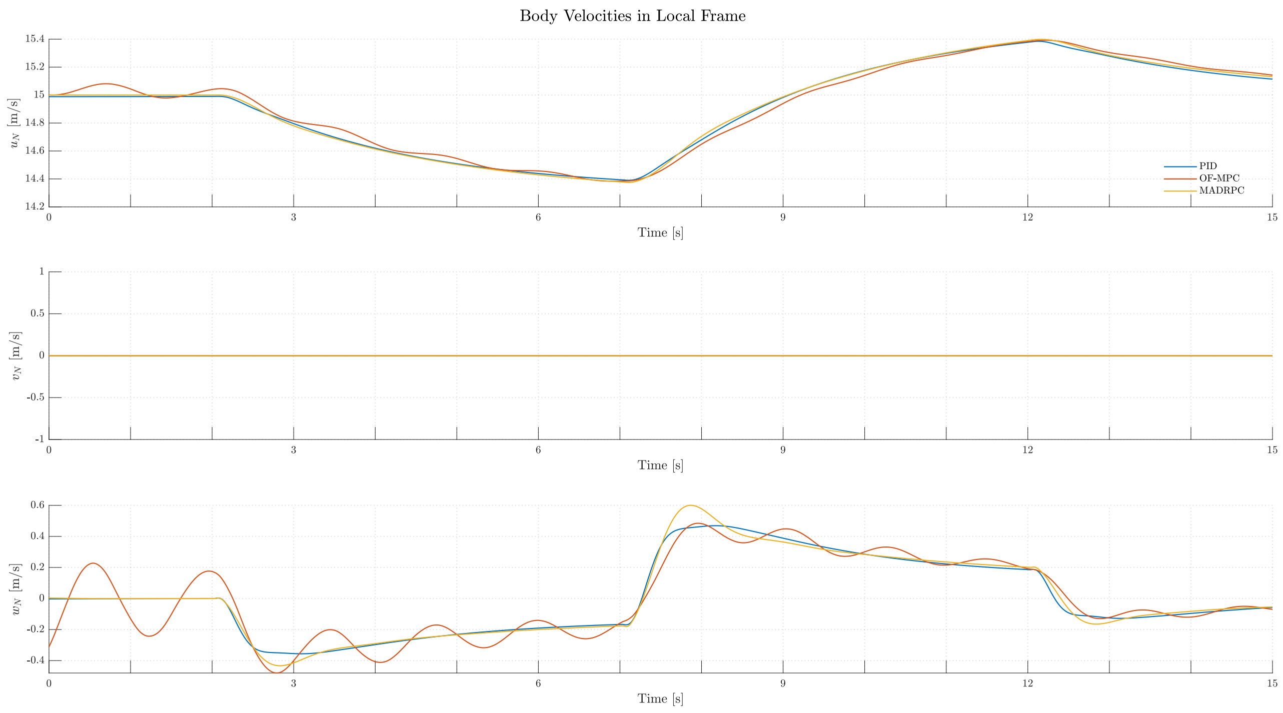


Figure 45: H200 body velocities expressed in local frame during **Test A.1**

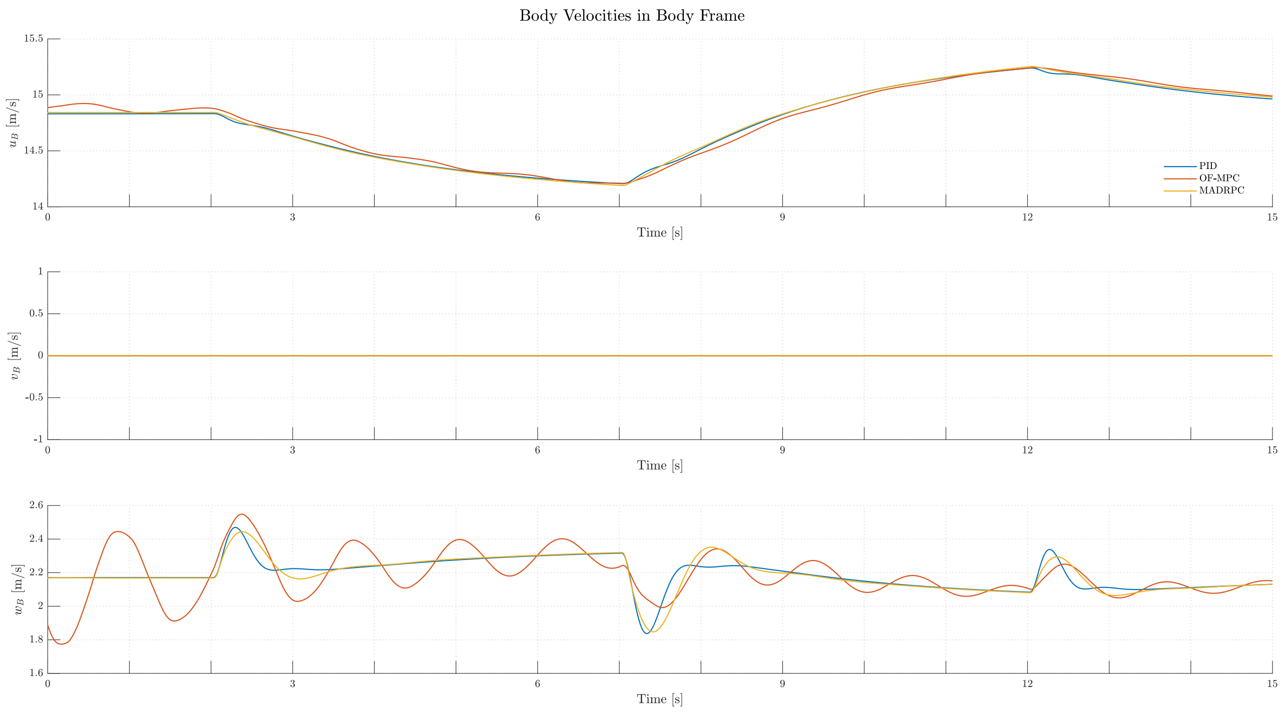


Figure 46: H200 body velocities expressed in body frame for **Test A.1**

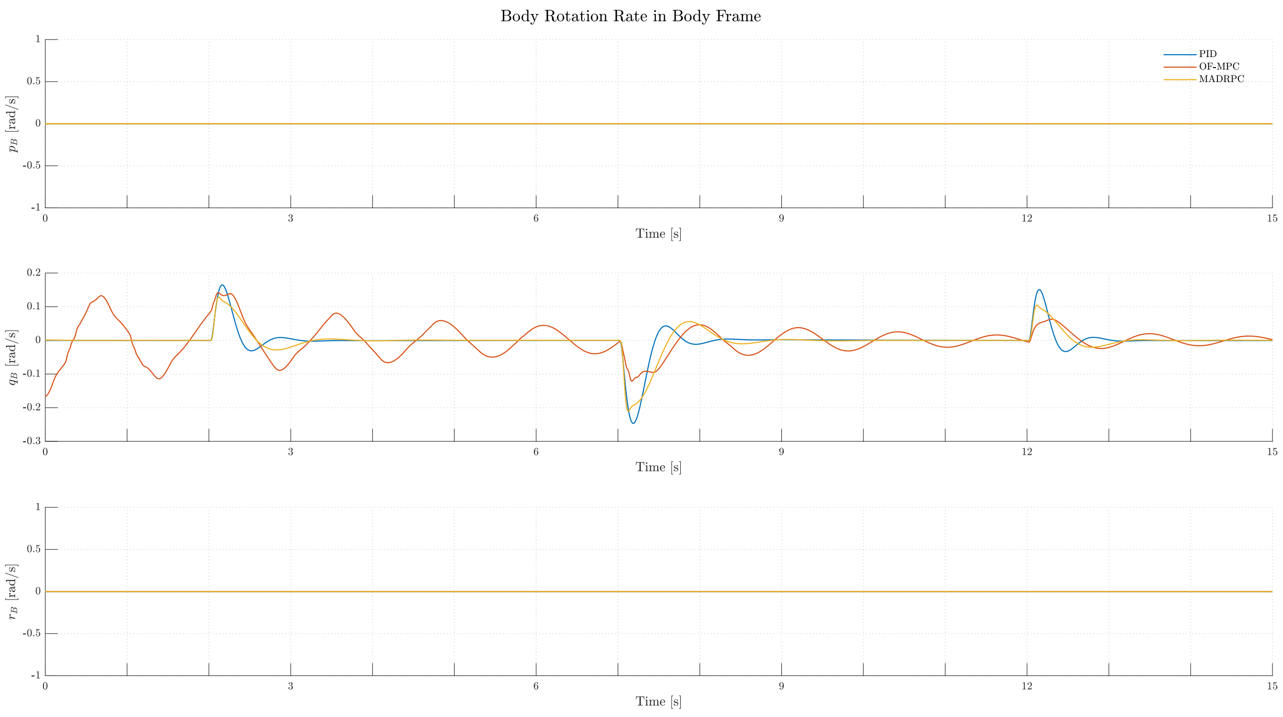


Figure 47: H200 body rotations during **Test A.1**

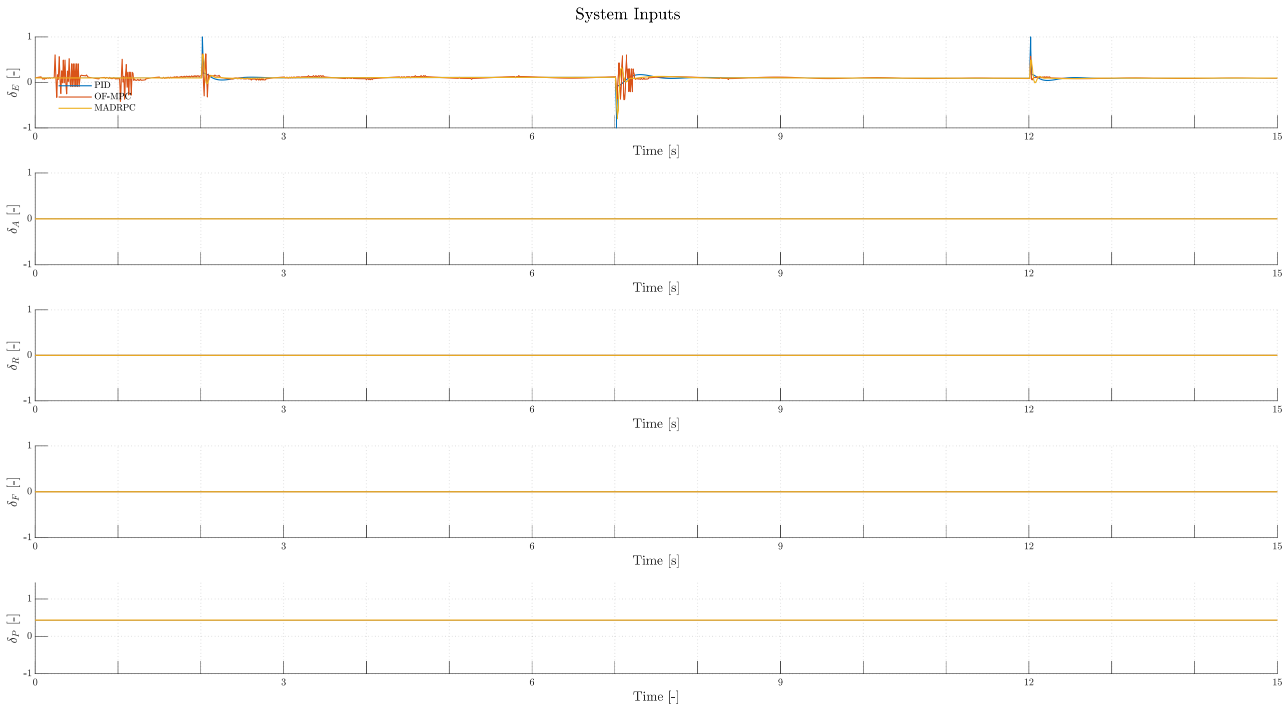


Figure 48: H200 system inputs during **Test A.1**

And the controllers performance indicators are:

	ITAE	ISE	IAE	MSE
PID	24.7999	0.058871	3.336	0.0039247
OF-MPC	59.7225	0.18317	11.7082	0.012212
MADRPC	30.5103	0.078314	4.2391	0.0052209

Table 17: Controller performance indicators for **Test A.1**

6.2.2.1.2 Test A2: Cruise Altitude, Greater Cruise Airspeed

The second test will focus on increasing the cruise airspeed during the H200 flight mission. The modified flight conditions for **Test A.2** will be:

Flight Conditions	
Altitude	100 m
Velocity	25 m/s

Table 18: Flight conditions for **Test A.2**

Once again, this modification in the cruise flight conditions imply a change in the trim value for the pitch angle:

$$\theta_{trim} = 0.8127\,\text{deg} \tag{86}$$

So, the reference pitch angle signal, with the same structure as in figure (30), will take the following numerical values to adapt it:

Symbol	Value	Units
T_1	2	s
T_2	7	s
T_3	12	s
θ_{trim}	0.8127	deg
θ_{ref_1}	2	deg
θ_{ref_2}	-2	deg

Table 19: Reference signal parameters for **Test A.2**

The result plots of **Test A.2** can be seen from figure (49) to (57).

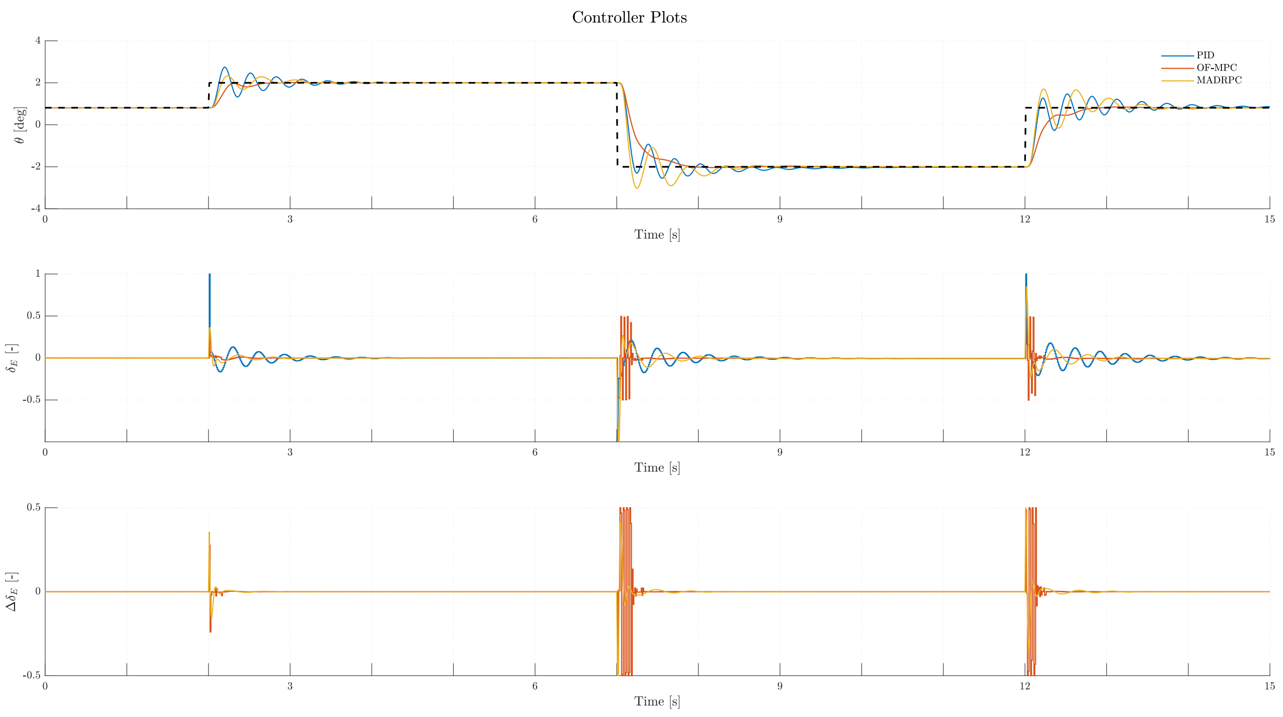


Figure 49: Controllers plots during **Test A.2**

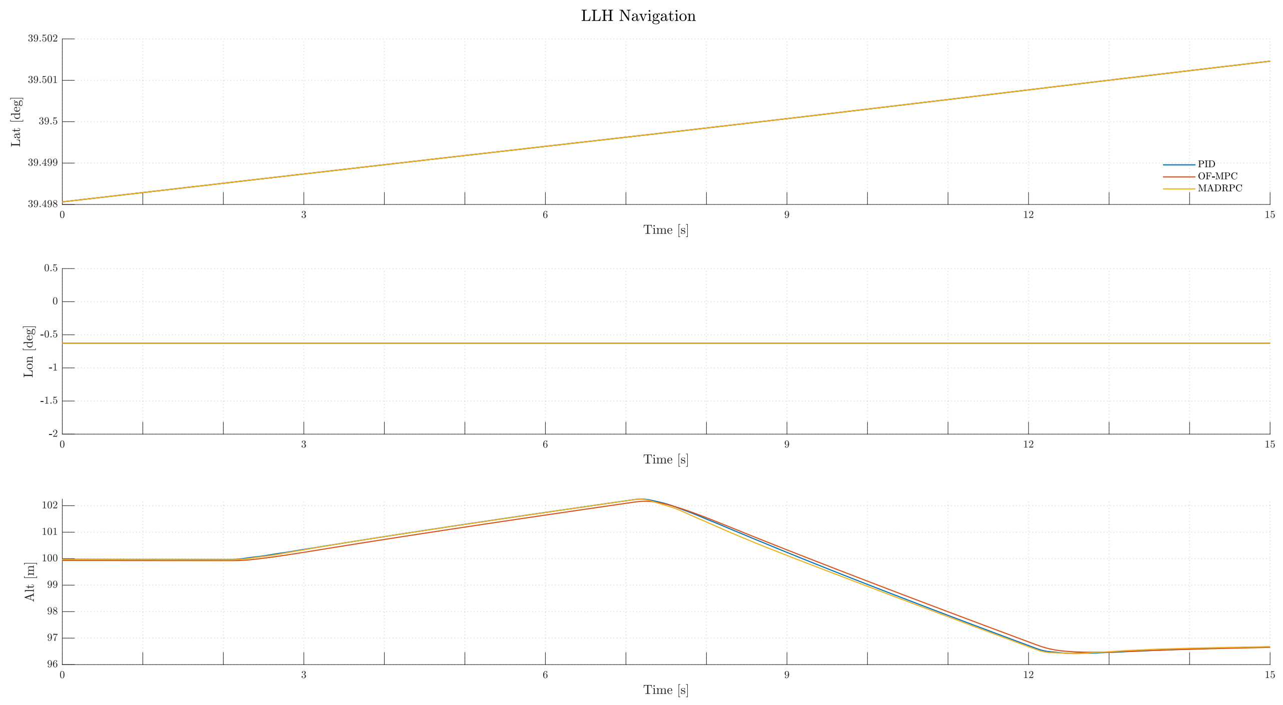


Figure 50: H200 LLH navigation during **Test A.2**

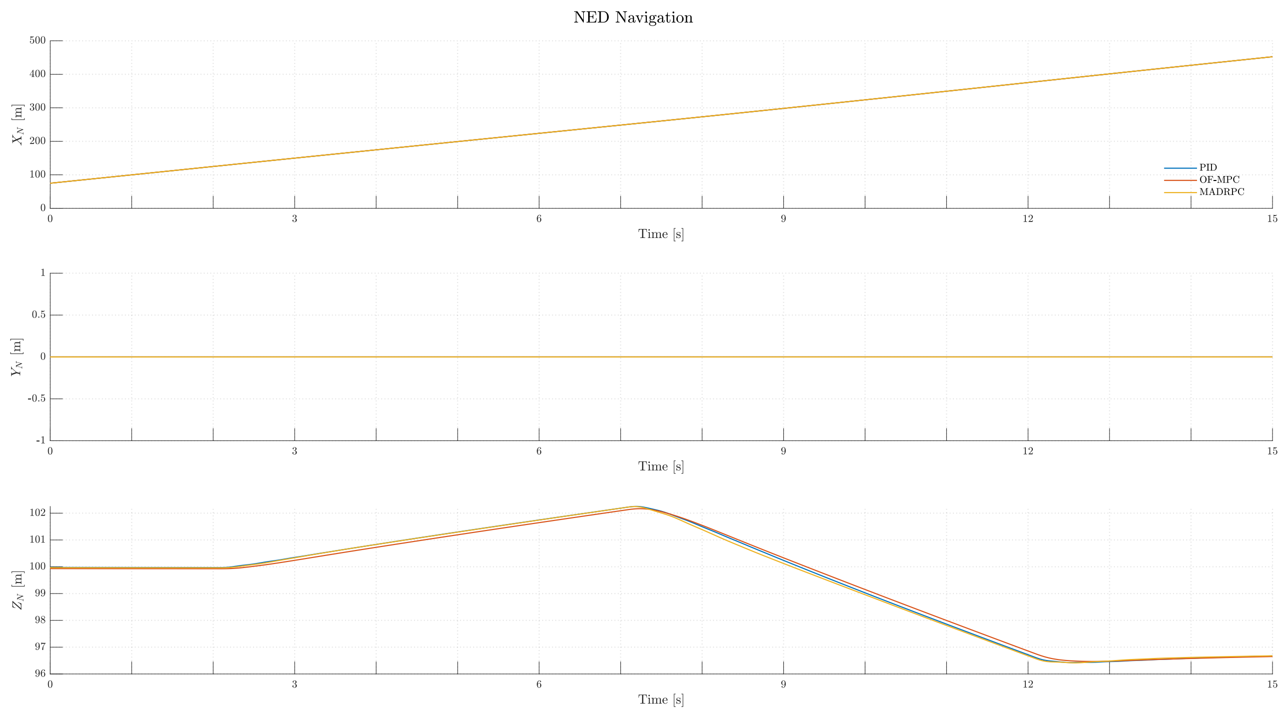


Figure 51: H200 NED navigation during **Test A.2**

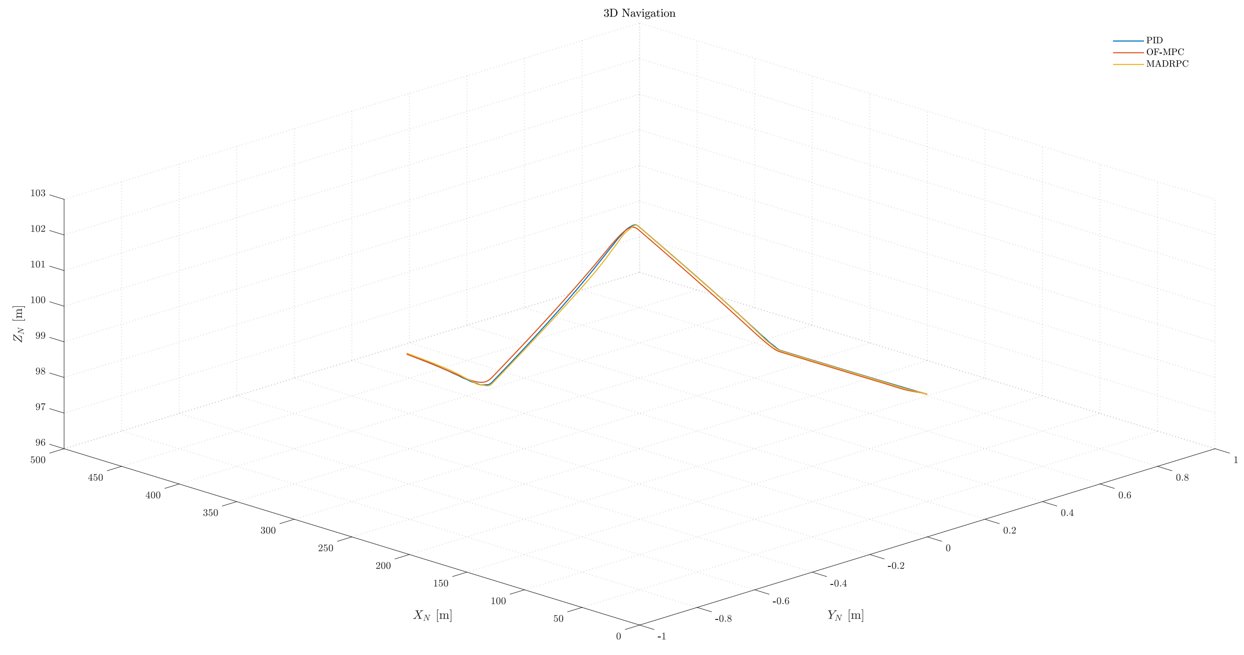


Figure 52: H200 3D navigation during **Test A.2**

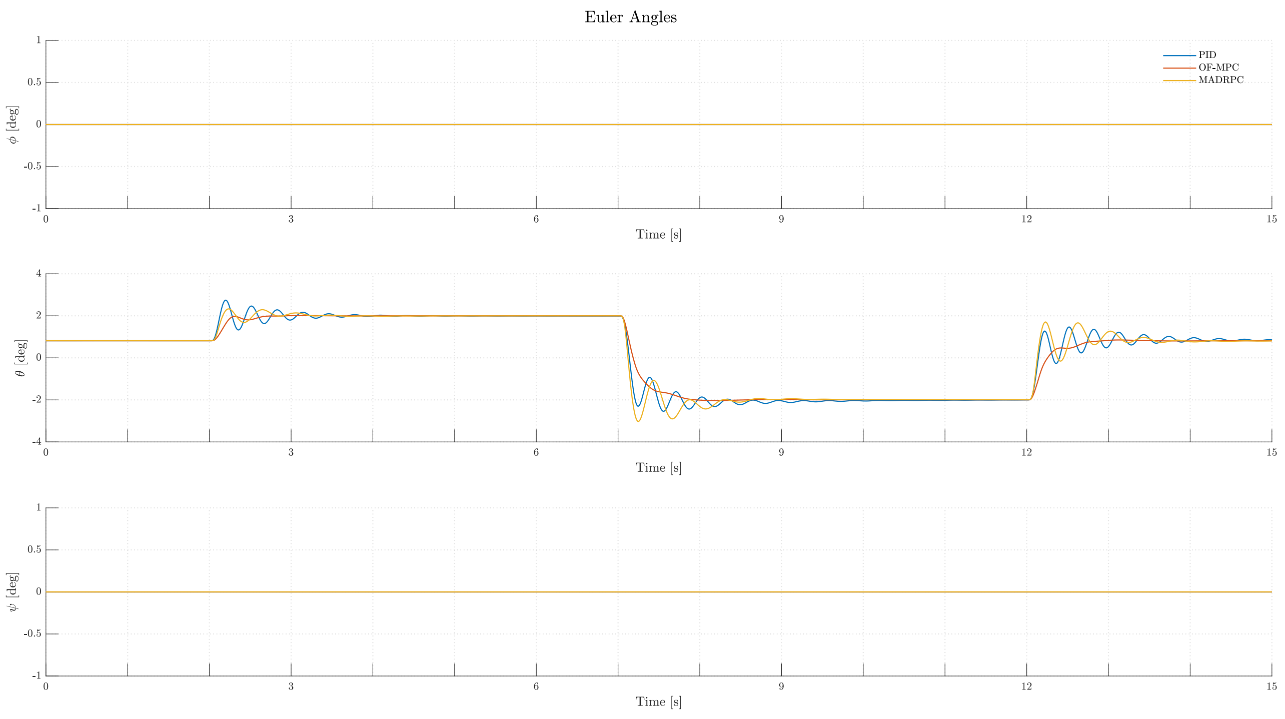


Figure 53: H200 Euler angles during **Test A.2**

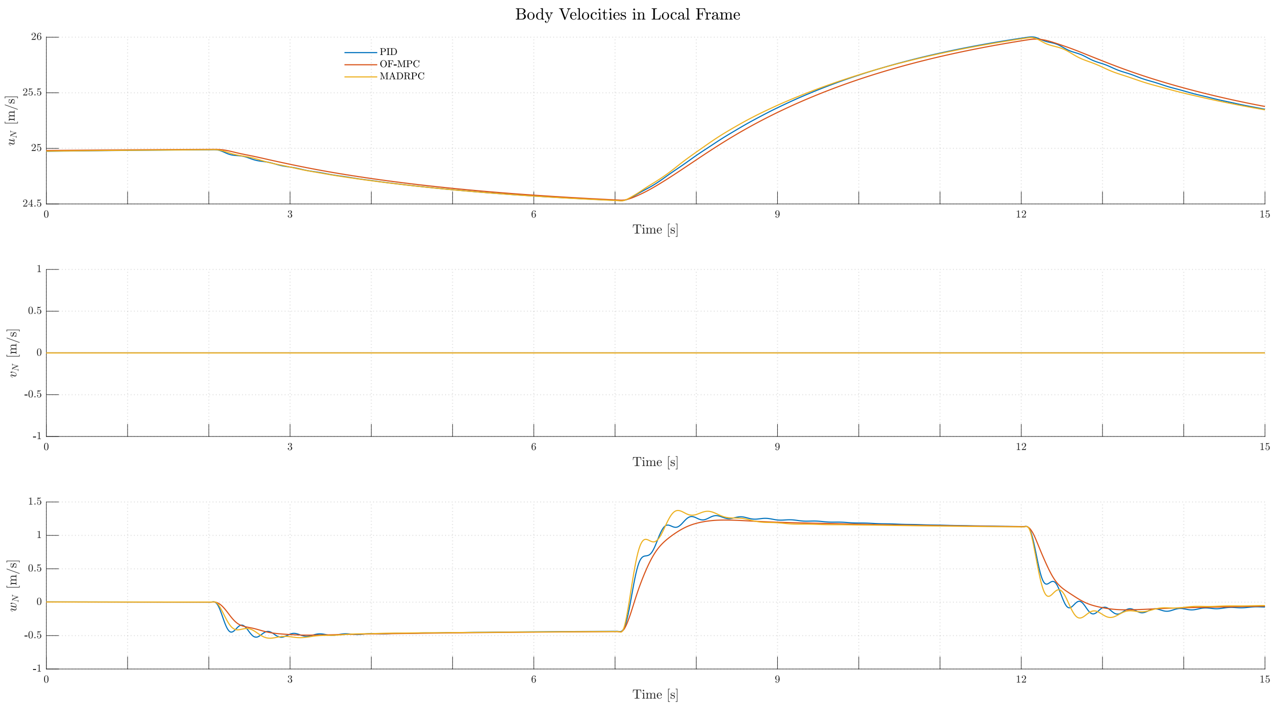


Figure 54: H200 body velocities expressed in local frame during **Test A.2**

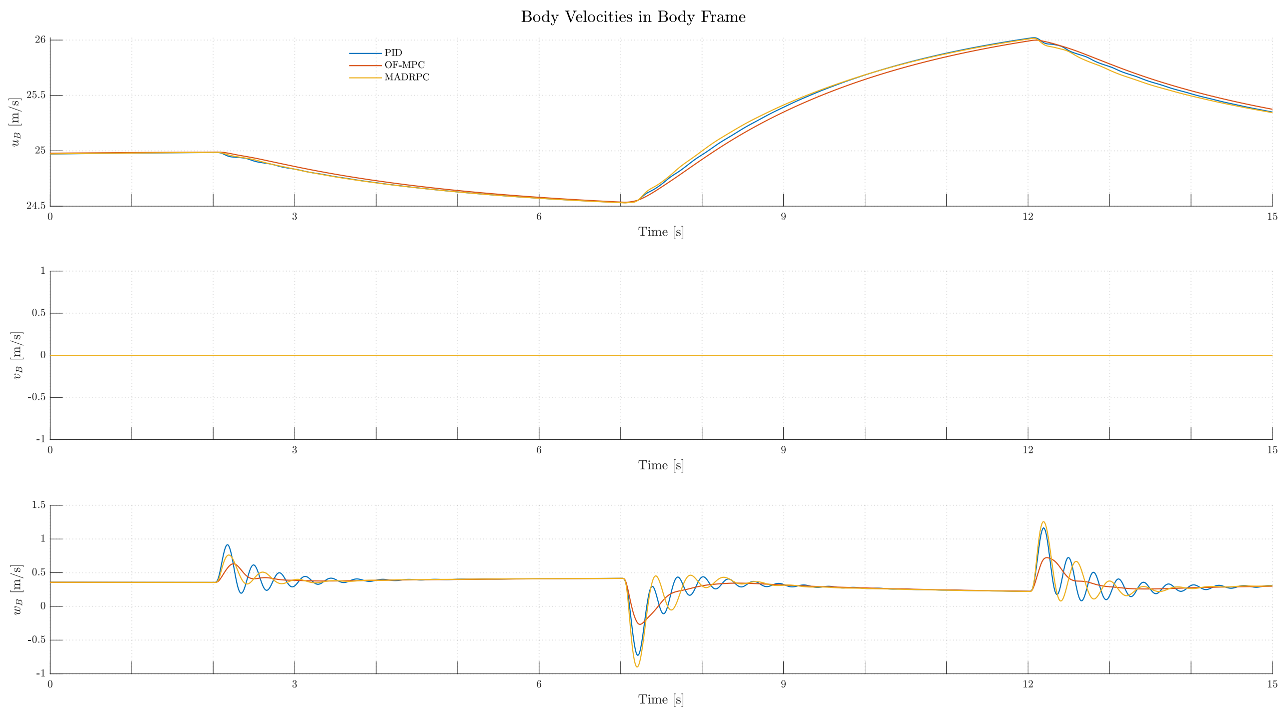


Figure 55: H200 body velocities expressed in body frame for **Test A.2**

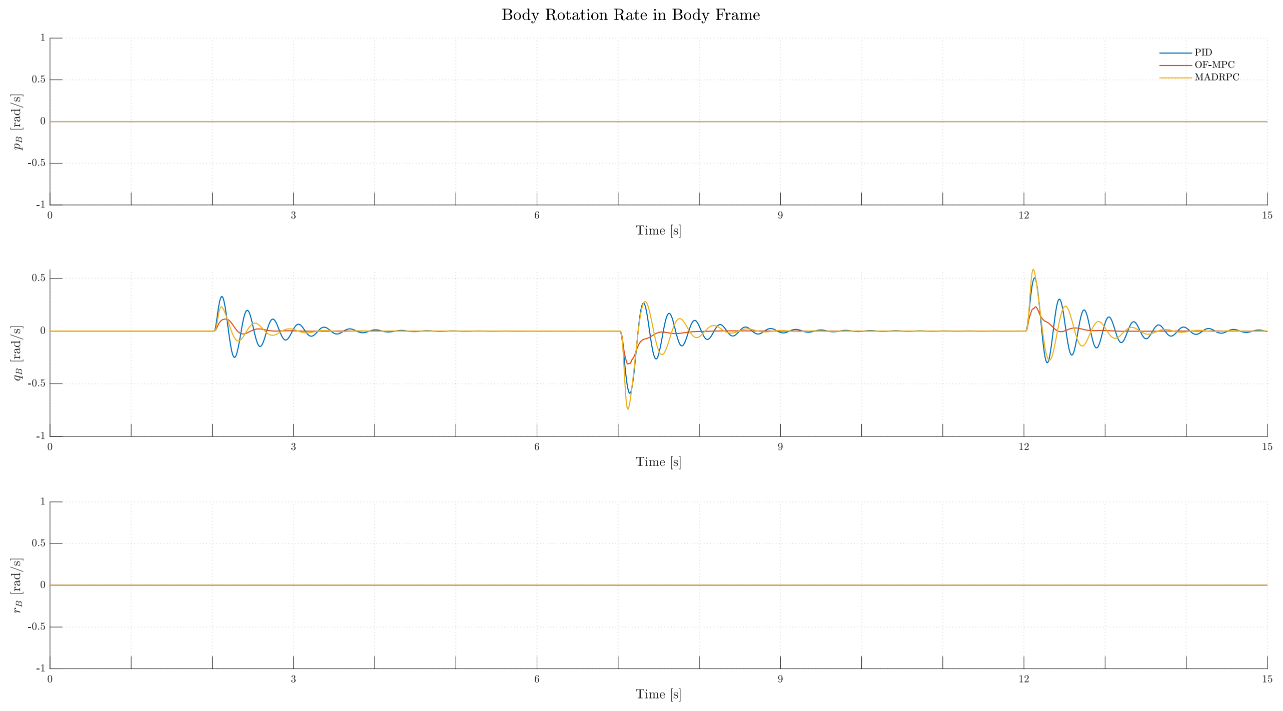


Figure 56: H200 body rotations during **Test A.2**

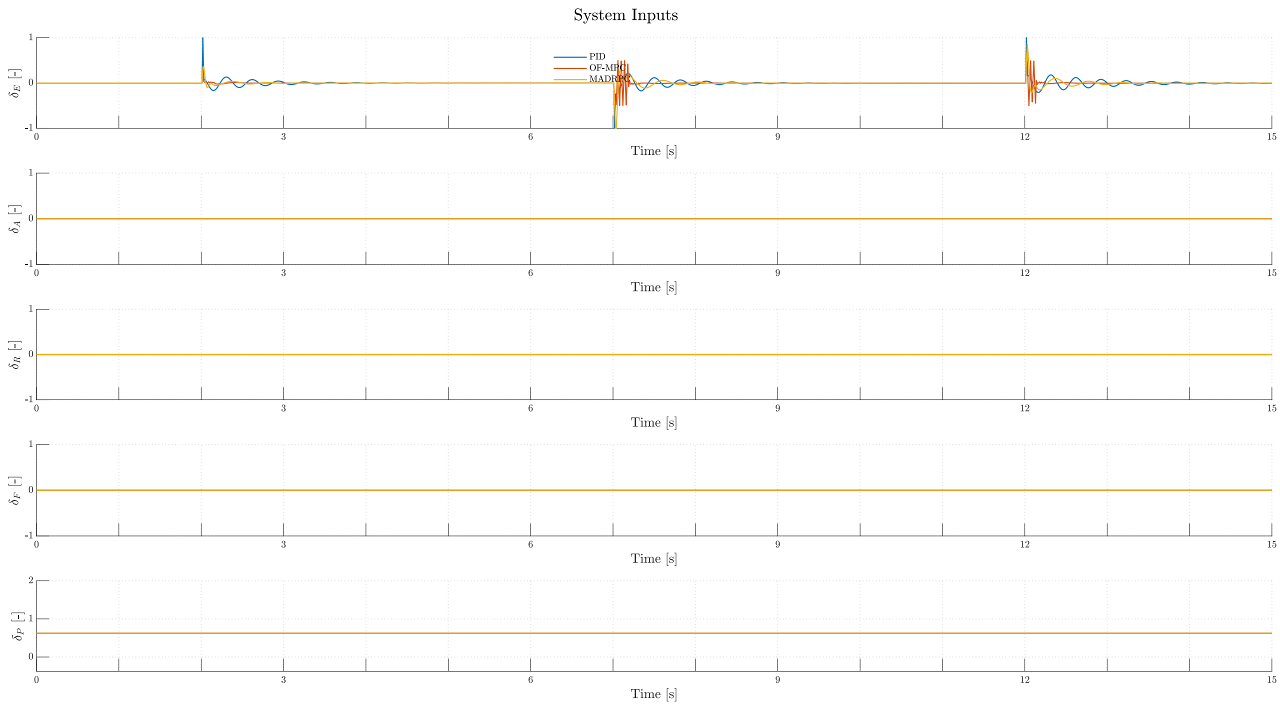


Figure 57: H200 system inputs during **Test A.2**

And the controllers performance indicators are:

	ITAE	ISE	IAE	MSE
PID	34.7006	0.095692	4.0974	0.0063795
OF-MPC	27.8623	0.11312	3.334	0.0075417
MADRPC	32.1284	0.092036	3.7105	0.0061357

Table 20: Controller performance indicators for **Test A.2**

6.2.2.1.3 Tests Conclusions

Once the tests have been performed, some conclusions may be drawn. First of all, although the flight conditions have been changed, the controllers are capable of governing the pitch angle, in the sense that the controllers response are not unstable.

Nevertheless, some commentaries may be given focusing on their performance. First of all, in figure (58) it may be seen the summary for the performance KPIs controller indicators.

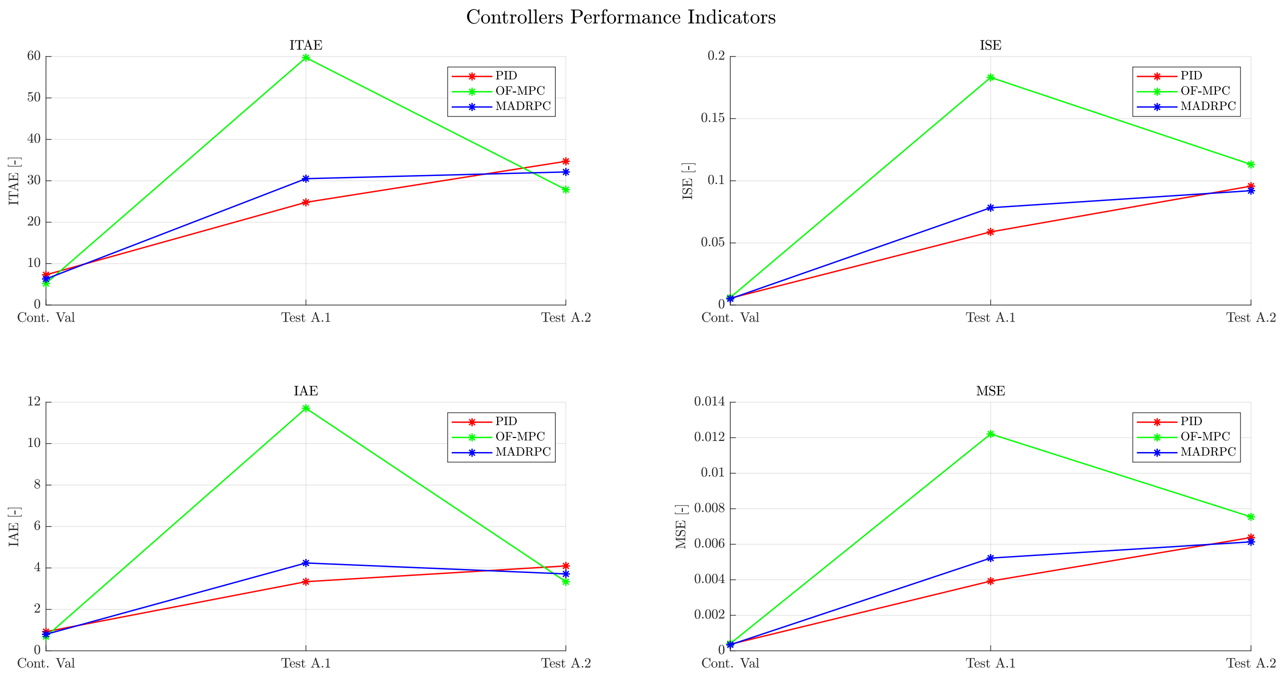


Figure 58: Controllers performance indicators for flight conditions perturbation tests

From figure (58), the performance of the controllers is worsened when compared with the nominal conditions tests, as expected. Specifically, in **Test A.1**, both PID and MADRPC control output the same performance, with the OF-MPC control being the worse from the three. This is a consequence of the OF-MPC relying heavily on its internal prediction and optimization process, so when the flight conditions are changed, the controller struggles to compensate the disturbance. Meanwhile, in **Test A.2**, the three controllers output a similar level of performance in all four KPIs. This is a remarkable fact taking into account that both PID and the OF-MPC control uses the complete information of the pitch angle process, either for design and tuning or inside the controller structure, respectively. On the other hand, the MADRPC controller, which is a control algorithm that uses the same optimization process as the OF-MPC, with the difference that instead of the real plant, it builds an modified first-order plant using only two natural parameters of the process is outputting the same performance level as the two traditional control schemes, which uses the complete information of the model, it's a remarkable fact and a powerful advantage over these two traditional control algorithms.

On the other hand, from the simulation plots, it may be seen how the flight speed that the H200 travels through the air plays a fundamental role inside the control process. The reason behind is that the lift force created by the H200 is directly dependent on the airspeed it travels, impacting the process behind the pitch angle model.

More specifically, in a controller by controller review, starting with the PID controller, being one of the most robust control schemes in industry, it is able to withstand the modification in flight conditions. In **Test A.1**, when the H200 generates less lift force (less airspeed), the response is more damped, with less overshoot as in the nominal conditions test. It may be seen that is one of the most robust from all three of them, as it only depends on the tracking error, not the model itself. It uses the information from the system to tune its parameters, not to actually control the process. From the control action plot, it may be seen how the PID controller is able to give a stable response without saturating the elevator deflection signal.

From **Test A.2**, with a higher flight airspeed, the PID controller introduces overshoot to the response, as it struggles to compensate the increase in lift force. From the elevator deflection signal plot, it may be seen how the PID control tries to stabilize the pitch angle, but with every step the controller is applied, the higher lift force make the H200 more difficult to control than in nominal conditions, arriving also to saturate the elevator deflection signal when the reference value changes, impacting the flight performance.

For the OF-MPC controller, in **Test A.1**, is incapable of governing the pitch angle through all the simulation, giving an oscillatory response during all the flight mission. The decrease in lift capabilities (less airspeed) than compared with the nominal conditions, has impacted the optimization process inside the controller, making it unable to give a stable response. Moreover, from the elevator deflection signal, it may be appreciated that through all the simulation is trying to compensate the oscillatory behavior, failing in its task. This is a consequence of the controller heavily relying on the information given from the system, as it has to predict the future behavior of the system, so with a different flight airspeed of the system, the controller has lost a high level of performance, rendering it almost unusable for maneuvers. For **Test A.2**, the performance is better than the first one, although the response may be appreciated that it is slower than in nominal conditions. This is a consequence of the underestimation of the lift capabilities inside the optimization process, makes an underguess of the H200 pitch angle, rendering the process slower. Nevertheless, it may be seen that every time the reference value changes, the OF-MPC saturates the control action signal, and when trying to compensate these drastic changes in the elevator deflection signal the controller has to go from the maximum to the minimum value of the control action signal. These sudden movements of the elevator deflection signal, they could affect the flight performance and destabilize the H200 and making it enter stall conditions.

For the MADRPC controller, its performance is similar to the PID controller in both tests. In **Test A.1**, with a decrease in lift capabilities, it makes the response more damped. This is a consequence of the assumed first-order model inside the optimization process being more closer to the real system, as it has decreased its level of non-linearity. It may be seen that, even with having the same optimization process of the OF-MPC controller, with the only difference of the actual predictive model that uses for calculating the optimized rate of control actions, the elevator deflection signal is far smoother than in the OF-MPC controller. In **Test A.2**, it may be drawn the same conclusions, where the response is similar to the PID controller, but with a far smoother elevator deflection signal than the OF-MPC controller. This is a remarkable fact, that a controller (MADRPC) that does not hold any prior information of the process, apart from two natural parameters of the system, but not requiring to have a complete description of the system, is able to give a similar or even better performance than these two traditional control schemes (PID and OF-MPC). This is achieved because of the assumed first-order system inside the optimization process, and the compensation of the difference between the real and the modified one inside the controller, which gives a smooth behavior as with compared to the OF-MPC and does not rely on the process, making it more flexible against external disturbances, similar to the PID controller.

