UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Design and Simulation of the guidance and control system for an F-86 Sabre

Final Bachelor's Degree Project

Author: Álvaro Ortiz Moya
Tutor: Sergio García-Nieto Rodríguez

Aerospace Engineering Bachelor's Degree

Escuela Técnica Superior de Ingeniería del Diseño
Universitat Politècnica de València
Spain, 15th September 2020

This document is composed of the following sections:

Project

Budget of the project

Project specification

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Design and Simulation of the guidance and control system for an F-86 Sabre

Project

Author: Álvaro Ortiz Moya
Tutor: Sergio García-Nieto Rodríguez

Aerospace Engineering Bachelor's Degree

Escuela Técnica Superior de Ingeniería del Diseño
Universitat Politècnica de València
Spain, 15th September 2020

Abstract

Nowadays, autonomous navigation systems constitute a great focus of innovation and investigation. These systems must provide the needed data to understand the route at each flight mission properly and generate the required control signals which guide an aircraft. Therefore, the aim of this project is to develop an application capable of conducting the dynamic path planning, guidance and control over an aircraft to accomplish the objectives of a mission example.

To achieve such goal, algorithms based on methodologies such as the Recursive Rewarding Adaptive Cell Decomposition and 3D Smooth Path Planning are employed, which on the basis of the mission's constraints, generate a suitable route to the final destination. Afterwards, a proportional guidance law, together with linear quadratic and model predictive controllers, are utilised to calculate the necessary control actions.

Keywords: aircraft, algorithm, bomb, control, dropping, guidance, Korean War, linearization, model, Octree, path planning, rectangloid, reference, subspace, system, trajectory, wind

Acknowledgements

Me gustaría agradecer su apoyo y esfuerzo a todas las personas que me han ayudado a lo largo de esta etapa, tanto a mi familia como amigos que, afortunadamente, siempre han estado ahí cuando se les ha necesitado y me han dado tan buenos momentos año tras año.

Tampoco puedo olvidarme de aquellos excelentes docentes con los que he tenido la oportunidad de aprender durante mi vida, no solamente en materia educativa, sino también emocional y ética, contribuyendo a formarme como persona en mi día a día. De este modo, sería difícil no acordarse de personas como José en estos momentos.

Contents

List of Figures

List of Tables

Nomenclature

Aerodynamic parameters

α	Angle of attack	(rad)
β	Sideslip angle	(rad)
γ	Flight path angle	(rad)
M	Mach number	(-)
M_{div}	Mach number of divergence	(-)
V	Total velocity modulus	(m/s)
V_a	Relative velocity modulus	(m/s)
$\overrightarrow{V}_B$	Total velocity vector in body axes	(m/s)
$\overrightarrow{V}_H$	Total velocity vector in local horizon axes	(m/s)
$\overrightarrow{V}_a$	Relative velocity vector in body axes	(m/s)
$\overrightarrow{V}_{wB}$	Total wind velocity vector in body axes	(m/s)
$\overrightarrow{V}_{wH}$	Total wind velocity vector in local horizon axes	(m/s)
X_{ACw}	Wing aerodynamic center	(m)
X_{ACt}	Aircraft tail aerodynamic center	(m)
X_{AC_b}	Bomb aerodynamic center	(m)
$X_{AC_{body}}$	Bomb body aerodynamic center	(m)
$X_{AC_{tail}}$	Bomb fins aerodynamic center	(m)
u	1^{st} Velocity component in x-body axis	(m/s)
v	2^{nd} Velocity component in y-body axis	(m/s)
w	3^{rd} Velocity component in z-body axis	(m/s)
u_w	1^{st} Wind velocity component in x-body axis	(m/s)
v_w	2^{nd} Wind velocity component in y-body axis	(m/s)
w_w	3^{rd} Wind velocity component in z-body axis	(m/s)
V_x	1^{st} Total velocity component in x-local horizon axis	(m/s)
V_y	2^{nd} Total velocity component in y-local horizon axis	(m/s)
V_z	3^{rd} Total velocity component in z-local horizon axis	(m/s)
V_{x_w}	1^{st} Wind velocity component in x-local horizon axis	(m/s)
V_{y_w}	2^{nd} Wind velocity component in y-local horizon axis	(m/s)
V_{z_w}	3^{rd} Wind velocity component in z-local horizon axis	(m/s)

Aerodynamic coefficients

CL	Total lift coefficient	(-)
CL_α	Lift coefficient derivative from angle of attack	(rad^{-1})
CL_{δ_E}	Lift coefficient derivative from elevator deflection	(rad^{-1})
CL_q	Lift coefficient derivative from the angular velocity q	(rad^{-1})
$CL_{\dot{\alpha}}$	Lift coefficient derivative from angle of attack derivative	(rad^{-1})
CY	Lateral force coefficient	(-)
CY_β	Lateral force coefficient derivative from sideslip angle	(rad^{-1})
CY_{δ_A}	Lateral force coefficient derivative from ailerons deflection	(rad^{-1})
CY_{δ_R}	Lateral force coefficient derivative from rudder deflection	(rad^{-1})
CY_p	Lateral force coefficient derivative from the derivative of angular velocity p	(rad^{-1})
CY_r	Lateral force coefficient derivative from the derivative of angular velocity r	(rad^{-1})
$C\mathcal{L}$	Total roll moment coefficient	(-)
$C\mathcal{L}_\beta$	Roll moment coefficient derivative from sideslip angle	(rad^{-1})
$C\mathcal{L}_{\delta_A}$	Roll moment coefficient derivative from ailerons deflection	(rad^{-1})
$C\mathcal{L}_{\delta_R}$	Roll moment coefficient derivative from rudder deflection	(rad^{-1})
$C\mathcal{L}_p$	Roll moment coefficient derivative from the angular velocity p	(rad^{-1})

$C\mathcal{L}_r$ Roll moment coefficient derivative from the angular velocity r (rad^{-1})

$C\mathcal{M}$ Total pitch moment coefficient (-)

$C\mathcal{M}_\alpha$ Pitch moment coefficient derivative from angle of attack (rad^{-1})

$C\mathcal{M}_{\delta_E}$ Pitch moment coefficient derivative from elevator deflection (rad^{-1})

$C\mathcal{M}_q$ Pitch moment coefficient derivative from the angular velocity q (rad^{-1})

$C\mathcal{M}_{\dot{\alpha}}$ Pitch moment coefficient derivative from angle of attack derivative (rad^{-1})

$C\mathcal{N}$ Total yaw moment coefficient (-)

$C\mathcal{N}_\beta$ Yaw moment coefficient derivative from sideslip angle (rad^{-1})

$C\mathcal{N}_{\delta_A}$ Yaw moment coefficient derivative from ailerons deflection (rad^{-1})

$C\mathcal{N}_{\delta_R}$ Yaw moment coefficient derivative from rudder deflection (rad^{-1})

$C\mathcal{N}_p$ Yaw moment coefficient derivative from the angular velocity p (rad^{-1})

$C\mathcal{N}_r$ Yaw moment coefficient derivative from the angular velocity r (rad^{-1})

CD_0 Parasitic drag coefficient (-)

CD Total drag coefficient (-)

CD_{wave} Wave drag coefficient of the aircraft(-)

CN Total lift coefficient of the bomb (-)

$CN_{\alpha,body}$ Lift coefficient derivative of the bomb body from angle of attack (rad^{-1})

$CN_{\alpha,tail}$ Lift coefficient derivative of the bomb fins from angle of attack (rad^{-1})

$CN_{q,body}$ Lift coefficient derivative of the bomb body from the angular velocity q (rad^{-1})

$CN_{q,tail}$ Lift coefficient derivative of the bomb fins from the angular velocity q (rad^{-1})

CY_b Total lateral force coefficient of the bomb (-)

$C\mathcal{M}_b$ Total pitch moment coefficient of the bomb (-)

$C\mathcal{M}_{\alpha,body}$ Pitch moment coefficient derivative of the bomb body from angle of attack (rad^{-1})

$C\mathcal{M}_{\alpha,tail}$ Pitch moment coefficient derivative of the bomb fins from angle of attack (rad^{-1})

$C\mathcal{M}_{q,body}$ Pitch moment coefficient derivative of the bomb body from the angular velocity q (rad^{-1})

$C\mathcal{M}_{q,tail}$ Pitch moment coefficient derivative of the bomb fins from the angular velocity q (rad^{-1})

$C\mathcal{N}_b$ Total yaw moment coefficient of the bomb (-)

$C\mathcal{N}_{\alpha,body}$ Yaw moment coefficient derivative of the bomb body from angle of attack (rad^{-1})

$C\mathcal{N}_{\alpha,tail}$ Yaw moment coefficient derivative of the bomb fins from angle of attack (rad^{-1})

$C\mathcal{N}_{q,body}$ Yaw moment coefficient derivative of the bomb body from the angular velocity q (rad^{-1})

$C\mathcal{N}_{q,tail}$ Yaw moment coefficient derivative of the bomb fins from the angular velocity q (rad^{-1})

CD_b Total drag coefficient of the bomb (-)

CD_{b_0} Parasitic drag coefficient of the bomb(-)

CD_{b_i} Induced drag coefficient of the bomb (-)

$(CD_0)_{body,friction}$ Parasitic drag coefficient of the bomb body due to friction (-)

$(CD_0)_{tail,friction}$ Parasitic drag coefficient of the bomb fins due to friction (-)

$(CD_0)_{base}$ Base drag coefficient of the bomb(-)

$(CD_0)_{body,wave}$ Wave drag coefficient of the bomb body (-)

$(CD_0)_{tail,wave}$ Wave drag coefficient of the bomb fins (-)

Physical parameters

g	Gravity constant	(m/s^2)
$\overrightarrow{g}_B$	Gravity vector in body axes	(m/s^2)
p_0	Pressure at sea level	(N/m^2)
ρ_0	Density at sea level	(kg/m^3)
T_0	Temperature at sea level	(K)
μ_0	Dynamic viscosity at sea level	(kg/ms)
γ_{ISA}	Adiabatic constant	$(-)$
R_{ISA}	Ideal gas constant	(J/kgK)
θ_{ISA}	Ratio between temperature $T(z)$ and sea level one	$(-)$
σ_{ISA}	Ratio between density $\rho(z)$ and sea level one	$(-)$
a_{ISA}	Sound speed	(m/s)
μ_{ISA}	Dynamic viscosity at a given altitude	(kg/ms)
m_a	Aircraft instantaneous mass	(kg)
m_{a_0}	Aircraft initial mass	(kg)
FM_0	Initial Fuel Mass	(kg)
m_b	Bomb mass	(kg)
m_{cyl}	Cylinder section bomb mass	(kg)
m_{cone}	Conical section bomb mass	(kg)

m_{tcone}	Truncated conical section bomb mass	(kg)
m_r	Rack bomb mass	(kg)
m_0	Initial mass of the complete system	(kg)
m	Instantaneous mass of the complete system	(kg)
W	Instantaneous weight of the complete system	(N)
$\dot{m}_f$	Fuel mass consumption	(kg/s)
x_{CG_a}	Mass center of the aircraft	(m)
x_{CG_f}	Mass center of the fuel	(m)
x_{CG_b}	Mass center of the bomb	(m)
$\overrightarrow{x}_{CG_{ba}}$	Position of the mass center of the bomb with respect to wing apex	(m)
$\overrightarrow{x}_{CG_r}$	Position of the mass center of the rack with respect to wing apex	(m)
x_{CG_0}	Initial mass center of the complete system	(m)
$x_{CG_{OEM}}$	Mass center of the operational empty mass	(m)
x_{CG}	Instantaneous mass center of the complete system	(m)
$x_{CG_{cyl}}$	Cylinder section bomb mass center position	(m)
$x_{CG_{cone}}$	Conical section bomb mass center position	(m)
$x_{CG_{tcone}}$	Truncated conical section bomb mass center position	(m)
T	Thrust	(kN)
$TSFC$	Thrust-Specific Fuel Consumption	$((kg/s)/N)$
T_{SL}	Maximum available thrust at sea level	(kN)
$TSFC_{SL}$	Thrust-Specific Fuel Consumption at sea level	$((kg/s)/N)$
I_{cyl}	Cylinder section bomb's inertia matrix	$(kg \cdot m^2)$
I_{cone}	Conical section bomb's inertia matrix	$(kg \cdot m^2)$
I_{tcone}	Truncated conical section bomb's inertia matrix	$(kg \cdot m^2)$
I_a	Aircraft's inertia matrix	$(kg \cdot m^2)$
I_b	Bomb's inertia matrix	$(kg \cdot m^2)$
I_0	Initial complete system's inertia matrix	$(kg \cdot m^2)$
I	Instantaneous complete system's inertia matrix	$(kg \cdot m^2)$
ϕ	Roll angle	(rad)
θ	Pitch angle	(rad)
ψ	Yaw angle	(rad)
p	1^{st} Angular velocity component in body axes	(rad/s)
q	2^{nd} Angular velocity component in body axes	(rad/s)
r	3^{rd} Angular velocity component in body axes	(rad/s)
$\mathcal{L}$	Roll moment	$(N \cdot m)$
$\mathcal{M}$	Pitch moment	$(N \cdot m)$
$\mathcal{N}$	Yaw moment	$(N \cdot m)$
L	Lift force	(N)
D	Drag force	(N)
Y	Lateral force	(N)
$\overrightarrow{\omega}$	Angular velocity vector in body axes	(rad/s)
$\overrightarrow{F}$	Force vector in body axes	(N)
$\overrightarrow{F}_A$	Aerodynamic force vector in body axes	(N)
$\overrightarrow{H}$	Angular moment in body axes	(N)
${}^BR_H$	Transformation matrix from local horizon axes to body axes	
${}^0R_H$	Transformation matrix from local horizon axes to fix axes	
M_ϕ	Rotation matrix with respect to x-axis	
M_θ	Rotation matrix with respect to y-axis	
M_ψ	Rotation matrix with respect to z-axis	
n	Load factor	$(-)$
n_{max}	Maximum allowed load factor	$(-)$
μ	Mean parameter in wind model	$(m/s, ^\circ)$
σ	Standard deviation parameter in wind model	$(m/s, ^\circ)$

Geometrical parameters

S_w	Wing surface	(m^2)
S_t	Horizontal stabilizer surface	(m^2)
S_v	Vertical stabilizer surface	(m^2)
b_w	Wing span	(m)
b_t	Tail span	(m)
c_w	Wing mean aerodynamic chord	(m)
c_{wr}	Wing root chord	(m)
c_{wt}	Wing tip chord	(m)
$\Lambda_{c/4w}$	Wing sweep angle at c/4	$(^\circ)$
$\Lambda_{c/4t}$	Horizontal stabilizer sweep angle at c/4	$(^\circ)$
$\Lambda_{c/4v}$	Vertical stabilizer sweep angle at c/4	$(^\circ)$
Γ_w	Wing dihedral angle	$(^\circ)$
Γ_t	Tail dihedral angle	$(^\circ)$
L_f	Fuselage length	(m)
h_a	Aircraft height	(m)
AR_w	Wing aspect ratio	(-)
e_w	Wing Oswald factor	(-)

S_b	Fins surface	(m^2)
c_b	Fins mean aerodynamic chord	(m)
b_b	Fins span	(m)
l_b	Bomb length	(m)
l_{b_1}	Bomb first section length (body)	(m)
l_n	Bomb nose length	(m)
l_{sl_1}	Length from nose top to first suspension lug	(m)
l_{sl_2}	Length from nose top to second suspension lug	(m)
d_b	Bomb body diameter	(m)
d_{b_b}	Bomb base diameter	(m)
Λ_{LE}	Fins sweep angle at leading edge	$(^\circ)$
AR_b	Fins aspect ratio	(-)
e_b	Fins Oswald factor	(-)
δ_P	Throttle lever position	(-)
δ_E	Elevator deflection	(rad)
δ_A	Ailerons deflection	(rad)
δ_R	Rudder deflection	(rad)

Control and guidance parameters

A	System matrix	$(-)$
B	Control matrix	$(-)$
A_d	Perturbations matrix	$(-)$
K	Gain matrix	$(-)$
Q	Weigh matrix for state variables	$(-)$
R	Weigh matrix for control variables	$(-)$
N_{xu}	Weigh matrix for combination of state and control variables	$(-)$
S	Riccati's solution	$(-)$
$\mathcal{J}$	Cost function	$(-)$
y	Output variables	
x	State variables	
u	Control variables	
Δx	Incremental state variables	
Δu	Incremental control variables	
d	Perturbation variables	
K_p	Proportional gain in PID	$(-)$
K_i	Integral gain in PID	$(-)$
K_d	Derivative gain in PID	$(-)$
$e(t)$	Error in the reference tracking	
λ_i	Eigenvalue i from the system matrix A	
T_s	Sample time	(s)
t_p	Prediction horizon time	(s)
t_c	Control horizon time	(s)
$\overrightarrow{a}$	Acceleration vector	(m/s^2)
$\overrightarrow{r}$	Relative position vector	(m)
r	Relative position vector modulus	(m)
$\overrightarrow{R}_1$	Target position vector	(m)
$\overrightarrow{R}_2$	Aircraft position vector	(m)
N	Constant of proportionality	$(-)$
η	Angle between $\overrightarrow{r}$ and $\overrightarrow{V}$	(rad)
Ψ	Desired course angle	(rad)
$\Delta\Psi$	Desired change in course angle	(rad)
R	Radius of gyration	(m)

Dynamic path planning parameters

q_i	Initial point of the trajectory	(m)
q_f	Final point of the trajectory	(m)
i	Decomposition level	
s_i	Subspace i	
R_{ij}	Partial cost function (i, j)	$(-)$
γ_0	Previous flight path angle	(rad)
ψ_0	Previous yaw angle	(rad)
$g(R_{ij})$	Gaussian rewarding function of R_{ij}	$(-)$
$G(i, j)$	Rewarding vector	$(-)$
ξ	Rewarding parameter from neighbour selection	$(-)$
$\mathcal{D}$	Final rewarding variable	$(-)$
$\mathcal{D}_1$	Rewarding variable from first path analysis	$(-)$
$\mathcal{D}_2$	Rewarding variable from second path analysis	$(-)$
η_1	Weigh parameters for $\mathcal{D}_1$	$(-)$
η_2	Weigh parameters for $\mathcal{D}_2$	$(-)$
O	Arc center	(m)
ds	Precision of trajectory	(m)
n_0	Minimum load factor	$(-)$
P_i	Waypoint i	(m)
V_{min}	Minimum velocity	(m/s)
V_{max}	Maximum velocity	(m/s)
V_0	Initial velocity	(m/s)
$\frac{\mathrm{d}V}{\mathrm{d}S}$	Derivative of velocity from curvilinear distance	$((m/s)/km)$
dt	Time step in dropping point estimation	(s)

Acronyms

ACD	Adaptive Cell Decomposition
$AFDS$	Autopilot Flight Director System
CDU	Control and Display Unit
CFD	Computational Fluid Dynamics
DNE	Down-North-East
DNW	Down-North-West
DSE	Down-South-East
DSW	Down-South-West
EMI	Electromagnetic Interference

FBW Fly By Wire
FCC Flight Control Computer
FCP Flight Control Panel
FM Fuel Mass
FMS Flight Management System
$FMSP$ Flight Mode Selector Panel
FSF Full State Feedback
$HIRF$ High Intensity Radiated Field
IAS Indicated Airspeed
ISA International Standard Atmosphere
KML Keyhole Markup Language
$KPAF$ Korean People's Air Force
LQR Linear Quadratic Regulator
LTI Linear Time Invariant
$MCDU$ Multi-function Control and Display
$MIMO$ Multiple-Input and Multiple-Output
MPC Model Predictive Control
NED North-East-Down
OEM Operational Empty Mass
$RR-ACD$ Recursive Rewarding Adaptive Cell Decomposition
$SISO$ Single-Input and Single-Output
TNT Trinitrotoluene
$TSFC$ Thrust-Specific Fuel Consumption
UN United Nations
UNE Up-North-East
UNW Up-North-West
US United States
$USAF$ United States Air Force
USE Up-South-East
USW Up-South-West

1 | Objectives and scope

1.1 Objectives

The objective of this project, consists in developing an autopilot application which enables to create a proposed trajectory, based on certain requirements such as altitude, velocity or obstacles, and specify the control actions that an aircraft North American F-86 Sabre should conduct so as to accomplish the designed path.

The environment selected for this purpose is the area of Korean peninsula, corresponding with the historical period of the Korean War (1950-1953); in particular, at the end of it, when the F-86 Sabre started to act as a fighter-bomber (see [1]). The mission lies in carrying two "750-lb Demolition Bombs M 117" from an initial point to a final one, in which the aircraft must throw them in order that the enemy objective on the ground is reached and eliminated.

The aircraft must be able to avoid restricted areas, according to radars, possible danger zones and the orography. In addition, the application has to find a possible route and design a trajectory which is easy for the aircraft to follow: quite demanding manoeuvres must be prevented, smooth transitions in direction changes should be imposed instead and always respecting the structural and control actions limits. Moreover, the Sabre has to fly stable as much as possible and present high robustness when wind perturbations happen, because in that way, the proposed trajectory will be achieved successfully. Therefore, the application must ensure the dynamic path building, guidance and control of the aircraft.

Finally, when the aircraft is close to the objective, the application must calculate the coordinates and time at which the Sabre should throw the bombs in order that they impact with the minimum possible error. This estimation must be conducted basing on the instantaneous flight parameters of the aircraft and the wind conditions in the area. Furthermore, another estimation must be made over the reference dynamic path so as to determine the deviation between the reference optimal dropping point and the real one, together with the error in the impact of both cases. In this way, the best control method to fulfil the mission will be selected based on the accuracy of the impact and the reference tracking.

1.2 Procedure

In order to accomplish the objectives previously mentioned, the project has been structured in the following steps:

1. First of all, a mathematical model of the F-86 Sabre and the demolition bombs has to be developed so as to analyse the motion of the complete system. In addition, an independent model of the bomb is needed to simulate the trajectory that they will follow when the racks of the aircraft release them.

2. Secondly, the model of the complete system is linearized with respect to specific values of mass of the aircraft, flight altitude and velocity at which the Sabre is trimmed in a vertical plane motion, as it will be explained in the Chapter 3. The linearization point is chosen taking into account the initial mass of the aircraft and the average altitude and speed awaited during the simulation.

3. After that, it is necessary to analyse different control methods that have been employed in similar previous projects, showing their advantages and disadvantages in the case of a possible application in this present project.

4. Once the best candidate models have been selected, the corresponding control laws are developed and applied on the linearized model. It is important to adjust the parameters of the different control laws by means of iterative simulations with the non-linear model until the stability and reference tracking of the system are acceptable. Afterwards, on the basis of the simulation results, the best control parameters are selected for the mission.

5. Then, it is needed to implement a guidance algorithm that enables the aircraft can follow any trajectory within its limitations. This algorithm must ensure that the Sabre does not deviate too much from the proposed trajectory; therefore, as it has been done before, different alternatives are studied in order to find the best option, which in combination with the respective control law, gives the less possible error in the path followed by the aircraft.

6. Afterwards, the dynamic path planning algorithm must be created. This algorithm will design the reference trajectory that the aircraft should conduct to reach the objective; also, the design has to avoid the Sabre from going through any obstacle and obtain a manageable route for the aircraft to follow. The trajectory obtained will consist in a series of segments joined with waypoints.

7. When the previous algorithm is developed, another one which optimises the trajectory should be implemented in order that the aircraft can follow the reference path properly and without high deviations. This is made by means of the introduction of curvatures where each waypoint is situated, creating smooth transitions between consecutive segments.

8. Then, a fourth algorithm must be designed, which must find the best possible dropping point in function of the instantaneous flight parameters.

9. And last but not least, a proper wind model is designed in order that the simulation can be performed in realistic environments and, consequently, the robustness of the control system is tested during the trajectory.

2 | Introduction and background

In this chapter, an introduction about different issues has been made. Firstly, a synopsis about the Korean War and its air battles between the MiG-15 and the F-86 Sabre is fundamental in order to understand the implication and relevance of the Sabre; secondly, an initial explanation of the bomb operation and the specific functionality of demolition bombs will be necessary to fully comprehend how this system works in the aircraft's mission, and finally, it has been conducted an introduction to control systems, starting with conventional ones, which is the one belonging to the Sabre, and finishing with Fly-By-Wire ones and the autopilot control structure, which will be implemented in this project.

2.1 Historical Background

2.1.1 The conflict

The Korean war began on June 25, 1950, when some 75,000 soldiers from the North Korean People's Army poured across the 38^{th} parallel, the boundary between the Soviet-backed Democratic People's Republic of Korea to the north and the pro-Western Republic of Korea to the south. This invasion was the first military action of the Cold War, which was caused by a series of geopolitical tensions between both states, due to the fact that the North half of Korea was a socialist state, under the communist leadership of Kim Il-sung, and the South one was a capitalist state in the south, under the anti-communist leadership of Syngman Rhee, and both governments claimed to be the sole legitimate government of all of Korea, and neither accepted the border as permanent. [5] [38]

By July, American troops had entered the war on South Korea's behalf, but American officials worked anxiously to fashion some sort of armistice with the North Koreans, since the alternative, which they feared, would be a wider war with Russia and China, or even, as some warned, the World War III. Finally, in July 1953, the Korean War came to an end. In all, some 5 million soldiers and civilians lost their lives in what many in the U.S. refer to as "the Forgotten War" for the lack of attention it received compared to more well-known conflicts like World War I and II or the Vietnam War.

2.1.2 The MiG-15 and F-86 Sabre

After successive battles, communist forces were pushed back to positions around the 38^{th} parallel, close to where the war started. Then, the front stabilised and the last two years of fighting became a war of attrition, the war in the air; however, was never a stalemate, considering that North Korea was subject to a massive bombing campaign by the United States (U.S.). Jet fighters confronted each other in air-to-air combat for the first time in history, and Soviet pilots covertly flew in defence of their communist allies.

In October 1950, the Chinese government bolstered the Korean People's Air Force (KPAF) with the MiG-15, one of the world's most advanced jet fighters. The heavily armed MiGs were faster than first-generation United Nations (UN) jets and therefore could reach and destroy US B-29 Superfortress bombers, despite their fighter escorts. With increasing B-29 losses, the U.S. Air Force (USAF) was forced to switch from a daylight bombing campaign to the safer but less accurate night-time bombing of targets.

The USAF countered the MiG-15 by sending over three squadrons of its most capable fighter, the F-86 Sabre, which arrived in December 1950. The MiG was designed as a bomber interceptor with a very high service ceiling (15,000 m) and heavily armed; on the other hand, the Sabre had lower ceiling (13,000 m) and lighter weaponry [1]. At high altitudes, manoeuvring and climbing the MiG had advantage, but the Sabre dived [2] faster, was more aerodynamically stable and had a radar gunsight that came in handy during high-speed jet dogfights [3]. The maximum speeds of both aircraft were fairly similar. [21] In the figure 2.1 (extracted from [24]), both can be observed.

Figure 2.1: Aircraft F-86F Sabre and MiG-15 in profile view.

Both sides appeared evenly matched in terms of pilot and aircraft quality,

[1] While MiG-15 carried one 37 mm cannon and two 23 mm cannons, the Sabre were armed with six 12.7 mm ones.

[2] Steep descending path.

[3] Aerial battles between fighter aircraft conducted at close range.

and the Americans had the disadvantage of political restrictions that prohibited hot pursuit of Communist MiGs to their bases across the Yalu in China. Fortunately, the Soviets replaced their aces with beginner pilots who soon demonstrated their inferior training and tactics versus their Western counterparts: they were supplemented by hordes of Chinese and North Korean pilots fresh from the farm plough. It was then the Sabre started racking up those big scores.

Perhaps the most incendiary statistic of the Korean War is the aircraft kill ratios. For years, a 10:1 kill ratio in favour of the Sabre was held to be true. That figure now appears extremely suspect, for instance, some historians calculate 224 Sabres lost, of which about a hundred were the result of aerial combat, and estimate that 566 MiG-15s were destroyed by Sabres, which would put the U.S. kill ratio at about 5.6 to 1. However, against those top Soviet WWII pilots, the ratio plunged to 1.4 to 1.

2.2 Contemporary bombs

2.2.1 Introduction to bombs

Historically, a bomb has usually consisted of a body, stabilizer, and means of detonation. The body can be filled with either an explosive, chemical, biological, nuclear, or inert filler. Stabilizers are attached to the bomb body and may consist of sheet-metal fins, parachutes, or cloth streamers. Moreover, the efficient destruction of the various types of enemy targets requires different types of bombs.[41]

A fundamental characteristic of the explosive charge of a bomb is its relative insensitivity to ordinary shock and heat incident to loading, transporting, handling, and storing. In this way, a fuze is used to provide a highly sensitive explosive element, which is called detonator, and the action necessary for detonating it. In addition, a link between the fuze detonator and the bomb's main charge, called booster, is necessary to ensure the full explosion. The fuze is fired by mechanical or electrical action.