6.2.2.2 Structural Perturbation Tests

The second family of tests will focus on changing the H200 structural parameters, implying a modification on the inner non-linear model, which results in a different linear transfer function for the design phase. The objective is to check the controller performance inside the operative range for their inner parameters, including:

- The aerodynamic model.
- The propulsion model.
- The aircraft mass.

The first two tests will focus on the aerodynamic model modification, as the aerodynamic coefficients computation come from numerical simulations (CFD, panel method, etc), semi-empirical laws, experimental data, etc. In general flight mechanics, calculating a correct aerodynamic model for an aircraft is incredibly difficult, and needs a lot of resources to get it right. These two tests will focus on the underestimation and overestimation for the aerodynamic coefficients.

Then, the second pair of tests will focus on the engine propeller model, relating the throttle input signal with the actual thrust of the aircraft. This is of key importance, as a wrong model will result in wrong engine thrust force and moments. The tests will focus, once again, in a, underestimation and overestimation of some of the propulsion coefficients, as they are the product of experimental data in a test bench.

Lastly, for the aircraft mass, the one used for the controllers design phase was taken as the nominal configuration for the H200. Nevertheless, the H200 has to be able to operate between a mass range, as the H200 may have more mass as part of its payload, and it has to be able to drop that payload and still operate, with a lower mass.

These tests will focus on changing these parameters and check the controllers performance against a modification of the H200 inner parameters. The reference signal that will be fed into the system will have the same structure as in figure (30). The numerical values for the reference signal, as with the flight conditions tests, the numerical values may vary, as the flight operation point will change, as the inner model from where it is derived will change.

6.2.2.2.1 Test B.1: Underestimation of the Aerodynamic Coefficients

The first test will focus on an underestimation of the aerodynamic coefficients, specifically, the ones directly affect by the pitch angle and the elevator deflection. The modified coefficients will take the following form:

$$\begin{aligned} C_{L_{\theta,mod}} &= C_{L_\theta} \cdot k \\ C_{m_{\theta,mod}} &= C_{m_\theta} \cdot k \\ C_{L_{\delta_E,mod}} &= C_{L_{\delta_E}} \cdot k \\ C_{m_{\delta_E,mod}} &= C_{m_{\delta_E}} \cdot k \end{aligned} \tag{87}$$

where k is the modification parameter inside the aerodynamic model. In this test, the modification parameter will have the value:

$$k = 0.5 \tag{88}$$

With this, the trim pitch angle for the flight will be:

$$\theta_{trim} = 5.1218\,\mathrm{deg} \tag{89}$$

So, the reference pitch angle signal will be:

Symbol	Value	Units
T_1	2	s
T_2	7	s
T_3	12	s
θ_{trim}	5.1218	deg
θ_{ref_1}	7	deg
θ_{ref_2}	4	deg

Table 21: Reference signal parameters for **Test B.1**

The result plots of **Test B.1** can be seen from figure (59) to (67):

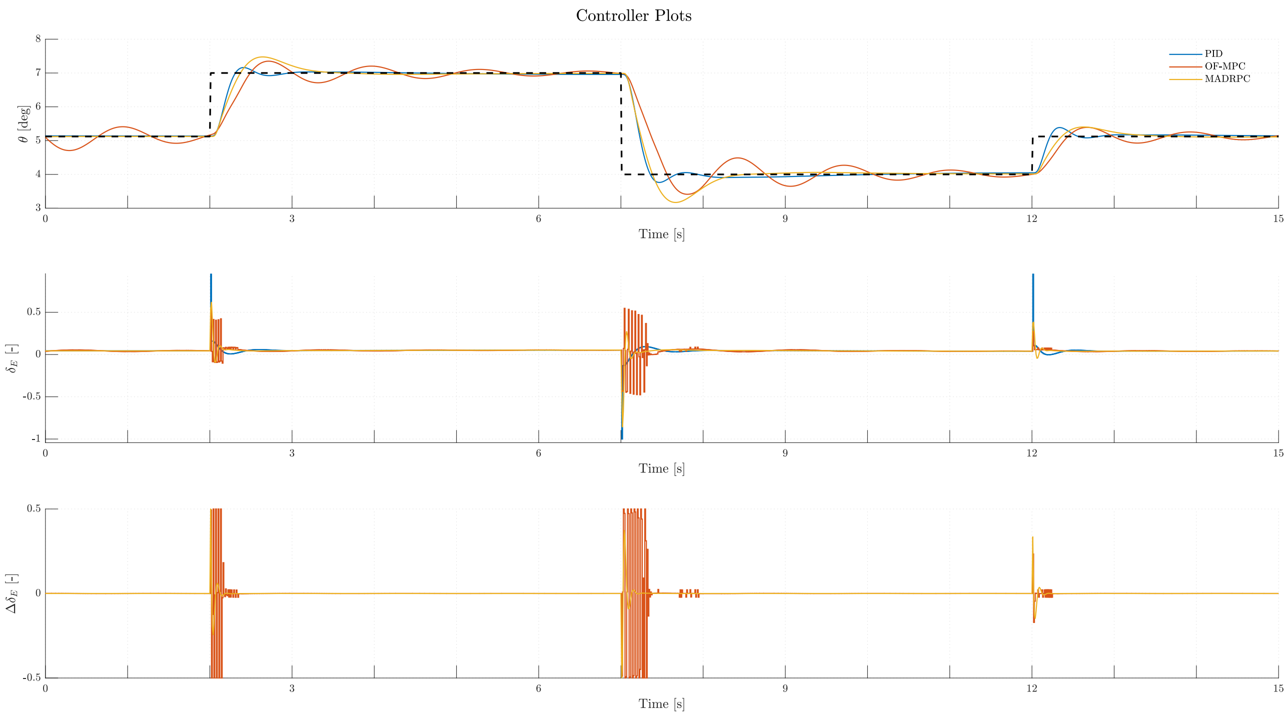


Figure 59: Controllers plots during **Test B.1**

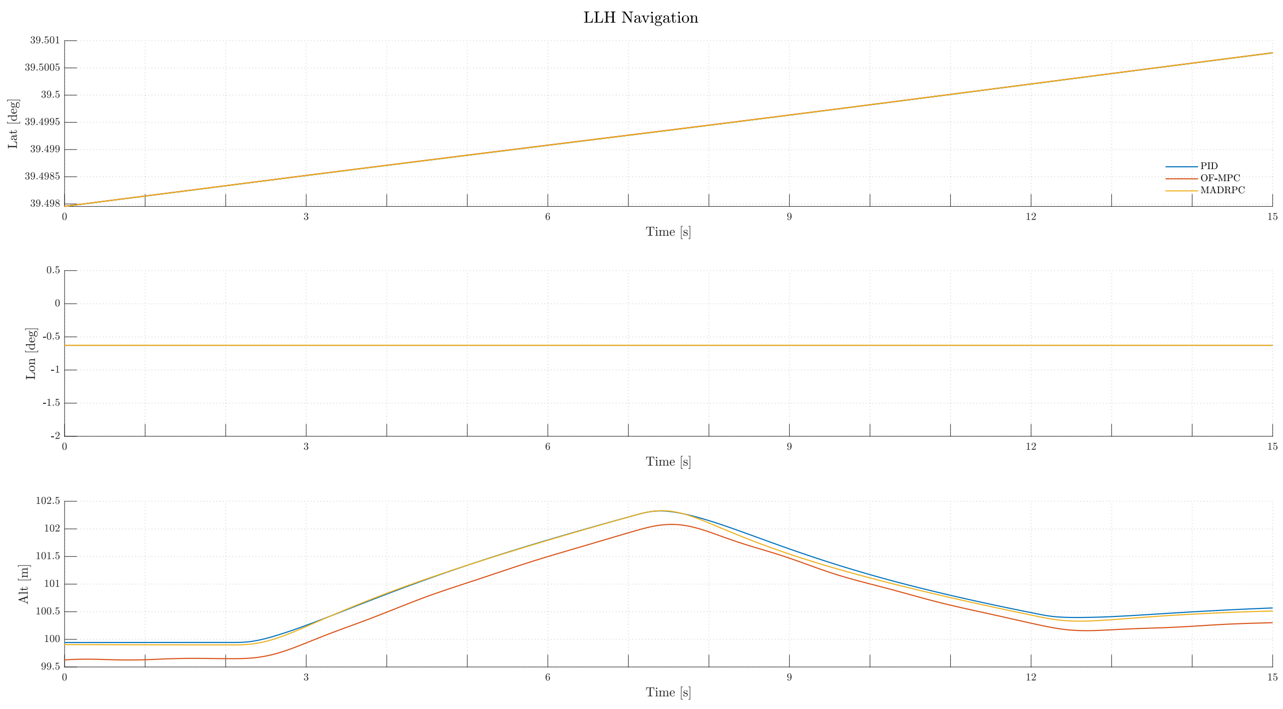


Figure 60: H200 LLH navigation during **Test B.1**

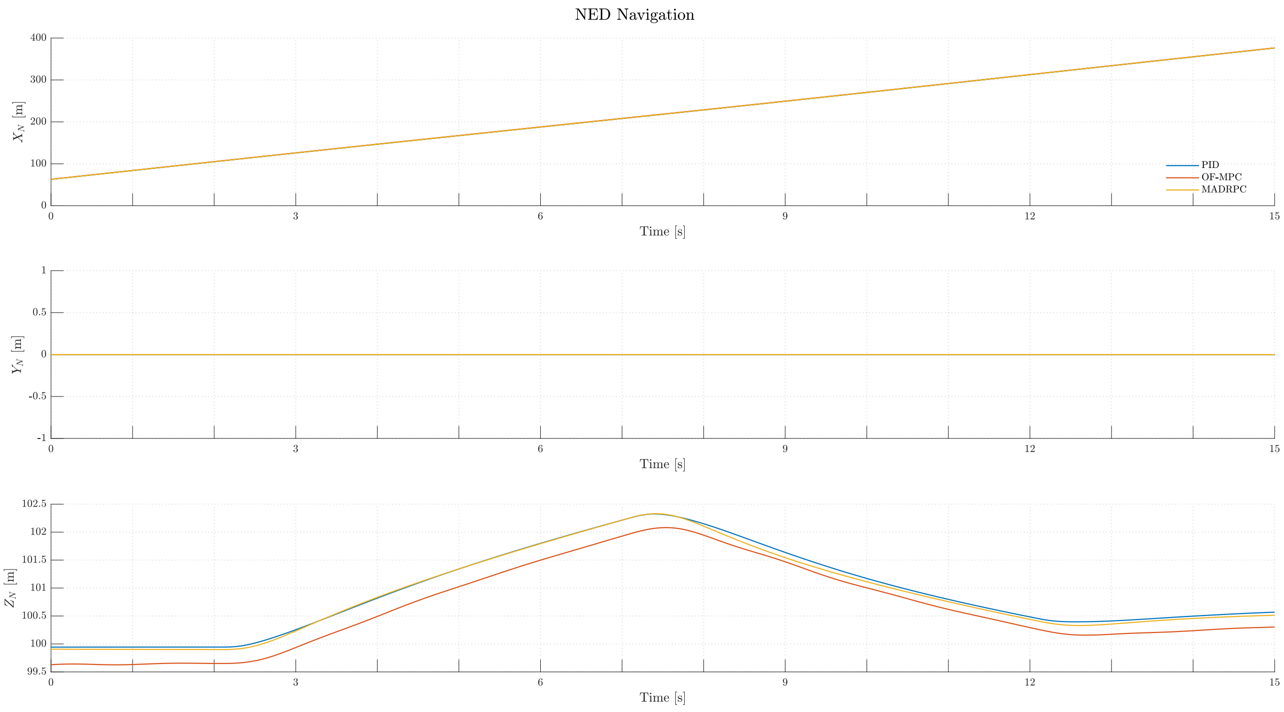


Figure 61: H200 NED navigation during **Test B.1**

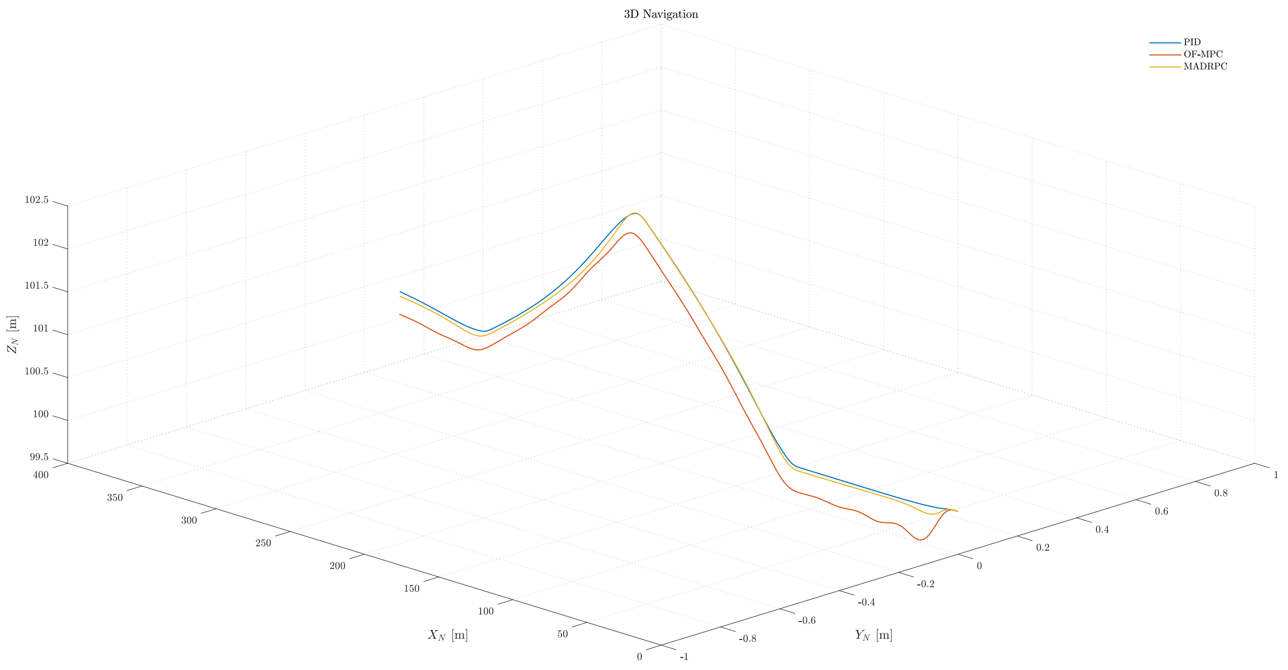


Figure 62: H200 3D navigation during **Test B.1**

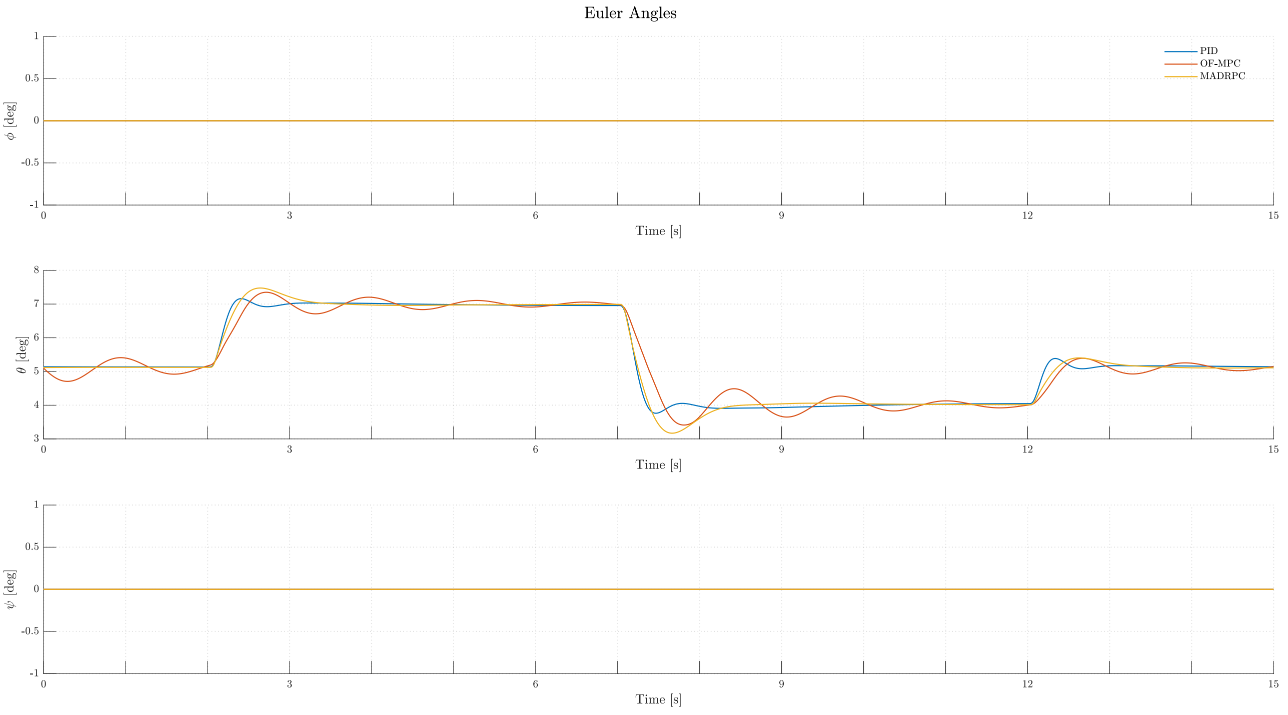


Figure 63: H200 Euler angles during **Test B.1**

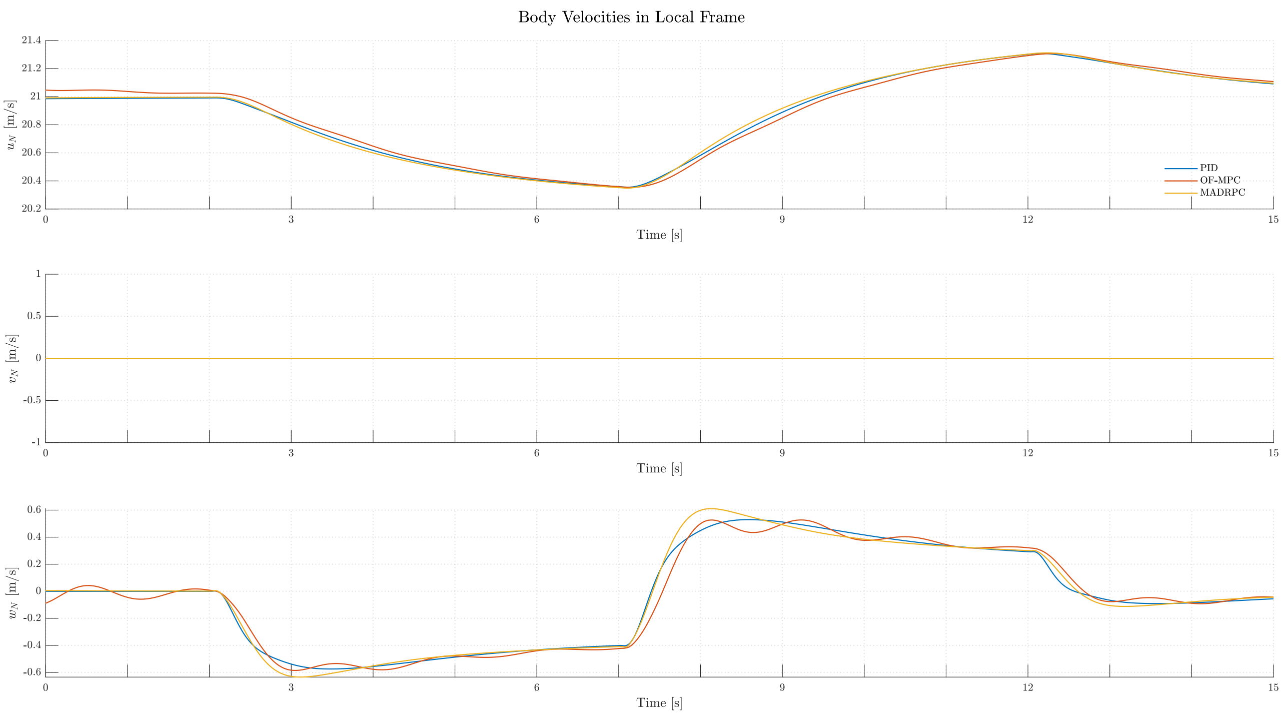


Figure 64: H200 body velocities expressed in local frame during **Test B.1**

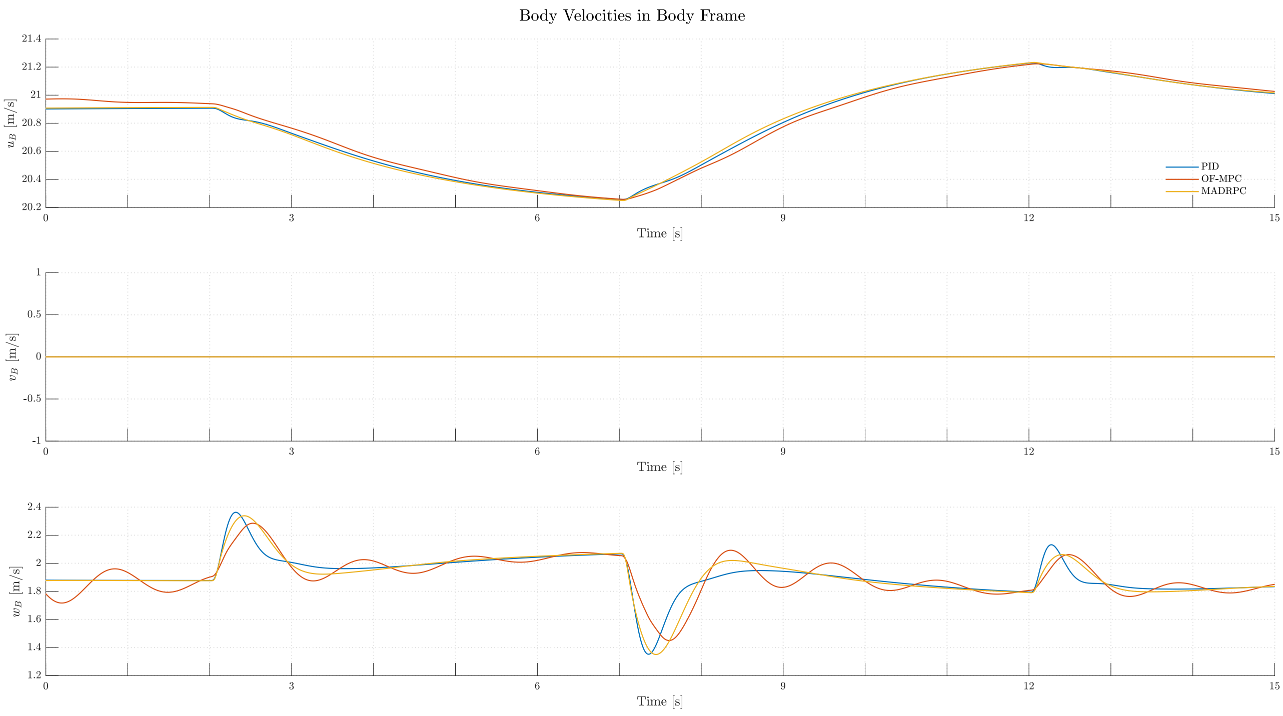


Figure 65: H200 body velocities expressed in body frame for **Test B.1**

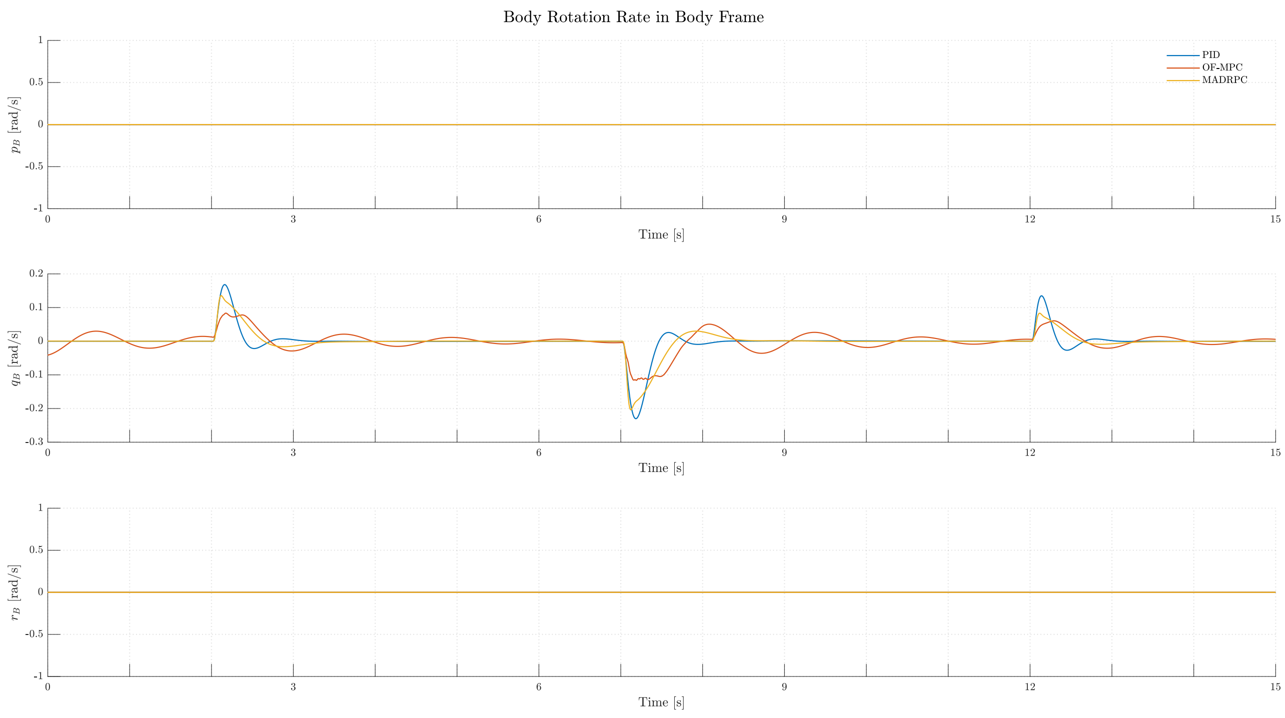


Figure 66: H200 body rotations during **Test B.1**

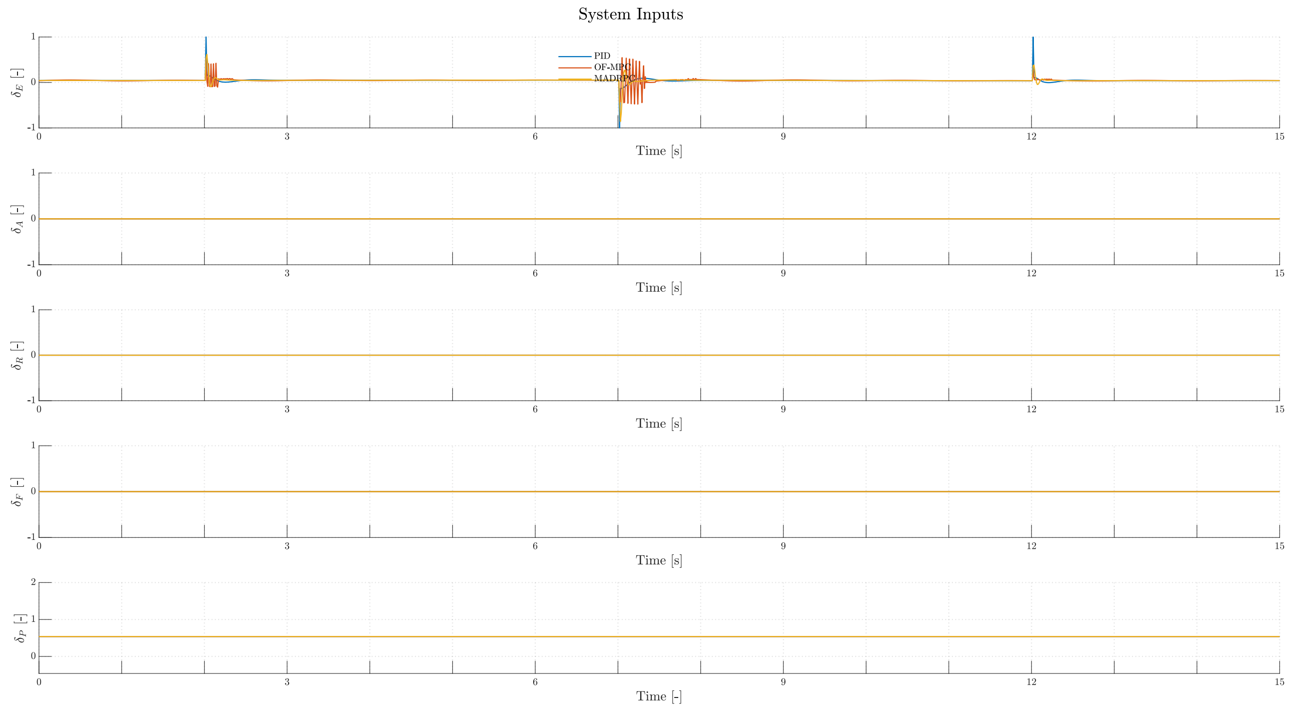


Figure 67: H200 system inputs during **Test B.1**

And the controllers performance indicators are:

	ITAE	ISE	IAE	MSE
PID	19.575	0.060771	2.7981	0.0040514
OF-MPC	42.6645	0.10371	2.7981	0.006914
MADRPC	27.6817	0.076262	3.9923	0.0050841

Table 22: Controller performance indicators for **Test B.1**

6.2.2.2.2 Test B.2: Overestimation of the Aerodynamic Coefficients

The second test will focus on the overestimation of the aerodynamic coefficients, to finish up setting the controllers boundaries for the aerodynamic operative range. Once again, the modified coefficients will take the following form:

$$\begin{aligned} C_{L_{\theta,mod}} &= C_{L_\theta} \cdot k \\ C_{m_{\theta,mod}} &= C_{m_\theta} \cdot k \\ C_{L_{\delta_E,mod}} &= C_{L_{\delta_E}} \cdot k \\ C_{m_{\delta_E,mod}} &= C_{m_{\delta_E}} \cdot k \end{aligned} \tag{90}$$

where k is the modification parameter inside the aerodynamic model. In this test, the modification parameter will have the value:

$$k = 1.5 \tag{91}$$

With this, the trim pitch angle for the flight will be:

$$\theta_{trim} = 1.7337\,\text{deg} \tag{92}$$

So, the reference pitch angle signal will be:

Symbol	Value	Units
T_1	2	s
T_2	7	s
T_3	12	s
θ_{trim}	1.7337	deg
θ_{ref_1}	3	deg
θ_{ref_2}	-1	deg

Table 23: Reference signal parameters for **Test B.2**

The result plots of **Test B.2** can be seen from figure (68) to (76):

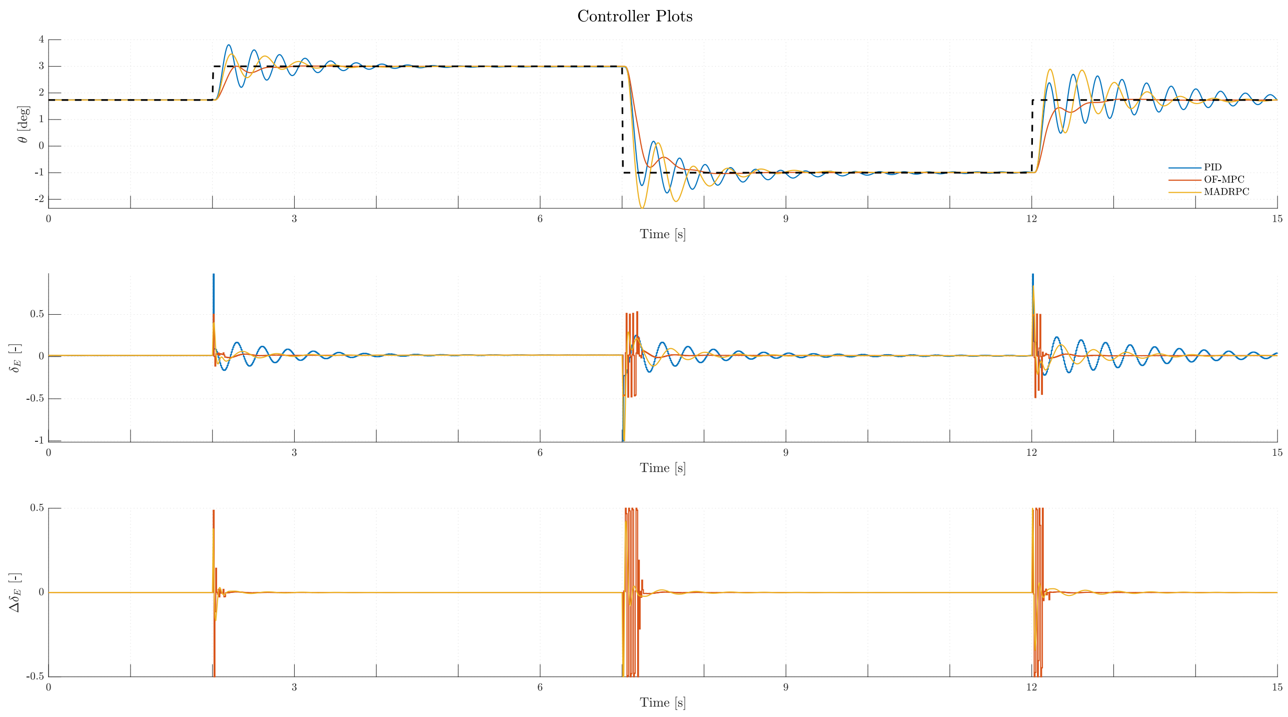


Figure 68: Controllers plots during **Test B.2**

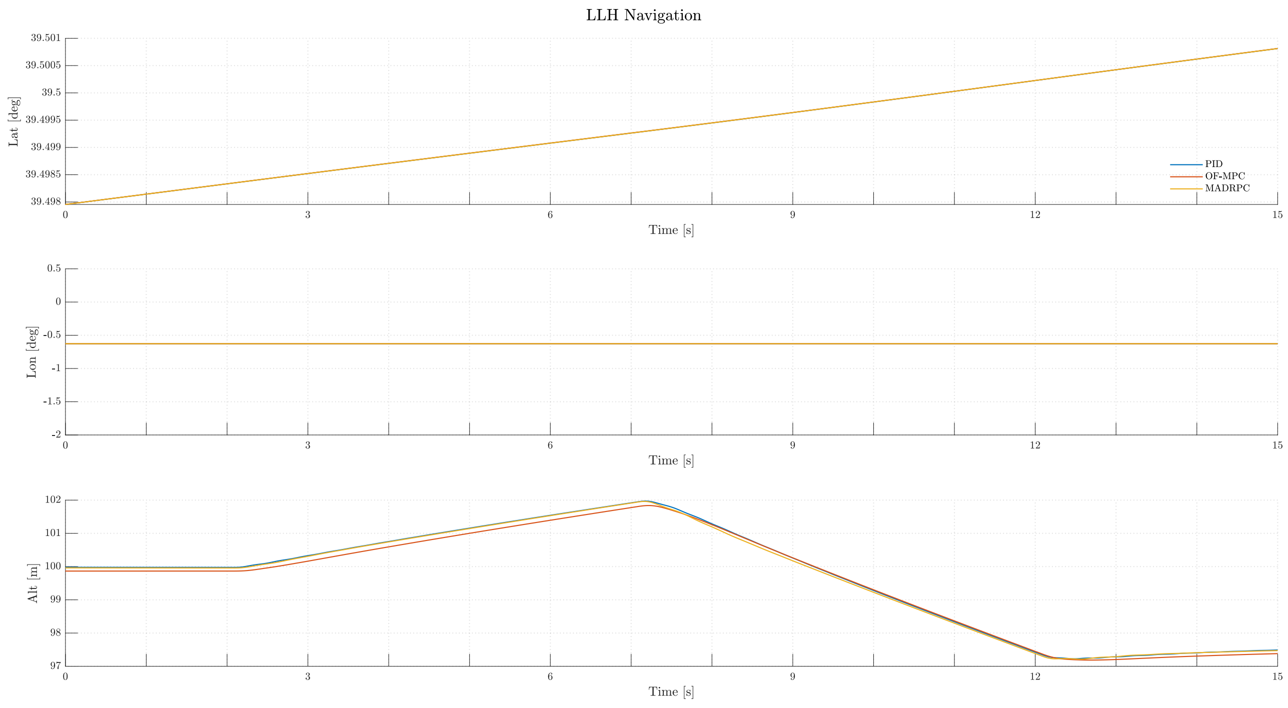


Figure 69: H200 LLH navigation during **Test B.2**

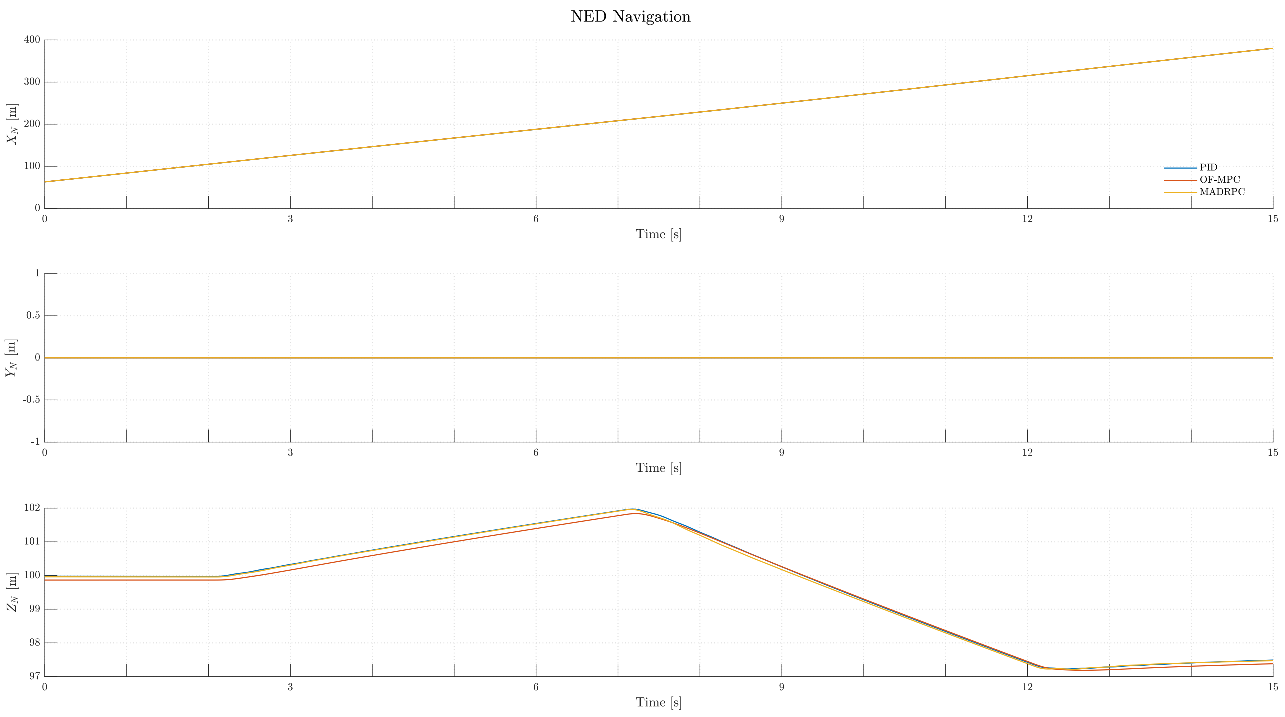


Figure 70: H200 NED navigation during **Test B.2**

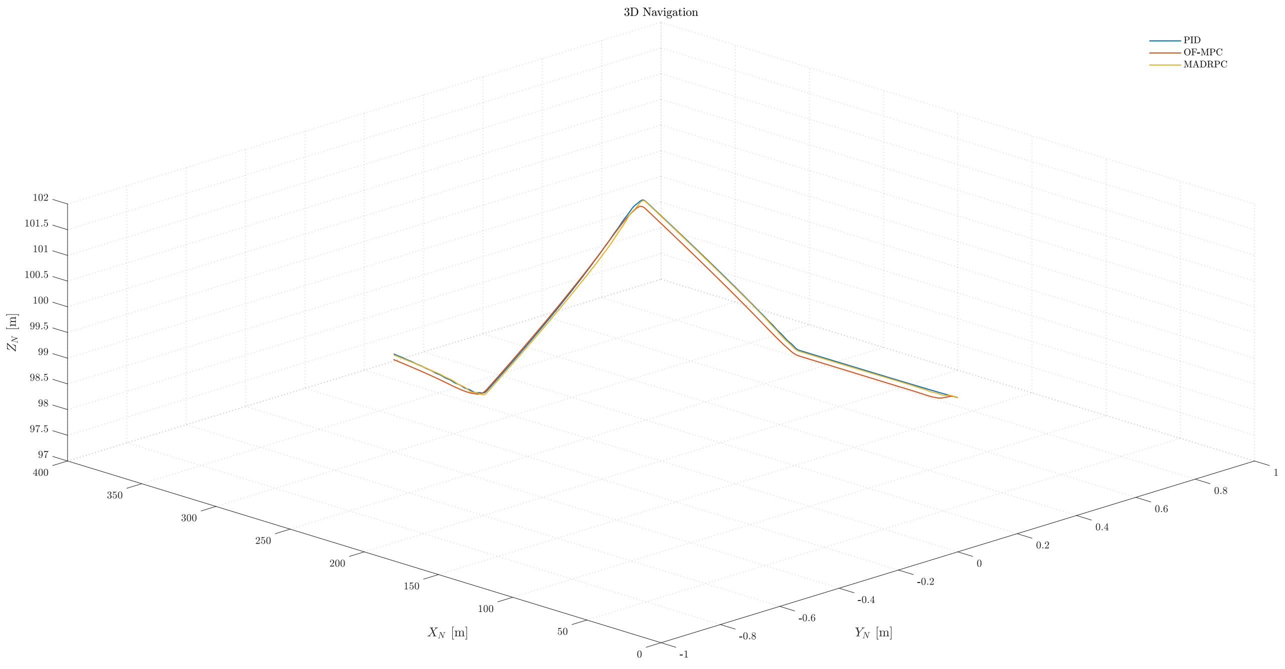


Figure 71: H200 3D navigation during **Test B.2**

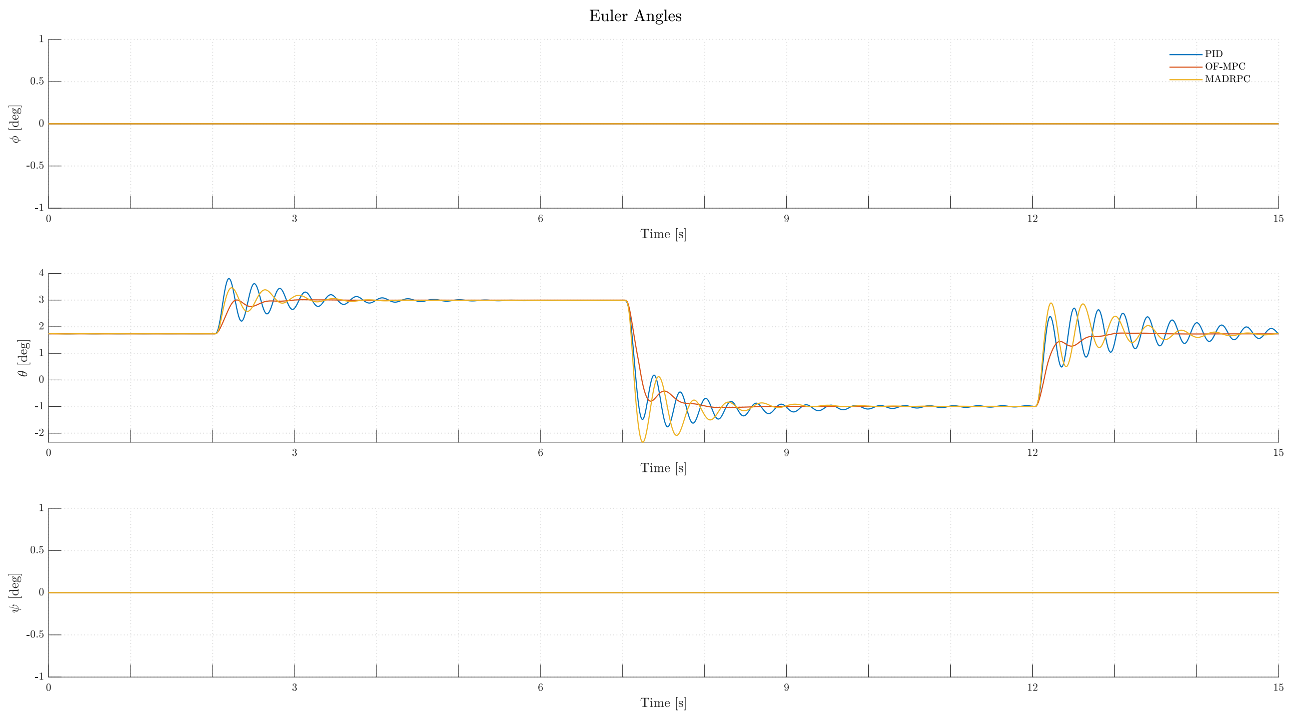


Figure 72: H200 Euler angles during **Test B.2**

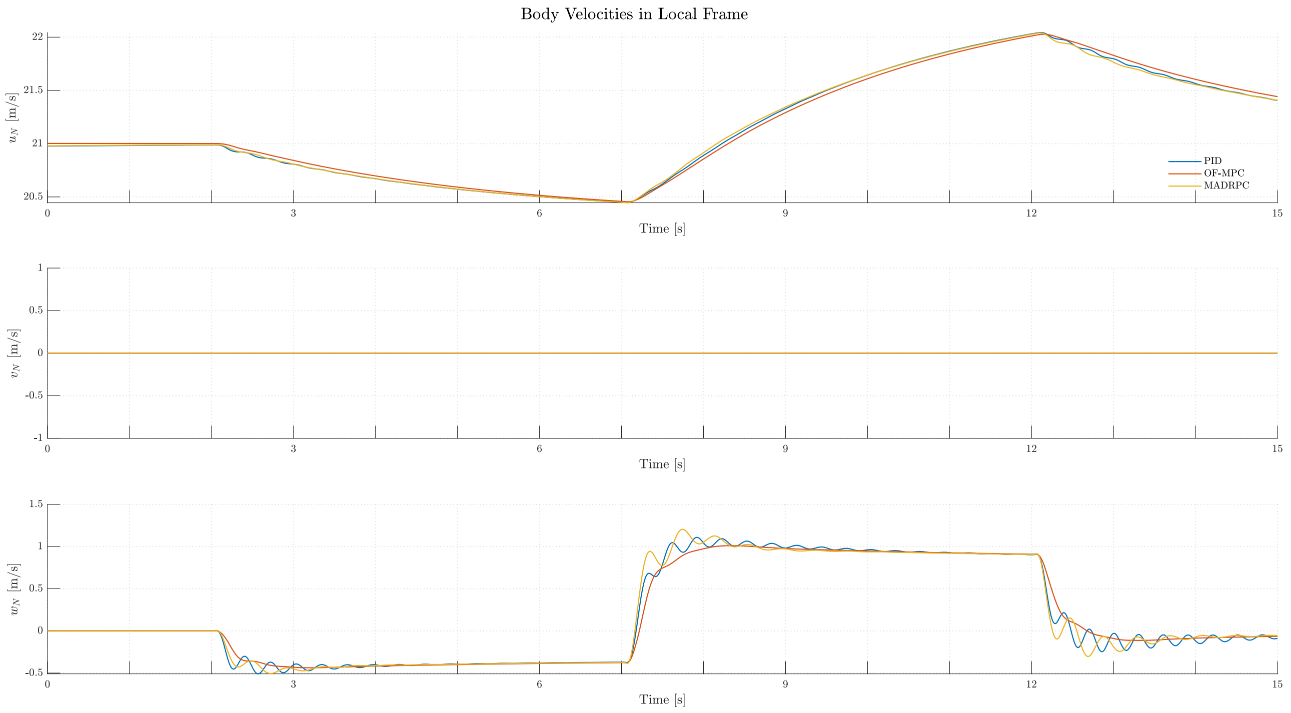


Figure 73: H200 body velocities expressed in local frame during **Test B.2**

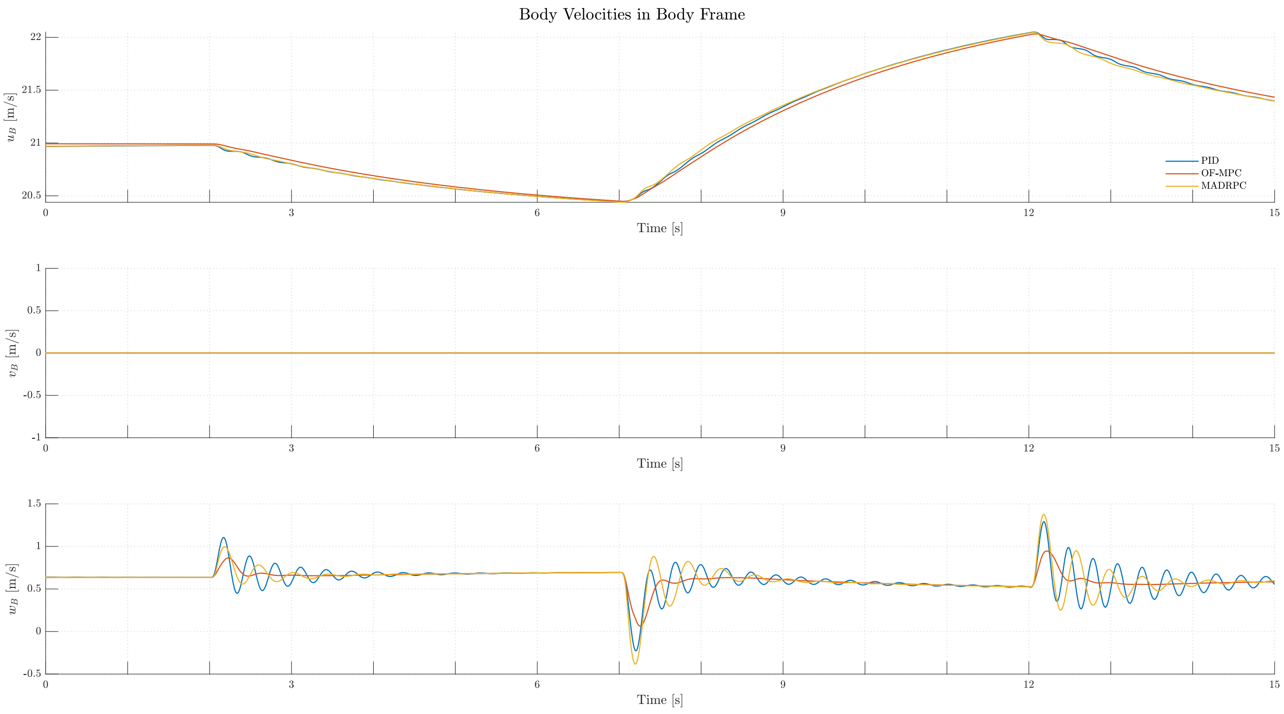


Figure 74: H200 body velocities expressed in body frame for **Test B.2**

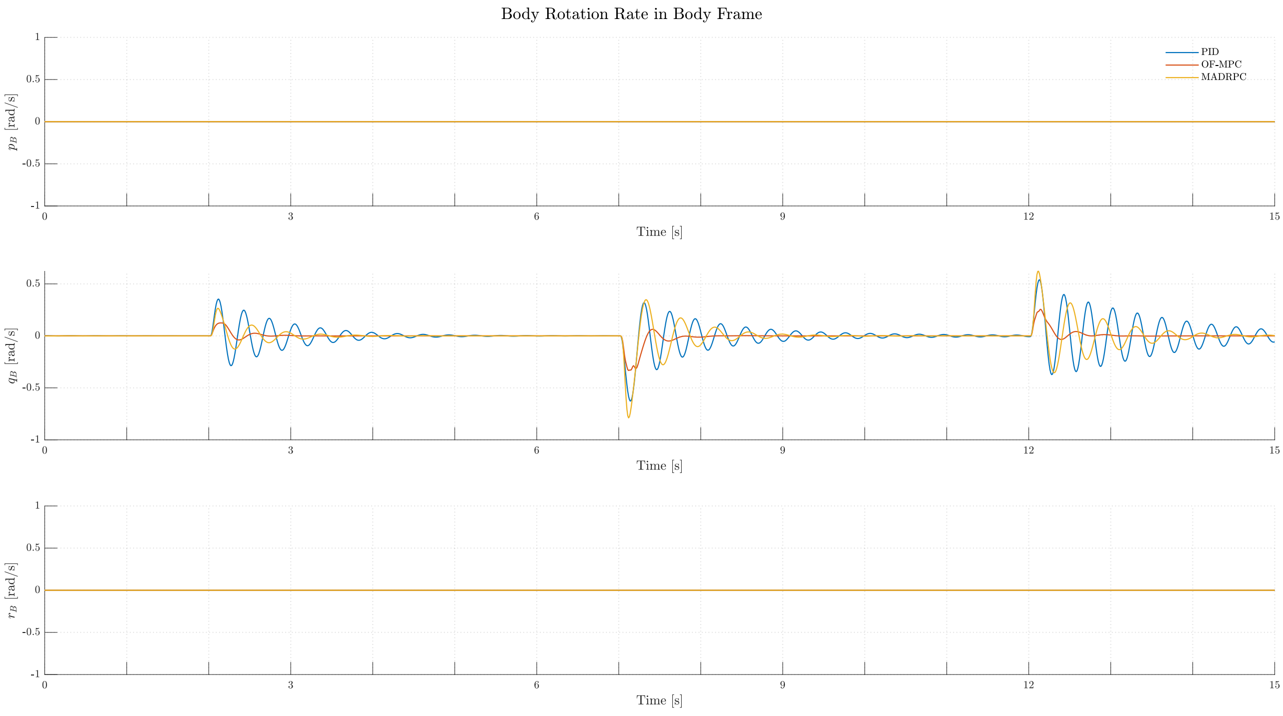


Figure 75: H200 body rotations during **Test B.2**

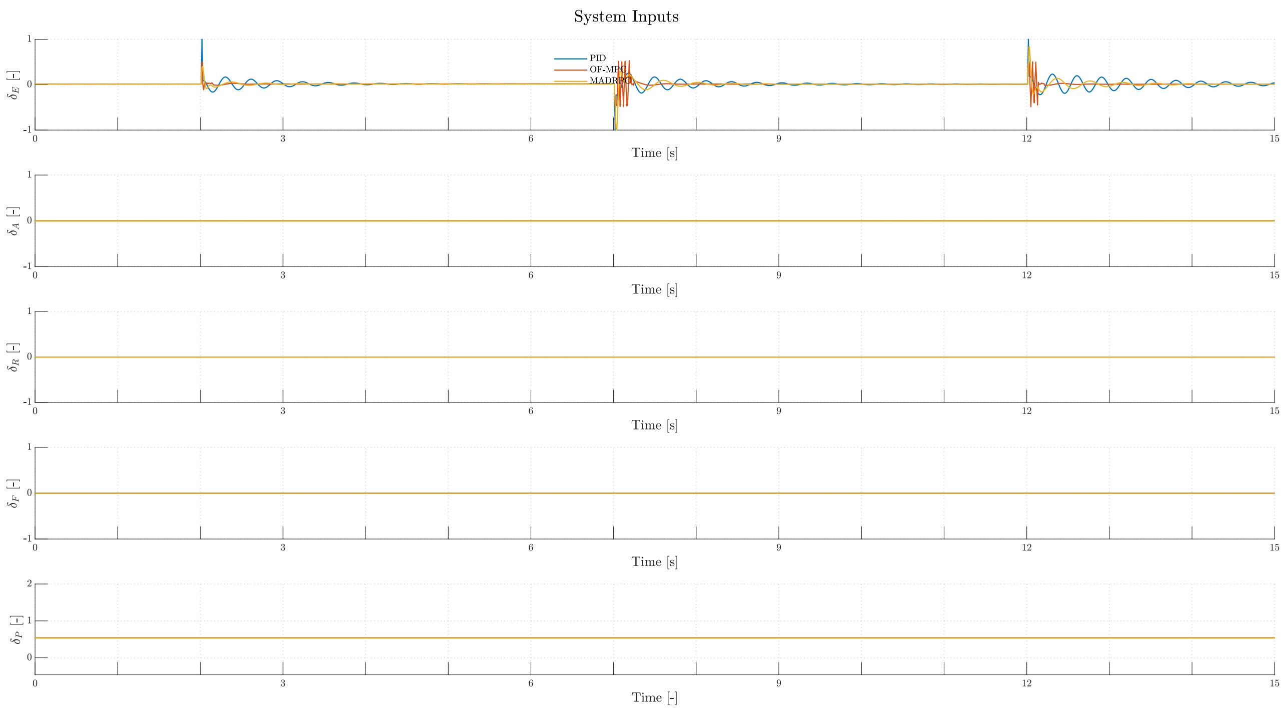


Figure 76: H200 system inputs during **Test B.2**

And the controllers performance indicators are:

	ITAE	ISE	IAE	MSE
PID	48.4655	0.10803	5.4492	0.0072021
OF-MPC	25.9953	0.10399	3.1413	0.0069326
MADRPC	39.4297	0.10215	4.4552	0.0068101

Table 24: Controller performance indicators for **Test B.2**

6.2.2.2.3 Test B.3: Underestimation of Propeller Coefficients

The next pair of tests will focus on the propeller coefficients modifications. In this one specifically, the focus will be the underestimation of the propeller coefficients. For this, it is going to be modified the engine thrust model, which relates the throttle level (using the non-dimensional propeller velocity) and the thrust force coefficient, given in equation (32).

Although all the models inside the propulsion model are calculated using experimental data, only the thrust force coefficient model will be modified, as the rest depend more on the actual electrical engine, and not the propulsion system as a whole.

For this, the modified thrust coefficients will be of the following form:

$$\begin{aligned} p_{C_{T_0},mod} &= p_{C_{T_0}} \cdot k \\ p_{C_{T_1},mod} &= p_{C_{T_1}} \cdot k \\ p_{C_{T_2},mod} &= p_{C_{T_2}} \cdot k \\ p_{C_{T_3},mod} &= p_{C_{T_3}} \cdot k \end{aligned} \tag{93}$$

where k is the modification parameter inside the propeller model. In this test, the modification parameter will have the value:

$$k = 0.5 \tag{94}$$

In this test, as the propeller model is independent of the pitch angle, the numerical values for the reference signal will be the same as the ones in table (13):

Symbol	Value	Units
T_1	2	s
T_2	7	s
T_3	12	s
θ_{trim}	2.5905	deg
θ_{ref_1}	3	deg
θ_{ref_2}	2	deg

Table 25: Reference signal parameters for **Test B.3**

So, the results may be seen from figure (77) to (85).

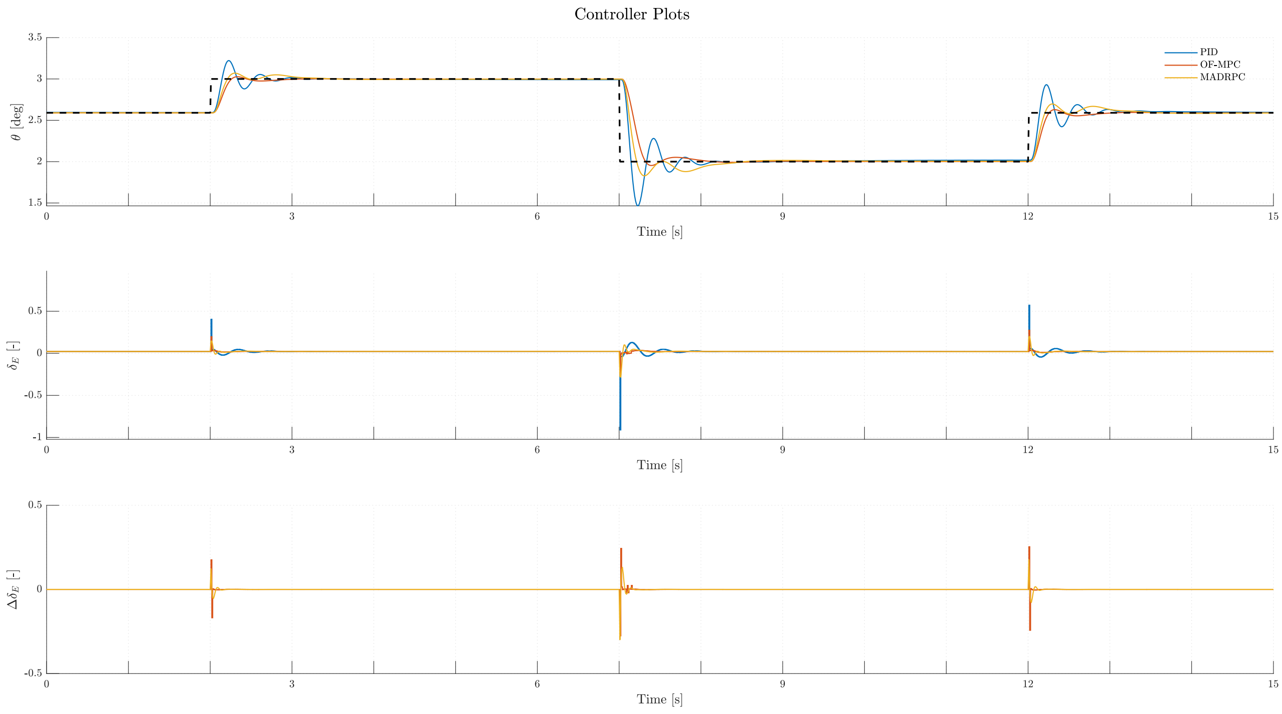


Figure 77: Controllers plots during **Test B.3**

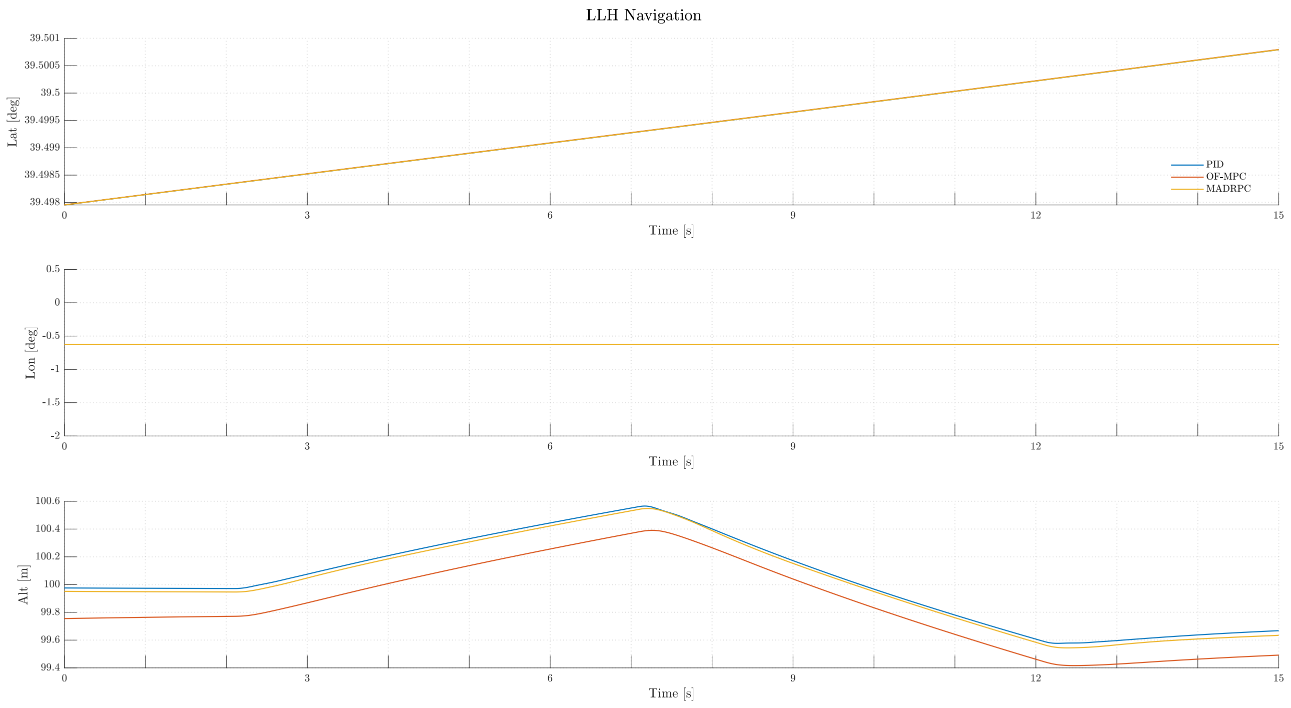


Figure 78: H200 LLH navigation during **Test B.3**

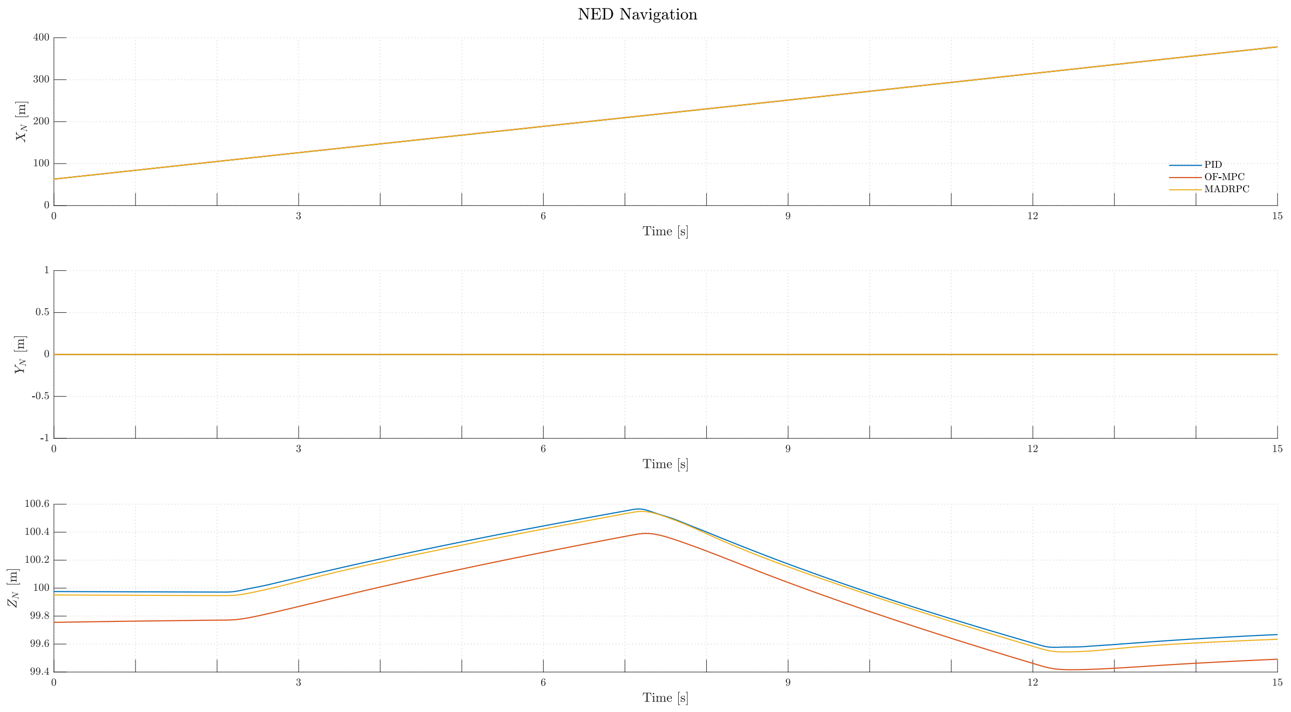


Figure 79: H200 NED navigation during **Test B.3**

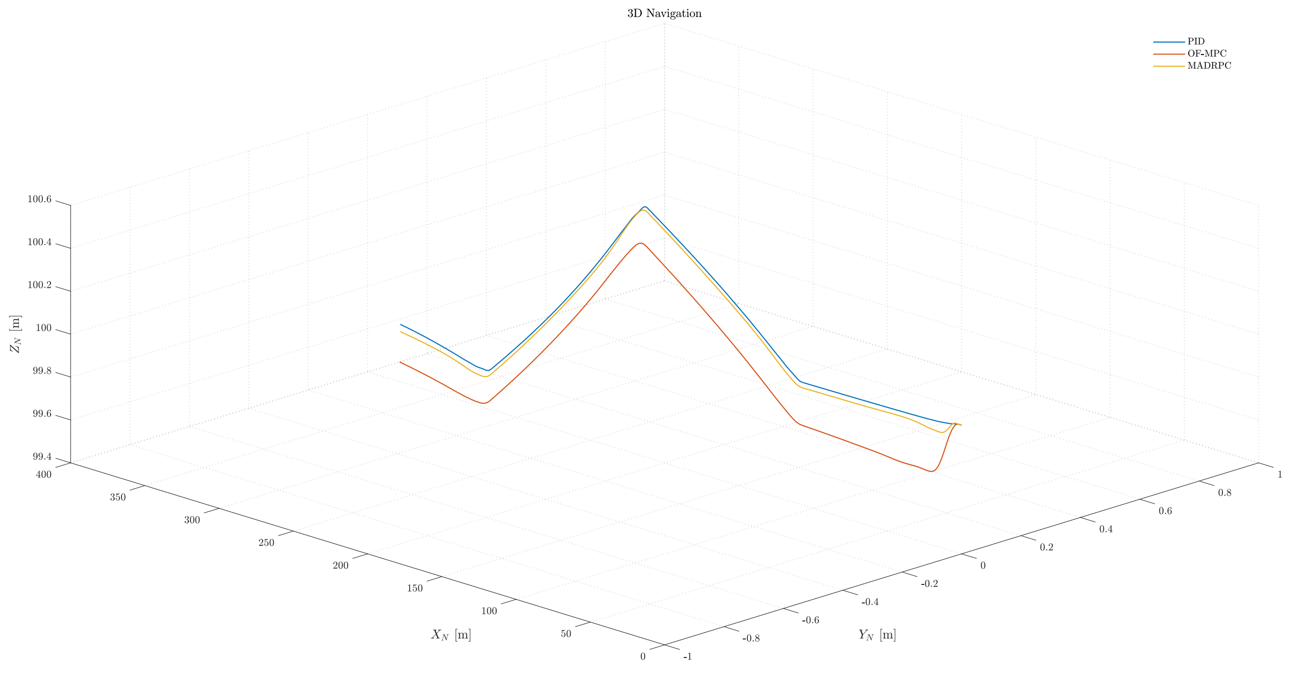


Figure 80: H200 3D navigation during **Test B.3**

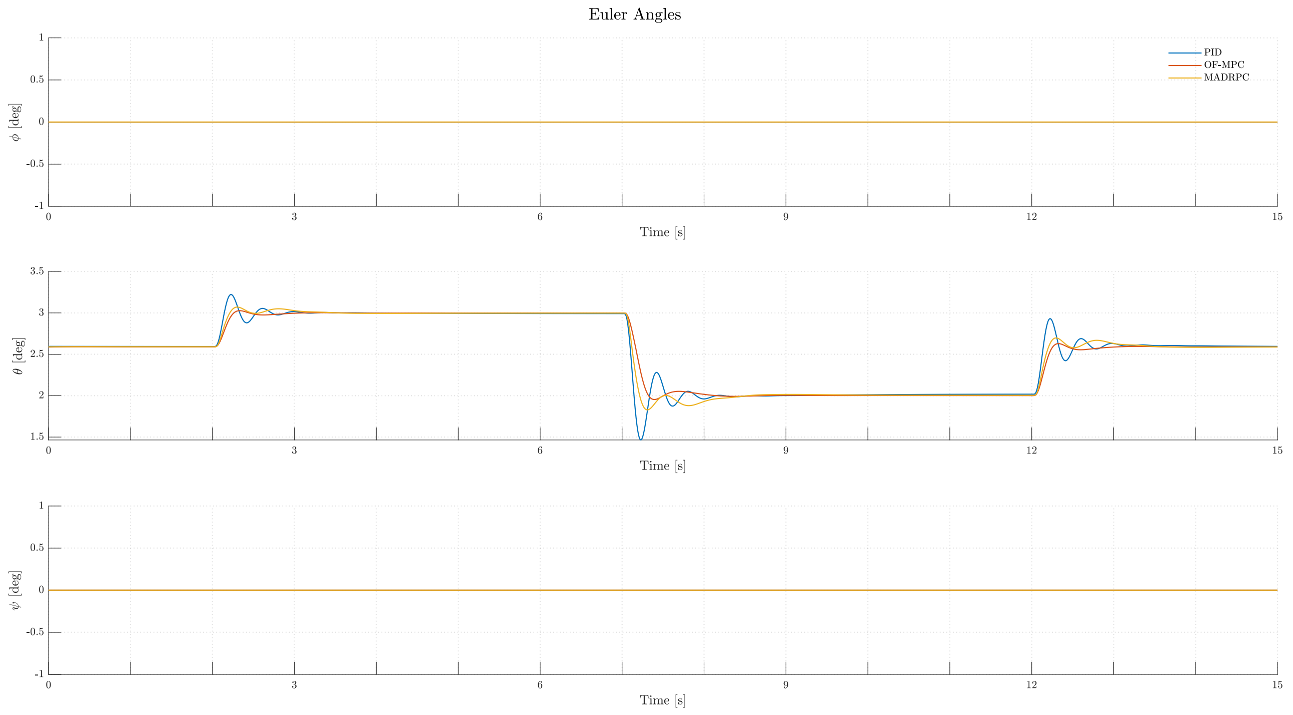


Figure 81: H200 Euler angles during **Test B.3**

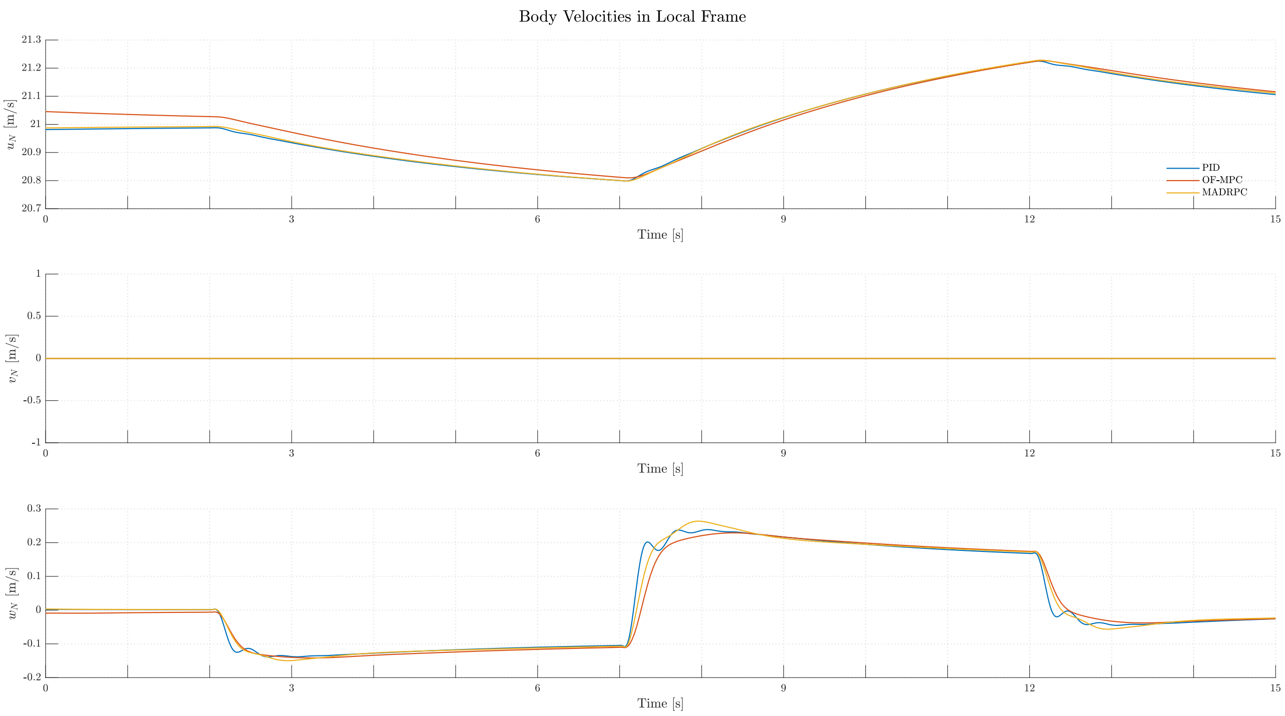


Figure 82: H200 body velocities expressed in local frame during **Test B.3**

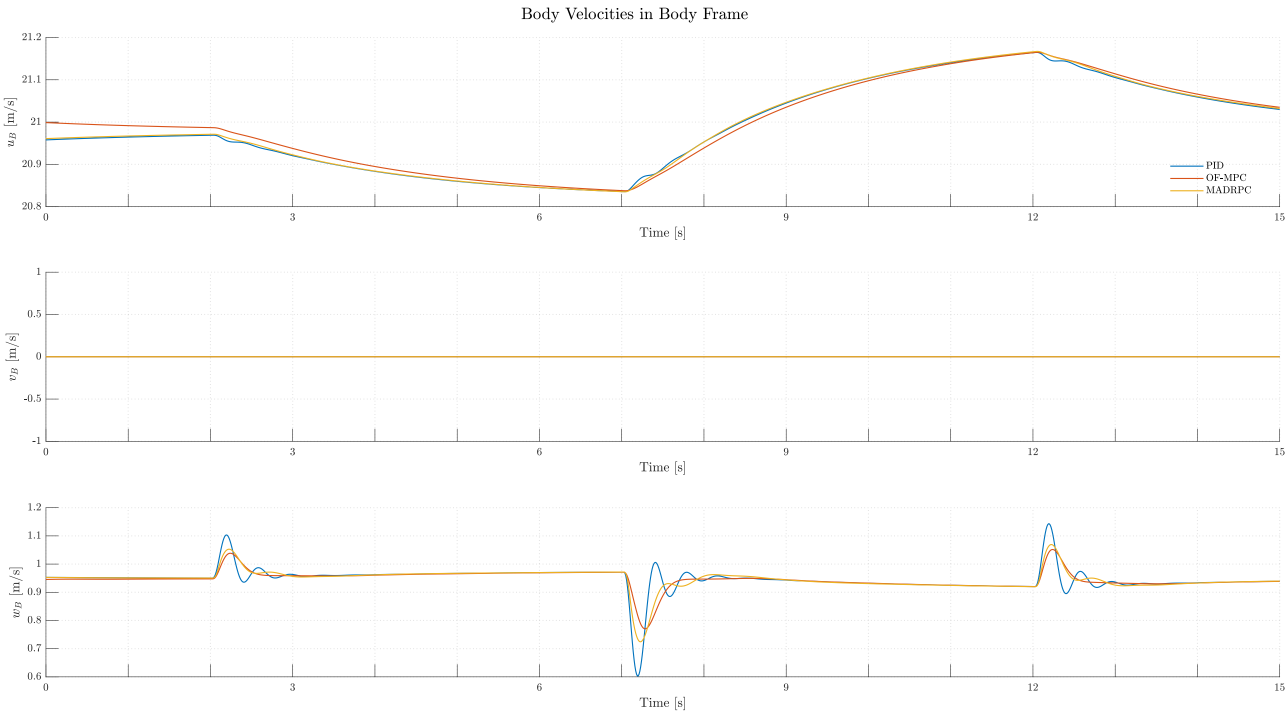


Figure 83: H200 body velocities expressed in body frame for **Test B.3**

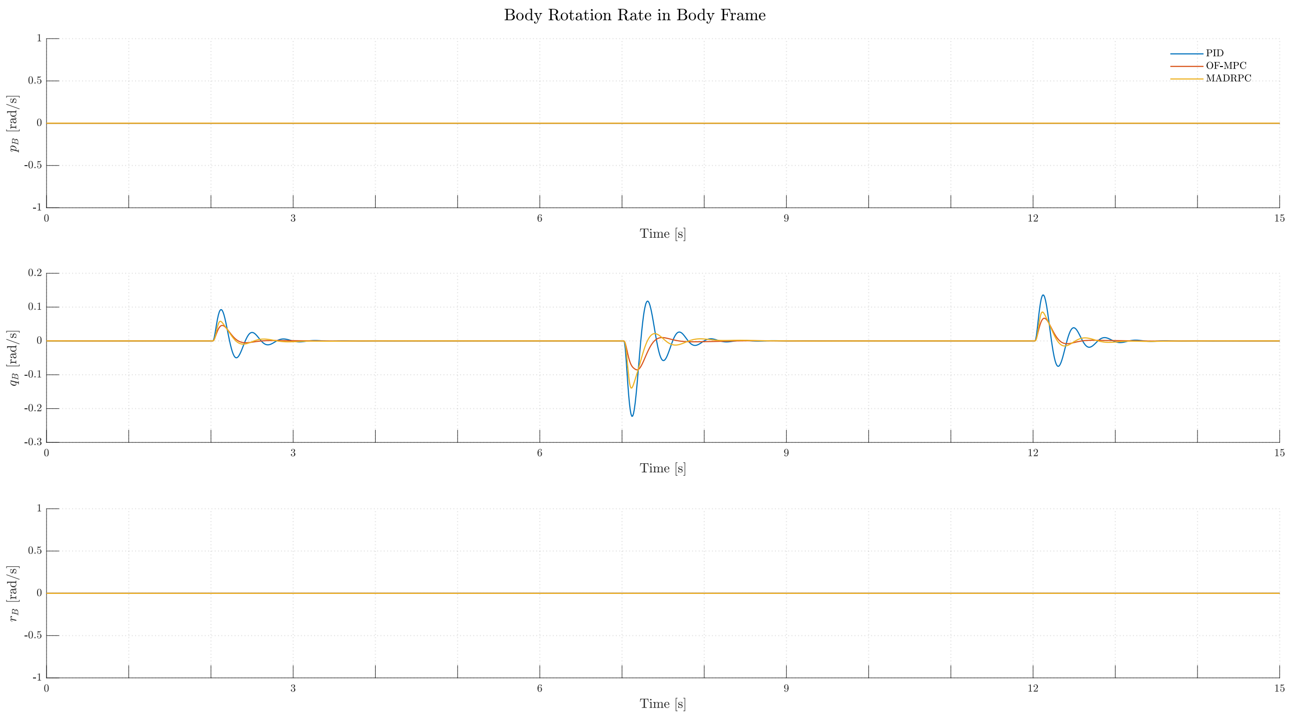


Figure 84: H200 body rotations during **Test B.3**

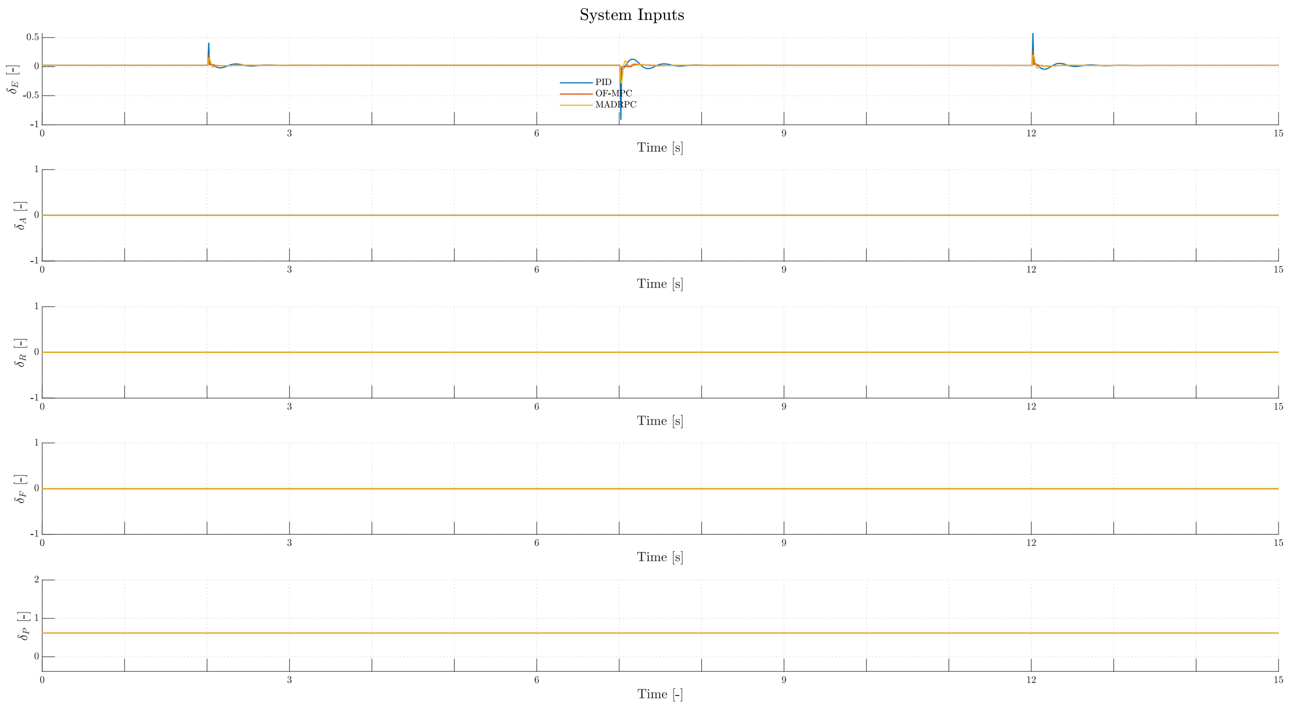


Figure 85: H200 system inputs during **Test B.3**

And the controllers performance indicators are:

	ITAE	ISE	IAE	MSE
PID	7.6248	0.005492	0.95055	0.00036613
OF-MPC	5.246	0.0060625	0.68858	0.00040416
MADRPC	6.412	0.005074	0.8075	0.00033827

Table 26: Controller performance indicators for **Test B.3**

6.2.2.2.4 Test B.4: Overestimation of Propeller Coefficients

The next test will focus on the overestimation of the propeller coefficients. The test will be carried out as **Test B.3**, where the thrust force model parameters will be ones modified. For this, the modified coefficients will have the following form:

$$\begin{aligned} p_{C_{T_0,mod}} &= p_{C_{T_0}} \cdot k \\ p_{C_{T_1,mod}} &= p_{C_{T_1}} \cdot k \\ p_{C_{T_2,mod}} &= p_{C_{T_2}} \cdot k \\ p_{C_{T_3,mod}} &= p_{C_{T_3}} \cdot k \end{aligned} \tag{95}$$

where k is the modification parameter inside the propeller model. In this test, the modification parameter will have the value:

$$k = 1.5 \tag{96}$$

Once again, in this test, the propeller model is independent of the pitch angle, the numerical values for the reference signal will be the same as the ones in table (13):

Symbol	Value	Units
T_1	2	s
T_2	7	s
T_3	12	s
θ_{trim}	2.5905	deg
θ_{ref_1}	3	deg
θ_{ref_2}	2	deg

Table 27: Reference signal parameters for **Test B.4**

So, the results may be seen from figure (86) to (94).