The pattern of action that consists on firing the detonator and amplifying and relaying the explosion to the main charge by the booster is called "explosive train". This pattern is shown in the figure 2.2 (obtained from [41]).

In relation to the explosives used in bombs, the most used ones are: TNT (Trinitrotoluene), a relatively insensitive high-explosive of great stability; Amatol, a mixture of ammonium nitrate and TNT; Composition B, a very powerful explosive; Explosive D (Ammonium picrate), the least sensitive one; Tritonal, the standard explosive filler in general purpose bombs in the 50s, and HBX, a mixture of RDX[4], TNT, aluminum powder, arid desensitizer.

[4]Formally named as Ciclotrimetilentrinitramina.

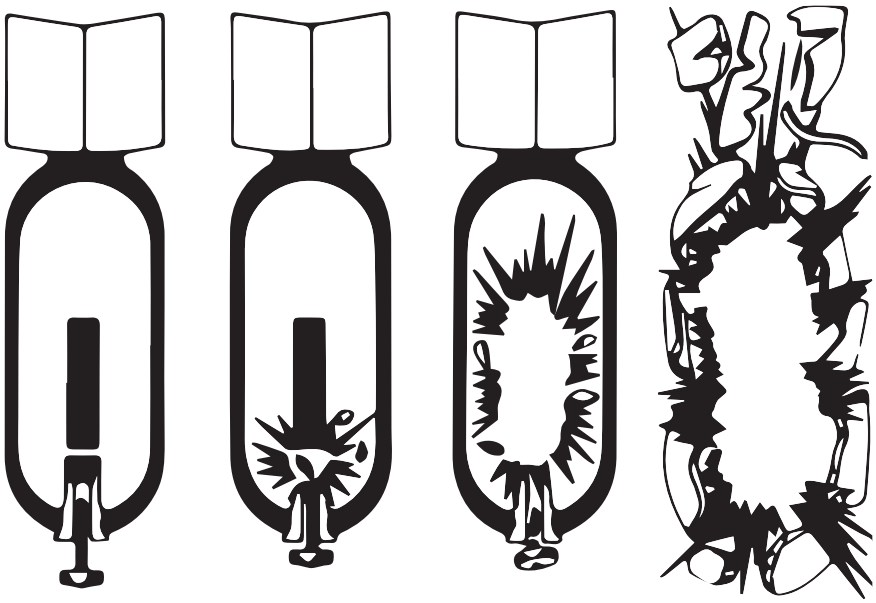

Figure 2.2: Explosive train. From left to right: armed fuze and bomb in flight; firing pin explodes primer detonator on impact; relay or booster amplifies the explosion, and shock explodes main charge in bomb.

2.2.2 Demolition bombs

Bombs are classified in many types, such as armor-piercing, fragmentation or general purpose, but in this case, the principal ones are the demolition ones, which were designed to carry maximum explosive charges: the percentage of explosive is 70 percent or more of the total bomb weight. These bombs should not be used where penetration is required and are best used where blast will provide maximum damage.

Demolition bombs have a short ogival nose and a cylindrical body that tapers to the base. To ensure functioning upon impact, both a nose and a tail fuze[5] are used. In the figure 2.3 (from [39]), the M117 bomb, which was used for the first time in the Korean War, can be observed.

Figure 2.3: M117 air-dropped demolition bomb.

[5]At first, the fuze was electrical, but was changed to mechanical due to adapter-boosters configuration.

The explosive filler used in demolition bombs is Tritonal. Demolition bombs were designed for a higher blast effect than general purpose bombs of comparable weights.

2.3 Control systems

2.3.1 Conventional

At first, mechanical flight control systems were used. In these systems, the control devices with the pilot are connected directly to the control surfaces of the aircraft by a system of rods, levers, cables and pulleys. The levels of stick and rudder-pedal were constrained by the physical capabilities of the pilot, which supposed a huge disadvantage in terms of manoeuvring. Two types of mechanical systems were employed: push-pull rods and cable-pulley. [9]

Due to the increase in size and flight envelope of aircraft[6], mechanical flight control systems were found inadequate. Finally, this fact led to the application of hydraulic power, which supplies high power and stiffness, being ideal as a medium for the operation of flight controls.

Hydro-mechanical controls have two parts: a mechanical circuit, similar as in mechanical flight control system, and a hydraulic circuit, in which servo valves are activated when the pilot moves the controls. This type of control is often used in older jets as well as high performance aeroplanes.

In the case of the F-86 Sabre, a fully powered hydraulic flight controlling was employed, which was highly sensitive to the inputs from the pilot, required far less physical and mental exertion during aerial manoeuvres than what, for example, did the MiG-15, which presented hydraulically boosted mechanical flight controls.[7]

2.3.2 Fly-By-Wire

In order to optimise the control system of an aircraft, by means of a lighter weight and an easier maintenance than mechanical systems, the Fly-By-Wire (FBW) replaced the conventional manual flight controls of an aircraft with an electronic interface.[9]

The movements of flight controls are converted to electronic signals and transmitted by wires. The flight control computer (FCC) determine how to move the actuators at each control surface to provide the ordered response.

In this way, the FBW system is used to control the Euler angles of an aircraft and its velocity, by means of the control column and the throttle lever, respectively. The information provided by motion and air data sensors gives the necessary tools to the FCC to calculate the proper control actions of the actuator, basing on the

[6]With the increasing speed of aircraft, it became more difficult to move the control surfaces due to high aerodynamic forces.

pilot's orders. [19] In the figure 2.4 (extracted from [25]), a scheme of a FBW system is shown.

The first aircraft with the FBW, came into service in the 70s, using analog implementation. Digital FBW systems have been in service since the late 80s. However, the susceptibility to Electromagnetic Interference (EMI) and High Intensity Radiated Field (HIRF), has entailed a considerable problem in FBW systems, and as a consequence, nowadays it is preferable to use optical fibre cables instead of electrical ones, but also to use optic sensors instead of electrical ones[7][9].

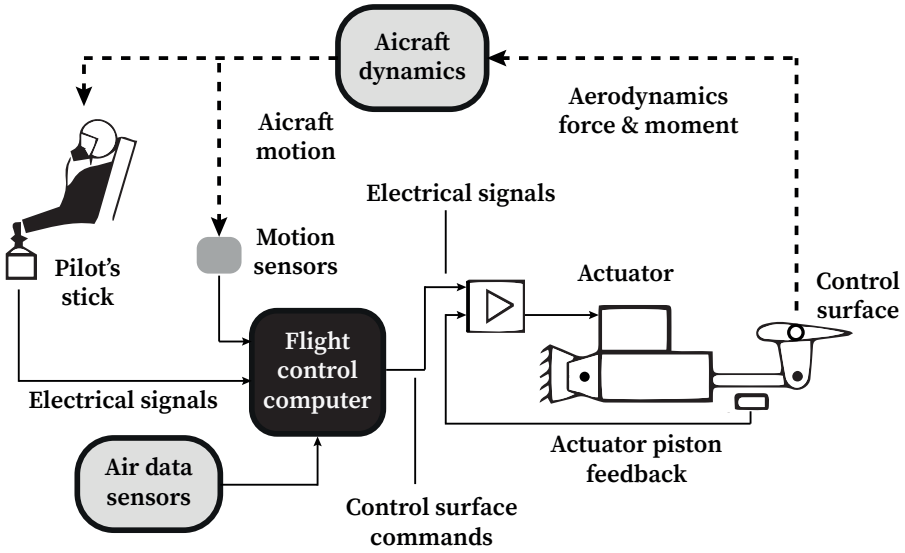

Figure 2.4: Flowchart of a FBW-FCC design.

Autopilot

In addition, the FBW system is not alone, there other elements in the complete control system that allows to follow a determined route with stability and even autonomously. In this way, the Autopilot Flight Director System (AFDS) plays a crucial role. [19]

The AFDS performs additional control loop closure to ensure that the aircraft conducts a determined trajectory. It controls the speed, altitude and heading angle at which the aircraft flies according to navigation functions associated with specific operations, making possible that, theoretically, no pilot would be needed to accomplish the mission of the aircraft.

The control and indication associated with these multiple autopilot modes, is provided by a Flight Mode Selector Panel (FMSP) or Flight Control Panel (FCP), which enables the selection of the principal operation modes and also provides information confirming that the modes are properly engaged and functioning correctly.

The operation modes can be: attitude modes, in which some of the Euler angles is kept as constant; data modes, parameters such as altitude, Mach number

[7]This control system is called Fly-By-Light.

or indicated airspeed (IAS) by means of the FCP, and acquisition modes, giving complete authority to the pilot to maintain a determined trajectory from a specific point.

Finally, the third loop closure is the Flight Management System (FMS), which performs the navigation or mission function, ensuring that the FBW and AFDS systems position the aircraft at the correct coordinates to coincide with the multiple waypoints belonging to the reference trajectory of the aircraft. The FMS is totally common in actual aircraft, in which preliminary mission planning is very strict.

In order to monitor the aircraft progress, a Multi-function Control and Display Unit (MCDU), also known as the Control and Display Unit (CDU), is employed, acting as an interface pilot-FMS.

The scheme of the whole control system can be observed in the figure 2.5 (extracted from [19]).

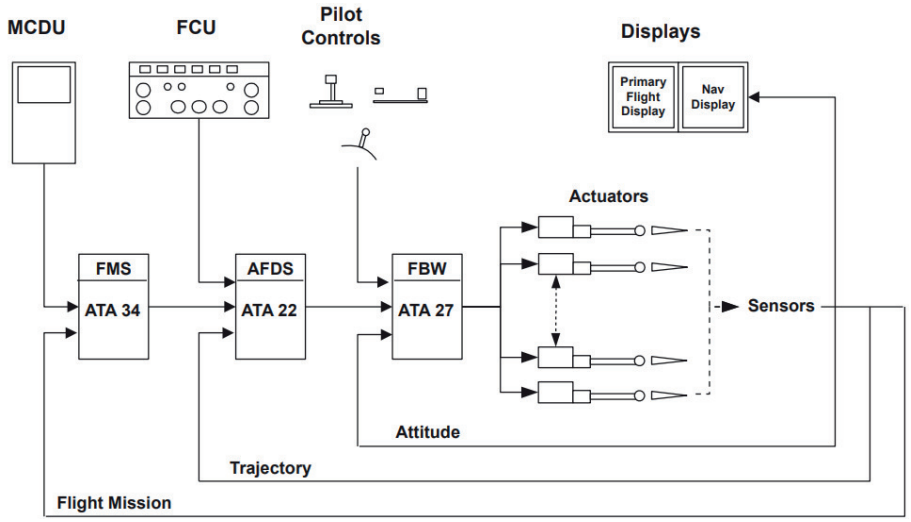

Figure 2.5: Interrelationship of flight functions.

3 | Model of the system

In this chapter, different dynamic models of the complete system and the 750-lb Demolition Bomb M117 are going to be developed by means of the appropriate literature. First of all, the dimensions and physical properties of the systems (the whole system and, independently, the M117 bomb), actuator characteristics and the limitations of the aircraft are compiled (extracted from [41, 35, 33, 34, 15]); afterwards, employing the fundamentals of flight mechanics (see [15, 13]), a theoretical and semi-empirical basis of aerodynamics (see [14, 15, 12, 6]) and the engine performance parameters (extracted from [26]), the non-linear dynamic models of the bomb and the whole system are created.

Once the non-linear model of the aircraft with the M117s is successfully developed, the equilibrium point of the system is calculated on the basis of given conditions of altitude, velocity, initial mass and heading angle, and then, the space-state model is linearized with respect to a symbolic equilibrium point. In addition, an example of the linearized state-space model is included.

3.1 Non-linear dynamic model of the complete system

Therefore, in this section the non-linear dynamic model of the complete system, consisting of the Sabre and the pair of M117s, is going to be developed. The theoretical basis that has been employed is the flight mechanics model based on [13], although varying some hypothesis as it is seen below.

3.1.1 Dimensions and physical parameters

Firstly, fundamental parameters of the Sabre and the bomb are recorded: dimensions, masses, centers of gravity and moments of inertia.

Aircraft in clean configuration

The F-86 Sabre aircraft has many different models, in which some parameters, mainly geometrical, change slightly. In this case, the F-86F-20 model is chosen for the whole analysis of this project.

The principal dimensions are recorded in the table 3.1. In addition, in the figure 3.1 the three views of the aircraft are shown. [35]

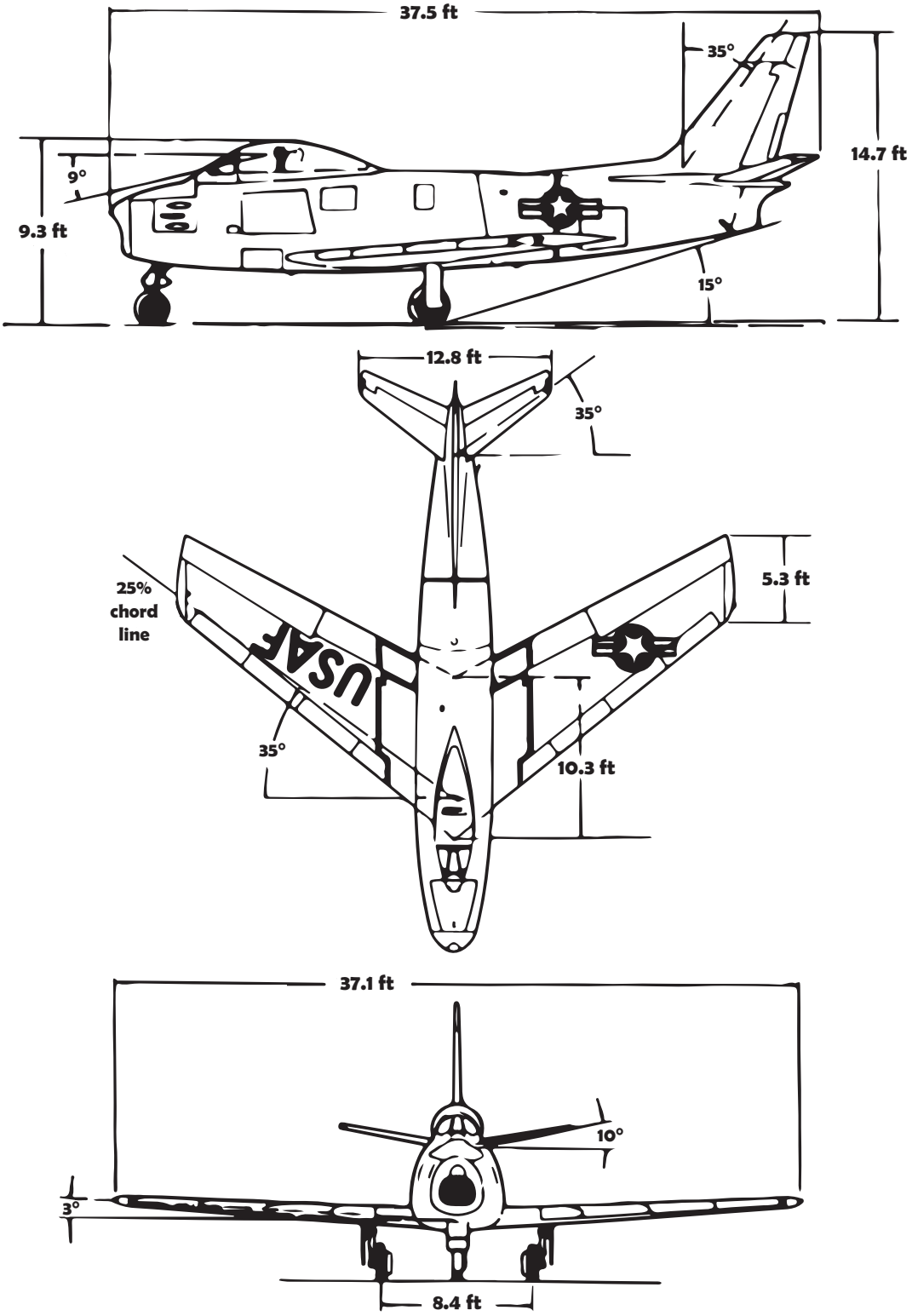

Figure 3.1: Airplane dimensions (ft) in three views.

Dimensions			
$S_w(m^2)$	26.88	$\Lambda_{c/4w}(^\circ)$	35.00
$S_t(m^2)$	3.27	$\Lambda_{c/4t}(^\circ)$	35.00
$S_v(m^2)$	3.03	$\Lambda_{c/4v}(^\circ)$	35.00
$b_w(m)$	11.31	$\Gamma_w(^\circ)$	3.00
$b_t(m)$	3.90	$\Gamma_t(^\circ)$	10.00
$c_w(m)$	2.46	$L_f(m)$	11.43
$c_{wr}(m)$	3.14	$h_a(m)$	4.48
$c_{wt}(m)$	1.62	$AR_w(-)$	4.76

Table 3.1: Principal dimensions of the F-86F-20 Sabre.

The initial mass of the aircraft will be the one corresponding to the combat one (according to [33]). Considering that the calculation of the mass center position is fairly difficult to accomplish, an acceptable value ($x_{CG_a} = 0.80 \cdot c_{wr}$) with respect to wing apex has been chosen for combat configuration (basing on [35]).

In addition, since the fuel mass is continuously diminishing during the flight, the mass center position of the aircraft changes, so that it is necessary to include these changes in aircraft's model. However, the fuel system is quite complex, thereby the whole fuel mass has been simplified as rectangular prism in order to make possible the calculation of fuel mass center position. This simplification can be observed easily in the figure 3.2 (obtained from [33]).

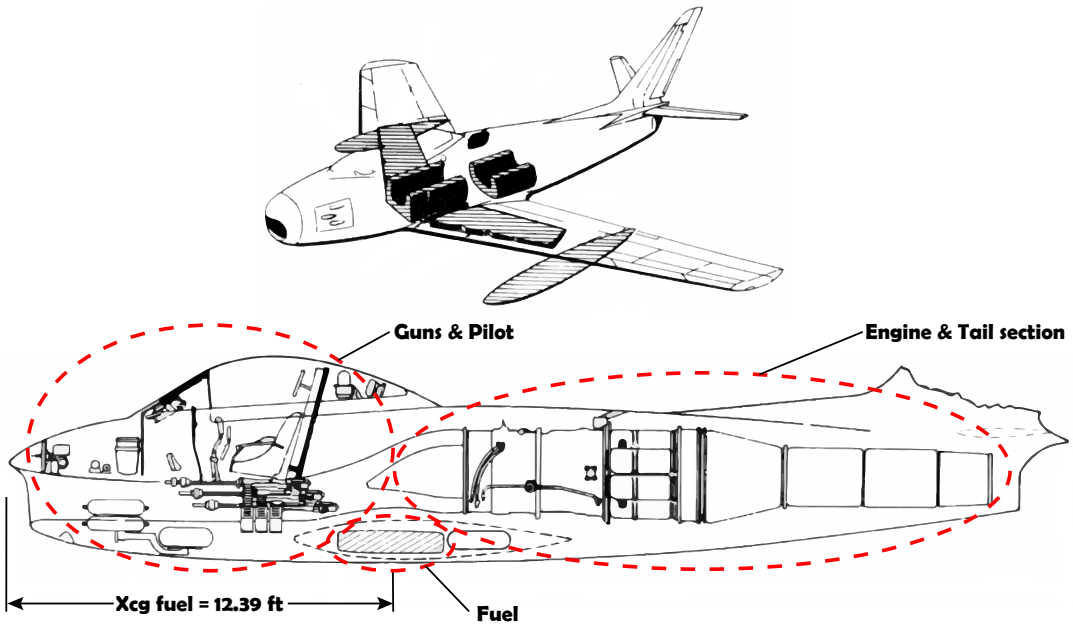

Figure 3.2: Profile and perspective views: fuel system and mass center position.

In table 3.2, initial mass and mass centers are shown, but also the fuel and operational empty masses.

Masses and x_{CG}				
$m_{a_0}(kg)$	$x_{CG_f}(m)$	$x_{CG_a}(m)$	$OEM(kg)$	$FM_0(kg)$
6739.03	0.48	2.51	4916.94	1822.08

Table 3.2: Initial masses and mass centers of the fuel and Sabre with respect to wing apex.

Respect to the matrix of moments of inertia, in the case of the terms in the diagonal, they can be easily computed by means of statistical data; nevertheless, the inertia products are quite complex to calculate (a division in CAD model of the mass aircraft in multiple sections should be conducted, as it is made in [22], or experimental procedures of the whole aeroplane to determine the real moments of inertia, as in [11]). The products I_{xy} and I_{yz} are negligible due to aircraft's symmetries, but I_{xz} not at all. However, an acceptable approximation can be made by assuming that the body axes are the principal axes of inertia (see [13]), so that the products of inertia are all null. Consequently, the inertia matrix of the aircraft is given by the expression 3.1 (from [15]). The non-dimensional radii of gyration values are shown in table A.1 (see Annex A).

$$\begin{aligned}[I_a] = \begin{pmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{pmatrix} \approx \begin{pmatrix} I_{xx} & 0 & I_{xz} \\ 0 & I_{yy} & 0 \\ I_{xz} & 0 & I_{zz} \end{pmatrix} \approx \\ \approx \begin{pmatrix} m_a(b_w/2)^2 R_x^2 & 0 & 0 \\ 0 & m_a(L_f/2)^2 R_y^2 & 0 \\ 0 & 0 & m_a((b_w + L_f)/2)^2 R_z^2 \end{pmatrix}\end{aligned} \tag{3.1}$$

M117 bomb

The principal measures of the bomb are given in the table 3.3. The figure 3.3 shows the dimensions of the table 3.3 in a two-views drawing (see [41]). The lengths l_{sl_1} and l_{sl_1} represent the position of the suspension lugs[1].

Dimensions			
$S_b(m^2)$	0.256	$l_{sl_1}(m)$	0.588
$b_b(m)$	0.569	$l_{sl_2}(m)$	0.943
$l_b(m)$	2.134	$d_b(m)$	0.409
$l_{b_1}(m)$	1.168	$d_{b_b}(m)$	0.102
$l_n(m)$	0.508	$AR_b(-)$	1.265
$c_b(m)$	0.490	$\Lambda_{LE}(^\circ)$	58.676

Table 3.3: Principal measures of the M117 bomb.

[1]These devices are hooks used to join the bomb with the aircraft.

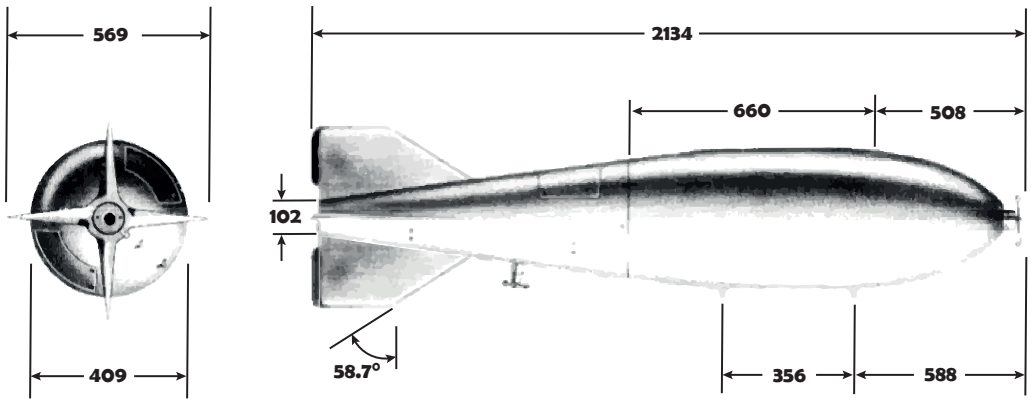

Figure 3.3: M117 bomb dimensions (mm).

The most of the bomb mass is concentrated in the bomb body (whose length is l_{b_1}) and the fins owns less than a 3%, as a consequence, an important simplification can be made in the calculation of moments of inertia: the bomb can be simplified as a cylinder and a cone, which form the body bomb, and a cone trunk[2]. Also, the mass center position will be simplified using the configuration mentioned. All values are calculated by means of the equations A.1 and A.2 (see Annex A), the final results are shown in the expression 3.2. The masses of the different sections of the bomb are shown in table 3.4[3].[6, 41]

Masses		
$m_b(kg)$		
362.44		
$m_{cyl}(kg)$	$m_{cone}(kg)$	$m_{tcone}(kg)$
266.44	76.04	19.96

Table 3.4: Masses of every part in which the bomb has been divided.

$$x_{CG_b} = 0.78m, \quad I_b = \begin{pmatrix} 6.94 & 0 & 0 \\ 0 & 40.70 & 0 \\ 0 & 0 & 40.70 \end{pmatrix} kg \cdot m^2 \tag{3.2}$$

The whole system

In order to hitch the M117s to the aircraft, it is necessary to use a device called as rack, which is shown in the figure 3.4.[35]

[2]The cone is the simplification of the nose and the cone trunk the one according to the fin assembly.

[3]Although the own name of the M117 has a "750 lb", the final mass with the fuze armed is 799 lb.

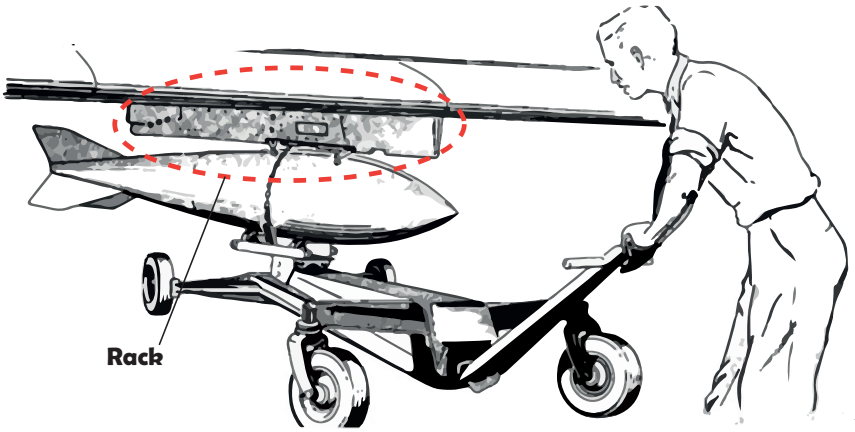

Figure 3.4: Bomb loading arrangement.

The situation of the bombs in the F-86F-20 Sabre will be defined by the coordinates of the suspension lugs of each bomb with respect to the mass center of the aircraft. These coordinates are specified by the rack collocation below the wing. The mass centers of the racks and bombs with respect to the wing apex are given by the table 3.5[4] (on the basis of [35]).

Masses		**Mass centers (x,y,z)**	
$m_b(kg)$	$m_r(kg)$	$\overrightarrow{x}_{CG_{ba}}(m)$	$\overrightarrow{x}_{CG_r}(m)$
362.44	57.15	(2.42,±2.00,0.67)	(2.40,±2.00,0.26)

Table 3.5: Masses and mass centers of the racks and bombs with respect to the wing apex.

The obtaining of the final mass, mass center and inertia matrix of the complete system are accomplished in equations A.3 (see Annex A). Final results are shown in the expression 3.3. The rack has been supposed as a point-mass.

$$\begin{cases} m_0 = 7578.21kg \\ x_{CG_0} = 2.50m \end{cases} \quad , \quad I_0 = \begin{pmatrix} 26050.35 & 0 & 0 \\ 0 & 26436.79 & 0 \\ 0 & 0 & 187654.60 \end{pmatrix} kg \cdot m^2 \qquad (3.3)$$

However, these results are only valid for the initial conditions, thus it is quite relevant to include the changes with time in the previous parameters in order to take into account the fuel mass consumption. The functions used will be the ones corresponding to the equations 3.4[5].

[4]In this case, the coordinates present the x component as positive, because the reference axes to calculate the parameters is located on the wing apex and x-axis is pointing to aircraft's aft.