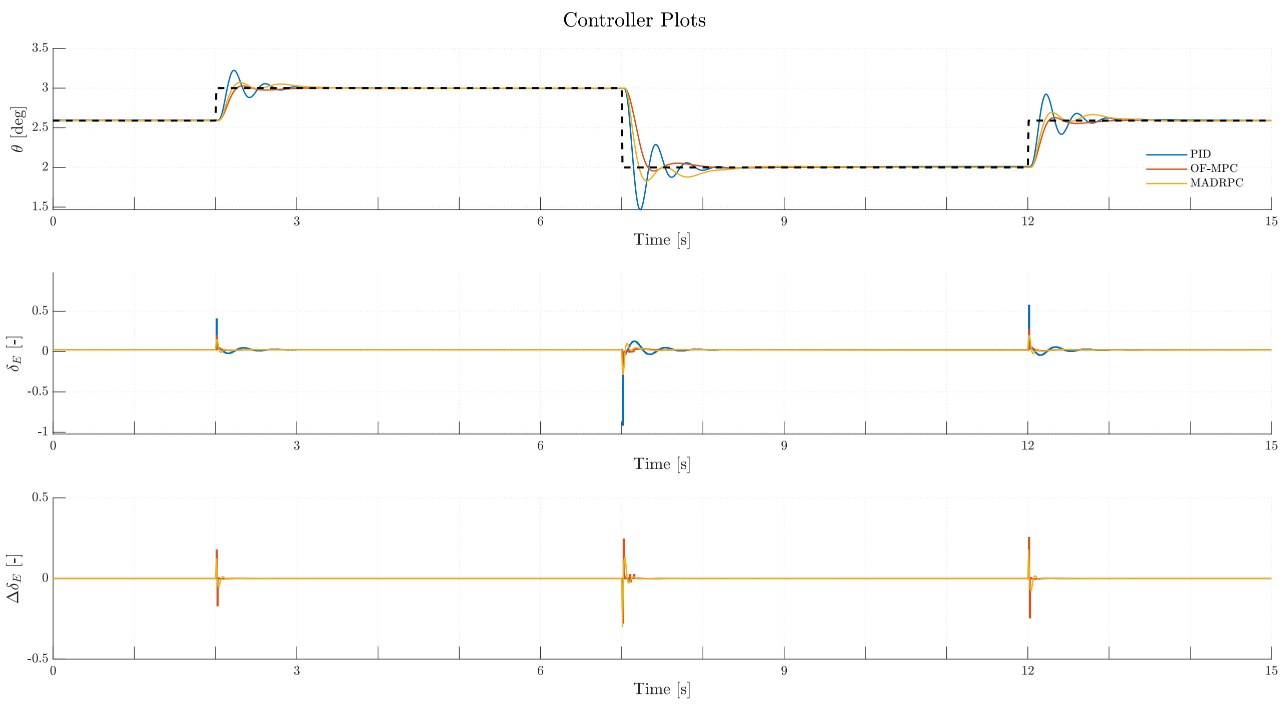


Figure 86: Controllers plots during **Test B.4**

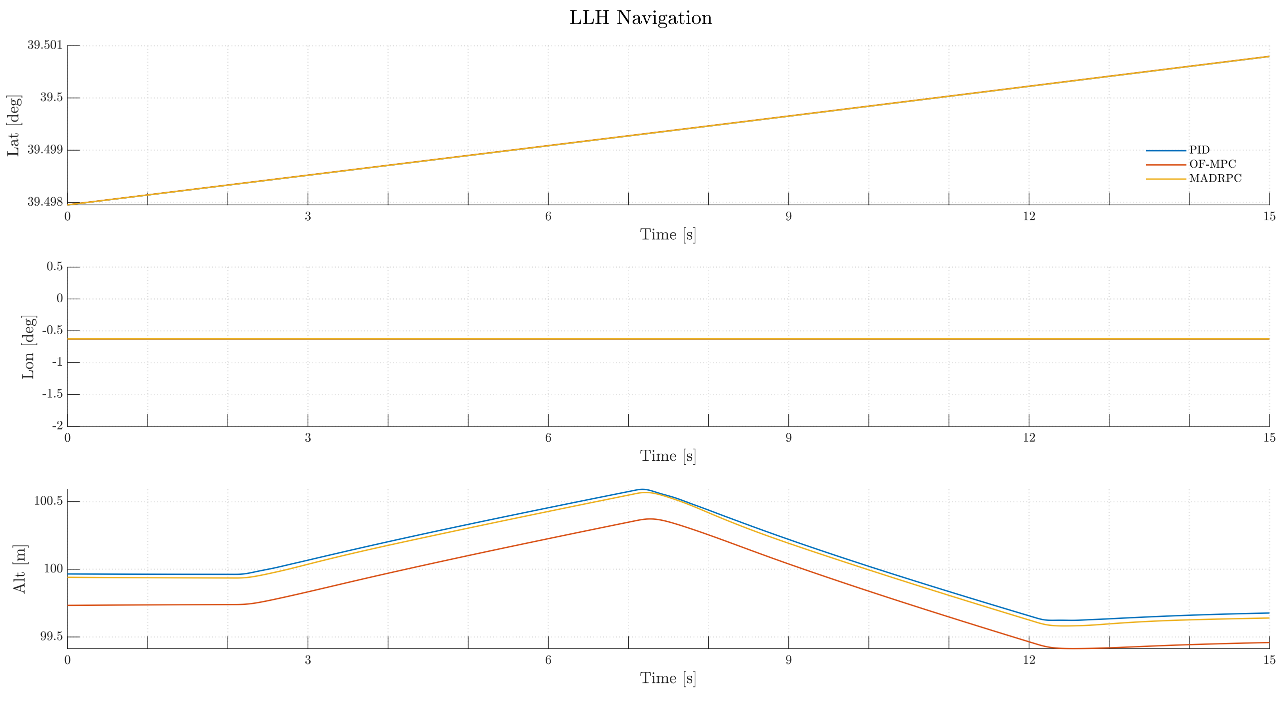


Figure 87: H200 LLH navigation during **Test B.4**

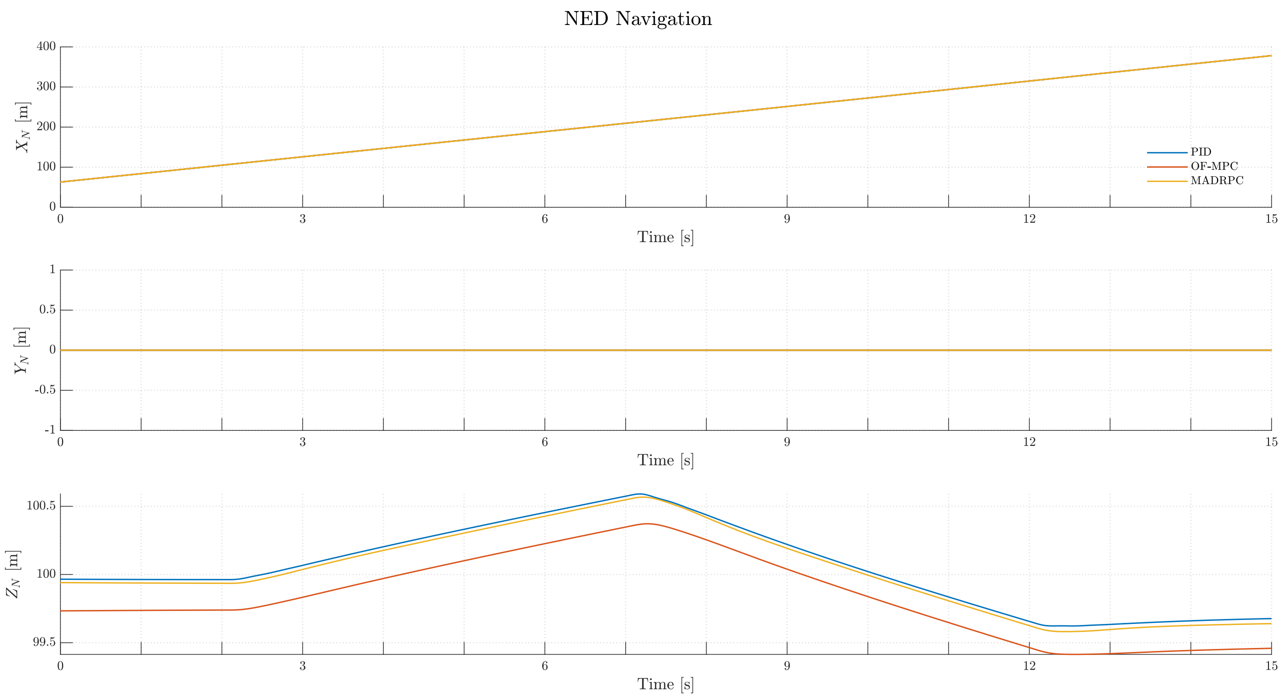


Figure 88: H200 NED navigation during **Test B.4**

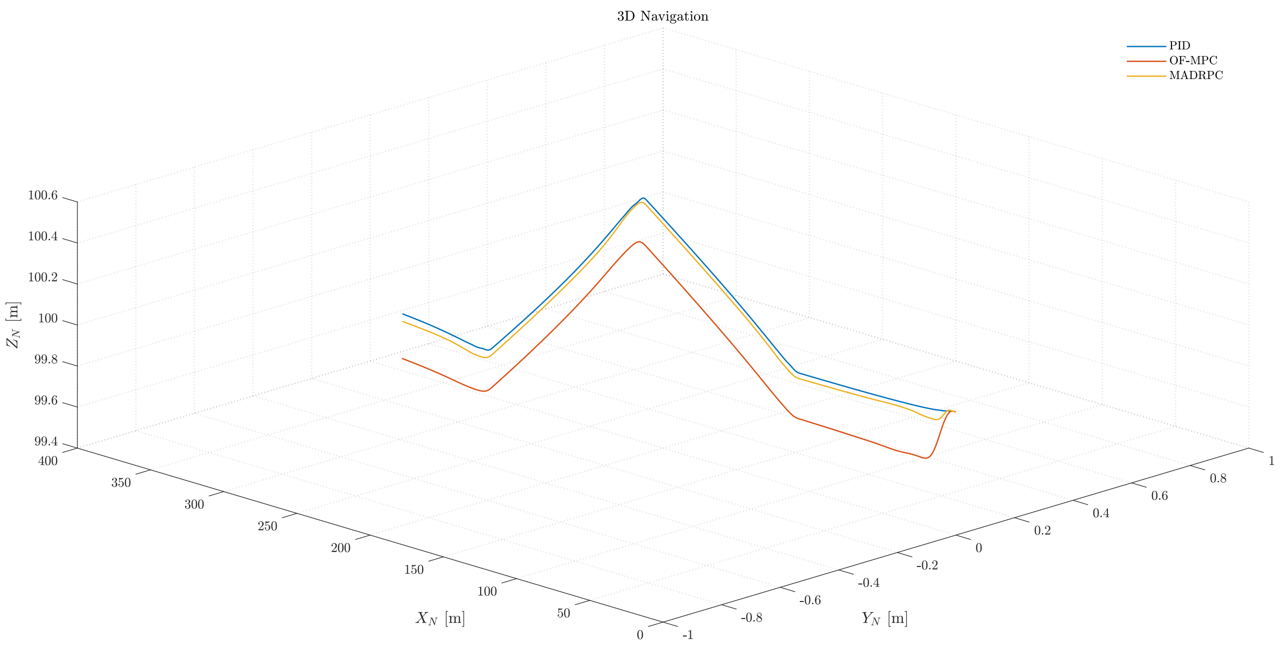


Figure 89: H200 3D navigation during **Test B.4**

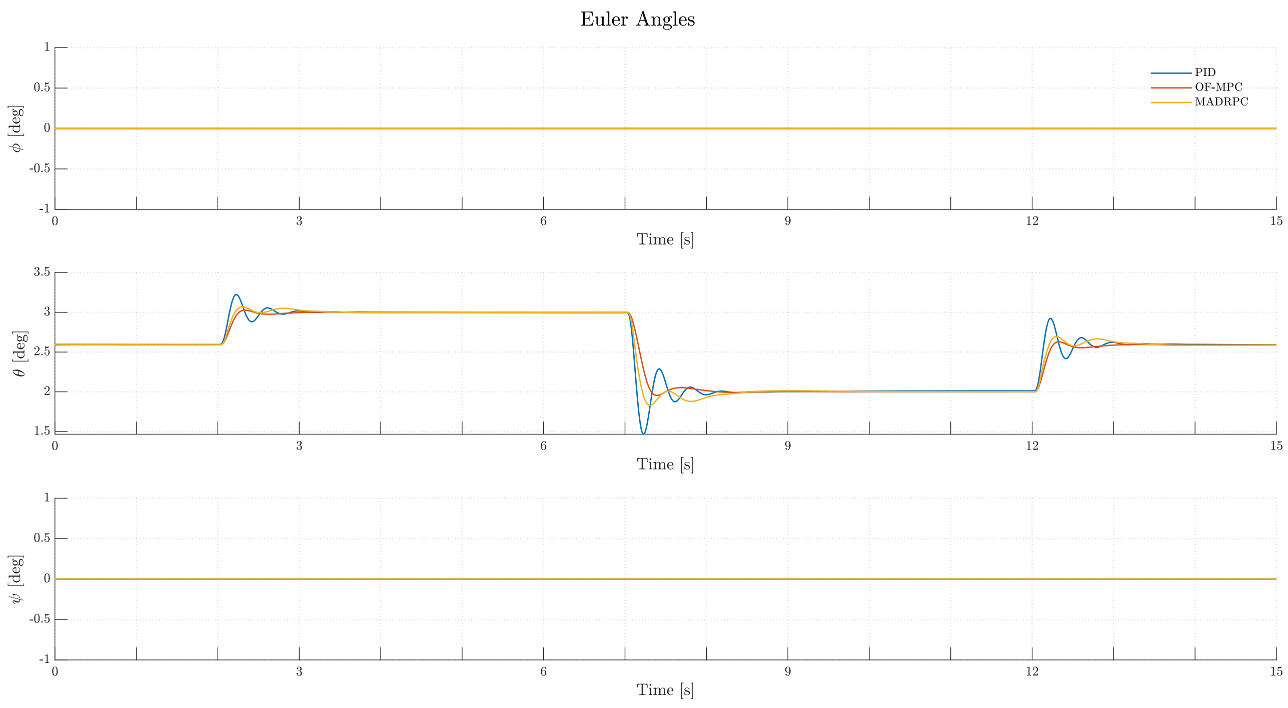


Figure 90: H200 Euler angles during **Test B.4**

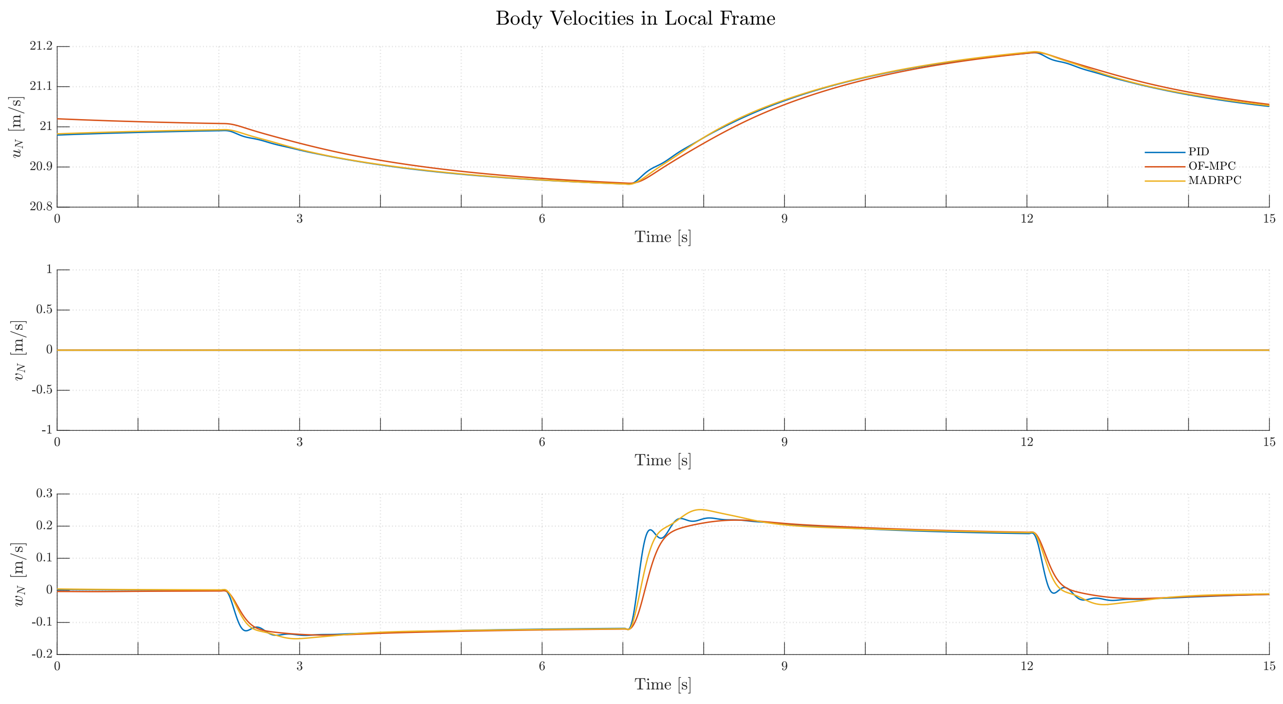


Figure 91: H200 body velocities expressed in local frame during **Test B.4**

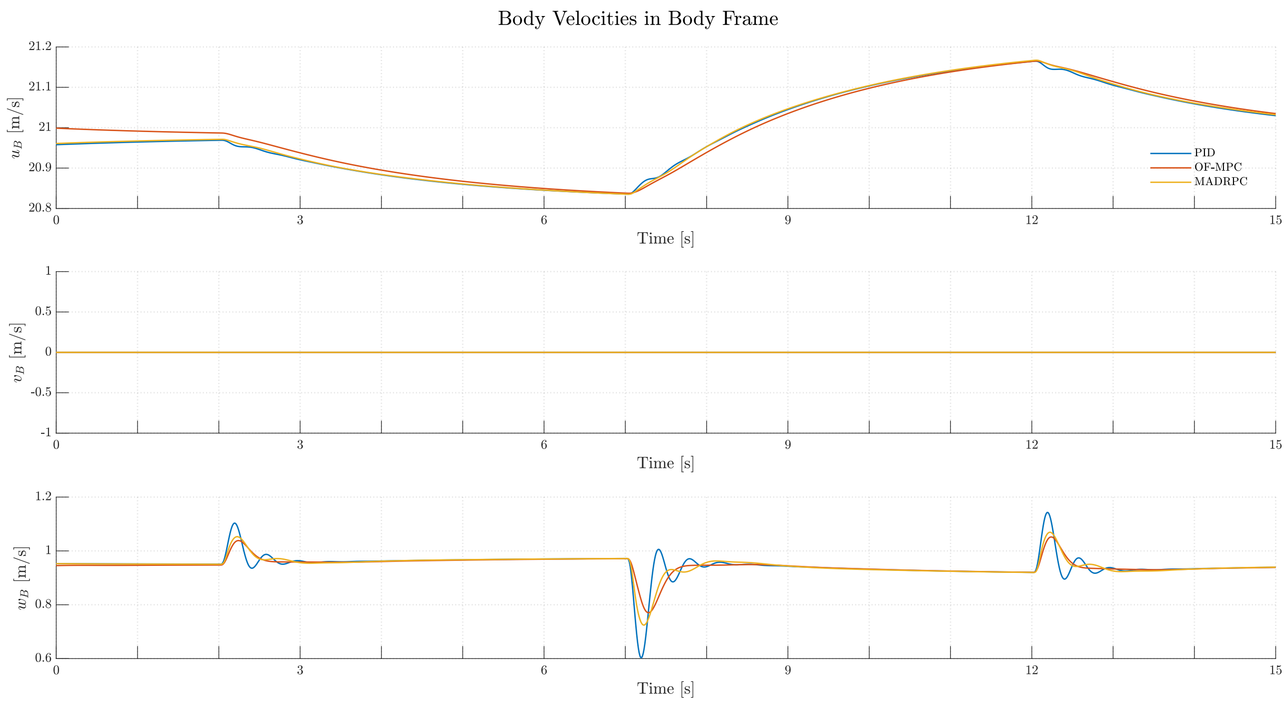


Figure 92: H200 body velocities expressed in body frame for **Test B.4**

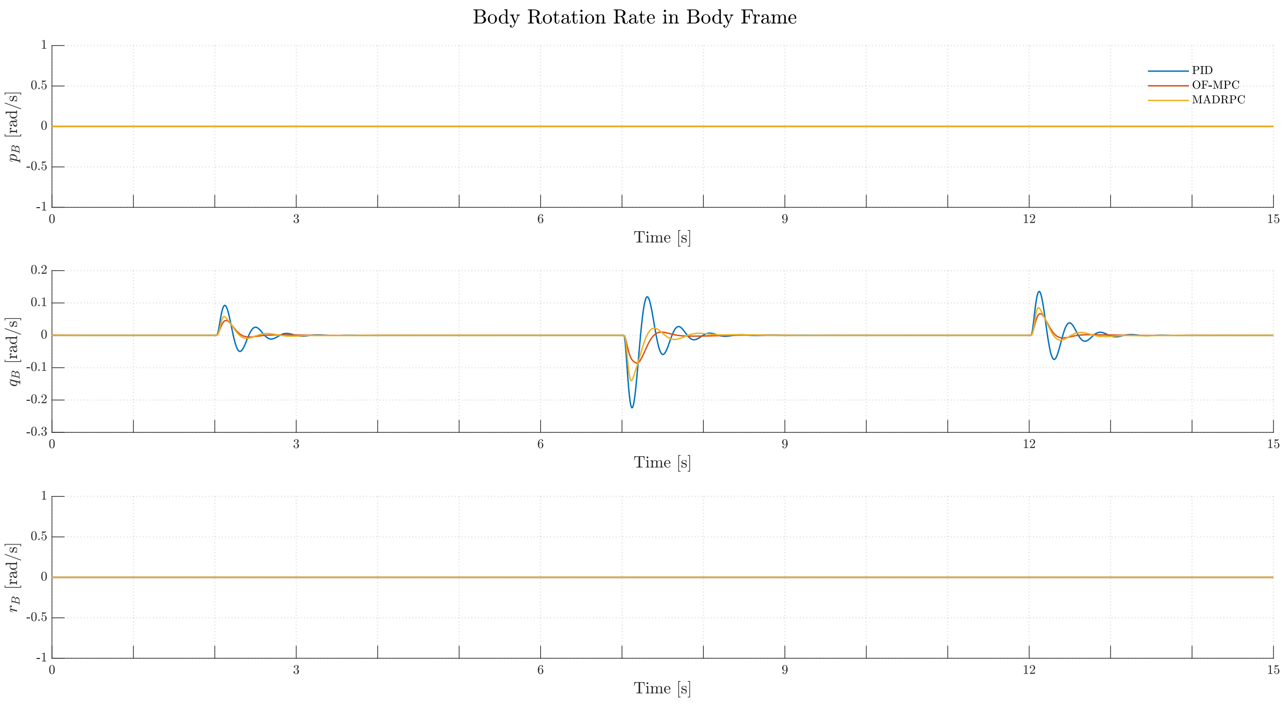


Figure 93: H200 body rotations during **Test B.4**

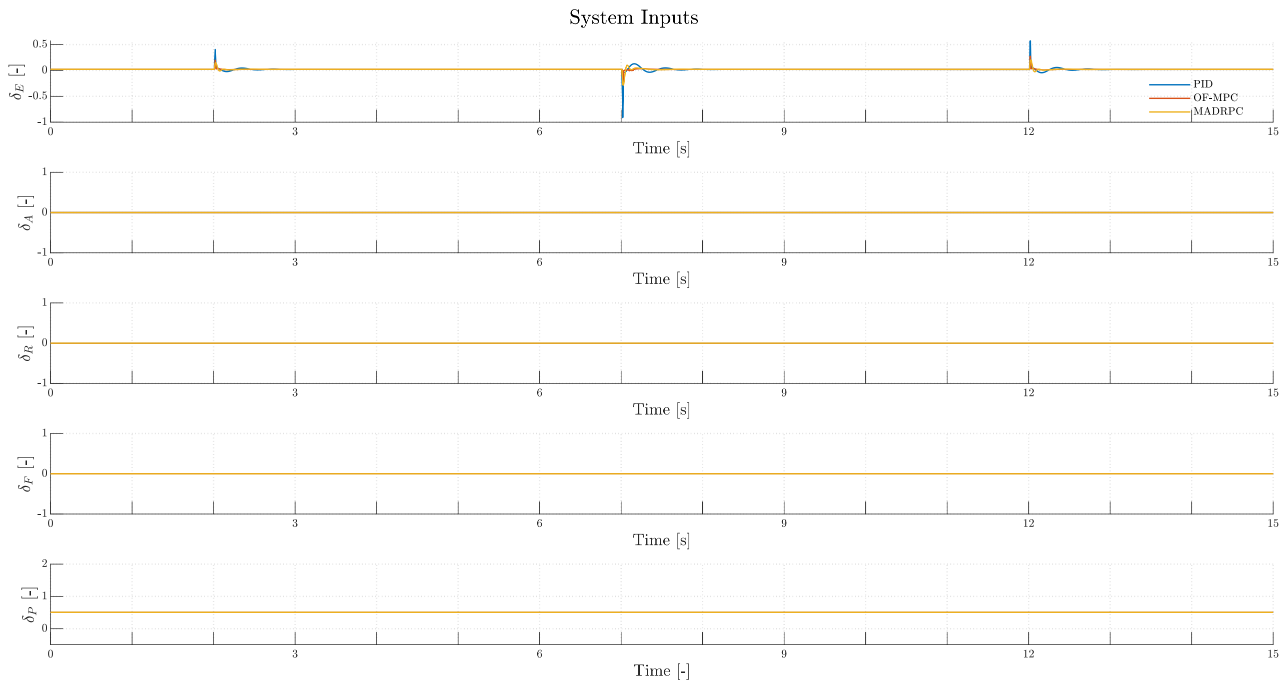


Figure 94: H200 system inputs during **Test B.4**

And the controllers performance indicators are:

	ITAE	ISE	IAE	MSE
PID	7.0476	0.0055024	0.89927	0.00036682
OF-MPC	5.1929	0.0060646	0.68181	0.00040431
MADRPC	6.2085	0.0050755	0.768	0.00033837

Table 28: Controller performance indicators for **Test B.4**

6.2.2.2.5 Test B.5: Flight with Higher Configuration Mass

The last pair of test will focus on the H200 flight mass. The H200 must be capable of flying with some payload weight and drop it at some point of its flight mission. These tests will check the correct controllers performance inside the operative range for the aircraft mass. Specifically, this test will focus on increasing the aircraft mass compared with the nominal configuration mass:

$$m_{aircraft} = 25\,\text{kg} \tag{97}$$

As it happened with the flight condition tests and the aerodynamic model tests, a change in the H200 mass will affect the trimming value for the pitch angle, as it has to compensate the increase in mass with more aerodynamic lift, meaning more pitch angle for flying in steady conditions:

$$\theta_{trim} = 6.5790\,\text{deg} \tag{98}$$

So, the reference pitch angle signal will be:

Symbol	Value	Units
T_1	2	s
T_2	7	s
T_3	12	s
θ_{trim}	6.5790	deg
θ_{ref_1}	8	deg
θ_{ref_2}	5	deg

Table 29: Reference signal parameters for **Test B.5**

So, the results may be seen from figure (95) to (103).

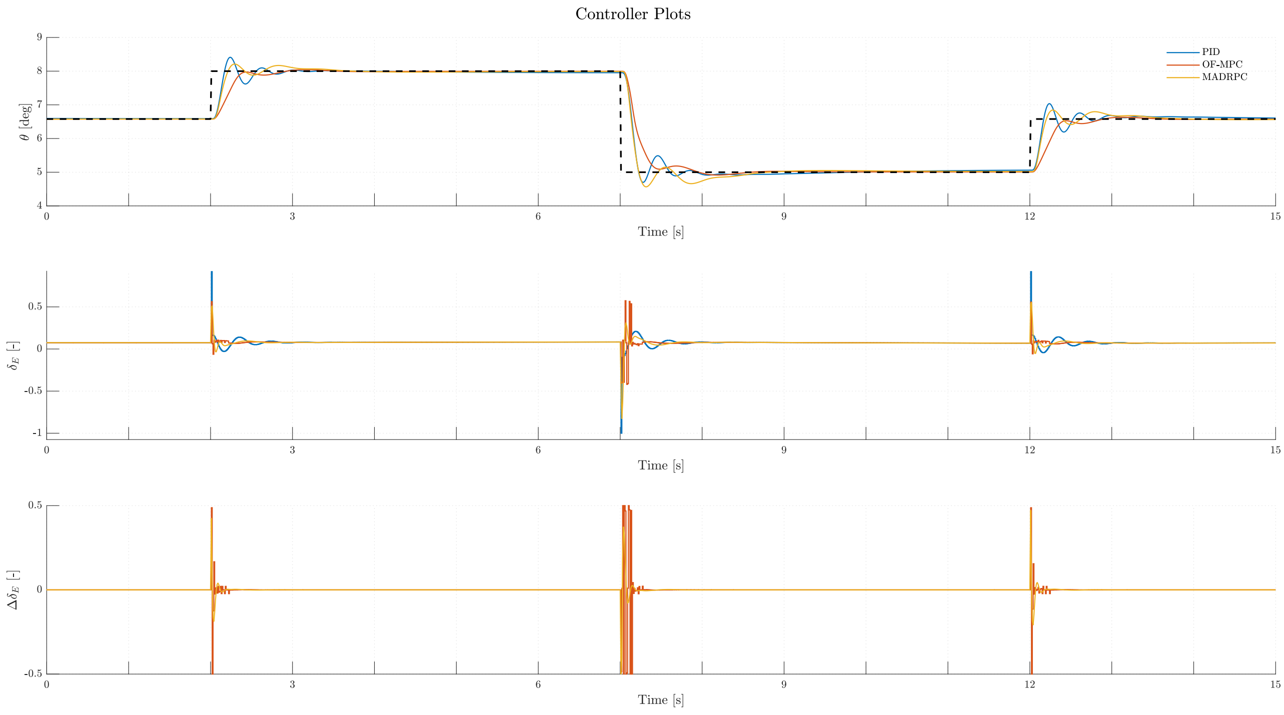


Figure 95: Controllers plots during **Test B.5**

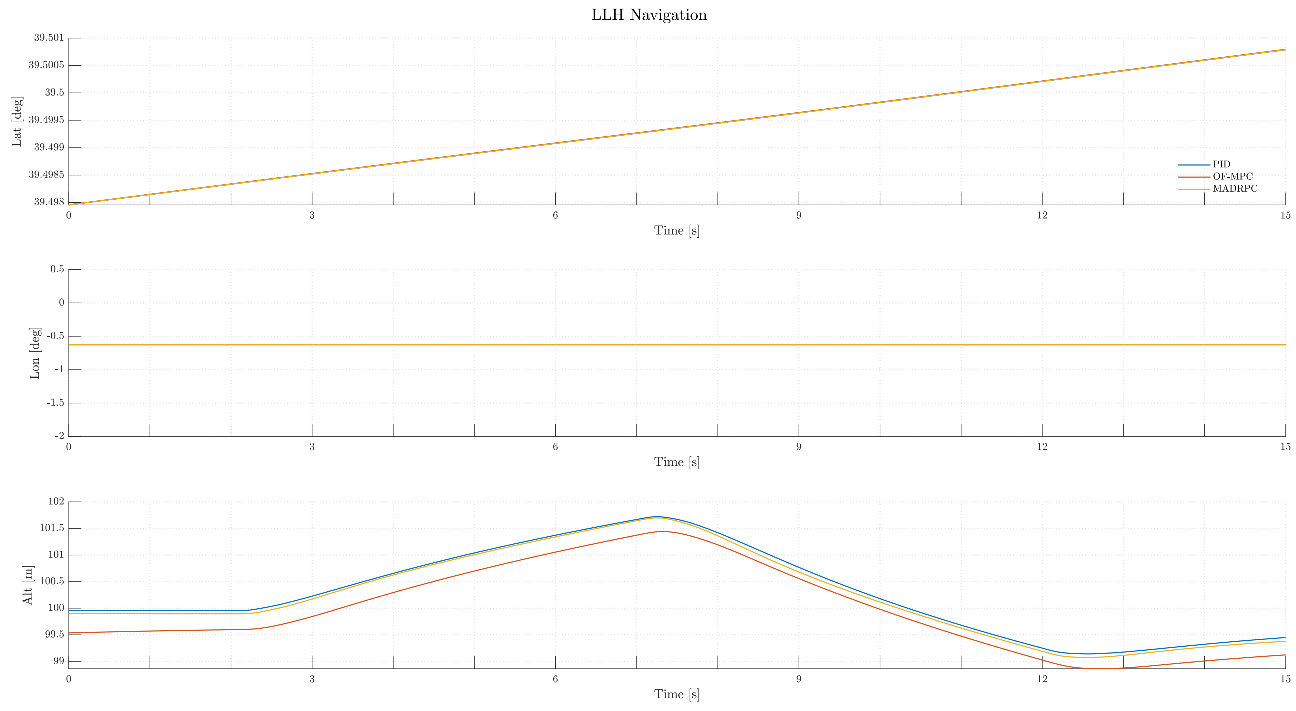


Figure 96: H200 LLH navigation during **Test B.5**

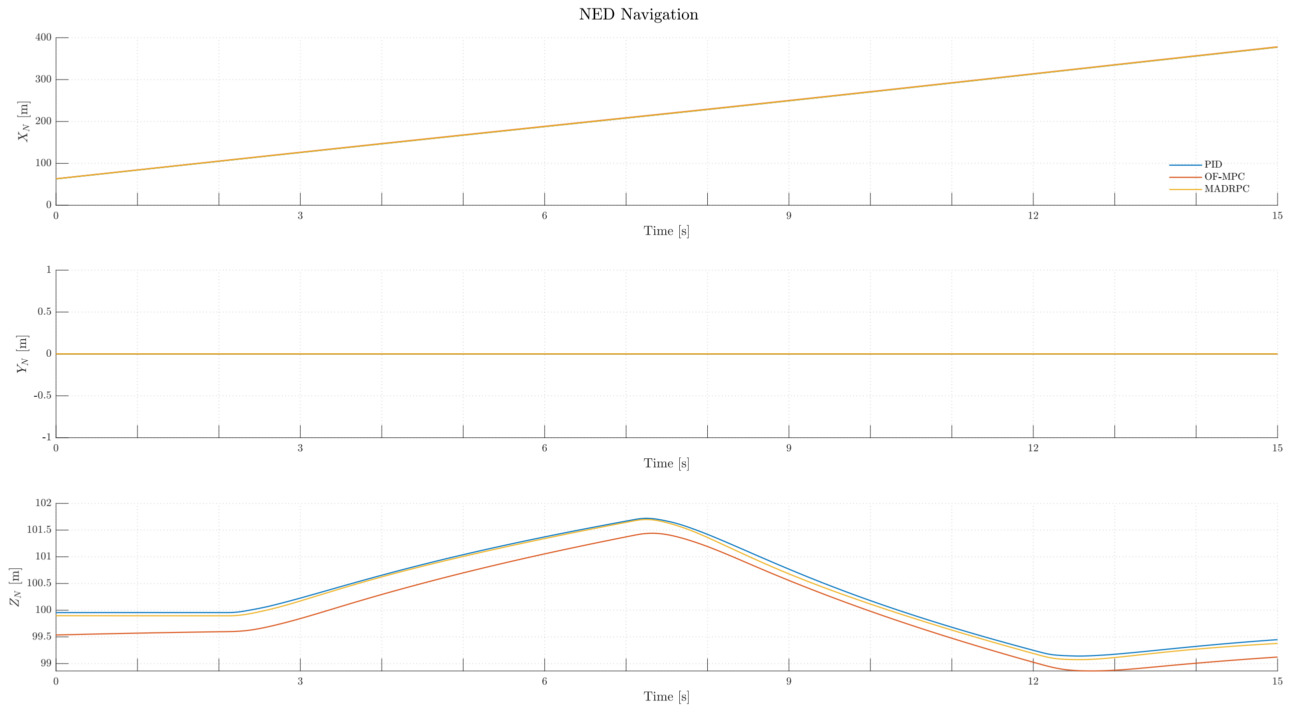


Figure 97: H200 NED navigation during **Test B.5**

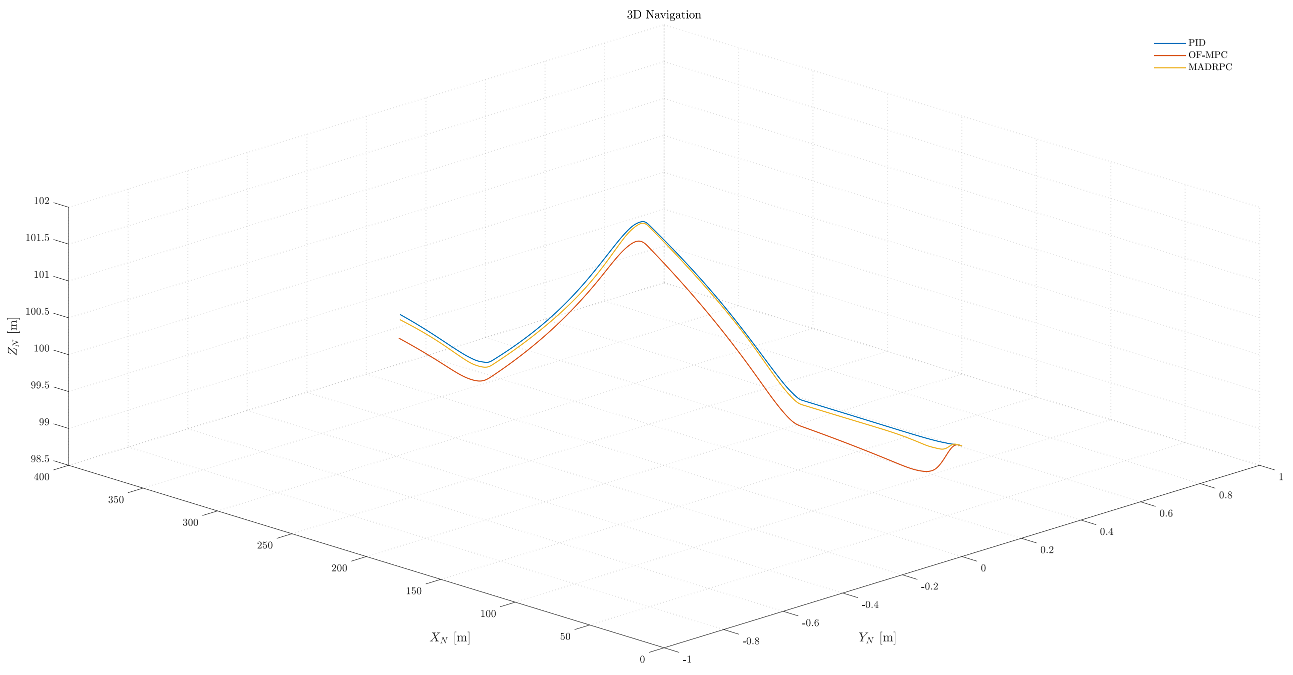


Figure 98: H200 3D navigation during **Test B.5**

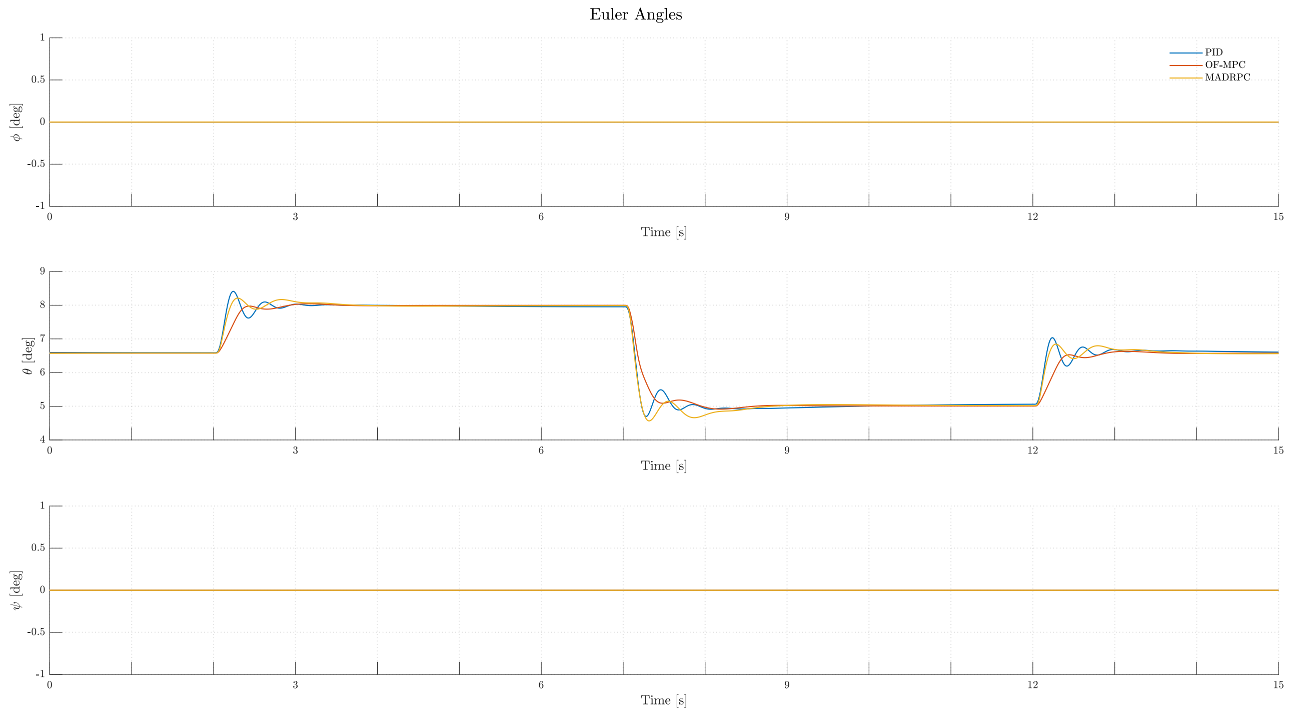


Figure 99: H200 Euler angles during **Test B.5**

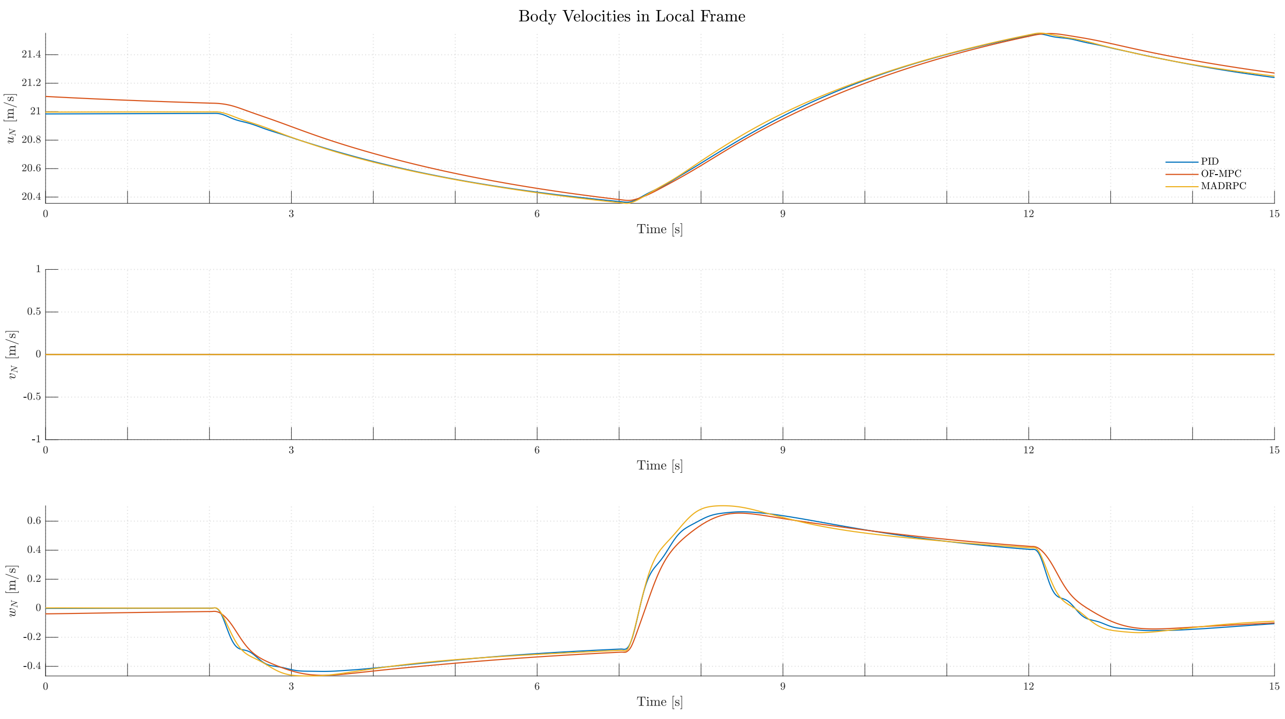


Figure 100: H200 body velocities expressed in local frame during **Test B.5**

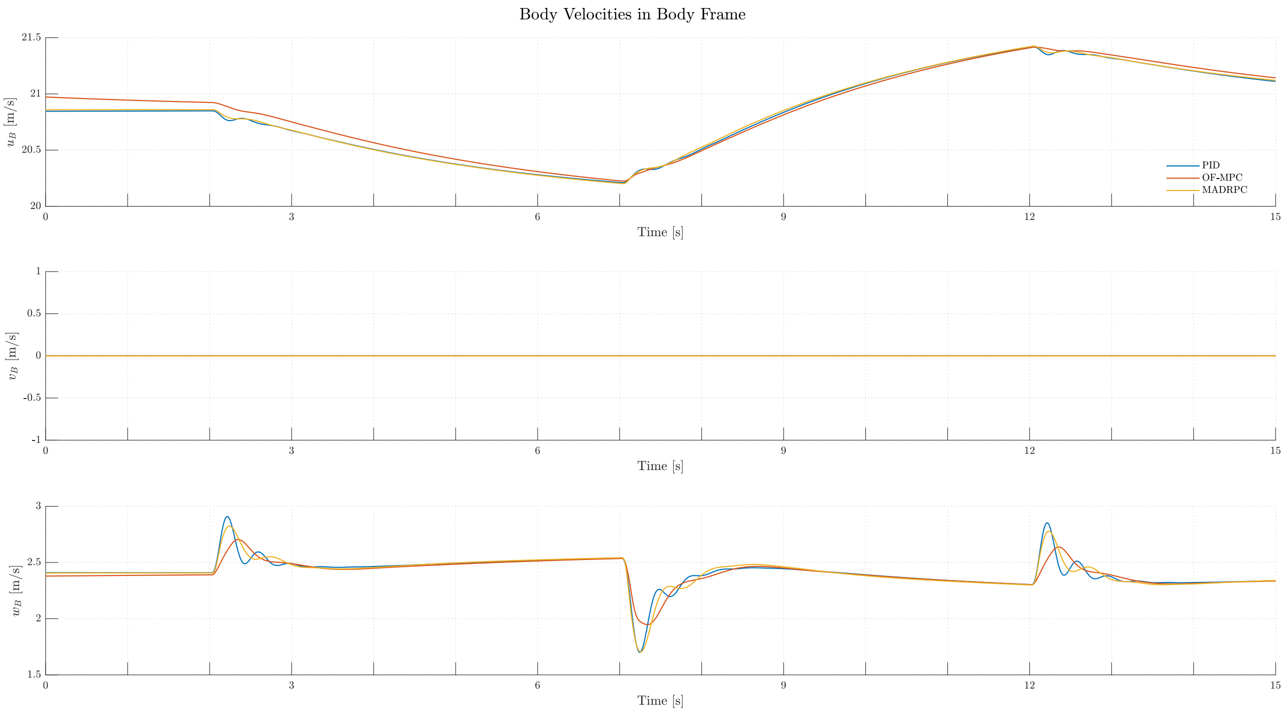


Figure 101: H200 body velocities expressed in body frame for **Test B.5**

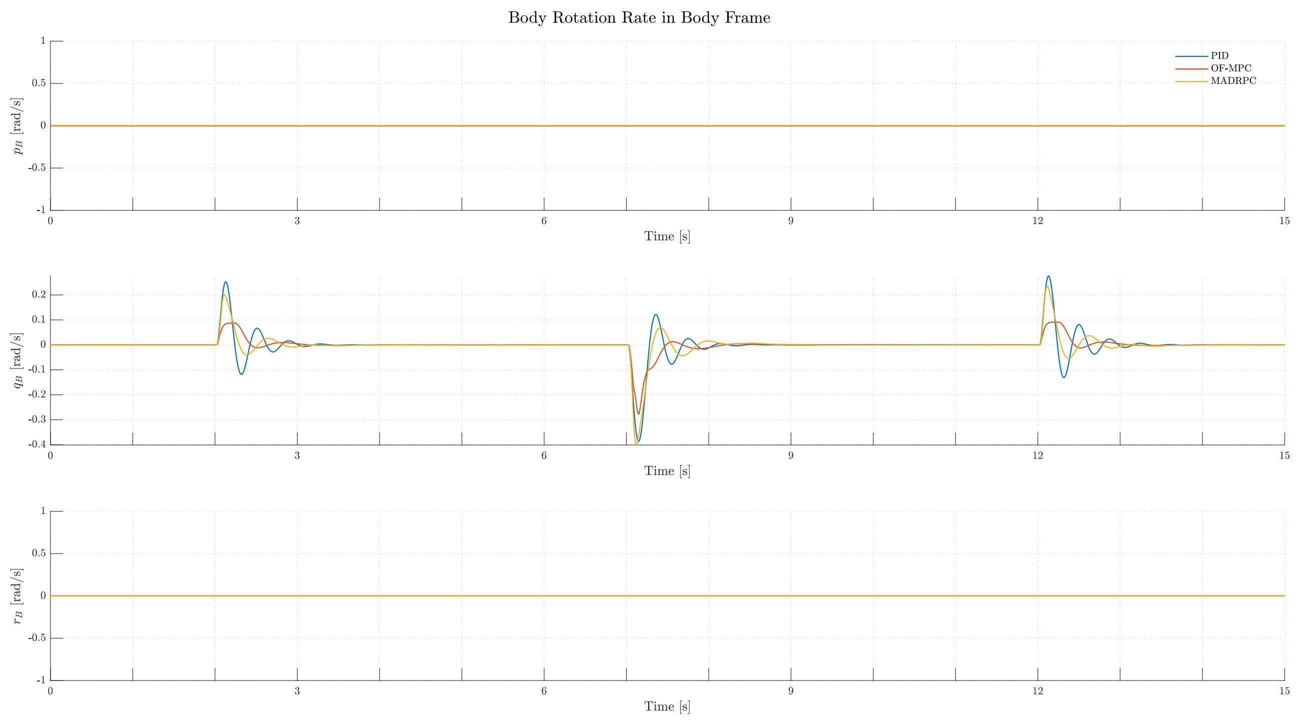


Figure 102: H200 body rotations during **Test B.5**

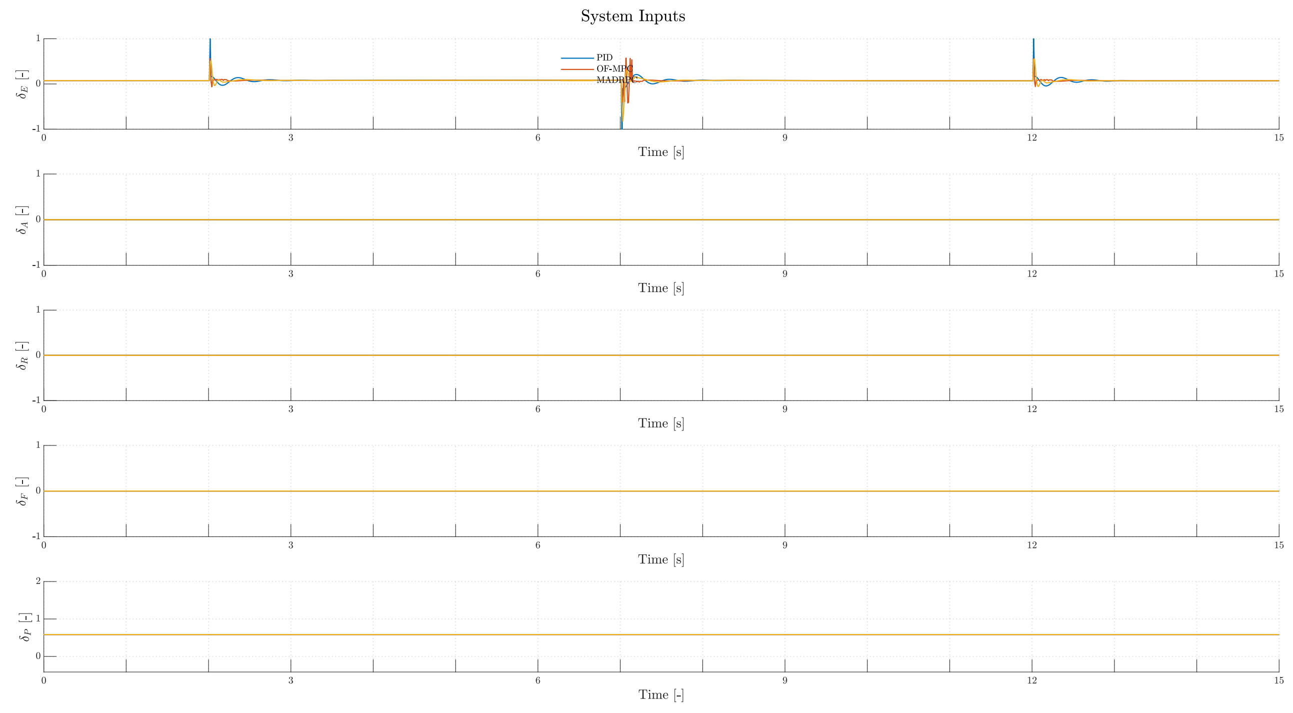


Figure 103: H200 system inputs during **Test B.5**

And the controllers performance indicators are:

	ITAE	ISE	IAE	MSE
PID	20.7271	0.045976	2.6398	0.003065
OF-MPC	19.8988	0.061871	2.6601	0.0041247
MADRPC	20.3336	0.0465	2.6259	0.0031

Table 30: Controller performance indicators for **Test B.5**

6.2.2.2.6 Test B.6: Flight with Lower Configuration Mass

This last test will focus on decreasing the aircraft mass compared with the nominal configuration mass:

$$m_{aircraft} = 5\,\text{kg} \tag{99}$$

Once again, as with the other tests, a change in the H200 mass will affect the trimming value for the pitch angle, as it has to compensate the decrease in mass with less aerodynamic lift, meaning less pitch angle for flying in steady conditions:

$$\theta_{trim} = -1.4435\,\text{deg} \tag{100}$$

So, the reference pitch angle signal will be:

Symbol	Value	Units
T_1	2	s
T_2	7	s
T_3	12	s
θ_{trim}	-1.4435	deg
θ_{ref_1}	1	deg
θ_{ref_2}	-3	deg

Table 31: Reference signal parameters for **Test B.6**

So, the results may be seen from figure (104) to (112).

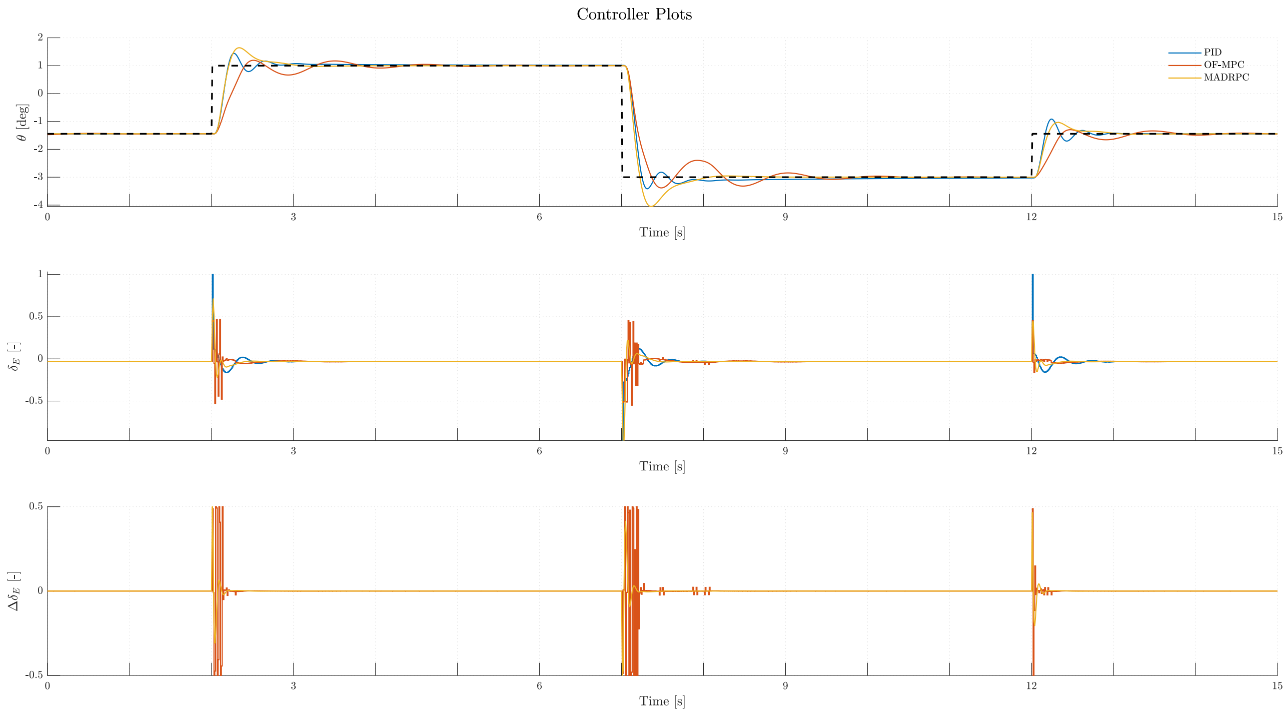


Figure 104: Controllers plots during **Test B.6**

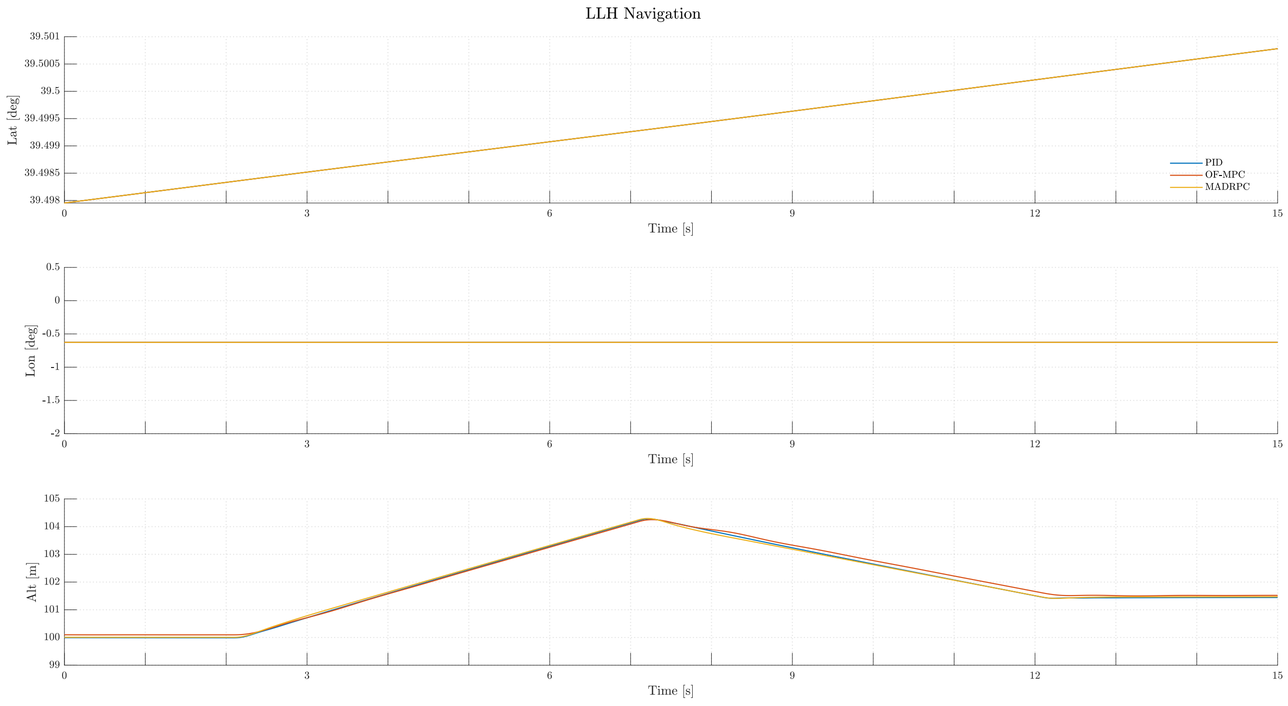


Figure 105: H200 LLH navigation during **Test B.6**

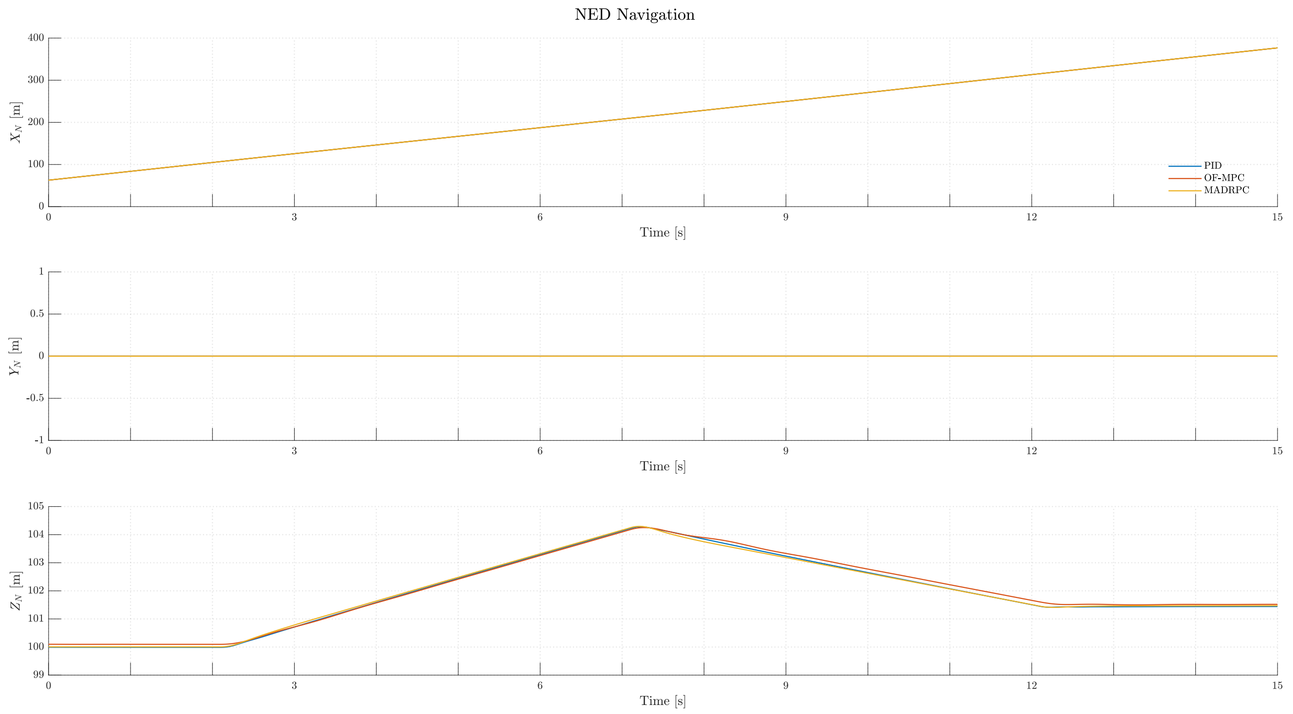


Figure 106: H200 NED navigation during **Test B.6**

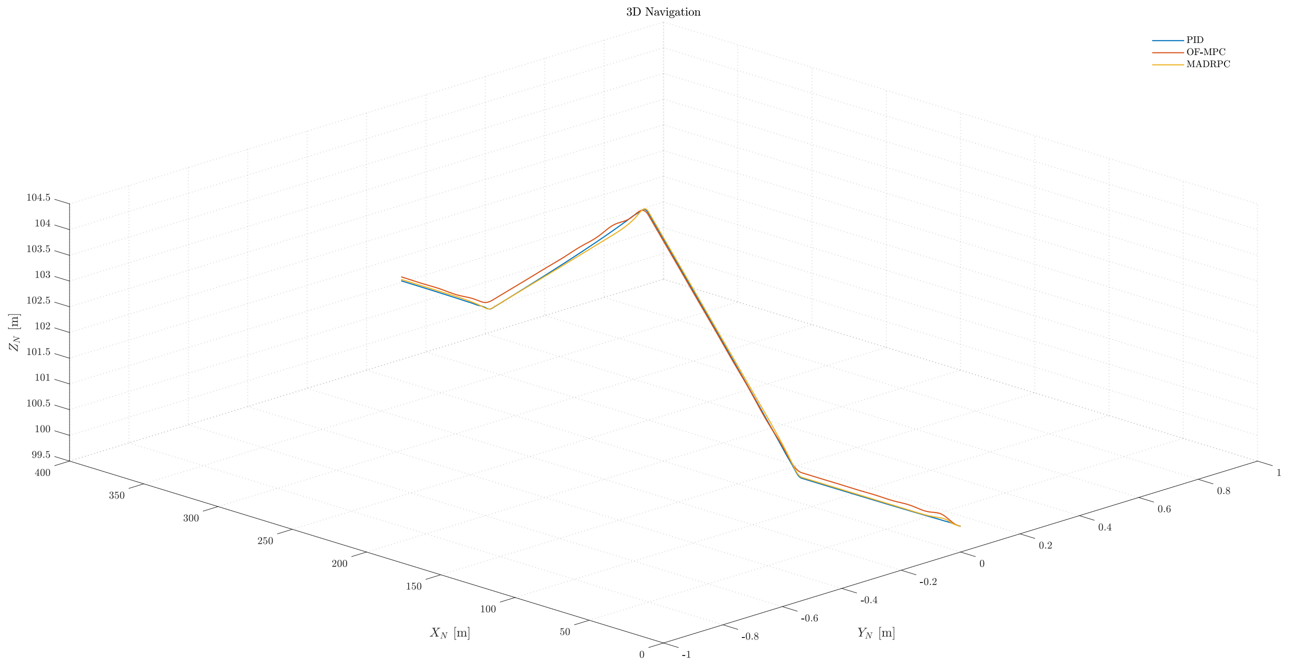


Figure 107: H200 3D navigation during **Test B.6**

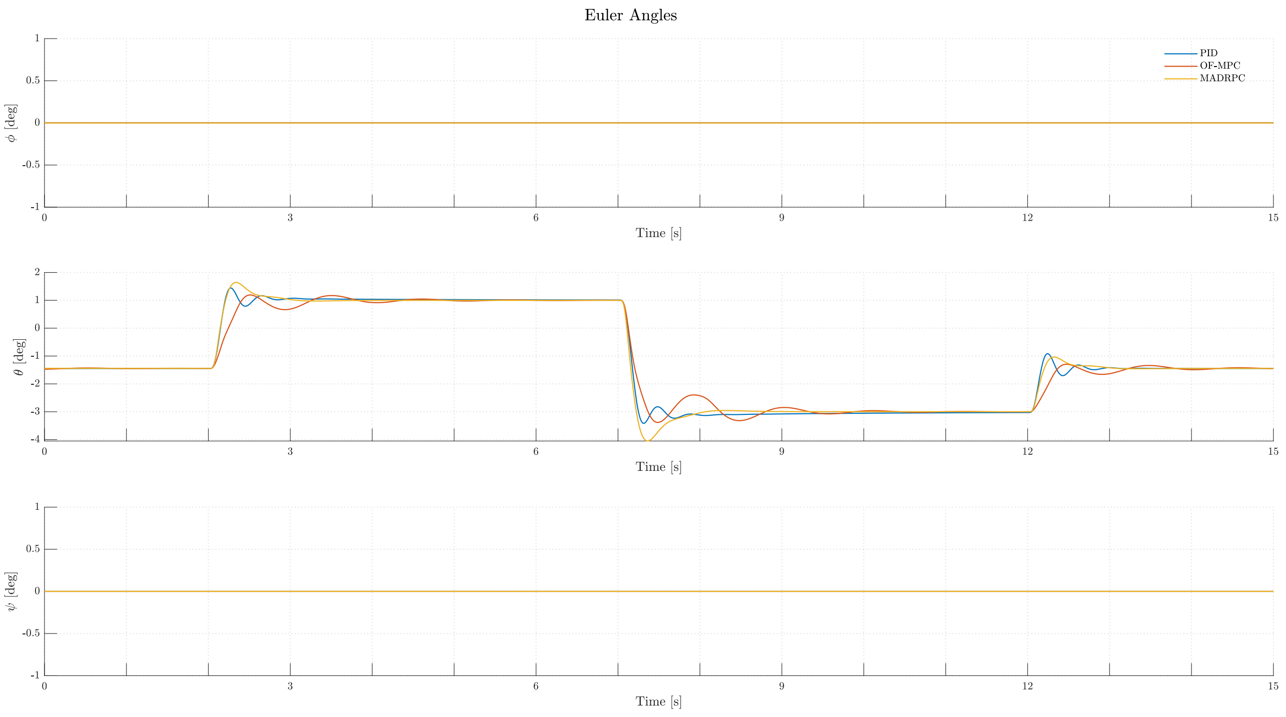


Figure 108: H200 Euler angles during **Test B.6**

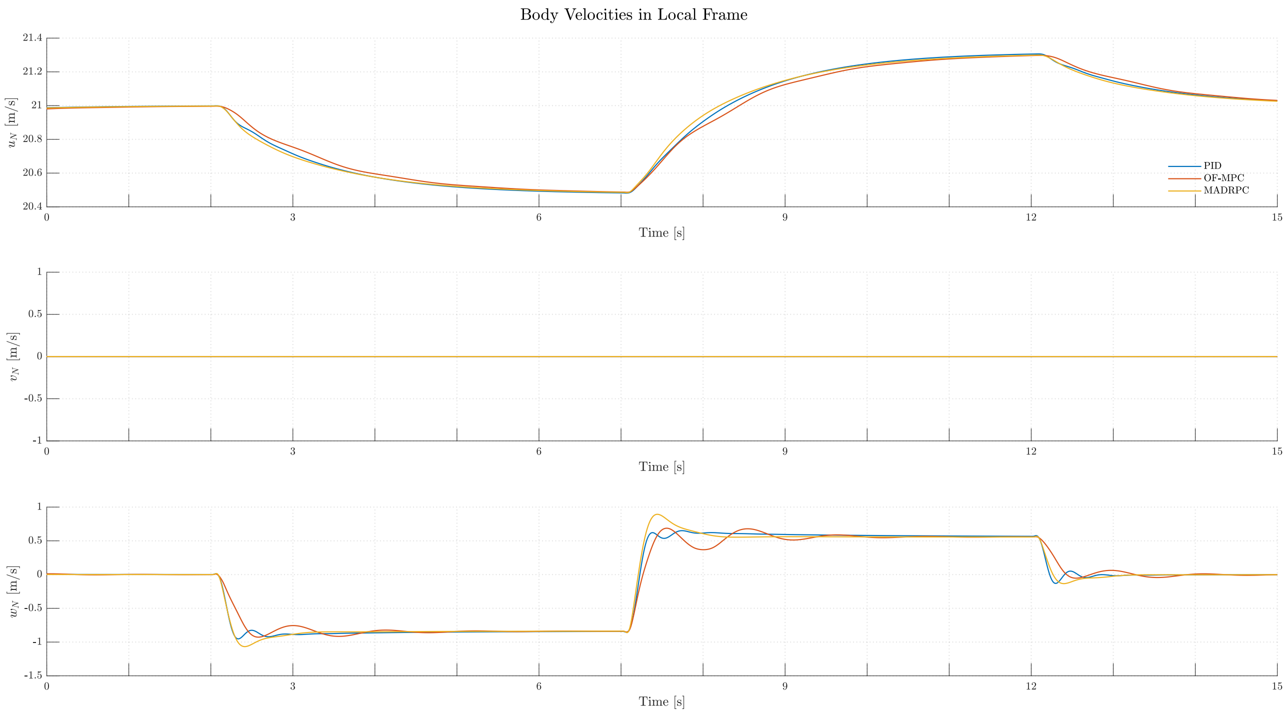


Figure 109: H200 body velocities expressed in local frame during **Test B.6**

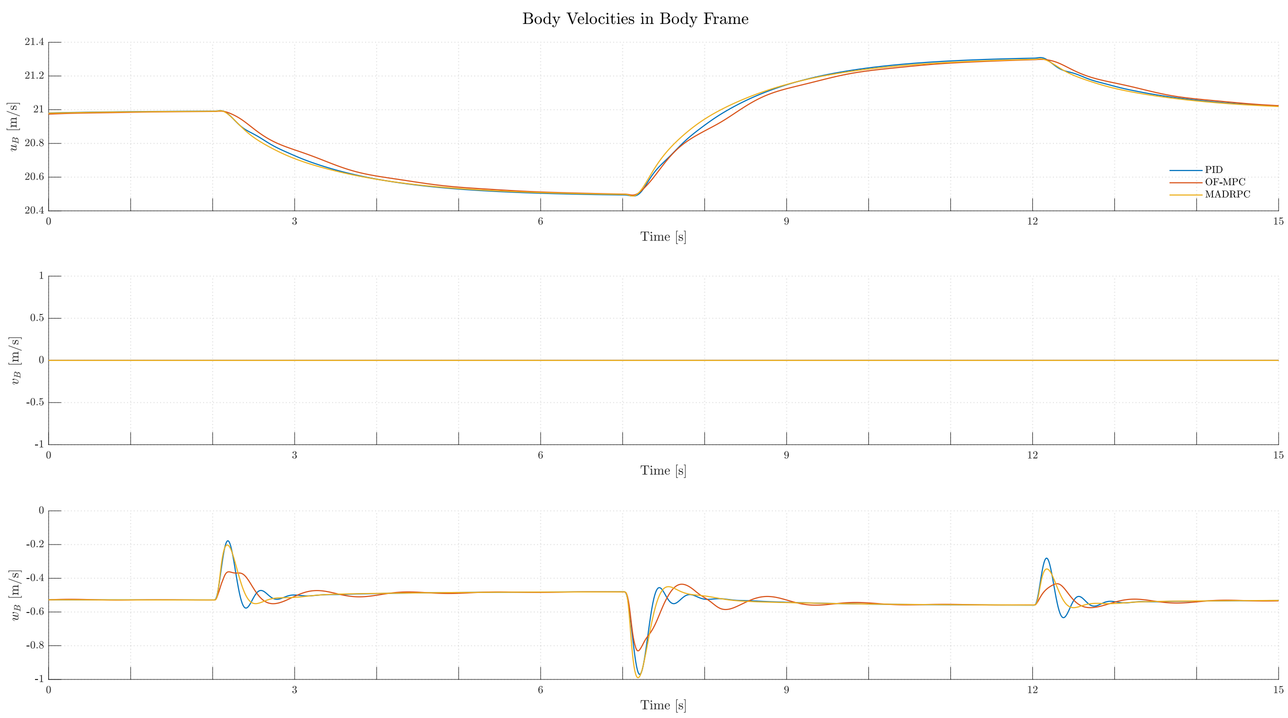


Figure 110: H200 body velocities expressed in body frame for **Test B.6**

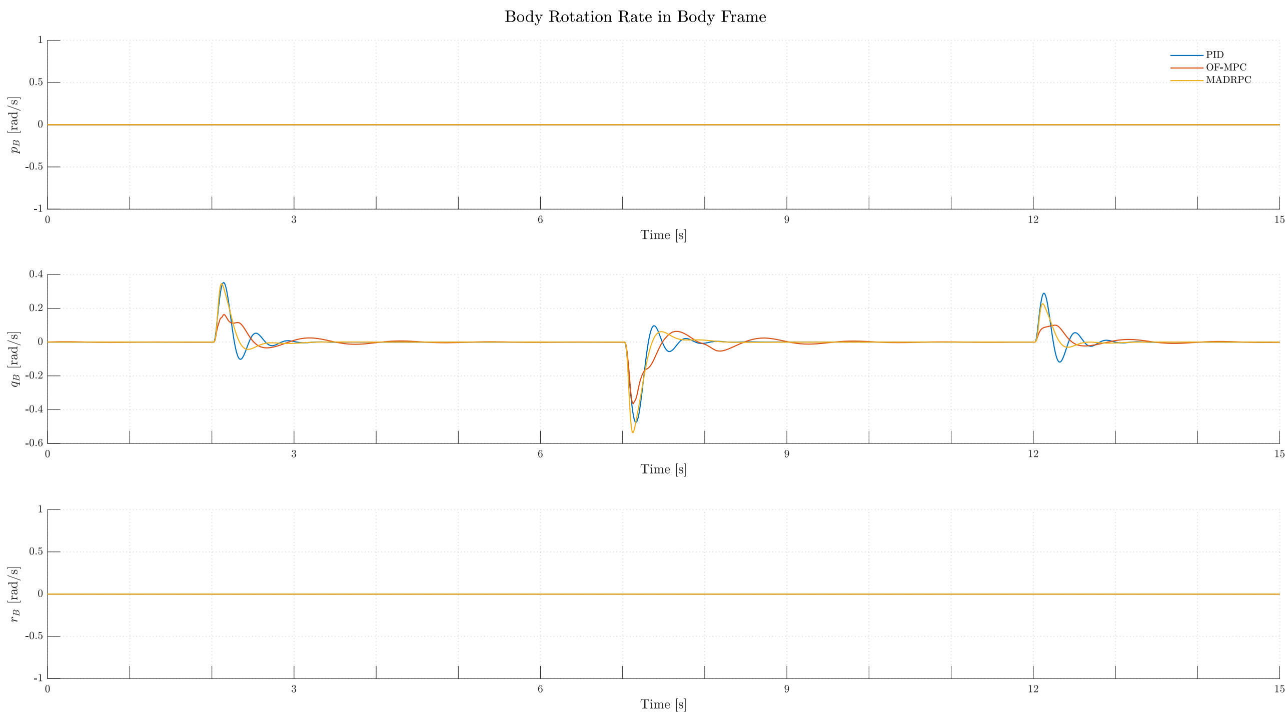


Figure 111: H200 body rotations during **Test B.6**

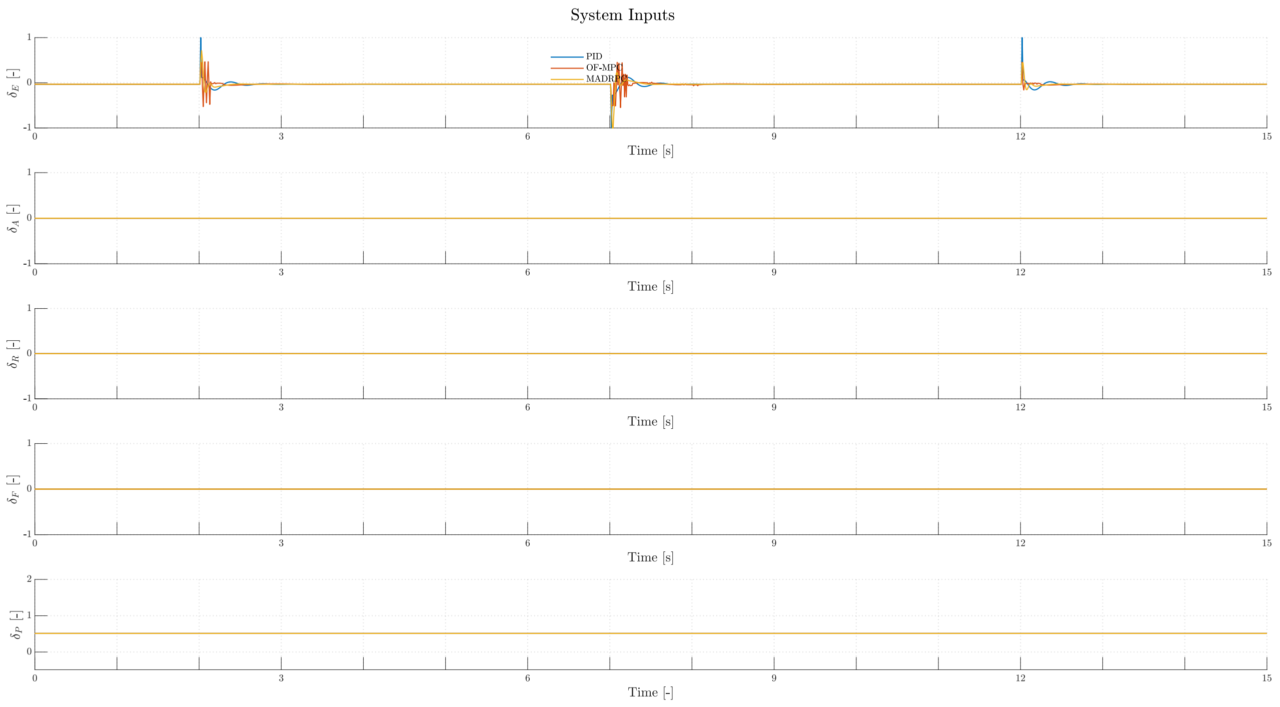


Figure 112: H200 system inputs during **Test B.6**

And the controllers performance indicators are:

	ITAE	ISE	IAE	MSE
PID	21.9675	0.090975	3.163	0.006065
OF-MPC	31.2185	0.11651	4.5032	0.0077673
MADRPC	21.1133	0.092242	3.1515	0.0061495

Table 32: Controller performance indicators for **Test B.6**

6.2.2.2.7 Tests Conclusions

With the tests finished, conclusions will be drawn. Once again, even with the modification on the structural parameters, the controllers have been able to give a stable response in all of the tests, with no flight capabilities lost.

Nevertheless, some commentaries about the flight performance may be given. First of all, in figure (113), it may be seen the summary for the performance KPIs controller indicators.

From figure (113), the performance of the controller is worsened as compared with the controllers validation tests (as expected). More specifically, in **Test B.1** and **Test B.2**, the OF-MPC and the PID controller increases their indicators respectively, while the MADRPC maintain its indicators between them. For the rests of the tests, for **Test B.3** and **Test B.4**, their performance stays in the same as the nominal tests. This verifies that a modification on the propeller model does not affect the pitch angle process. For the different mass tests, **Test B.5** and **Test B.6**, the PID and MADRPC controllers output the same performance indicators, while the OF-MPC controller loses effectiveness. As a remark from figure (113), the MADRPC controller is able to match the PID and the OF-MPC, and even being more effective than them, with no prior information from the process to control, apart from two

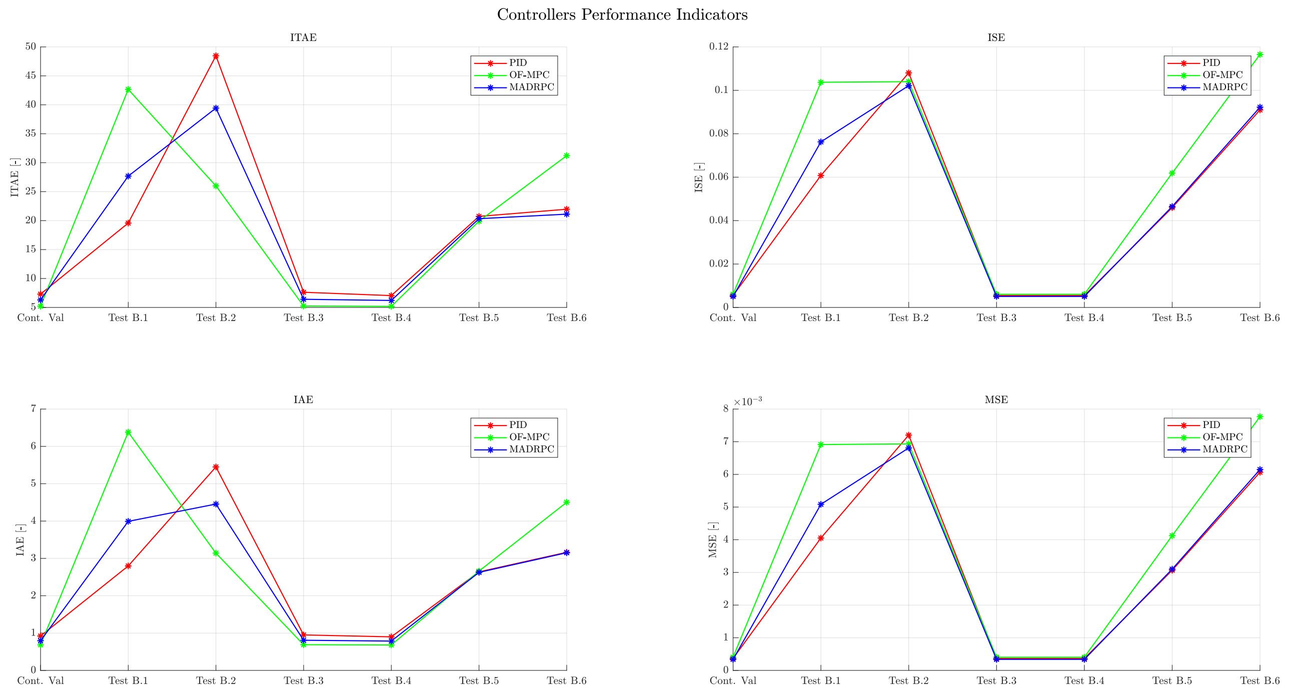


Figure 113: Controllers performance indicators for structural perturbation tests

natural parameters. The PID and the OF-MPC controllers need the full model information, either for tuning or predicting, while the MADRPC controller has an assumed model used for the predictive controller and a disturbance rejection for the difference between the real and the assumed plant.

On the other hand, from the simulation plots, it may be appreciated that the aerodynamic model and the mass play a fundamental role inside the pitch angle process, while the propeller model has not significant influence. The reason behind it is that the propeller model is only dependent on the throttle level, and the thrust force acts on the horizontal axis within the H200 body. Meanwhile, the aerodynamic model and the mass both define the lift and gravity force acting on the vertical axis, greatly impacting the equilibrium point and the pitch angle model.

More specifically, in a controller by controller review, the PID controller is able to withstand all the perturbations thrown at it. With the aerodynamic model, **Test B.1** and **Test B.2**, it gives a better performance when the model is underestimated, as when the coefficients are overestimated, it arrives almost to an oscillatory response. This may be explained as, with the loss of lift capabilities (**Test B.1**), the controller overestimate the lift force applied to the H200, making the response more damped. In the other case, where the lift force is underestimated, the controller action provokes the H200 creates more lift than from the design phase, so the response is higher than expected and the controller must compensate this. For the control actions, they are higher than with the nominal conditions test, even during **Test B.2** giving an oscillatory behavior, although not arriving to saturate it. From **Test B.3** and **Test B.4**, the PID controller gives the same performance as with in nominal conditions, as the propeller model does not affect the pitch angle process and for the control actions are the same as with nominal conditions. From the last tests, **Test B.5** and **Test B.6**, the controller is able to withstand the changes in mass, as this control algorithm is one of the most robust ones, although for its control action signal, it's not saturated, but it has an oscillatory behavior, trying to compensate the difference in mass as with the tuning phase.

For the OF-MPC controller, the aerodynamic model plays a fundamental role inside the algorithm, as expected, as it has to predict the future behavior of the model, so a wrong information about the lifting force capabilities will highly impact its performance. When the model coefficients are underestimated, **Test B.1**, the OF-MPC gives an oscillatory response during all the simulation, even from the initial time, not accomplishing giving a full stable flight. For **Test B.2**, with an overestimation of the model coefficients, the response is more damped, even struggling to arrive to the reference pitch angle, as the controller is giving a response signal that based on an optimization process with less lift force than the real model, making the response slower. Nevertheless, although the OF-MPC gives a stable enough response, its control action signal are far worsened, as the optimization process makes it saturate in both tests. This is a consequence of the difference between the prediction model and the real model, as the controller relays heavily on it to predict the behavior and thus, calculate the optimized rate of elevator deflections. With the difference in aerodynamic model, the optimization process is heavily affected, as the rate of control actions will not be the correct ones, and will lose performance. For the propeller model perturbations, **Test B.3** and **Test B.4**, as with the PID controller, the performance is similar as with the nominal conditions test. With the last set of tests, **Test B.5** and **Test B.6**, a similar occurrence happens as with the first tests, first, the OF-MPC is able to give a stable enough response, although there is present some overshoot in the response, but the reference is achieved. This may be explained, as with the difference in mass it will make the response slower and damped or more aggressive (**Test B.5** and **Test B.6** respectively), and with the control action, the optimization process struggles to find the correct control action rates, as, again, relies heavily on its inner prediction model, making it saturate whenever the reference changes. Once again, this fact affects whole flight performance, as those not optimized movements of the elevator deflection signal could make the UAV go into stall conditions.

Lastly, for the MADRPC controller, it has to be remarked that, although not having any prior information of the process plant beforehand, it is able to give a stable enough response in all tests. Starting with the aerodynamic model tests, **Test B.1** and **Test B.2**, it is appreciated a similar behavior as with the PID controller, when the model coefficients are underestimated (**Test B.1**), the response is more damped, as with the loss of lifting force capabilities, the assumed plant inside the controller is closer to the real plant. The reason behind it is that the pitch angle process is a highly non-linear system, that responds fast at any input, so making the H200 not have as much of lifting force, make closer the assumed first-linear model and the real one. On the other side, when the model is overestimated, it gives an oscillatory but stable response, similar to the one as the PID controller. The reason is the same, with the high non-linear system for the pitch angle, an increase in the lifting force capabilities makes the assumed plant and the real plant come apart. Nevertheless, taking into account that the controller has no prior information of the real process (apart from two natural parameters), it is capable of governing the pitch angle, with a smooth elevator deflection signal, unlike the saturated one from the OF-MPC. This is a consequence of the first-order assumed plant inside the controller, which the optimization process uses to calculate the rate of control actions, and then a disturbance rejection controller compensates the difference between the real and the assumed model. Then, once again, with the propeller model perturbations, the controller behaves as with nominal conditions, verifying the fact that the propeller model does not affect the pitch angle process. Then, with the last set of tests, varying the flight mass, again it may be seen a performance similar to the PID controller, even surpassing in performance (from the

KPIs in figure (113)), from the traditional predictive controller. Once again, the controller is able to compensate the difference in mass through the flight, with a smooth control action signal. This is a remarkable fact, as again, the MADRPC controller has no information from the controlled plant, an assumed first-order model is given to the optimization process beforehand, and it is able to output a similar performance as with the PID controller.

6.2.2.3 Input Perturbation Tests

The last family of tests will focus on modifying the control action signals during the simulation. Meaning that the tests will not focus on the controllers ability to adapt in a change of the H200 itself, either from the flight conditions or the inner model, but to introduce a perturbation during the flight mission.

During a normal operation, an aircraft will experience a wide range of disturbances during its flight mission. In these tests, it will be simulated the tow main ones: a wind gust during the operation and a perturbation in the elevator control signal.

For these tests, the H200 will fly in nominal conditions, using the numerical values from table (13), as the H200 will fly in cruise conditions, the same conditions as the controllers validation tests. Then, it will be introduced each type of disturbance at a specific time,both as an instantaneous perturbation and a maintained over time one, which will test the disturbance rejection capability of the controllers.

For this family of tests, it will be performed four tests: the first two will focus on the wind perturbation (instantaneous and maintained), and the latter two will focus on the input signal perturbation (instantaneous and maintained).