[5]There is a change in inertia matrix due to variation of mass center, however this is so small in comparison with matrix values that it can be considered as negligible.

$$x_{CG_{OEM}} OEM = m_0 x_{CG_0} - x_{CG_f} FM_0 \Rightarrow x_{CG}(t) = \frac{m_0 x_{CG_0} + x_{CG_f}(FM(t) - FM_0)}{m(t)} \tag{3.4a}$$

$$I(t) = \begin{pmatrix} 3.316m(t) + 3703.73 & 0 & 0 \\ 0 & 3.910m(t) + 87.18 & 0 \\ 0 & 0 & 27.833m(t) + 87.18 \end{pmatrix} kg \cdot m^2 \tag{3.4b}$$

3.1.2 Reference frames

In this model, three different reference systems are employed. In the first place, a fixed reference frame located on the ground, but incorporating some variations with respect to the typical model of North-East-Down (NED) as it is explained below. The second one, a moving reference frame called as local horizon whose origin is the mass center of the aircraft and its axes are parallel to the ones corresponding to the fixed reference frame. And finally, the body-axes, which are also moving axes but in this case, they are fixed to the aeroplane[6] and their orientation is defined by the Euler angles (ϕ, θ, ψ).[13]

The rotation matrices employed to establish the the relation between local horizon and body axes reference frames are given by the equation 3.5. In addition, it is important to mention that rotation matrices are orthogonal, so that the inverse of a rotation matrix is equal to its transpose.[4]

$$M_\phi = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{pmatrix}, \; M_\theta = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix}, \; M_\psi = \begin{pmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.5}$$

These rotation matrices are combined in a Roll-Pitch-Yaw sequence in order to build the final rotation matrix from body axes to local horizon axes. However, the inverse of this matrix, ${}^B R_H$, will be most used than ${}^H R_B$ in the general equations, which is shown in the equation 3.6.[4]

$$\begin{aligned} {}^H R_B = M_\psi M_\theta M_\phi \Rightarrow {}^B R_H = {}^H R_B^{-1} = M_\phi^{-1} M_\theta^{-1} M_\psi^{-1} = \\ = \begin{pmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \cos\psi\sin\theta\sin\phi - \cos\phi\sin\psi & \cos\phi\cos\psi + \sin\theta\sin\phi\sin\psi & \cos\theta\sin\phi \\ \cos\phi\cos\psi\sin\theta + \sin\phi\sin\psi & -\cos\psi\sin\phi + \cos\phi\sin\theta\sin\psi & \cos\theta\cos\phi \end{pmatrix} \end{aligned} \tag{3.6}$$

The fixed axes use the NED model, in which the x-axis points to the north, y-axis to the east and z-axis to the earth's center, but applying a rotation matrix

[6] x-axis points to the nose of the aircraft, y-axis to right semi-wing and z-axis downwards.

given by the expression 3.7. In this way, the z-axis of this fixed reference frame is pointing upwards. This change is conducted due to the fact that the coordinate z intervenes in the values of the atmospheric parameters defined by the International Standard Atmosphere (ISA), thus it is necessary to obtain the altitude as positive values and not negative.

$$ {}^{0}R_{H} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \Rightarrow \begin{pmatrix} x \\ y \\ z \end{pmatrix}_{0} = {}^{0}R_{H}\,{}^{H}R_{B} \begin{pmatrix} x \\ y \\ z \end{pmatrix}_{B} \tag{3.7} $$

In the figure 3.5, a representation of the three reference frames used in the model can be observed.

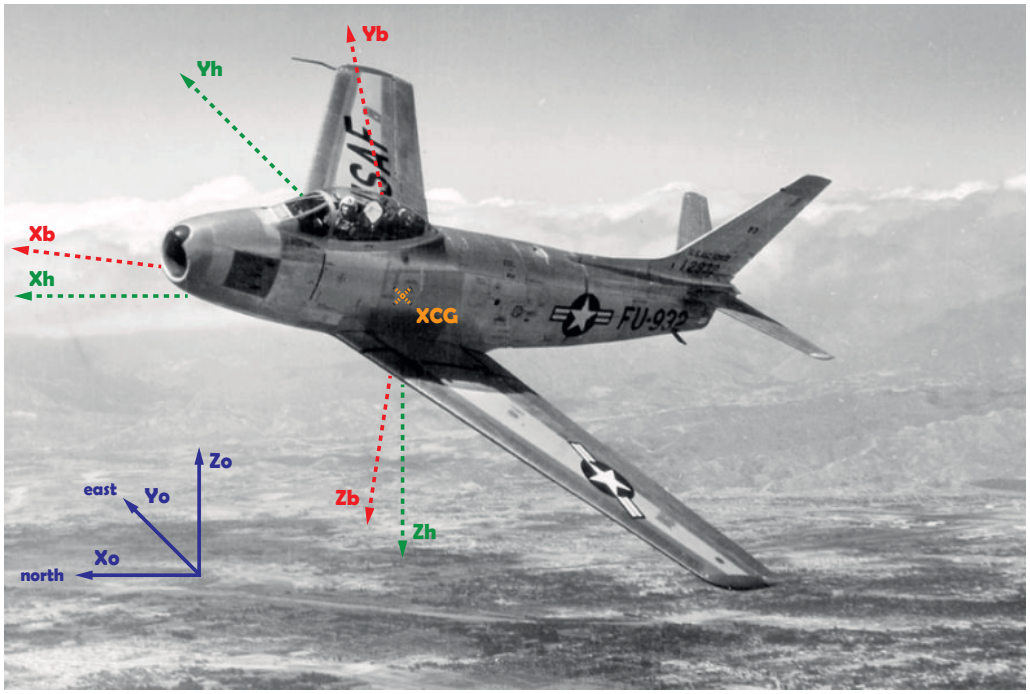

Figure 3.5: Reference frames: fixed axes in blue, local horizon axes in green and body axes in red.

3.1.3 Actuators and limitations

In normal conditions, the F-86 Sabre utilises till 8 different elements of control in a complete flight: the throttle lever to have the power over the engine and 7 control surfaces to adapt the aircraft motion to flight requirements; however, in this project, only 4 actuators will be considered.[35]

Regarding the control surfaces, the Sabre offers the possibility of utilising flaps and slats in taking-off and landing situations, and spoilers[7] in those circumstances too, but also during the flight to accomplish demanding manoeuvres. In addition,

[7]In the Sabre, these devices are quite unusual, because they are located on the fuselage sides instead of the wing.

during the whole flight, the Sabre can use the elevators, horizontal stabilizer, rudder and ailerons.

In this case, obviously flaps and slats are not deflected. Furthermore, spoilers are deactivated, because the motion is restricted when the aircraft carries bombs and therefore, manoeuvres are not allowed. And finally, the horizontal stabilizer is assumed to be in its original position, since the flight of the mission is going to be achieved at low altitudes, so that low elevator deflections produce high pitch moments on the aircraft and there is no necessity in using the horizontal stabilizer as a control surface in order to fulfil the motion requirements.

In summary, the actuators that are going to be utilised are:

1. Throttle lever (activation level represented by δ_P): modification of the thrust provided by the engine.
2. Elevators (deflection represented by δ_E): used to vary the pitch angle (θ).
3. Ailerons (deflection represented by δ_A): used to vary the roll angle (ϕ).
4. Rudder (deflection represented by δ_R): used to vary the yaw angle (ψ).

In addition, the actuators have limitations in their values and time derivatives, and consequently they must have taken into account in the control design. The specification values of each parameter are provided by 3.6 (see [35]).

Specifications			
Parameter	**Maximum value**	**Minimum value**	**Time variation limit**
Engine δ_P	1.00	0.10	0.20/s
Elevators δ_E	2.83 °	-21.00 °	30.00 °/s
Ailerons δ_A	15.00 °	-15.00 °	60.00 °/s
Rudder δ_R	25.50 °	-25.50 °	30.00 °/s

Table 3.6: Restrictions in the dynamics of throttle lever and control surfaces.

In the figure 3.6, an example of actuator dynamics (throttle lever) is shown.

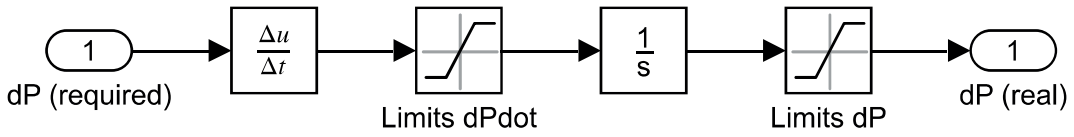

Figure 3.6: Scheme of the dynamic restrictions in throttle lever actuator.

Moreover, two additional limitations must be imposed during the flight. Seeing that the inboard stations of the wing (where the M117s are located) are being utilised and according to the flight manual of the F-86F Sabre (see [34]), no rolls are allowed and the load factor is limited to a specific interval. Those restrictions are given by the expressions 3.8.

$$n = \frac{L}{W}, \quad -2 \leq n \leq 5.5 \tag{3.8a}$$

$$-90^\circ \leq \phi \leq 90^\circ \tag{3.8b}$$

3.1.4 General equations

In order to develop the set of equations from which the non-linear model will be built, it is fundamental to previously determine a series of hypothesis that define the system. These hypothesis are (see [13]):

1. The motion is studied by means of the consideration of the aircraft as a rigid body, so that it means that the distance between two any points of the aircraft keeps constant. This surmise is not strictly valid, because, for example, aeroelastic effects or changes in mass center due to deflections of control surfaces are not taken into account; however this assumption is fairly close to the reality and it is enough to describe the aircraft motion as a translation and rotation with respect to the mass center of the system.

2. The mass center of the system can vary according to fuel mass consumption, but its time derivative is negligible, seeing that this change is extremely slow. In addition, in respect of the gravity force, it does not generate moment due to the mass center definition.

3. The total mass and inertia matrix of the system are variable due to fuel mass consumption, as it has been seen in the Section 3.1.

4. When the set of equations are described using the mass center of the system as the coordinates origin, rotation motion equations are independent from translation ones, but not in the opposite sense, as it is seen next.

5. Due to the fact that the aeroplane does not achieve hypersonic velocities or high supersonic ones, the Coriolis acceleration[8] is negligible and its influence in the translation motion can be disregarded.

6. The atmosphere motion is not null, thus the wind velocity will be considered in the equations. The wind velocity and direction models are shown in this chapter.

7. As it has been seen before, some assumptions have been made in relation to the inertia matrix of the aircraft. I_{xy} and I_{yz} are practically null, owing to aircraft symmetry in planes XY and YZ. In the case of I_{xz}, it will be also null because the body axes are supposed as the principal axes of inertia.

8. Rotational inertia effects from engine axis are not taken into account. In addition, any miss-alignment of the thrust vector from the axis is supposed as null.

Once the hypothesis have been detailed, two different velocities related to the system must be determined: system velocity and relative one. The first one is

[8] Only when the flight velocity with respect to the earth is higher than 900 m/s, this term starts to become relevant.[13]

the velocity of the body and the second one is the indicated airspeed (IAS), both calculated by means of the equations 3.9.

$$\overrightarrow{V}_H = \begin{pmatrix} V_x \\ V_y \\ V_z \end{pmatrix} \Rightarrow \overrightarrow{V}_B = {}^B R_H \overrightarrow{V}_H = \begin{pmatrix} u \\ v \\ w \end{pmatrix} \tag{3.9a}$$

$$\overrightarrow{V}_{w_H} = \begin{pmatrix} V_{x_w} \\ V_{y_w} \\ V_{z_w} \end{pmatrix} \Rightarrow \overrightarrow{V}_{w_B} = {}^B R_H \overrightarrow{V}_{w_H} = \begin{pmatrix} u_w \\ v_w \\ w_w \end{pmatrix} \Rightarrow \overrightarrow{V}_a = \begin{pmatrix} u - u_w \\ v - v_w \\ w - w_w \end{pmatrix} \tag{3.9b}$$

In addition, the gravity vector in body axes is calculated in the equation 3.10.

$$\overrightarrow{g}_B = {}^B R_H \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} = \begin{pmatrix} -g \sin\theta \\ g \cos\theta \sin\phi \\ g \cos\theta \cos\phi \end{pmatrix} \tag{3.10}$$

In translation and rotation equations, the mass variation is discounted because of the fact that it is negligible in comparison with the rest of the terms in those equations. In this way, the mass variation is considered by means of the actualization of the total mass value of the system at each instant of time. The translation equations are given in the expressions 3.11.

$$\overrightarrow{\omega} = \begin{pmatrix} p \\ q \\ r \end{pmatrix}, \quad \dot{\overrightarrow{V}}_{ABS} = \dot{\overrightarrow{V}}_B + \overrightarrow{\omega} \times \overrightarrow{V}_B = \begin{pmatrix} \dot{u} + qw - rv \\ \dot{v} + ru - pw \\ \dot{w} + pv - qu \end{pmatrix} \tag{3.11a}$$

$$\overrightarrow{F} = \frac{\mathrm{d}(m\overrightarrow{V}_{ABS})}{\mathrm{d}t} \approx m\dot{\overrightarrow{V}}_{ABS} \Rightarrow \overrightarrow{F} = \overrightarrow{T} + \overrightarrow{F}_A + \overrightarrow{g}_B \tag{3.11b}$$

$$\begin{cases} m(\dot{u} + qw - rv) = T + \frac{1}{2}\rho S_w V_a^2 C_X - mg\sin\theta \\ m(\dot{v} + ru - pw) = \frac{1}{2}\rho S_w V_a^2 C_Y + mg\cos\theta\sin\phi \\ m(\dot{w} + pv - qu) = \frac{1}{2}\rho S_w V_a^2 C_Z + mg\cos\theta\cos\phi \end{cases} \tag{3.11c}$$

In order to determine the rotation equations, the expression of the angular moment of the system must be obtained (see equations 3.12). Since the products of inertia are supposed to be null or negligible, these equations are noticeably simplified.

$$\overrightarrow{H} = \iiint \rho(x,y,z) \cdot \overrightarrow{r} \times (\overrightarrow{w} \times \overrightarrow{r}) dV = \begin{pmatrix} I_{xx}p - I_{xy}q - I_{xz}r \\ I_{yy}q - I_{yx}p - I_{yz}r \\ I_{zz}r - I_{zx}p - I_{zy}q \end{pmatrix} \approx \begin{pmatrix} I_{xx}p \\ I_{yy}q \\ I_{zz}r \end{pmatrix} \tag{3.12a}$$

$$\dot{\overrightarrow{H}}_{ABS} = \dot{\overrightarrow{H}} + \overrightarrow{\omega} \times \overrightarrow{H} =\approx \begin{pmatrix} I_{xx}\dot{p} + (I_{zz} - I_{yy})qr \\ I_{yy}\dot{q} + (I_{xx} - I_{zz})pr \\ I_{zz}\dot{r} + (I_{yy} - I_{xx})pq \end{pmatrix} \tag{3.12b}$$

$$\overrightarrow{M} = \begin{pmatrix} \mathcal{L} \\ \mathcal{M} \\ \mathcal{N} \end{pmatrix} = \dot{\overrightarrow{H}}_{ABS} \Rightarrow \begin{cases} \frac{1}{2}\rho S_w b_w V_a^2 C_{\mathcal{L}} = I_{xx}\dot{p} + (I_{zz} - I_{yy})qr \\ \frac{1}{2}\rho S_w c_w V_a^2 C_{\mathcal{M}} = I_{yy}\dot{q} + (I_{xx} - I_{zz})pr \\ \frac{1}{2}\rho S_w b_w V_a^2 C_{\mathcal{N}} = I_{zz}\dot{r} + (I_{yy} - I_{xx})pq \end{cases} \tag{3.12c}$$

Three more equations, which are given by the expressions in 3.13, are needed to evaluate the relation between Euler angles derivatives and angular velocities of the system.

$$\begin{pmatrix} p \\ q \\ r \end{pmatrix} = \begin{pmatrix} \dot{\phi} \\ 0 \\ 0 \end{pmatrix} + M_{(-\phi)} \begin{pmatrix} 0 \\ \dot{\theta} \\ 0 \end{pmatrix} + M_{(-\phi)}M_{(-\theta)} \begin{pmatrix} 0 \\ 0 \\ \dot{\psi} \end{pmatrix} \Rightarrow \begin{cases} p = \dot{\phi} - \dot{\psi}\sin\theta \\ q = \dot{\theta}\cos\phi + \dot{\psi}\cos\theta\sin\phi \\ r = \dot{\psi}\cos\theta\cos\phi - \dot{\theta}\sin\phi \end{cases} \tag{3.13}$$

Finally, the cinematic equations 3.14, provide the velocities of the system in the fixed reference frame.

$$\begin{pmatrix} V_x \\ V_y \\ V_z \end{pmatrix} = {}^0R_H \, {}^H R_B \begin{pmatrix} u \\ v \\ w \end{pmatrix} \tag{3.14a}$$

$$\begin{cases} \frac{\mathrm{d}x}{\mathrm{d}t} = u\cos\psi\cos\theta + v(\cos\psi\sin\theta\sin\phi - \cos\phi\sin\psi) + w(\sin\theta\cos\phi\cos\psi + \sin\phi\sin\psi) \\ \frac{\mathrm{d}y}{\mathrm{d}t} = u\cos\theta\sin\psi + v(\cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi) + w(-\cos\psi\sin\phi + \cos\phi\sin\theta\sin\psi) \\ \frac{\mathrm{d}z}{\mathrm{d}t} = u\sin\theta - v\cos\theta\sin\phi - w\cos\theta\cos\phi \end{cases} \tag{3.14b}$$

3.1.5 ISA model and engine performance

In order to have an estimation of the fundamental atmospheric parameters in function of the flight altitude, the ISA model is used (see [15]). This model is defined by the equations 3.15, which are only valid for the Troposphere[9], since the mission is accomplished at low altitudes.

[9]This is the first layer of the Atmosphere, in which the altitude is less than 11000 meters.

$$\begin{cases} g = 9.81 m/s^2 \\ p_0 = 101325.00 N/m^2 \\ \rho_0 = 1.225 kg/m^3 \\ T_0 = 288.15 K \\ \mu_0 = 1.789 \times 10^{-5} kg/ms \\ \gamma_{ISA} = 1.4 \\ R_{ISA} = 286.9 J/kgK \end{cases} \quad \begin{cases} \theta_{ISA}(z) = \frac{T(z)}{T_0} = 1 - 22.57 \cdot 10^{-6} z \\ \sigma_{ISA}(z) = \frac{\rho(z)}{\rho_0} = (1 - 22.57 \cdot 10^{-6} z)^{4.256} \\ a_{ISA}(z) = \sqrt{\gamma R T(z)} \\ \mu_{ISA}(z) = \frac{1.458 \cdot 10^{-6} T(z)^{1.5}}{110.4 + T(z)} \, [kg/ms] \end{cases} \tag{3.15}$$

In relation to the engine parameters, utilising the D. Hull model and the specifications of the engine J47-GE-27 (see [15] and [26], respectively), functions of thrust and Thrust-Specific Fuel Consumption (TSFC) of the aircraft are created. In the model, which is shown in equations 3.16, both the thrust and TSFC are dependent on the altitude and performance parameters at sea level, but the thrust will be function of the throttle lever activation.

$$\left\{ \begin{array}{c} T_{SL} = 26.289 kN \\ TSFC_{SL} = 2.56 \times 10^{-5} (kg/s)/N \end{array} \right\} \Rightarrow \begin{cases} T(z, \delta_P) = T_{SL} \left[\sigma(z)\right]^{1.2} \delta_P \\ TSFC(z) = TSFC_{SL} \left[\sigma(z)\right]^{0.1} \end{cases} \tag{3.16}$$

Once the thrust and TSFC models have been obtained, it is possible to compute the mass fuel consumption through the expression 3.17 and then, the instantaneous value of the system mass.

$$\dot{m}_f = TSFC(z) T(z, \delta_P) \Rightarrow m(t) = m_0 - \dot{m}_f t \tag{3.17}$$

3.1.6 Aerodynamic coefficients

The calculation of aerodynamic coefficients is quite complex and requires the use of Computational Fluid Dynamics (CFD) and different methods to evaluate their values at different conditions of velocity, altitude or aerodynamic angles, among others. In this case, longitudinal dynamics coefficients[10] are calculated from a theoretical and semi-empirical basis (see [15, 14]); however, lateral-directional coefficients[11] are computed from estimation and semi-empirical methods mainly (see [12]).

First of all, it is fundamental to define the aerodynamic angles (attack and sideslip) and their derivatives with respect to time. These are shown in equations 3.18.[13]

[10] Coefficients that affect to the motion in the vertical plane, such as CL or CD.

[11] Those which affect the 3-D motion, such as CY or $C\mathcal{L}$

$$\begin{cases} V_a = \sqrt{(u-u_w)^2+(v-v_w)^2+(w-w_w)^2} \\ \alpha = \arctan(\frac{w-w_w}{u-u_w}) \\ \beta = \arcsin(\frac{v-v_w}{V_a}) \end{cases} \tag{3.18a}$$

$$\begin{cases} \frac{\mathrm{d}V_a}{\mathrm{d}t} = \frac{(u-u_w)(\dot{u}-\dot{u}_w)+(v-v_w)(\dot{v}-\dot{v}_w)+(w-w_w)(\dot{w}-\dot{w}_w)}{V_a} \\ \frac{\mathrm{d}\alpha}{\mathrm{d}t} = \frac{(u-u_w)(\dot{w}-\dot{w}_w)-(w-w_w)(\dot{u}-\dot{u}_w)}{(u-u_w)^2+(w-w_w)^2} \\ \frac{\mathrm{d}\beta}{\mathrm{d}t} = \frac{(\dot{v}-\dot{v}_w)V_a-(v-v_w)\dot{V}_a}{V_a^2\sqrt{1-\frac{(v-v_w)^2}{V_a^2}}} \end{cases} \tag{3.18b}$$

$$\begin{pmatrix} \frac{\mathrm{d}\dot{u}_w}{\mathrm{d}t} \\ \frac{\mathrm{d}\dot{v}_w}{\mathrm{d}t} \\ \frac{\mathrm{d}\dot{w}_w}{\mathrm{d}t} \end{pmatrix} = {}^B\dot{R}_H \begin{pmatrix} V_{x_w} \\ V_{y_w} \\ V_{z_w} \end{pmatrix} + {}^BR_H \begin{pmatrix} \dot{V}_{x_w} \\ \dot{V}_{y_w} \\ \dot{V}_{z_w} \end{pmatrix} \tag{3.18c}$$

Afterwards, lift and drag coefficients are calculated. Lift coefficient is function of angle of attack and elevators deflection, but also the angular velocity q and time derivative of the angle of attack; finally, CL is dependent on body velocities (u,v,w), altitude, δ_E and aircraft geometry. The positions of the aerodynamic centers of the wing and horizontal stabilizer are assumed to be a the first quarter of each Mean Aerodynamic Chord (MAC), although in transonic regime this position is slightly modified and tends to be at the half of the MAC[12].

In relation to the drag coefficient, there is a dependence on lift coefficient due to the induced drag, so that CD will be also function of those parameters. In addition, a drag polar of constant coefficients has been chosen for the linearization process[13], although the real value is computed in the non-linear model. Regarding the M117s, their contribution to aerodynamic coefficients of the complete system is only noticeable in drag values, contributing to the total drag coefficient in the same way as nacelle engines or fuel tanks.

Total expressions of lift and drag coefficients are given by the equations 3.19.

$$CL = CL_\alpha \alpha + CL_{\delta_E}\delta_E + CL_q \frac{c_w}{2V_a} q + CL_{\dot{\alpha}} \frac{c_w}{2V_a}\dot{\alpha} \tag{3.19a}$$

$$CD = CD_0 + K \cdot CL^2 + CD_{wave} = CD_0 + \frac{CL^2}{\pi A R_w e_w} + CD_{wave} \tag{3.19b}$$

$$CD_{wave} = \begin{cases} 0 & M < M_{div} \\ 29.2(M - M_{div}) & M \geq M_{div} \end{cases}, \quad M_{div} = 0.85 \tag{3.19c}$$

The formula of the pitch moment coefficient, given by the expression 3.20, is fairly similar to the lift coefficient one, but including negative signs in its derivatives.

[12] In supersonic regime, the aerodynamic center position is located very close to MAC/2.

[13] The Reynolds Cut-off incorporates If conditionals in the final result of CD_0, therefore a mean value has been chosen in the linearization to make possible an estimation.

Neither the lift coefficient nor the pitch moment one present an initial component (CL_0 or $C\mathcal{M}_0$), this is due to the fact that the airfoil section is symmetrical.

$$C\mathcal{M} = C\mathcal{M}_\alpha \alpha + C\mathcal{M}_{\delta_E}\delta_E + C\mathcal{M}_q \frac{c_w}{2V_a} q + C\mathcal{M}_{\dot{\alpha}} \frac{c_w}{2V_a}\dot{\alpha} \tag{3.20}$$

Finally, each one of the lateral-directional coefficients follows the same structure. All of them are function of the same variables: $CY, C\mathcal{L}, C\mathcal{N} = f(\beta, \dot{\beta}, \delta_A, \delta_R, p, r)$. Their expressions are given by the equations 3.21. However, the coefficient derivatives are not all dependent on the same parameters: some of them are function of variables such as Mach, altitude or angle of attack. In table 3.7, these dependencies are shown.

$$CY = CY_\beta \beta + CY_{\dot{\beta}} \frac{b_w}{2V_a}\dot{\beta} + CY_p \frac{b_w}{2V_a} p + CY_r \frac{b_w}{2V_a} r + CY_{\delta_A}\delta_A + CY_{\delta_R}\delta_R \tag{3.21a}$$

$$C\mathcal{L} = C\mathcal{L}_\beta \beta + C\mathcal{L}_{\dot{\beta}} \frac{b_w}{2V_a}\dot{\beta} + C\mathcal{L}_p \frac{b_w}{2V_a} p + C\mathcal{L}_r \frac{b_w}{2V_a} r + C\mathcal{L}_{\delta_A}\delta_A + C\mathcal{L}_{\delta_R}\delta_R \tag{3.21b}$$

$$C\mathcal{N} = C\mathcal{N}_\beta \beta + C\mathcal{N}_{\dot{\beta}} \frac{b_w}{2V_a}\dot{\beta} + C\mathcal{N}_p \frac{b_w}{2V_a} p + C\mathcal{N}_r \frac{b_w}{2V_a} r + C\mathcal{N}_{\delta_A}\delta_A + C\mathcal{N}_{\delta_R}\delta_R \tag{3.21c}$$

Lateral-directional coefficients

Coefficient	Function	Coefficient	Function	Coefficient	Function
CY_β	$Const.$	$C\mathcal{L}_\beta$	$f(M, \alpha, \delta_E)$	$C\mathcal{N}_\beta$	$f(M, z, \alpha)$
$CY_{\dot{\beta}}$	$Null$	$C\mathcal{L}_{\dot{\beta}}$	$Null$	$C\mathcal{N}_{\dot{\beta}}$	$Null$
CY_p	$f(\alpha)$	$C\mathcal{L}_p$	$f(M)$	$C\mathcal{N}_p$	$f(M, z, \alpha, \delta_E)$
CY_r	$f(\alpha)$	$C\mathcal{L}_r$	$f(M, \alpha, \delta_E)$	$C\mathcal{N}_r$	$f(M, \alpha, \delta_E)$
CY_{δ_A}	$Const.$	$C\mathcal{L}_{\delta_A}$	$f(M)$	$C\mathcal{N}_{\delta_A}$	$f(M, \alpha, \delta_E)$
CY_{δ_R}	$Const.$	$C\mathcal{L}_{\delta_R}$	$f(\alpha)$	$C\mathcal{N}_{\delta_R}$	$f(\alpha)$

Table 3.7: Dependencies on the different flight parameters of each aerodynamic coefficient.

The values of lateral-directional coefficients are interpolated whether it is necessary and are introduced in the linear and non-linear model, as opposed to the drag parasitic coefficient, whose complete expression is only used in the non-linear model.

And last but not least, once the set of coefficients CL, CD and CY are computed, a change of reference must be applied in order to obtain the set C_X, C_Y and C_Z. This is defined by the expression 3.22.

$$\begin{pmatrix} C_X \\ C_Y \\ C_Z \end{pmatrix} = \begin{pmatrix} \cos\alpha & 0 & -\sin\alpha \\ 0 & 1 & 0 \\ \sin\alpha & 0 & \cos\alpha \end{pmatrix} \begin{pmatrix} -CD \\ CY \\ -CL \end{pmatrix} \tag{3.22}$$

3.1.7 Wind model

Different wind models can be found, showing a higher or lower precision in the reconstruction of the simulation environment on the basis of turbulence models and the meteorological data of the area. In this case, a simple approach to a turbulence model which utilises the Gaussian distribution is going to be developed; in addition, the velocity of the wind is going to be supposed in accordance with the data from [2] and its direction from [18].

In relation to the Gaussian model, it consists in a continuous probability distribution for real random variables. The probability density function is given by the expression 3.23, in which μ is the mean of the distribution and σ the standard deviation. [8]

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{3.23}$$

Therefore, a random value following the density function of the normal distribution will be calculated at each time step, creating a normal distribution for the wind velocity modulus, but also for the direction and the elevation angles.