6.2.2.3.1 Test C.1: Maintained Wind Perturbation

As commented, the first test will focus on a wind perturbation, which duration will be from the specific time of application until the end of simulation. For this test, and the next one, it will introduced the wind as a vector expressed with respect to the local frame (flat-Earth representation):

$$^{E}W = \begin{bmatrix} 0 \\ 0 \\ -5 \end{bmatrix} \tag{101}$$

where the units are m/s. The physical meaning for the negative sign in the third element is that the wind gust will come from the bottom of the aircraft, forcing its nose up, so the controllers will have to compensate it. The time that the wind gust will be introduced in the system is:

$$T_{dis,wind} = 7.5\,s \tag{102}$$

The reference pitch signal parameters for **Test C.1** will be the same as the controllers validation test, as the perturbation comes from an external disturbance, not from the inner model:

Symbol	Value	Units
T_1	2	s
T_2	7	s
T_3	12	s
θ_{trim}	2.5905	deg
θ_{ref_1}	3	deg
θ_{ref_2}	2	deg

Table 33: Reference signal parameters for **Test C.1**

And the results from the test may be seen from figure (114) to (122):

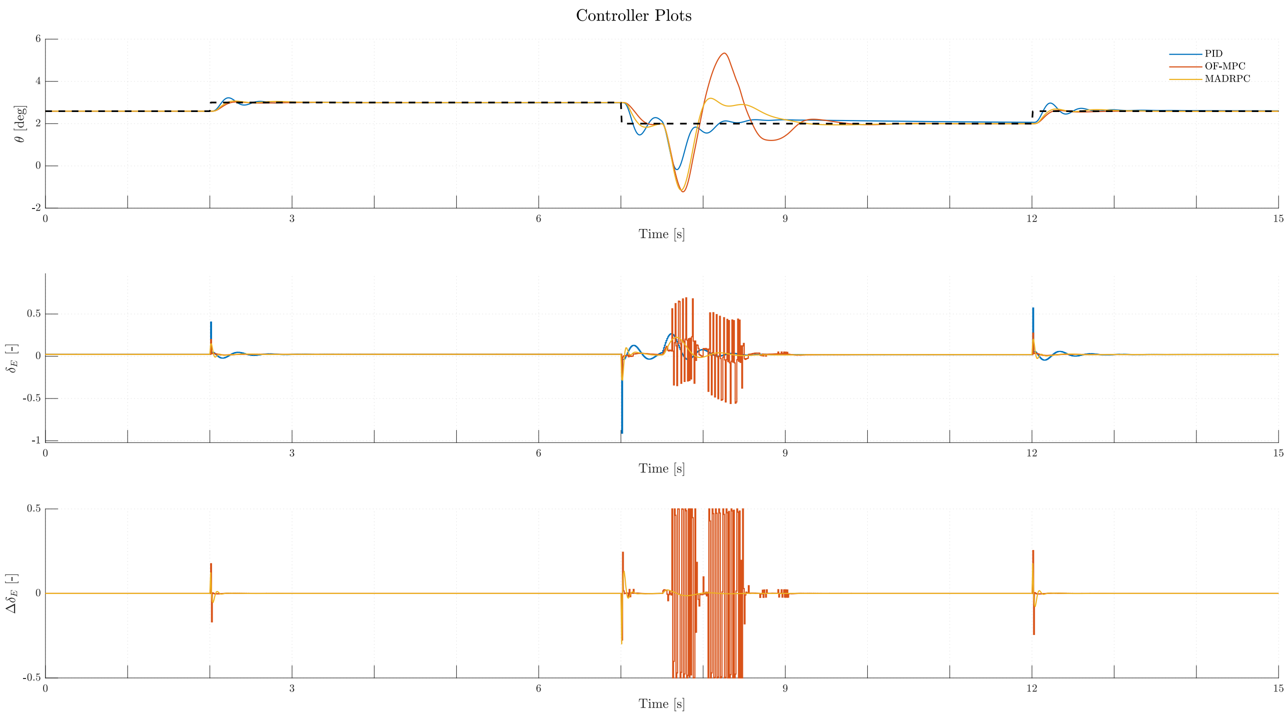


Figure 114: Controllers plots during **Test C.1**

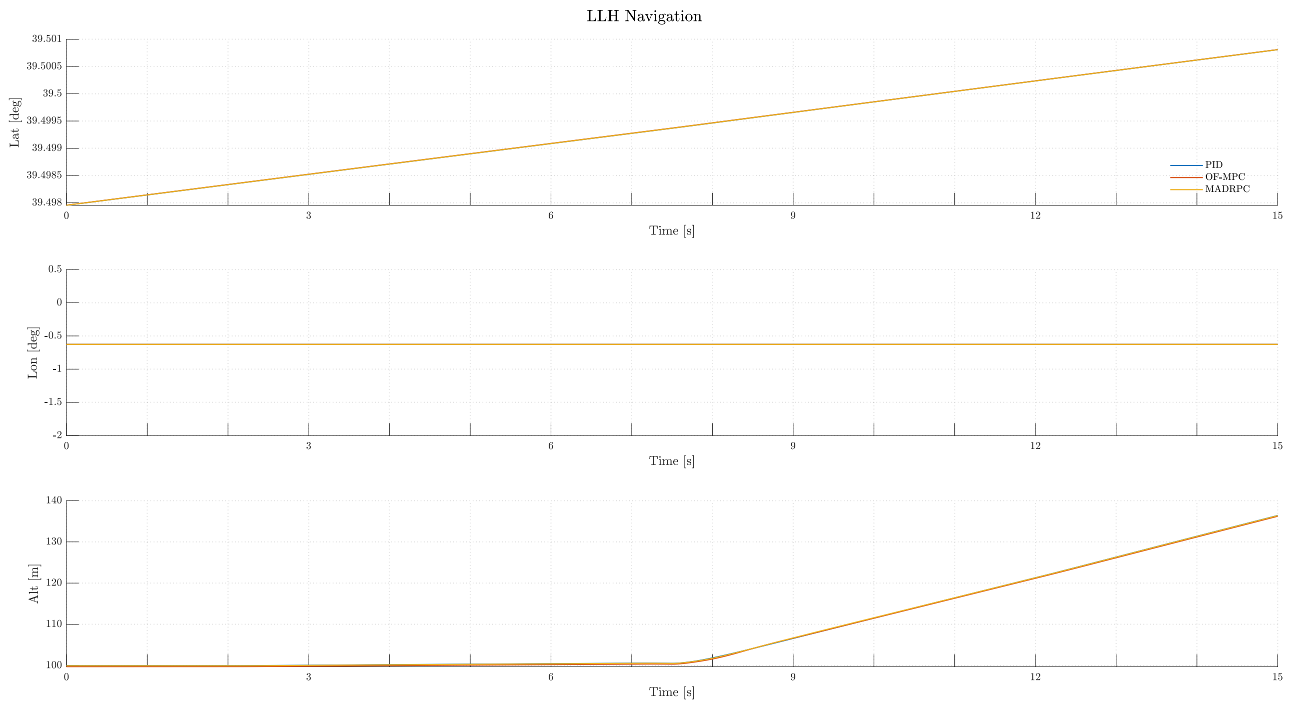


Figure 115: H200 LLH navigation during **Test C.1**

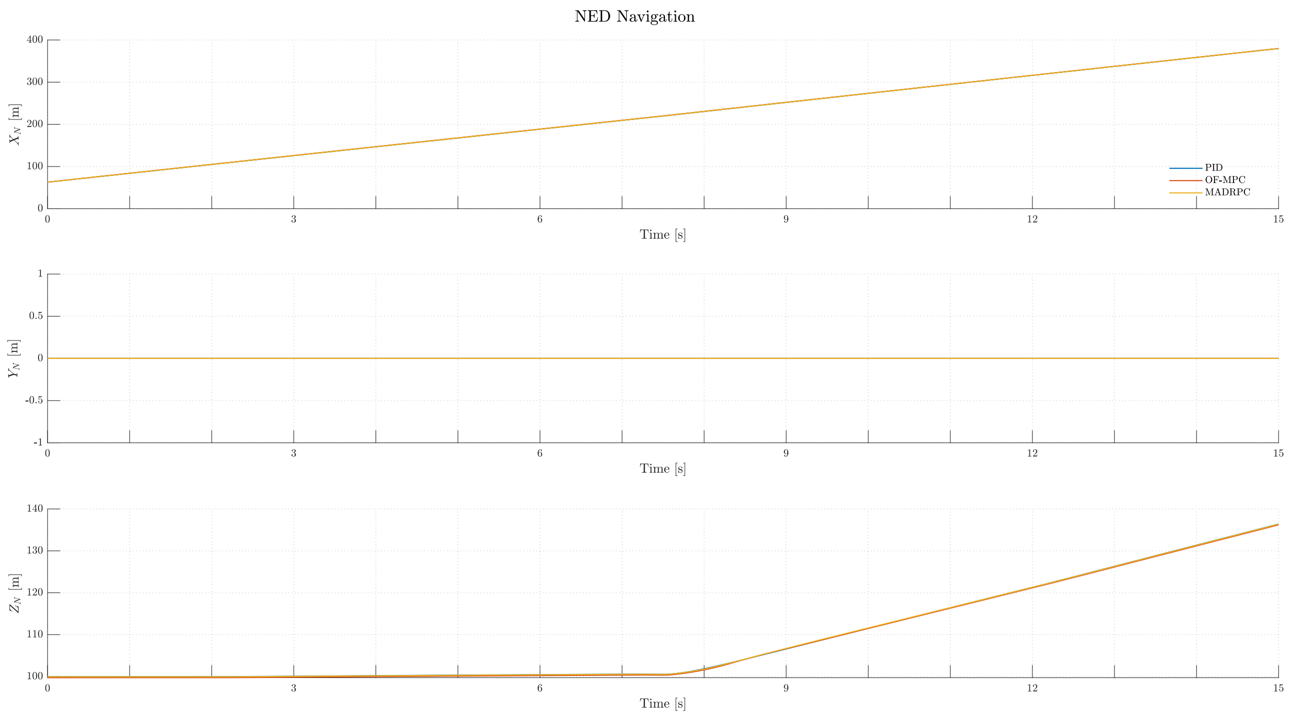


Figure 116: H200 NED navigation during **Test C.1**

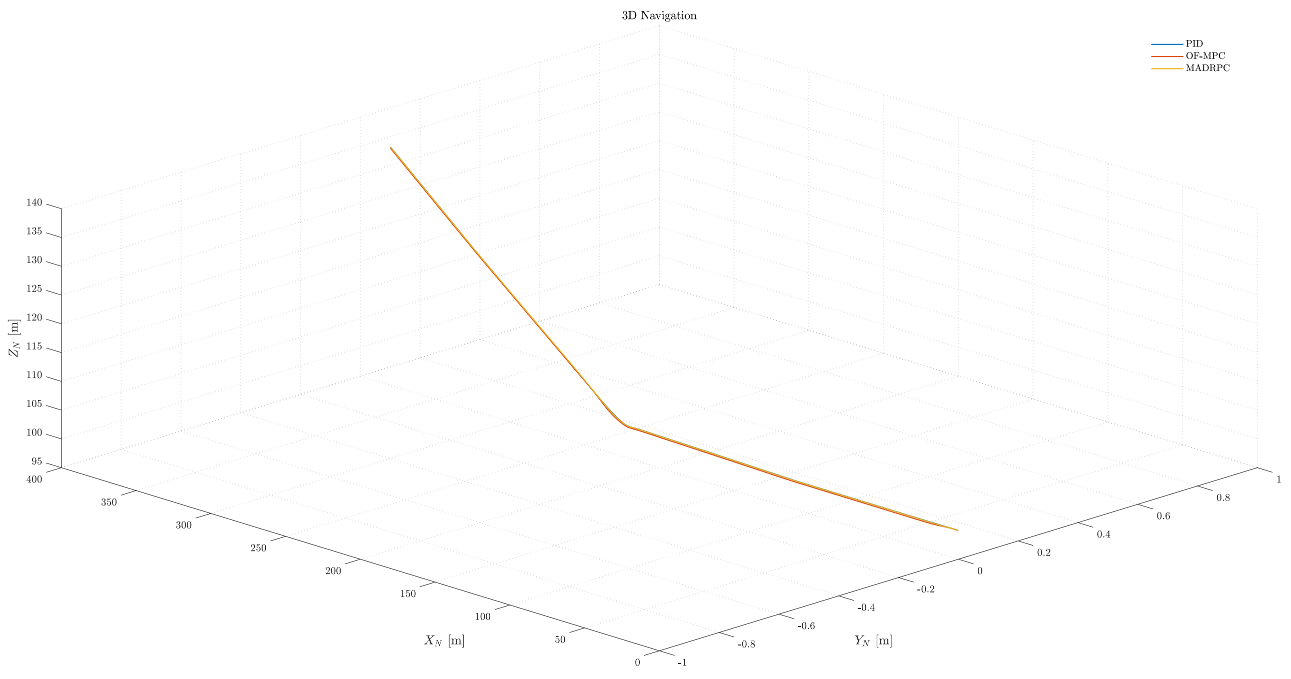


Figure 117: H200 3D navigation during **Test C.1**

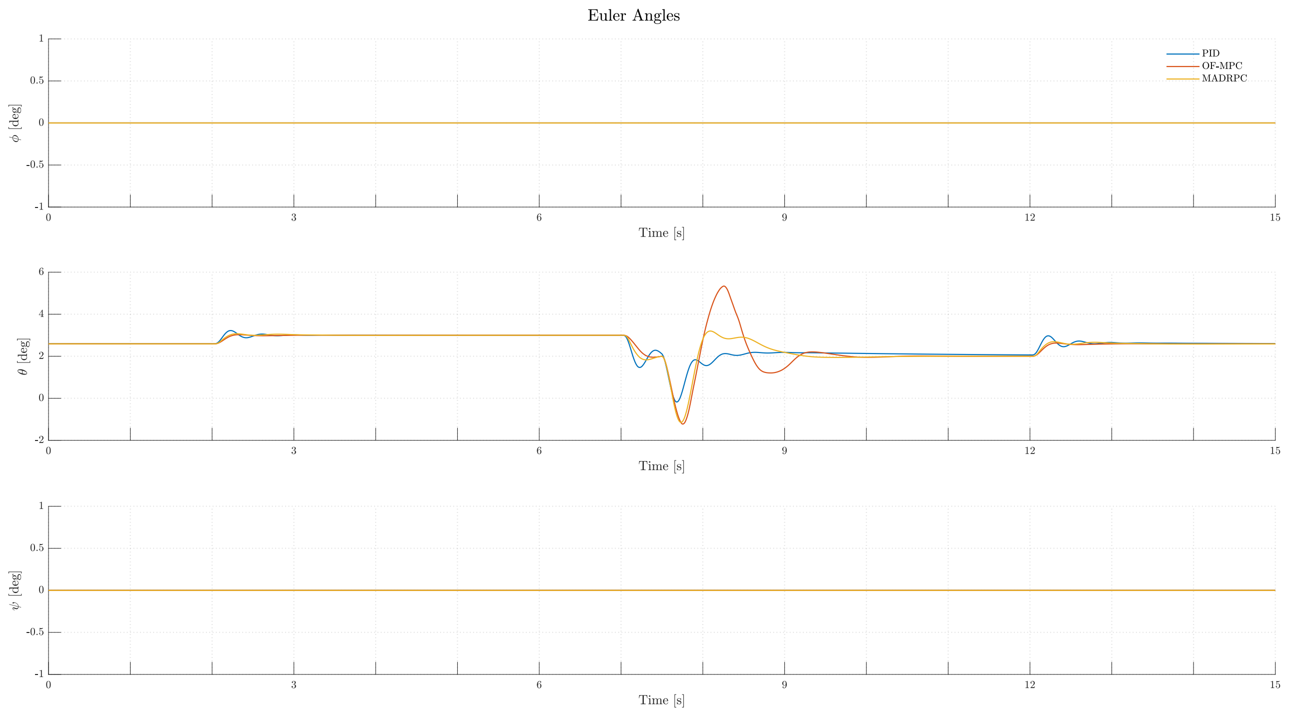


Figure 118: H200 Euler angles during **Test C.1**

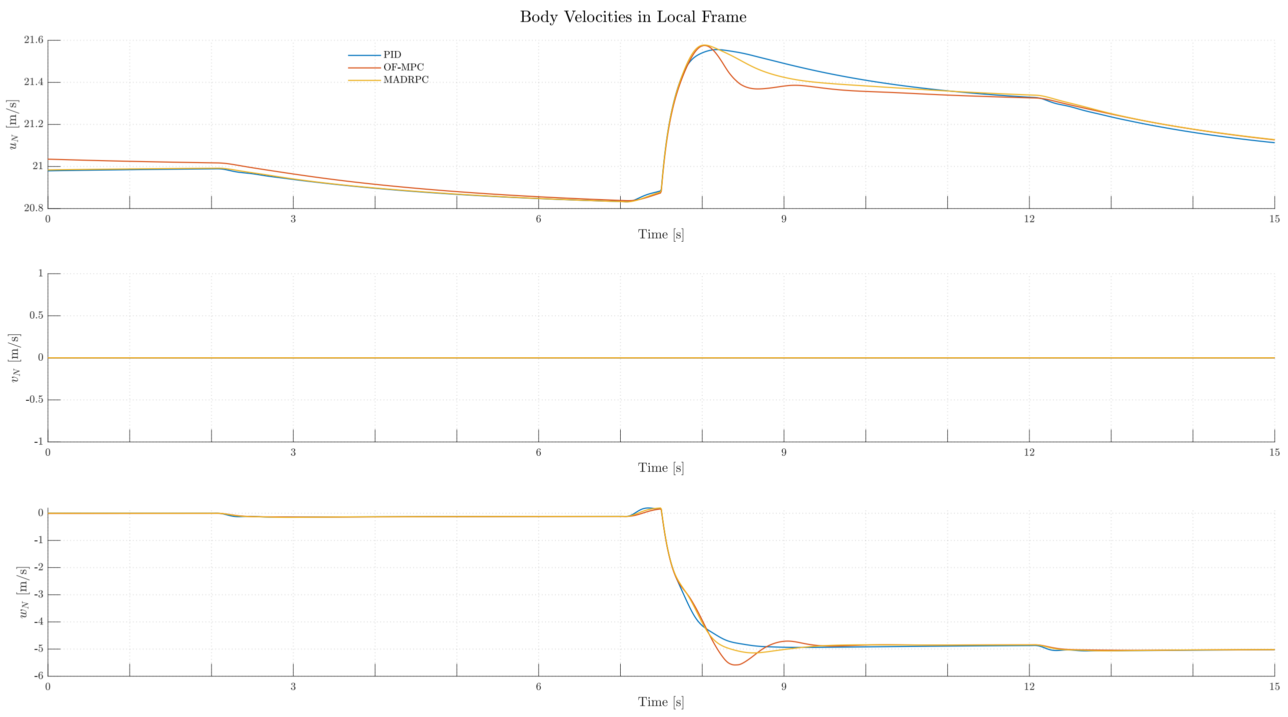


Figure 119: H200 body velocities expressed in local frame during **Test C.1**

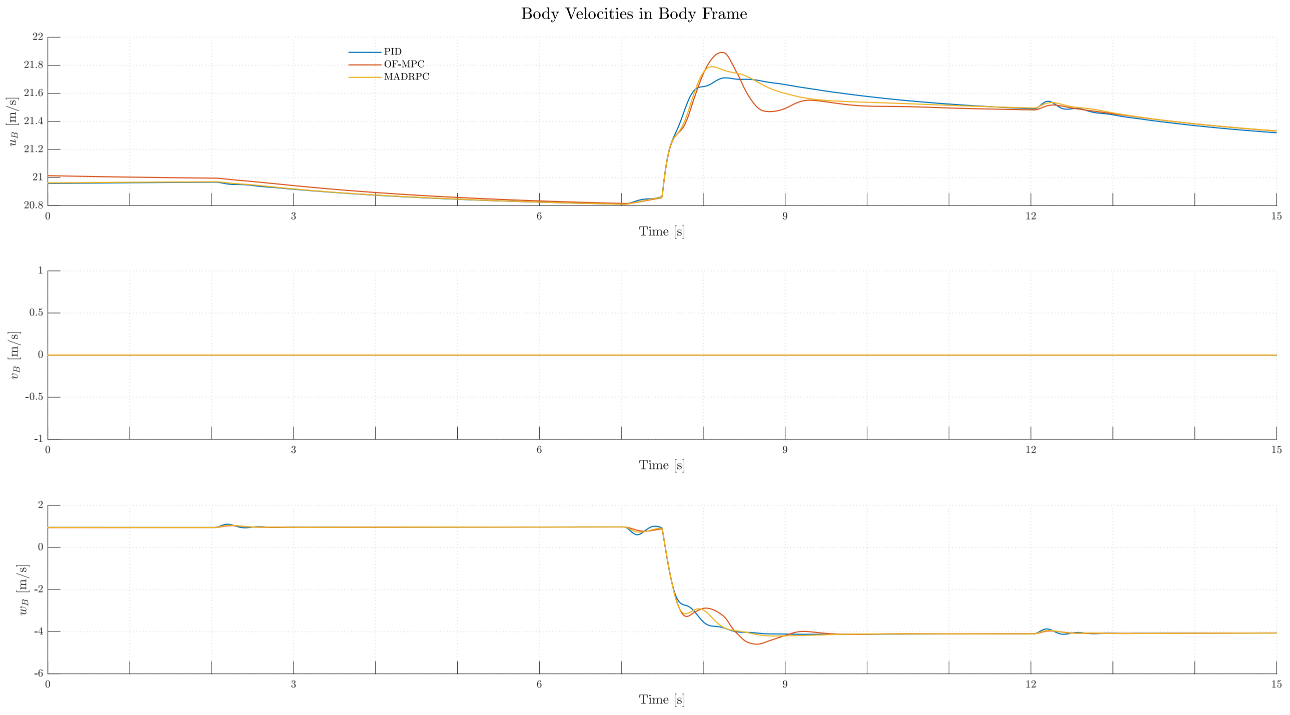


Figure 120: H200 body velocities expressed in body frame for **Test C.1**

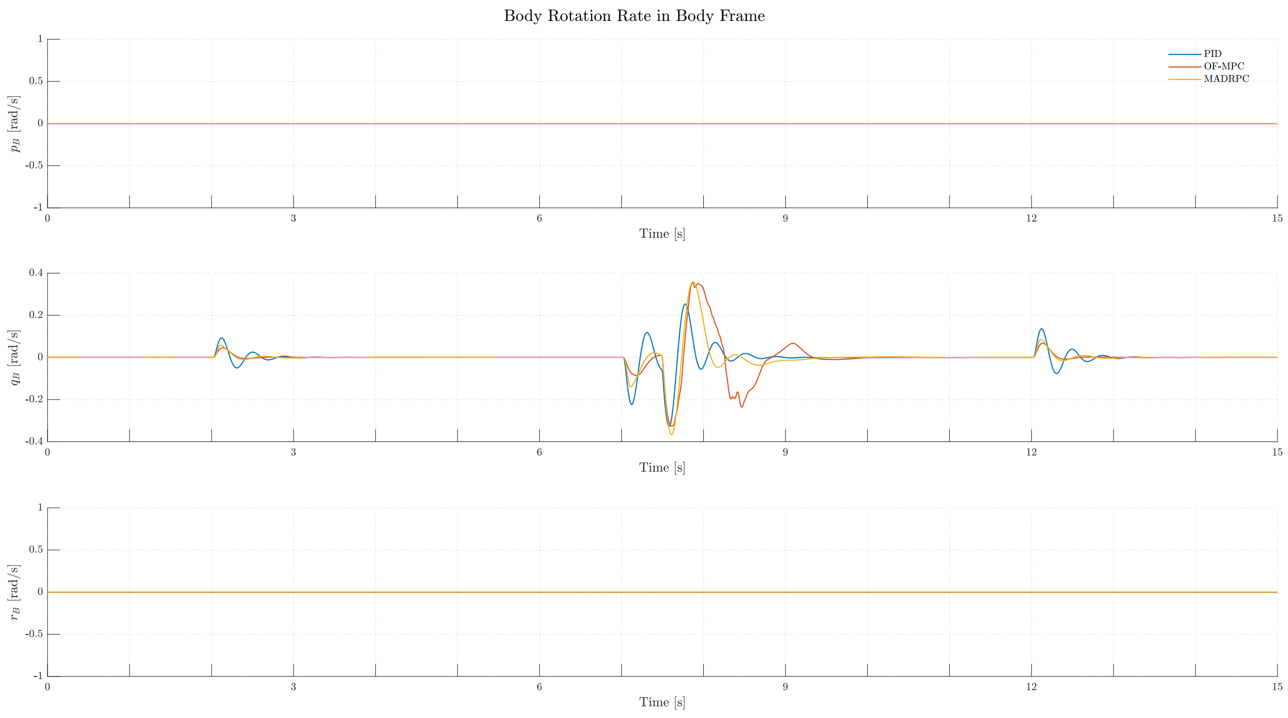


Figure 121: H200 body rotations during **Test C.1**

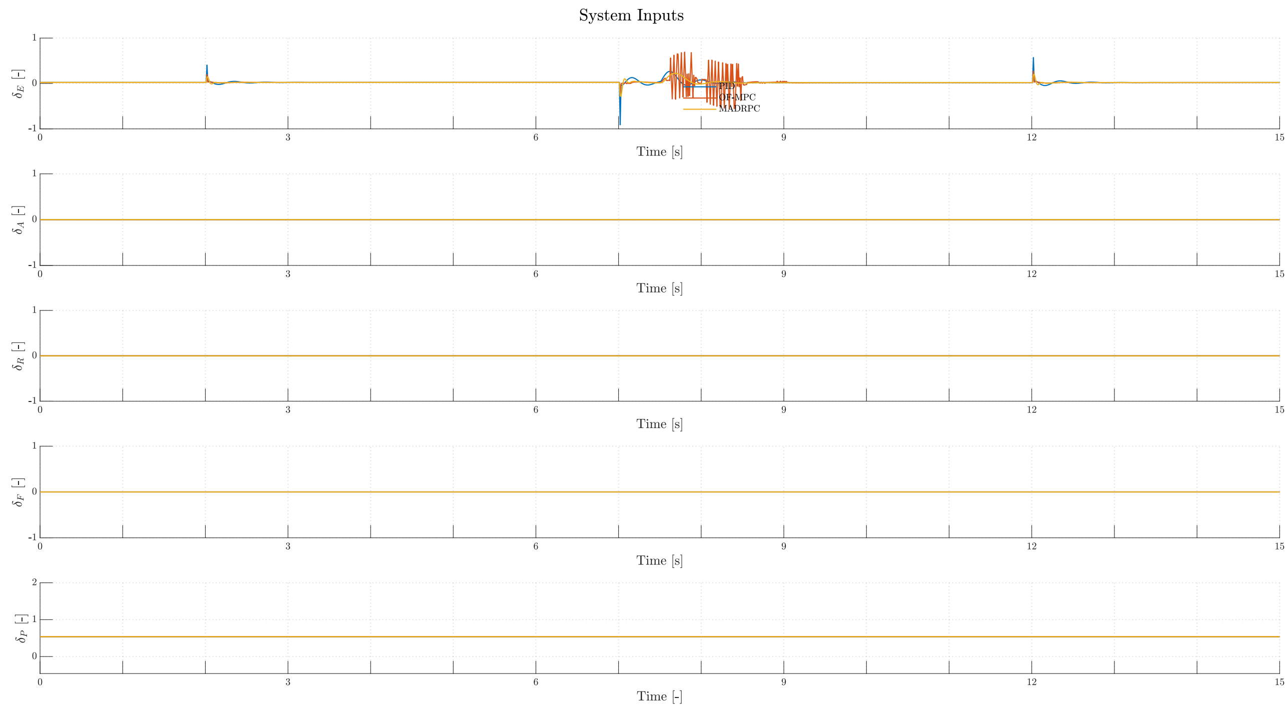


Figure 122: H200 system inputs during **Test C.1**

And the controllers performance indicators are:

	ITAE	ISE	IAE	MSE
PID	22.2365	0.030227	2.6005	0.0020151
OF-MPC	40.2131	0.16792	4.9425	0.011195
MADRPC	27.8174	0.080309	3.4497	0.0053539

Table 34: Controller performance indicators for **Test C.1**

6.2.2.3.2 Test C.2: Instantaneous Wind Perturbation

For the second test, it will be once again introduced a wind perturbation in the system. The wind vector will take the form as in the last test, as in equation (101). The difference with the last test, is that this time, the wind vector will be limited in time, to test the disturbance rejection capabilities at the beginning and at the end of the introduction of the perturbation. The wind vector will be introduced at the same time as **Test C.1**:

$$T_{dis,wind} = 7.5\,s \tag{103}$$

The duration of the wind perturbation will be:

$$T_{dur\,wind} = 3\,s \tag{104}$$

Once again, the reference pitch signal parameters for **Test C.2** will be the same as the controllers validation test:

Symbol	Value	Units
T_1	2	s
T_2	7	s
T_3	12	s
θ_{trim}	2.5905	deg
θ_{ref_1}	3	deg
θ_{ref_2}	2	deg

Table 35: Reference signal parameters for **Test C.2**

And the results from the test may be seen from figure (123) to (131):

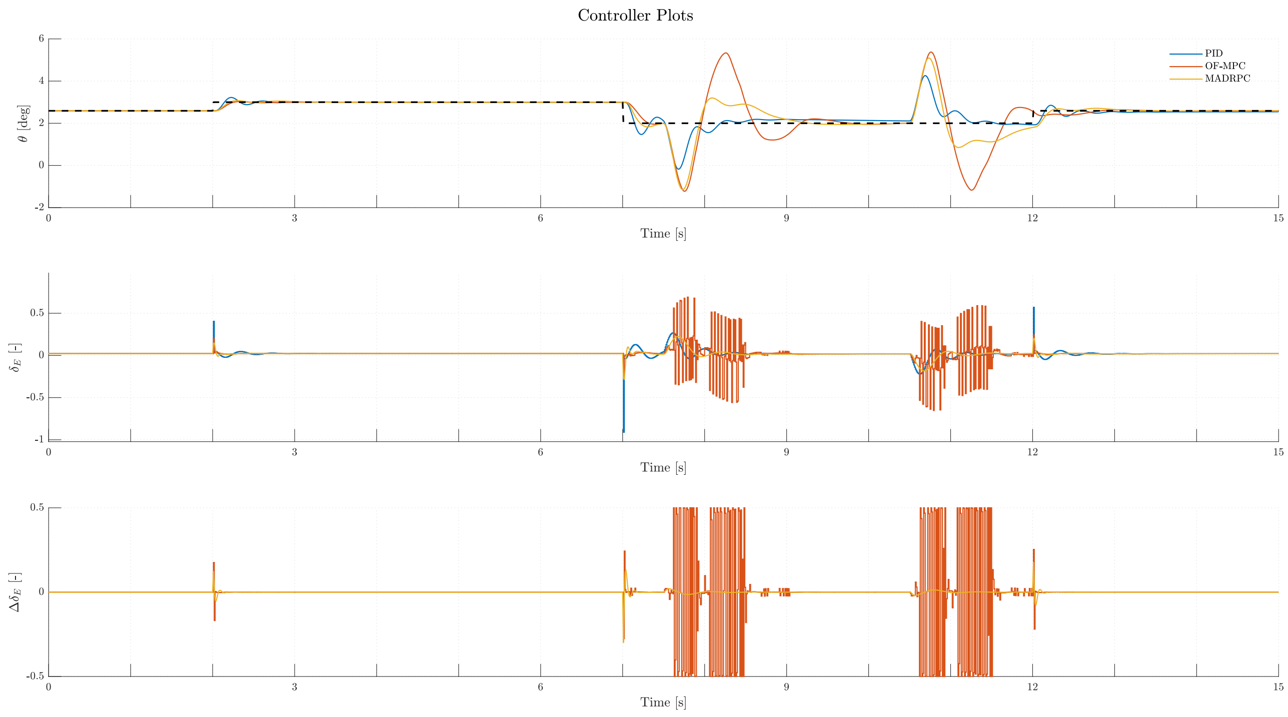


Figure 123: Controllers plots during Test **Test C.2**

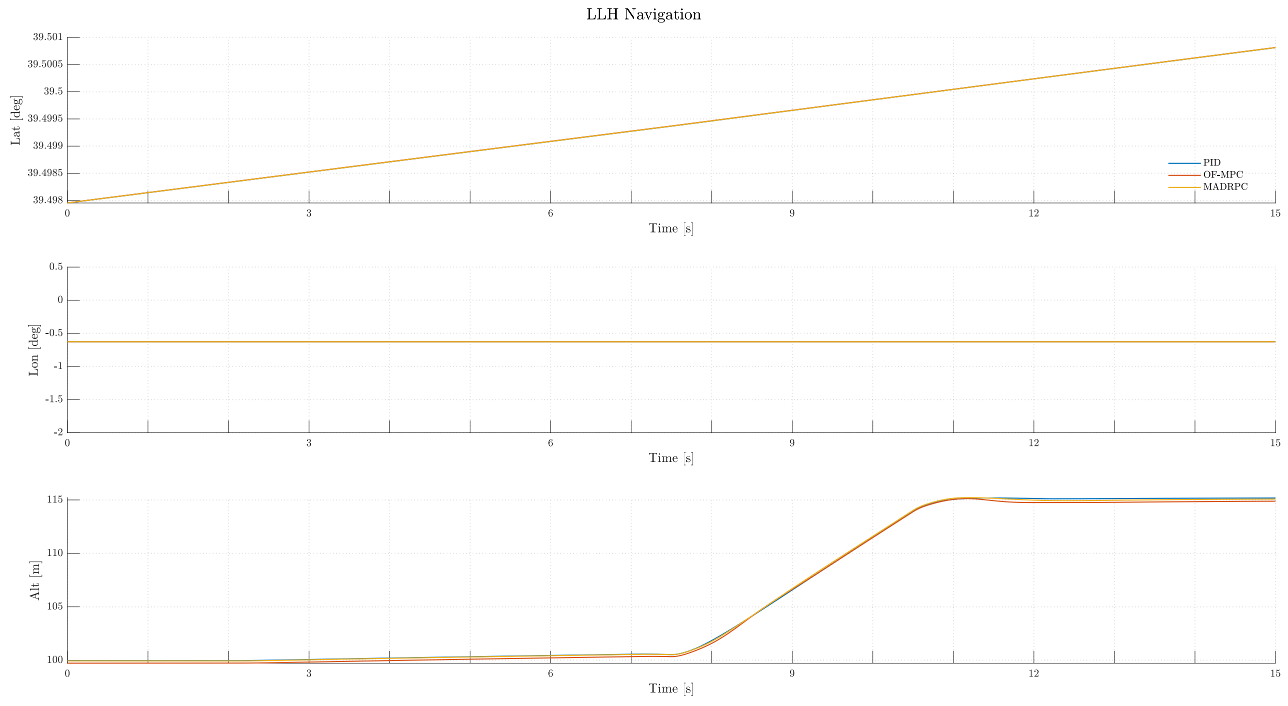


Figure 124: H200 LLH navigation during **Test C.2**

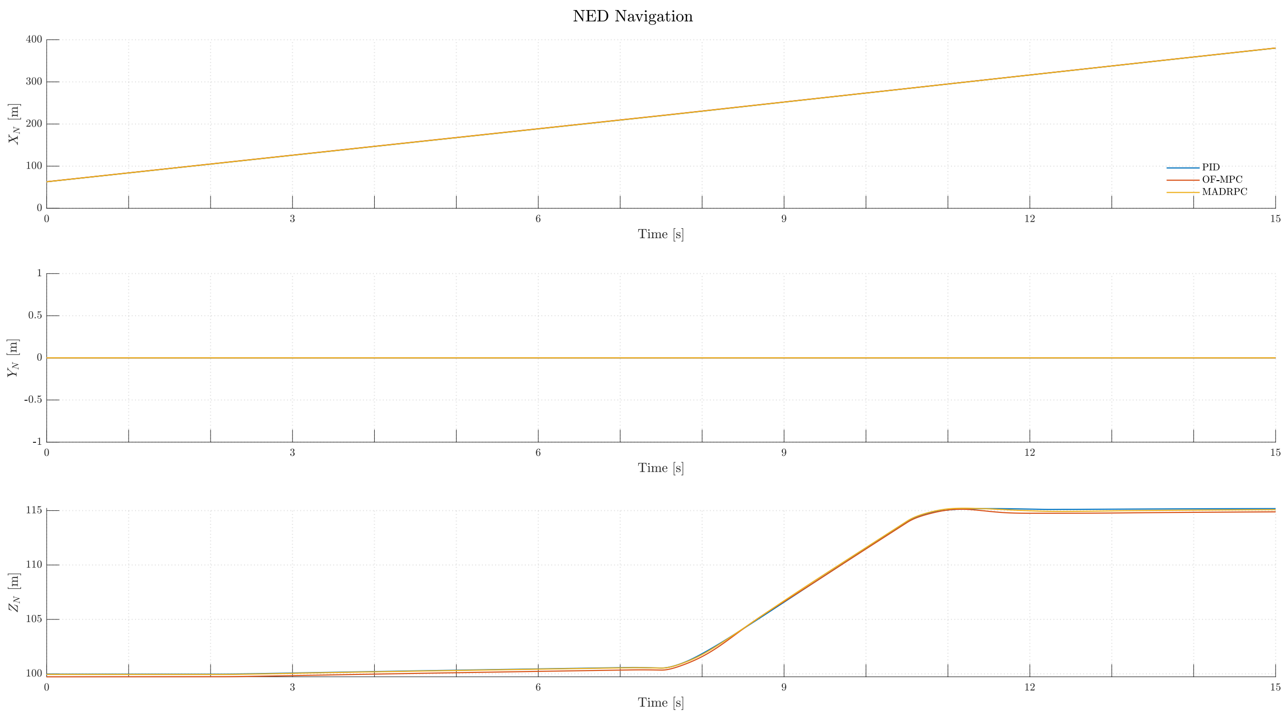


Figure 125: H200 NED navigation during **Test C.2**

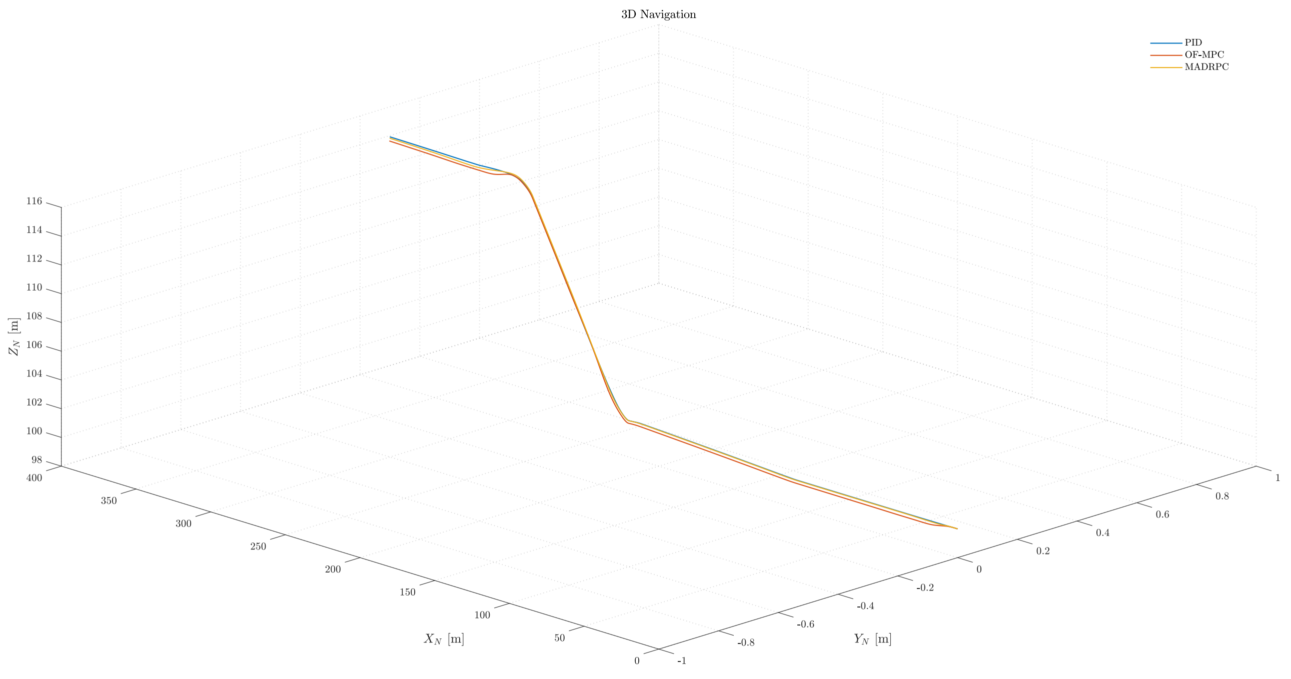


Figure 126: H200 3D navigation during **Test C.2**

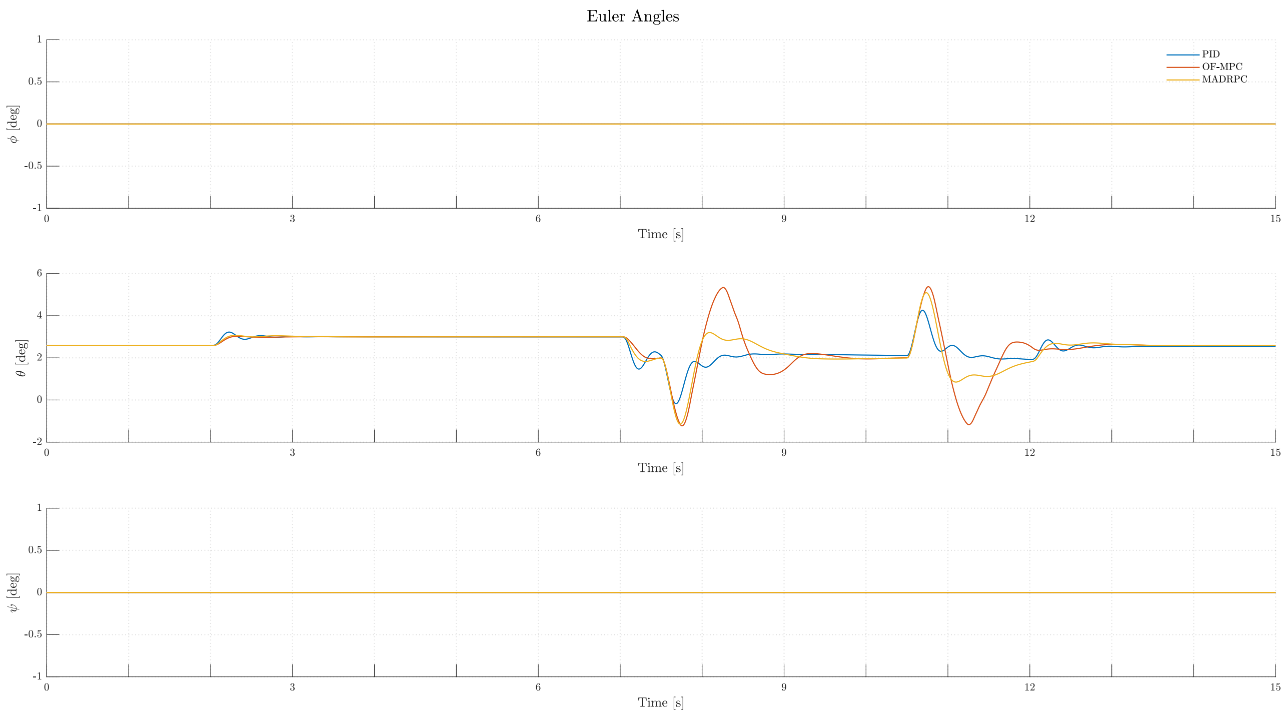


Figure 127: H200 Euler angles during **Test C.2**

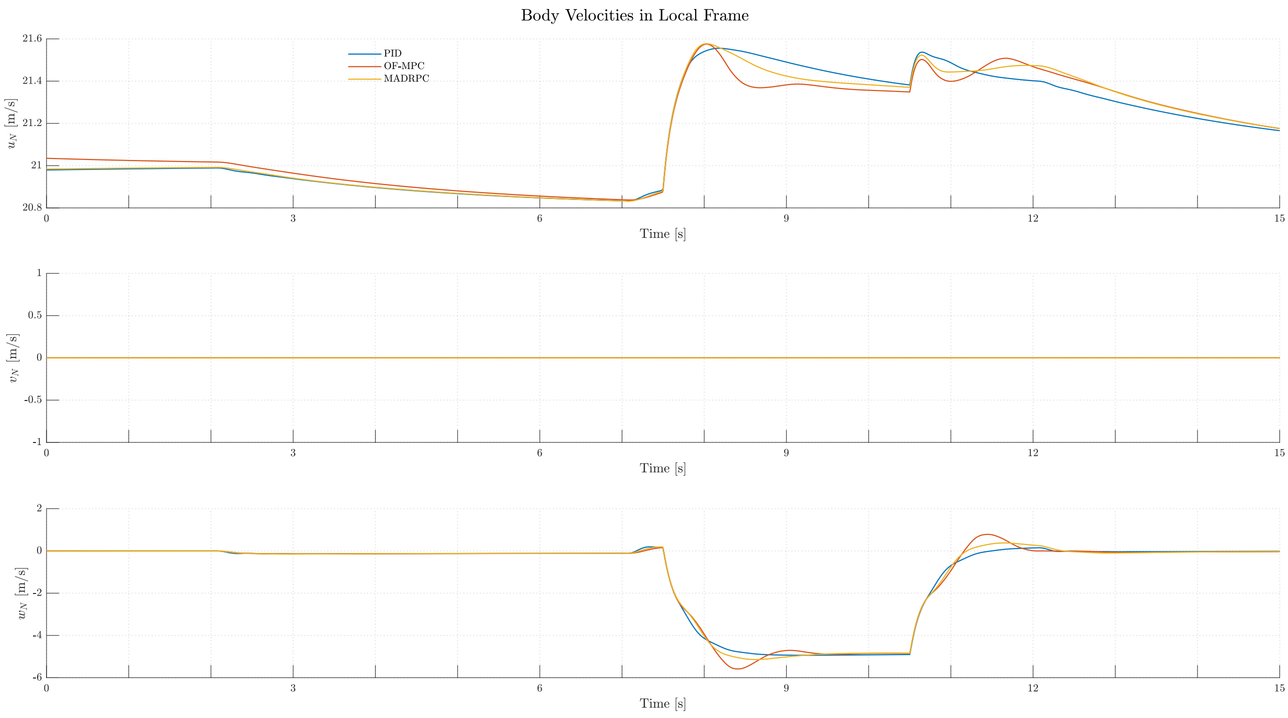


Figure 128: H200 body velocities expressed in local frame during **Test C.2**

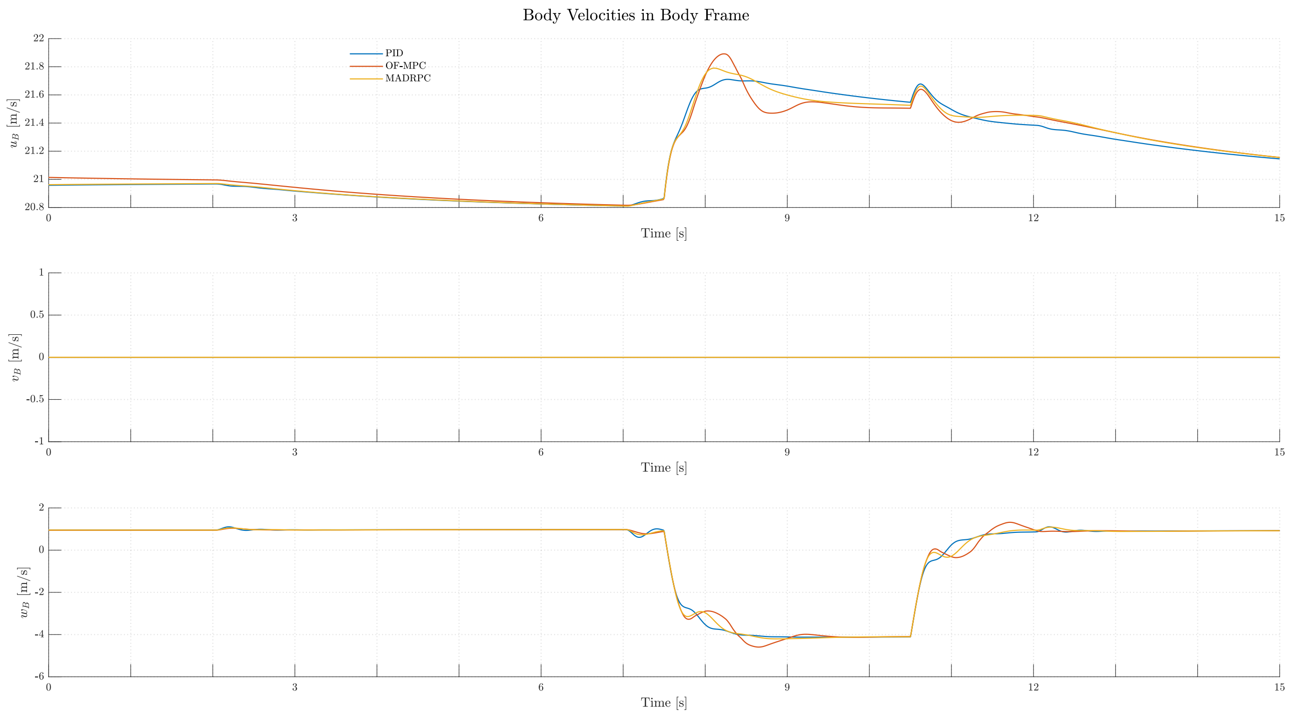


Figure 129: H200 body velocities expressed in body frame for **Test C.2**

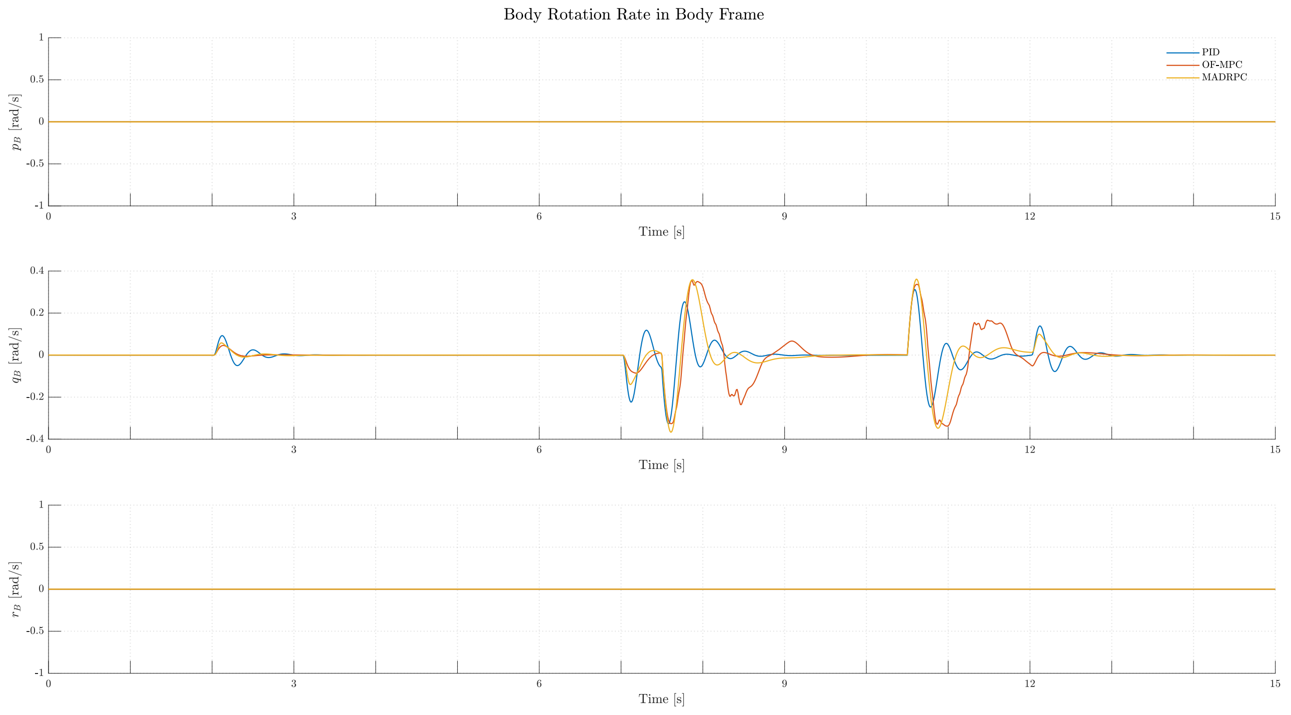


Figure 130: H200 body rotations during **Test C.2**

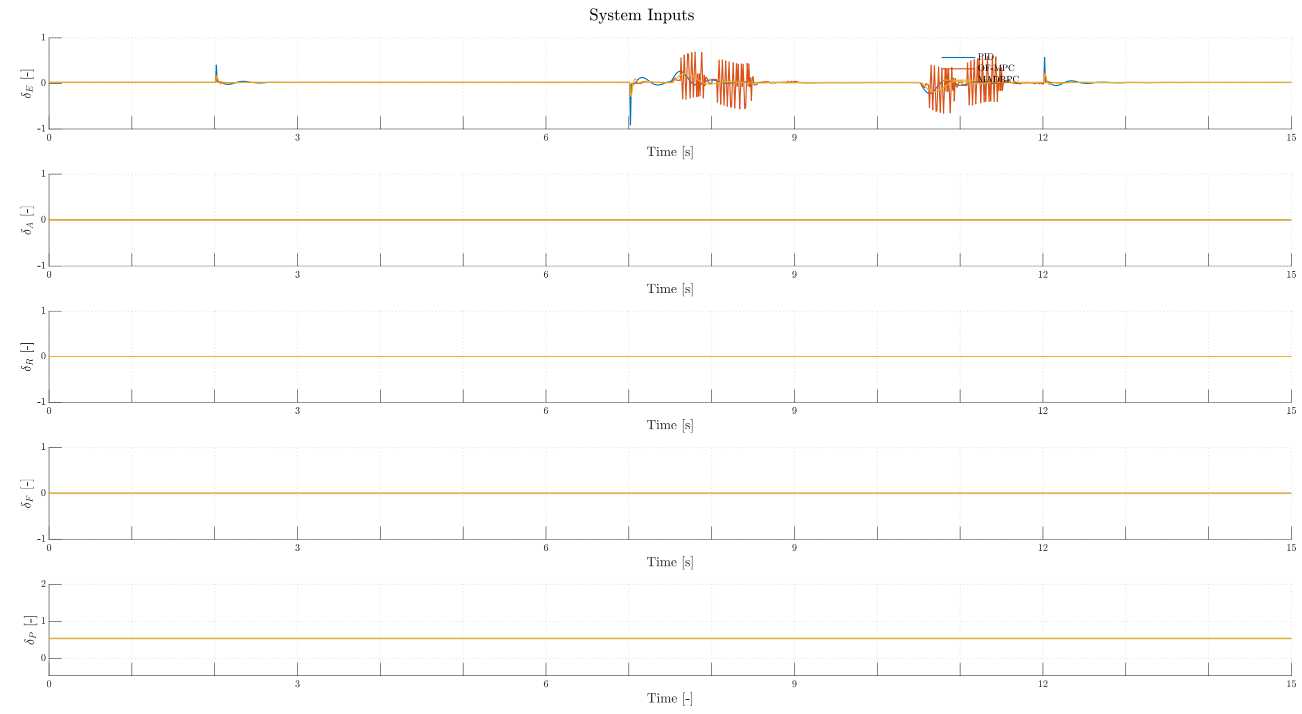


Figure 131: H200 system inputs during **Test C.2**

And the controllers performance indicators are:

	ITAE	ISE	IAE	MSE
PID	34.0417	0.057262	3.6661	0.0038175
OF-MPC	86.4345	0.32814	9.0793	0.021876
MADRPC	57.9453	0.15366	6.16	0.010244

Table 36: Controller performance indicators for **Test C.2**

6.2.2.3.3 Test C.3: Maintained Elevator Deflection Action Perturbation

This test, and the next one, will focus on the second type of control action perturbations: a disturbance on the control signal directly. This will simulate a faulty connection between the actuator and the controller, producing a non-desired control action in the middle of the simulation. The controllers will have to face it, as with the wind gust tests.