The possible direction of the wind can be assumed from the wind rose of the area, so that analysing the one of Seoul from 2007 to 2020 in the figure 3.7 (see [18]), the predominant region from which the wind blows is the third quadrant; consequently, the mean wind azimuth will be approximated in this project as 45°or NE direction.

In relation to the mean wind velocity modulus, since the velocities at the ground level are not the same as the ones at higher altitudes, the meteorological sources found are no longer valid so as to find a believable value of the velocity. Hence, the wind velocity has been obtained at a logical flight altitude (based on [2]).

Finally, the mean elevation wind angle is considered as null, although a short range of values close to zero is going to be allowed. The parameters selected for the simulations (see Chapter 7) are given in the table 3.8.

Wind parameters			
Parameter	Modulus (m/s)	Direction angle (°)	Elevation angle (°)
μ	15	45	0
σ	1	5	1
Maximum	20	90	5
Minimum	10	0	-5

Table 3.8: Wind parameters employed to build the wind model of the required simulations.

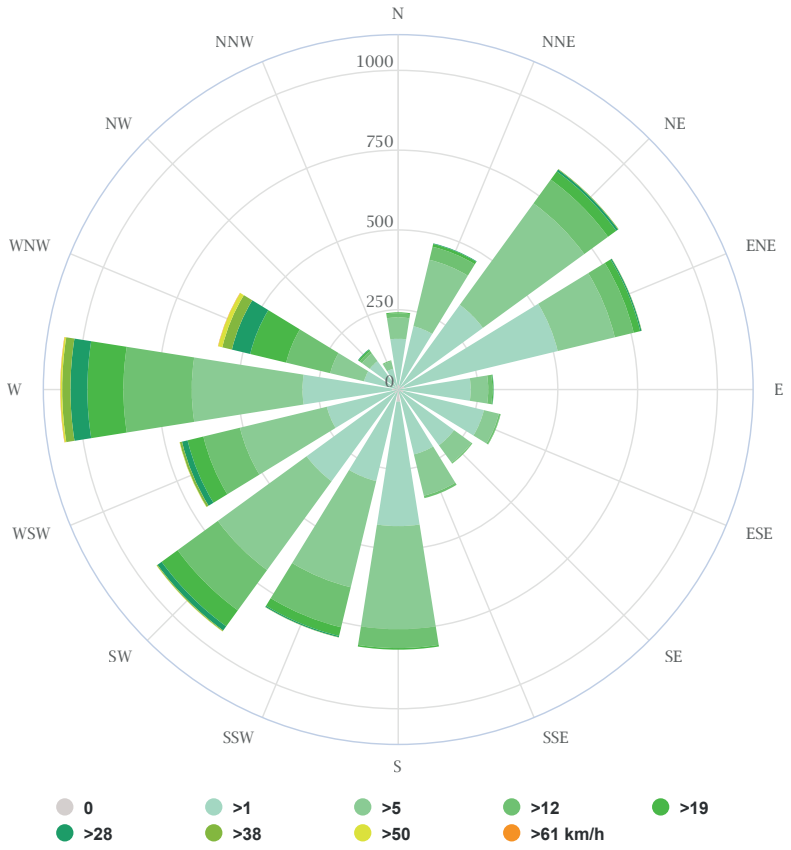

Figure 3.7: Wind rose of Seoul from 2007 to 2020, which indicates how many hours per year the wind blows from the indicated direction.

3.2 Non-linear dynamic model of the M117 bomb

In order to evaluate the point of the trajectory at which the aircraft should throw the bombs so as to impact accurately to the objective, a realistic model of the M117 must be develop. This model will no be linearized, it is just created to analyse the bomb trajectory as a ballistic missile, since it has no possibility of controlling.

The equations of this model will be the ones which were employed in the complete system, but introducing the bomb coefficients and taking into account that neither thrust nor control variables are considered in this case. The same hypothesis that have been utilised in the aircraft are considered here, with the exception of mass consumption (because there is no mass loss) and thrust ones.

3.2.1 Aerodynamic coefficients

The aerodynamic coefficients of a ballistic missile differ from the ones of an aircraft, so that specific literature is chosen for this case (see [6]).

First of all, the reference surface will be the transversal one, changing the drag formula as it is shown in the equation 3.24. The same happens in the rest of aerodynamic coefficients.

$$D = \frac{1}{2}\rho V_a^2 S_{ref} CD_b, \quad S_{ref} = \frac{\pi}{4} d_b^2 \tag{3.24}$$

The normal force coefficient[14] is decomposed in two parts, the first one corresponds with the body section and the another one with the tail. In this case, the lift coefficient only depends on the geometry, Mach number, angle of attack and the time variation in pitch angle. This is due to the fact that no elevators are used and the time variation of angle of attack does not influence to the normal force coefficient because there is no wing in the bomb.

The aerodynamic center position, as well as the normal force coefficient, is divided into body and tail. These positions are just function of the Mach number, bomb geometry and angle of attack. The final expressions are provided by the equations 3.25.

$$CN = (CN_{\alpha,body} + CN_{\alpha,tail})\alpha + (CN_{q,body} + CN_{q,tail})q \tag{3.25a}$$

$$X_{AC_b} = \frac{X_{AC_{body}}(CN_{\alpha,body}\alpha + CN_{q,body}q) + X_{AC_{tail}}(CN_{\alpha,tail}\alpha + CN_{q,tail}q)}{CN} \tag{3.25b}$$

In the case of the lateral force coefficient, CY_b, it presents the same expression as the lift coefficient due to the bomb symmetry, although utilising β and r as it is shown in the equation 3.26. The angular velocity p does not affect in none of the coefficients.

$$CY_b = (CN_{\alpha,body} + CN_{\alpha,tail})\beta + (CN_{q,body} + CN_{q,tail})r \tag{3.26}$$

In this way, pitch and yaw moments have also the same structure, as it is seen in expressions 3.27.

$$C\mathcal{M}_b = (C\mathcal{M}_{\alpha,body} + C\mathcal{M}_{\alpha,tail})\alpha + (C\mathcal{M}_{q,body} + C\mathcal{M}_{q,tail})q \tag{3.27a}$$

$$C\mathcal{N}_b = (C\mathcal{N}_{\alpha,body} + C\mathcal{N}_{\alpha,tail})\beta + (C\mathcal{N}_{q,body} + C\mathcal{N}_{q,tail})r \tag{3.27b}$$

The drag coefficient will be given by the expression 3.28. The friction and wave parasitic drag is divided into contributions from body and fins, and base parasitic drag is caused by the wake after the end of the bomb[15]. In the case of the induced drag, it has two components: the first one owing to pitch moment and the another one to yaw moment; because drag due to normal force occurs in both motions.

[14] Which is the lift coefficient.

[15] This coefficient corresponds to coast conditions.

$$\begin{aligned} CD_b = CD_{b_0} + CD_{b_i} =&[(CD_0)_{body,friction} + (CD_0)_{tail,friction} + (CD_0)_{base} + \\ &+(CD_0)_{body,wave} + (CD_0)_{tail,wave}] + [\frac{(CN_{\alpha,tail}\alpha + CN_{q,tail}q)^2}{\pi A R_b e_b} + \\ &+ \frac{(CN_{\alpha,tail}\beta + CN_{q,tail}r)^2}{\pi A R_b e_b}] \end{aligned} \tag{3.28}$$

Finally, the roll moment coefficient is supposed to be null. In the real behaviour of a free fall of the bomb, this is no longer true, but since neither aerodynamic angles nor angular velocities q and r have effect on the roll moment and no deviations[16] are taken into account, the M117s will not have roll motion during the simulation. As a consequence, Euler angle ϕ and angular velocity p are considered to be zero.

3.2.2 System of equations

The system that must be utilised to model the behaviour of the M177s in a free fall are basically the composition of equations 3.11c, 3.12c, 3.13 and 3.14b. The complete system of equations of the bomb M117, once the proper modifications according to the bomb properties have been introduced, are given by the set of expressions 3.29. The reference systems are the similar to the ones employed in the aircraft.

$$\begin{cases} m_b(\dot{u} + qw - rv) = \frac{1}{2}\rho S_{ref} V_a^2 C_{X_b} - m_b g \sin\theta \\ m_b(\dot{v} + ru) = \frac{1}{2}\rho S_{ref} V_a^2 C_{Y_b} \\ m_b(\dot{w} - qu) = \frac{1}{2}\rho S_{ref} V_a^2 C_{Z_b} + m_b g \cos\theta \\ \frac{1}{2}\rho S_{ref} c_b V_a^2 C_{\mathcal{M}_b} = I_{yy_b}\dot{q} \\ \frac{1}{2}\rho S_{ref} c_b V_a^2 C_{\mathcal{N}_b} = I_{zz}\dot{r} \\ q = \dot{\theta} \\ r = \dot{\psi}\cos\theta \\ \frac{\mathrm{d}x}{\mathrm{d}t} = u\cos\psi\cos\theta - v\sin\psi + w\sin\theta\cos\psi \\ \frac{\mathrm{d}y}{\mathrm{d}t} = u\cos\theta\sin\psi + v\cos\psi + w\sin\theta\sin\psi \\ \frac{\mathrm{d}z}{\mathrm{d}t} = u\sin\theta - w\cos\theta \end{cases} \tag{3.29}$$

3.3 Linear dynamic model of the complete system

In this section, the linear dynamic model of the whole system is going to be calculated and simplified to a Linear Time Invariant system (LTI) in order that the control laws can be applied to the system. This model will consist of 11 linearized equations with respect to a given equilibrium point, in which system variables, perturbations and control variables affect to the behaviour of the whole linearized

[16]The bomb can roll due to changes in pressure in one fin or geometric asymmetries during the free fall.

system. Thus, first of all, it is mandatory to build the non-linear system of 11 equations and calculate the equilibrium point, and then, the linearized model is calculated. Finally, an example of linearized system is shown at the end of the section.

3.3.1 Equilibrium point

The equilibrium point is the one at which stationary motion is achieved, therefore, at this set of values, the aircraft will fly without changes in direction, velocity or Euler angles values. This point sets those values used in the linearization, but it must be taken into account it could not be the initial conditions of flight.

To calculate this point, all derivatives and perturbations are set to zero and the 5 equations of the vertical plane motion (see equations 3.30) are chosen to obtain the trimmed point (null upward velocity). This is conducted by means of the code in the Algorithm 1, firstly the trimmed variables are defined and an optimisation variable is created, afterwards, a non-linear solver is utilised with the 5 equations mentioned.

$$\begin{cases} v = 0 \\ \phi = \psi = 0 \\ p = q = r = 0 \\ \delta_A = \delta_R = 0 \end{cases} \qquad \begin{cases} V_{x_w} = 0 \\ V_{y_w} = 0 \\ V_{z_w} = 0 \\ CD_{wave} = 0 \end{cases} \qquad \begin{cases} T + \frac{1}{2}\rho S_w V^2 C_X - mg\sin\theta = 0 \\ \frac{1}{2}\rho S_w V^2 C_Z + mg\cos\theta = 0 \\ C_{\mathcal{M}} = 0 \\ V = u\cos\theta + w\sin\theta \\ u\sin\theta - w\cos\theta = 0 \end{cases} \tag{3.30}$$

Algorithm 1 Trimmed Point Calculation

1: define $V_{trim}, z_{trim}, \psi_{trim}$
2: $[optimization\_variable] = optimvar('var', 5)$
3: $[equations] = trimmed\_eqs(optimization\_variable, V_{trim}, z_{trim}, \psi_{trim})$
4: define $solving\_options, initial\_values$
5: $[solution_{trim}] = fsolve(equations, initial\_values, solving\_options)$
6: $[full\_solution_{trim}] = total\_solution(solution_{trim})$
7: **return** $full\_solution_{trim}$

3.3.2 Linearized state-space model

The rearranged complete model of equations represents a state-space system, which is given by the expression 3.31a; however, the system of equations can not be obtained symbolically in such form due to the high complexity of the model, and as a consequence, the whole system is linearized, obtaining the expression 3.31b.

$$\dot{x} = Ax + Bu + A_d d \tag{3.31a}$$

$$\dot{\Delta x} = A \cdot \Delta x + B \cdot \Delta u + A_d \cdot \Delta d = A \cdot \Delta x + B \cdot \Delta u + A_d \cdot d \tag{3.31b}$$

In the state-space model, x refers to the state variables, u to the control variables and d to perturbation variables. Since the trajectory and velocity are the objective of control, the positions x and y will not be state variables, although the body velocity will, as it is explained in the Chapter 4. The control variables are the throttle lever and control surfaces of the aircraft, as it has been mentioned previously. The perturbations will be the wind velocities in the local horizon frame, but also the wave drag coefficient CD_{wave}, due to the fact that this drag only exists at higher velocities than the one used in the equilibrium point, and therefore, it can be modelled as a perturbation.

In relation to state-space model representation in Simulink, it is mandatory to specify the outputs of the system by means of another system defined by the matrices C and D, despite the fact that all state variables are measured. In addition, due to the Simulink blocks configuration, in the state-space model is required that the control and perturbation matrices are joined in a single one, hence, the model utilised in this project is the one corresponding to the expressions 3.32.

$$\overrightarrow{\Delta x} = \left\{ \begin{array}{cccccccccccc} \Delta z & \Delta u & \Delta v & \Delta w & \Delta\phi & \Delta\theta & \Delta\psi & \Delta p & \Delta q & \Delta r & \Delta V \end{array} \right\}^t \tag{3.32a}$$

$$\overrightarrow{\Delta u} = \left\{ \begin{array}{cccccccc} \Delta\delta_P & \Delta\delta_E & \Delta\delta_A & \Delta\delta_R & V_{x_w} & V_{y_w} & V_{z_w} & CD_{wave} \end{array} \right\}^t \tag{3.32b}$$

$$\begin{cases} \dot{\Delta x}_{11\times1} = A_{11\times11} \cdot \Delta x_{11\times1} + \begin{bmatrix} B & A_d \end{bmatrix}_{11\times8} \cdot \Delta u_{8\times1} \\ y_{11\times1} = C_{11\times11} \cdot \Delta x_{11\times1} + D_{11\times8} \cdot \Delta u_{8\times1} \end{cases} , \quad \begin{cases} C_{11\times11} = I_{11\times11} \\ D_{11\times8} = [0]_{11\times8} \end{cases} \tag{3.32c}$$

The system matrix A and control one $\begin{bmatrix} B & A_d \end{bmatrix}$ are obtained through Taylor series of first order from the equilibrium point, therefore, in fact, they are essentially Jacobian matrices of the function $f(\overrightarrow{x}, \overrightarrow{u})$, which contains the 11 equations, with respect to the state variables and control+perturbation ones, respectively. The symbolic expression of both matrices is shown in the equations 3.33 and is calculated by means of the Algorithm 2.

In the algorithm, the whole set of variables is defined and their symbolic initial values (which define the linearization point) as well, after that, the 11 equations are derived with respect to each one of the variables with respect to the initial values to build a symbolic matrix of coefficients. Then, the trimmed point is obtained from the Algorithm 1 and the symbolic coefficients are substituted by the trimmed point parameters. Finally, the coefficients are properly arranged and the matrices are calculated from these ones.

$$A_{ij} = \left. \frac{\partial f_i(\overrightarrow{x}, \overrightarrow{u})}{\partial x_j} \right|_{\overrightarrow{x}_0 \overrightarrow{u}_0} \qquad B_{ik} = \left. \frac{\partial f_i(\overrightarrow{x}, \overrightarrow{u})}{\partial u_k} \right|_{\overrightarrow{x}_0 \overrightarrow{u}_0} \tag{3.33a}$$

$$\dot{\Delta x} = \begin{pmatrix} \frac{\partial f_1}{\partial z}\Big|_{\vec{x}_0 \vec{u}_0} & \cdots & \frac{\partial f_1}{\partial V}\Big|_{\vec{x}_0 \vec{u}_0} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_{11}}{\partial z}\Big|_{\vec{x}_0 \vec{u}_0} & \cdots & \frac{\partial f_{11}}{\partial V}\Big|_{\vec{x}_0 \vec{u}_0} \end{pmatrix} \Delta x + \begin{pmatrix} \frac{\partial f_1}{\partial \delta_P}\Big|_{\vec{x}_0 \vec{u}_0} & \cdots & \frac{\partial f_1}{\partial CD_{wave}}\Big|_{\vec{x}_0 \vec{u}_0} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_{11}}{\partial \delta_P}\Big|_{\vec{x}_0 \vec{u}_0} & \cdots & \frac{\partial f_{11}}{\partial CD_{wave}}\Big|_{\vec{x}_0 \vec{u}_0} \end{pmatrix} \Delta u \tag{3.33b}$$

Algorithm 2 Linearization of the system

define $V_{trim}, z_{trim}, \psi_{trim}$
$var_1 = \left[z\ u\ v\ w\ \phi\ \theta\ \psi\ p\ q\ r\ V\ \dot{z}\ \dot{u}\ \dot{v}\ \dot{w}\ \dot{\phi}\ \dot{\theta}\ \dot{\psi}\ \dot{p}\ \dot{q}\ \dot{r}\ \dot{V}\right]$
$var_2 = [\delta_P\ \delta_E\ \delta_A\ \delta_R\ V_{x_w}\ V_{y_w}\ V_{z_w}\ CD_{wave}]$
$variables = [var_1\ var_2]$
$var_{1_0} = \left[z_0\ u_0\ v_0\ w_0\ \phi_0\ \theta_0\ \psi_0\ p_0\ q_0\ r_0\ V_0\ \dot{z}_0\ \dot{u}_0\ \dot{v}_0\ \dot{w}_0\ \dot{\phi}_0\ \dot{\theta}_0\ \dot{\psi}_0\ \dot{p}_0\ \dot{q}_0\ \dot{r}_0\ \dot{V}_0\right]$
$var_{2_0} = [\delta_{P_0}\ \delta_{E_0}\ \delta_{A_0}\ \delta_{R_0}\ 0\ 0\ 0\ 0]$
$variables_0 = [var_{1_0}\ var_{2_0}]$
for $i = 1 : 11$ **do**
    $[equation_{number}] = eval(append('ec', string(i)))$;
    for $j = 1 : length(variables)$ **do**
        $[coefficients(i,j)] = subs(diff(equation_{number}, variables(j)), variables, variables_0)$;
    end for
end for
$values_0 = trimmed\_point(V_{trim}, z_{trim}, \psi_{trim})$
$matrix_{cfs} = double(subs(coefficients, variables_0, values_0))$
$matrix_{inv} = inverse\_matrix(matrix_{cfs}(1:11, 23:30))$
$A = -matrix_{inv} \cdot matrix_{cfs}(1:11, 1:11)$
$B = -matrix_{inv} \cdot matrix_{cfs}(1:11, 23:30)$
$C = eye(11)$
$D = zeros(11, 8)$

The longitudinal and lateral-directional motions have been not divided due to the fact that both are coupled in the system; thereby, longitudinal state variables affect to lateral-directional ones and vice versa[17]. From an example of linearized state-space model, from the equilibrium point utilised in the Chapter 7, at velocity $V = 215m/s$ and altitude $z = 787m$, this fact is fairly noticeable (see equations A.4-A.5, Annex A).

[17]However, the system will be divided into both motions in the control laws design so as to facilitate the controlling of the aircraft.

4 | Control design

In this chapter, different alternatives of controller are briefly analysed in order to find an optimal method by means of exposing advantages and disadvantages in terms of complexity and quality of the results. Afterwards, two control design methods (LQR and MPC) are chosen for the purpose of this project, giving a complete justification of this selection and an explanation of their schemes in closed loop.

Finally, the scheme used in Simulink to perform the simulation in FlightGear of the flight is shown and explained as well.

4.1 Control law alternatives

The high non linearity of the system provokes that the control of the aircraft becomes a quite hard work, hence it is very important to find a control scheme that provides enough robustness facing perturbations[1] and enough aggressiveness to follow successfully the sudden changes in the reference. Ones of the most typical control strategies are described below.

4.1.1 PID control

The control by PIDs is the most extended tool in the industrial environment due to its development and satisfactory results. A PID controller enables to control a system in closed loop in order that it achieves the desired output, it is composed of three elements that provide a proportional, integral and derivative action. The equation 4.1 defines the control action at each time depending on the error signal and PID parameters and, the figure 4.1, shows the scheme in closed loop used in a PID configuration (see [23]).

$$c(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{\partial e(t)}{\partial t} \tag{4.1}$$

The best advantages of this control method are its excellent robustness and the design easiness owing to the numerous techniques, such as the root locus design.

[1] Not only wind and wave drag perturbations, but also the behaviour of non linear system which is different from the expected linear one.

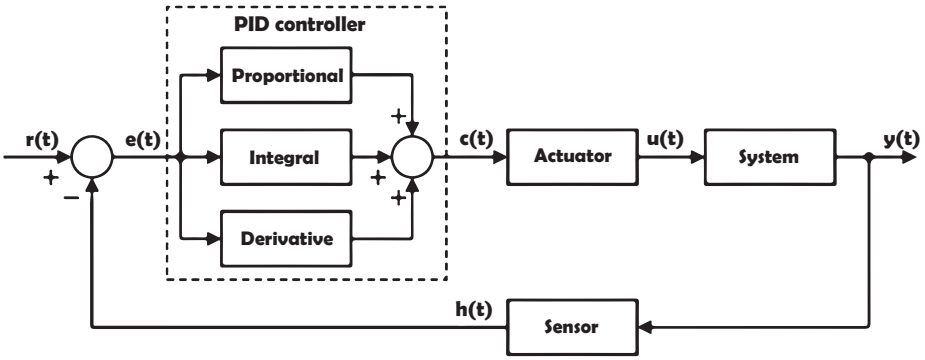

Figure 4.1: PID controller scheme.

However, the main issue consists in the fact that the PID controller is only applicable to SISO systems, composed of an unique input and output, and seeing that in this project a MIMO system is employed, which takes multiple inputs and outputs, the usage of this method is more complex because every combination of input-output would require a PID controller. In addition, the non-linearity of the system complicates its application.

4.1.2 Full State Feedback

The FSF (Full State Feedback) or pole placement, is a method employed to modify the position of all poles of the system in closed loop configuration to a desired one, provided that the system fulfils the control requirements. In this case, this control method is designed for MIMO systems, working with state-space models, which allows to incorporate it to the control system of the aircraft much more easier than a group of PIDs.

In the FSF control method, the control signal is proportional to the state vector by means of a matrix K, taking the expression $u(t) = -Kx \cdot (t)$ and giving as a result that the new poles are the eigenvalues of the matrix $A - BK$. The scheme is given by the image 4.2.

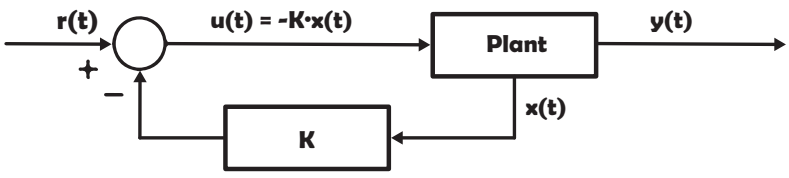

Figure 4.2: FSF controller scheme.

The main issue of this method is the search of the desired poles, due to the fact that at each conditions the required poles will be different and specifications such as error or settling time of each variable must be adjusted. In a system with a high quantity of variables and huge complexity, this fact can suppose the inability of applying this method; consequently, this control strategy is not commonly used.

4.1.3 Linear Quadratic Regulator

Following the basis of the FSF method, the Linear Quadratic Regulator was born in order to develop a controller that also modifies the dynamic of the system in closed loop by means of state feedback, but in this case, calculating the static gain on the basis of the objective of diminishing a cost function which measures the change in state and control variables. This cost function is given by the expression 4.2 (see [16]), although typically the matrix N_{xu} that weighs the combinations of state and control variables is considered null, so that the cost function becomes a quadratic one.

$$\mathcal{J} = \int_0^\infty (x^T Q x + u^T R u + 2x^T N_{xu} u) dt \tag{4.2}$$

Matrices Q and R weighs states and control signals (respectively) and are diagonal and positive defined ones. The higher the weigh value, the lower the state or control action associated to that value is.

Regarding the matrix K calculation, its value is obtained by the mathematical resolution of the Riccati equation from a combination of cost matrices and the ones of the system $\dot{x} = Ax + Bu$. Taking the matrix N_{xu} as null, as it has been said before, the expressions of Riccati's solution S and the final form of feedback matrix K are given by the equations 4.3. [16]

$$A^T S + SA - SBR^{-1}B^T S + Q = 0 \tag{4.3a}$$

$$K = R^{-1}B^T S \tag{4.3b}$$

Selecting carefully the values of these matrices, the user can vary the dynamic properties of the system and the control action capabilities in an easier way than in the FSF method; therefore, this strategy is much more useful and applicable to this project than the FSF. The LQR method is also well known for its satisfactory robustness properties, so that an adequate stability is obtained in the results of control.

Nevertheless, the incorporation of restrictions is not a simple goal, it usually requires a considerable quantity of attempts to find an optimal combination of matrices Q and R, and in many cases, it is not possible due to the fact that the values of Q and R sometimes cause that the minimisation problem becomes unsolvable, giving as a result an indeterminacy in the solution finding.

4.1.4 Model Predictive Control

Model predictive control (MPC) is an advanced method used mainly in MIMO systems which is employed to control a system while a set of constraints is properly satisfied. In addition, MPC utilises the model of a system to predict its future

behaviour, and it solves an online optimisation algorithm to select the best control action that drives the predicted output to the reference.

The controller simulates multiple scenarios using more or less aggressive control actions[2] from the actual instant of time to the prediction horizon, which is the time at which the desired reference is thought to be achieved. In order that the controller ensures that the predicted future plant output tracks the desired reference, an optimiser is used to try to diminish the error with respect to the reference and find the best scenario of control actuation. This optimisation process is defined by the minimisation of a cost function similar to the LQR's one. In the figure 4.3, it is intuitively shown how the optimiser works.

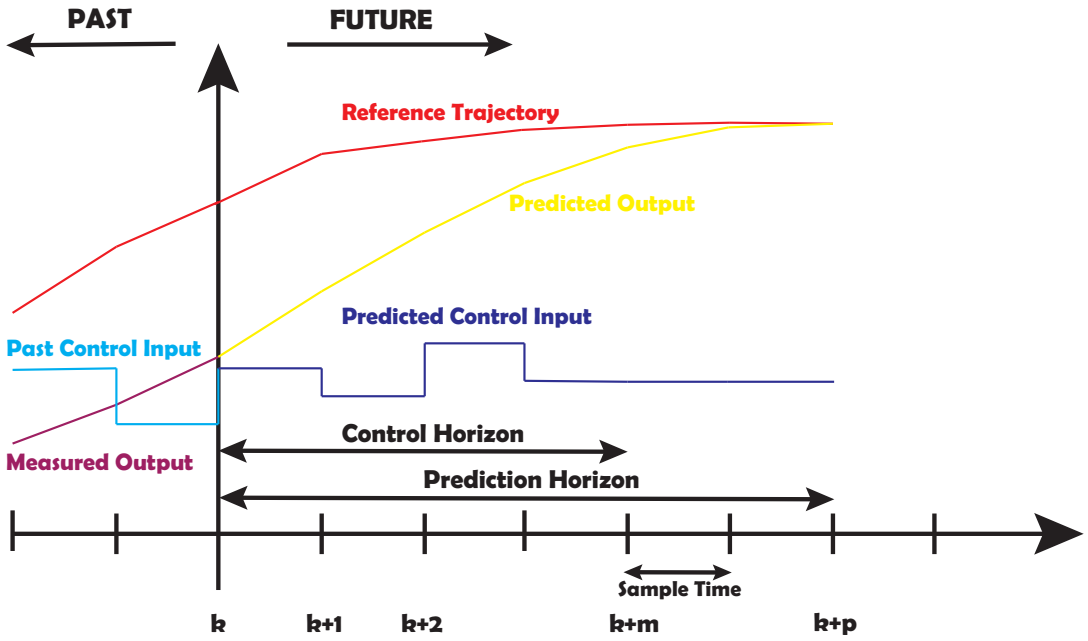

Figure 4.3: MPC future prediction scheme.

The prediction horizon parameter has to be selected taking into account the capabilities of the system, since a quite demanding value could make that the reference is impossible to be followed and a higher one would produce that the error in the reference is not being reduced fast enough. In addition, the sample time must be chosen carefully too, because high values can lead to poor reactions when facing disturbances and low ones increase heavily the computation load.

Furthermore, there is another important design parameter: the control horizon. This value defines the number of times in which the controller can modify the control input in order to reduce the error in the reference, after those times, the control input remains as constant in the prediction process (as it can be observed in the figure 4.3). The control horizon must be always lower than the prediction one, but can not have any value: low values could give poor quality manoeuvres and the reference would not be followed successfully, on the other hand, excessive values lead to an important increment in the computational effort and demanding control actions.