The perturbation will be modeled as a step function, which will start from the moment that it is introduced to the system, up until the end of the simulation. The amplitude of the step will be:

$$A_{input,amp} = 0.2 \tag{105}$$

and the time when the perturbation is applied will be:

$$T_{input,dis} = 7.5\,s \tag{106}$$

For the reference pitch signal parameters for **Test C.3**, as with **Test C.1** and **Test C.2**, will be the same as the controllers validation test, as the perturbation comes from an external input disturbance, not from the inner model:

Symbol	Value	Units
T_1	2	s
T_2	7	s
T_3	12	s
θ_{trim}	2.5905	deg
θ_{ref_1}	3	deg
θ_{ref_2}	2	deg

Table 37: Reference signal parameters for **Test C.3**

And the results from the test may be seen from figure (132) to (140):

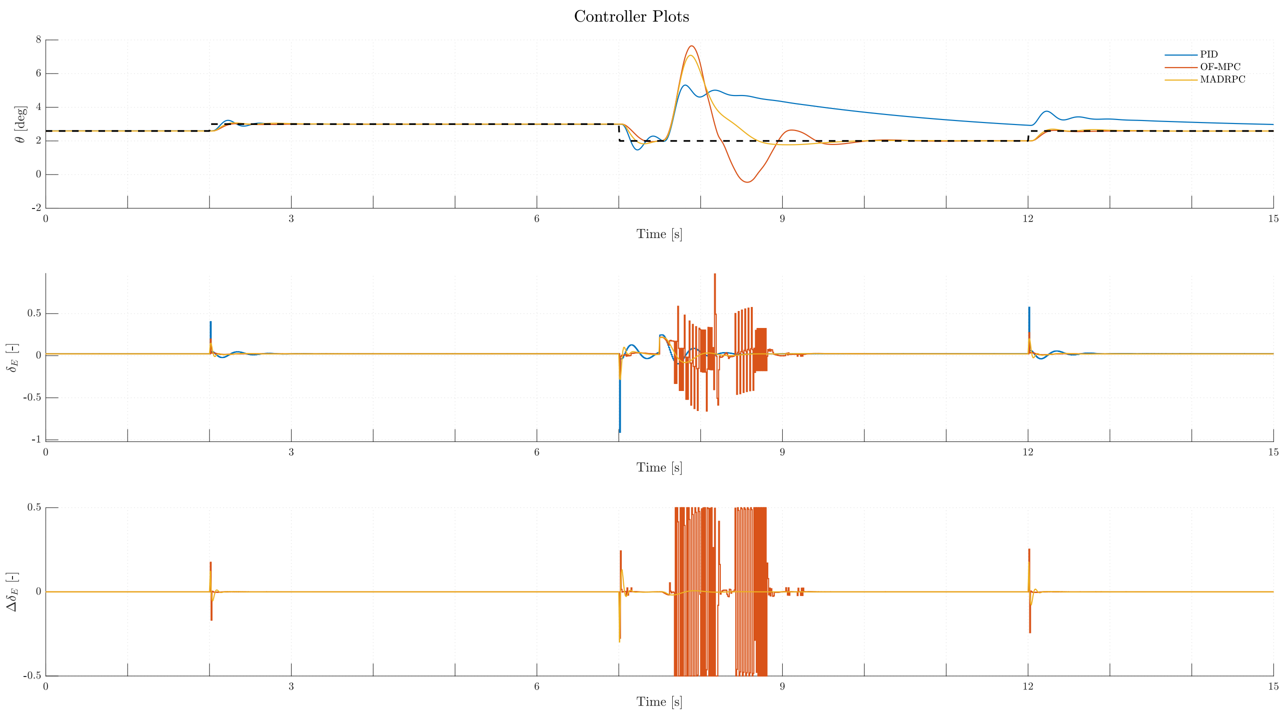


Figure 132: Controller plots during **Test C.3**

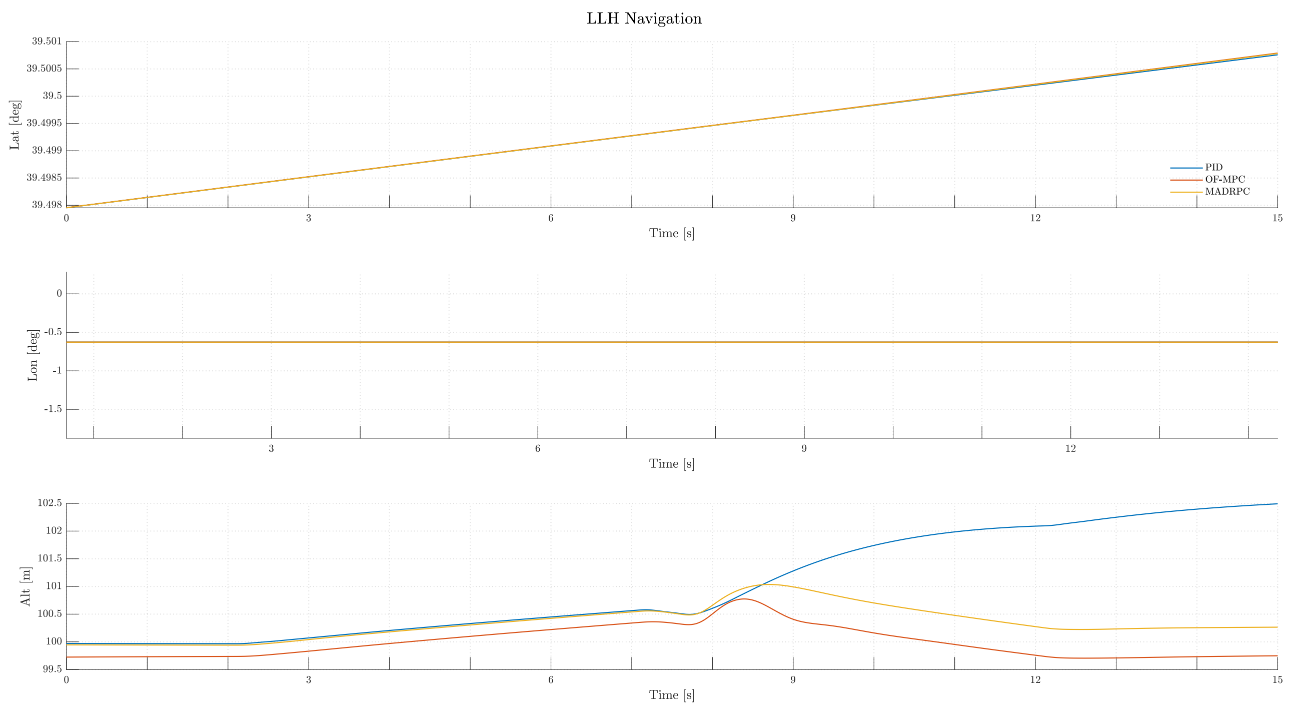


Figure 133: H200 LLH navigation during **Test C.3**

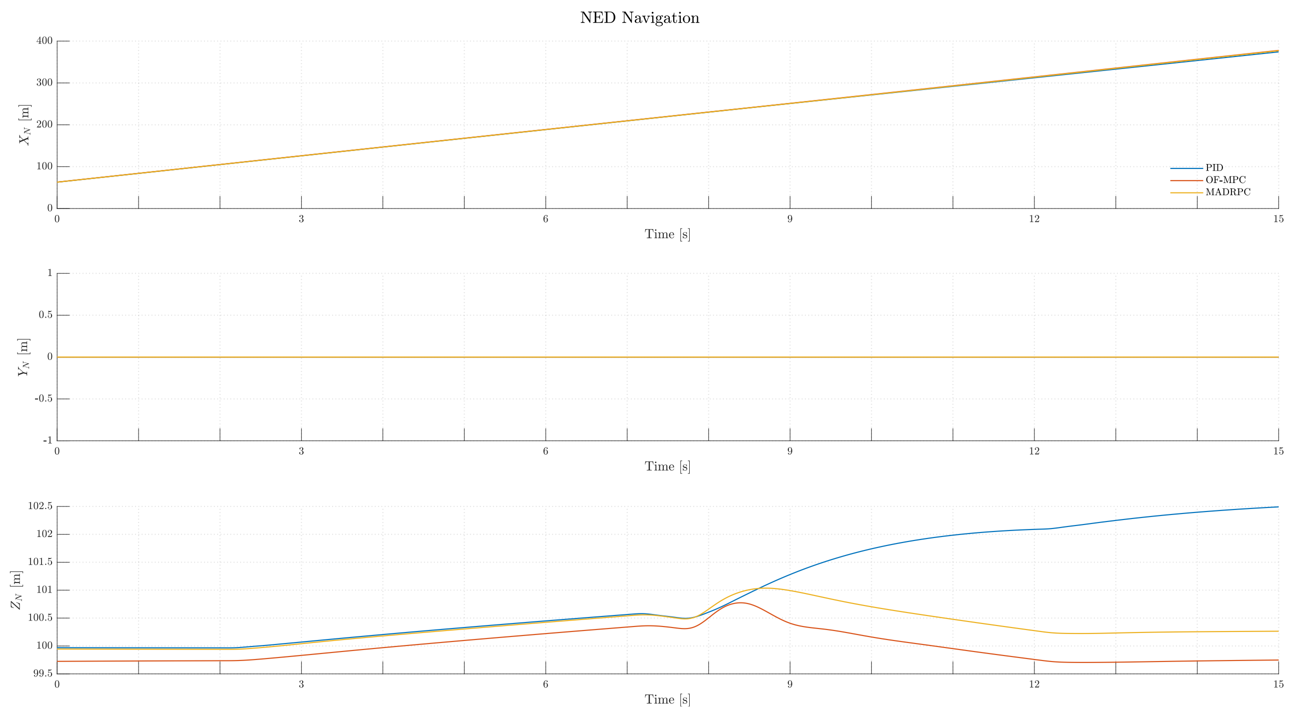


Figure 134: H200 NED navigation during **Test C.3**

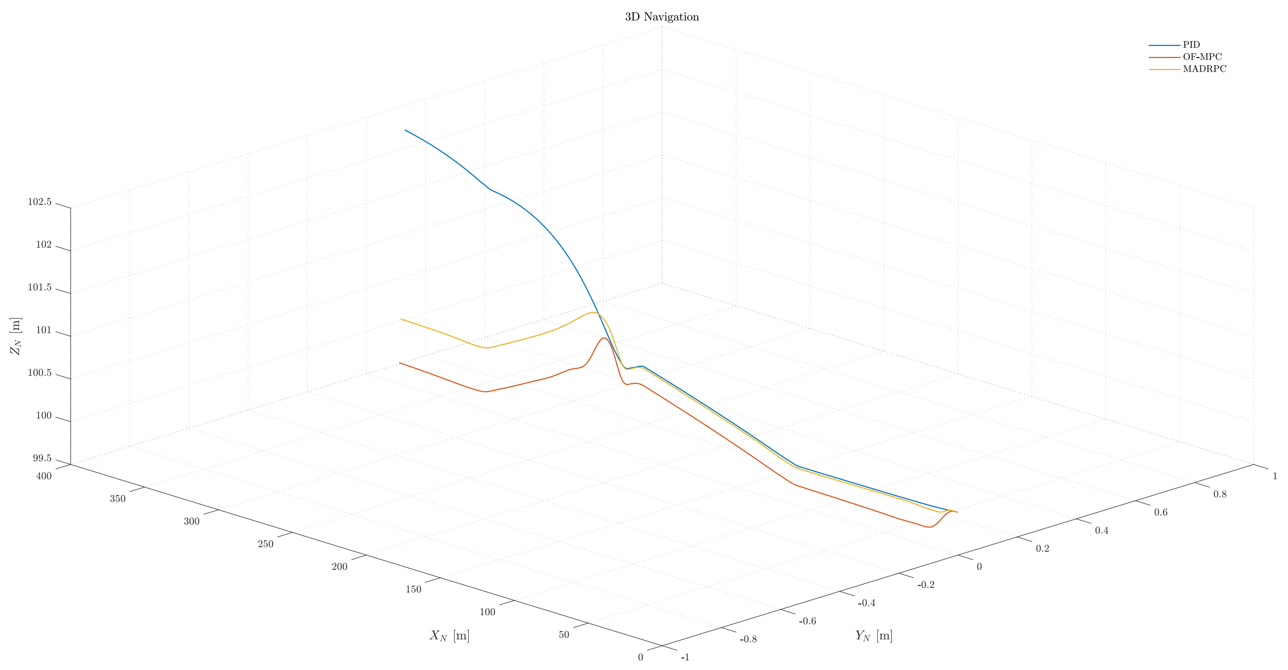


Figure 135: H200 3D navigation during **Test C.3**

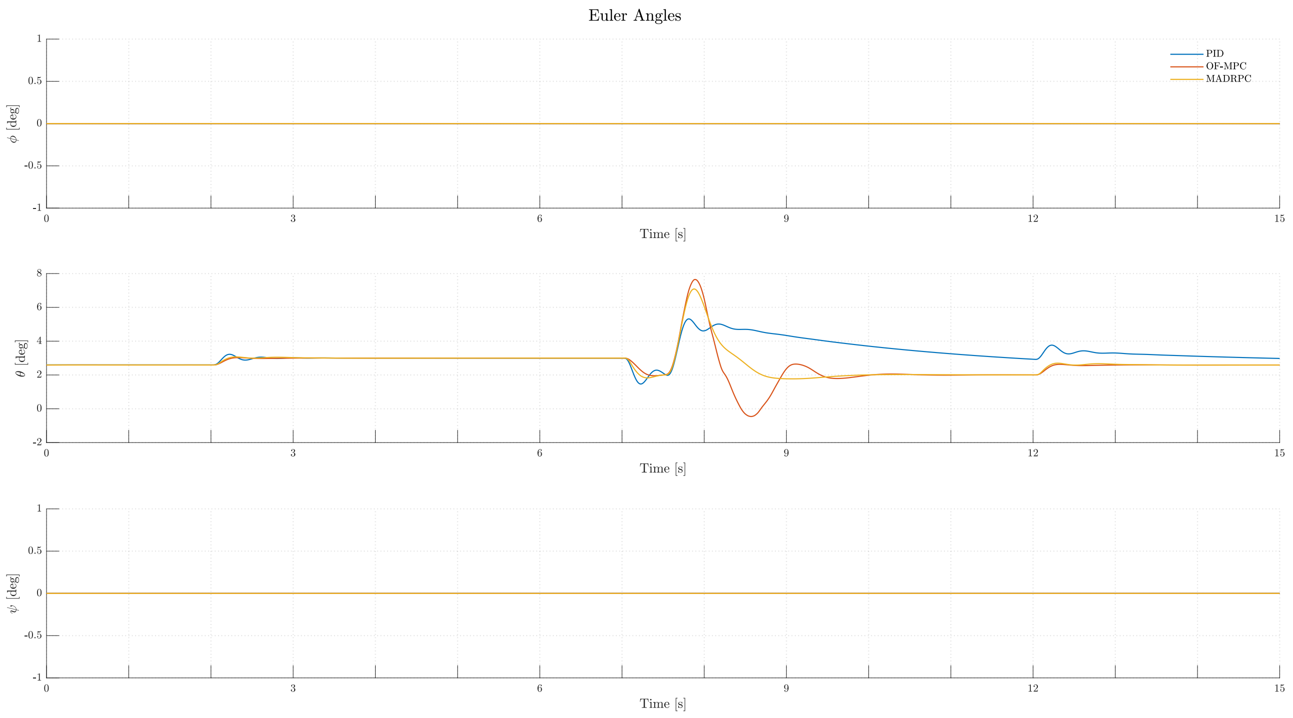


Figure 136: H200 Euler angles during **Test C.3**

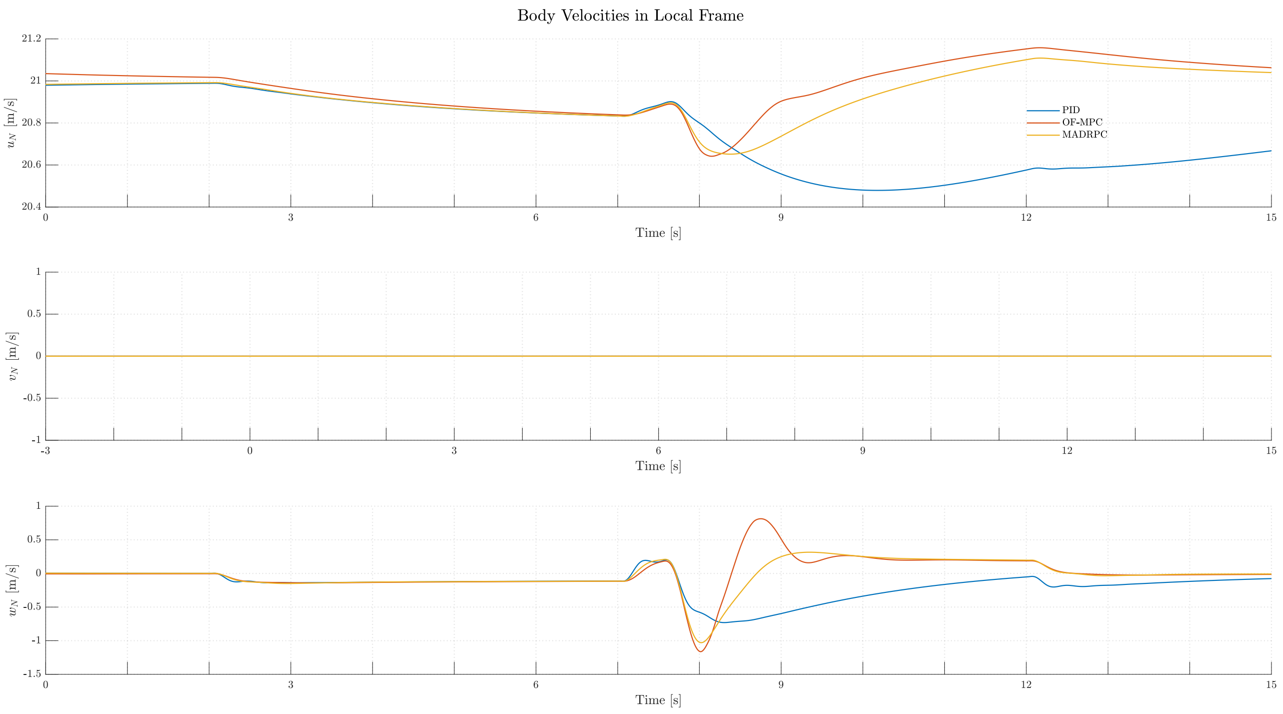


Figure 137: H200 body velocities expressed in local frame during **Test C.3**

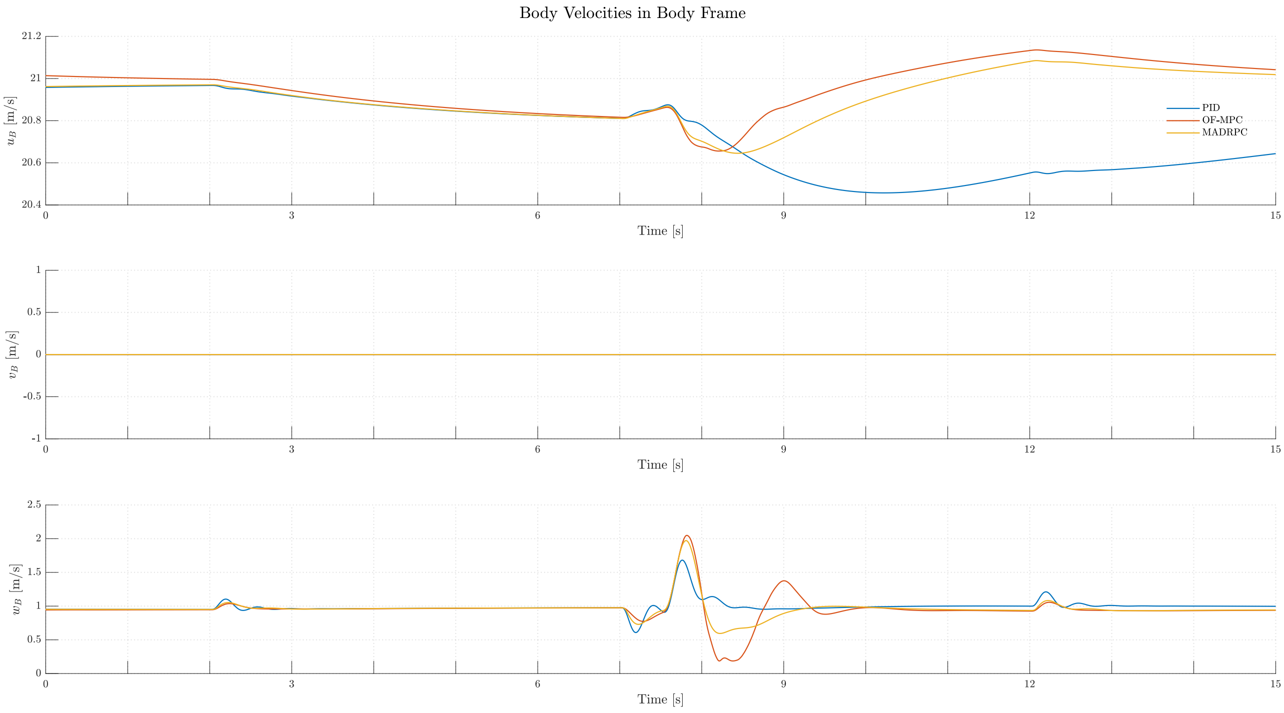


Figure 138: H200 body velocities expressed in body frame for **Test C.3**

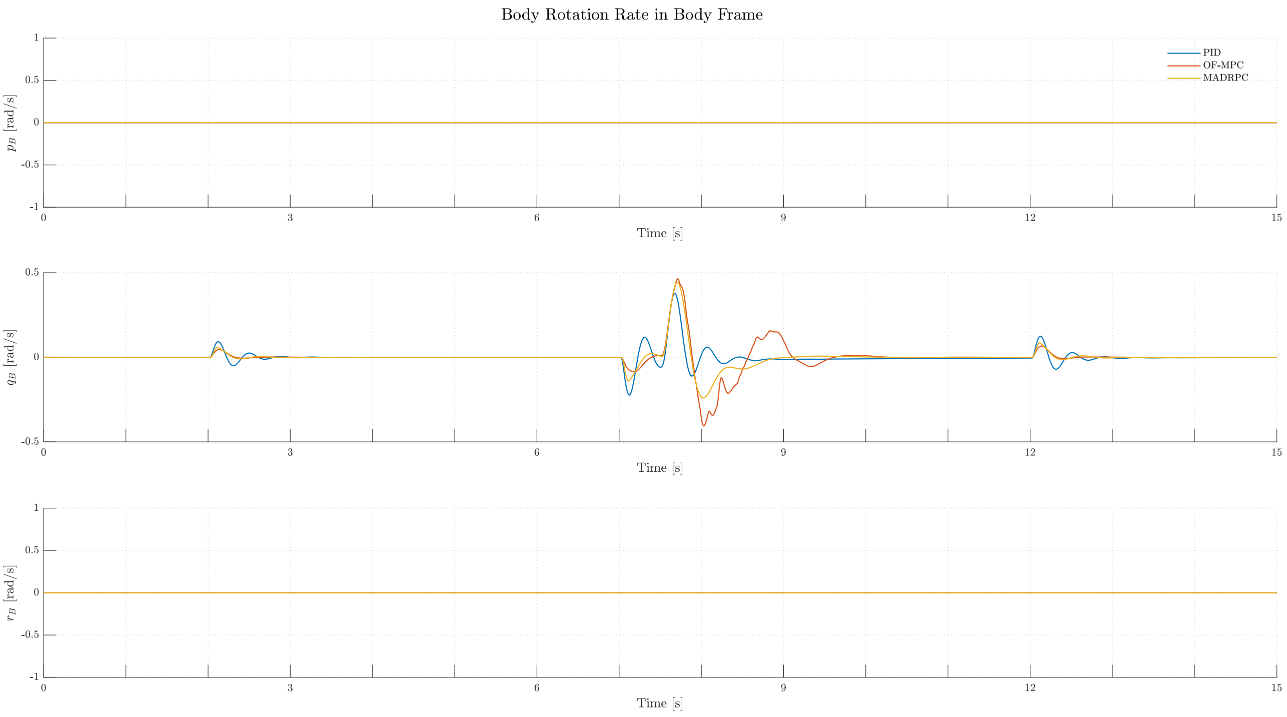


Figure 139: H200 body rotations during **Test C.3**

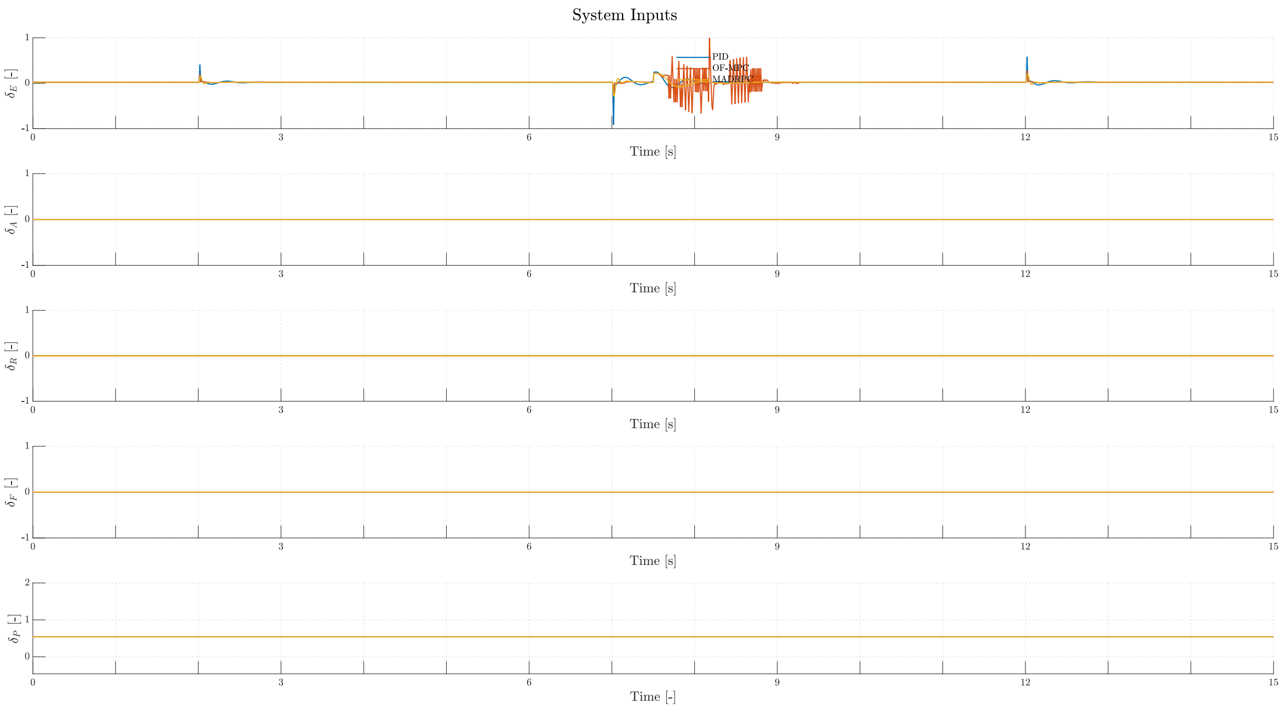


Figure 140: H200 system inputs during **Test C.3**

And the controllers performance indicators are:

	ITAE	ISE	IAE	MSE
PID	179.8525	0.56638	18.0442	0.037758
OF-MPC	53.4546	0.33142	6.5356	0.022095
MADRPC	42.1603	0.25598	5.2321	0.017065

Table 38: Controller performance indicators for **Test C.3**

6.2.2.3.4 Test C.4: Instantaneous Elevator Deflection Signal Perturbation

For the last test of the family, it will be introduced the same elevator deflection signal perturbation, as a step type with the same amplitude as last test, as in equation (105). The difference will reside in that the disturbance will only have a limited duration. So, the perturbation will be introduced at the same time as **Test C.2**:

$$T_{dis,input} = 7.5\,s \tag{107}$$

And the duration of the control action perturbation will be:

$$T_{dur\,input} = 3\,s \tag{108}$$

For the reference pitch signal parameters for **Test C.4**, as with **Test C.3**, will be the same as the controllers validation test, as the perturbation comes from an external input disturbance, not from the inner model:

Symbol	Value	Units
T_1	2	s
T_2	7	s
T_3	12	s
θ_{trim}	2.5905	deg
θ_{ref_1}	3	deg
θ_{ref_2}	2	deg

Table 39: Reference signal parameters for **Test C.4**

So, the test results may be seen from figure (141) to (149):