There are different types of MPC controllers depending on the application of

[2]Although those control actions are always carried out between imposed limits.

constraints and the cost function and the linearity of the system. MPC controllers can be divided into (see [17]):

1. Linear: the system is linear or non-linear, but the constraints are linear and the cost function is defined as simple quadratic. In these controllers, the quadratic cost function has a single global minimum at each step.

   (a) Time-invariant: this is the best option when the system is linear or has a strong linear behaviour, since the controller utilises all the time the same plant model and it is not necessary to linearize the model again at each time step.

   (b) Adaptive: if the system is non-linear and the values of the state variables are very far from the linearization point, a time-invariant could not diminish the error in the reference properly; in that way, a acceptable option would be to employ an adaptive MPC controller and recalculate the plant model by means of a new linearization at each time step.

   (c) Gain-scheduled: in the case at which there are different number of states and constraints across the different operating conditions, a gain-scheduled controller is mandatory. Different MPC controllers are designed for each operation, so that at the time that there is an operation change, a switch algorithm reject the actual MPC and selects the next one.

2. Non-linear: the system is non-linear, although the constraints and the cost function are non-linear as well. The optimisation process becomes non-convex and there are multiple global optimal solutions, therefore finding the most adequate one is a really hard task in this controller.

MPC controllers offer satisfactory conditions of versatility and easiness for the user and more adequate results, but computational effort is increased with respect to the LQR method, since the MPC has to calculate the future cases at each time and LQR has a single solution from the beginning. In addition, the complexity of utilising the MPC controller can grow a lot depending on type chosen; for example, the non-linear one is the most difficult to apply.

4.1.5 Robust Control

Controllers based on robust control methods are explicitly built in order to deal with uncertainty, considering a system with possibly multiple sources of uncertainties, noises, and disturbances and combining frequency design techniques such as Bode or Nyquist and LQR control.

The general diagram of system which employs a robust control strategy is shown in the figure 4.4 (see [42]), where P is the plant of the system, K_1 in combination with signals q and t are the set of all possible uncertainty, K_2 the controller, u the control signal, y the output and w a vector signal including noises, disturbances, and reference signals.

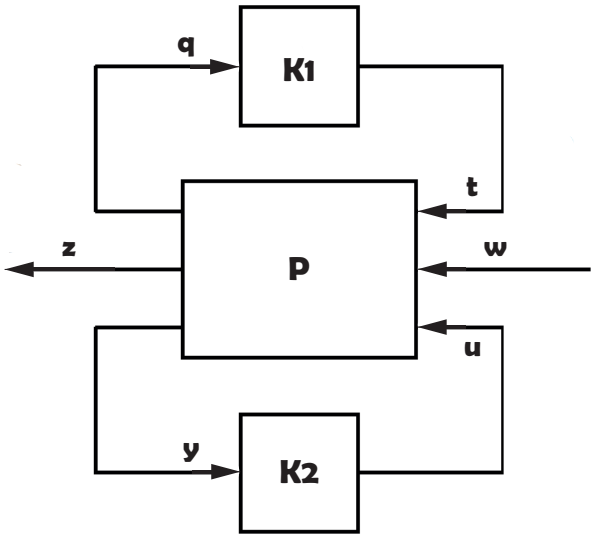

Figure 4.4: Robust Control scheme.

These controllers proportionate a very stable response when the system faces perturbations or non-modelled dynamic parts of the system, so that it would very useful in the case of an aircraft's system due to the high non-linearity of the plant and possible high wind velocities.

However, the mathematical complexity of these methods is very high and the understanding process of the model is quite long, so that it is harder to introduce them into the control design than other ones.

4.2 Chosen control strategies

Among all the alternatives presented in this work, the best two control methods selected to be implemented in the F-86 Sabre's system are the LQR and Linear Time-invariant MPC.

The LQR has been chosen since it is the improvement of the FSF control method and offers adequate conditions of robustness when the system faces perturbations. It also provides distinct possibilities of configuration, so that, for instance, the longitudinal and lateral directional motions can be separated as it is seen below.

In relation to the MPC, among MPC controllers the two best available options were the Linear Time-invariant and Adaptive ones, due to the fact that the Gain-scheduled and non-linear ones could not guarantee the global stability of the system and, consequently, other complex techniques should be utilised together with these MPCs, making the whole procedure more difficult. The Adaptive MPC is really the best one, because it is able to manage non-linear systems and accept a new plant model for the system at each iteration; nevertheless, its implementation is more difficult and the model is also fairly complex to be successfully linearized in a motion out of the vertical plane one, since the aircraft would be not trimmed. Hence, the Time-invariant option has been chosen, which will supply enough aggressiveness

in the control to conduct a proper reference tracking and adequate capabilities of robustness.

Both in LQR as in MPC, the sample time must be small enough to allow an acceptable control, basing on the modes of the system matrix in open loop. From the eigenvalues of the system matrix A (see equations B.1), with the equilibrium point used in the Chapter 7, the fastest modes are λ_6 and λ_7, since their real parts are more negative than the ones of other modes. To obtain a proper response of the controller according to the Nyquist theorem (see [20]), the sample time must be less than the half of the settling time of the system, which will be the inverse of the most negative real part from the eigenvalues; hence, it must be satisfied:

$$T_s < \frac{1}{2 \cdot Re(\lambda_6)} = 0.18s \tag{4.4}$$

In this case, the sample time will be much more smaller than that restriction: $T_s = 0.021s$. This value has been carefully adjusted, seeing that it determines the performance of the MPC controller in the prediction and control horizons, so that the optimal sample time has been selected in terms of computational time and effectiveness of the controller.

Regarding the schemes of the complete system in Simulink, LQR and MPC case are going to present a very similar structure, although there are some differences depending on the controller and its required inputs together with the characteristics of the output that it provides.

4.2.1 LQR

The general scheme used in the LQR case is shown in the figure 4.5, in which the guidance algorithm (explained in Chapter 5) gives the references in altitude, velocity and roll and yaw angles that the aircraft must follow, based on the actual total parameters of the aircraft (not state variables from the equilibrium point) and the designed dynamic path (see Chapter 6). Afterwards, the LQR controller processes the references and the actual state variables of the airplane (now, they are from the equilibrium point) and provides its output which will be the control actions. Then, the control limitations explained in the Section 3.1.3 are applied to the signal from the LQR controller, giving as a result control actions that the aircraft is able to perform.

Once the control signals are generated, they are translated into complete variables in the control actions block, since they are given with respect to the equilibrium point. After, they are introduced in the non-linear model block and are computed, in order that the whole set of equations of the mathematical model of the system can calculate the future value of the variables which are result of those control actions. In addition, the wind model is introduced into the non-linear block, together with the time[3]. The output of the block is composed of the 13 variables (the 11

[3]The time is only an input so as to specify to the block when the initial conditions must be utilised.

state variables and the x and y positions) that will be the input of the LQR block, therefore, another block called "Total Variables Building" is used to generate the group of complete variables, which are needed to the functioning of the Guidance block. In addition, these complete variables will include the 11 state variables, x and y coordinates and the course angle Ψ.

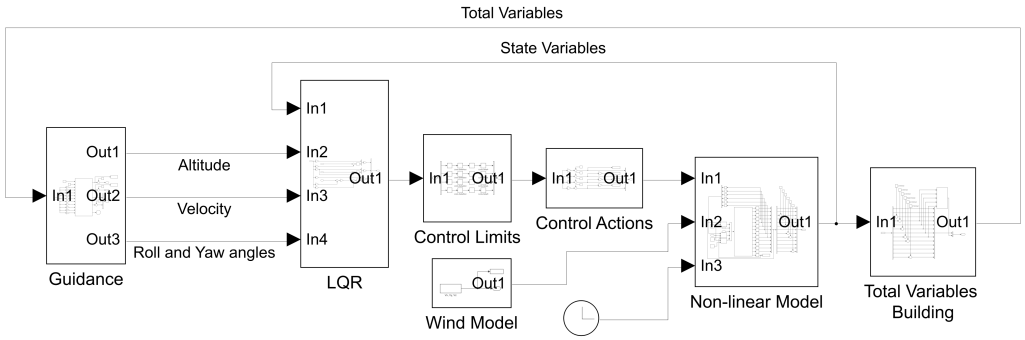

Figure 4.5: General scheme with LQR controller.

The guidance block scheme is shown in the figure 4.6. The waypoints of the route obtained in the Chapter 6 are an input of the Matlab function which contains the Guidance algorithm (see Section 5.3), together with the sample time and the rest of variables needed to apply the algorithm $(x, y, z, u, v, w, V, \Psi, \phi, \theta)$. The outputs will be the altitude, velocity and roll and yaw angles of reference in increments, therefore the subtraction of trimmed conditions values is applied at each one (except to ϕ, whose trimmed value is already null). Furthermore, another output will be the signal that feeds the STOP block, which will end the simulation if this signal is a 1. The simulation will be stopped whether the aircraft has completed the reference path (see Section 5.3).

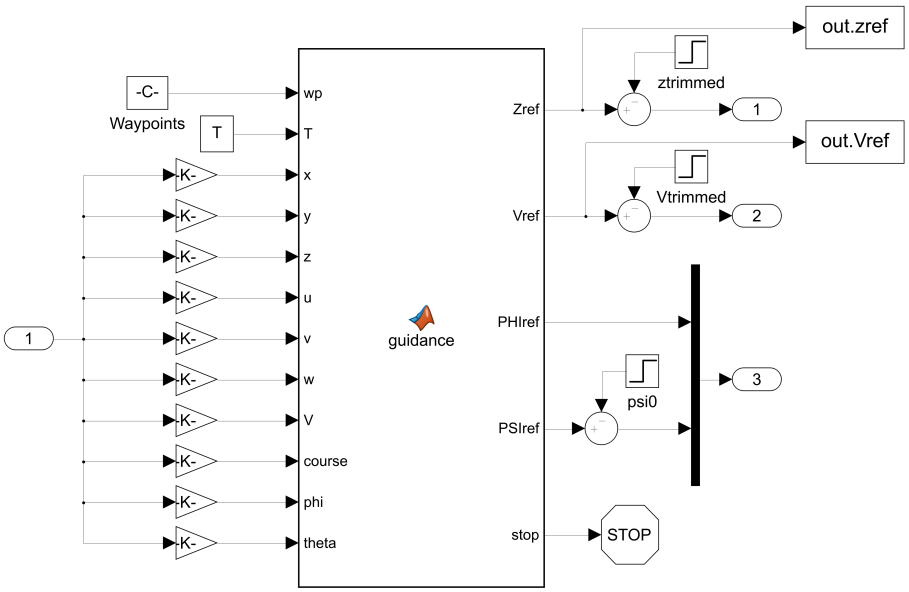

Figure 4.6: Guidance block scheme.

Regarding the Control Limits block, it is based on the procedure shown in the figure 3.6, in which the ranges of maximum and minimum values in amplitude and

time variation are applied to the control signal from LQR controller. The image 4.7 shows the structure of the block.

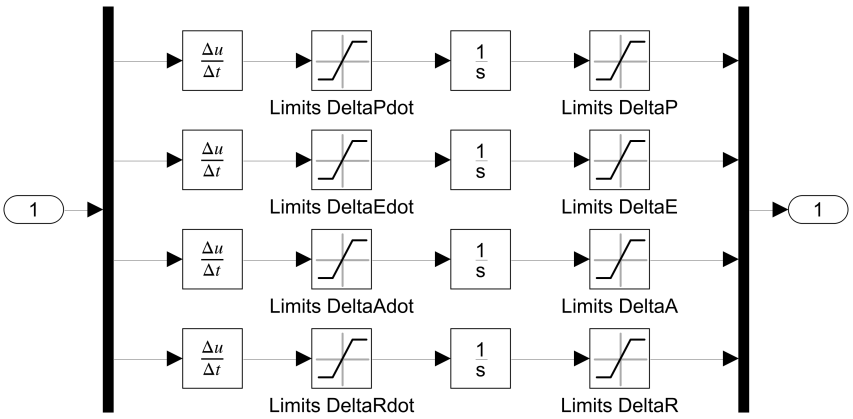

Figure 4.7: Control Limits block scheme.

In the Control Actions block, the control variables in increments are converted into total control ones by means of adding the trimmed values. The control signals are saved to their display in the Chapter 7. This block is shown in the figure 4.8.

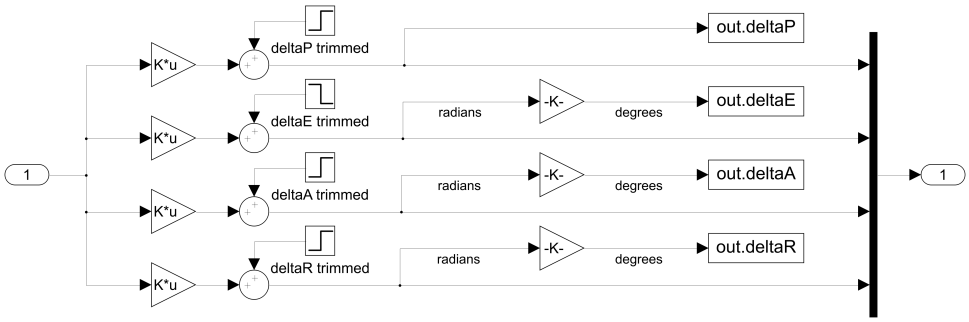

Figure 4.8: Control Actions block scheme.

In relation to the non-linear model block, which can be observed in the figure 4.9, the configuration is more complex. Switch blocks are used to change from the initial conditions at time $t = 0s$ to the instantaneous variables given by the Matlab function "non-linear model". In this function, all state variables, coordinates x and y and fuel mass (FM) are inputs, but also the derivatives of u, v, w, ϕ, θ and ψ. In addition, the total control action variables and wind velocities are also inputs. Attending to the outputs, they will be the derivatives of all state variables and positions x and y, together with the fuel mass consumption, which are result of the application of the flight mechanics model described in the Chapter 3.

The function entirely works with total variables, so that the trimmed values are subtracted to them (only those which are not zero) and the variables in increments that are needed in the controller are obtained as result. Furthermore, the outputs of the Matlab function, which are already derivatives, are not used in the input, they are integrated and then derived: apparently, this could not make sense, however, this structure is conducted so as to avoid algebraic loops during the simulation, which increment the computational time excessively.

Moreover, the fuel mass consumption is the unique variable that is not sent to the output of the block, it is utilised in the Matlab function exclusively. Its data, once integrated, is saved in order to be represented in the final results (see Chapter 7, figure 7.14).

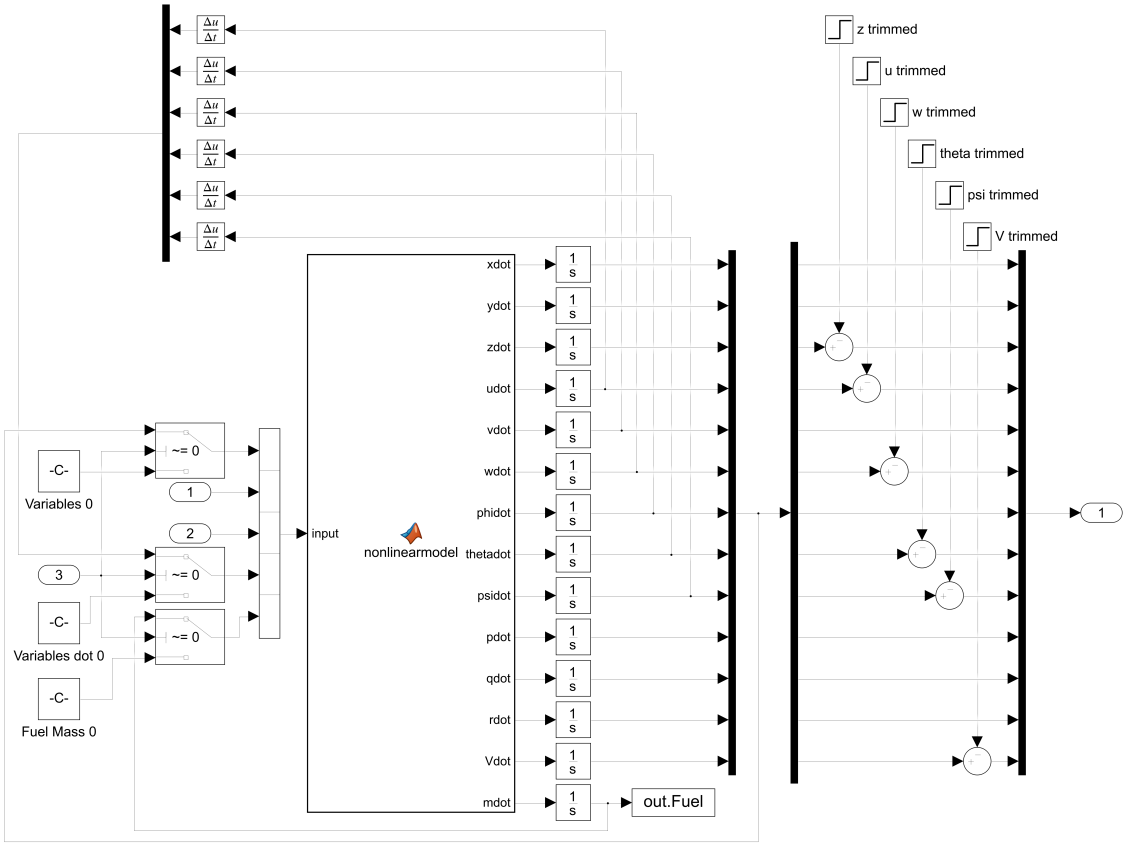

Figure 4.9: Non-linear model block scheme.

Once the state variables (together with coordinates x and y) are created, the complete variables are generated by means of the Total Variables Building block. The procedure is the inverse one, the trimmed values different from zero are summed. The course angle is calculated in a Matlab function on the basis of the relation explained in the Section 5.2.1 (see equation 5.3). This block is presented in the following image 4.10:

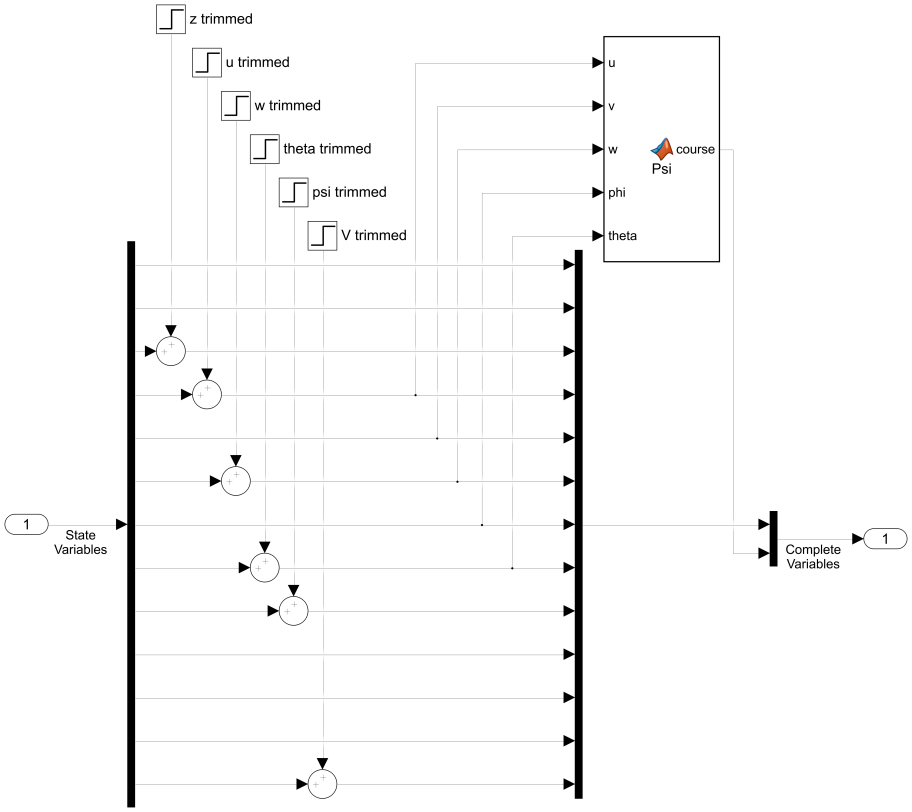

Figure 4.10: Total Variables Building block scheme.

Finally, the design of the LQR structure is explained. Although the longitudinal and lateral-directional motions are coupled, it is necessary to separate the motions in order to develop an LQR controller which is able to generate suitable control actions. Therefore, four different sub-controllers are created so as to fulfil the requirements: K_{1z} is related to the altitude, K_{1V} to the speed and K_{21} and K_{22} to the roll and yaw angles tracking. The following in the reference of ϕ and ψ is complex, since the use of static gains does not facilitate an adequate response and the error in the reference becomes very large (the gains do not provide enough aggressiveness in the ailerons and rudder deflection), so that those two gains are utilised in an unusual configuration in order to manipulate ailerons and rudder and get a proper reference tracking.

In this way, K_{1z} and K_{1V} will assign the required elevator deflection and throttle lever action to follow the reference in altitude and velocity from the guidance algorithm, respectively. In relation to roll and yaw angles, K_{21} will create a reference of angular velocities p and r on the basis of the error in those angles with respect to the reference; however, this error is not integrated as in the other cases, since a very aggressive and fast action is demanded. Once the reference angular velocities are calculated, the gain K_{22} is applied to the error in them, but also to the actual roll and yaw angles values, obtaining a more appropriate control action in ailerons and rudder instead of the one resulting of applying an unique gain. The figure 4.11 shows the structure of the LQR controller implemented in this project.

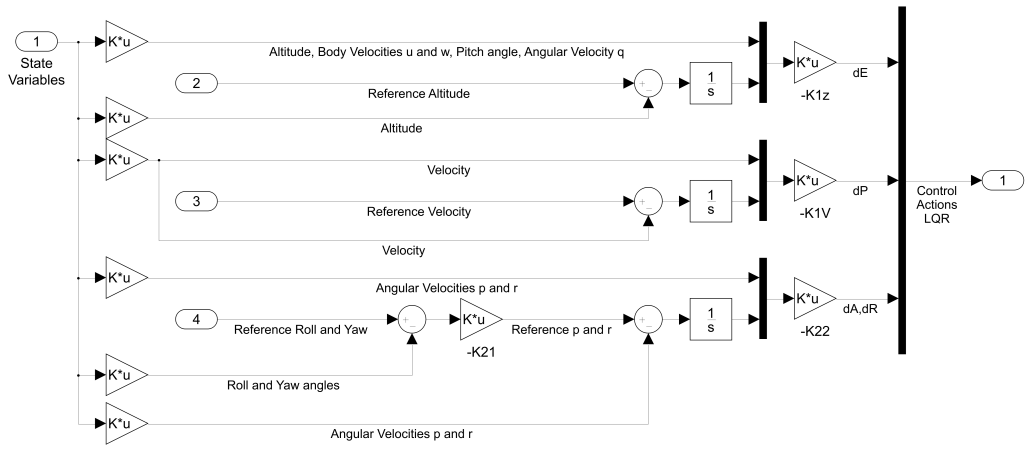

Figure 4.11: LQR controller scheme.

Therefore, there will be four cost functions to optimise with four different Q and R weigh matrices, as it is seen in the equations 4.5, having to solve the Riccati's equation each time to calculate every one of the gains (see equations 4.3). The whole solution in K matrices is taken in each one of the cases, except for the K_{21} one, since as it has been said before, this gain is only to generate an acute control action, therefore, the actual values of roll and yaw angles are not taken into account to define the reference angular velocities p and r. In addition, system and control matrices are extracted from the main ones at each case, making subdivisions of them: this entails a problem, in such a way that the values of these matrices that demonstrate the coupling of the two motions are lost, so that the controller is provided with less information; however, this division must be done in order that the controller can focus on where exactly the errors are and apply the corresponding control action in the elevator, ailerons, rudder or throttle lever at each case.

In order to build the K gains, amplified system matrices must be created. These matrices include the errors (ϵ) in the variables that are wanted to be reduced, giving as a result an appropriate reference tracking. Hence, the state vector variables will be an amplified one, which also contains the variables of error. In addition, matrices Q will present in their diagonals the weigh parameters for each variable of the new amplified state vector.

$$\overline{x}_{1z} = \begin{pmatrix} \Delta z & \Delta u & \Delta w & \Delta\theta & \Delta q & \epsilon_z \end{pmatrix}^T, u_{1z} = (\Delta\delta_E) \Rightarrow \mathcal{J}_1 = \int_0^\infty (\overline{x}_{1z}^T Q_{1z} \overline{x}_{1z} + u_{1z}^T R_{1z} u_{1z}) dt \tag{4.5a}$$

$$\overline{x}_{1V} = \begin{pmatrix} \Delta V & \epsilon_V \end{pmatrix}^T, u_{1V} = (\Delta\delta_E) \Rightarrow \mathcal{J}_2 = \int_0^\infty (\overline{x}_{1V}^T Q_{1V} \overline{x}_{1V} + u_{1V}^T R_{1V} u_{1V}) dt \tag{4.5b}$$

$$\overline{x}_{21} = \begin{pmatrix} \Delta\phi & \Delta\psi & \epsilon_\phi & \epsilon_\psi \end{pmatrix}^T, u_{21} = \begin{pmatrix} \Delta p & \Delta r \end{pmatrix}^T \Rightarrow \mathcal{J}_3 = \int_0^\infty (\overline{x}_{21}^T Q_{21} \overline{x}_{21} + u_{21}^T R_{21} u_{21}) dt \tag{4.5c}$$

$$\overline{x}_{22} = \begin{pmatrix} \Delta p & \Delta r & \epsilon_p & \epsilon_r \end{pmatrix}^T, u_{22} = \begin{pmatrix} \Delta\delta_A & \Delta\delta_R \end{pmatrix}^T \Rightarrow \mathcal{J}_4 = \int_0^\infty (\overline{x}_{22}^T Q_{22} \overline{x}_{22} + u_{22}^T R_{22} u_{22}) dt \tag{4.5d}$$

The matrices Q, R and K applied in the mission are shown in the Annex B (equations B.2 and B.3). As it can be observed in matrices Q, the errors are weighed

more than the rest of the state variables in each case, in order that the cost function can focus on reducing the errors in the reference. In the case of matrices R, their weigh diagonal parameters are comparable to the ones of the error variables in Q matrices, which is made in order to decrease the utilisation of the control actions, since it was excessive in some cases. In some combinations of matrices Q and R, the weigh parameter of the error is lower than the control action one, which could lead to a fairly low control actions; however, it must be taken into account that the errors and the control actions sometimes have distinct range of values: for instance, the error in altitude is in meters, while the elevator deflection is in radians, therefore the weigh parameter of the control action could be higher.

4.2.2 MPC

In the MPC controller case, the configuration is much more simpler than the LQR one, since Simulink provides a block with the whole structure already designed. In the image 4.12 the scheme used is shown, which is very similar to the LQR's one. The inputs of the MPC controller are: the measured outputs (MO) as the 11 state variables; the measured disturbances (MD) as the wind model and the wave drag coefficient, and the reference, in which the 11 variables are introduced, although only altitude, velocity and roll and yaw angles are really inputs, the error in the rest of the variables will be null[4]. The output will be the group of four control actions with respect to the equilibrium point, called as manipulated variables (MV); though, in this case, the control limits block is unnecessary since the MPC controller does restrict the control actions in terms of amplitude and time variation rate.

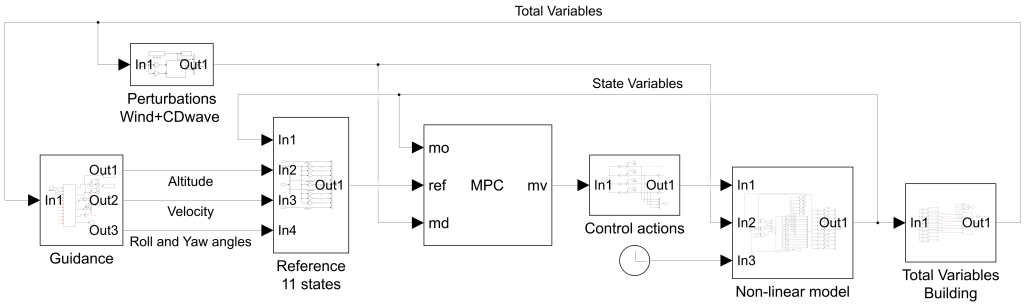

Figure 4.12: General scheme with MPC controller.

The scheme is very similar to the one used in LQR configuration. However, the perturbations and references blocks had not been introduced before, since it was not necessary due to the fact that there were only four references and the wave drag coefficient did not have to be introduced in the controller. In addition, a slight modification has been made in the non-linear block: the input of perturbations is only selected in their first three values, because the wave drag coefficient is already calculated in the Matlab function of the block.

In relation to the perturbations block, the inputs will be altitude and velocity in order to calculated the Mach number and obtain the wave drag coefficient. Here

[4]The other 7 state variables are introduced again in the MO input, so that the controller only focus on the required tracking.

the wind model configuration from the LQR general scheme can be observed as well. The figure 4.13 describe the configuration of the block.

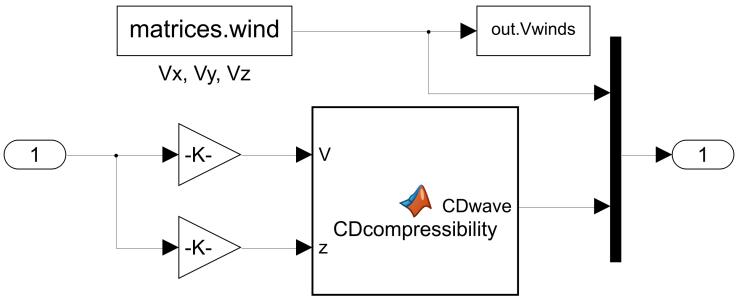

Figure 4.13: Perturbations block scheme.

Attending to the References block, the structure is fairly simple, since it is only utilised to the combination of the signals in order to set the reference in the 11 state variables. In the image 4.14, its scheme is shown.

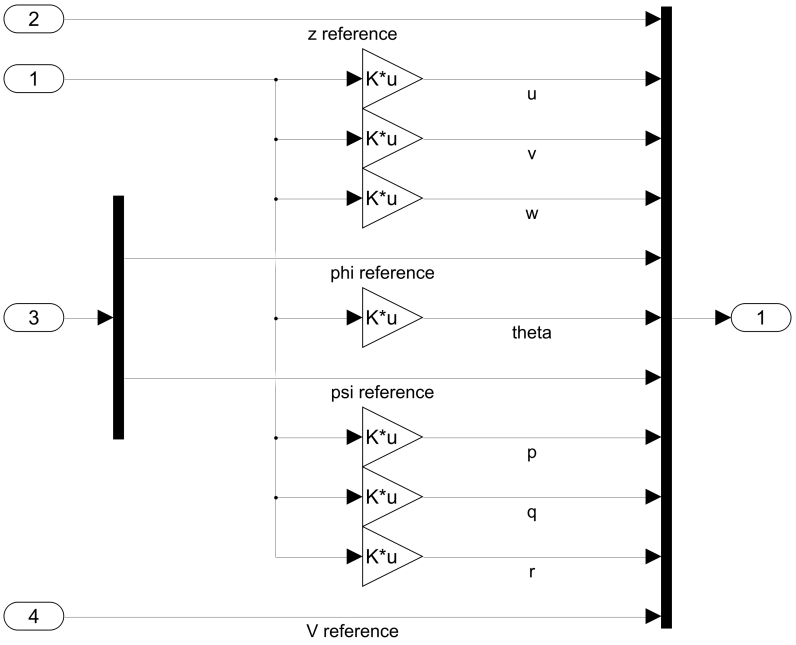

Figure 4.14: References block scheme.

Lastly, regarding the prediction and control horizon parameters, they have been set to 100 and 10, respectively. Therefore, at each iteration, the controller will calculate the performance of the system a total of time $t_p = 100 \cdot T_s = 2.1s$, by means of applying control actions until the time $t_c = 10 \cdot T_s = 0.21s$, selecting the best options among all the calculations done. These parameters have been chosen by taking into account the computational effort made and the capabilities of the controller to predict the future response.

In relation to the weigh parameters, which are shown in equations B.3 (see Annex B), the motion in the vertical plane has been restricted as far as possible, since low weigh values produced an excessively aggressive response of the aircraft due to acute elevator deflections. Hence, the angular velocity q and elevator deflection

magnitude and time variation rate, are weighed much more than other variables. The parameter ECR defines the weigh that will have the errors in reference tracking of the variables which are required to follow; as a consequence, it will have the greatest value.

4.3 Simulation in FlightGear

When the results from the simulation in Simulink are obtained (see Chapter 7), the parameters of position and attitude are employed to simulate the whole flight in FlightGear. The scheme that has been used can be seen in the figure 4.15, in which the position in x and y coordinates is translated into longitude and latitude magnitudes by means of the "Flat Earth to LLA" block. This block uses a reference longitude and latitude, which will be the ones according to the $(x, y) = (0, 0)$ coordinates, and from them it calculates the next ones.

Finally, the simulation block of FlightGear (version 2018.3) receives the six degrees of freedom in terms of position and attitude of the aircraft and develops the simulation. It is important to mention that the simulation is performed in real time, despite the sample time value.

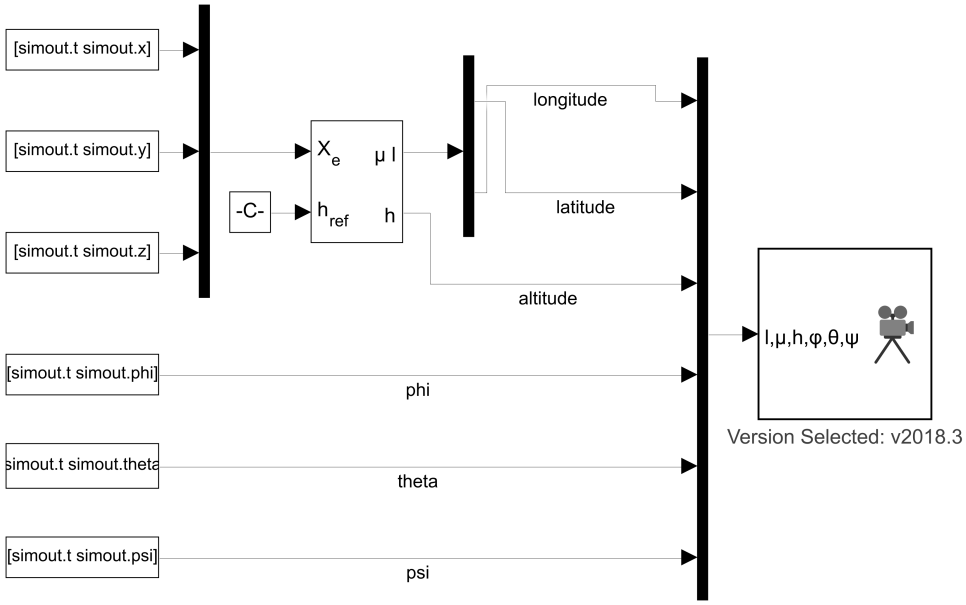

Figure 4.15: Scheme used in Simulink to the simulation in FlightGear.

5 | Guidance

In the following section, some guidance laws are going to be exposed which can make possible that the aircraft can follow successfully the reference path imposed. There are two main groups of guidance laws commonly used in autonomous aeroplanes: proportional and based on predictive control. The first one offers a simple solution to the problem, although an adjustment of the algorithm parameters has to be made at each configuration in order to get an adequate tracking of the reference path; on the other hand, the second guidance type provides optimal results, but also it is much more complex than the first one.

All the fundamentals which have been utilised to design the guidance system of the aircraft are based on [10] and [32].

5.1 Guidance law possibilities

In many cases, the concept of guidance is applied to the problem of generation and control of trajectories as a whole, nevertheless, in this project, a guidance system is understood as a tool to obtain a velocity vector (transmitted to the attitude control system) which ensures the aircraft follows properly the reference.

The block diagram with inputs and outputs that the guidance system employed in this project is shown in the figure 5.1. The system will give the proper orders to the control block about what the desired flight parameters are, on the basis of the reference dynamic path and the actual parameters of the aircraft (position, attitude and velocity).

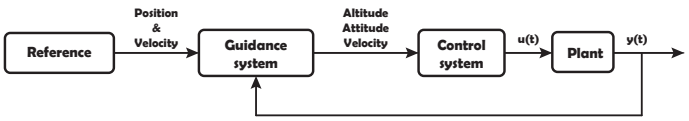

Figure 5.1: Conceptual guidance system scheme of the aircraft.

Most of the guidance bibliography is focused on the control of missiles, which principally attempts to design algorithms of chasing of a target. These methods can be applied to an autonomous aircraft in order to pursue an imaginary objective that follows the reference trajectory at the speed imposed by the mission specifications.

Among the first works in relation to missile guidance, two principal guidance laws can be found: pure chasing ones and proportional ones. Both laws calculate

the position vector between the moving target (the following point of reference trajectory) and the missile, in such a way that the needed accelerations which the missile must conduct are obtained from that position vector.

Pure chasing laws (also known as carrot chasing, see [32]) are based on creating a velocity vector that exactly points to the target, giving as a result excessive accelerations that could be a hindrance in the reference following. On the other hand, proportional guidance laws were born as an evolution of pure chasing laws, modifying the direction of the velocity vector in accordance with the rotation of the line which goes from the aircraft to the target, obtaining proportional accelerations depending on the change in the view line. Therefore, an acceleration $\overrightarrow{a} = -N\frac{\mathrm{d}r}{\mathrm{d}t}\frac{\mathrm{d}(\overrightarrow{r}/r)}{\mathrm{d}t}$ is applied, where $\overrightarrow{r}$ is the relative position vector and N a constant of proportionality. The image 5.2 shows the scheme of the classical proportional guidance law.

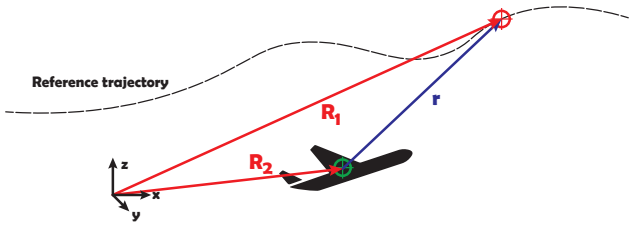

Figure 5.2: Proportional guidance law scheme: target position vector (R_1, red), aircraft position vector (R_2, red) and relative position vector (r, blue).

The disadvantages of this method are the generation of an acceleration instead of a velocity, the lack of calculation of the future position of the target and the inability of introducing constraints and taking into account perturbations. However, the accelerations are lower than pure chasing ones and the implementation is relatively easy.

In addition, nowadays more complex guidance laws based on the modern control fundamentals are being used in missiles (but also applicable to aeroplanes). For instance, laws based on optimal control algorithms such as LQR or techniques as the Sliding Mode Control (SMC) are used, resulting in a improved control and the possibility of taking into account the perturbations in some cases. But, one of the most important methods and applicable to aircraft is the predictive control (MPC), which as it is explained in the Chapter 4, it is valid for all systems, such as linear and non-linear or slow or fast processes. [10]

Predictive control laws have the advantage of anticipating the future, optimising the response of the system and minimising the cost of the operation, in addition to incorporate possible constraints to the state variables and control ones. Nevertheless, the more complex the system is, the more difficult applying this method is, so that the proportional guidance laws are selected for this project due to the mathematical complexity of the aircraft's model.

5.2 Proportional guidance

The guidance law employed in this work takes the concepts of the proportional guidance based on forces (see [10]) and the non-linear guidance algorithm (extracted from [32]).

5.2.1 Classical proportional guidance law

As it has been explained before, proportional guidance attempts to induce an acceleration on the aircraft to incorporate it to the reference trajectory. In this case, the acceleration will be only in the horizontal plane, seeing that the control is conducted on the altitude state variable and not over the flight path angle.

This force will be applied perpendicular to the velocity vector of the aircraft, following the expression of the equation 5.1. The constant N is the proportional gain, V the aircraft speed, r the relative position vector and η the angle between the velocity vector and the relative position vector. The figure 5.3 shows this idea over the image 5.2.

$$a_{horizontal} = N\frac{V^2}{r}\sin(\eta) \tag{5.1}$$

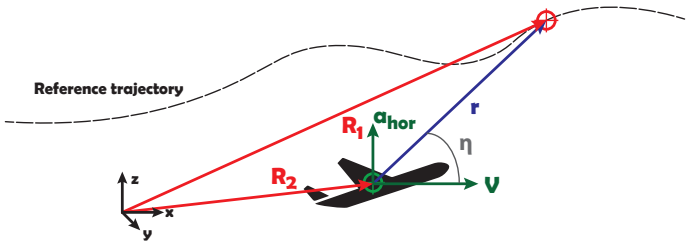

Figure 5.3: Proportional guidance (horizontal plane movement): horizontal acceleration (a_{hor}, green), velocity vector (V, green), angle between relative position and velocity vectors (η, grey).

The constant N must be adjusted to get greater or lower accelerations depending on the case, because an excessive value can produce an unstable flight with many oscillations while a low one generates a poor adaptive guidance to changes in the reference. The length of r has to be modified too, since low values produce great sensitivity when the airplanes faces variations in the reference and great values give as a result high actions in advance.

The attitude control can not accept accelerations, consequently, the horizontal force is translated into a desired course angle (Ψ), from which is obtained the required yaw angle (ψ) that the aircraft must achieve to ensure it. At each sampling time (T_s) a command of course angle change ($\Delta\Psi$) is given, creating an approximated result of the one according with the acceleration effect. The expression of this angle change is the following one:

$$\Delta\Psi \approx \frac{a_{horizontal}}{V} T_s = N\frac{V}{r}\sin(\eta)T_s \tag{5.2}$$

The relation between the course and yaw angle can be obtained by means of the rotation matrices and the body velocities, utilising the following expressions:

$${}^{\Psi}R_B = M_{\phi}^{-1}M_{\theta}^{-1} \rightarrow \begin{pmatrix} u \\ v \\ w \end{pmatrix}_{\Psi} = {}^{\Psi}R_B \begin{pmatrix} u \\ v \\ w \end{pmatrix}_{B} \Rightarrow \psi = \Psi - \arctan\left(\frac{v_{\Psi}}{u_{\Psi}}\right) \tag{5.3}$$

In the case of the velocity, the value of velocity change (ΔV) is very simple, the difference between reference speed and the actual one: $\Delta V = V_{ref} - V$.

This guidance law is easy to implement, although presents some deficiencies: the gains and the distance to the target are static, so that it would be necessary to include some kind of dynamic variation, in order to take into account the different flight conditions and reduce the instability of the system due to excessive oscillations.

5.2.2 Proportional guidance law based on forces

In order to solve the problem of the classical guidance, a modification of the law in which the accelerations are estimated from the forces is accomplished. In this way, the equations of flight mechanics are used to find an optimal solution to the problem, giving command orders to the control block about what roll and yaw angle the aircraft must achieve to follow the reference path.

In this case, circular trajectories from the aircraft to the target are preferred, so that arcs of radius R are created instead of straight lines, as it can be seen in the figure 5.4. It is important to notice that the triangle composed of the aircraft, target and center of the arc defines an isosceles triangle, where the angle between both radius R is 2η.

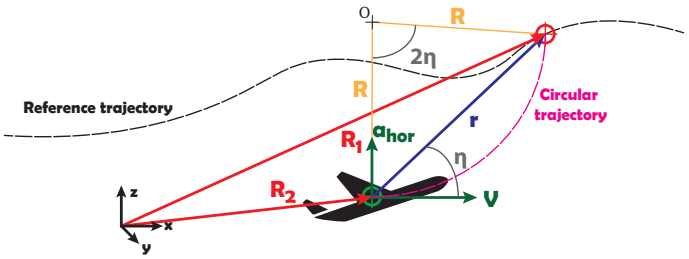

Figure 5.4: Proportional guidance (horizontal plane movement): radius of curvature (R, orange), desired circular trajectory (magenta, discontinuous).

Therefore, some assumptions are conducted so as to simplify the problem: the velocity and bank angle (μ) are assumed as constants and the flight path angle (γ) as null, as well as a symmetrical flight is supposed, giving as a result that the bank angle and roll angle (ϕ) are equal. These hypothesis, defined by the expressions 5.4 of an

uniform circular motion in the horizontal plane, are no longer true if the sideslip and flight path angles are very different from zero, but this is the best approximation to get a set of equations which allows to obtain a manageable solution of the reference roll angle.

$$T = D \tag{5.4a}$$

$$\frac{W}{g}\frac{V^2}{R} = L\sin(\eta) = L\sin(\phi) \tag{5.4b}$$

$$W = L\cos(\eta) = L\cos(\phi) \tag{5.4c}$$

Hence, the new horizontal acceleration will be $\frac{V^2}{R}$. These conditions are translated into expressions for the load factor and the roll angle of the aircraft, defined by the following equations:

$$W = L\cos(\phi) \Rightarrow n = \frac{1}{\cos(\phi)} \Rightarrow \phi = \arccos\left(\frac{1}{n}\right) \tag{5.5a}$$

$$\begin{gathered}\frac{W}{g}\frac{V^2}{R} = L\sin(\phi) \Rightarrow \frac{V^2}{gR} = \sqrt{n^2-1} \Rightarrow \\ \Rightarrow n = \sqrt{\left(\frac{V^2}{gR}\right)^2 + 1} \Rightarrow n = \sqrt{\left(\frac{2V^2\sin(\eta)}{gr}\right)^2 + 1}\end{gathered} \tag{5.5b}$$

In addition, the load factor must be between a boundaries in order to guarantee the stall avoidance and exceeding the structural limits; therefore, the load factor value must ensure that:

$$n \leq min(n_{max} = 5.5, \frac{\rho V^2 S_w C L_{max}}{2W}) \tag{5.6}$$

5.3 Guidance algorithm

On the basis of the equations from the previous section, the Algorithm 3 is implemented, which attempts to find the desired target point at each time step and defines the reference in velocity, altitude, roll and yaw angles.

Firstly, the actual flight parameters are defined, together with the aircraft parameters (S_w, m, CL_{max}) and the threshold, which is the length that specifies if the simulation stops when the distance between the aircraft and the final destination is less than this one. In addition, the dynamic path (*waypoints*, see Section 6.5) is given.

The length r is calculated from the velocity, taking greater values for high velocities and lower ones if the speed is minimum. Afterwards, the coordinates (x,y) of the intersection point between the circle, whose radius r and the center

is the aircraft position, and the trajectory is obtained[1]. Secondly, repeating the same procedure but without radius, the following point which is the closest to the aircraft position is selected, so that this one will specify the altitude and velocity of reference.

Afterwards, the change in the course angle ($\Delta\Psi$), the load factor (n) and the reference roll angle (ϕ) are calculated by means of the equations previously explained. In the same way, the reference yaw angle is obtained. Finally, the distance from the aircraft to the final destination is analysed, so that if it is lower than the threshold, the simulation ends.

Algorithm 3 Guidance Law

define $waypoints, T_s, x, y, z, V, \Psi, u, v, w, \phi, \theta, n_{max}, S_w, m, CL_{max}, threshold$
$r = assign\_radius(V)$
$[index_1] = next\_point(waypoints, r, x, y)$
$x_{ref} = waypoints(index_1, 1)$
$y_{ref} = waypoints(index_1, 2)$
$[index_2] = closest\_point(waypoints, x, y)$
$z_{ref} = waypoints(index_2, 3)$
$V_{ref} = waypoints(index_2, 4)$
$\overrightarrow{V}_0 = [V\cos(\Psi)\ V\sin(\Psi)]$
$\overrightarrow{r} = [x_{ref} - x\ y_{ref} - y]$
$[\eta] = angle(\overrightarrow{V}_0, \overrightarrow{r})$
if $\eta == 0$ **then**
    $\Delta\Psi = 0$
else
    $\Delta\Psi = sign(\eta)\left(2\frac{V}{r}T_s\right)$
end if
$\Psi = \Psi + \Delta\Psi$
$n = min\left(\left(\sqrt{\left(\frac{2V^2\sin(\eta)}{gr}\right)^2 + 1}\right), n_{max}, \left(\frac{\rho V^2 S_w CL_{max}}{2W}\right)\right)$
$\phi_{ref} = sign(\eta)\arccos(1/n)$
$\left[{}^{\Psi}R_B\right] = rotation\_matrix(\phi, \theta)$
$\overrightarrow{V} = [u; v; w]$
$\overrightarrow{V}_{\Psi} = {}^{\Psi}R_B \cdot \overrightarrow{V}$
$\psi = \Psi - \arctan\left(\frac{\overrightarrow{V}_{\Psi}(2,1)}{\overrightarrow{V}_{\Psi}(1,1)}\right)$
if $|[x, y, z] - waypoints(end, 1:3)| < threshold$ **then**
    $stop = 1$
else
    $stop = 0$
end if

[1]There are two intersections, so that the one with a higher index in *waypoints* is chosen.

6 | Dynamic path planning

In the following section, the dynamic path planning algorithm used in the mission of this project is going to be developed. After a brief introduction, the different spatial obstacles are going to be defined and the dynamic constraints applied to the aircraft, as well as the general procedure which should be followed so as to generate successfully the algorithm code. Then, on the basis of specific bibliography (see [29][27]),the Recursive Rewarding Adaptive Cell Decomposition (RR-ACD) algorithm is explained, exposing its advantages and disadvantages and showing the solution for the example case.

Once the RR-ACD algorithm is created, it is needed to apply radii of gyration at each waypoint of the generated route and discretize every segment into shorter ones in order that the aircraft can follow properly the reference trajectory, and in addition, define the velocity profile that the aeroplane has to make during the route. This is done by the Smooth 3D Path Planning algorithm, extracting the principal ideas from [28].

Finally, utilising the model of the M117 bomb, an estimation of which point of the final route should be the best one to throw the bombs to obtain the closest impact to the objective is made.

6.1 Introduction

Nowadays, 3D autonomous navigation algorithms are being fairly demanded in order to optimise the route and diminish the need of the human hand, resulting in a more efficient navigation. In this way, path planning tries to define the methodology which an autonomous aircraft should accomplish in order to move from an initial location to a final one, employing all its capabilities while avoiding obstacles during the trip.

There is a high diversity of options to complete the task of 3D path planning. For example, the algorithms are based on sampling fundamentals, node-based ones, bio-inspired ones and mathematical models, among other techniques.

In this case, an approach to node-based algorithm in a discrete space is implemented. Node-based algorithms are mathematical structures which are utilised to model pairwise relations[1] and the final purpose consist in analysing the cost of exploring nodes in order to find an optimal path. There is a wide variety of algorithms

[1]Building structures with vertices and edges.

that incorporate this methodology or subsequent variations, such as Dijkstra's algorithm, A^*, D^* and $Theta^*$. [27]

In relation to obstacles, they can be static or dynamic, for example, the terrain elevations would be a static one, while an enemy patrol would have to be considered as a dynamic one. In order to include dynamic obstacles, the constrained discrete space would have to be actualised at each time step that the aircraft is moving; nevertheless, for the aim of this project, only static ones have been considered, simplifying the problem.

Regarding the optimisation purposes, many planners attempt to optimise the path planning distance, however, in the algorithm described below, the objective is also minimise the rates of change in yaw and flight path angle to obtain a more achievable trajectory for the aircraft, sacrificing to the shortest path distance.

An Adaptive Cell Decomposition (ACD) is a strong method to solve physical systems led by partial differential equations, which is utilised in refined complex 3D Cartesian geometry reconstructions. Hence, an ACD is proposed for this project, in which the 3D environment is explored and decomposed using recursive rewarding cost functions, whose purpose is determining the parts of the total space that must be analysed and divided into different subspaces, giving as a result that now it is not necessary to reconstruct the whole 3D space and, consequently, saving a huge computational time.

6.2 Problem definition

6.2.1 3D space creation

The 3D Cartesian space is divided recursively into subspaces called as rectangloids, in such a way that every one of them can be split into other eight ones. It is really important that rectangloids do not collide among them and the division is conducted in an uniform way.

The whole space is defined by an initial point, q_i, and a final one, q_f, assigning also a top altitude and a bottom one (which will be the sea level). In the real trajectory of the aircraft, no Cartesian coordinates should be applied, since the Earth's curvature does not allow it, so that an ideal bench of work would be the spherical coordinates; however, the problem would become unmanageable, seeing that the space could not be divided into regular 3D objects, and a trustworthy model of the Earth should be utilised[2]. Thereby, the total space analysed by this 3D Cartesian algorithm must be small enough in order that it is not really affected by the Earth's curvature and the distances among real points have to be calculated by means of a great-circle function.

The great-circle distance, or orthodromic distance, is the shortest distance between two points on the surface of a sphere, measured along the surface of the

[2]The fact of considering the Earth as a perfect sphere would led to noticeable errors when long paths are being inspected.

sphere, in such a way that this distance belongs to a circumference whose center and radius are the same as the sphere's ones. In the figures 6.1 and 6.2 (extracted from [37, 31]), examples of an orthodromic route between points on a sphere and a Mercator's projection map are shown, respectively.

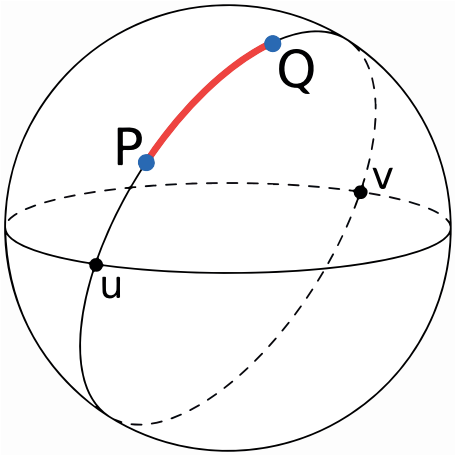

Figure 6.1: Great-circle distance between two points on a sphere.

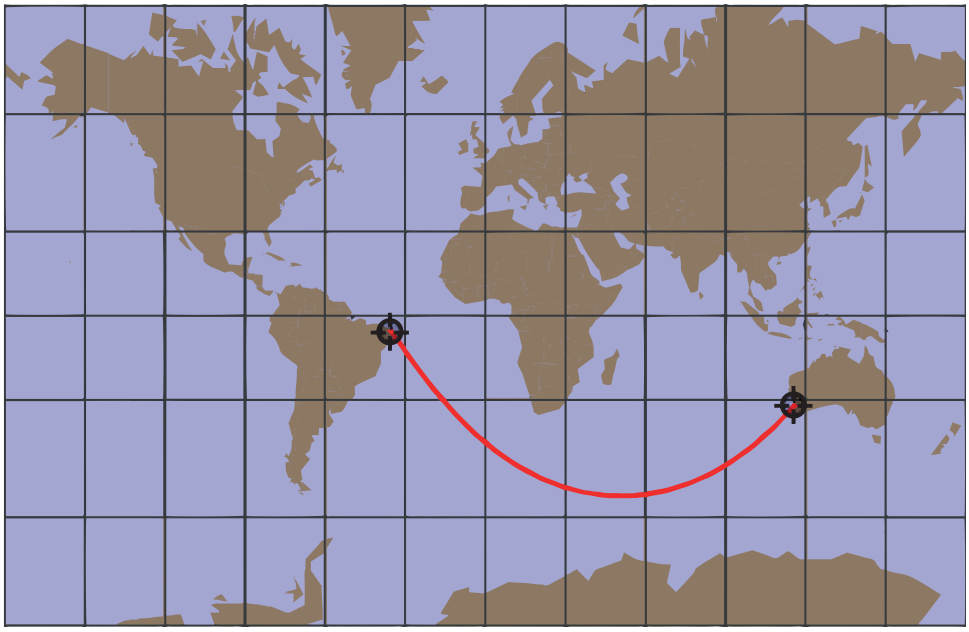

Figure 6.2: Great-circle route on a Mercator's projection map.

6.2.2 Static obstacles

In order to obtain a realistic mission in which the aircraft has to avoid crashing on the ground and do not be recognised by a radar or enter into specified areas while the bomb dropping is fulfilled successfully, different obstacles are defined:

1. Danger Area: zone in which the aircraft can under no circumstances enter inside of it. It is generated as a cylinder from sea level altitude to the top altitude and with the radius and center selected by the user.

2. Radar Area: zone similar to the danger area, although in this case the bottom altitude is not the sea level one, but the minimum altitude at which the airplane is detected by the radar. As before, a cylinder is created with radius and center specified by the user[3]. It is important to mention that this obstacle does not represent exactly the restriction of a radar, since radar range equations should be utilised, it is only a simple approximation.
3. Dropping Slope: since the algorithm could permit that the aircraft achieves the coordinates of the final point at a higher altitude than the required one, a reversed cone is created in order to avoid the selection of problematic points in the trajectory which can lead to an algorithm error. The vertex of this cone is the final point in the trajectory with an increment in its altitude.
4. Ground elevation: the mountains of the whole space are introduced as a restriction. The procedure used to obtain this data and create the three-dimensional object are explained next.