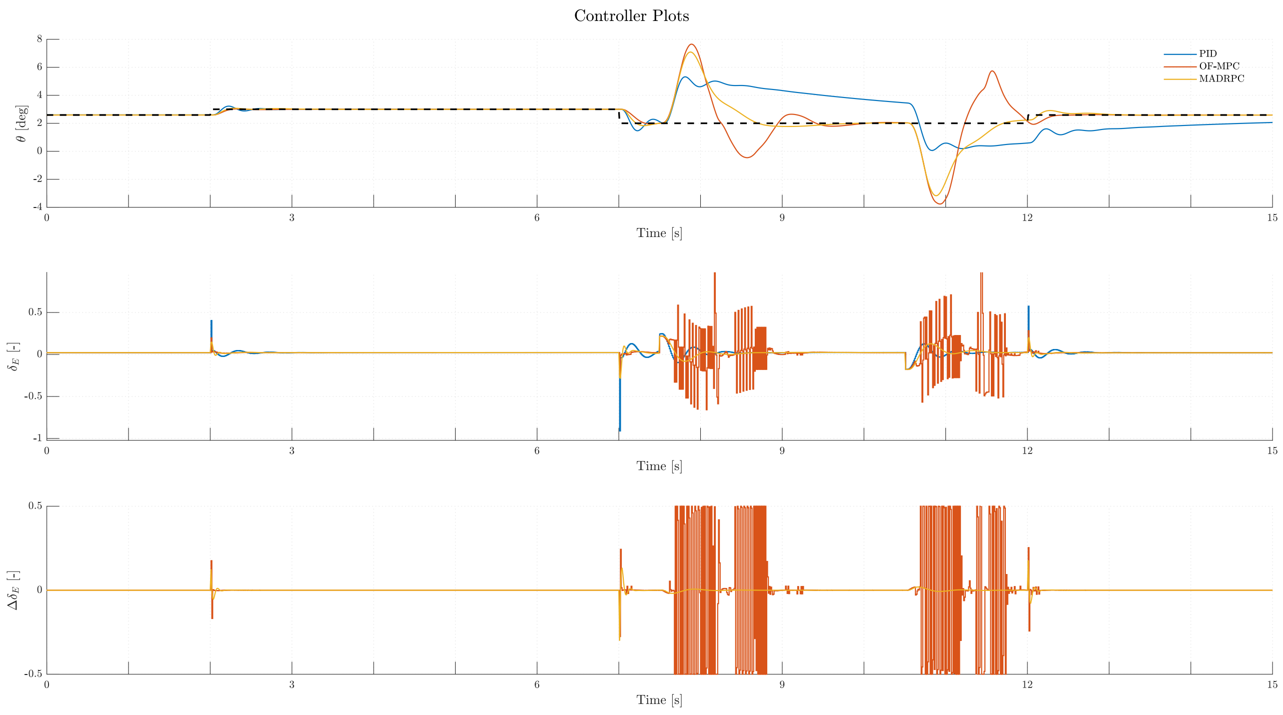


Figure 141: Controller plots during **Test C.4**

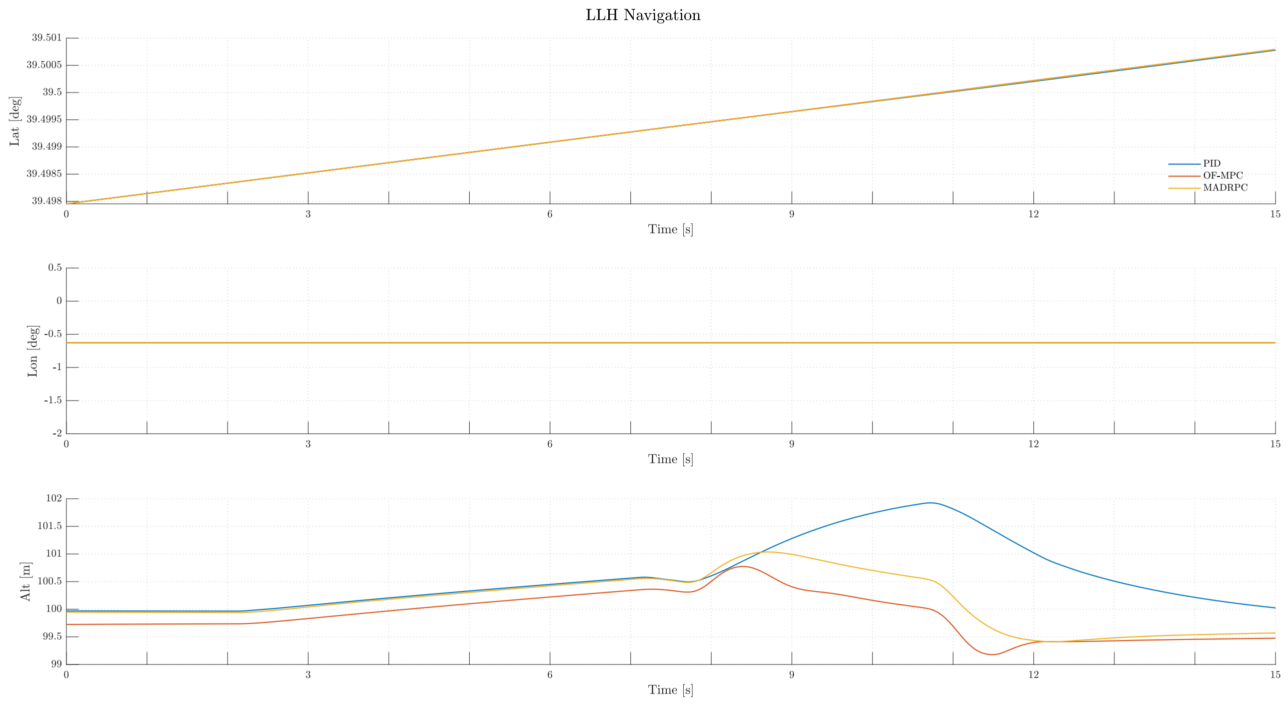


Figure 142: H200 LLH navigation during **Test C.4**

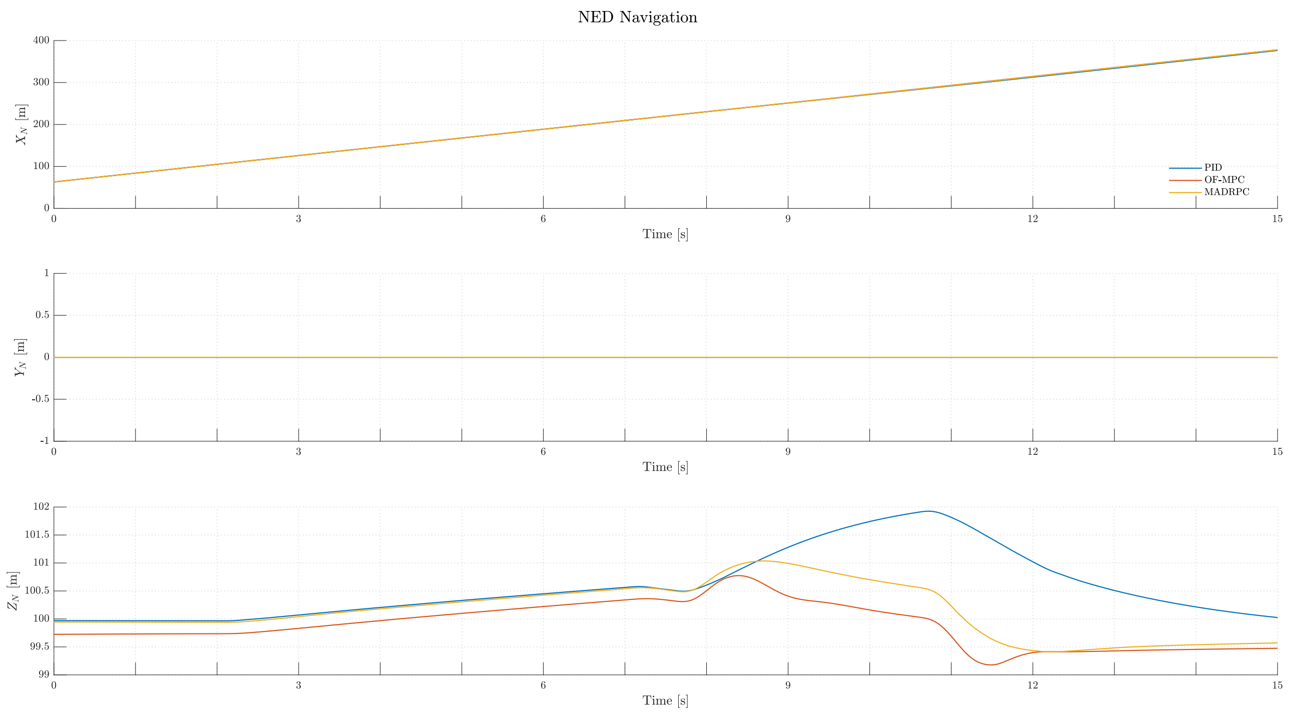


Figure 143: H200 NED navigation during **Test C.4**

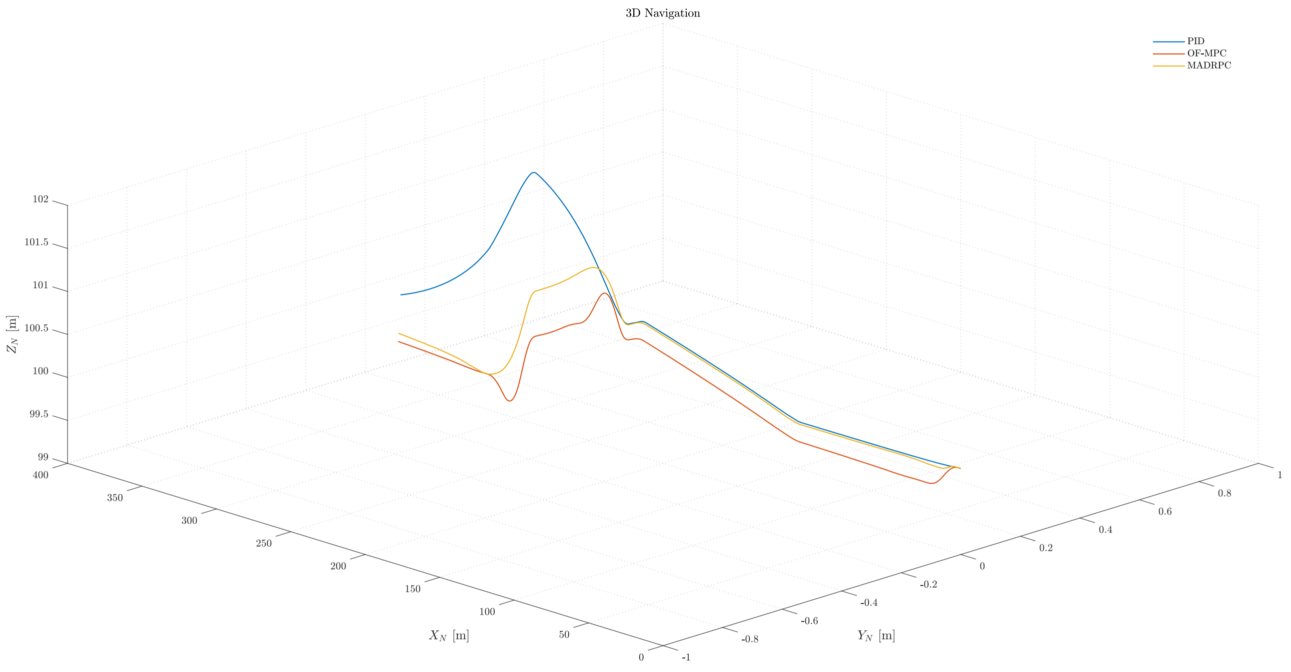


Figure 144: H200 3D navigation during **Test C.4**

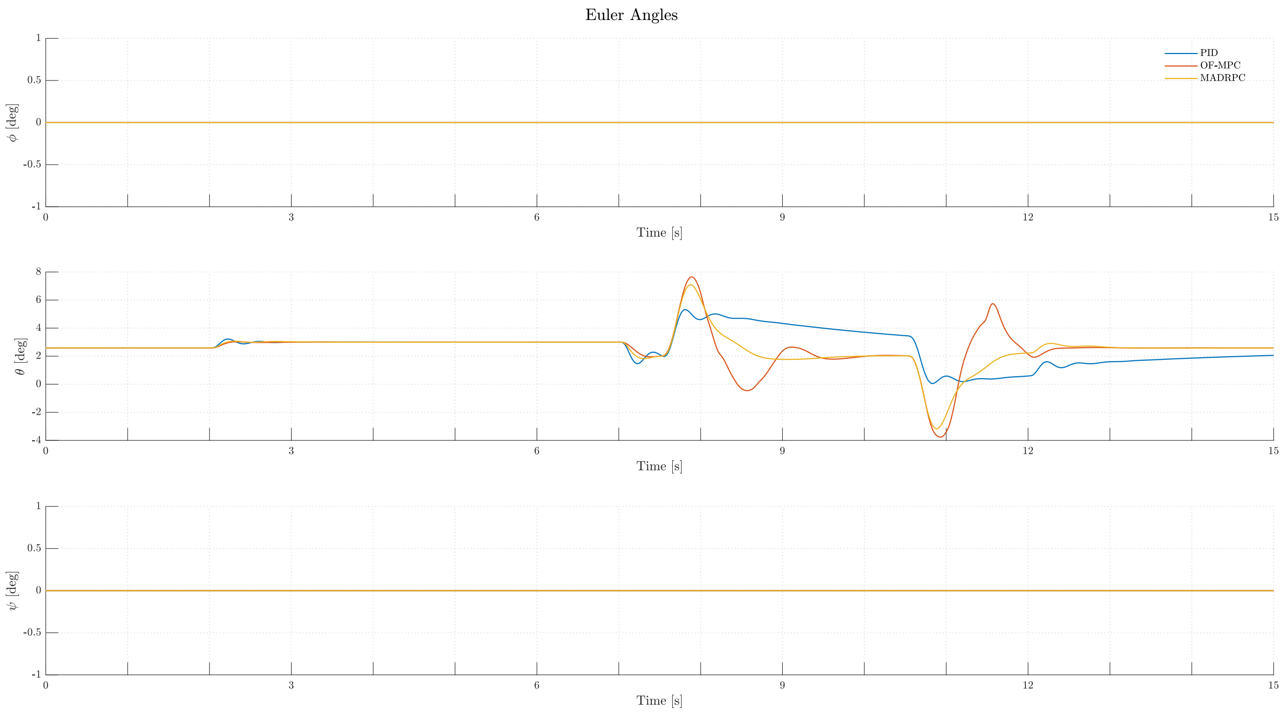


Figure 145: H200 Euler angles during **Test C.4**

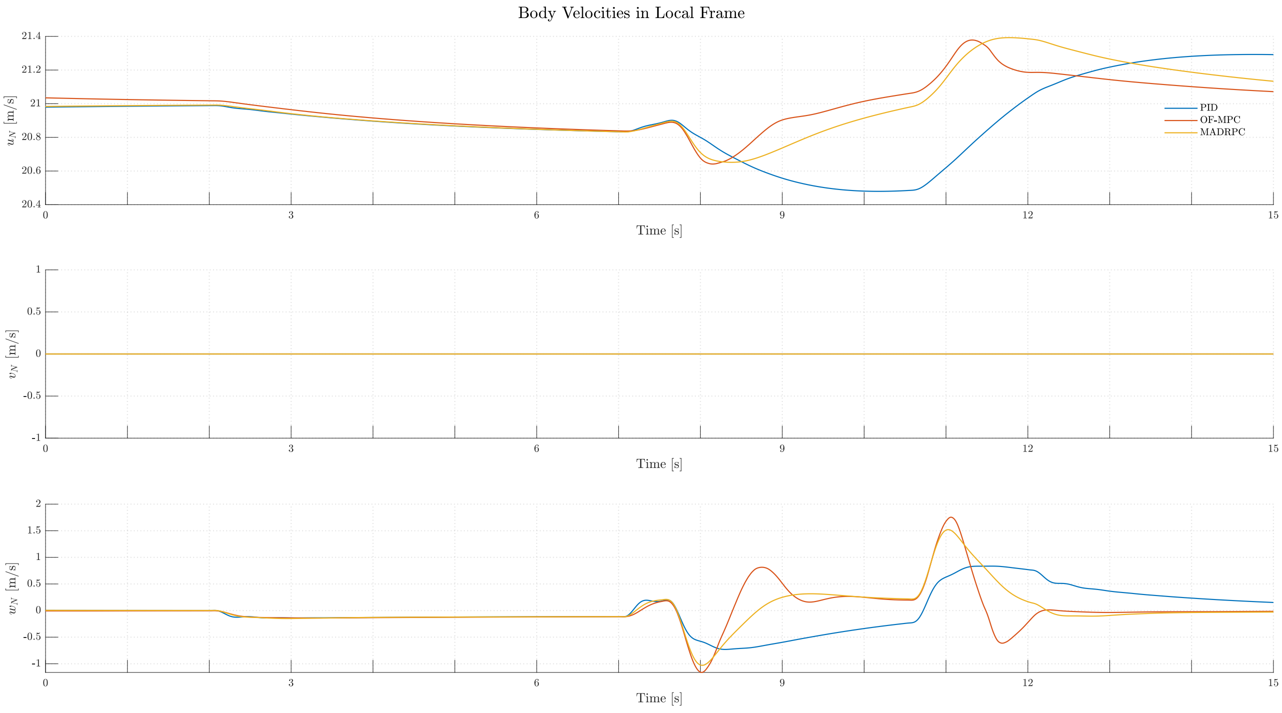


Figure 146: H200 body velocities expressed in local frame during **Test C.4**

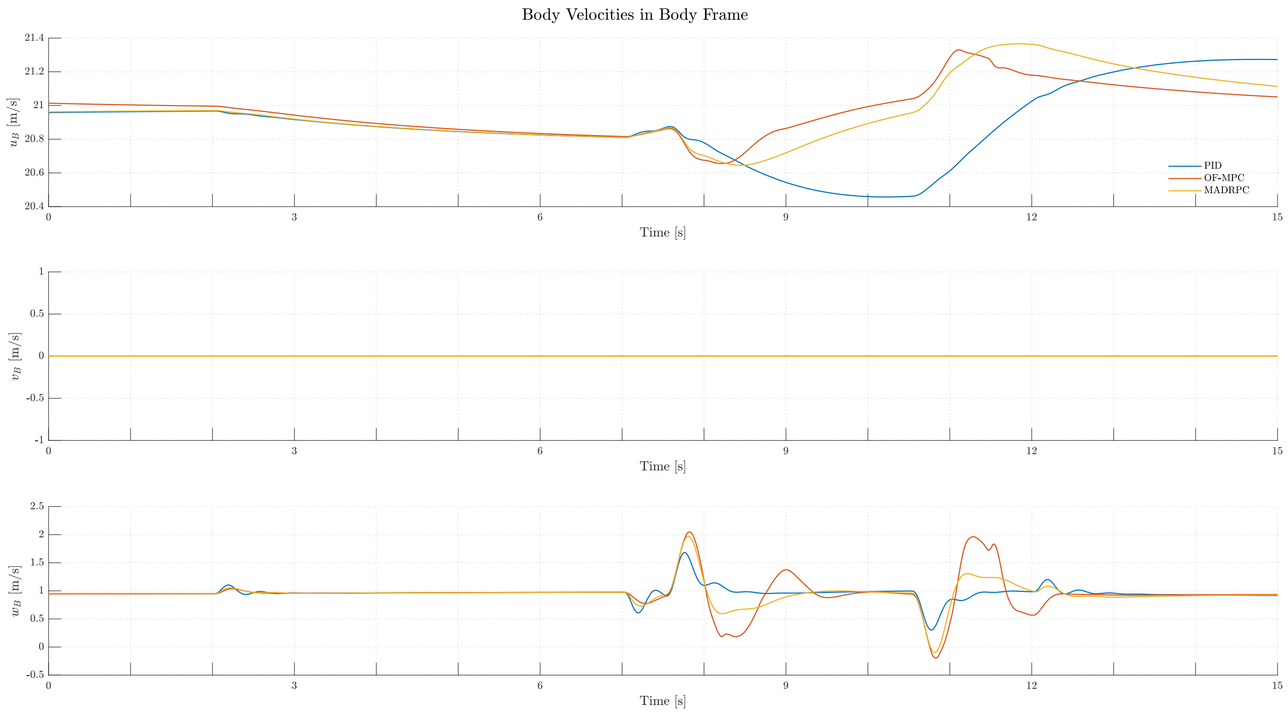


Figure 147: H200 body velocities expressed in body frame for **Test C.4**

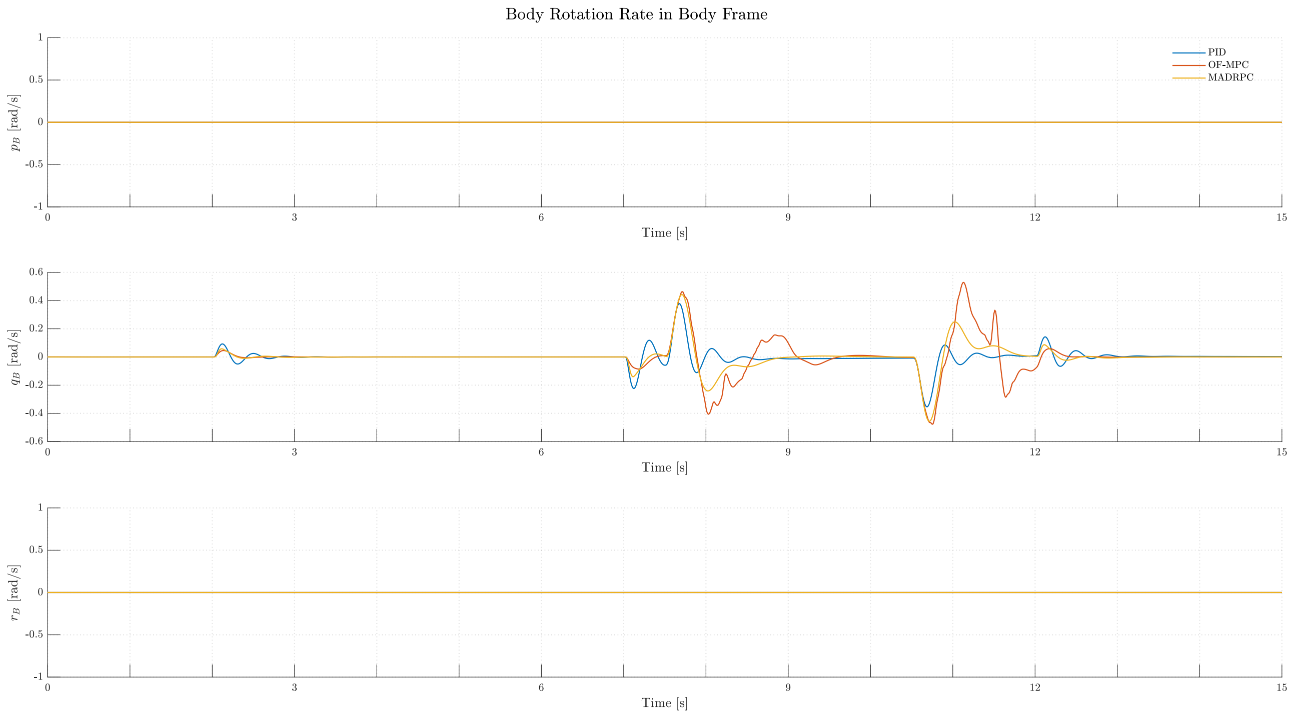


Figure 148: H200 body rotations during **Test C.4**

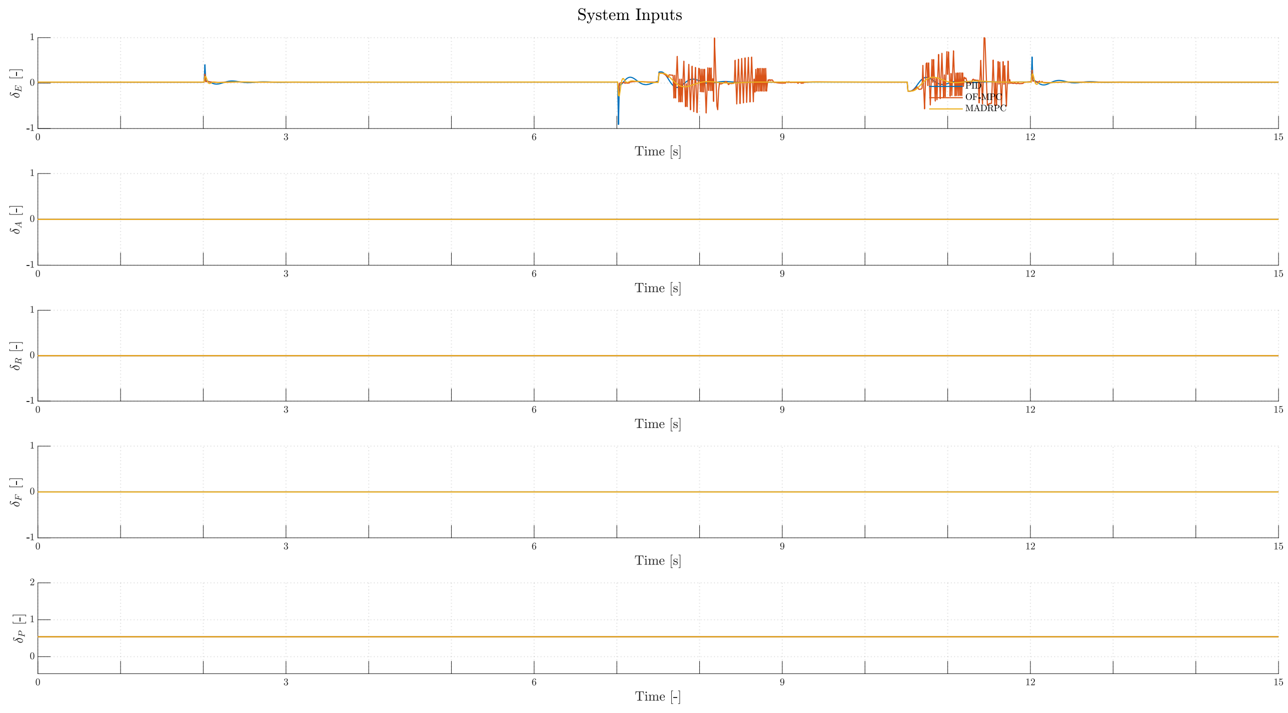


Figure 149: H200 system inputs during **Test C.4**

And the controllers performance indicators are:

	ITAE	ISE	IAE	MSE
PID	210.3902	0.65728	20.4754	0.043819
OF-MPC	124.143	0.73942	12.8549	0.049295
MADRPC	91.1927	0.51298	9.6672	0.034199

Table 40: Controller performance indicators for **Test C.4**

6.2.2.3.5 Test Conclusions

With the last family of tests finished, conclusions will be drawn from them. First of all, as it may be seen from the plots, the controllers are capable of a stable response against the control action perturbations: the wind gust perturbation and the elevator deflection perturbation.

Nevertheless, some key points about their performance will be given. First of all, in figure (150) it may be seen the summary for the performance KPIs controller indicators.

From figure (150), as expected, it may be appreciated a worsening in performance compared with the nominal conditions test. Focusing on the wind perturbation tests, **Test C.1** and **Test C.2**, it may be seen how the OF-MPC is the one losing the most in its performance, worsening all of its indicators, while the PID controller is able not to be affected heavily by the wind gust introduced to the system. For the MADRPC controller, it is able to stay in the middle, indicator-wise, giving a middle ground in performance between the PID controller and the pure predictive controller. This is a remarkable fact, as the MADRPC has no information from the pitch angle plant whatsoever, apart from two natural parameters. With this, it is able to be better than the OF-MPC controller when a wind gust is introduced, as the predictive controller has no external disturbances modeled inside the optimization process. With the MADRPC

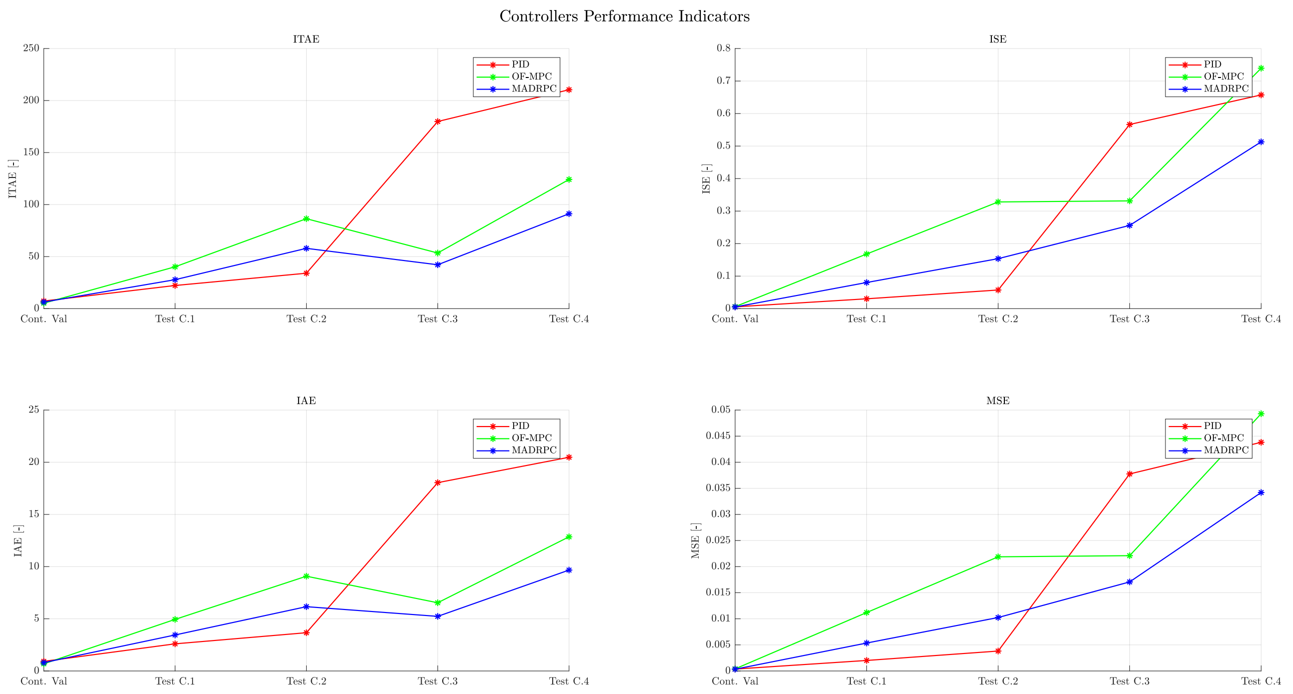


Figure 150: Controllers performance indicators for input perturbation tests

controller, this disturbance can be rejected with its internal disturbance rejection controller, which also compensates the difference between the internal first-order assumed model and the real one. From the other two tests, **Test C.3** and **Test C.4**, when introduced a perturbation inside the control action signal itself, it may be seen how the PID controller has been heavily worsening, increasing all of its indicators, similarly to the OF-MPC controller, worsening its performance when the control action signal is disturbed. It is the MADRPC controller, which has no prior information from the pitch angle model, the one giving the best performance from the three of them.

Now, with the simulation plots, it is going to be given a more controller by controller review. Starting with the PID control, with the wind gusts tests, **Test C.1** and **Test C.2**, it may be seen that it is capable of giving a stable response when the wind gust is introduced. Although it introduces some overshoot, it is able to withstand the perturbation. Additionally, it is able to give this stable response with a smooth control action signal, without saturating any of it. This is a consequence of the robustness of the PID controller against external disturbances. Nevertheless, with the elevator deflection disturbance tests, **Test C.3** and **Test C.4**, it may be appreciated that its performance is heavily worsened, as it is not able to compensate the disturbance introduced inside the elevator deflection signal. It may be appreciated that the pitch angle, from the moment that the disturbance is introduced and when it is removed, is not following anymore the reference signal, even as the control action signal goes to zero, the response signal is unable to track the reference signal. This is due as with the constant elevator signal inside the simulation, the controller is giving more control input to the process than needed, heavily worsening its performance.

Secondly, with the OF-MPC, it may be appreciated a worsening of performance in all tests. With the wind gust tests, **Test C.1** and **Test C.2**, the controller is able to compensate the external disturbance, although it introduces some overshoot into the response, but it returns to the value given by the reference signal. Nevertheless, going through its control action plots, a similar behavior may be seen as with the structural perturbation tests, where, as the disturbance is not taken into account inside the optimization process, the optimized rate of elevator deflection signal are greatly degraded, making the elevator deflection signal also degraded. In both tests, the input are saturated as the prediction model is not taking into account the external disturbance, making the optimized calculated elevator deflection rates not correct, which makes the process overshoot and slower than in nominal conditions. Once again, as it happened with the structural perturbation tests, this elevator deflection greatly impacts the complete flight performance of the H200, as it could take it into stall conditions. With the input signal perturbation tests, **Test C.3** and **Test C.4**, a similar behavior of the controller is appreciated as with the first two tests. Although giving a stable enough response, introducing overshoot into it, the elevator deflection signals are once again heavily worsened. This is a consequence of not having the disturbance not taken into account the prediction model, impacting the optimization process, and making the optimized control action rates not correct with the disturbance introduced, making them saturated and impacting the flight quality.

Lastly, with the MADRPC controller, for the wind gust tests, **Test C.1** and **Test C.2**, it gives a similar performance as the OF-MPC controller, introducing some overshoot into the response, but returning to the reference signal value. From the control action plots, the MADRPC controller is able to output a similar level of performance as the traditional predictive controller, with a smoother control action signal. This is a remarkable fact, as commented previously, this controller does not hold any information of the controlled model, apart from two natural parameters, but unlike the predictive controller having the complete plant inside its optimization process, this controller assumes a first-order process and then compensates the difference between the real and the assumed model. Taking the control action perturbation tests, **Test C.3** and **Test C.4**, again it outputs a similar performance as the traditional predictive controller, with less overshoot to the response, it is able to be the best controller among the three of them, giving the best performance in the response and in the elevator deflection signal. Again, taking into account that this controller does not use the complete model for this to work, and is able to achieve the best performance as these other two controllers that uses the complete information of the plant, it is a very remarkable advantage over these two traditional control schemes.

7 Conclusions

As this project conclusions, the fulfillment of this work has allowed the validation of a new control scheme based on predictive control and disturbance rejection, the MADRPC. It is a new control algorithm where no prior information of the model to be controlled is needed, apart from two natural parameters of the system, the critical gain and the apparent time, being advantageous against traditional forms of control reducing the time spent studying the plant. The MADRPC has been validated for controlling the pitch angle of an unmanned aircraft. Its performance has been compared then with two traditional control algorithms (PID and MPC) in various flight conditions, where the MADRPC controller has matched its performance as with the PID and the MPC controllers, which uses the complete information of the system.

A flight platform has been built for this project in order to validate the MADRPC controller and to be compared with the PID and the MPC controllers. It has been used as reference aircraft the unmanned aircraft H200, an electrical with hydrogen fuel UAV. The first step was to develop the non-linear model governing the complete kinematics and dynamics for the UAV, needing the aircraft geometric information and the propeller and aerodynamic model for flight.

Then, from the non-linear system, it has been extracted the linear plant describing the angle of attack process, using the elevator deflection as the control input. Then, with the angle of attack linear system, it has been used for tuning the three controllers. For the PID and the MPC controllers, the complete linear system is needed either for tuning the control parameters or for being used inside the controller, respectively. For MADRPC, it was not needed the complete description, only two natural parameters from the system to actually tuning it, as the prediction process is used with an assumed first-order system inside.

With the controllers tuned up using the linear plant, they had been used up with the complete non-linear H200 model in order to validate the design. For this, a maneuver has been designed where the H200 is demanded to travel in various pitch angles, both above and below the equilibrium angle of attack. First, the controllers were used in nominal flight conditions, the ones used to calculate the equilibrium angle of attack linear model. Then, a wide range of tests were developed, introducing to the system various perturbations: different airspeed, structural perturbations (different aerodynamics, propeller model and mass) and control input disturbances. Error indicators were used to give a numerical performance result for each of the controllers, as seen in figure (151).

As appreciated in figure (151), when the MADRPC is put against the PID and the MPC controllers, it may be seen that not only is able to match their performance, but even surpass them in some of the tests. It has to be remember that this MADRPC controller has been tuned up and designed using only two natural and superficial parameters from the system, while the other two uses the complete description of the system. This fact proves the controller ability to adapt to external perturbations to the system and its advantage against the traditional control schemes, reducing significantly the time spent for studying and modeling of the system. As this process may be cut out deeply to only extract the system critical gain and the apparent time, in order to build the assumed internal first-order system in the MADRPC.

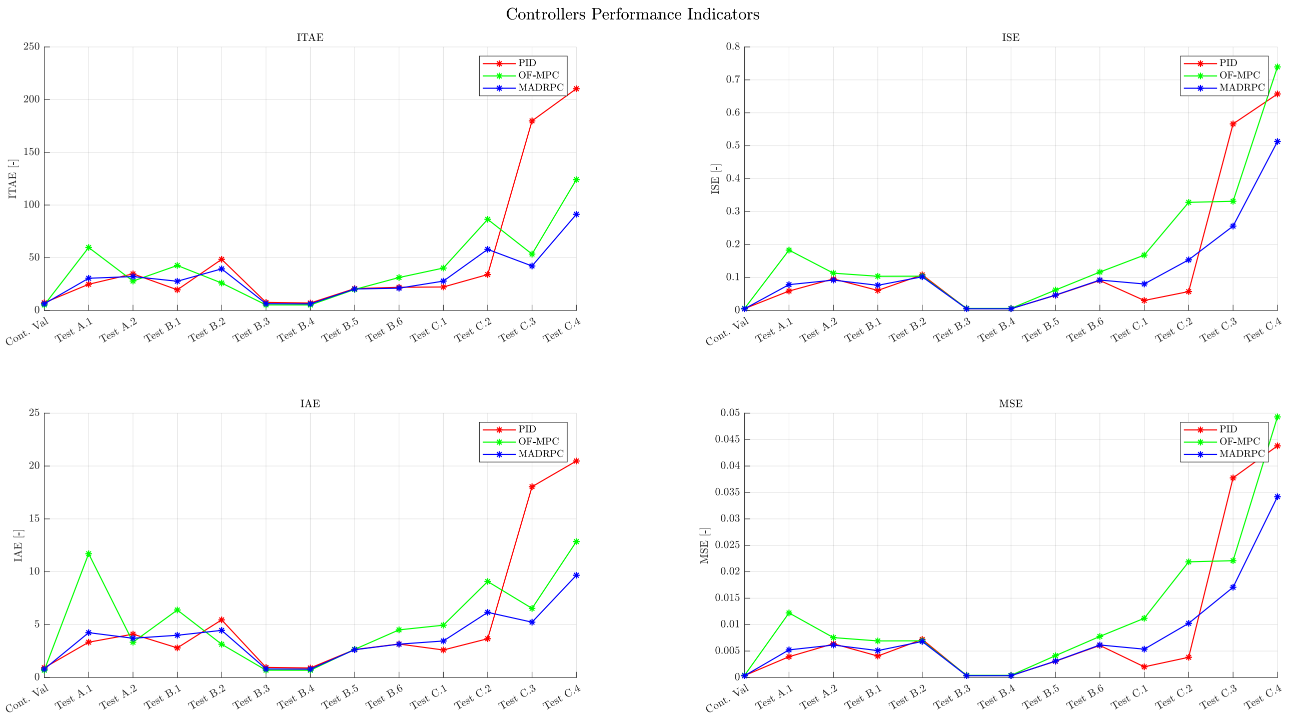


Figure 151: Controllers performance indicators along all tests

To sum up, through the fulfillment of this project, it has been validated a new control scheme, the MADRPC, combining the predictive control and disturbance rejection algorithms, to build a new control law, capable of reducing significantly the amount of time used to describe and model the system. It has been able to match in performance against traditional control schemes (PID and MPC), which requires the full complete system information, as the MADRPC control practically does not need the system model, apart from two easy to calculate natural parameters from the input-output response signal and the complexity coming from the internal optimization problem (inherited from the MPC) does not increase. In fact, as the MADRPC only uses two states to describe the system, the complexity of the optimization problem remains bounded and significantly reduced against higher order systems.

A Annex A: SDG Objectives

This master's thesis SDG objectives [**SDGWebsite**] are:

Objective	**High**	**Mid**	**Low**	**Not Applicable**
No poverty			X	
Zero hunger			X	
Good health and well-being		X		
Quality education			X	
Gender equality			X	
Clean water and sanitation			X	
Affordable and clean energy	X			
Decent work and economic growth		X		
Industry, innovation and infrastructure	X			
Reduced inequalities			X	
Sustainable cities and communities		X		
Responsible consumption and production		X		
Climate action	X			
Life below water		X		
Life on land		X		
Peace, justice and strong institutions			X	
Partnerships for the goals			X	

Table 41: Master's thesis SDGs objectives

This master's thesis is aligned more closely with various of the Sustainable Development Objectives, promoted by the United Nations. More significantly, this project is aligned with the ones about innovation (SDG 9: Industry, Innovation and Infrastructure) and action against climate change (SDG 13: Climate Action)

The main project's goal is the validation of a new control system for digital systems, so new knowledge is introduced inside the industry and it is the first step to adapt it inside the industrial and scientific environments. Moreover, the development of new control systems is vital for a better growth and modernization of current infrastructures and the promotion of sustainable solutions in various situations.

In this project, the work describes the validation of a new control system using an unmanned aircraft, which uses clean and efficient energy, hydrogen. With this, it is reinforced its bond with the SDGs mentioned before. As it is introduced improvements in efficiency and reliability of the control systems, it is promoted a more efficient use of resources, contributing directly in the reduction of greenhouse-effect gases, advancing on the current energy transition.

Bibliography

[1] Salam Ibrahim Khater Abdulla I. Abdulla Mohammed Almaged. “Aircraft Pitch Angle Control Using Pole Placement Approach Based on GA and ABC Optimization Techniques”. In: ().

[2] laith Abualigath. “Optimizing Aircraft Pitch Control Systems: A Novel Approach Integrating Artifical RAbbits Optimizer with PID-F Controller”. In: *International Journal of Robotics and Control Systems* 4.1 (Mar. 2024), pp. 354–364. ISSN: 2775-2658.

[3] *Aircraft Year Book For 1919.* Manufacturers Aircraf Association Inc., 1919.

[4] Mohammed Alhajeri and Masoud Soroush. “Tuning Guidelines for Model-Predictive Control”. In: *Industrial & Engineering Chemistry Research* 59.10 (2020), pp. 4177–4191. DOI: 10.1021/acs.iecr.9b05931. eprint: https://doi.org/10.1021/acs.iecr.9b05931. URL: https://doi.org/10.1021/acs.iecr.9b05931.

[5] Aishwarya Apte et al. “Disturbance-Observer-Based Sensorless Control of PMSM Using Integral State Feedback Controller”. In: *IEEE Transactions on Power Electronics* 35.6 (June 2020), pp. 6082–6090. ISSN: 1941-0107. DOI: 10.1109/TPEL.2019.2949921.

[6] Hanum Arrosida and Mohammad Erik Echsony. “Aircraft pitch control design using observer-state feedback control”. In: *Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control* (2017), pp. 263–272.

[7] Adnan Ashraf et al. “Linear Feedback and LQR Controller Design for Aircraft Pitch Control”. In: *2018 IEEE 4th International Conference on Control Science and Systems Engineering (ICCSSE).* Aug. 2018, pp. 276–278. DOI: 10.1109/CCSSE.2018.8724780.

[8] H.R. Berenji et al. “Pitch control of the space shuttle training aircraft”. In: *IEEE Transactions on Control Systems Technology* 9.3 (May 2001), pp. 542–551. ISSN: 1558-0865. DOI: 10.1109/87.918906.

[9] Steven L. Brunton, Scott T.M. Dawson, and Clarence W. Rowley. “State-space model identification and feedback control of unsteady aerodynamic forces”. In: *Journal of Fluids and Structures* 50 (2014), pp. 253–270. ISSN: 0889-9746. DOI: https://doi.org/10.1016/j.jfluidstructs.2014.06.026. URL: https://www.sciencedirect.com/science/article/pii/S0889974614001467.

[10] Roland Büchi. *State Space Control, LQR and Observer. step by step introduction, with Matlab examples.* en. Norderstedt: Books on Demand, 2010. ISBN: 978-3-8370-2016-8. DOI: 10.3929/ethz-b-000552437.

[11] C.Y. Patil C.B. Kadu. “Design and Implementation of Stable PID Controller for Interacting Level Control System”. In: *7th International Conference on Communication, Computing and Virtualization* (2016).

[12] Rui Cao, Yanbin Liu, and Yuping Lu. “Robust Multiple Model Predictive Control for Ascent Trajectory Tracking of Aerospace Vehicles”. In: *IEEE Transactions on Aerospace and Electronic Systems* 58.2 (Apr. 2022), pp. 1333–1351. ISSN: 1557-9603. DOI: 10.1109/TAES.2021.3117058.

[13] Chris Carpenter. *Flightwise: Principles of Aircraft Flight.* Airlife Publishing Ltd., 1996.

[14] Blanca Viviana Martínez Carvajal. “Design of controllers based on Active DIsturbance Rejection Control (ADRC) and its integration with Model Predictive Control (MPC)”. PhD thesis. Universitat Politècnica de València, Apr. 2023, p. 164.

[15] OpenModelica Consortium. *OpenModelica Official Web*. Online. Accessed Online. Aug. 2024. URL: https://openmodelica.org.

[16] Manas Ranjan Das. "Developing a Generic Purpose OpenModelica Package for Embedded Applications". MA thesis. Department of Chemical Engineering: IIT Bombay, 2019, p. 91. URL: https://www.openmodelica.org/images/M_images/Masterthesis/MTP_Report.pdf.

[17] Stefano Di Cairano and Alberto Bemporad. "Model Predictive Control Tuning by Controller Matching". In: *IEEE Transactions on Automatic Control* 55.1 (Jan. 2010), pp. 185–190. ISSN: 1558-2523. DOI: 10.1109/TAC.2009.2033838.

[18] William Edwards et al. "Automatic Tuning for Data-driven Model Predictive Control". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. May 2021, pp. 7379–7385. DOI: 10.1109/ICRA48506.2021.9562025.

[19] Mariano I. Lizarraga Fernandez. "Design, implementation and flight verification of a versatile and rapidily reconfigurable UAV GNC research platform". PhD thesis. University of California, Dec. 2009.

[20] Raony M. Fontes, Márcio A. F. Martins, and Darci Odloak. "An Automatic Tuning Method for Model Predictive Control Strategies". In: *Industrial & Engineering Chemistry Research* 58.47 (2019), pp. 21602–21613. DOI: 10.1021/acs.iecr.9b03502. eprint: https://doi.org/10.1021/acs.iecr.9b03502. URL: https://doi.org/10.1021/acs.iecr.9b03502.

[21] Joan Albert Such García. "Preliminary design and development management of anUnmanned Air Vehicle with distributed propulsion and ahydrogen fuel cell". In: *Master Thesis, Master's Degree in Aeronautical Engineering, UPV* (Sept. 2023), p. 222.

[22] Jorge L. Garriga and Masoud Soroush. "Model Predictive Control Tuning Methods: A Review". In: *Industrial & Engineering Chemistry Research* 49.8 (2010), pp. 3505–3515. DOI: 10.1021/ie900323c. eprint: https://doi.org/10.1021/ie900323c. URL: https://doi.org/10.1021/ie900323c.

[23] P Gopi et al. "Model based Tuning of PID Controller". In: *Journal of control and instrumentation* 4 (Jan. 2013), p. 16.

[24] Tonoy Chowdhury Gp Capt Atul Garg Rezawana Islam Linda. "Evolution of Aircraft Flight Control System and Fly-by-Light Flight Control System". In: *International Journal of Emerging Technology and Advanced Engineering* 3.12 (2013). ISSN: 2250-2459.

[25] Michael Harmse et al. "Robust optimization-based multi-loop PID controller tuning: A new tool and an industrial example". In: *IFAC Proceedings Volumes* 42.11 (2009). 7th IFAC Symposium on Advanced Control of Chemical Processes, pp. 548–553. ISSN: 1474-6670. DOI: https://doi.org/10.3182/20090712-4-TR-2008.00088. URL: https://www.sciencedirect.com/science/article/pii/S1474667015303311.

[26] S. Heise and J. Maciejowski. "Model predictive control of a supermaneuverable aircraft". In: *Guidance, Navigation, and Control Conference*. DOI: 10.2514/6.1996-3768. eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.1996-3768. URL: https://arc.aiaa.org/doi/abs/10.2514/6.1996-3768.

[27] KS Holkar and Laxman M Waghmare. "An overview of model predictive control". In: *International Journal of control and automation* 3.4 (2010), pp. 47–63.

[28] A. Idir et al. “Performance improvement of aircraft pitch angle control using a new reduced order fractionalized PID controller”. In: *Asian Journal of Control* 25.4 (2023), pp. 2588–2603. DOI: `https://doi.org/10.1002/asjc.3009`. eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1002/asjc.3009`. URL: `https://onlinelibrary.wiley.com/doi/abs/10.1002/asjc.3009`.

[29] Davut Izci et al. “HHO Algorithm based PID Controller Design for Aircraft Pitch Angle Control System”. In: *2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*. June 2020, pp. 1–6. DOI: `10.1109/HORA49412.2020.9152897`.

[30] Christoffer Fors Johansson. “OpenModelica Interactive Simulation using an OPC UA client”. Bachelors thesis. Department of Computer and Information Science: Linköping University, June 2017. URL: `http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-139085` (visited on 11/03/2017).

[31] Yun Li Kiam Heong Ang Gregory Chong. “PID Control System Analysis, Design and Technology”. In: *IEEE Transactions on Control Systems Technology* 13.4 (July 2005).

[32] S. Senthil Kumar. *Design of PID Controller using Root Locus method*. Dept. of Aeronautical Engineering. KCT, Coimbatore 641049.

[33] et al Kurucsó. “State space control of quadratic boost converter using LQR and LQG approaches”. In: *2015 Intl Aegean Conference on Electrical Machines Power Electronics (ACEMP), 2015 Intl Conference on Optimization of Electrical Electronic Equipment (OPTIM) 2015 Intl Symposium on Advanced Electromechanical Motion Systems (ELECTROMOTION)*. Sept. 2015, pp. 642–648. DOI: `10.1109/OPTIM.2015.7427003`.

[34] Haider Al-Lami. “The Evolution of Flight Control Systems Technology Development, System Architecture and Operation”. In: (2015).

[35] Yan Lan and Minrui Fei. “Design of state-feedback controller by pole placement for a coupled set of inverted pendulums”. In: *IEEE 2011 10th International Conference on Electronic Measurement and Instruments*. Vol. 3. Aug. 2011, pp. 69–73. DOI: `10.1109/ICEMI.2011.6037857`.

[36] J. Lévine. “Are there New Industrial Perspectives in the Control of Mechanical Systems?” In: *Centre Automatique et Systèmes* (Jan. 1999).

[37] James Jizhi Li. “Requirement Verification in Modelica For a Small Scale Network Simulation”. MA thesis. Department of Computer and Information Science: Linköping University, June 2015.

[38] H Lukman. “Enhancing the Stabilization of Aircraft Pitch Motion Control Via Intelligent and Classical Method”. In: (2017).

[39] Arak M. “PID Control”. In: *CONTROL SYSTEMS, ROBOTICS, AND AUTOMATION* II ().

[40] Blanca Viviana Martínez Carvajal et al. “Modified Active Disturbance Rejection Predictive Control: A fixed-order statespace formulation for SISO systems”. In: *ISA Transactions* 142 (Nov. 2023), pp. 148–163. ISSN: 0019-0578. DOI: `10.1016/j.isatra.2023.08.011`.

[41] MathWorks. *Custom Variable Mass 6DOF (Quaternion)*. Online. URL: `https://es.mathworks.com/help/aeroblks/customvariablemass6dofquaternion.html`.

[42] Mathworks. *MATLAB Official Webpage*. Online. Accessed online. Aug. 2024.

[43] P. M. Meshram and Rohit G. Kanojiya. “Tuning of PID controller using Ziegler-Nichols method for speed control of DC motor”. In: *IEEE-International Conference On Advances In Engineering, Science And Management (ICAESM -2012)*. 2012, pp. 117–122.

[44] Carlos Alfano Neto and Marcelo Embiruçu. “Tuning of PID Controllers: An Optimization-Based Method”. In: *IFAC Proceedings Volumes* 33.4 (2000). IFAC Workshop on Digital Control: Past, Present and Future of PID Control, Terrassa, Spain, 5-7 April 2000, pp. 367–372. ISSN: 1474-6670. DOI: `https://doi.org/10.1016/S1474-6670(17)38271-X`. URL: `https://www.sciencedirect.com/science/article/pii/S147466701738271X`.

[45] M. Angelin Ponrani and A. Kirthini Godweena. “Aircraft Pitch Control using PID Controller”. In: (July 2021), pp. 1–4. DOI: `10.1109/ICSCAN53069.2021.9526373`.

[46] M Prasad and M Inayathullah. “Root locus approach in design of PID controller for cruise control application”. In: *Journal of Physics: Conference Series* 2115 (Nov. 2021), p. 012023. DOI: `10.1088/1742-6596/2115/1/012023`.

[47] APC Propellers. *13x6.5 (F2B) propeller specifications*. Online. Available at: https://www.apcprop.com duct/13x6-5ef2b/. URL: `https://www.apcprop.com/product/13x6-5ef2b/`.

[48] *Python’s Official Webpage*. Online. Accessed Online. Aug. 2024.

[49] Joe Qin and Thomas Badgwell. “An Overview Of Industrial Model Predictive Control Technology”. In: *AIChE Symposium Series* 93 (Jan. 1997).

[50] MBB Sharifian, R Rahnavard, and H Delavari. “Velocity control of DC motor based intelligent methods and optimal integral state feed back controller”. In: *International Journal of Computer theory and engineering* 1.1 (2009), p. 81.

[51] C. Shearer and S. Heise. “Constrained model predictive control of a nonlinear aerospace system”. In: *Guidance, Navigation, and Control Conference and Exhibit*. DOI: `10.2514/6.1998-4235`. eprint: `https://arc.aiaa.org/doi/pdf/10.2514/6.1998-4235`. URL: `https://arc.aiaa.org/doi/abs/10.2514/6.1998-4235`.

[52] Nicke Sievert. “Modelica Models in a Distributed Environment Using FMI and HLA”. MA thesis. Department of Computer and Information Science: Linköping University, Dec. 2015. URL: `http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-131385` (visited on 11/03/2017).

[53] Brian L. Stevens. *Aircraft control and simulation*. Ed. by Frank L. Lewis. A @Wiley-Interscience publication. Literaturangaben. New York [u.a.]: Wiley, 1992. 617 pp. ISBN: 0471613975.

[54] Try Susanto et al. “Application of Unmanned Aircraft PID Control System for Roll, Pitch and Yaw Stability on Fixed Wings”. In: *2021 International Conference on Computer Science, Information Technology, and Electrical Engineering (ICOMITEE)*. Oct. 2021, pp. 186–190. DOI: `10.1109/ICOMITEE53461.2021.9650314`.

[55] T-Motor. *AT3520 Long Shaft Fixed Wing UAV*. Online. Available at: https://store.tmotor.com/goods-794-AT3520+Long+Shaft.html.

[56] José Domingo Cerdán Torres. “Design, analysis and manufacturing of an UnmannedAircraft with distributed propulsion and Hydrogen Fuel Cellpower supply”. In: *Master Thesis, Master’s Degree in Aeronautical Engineering* (Sept. 2023), p. 250.

[57] A. Visioli. “Model-based PID tuning for high-order processes: when to approximate”. In: *Proceedings of the 44th IEEE Conference on Decision and Control*. Dec. 2005, pp. 7127–7132. DOI: `10.1109/CDC.2005.1583310`.

[58] T. Yucelen, O. Kaymakci, and S. Kurtulan. “Self-Tuning PID Controller Using Ziegler-Nichols Method for Programmable Logic Controllers”. In: *IFAC Proceedings Volumes* 39.14 (2006). 1st IFAC Workshop on Applications of Large Scale Industrial Systems, pp. 11–16. ISSN: 1474-6670. DOI: `https://doi.org/10.3182/20060830-2-SF-4903.00003`. URL: `https://www.sciencedirect.com/science/article/pii/S1474667015325866`.

[59] Ridong Zhang, Anke Xue, and Furong Gao. *Model Predictive Control Approaches Based on the Extended State Space Model and Extended Non-minimal State Space Model. Approaches based on the extended state space model and extended non-minimal state space model*. Ed. by Anke Xue and Furong Gao. Literaturangaben. Singapore: Springer, 2019. 137 pp. ISBN: 9789811300820.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCOLA TÈCNICA SUPERIOR DE ENGINYERIA INDUSTRIAL

Máster en Automática e Informática Industrial

"Design and Implementation of Angle-of-Attack Control System Using Active Disturbance Rejection Control Algorithms for Fixed-Wing Unmanned Aerial Vehicles"

FINAL MASTER'S THESIS
Budget

Autor:
José Luis Musoles Aguña

Profesor:
Sergio García-Nieto Rodríguez

València; 2023 - 2024

Contents

1 Introduction

The Budget is the document used for the prevision plan or as the baseline to determine the line of action. The main goal is to collect the future economic activities forecast and the result prediction. More in particular, this present Budget has the objective of providing all the detailed information about the associated costs to the project execution.

In order to organize the expenses inside the project, it is going to be divided in three work units:

1. Development of the Non-Lineal Model (W1).
2. Implementation in MATLAB and Simulink (W2).
3. Tests running and validation of results (W3).

For each of the work units before stated, it is going to be analyzed the following economical aspects:

- Workforce costs: It is the costs generated by the employees who have taken part during the project execution. It may be divided in three categories attending to its function:
  - Technical: Responsible for the designed systems development or the required tests.
  - Engineering: Responsible for the study, the conceptual development for the software implementation and the tests design.
  - Management: Responsible for the project direction.
- Material costs: They are the costs associated with the acquisition of the required materials for the project execution.
- Software licenses costs: They are the costs associated with the use of the different software required for the project execution.

2 Workforce Price Table

The objective of this section is to calculate the generated cost due to the employees who have taken part in the project execution. The workforce may be divided in engineering and technical according to the type of the performed or proposed tasks.

It will be considered the following costs per hour attending to the type of employee:

Workforce	**Cost [€/h]**
Technical	10.00
Engineering	25.00

Table 1: Price table depending on the employee type

Having established the different prices attending to the employee type, here is the final table of the estimated hours to achieve each of the work units:

Workforce	**W1**	**W2**	**W3**	**Total [€]**
Technical	5	0	0	50.00
Engineering	290	25	25	8500.00
Total [€]	7300.00	625.00	625.00	8550.00

Table 2: Price table depending on the employee type and the work unit

3 Material Price Table

The objective of this section is to calculate the generated costs due to the materials and equipment used during the project execution. The amortization time for both of the computers will be of 4 years giving information for calculating the costs per hour:

Device	**Total Cost [€]**	**Cost [€/h]**	**Usage Time [h]**	**Total [€]**
Working Computer	1800.00	0.051	240	12.33
Laptop	1000.00	0.029	100	2.85
Total	2800.00	0.080	340	15.18

Table 3: Equipment price table for the project execution

So, the materials needed for the execution of the project and its costs are:

Material	**Cost [€]**
Equipment	15.18
Bibliography, research papers...	0.00
Total [€]	15.18

Table 4: Material price table for the project execution

4 Software Licenses Price Table

Regarding this section, the objective is to calculate the costs generated by the purchase and use of the different software used during the project execution. These licenses have to be installed in each of the employee's working computers. The working hours may be reduced by contracting more employees, until a certain limit. For this project, it will be sufficient with only one working computer, with one employee, with the necessary software licenses installed on it. The software licenses used for the project execution are:

License	**Annual Cost [€]**
MATLAB and Simulink	800.00
Total [€]	800.00

Table 5: Price table for the annual software license costs

As the project has a duration of a full year, these are the cost for the software licenses.

5 Execution Budget

The following price table sums up the partial budget for the project execution. It is divided into the three work units, in order to differentiate the expenses between them:

Cost Type	**W1 [€]**	**W2 [€]**	**W3 [€]**	**Total [€]**
Workforce	7300.00	625.00	625.00	8550.00
Material	23.17	2.00	2.00	27.17
Software Licenses	266.67	266.67	266.67	800.00
Total [€]	7589.84	893.66	893.66	9377.17

Table 6: Price table for the partial budget of project execution

Nevertheless, for the project execution, it is wise to overestimate the budget in order to take into account general expenses. They will be set as the 15% of the total budget. In addition, it will be considered a 6% as industrial benefits. With all of this, it is obtained and estimation for the execution budget:

Expenses [€]	**W1 [€]**	**W2 [€]**	**W3 [€]**	**Total [€]**
Material Execution Sum [€]	7589.84	893.66	893.66	9377.17
15% General Costs	1138.48	134.05	134.05	1406.58
6% Industrial Benefit	455.39	53.62	53.62	562.63
Investment Budget	9183.71	1081.33	1081.33	11346.37
21% VAT	1928.58	227.08	227.08	2382.74
Initial Bidding Budget	11112.29	1308.41	1308.41	13729.11

Table 7: Final price table for the project execution

So, the expenses for the project execution amount to:

- The Material Execution Sum amounts to

  NINE THOUSAND THREE HUNDRED SEVENTY-SEVEN EUROS AND SEVENTEEN CENTS

- The Investment Budget amounts to

  ELEVEN THOUSAND THREE HUNDRED AND FORTY-SIX EUROS AND THIRTY-SEVEN CENTS

- The Initial Bidding Budget amounts to

  THIRTEEN THOUSAND SEVEN HUNDRED AND TWENTY-NINE EUROS AND ELEVEN CENTS