6.2.3 Mission example

Description of the mission

In this subsection, the mission example, from which the algorithm is tested and the obstacles are created, is defined. As it was explained in the second chapter of this project, the F-86 Sabre started to act as a fighter-bomber mainly by the end of the Korean War, period in which the Battle of White Horse (6^{th}–15^{th} October, 1952) took place (see [36]). In this case, since the bibliography about the military actions that happened during this battle, but also the whole Korean War, is very limited, a hypothetical mission that could have occurred is going to be created basing on the mentioned battle.

The Battle of White Horse happened in the region called as the Iron Triangle, which was formed by the North Korean city of Pyonggang at its top and the Gimhwa-eup and Cheorwon-eup at its base, as it is shown in the image 6.3. [36]

In this way, the mission will consist in avoiding this war zone and attacking Pyonggang, supposing that is defenceless while the successive confrontations are taking place in the Iron Triangle. A radar is located on the southwest from Pyonggang (see figure 6.4).

The mission starts at the Yeouido airport[4], which was the air base utilised by the USAF during the Korean War in Seoul (see [40]), at 1000 meters of altitude and ends at Pyonggang. The objective of the mission consists in bombing the center of Pyonggang from an minimum altitude of 200 meters while the danger and radar areas are properly respected. These areas and the initial and final points of the trajectory are represented in the figure 6.4.

[3]The radius will be the radar range and the center, the radar location.
[4]Which is currently in disuse.

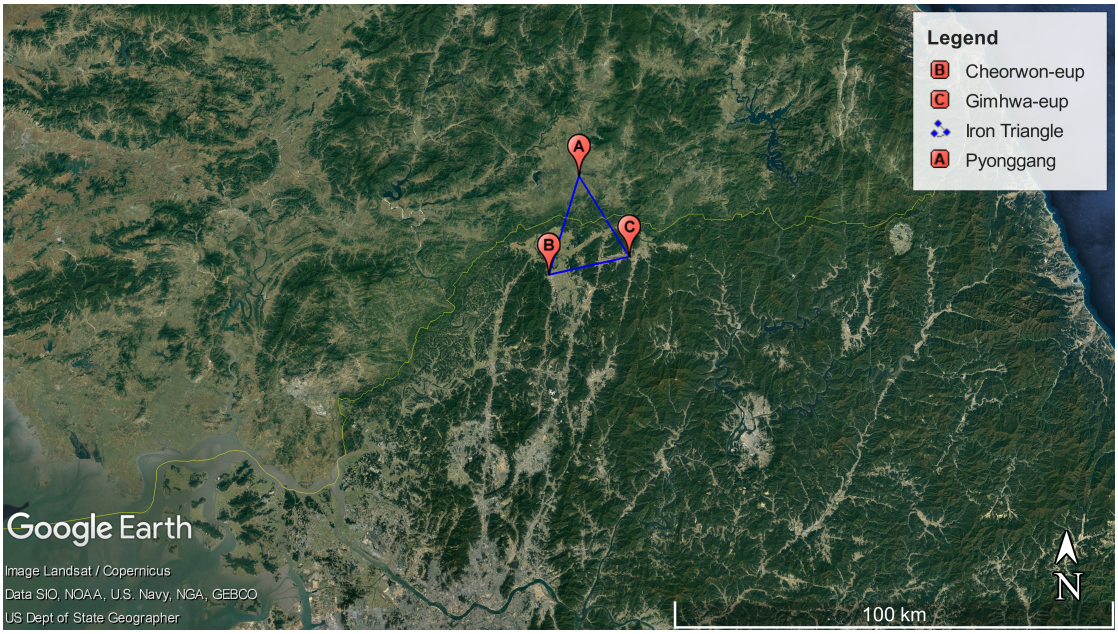

Figure 6.3: Iron triangle composed of Pyonggang, Cheorwon-eup and Gimhwa-eup.

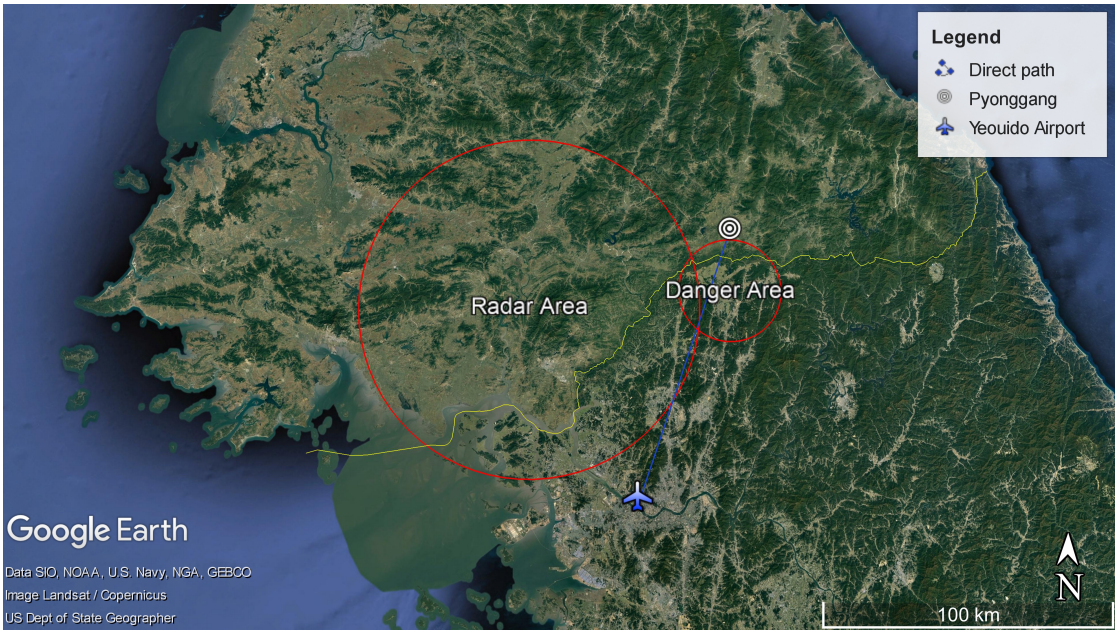

Figure 6.4: Initial and final points of the mission and restriction areas.

Obstacles during the mission

The obstacles are built in Matlab as collision objects: danger and radar areas are designed as collision cylinders and the dropping slope and ground elevation as a collision mesh. In the mission, the radar area is designed to include all points at an altitude higher than 700 meters and inside of the zone. In the case of cylinders, the creation is very simple, however, both collision meshes require a different approach.

Regarding the dropping restriction, a slope of 5 degrees has been selected, so that the aircraft's maximum dropping slope can not increase that one. A set of

points belonging to the circumference base of the cone is generated, creating the quarter of the reverse cone base that is inside of the 3D environment of analysis.

The danger and radar areas and the dropping slope collision objects of the mission are shown in the figure 6.5.

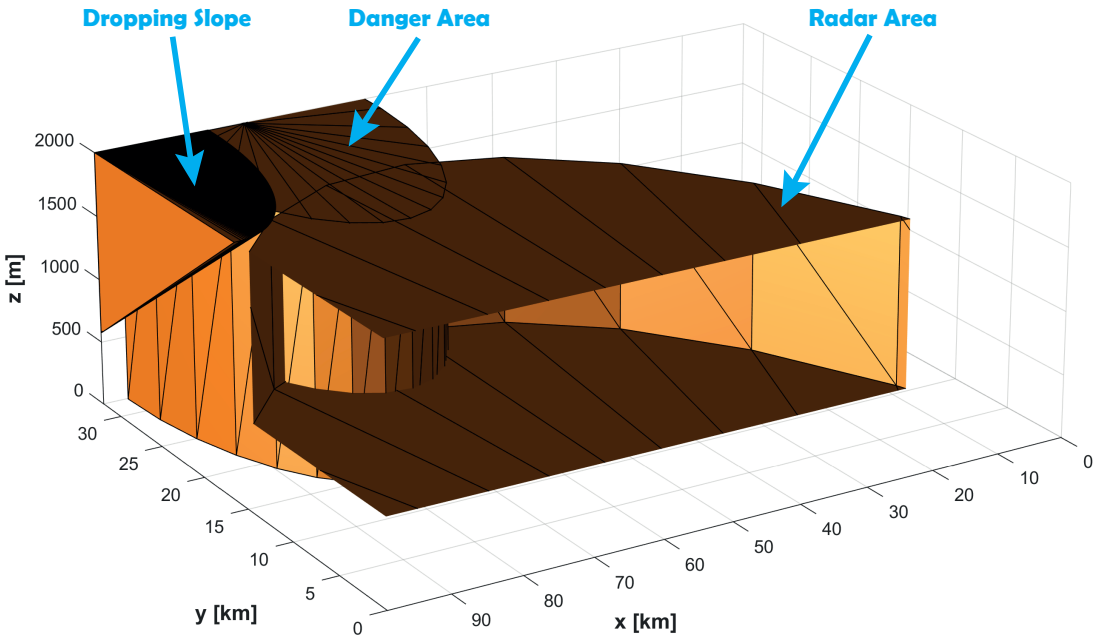

Figure 6.5: Obstacles of the mission example.

In relation to the ground elevation, the first step is obtain the elevation data of the terrain. In order to achieve such a goal, the XY plane is divided into a set of points separated among them by a precision distance, which in this case is 350 meters, afterwards, a route with those points is produced in Keyhole Markup Language (KML) and, then, converted to GPS eXchange Format (GPX) using a specific website (see [30]); once the GPX file is obtained, it is exported to a text file and after, to a Excel file, which is readable by MATLAB. The precision distance parameter is quite important, since the greater the precision, the more accurate the results are, but giving as a result that computational effort in the RR-MACD algorithm will be increased[5].

Finally, the elevation data is divided into groups of four coordinates in order to generate the total mesh piece by piece. The ground elevation of the environment can be observed in the figure 6.6.

[5]In many cases, the precision is limited by the maximum allowed size of the file uploaded to the website.

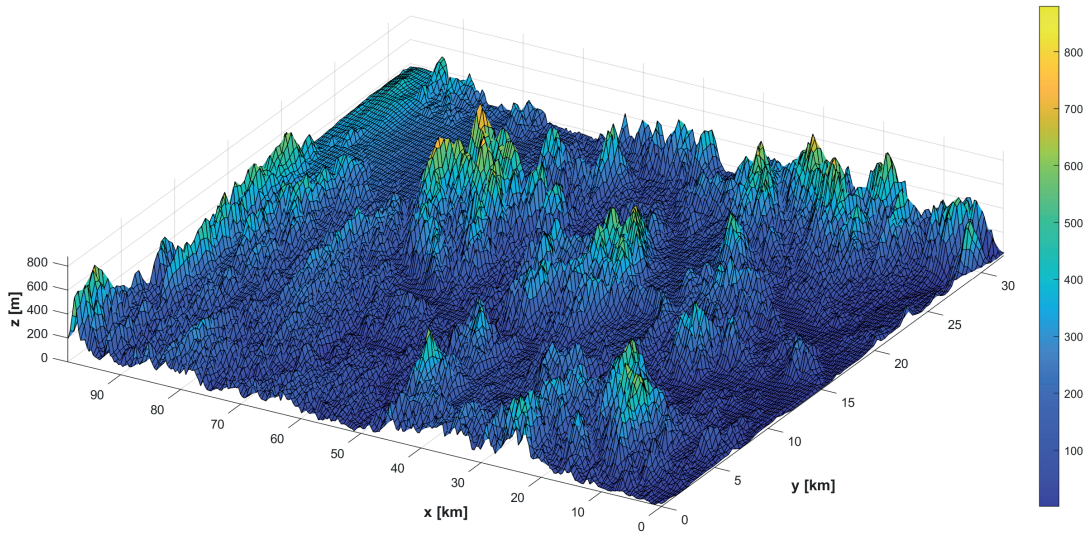

Figure 6.6: Ground elevation of the space exported to 3D Cartesian coordinates.

6.3 Adaptive Cell Decomposition

A typical 3D ACD algorithm usually tries to conduct a discretisation of the environment in a tree data structure known as Octree[6]. The use of a ACD algorithm implies that each little space in the environment is analysed, involving high computational effort; in this way, a modification of this algorithm should be made in order that only the necessary space is divided into other subspaces, nevertheless, the variety of future waypoints at each step is reduced and it is more complex to obtain a trajectory in accordance with the dynamic restrictions imposed when the decomposition level is high.

A modified ACD algorithm, as the one presented in [27], would attempt to verify the collision of every rectangloid at each decomposition level and create a new Octree only for those that collide with the obstacles. In the figure 6.7, it is shown at the top the result of a modified ACD at a decomposition level $i = 5$ of the whole 3D space of the mission, so that it can be observed that the rectangloids which represent the free space have different sizes among them depending on the level at which they are found (the ones closer to the obstacles present a lower size due to the higher level of discretisation). On the other hand, at the bottom of this figure 6.7, it can be appreciated the difference in the solution utilising the ACD algorithm of this project from the one employing the modified code, where all subspaces have the same size and belong to the same decomposition level.

[6]Data structure in which all internal nodes own eight children.

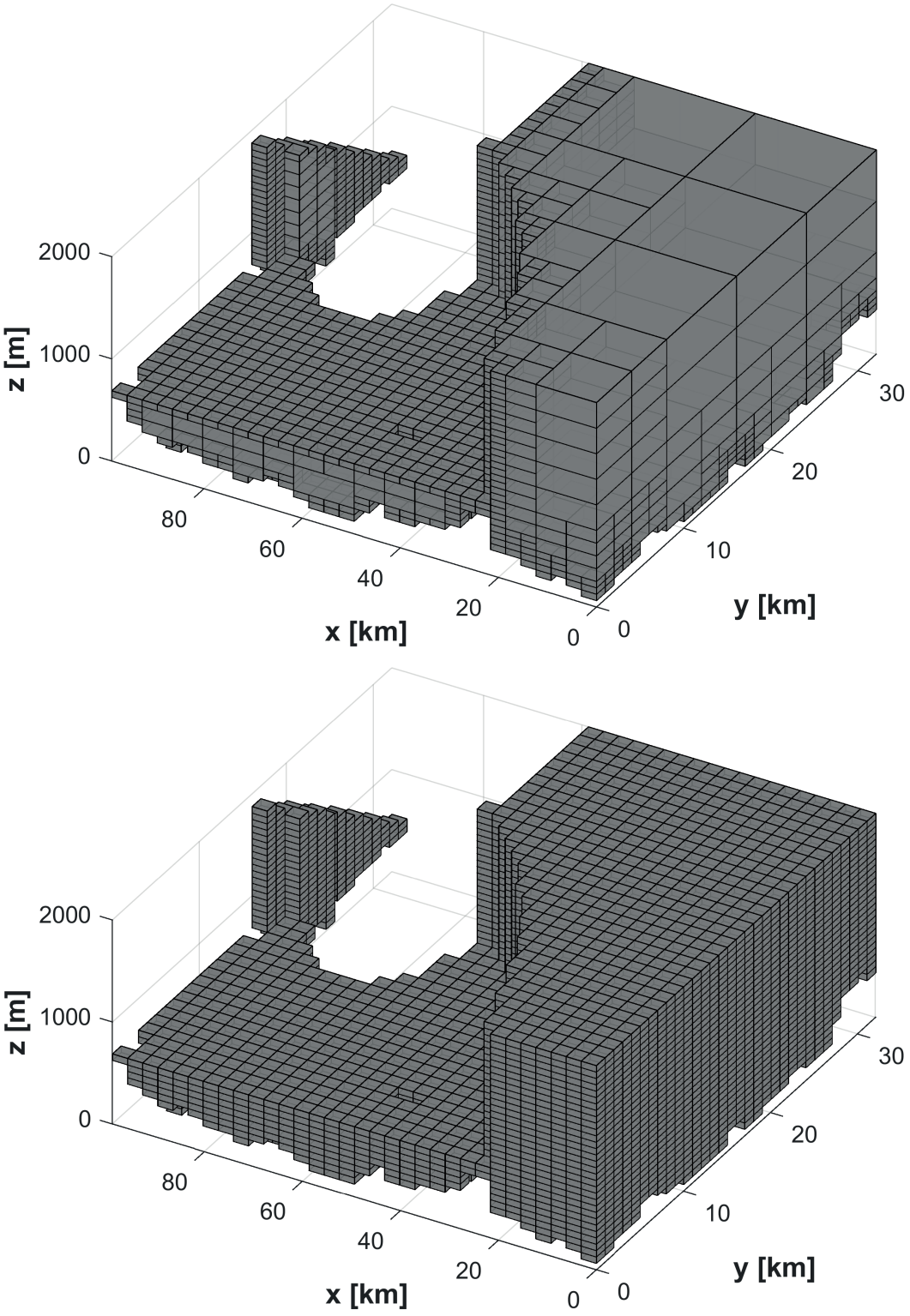

Figure 6.7: Total free 3D space calculated by a modified ACD algorithm (top) and ACD one (bottom) with decomposition level $i = 5$.

However, as it has been mentioned before, this modification has resulted in many errors in the final solution of the trajectory, therefore, it has been preferred to utilise a normal ACD algorithm[7] (see Algorithm 4).

[7]Although introducing a slight variation, consisting in not decomposing rectangloids which are

Algorithm 4 Adaptive Cell Decomposition (ACD)

```
1: n → decomposition_level
2: define Obstacles
3: [rectangloid_size, collision_rectangloids(1)] = initial_rectangloid
4: position_0 = position = 1
5: free_rectangloids = [ ]
6: for i = 1 : n do
7:    for selection = position_0 : position do
8:       if selection == position_0 then
9:          position_0 = position + 1;
10:      end if
11:      [−, complete_collision] = collision_box_objects(collision_rectangloids
         (selection), obstacles);
12:      if complete_collision == 0 then
13:         [new_vertices] = vertix_generation(rectangloids(selection));
14:         for j=1:8 do
15:            rectangloids(:, :, position + j) = new_vertices(j);
16:            [collision_rectangloids(position + j)] = collisionBox(i,rectangloids
               (:, :, position + j), rectangloid_size);
17:         end for
18:         position = position + 8;
19:      end if
20:   end for
21: end for
22: for selection = position_0 : Position do
23:   [collision, −] = collision_box_objects(collision_rectangloids(selection),
      obstacles);
24:   if collision==0 then
25:      free_rectangloids = [free_rectangloids, selection] ;
26:   end if
27: end for
28: output → planner
```

First of all, the precision of the solution is defined by the number of decomposition levels, although taking into account that a greater level involves a more accurate solution but also an increase in computation time. Secondly, the domain of the 3D space is introduced and the algorithm starts to decompose it into eight children and analysing their possible collisions with the obstacles.

proved to be completely inside of an obstacle.

At every respective decomposition level, each rectangloid is proved whether it is completely inside of an obstacle (complete collision) and if it is not, it is divided into eight different children of the same size and a collision box is created for each one of them. In this way, the boxes with complete collision are not successively split. In addition, rectangloids are always represented by means of their eight vertices (what will facilitate the neighbourhood finding in the RR-ACD algorithm), which are saved in a 3D matrix[8].

Once the set of rectangloids from the final decomposition level is created, the data is assigned to them. It is verified whether these subspaces collide or not with the obstacles and then, their own indexes are included in the free rectangloids vector: only those ones whose collision is not proved are in this list, so that, in this way, it is not necessary to give to each one the grade of occupied or free space.

Finally, a planning path function (such as Dijkstra's algorithm) is applied to define the optimal trajectory, what is going to be detailed in the next section.

In relation to the order in which subspaces are created, it is important to mention that are inspired on works such as [29], although focusing on obtaining the best reconstruction of the environment instead of getting an optimised solution. In the image 6.8, an example of a tree representation is shown with decomposition level $i = 2$ and many of their rectangloids producing a complete collision to illustrate the concept adequately. Black circles represent simple collisions, black squares show complete collisions and white circles are referred to free subspaces. In addition, each Octree division follows the same pattern; the subspaces are created utilising this sequence: up-northwest (UNW) - up-northeast (UNE) - up-southeast (USE) - up-southwest (USW) - down-northwest (DNW) - down-northeast (DNE) - down-southeast (DSE) - down-southwest (DSW).

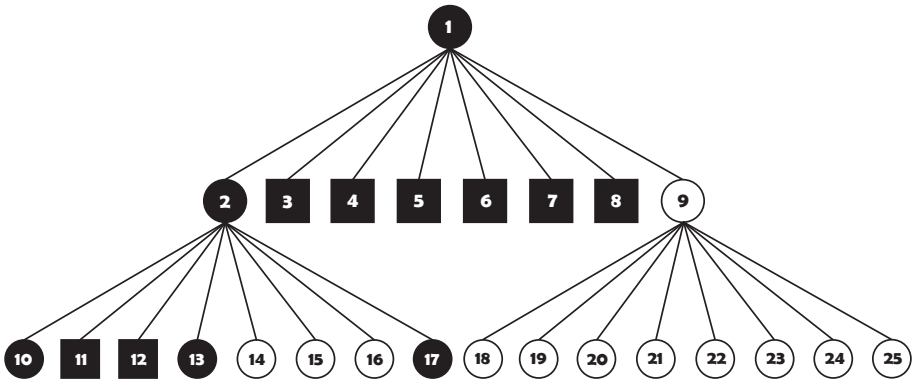

Figure 6.8: Tree representation of an hypothetical Octree block decomposition.

6.4 RR-ACD algorithm

In this section, a new strategy for path planning 3D spaces is presented, therefore it is not necessary to utilise node-based algorithms such as the ones mentioned in the Introduction section. The RR-ACD algorithm attempts to define a final path

[8]The vertices matrix presents a size of 8×3×n°rectangloids

ρ on the basis of the starting conditions, so that each subspace will be initially a collision-free neighbour rectangloid. Firstly, the total 3D environment is divided using the Octree method previously explained. The decomposition level utilised in this division will mark the future aspects of the final trajectory result, seeing that a low value provides poor accuracy measuring the obstacles, although a high one increases this accuracy but also the computational effort[9]. In this way, there are two different decomposition levels, an initial one of the total space and a local one at each neighbour, so that both values must be selected carefully depending on the mission and its obstacles. In this case, the starting level is $i = 4$ and the local one $i = 3$; in the figure 6.9, the initial division can be observed, showing a total of 8^4 subspaces.

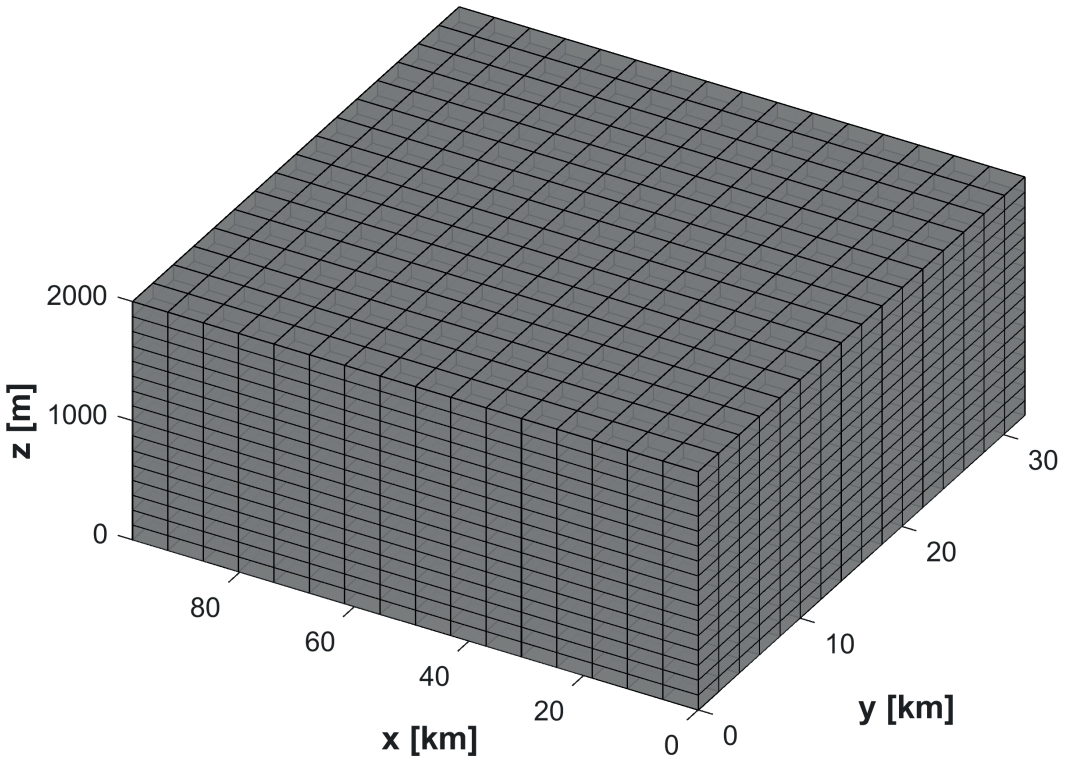

Figure 6.9: Initial decomposition of the total environment at level $i = 4$.

Afterwards, assuming that the initial point (q_i) is included in an initial free subspace (s_k), which is reasonable basing on the mission data (see figure 6.7), the aircraft must search for neighbour subspaces (S_{k+1}) from the initial decomposition and divide them into new ones (s_{k+1}) utilising the ACD algorithm, repeating successively the procedure until reaching the final point of the route (q_f). In order to find the best neighbour option (s_{k+1}), two cost measurements are conducted: D_1 and D_2, being the total cost the sum of both of them.

The first cost variable, $\mathcal{D}_1$ is obtained basing on analysing the path from s_k to s_{k+1} ($s_k \rightarrow s_{k+1} \Rightarrow \mathcal{D}_1$) and $\mathcal{D}_2$ from s_{k+1} to q_f ($s_{k+1} \rightarrow q_f \Rightarrow \mathcal{D}_2$). Each one is defined by a set of R_m partial functions, which studies the distance between points and yaw and flight path angles, in combination with a Gaussian function $g(R_m)$ and

[9]The number of total neighbours that the code must analyse is higher.

weigh parameters η_1 (for $\mathcal{D}_1$) and η_2 (for $\mathcal{D}_2$). The method used in this project is based on works as [27], although some important variations are accomplished.

The set R_m is a group of $m = 1...N$ partial functions that are involved in the 3D navigation characteristics and determine the viable options. In the algorithm, a total of $N = 3$ functions are defined. Furthermore, they are divided into types 1 and 2, in relation to sub-paths $s_k \rightarrow s_{k+1}$ and $s_{k+1} \rightarrow q_f$, respectively.

1. Type 1, sub-path $s_k \rightarrow s_{k+1}$: previous yaw and flight path angles are taken into account in order to measure the change in both.

   (a) Distance: given by the expressions below, where $M_{distance}(s_i \rightarrow s_j)$ is the Euclidean distance between the chosen subspaces s_i and s_j, and M_{direct}, the straight line between q_i and q_f.

$$R_{11}(i,j) = \frac{M_{distance}(s_i \rightarrow s_j)}{M_{direct}} \in \mathbb{R} : [0,1] \tag{6.1a}$$

$$M_{distance}(s_i \rightarrow s_j) = \sqrt{(s_{i_x} - s_{j_x})^2 + (s_{i_y} - s_{j_y})^2 + (s_{i_z} - s_{j_z})^2} \tag{6.1b}$$

   (b) Flight path angle: where γ_0 is the flight path angle of the previous branch of the path and γ the one of the vector given by $s_i \rightarrow s_j$. The R_{12} function must provide a value between -1 and 1, so that in cases in which $|\gamma - \gamma_0| > 45°$, the output in the code will be 1.

$$R_{12}(i,j) = \tan(\gamma - \gamma_0) \in \mathbb{R} : [0,1] \tag{6.2a}$$

$$\gamma = \arctan\left(\frac{s_{j_z} - s_{i_z}}{\sqrt{(s_{j_x} - s_{i_x})^2 + (s_{j_y} - s_{i_y})^2}}\right) \tag{6.2b}$$

   (c) Yaw angle: ψ_0 is the yaw angle of the previous branch of the path and ψ the one of the vector given by $s_i \rightarrow s_j$. Similarly as the previous case, when $|\psi - \psi 0| > 45°$, the output in the code will be 1.

$$R_{13}(i,j) = \tan(\psi - \psi_0) \in \mathbb{R} : [0,1] \tag{6.3a}$$

$$\psi = \arctan\left(\frac{s_{j_y} - s_{i_y}}{s_{j_x} - s_{i_x}}\right) \tag{6.3b}$$

2. Type 2, sub-path $s_{k+1} \rightarrow q_f$: only yaw and flight path angles of this sub-path are considered.

   (a) Distance: the same method than the one used in type 1 is employed here, although obviously utilising the points s_{k+1} and q_f.

$$R_{21}(i,j) = R_{11}(i,j) \tag{6.4}$$

(b) Flight path angle: now, the flight path angle of the previous branch is not employed in this type.

$$R_{22}(i,j) = \tan(\gamma) \in \mathbb{R} : [0,1] \tag{6.5a}$$

$$\gamma = \arctan\left(\frac{s_{j_z} - s_{i_z}}{\sqrt{(s_{j_x} - s_{i_x})^2 + (s_{j_y} - s_{i_y})^2}}\right) \tag{6.5b}$$

(c) Yaw angle: as before, the angle of the previous branch is not used.

$$R_{13}(i,j) = \tan(\psi) \in \mathbb{R} : [0,1] \tag{6.6a}$$

$$\psi = \arctan\left(\frac{s_{j_y} - s_{i_y}}{s_{j_x} - s_{i_x}}\right) \tag{6.6b}$$

These partial functions are separated into two types due to the fact that, only in the first sub-path, the change in yaw and flight path angles can be measured, so that it is needed two different functions for each case. In addition, a fourth partial function that measures the fuel mass consumption could be used, but the distinct values of consumption for each neighbour are really almost equal, so that no variation can be measured accurately.

Regarding the Gaussian function $g(R_m)$, utilised to analyse the reward in executing a possible action, it is defined by the expression 6.7, in which the transition cost values ($s_k \to s_{k+1}, s_{k+1} \to q_f$) have been normalised within the boundaries $[0, 1]$.

$$g(R_m) = \frac{\sin(\pi \cdot R_m + \pi/2) + 1}{2} \tag{6.7}$$

It is important to notice that the higher the effort R_m, the smaller the reward $g(R_m)$ and vice versa. Consequently, all state transitions from s_i to different neighbours s_j will have different costs, producing these costs lower rewards if they present elevated values.

All these rewards are collected in the vector $G(i,j)$, as it is seen in the expression 6.8.

$$G(i,j) = [g(R_1(i,j)), g(R_2(i,j)), ..., g(R_N(i,j))]^T \tag{6.8}$$

Finally, $G(i,j)$, the priority parameters (η_1, η_2) and a new parameter ξ are employed to build the a final reward function $\mathcal{D}$. $\mathcal{D}$, which is composed of $\mathcal{D}_1$ and $\mathcal{D}_2$, is the total reward associated with weigh parameters for conducting an action on each Gaussian function $g(R_m)$, defined by the expressions 6.9. The neighbour with the greatest reward will be selected as the next waypoint on the route.

$$\mathcal{D}_1 = \eta_1 \cdot G(i,j) + \xi_j, \quad \{i = 1, j = 2...(M-1)\} \tag{6.9a}$$

$$\mathcal{D}_2 = \eta_2 \cdot G(i,j), \quad \{i = 2...(M-1), j = M\} \tag{6.9b}$$

$$\mathcal{D} = \mathcal{D}_1 + \mathcal{D}_2 \tag{6.9c}$$

Therefore, in order to compute the reward of selecting a neighbour s_j, the formula would be as follows:

$$\eta_1 = (\eta_{1D}, \eta_{1\gamma}, \eta_{1\psi}) \tag{6.10a}$$

$$\eta_2 = (\eta_{2D}, \eta_{2\gamma}, \eta_{2\psi}) \tag{6.10b}$$

$$\mathcal{D}_1 = \{\eta_{1\mathcal{D}} \cdot g(R_{11}(s_1, s_j)) + \eta_{1\gamma} \cdot g(R_{12}(s_1, s_j)) + \eta_{1\psi} \cdot g(R_{13}(s_1, s_j))\} + \xi_j \tag{6.10c}$$

$$\mathcal{D}_2 = \{\eta_{2\mathcal{D}} \cdot g(R_{21}(s_j, q_f)) + \eta_{2\gamma} \cdot g(R_{22}(s_j, q_f)) + \eta_{2\psi} \cdot g(R_{23}(s_j, q_f))\} \tag{6.10d}$$

$$\mathcal{D} = \mathcal{D}_1 + \mathcal{D}_2 \tag{6.10e}$$

The parameter ξ is a variable reward value that is assigned to those neighbours s_{k+1} which are considered as very problematic or extremely necessary given the aircraft position and the obstacle locations. For instance, before entering into the radar area, the aircraft must decrease its altitude significantly so as to cross it successfully, so that only those neighbours which are located in a lower position at each step will be scored with a positive ξ. Other case is referred to the zone previously located to the danger area: when a danger area is touching a boundary border, the aircraft can not go to the area in which both are together, since the aeroplane could not escape from that location, therefore, that region is scored with a negative ξ.

Regarding the neighbourhood choice, only those blocks subspaces s_{k+1} located on the top, bottom, sides and in front of the subspace s_k will be evaluated as neighbours. In addition, if a subspace is a possible neighbour, but includes one of the waypoints of the path that is being generated (the aircraft has already gone to this subspace), this is automatically eliminated from the list of destinations. In the figure 6.10, an example of neighbourhood selection is shown, in which all five possible subspaces are available.

Once the list of available rectangloids from the initial decomposition is detailed, the ACD algorithm is applied to each one of them and the offered new subspaces (sub-rectangloids) are obtained. Those subspaces, represented each one by a group of 8 vertices, are converted into a different data in order to give to the aircraft the proper information of the next possible destinations: the set of vertices are translated into the centers of each sub-rectangloid, creating the list of feasible waypoints as a consequence.

After creating such list, a filtering process is conducted so as to reject all possible waypoints that could generate a very complex trajectory for the aircraft to follow. In this regard, two different constraints are applied to yaw and flight path angles: firstly, all waypoints possibilities that generate a new branch of the path which flight path angle is higher than 5°in absolute value will be removed from the list[10]; and secondly, every available waypoint whose change in yaw angle is greater

[10]Therefore, the maximum flight path angle selected for the trajectory is ± 5°, so that any part

than 90°in absolute value will be disregarded. Furthermore, at each step a fully-forbidden area is generated from the position of the initial waypoint s_k, which can be used under no circumstances (even if the change in yaw angle exceeds $\pm$ 90°), to avoid that the aircraft does not start a trajectory which could return to the initial point q_i.

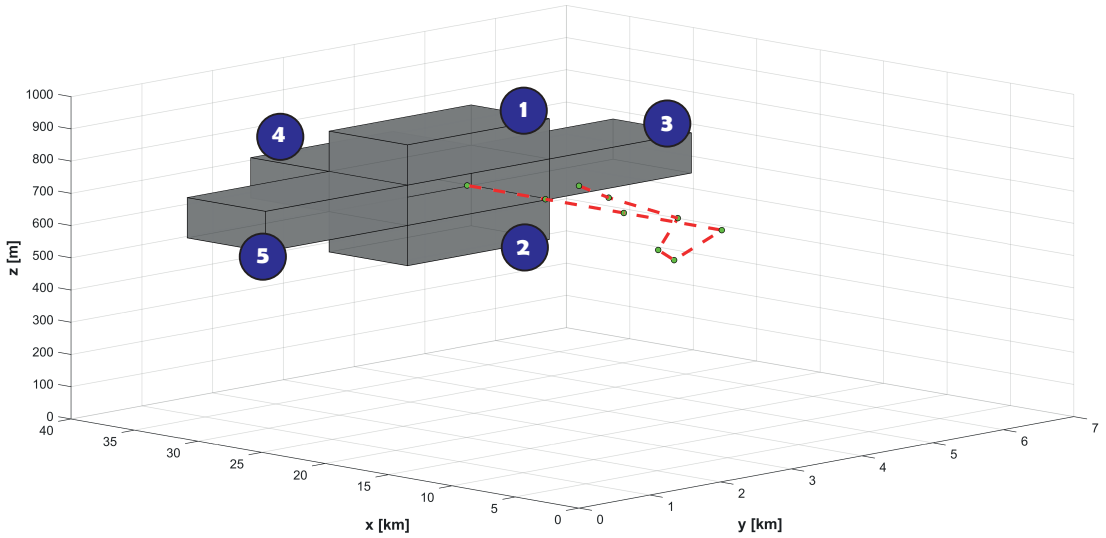

Figure 6.10: Example of possible neighbour subspaces at a point in the reference path (red, discontinuous).

In the images 6.11 and 6.12, examples of allowed area of centers selection and available waypoints list can be observed, respectively.

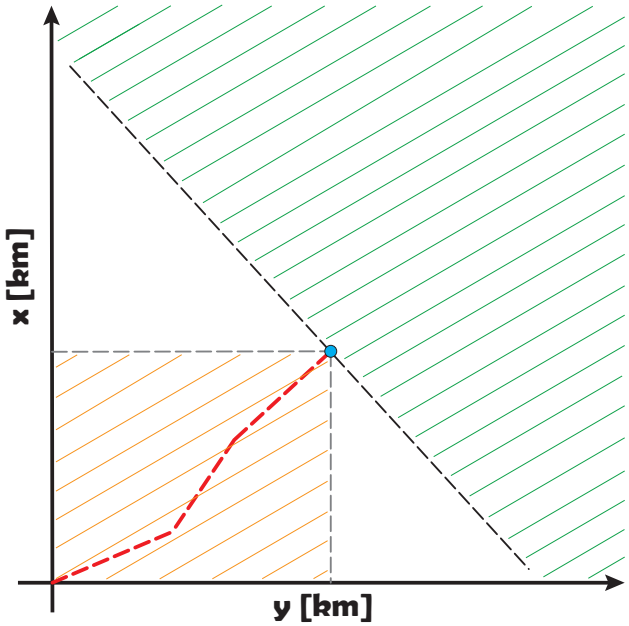

Figure 6.11: Example of available area (green) in the XY plane of a reference path (red, discontinuous) and fully-forbidden zone (orange).

of the path can exceed that restriction.

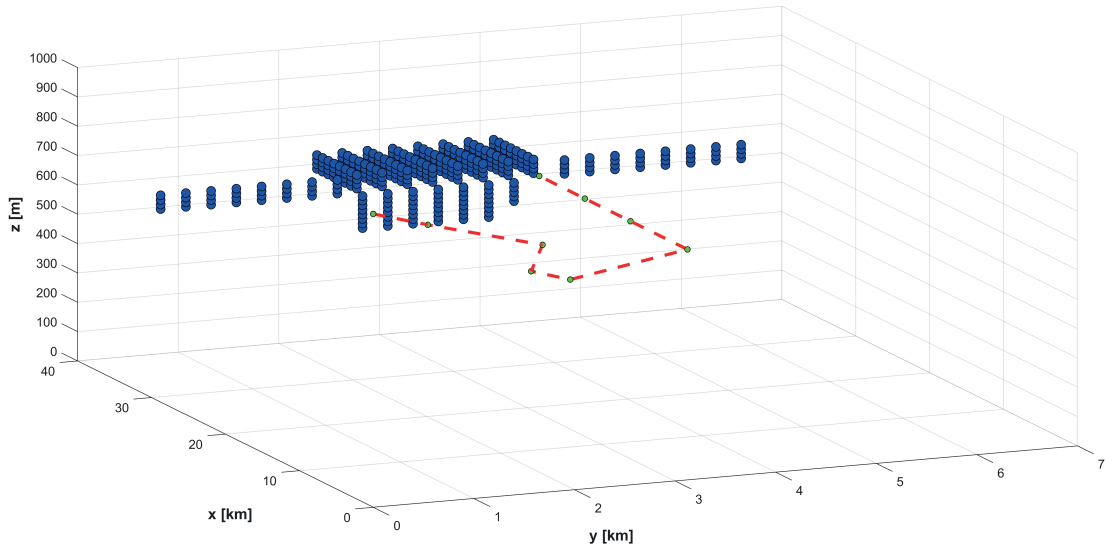

Figure 6.12: Available future waypoints (blue) to the path (red, discontinuous) from example of figure 6.10

Summarising, in first place, the whole RR-ACD algorithm (see algorithm 5) assigns initial and local decomposition levels and create the initial group of subspaces. Then, the parameters of path (wp), distance ($s_k \rightarrow q_f$), first subspace (s_k) and yaw angle (ψ) are initialised[11].

Afterwards, a while loop starts to iterate until the distance ($s_k \rightarrow q_f$) is lower than a predefined parameter d_{min}, moment at which the next waypoint is automatically the final point (q_f). The path is built in this way since the dropping restriction is blocking the final point (the vertex of the reversed cone is q_f), hence, when the aircraft is very closed to the final destination, the loop is finished and the airplane goes directly to q_f.

At each iteration, the algorithm finds the neighbours of the actual subspace (rejecting those which owns any waypoint of the path), recalculates the distance ($s_k \rightarrow q_f$) and applies the ACD algorithm at each neighbour to select the available sub-rectangloids. Once the new subspaces are created, the function *center_selection* obtains the centers of all sub-rectangloids from their vertices. Another while loop is included in this process, seeing that it is possible that no sub-rectangloids are available, in order to implement the decomposition of neighbours but at a higher level and, consequently, the code can find new subspaces. Finally, the costs of selecting each one of the possible destinations are calculated, choosing the reward that is greater and adding it to the path vector.

A final result is obtained for the mission path problem, whose trajectory and obstacles can be observed in the figure 6.13. The weigh parameters utilised are the following ones:

$$\eta_1 = (10000, 10, 10), \quad \eta_2 = (5000, 50, 500) \tag{6.11}$$

[11]The flight path angle (γ) is not needed to be created, since the restriction of $\pm$ 5°is defined by the function *center_selection*

Algorithm 5 Recursive Rewarding Adaptive Cell Decomposition (RR-ACD)

define $q_i, q_f, obstacles, \eta_1, \eta_2, d_{min}, \xi_{areas}$
$n_1 \rightarrow$ *initial decomposition level*
$n_2 \rightarrow$ *local decomposition level*
$[rectangloids, free\_rectangloids] = ACD(initial\_rectangloid, n_1)$
$wp = q_i$
$M_{distance} = Euler\_distance(q_i, q_f)$
$[s_k] = [s_{k_i}] = cube\_finding(q_i, rectangloids, free\_rectangloids)$
$[\psi] = initial\_angle(q_i, q_f)$
while $M_{distance} > d_{min}$ **do**
  if $s_k == s_{k_i}$ **then**
    $free\_rectangloids(find(freerectangloids == s_k)) = [\,]$;
  else
    for $j = 1 : length(s_{k+1})$ **do**
      $free\_rectangloids(find(freerectangloids == s_{k+1}(j))) = [\,]$;
    end for
  end if
  $s_{k+1} = neighbourhood(s_k, rectangloids, free\_rectangloids)$;
  $M_{distance} = Euler\_distance(q_i, q_f)$;
  $local\_divisions = n_2$;
  $[centers] = []$;
  while *length(centers)==0* **do**
    for $i = s_{k+1}$ **do**
      $[sub\_rectangloids, free\_sub\_rectangloids] = ACD(rectangloids(:, :, s_{k+1}), local\_divisions)$;
      $[centers] = center\_selection(sub\_rectangloids, free\_sub\_rectangloids)$;
    end for
    $local\_divisions = local\_divisions + 1$;
  end while
  $[center\_chosen, s_k, \psi] = rewards(wp(end, :), centers, q_f, \xi_{areas}, \psi(end))$;
  $wp = [wp; center\_chosen]$;
end while
$wp = [wp; q_f]$
return $Final\_path\_waypoints \rightarrow wp$

The final reference trajectory offers an acceptable set of conditions for the aircraft to follow it and avoids perfectly all obstacles, nevertheless, it is not the optimal result among all possibilities, so that a different selection of the weigh parameters or ξ values should be conducted so as to improve the result. Therefore, this is the main disadvantage of this method: at each mission configuration, the

parameters must be modified in order to get a valid path, taking many attempts before achieving this goal.

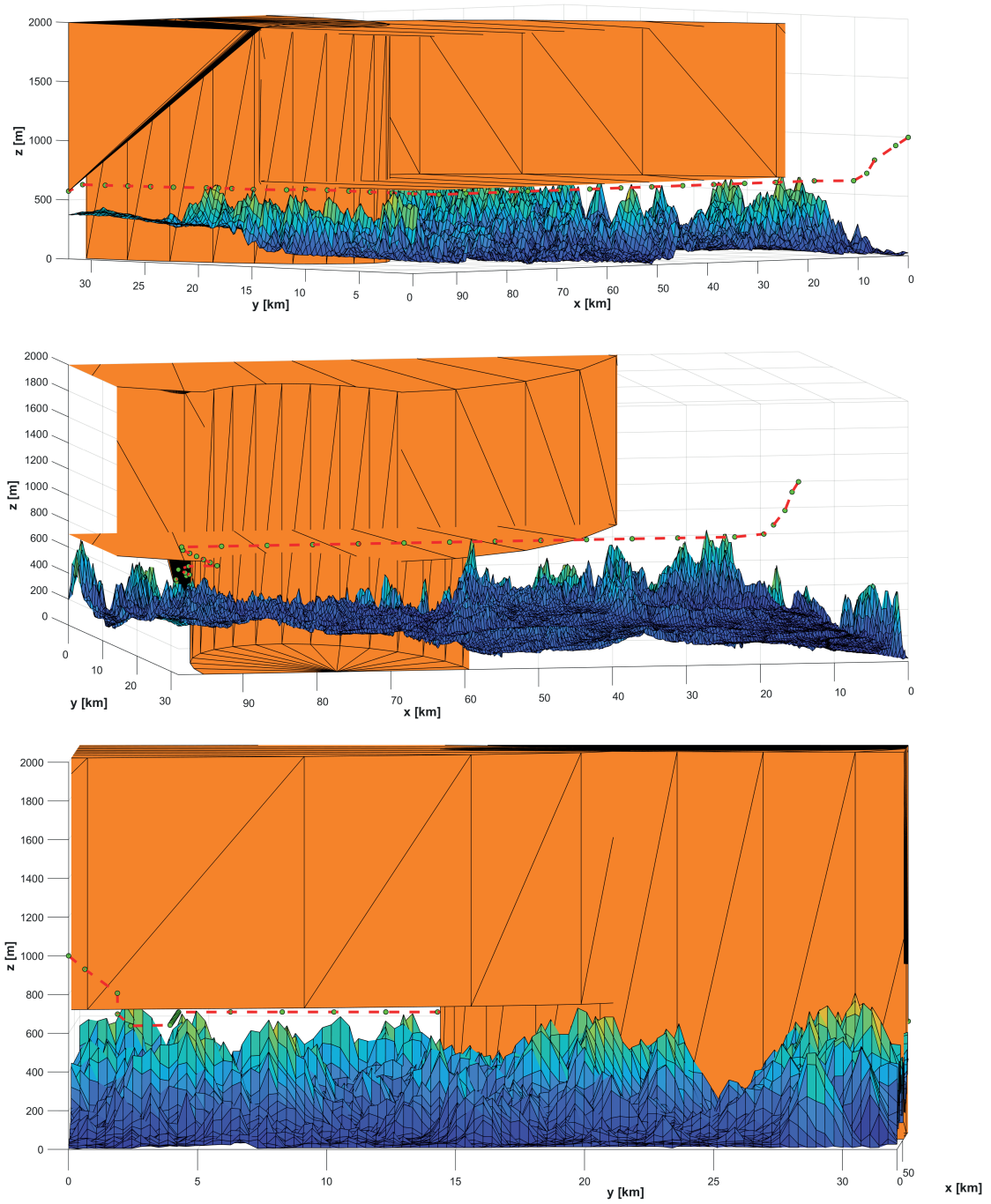

Figure 6.13: Final trajectory (red, discontinuous) delimited by waypoints (green dots) and obtained from RR-ACD algorithm for the mission example, seen from different perspectives.

6.5 Smooth 3D Path Planning

Although the path obtained by means of the RR-ACD algorithm is very appropriate for the aircraft guiding, technically, the aircraft can not follow it since it is not

able to turn any angle in a matter of null time; hence, the algorithm implemented in this section will attempt to apply radii of gyration to each waypoint location in order to obtain a smooth 3D path. However, this softened trajectory could collide with the obstacles which the RR-ACD code has tried to avoid, so that it is mandatory that the Smooth 3D Path Planning algorithm analyse if any collision exists between the new trajectory and the defined obstacles and, in the case that such event takes place, reduce the radii of gyration so as to diminish the collision probability. In the image 6.14, it is shown the concept that is wanted to be applied to this case so as to modify the reference trajectory.

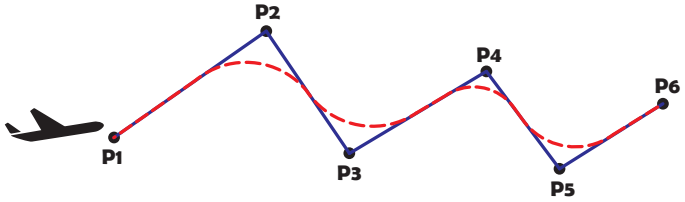

Figure 6.14: Conceptual example of the smooth 3D path planning: reference trajectory (blue), waypoints (black dots), soften path (red, discontinuous).

In this way, the algorithm finds the optimal path in terms of low angle variation rates and collision avoiding, basing on some concepts shown in [28]. Nevertheless, it is relevant to notice that this algorithm could not be necessarily applied, since incorporating a variable catching point length in the guiding algorithm from the Chapter 5, the aircraft could make those radii of gyration by itself and without the help of a Smooth 3D code; but, in that case, the optimal trajectory followed by the aircraft is not guaranteed (giving as result the possible deviation from the reference path), nor the collision avoiding.

The algorithm will be applied to each group of three waypoints which are not aligned among them. In the figure 6.15, an example of application of radius of gyration is represented, in which points s_1 and s_2 are the tangential points among the arc and both lines, vectors ($\overrightarrow{v}_1$,$\overrightarrow{v}_2$, $\overrightarrow{r}_1$, $\overrightarrow{r}_2$) the tools utilised to define the points s_1 and s_2, but also the arc center (O), and R is the radius. The direction of vectors $\overrightarrow{r}_1$ and $\overrightarrow{r}_2$ are result of the following vectorial product: $\overrightarrow{r}_i = \overrightarrow{v}_i \times (\overrightarrow{v}_1 \times \overrightarrow{v}_2)$.

The generation of the final soften trajectory is made by means of two steps at each iteration: firstly, the generation of a general soften path and velocity profile by means of the Algorithm 6 and, secondly, the analysis of errors in the trajectory building and collision of the new path with the obstacles, what is conducted owing to the Algorithm 7.

Regarding the first algorithm, the first step consists in initialise the needed variables: the path, yaw and flight path angles obtained from RR-ACD code; maximum, minimum and initial speeds that the velocity profile must present; the maximum positive velocity variation per meter ($\frac{\mathrm{d}V}{\mathrm{d}S}$); the precision of the trajectory building (ds)[12], and load factor parameters, such as the minimum load factor used at each curve (n_0), the maximum one allowed (n_{max})[13] and the vector $factors$.

[12]The Euclidean distance between consecutive points

[13]Remember that in this case: $n_{max} = 5.5$, see section 3.1.3.

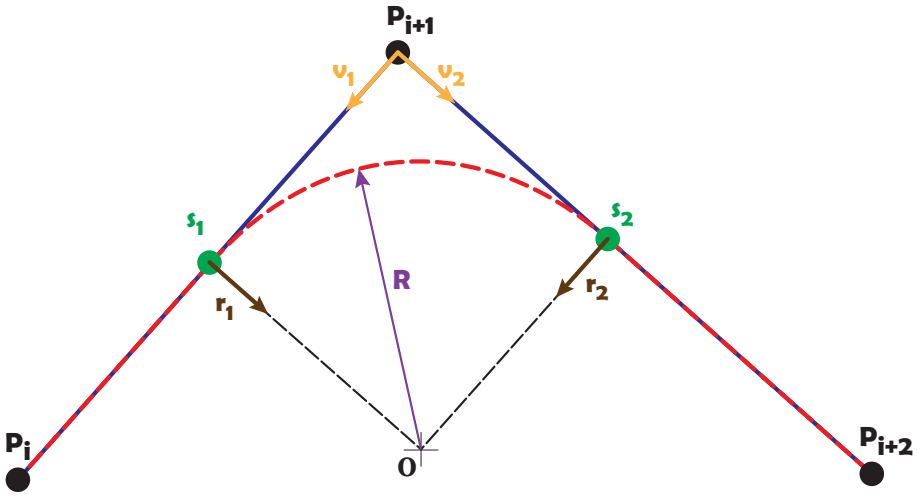

Figure 6.15: Application of radius of gyration: reference trajectory (blue), waypoints (black dots), soften path (red, discontinuous), radius (violet), center (black cross), s points (green dots), $v_{wp_{i+1}\to s}$ vectors (orange), $r_{s\to O}$ vectors (brown).

Velocity values are calculated utilising the velocity gradient ($\frac{\mathrm{d}V}{\mathrm{d}S}$), which will be the maximum positive increment in the total speed per meter that the aircraft can follow; however, it does not fix the maximum deceleration, which can be higher in absolute value, since a deceleration is a bit easier to achieve than an acceleration (as it can be observed in the graph 6.17). The maximum and minimum limits during the flight are stipulated by V_{max} and V_{min}, although when the aircraft is close to its final destination, it decreases the velocity until the desired impact velocity $V_{impact} = 150m/s$, which is set by default since it is the minimum speed that will allow an accurate impact of the bombs at the dropping time.

In relation to the vector $factors$, it contains the gradient of load factor with respect to yaw angle change ($\frac{\mathrm{d}n}{\mathrm{d}\psi}$), so that, when a change in ψ takes place, and additional load factor is applied; however, in the case that only a change in flight path angle happens, no gradient is applied and the minimum load factor (n_0) is used. In the expression 6.12, the formula in a case i is shown.

$$n_i = n_0 + factors(i) \cdot |\psi_{i+1} - \psi_i|/90^\circ \tag{6.12}$$

Attending to the radius of gyration (see [15]), there is an assumption in which, considering that the change in altitude is negligible in comparison with the movement in the XY plane, the radius can be approximated to the following function of velocity and load factor: $R = \frac{V^2}{9.81 \cdot \sqrt{n^2-1}}$. This radius will be also constrained to the maximum load factor that the aircraft can bear (n_{max}).

Afterwards, a while loop is started to analyse all waypoints. In the case that there is a change in yaw or flight path angle, the procedure shown in the figure 6.15 is employed, but when the three waypoints of study are aligned, a linear path is built from P_i to P_{i+1} utilising the function $build\_path2$. In the first assumption, there are two different possibilities: the points P_{i+1}, P_{i+2} and P_{i+3} are not lined up, so that the function $build\_path1$ gives the new part of the trajectory $path_{ref}(end)$-s_1-arc; and, on the other hand, whether they are in the same line, so the new section

of the path is $path_{ref}(end)$-s_1-arc-$path(i+2)$. The application of one case or the another depends on the value of variable $flag$, which can be 1 or 0. In addition, the selection of the next waypoint i can vary in function of the case of analysis, being as it follows: the new part $path_{ref}(end)$-s_1-arc provides $i+1$, but the one $path_{ref}(end)$-s_1-arc-$path(i+2)$ gives as result $i+2$, since all points between P_i and P_{i+2} have been included.

Algorithm 6 Smooth 3D Path Simple Design (S-3D-PSD)

define $path, \gamma, \psi, V_{min}, V_{max}, V_0, \frac{\mathrm{d}V}{\mathrm{d}S}, ds, n_0, n_{max}, factors$
$path_{ref} = [path(1,:)\ V_0]$;
$S = [0]$;
$s_v = []$
$i = 1$;
$flag = 0$;
while not $(i == size(path, 1))$ **do**
  if not $((\psi(i+1) - \psi(i)) == 0)$**or not** $((\gamma(i+1) - \gamma(i)) == 0)$ **then**
    $[\overrightarrow{v_1}, \overrightarrow{v_2}] = vector\_v(path(i : i+2, :))$;
    $\overrightarrow{v_{12}} = \overrightarrow{v_1} \times \overrightarrow{v_2}$;
    $[\alpha] = angle(\overrightarrow{v_1}, \overrightarrow{v_2})$;
    $n = load\_factor(n_0, factors(i), \psi)$;
    $R = max(\frac{(path_{ref}(end,4))^2}{9.81 \cdot \sqrt{n^2-1}}, \frac{(path_{ref}(end,4))^2}{9.81 \cdot \sqrt{n_{max}^2-1}})$;
    $[s] = s\_distance(R, \alpha)$
    $s_v = [s_v\ s;i]$;
    $\overrightarrow{r_1} = \overrightarrow{v_{12}} \times \overrightarrow{v_1}$;
    $\overrightarrow{r_2} = \overrightarrow{v_2} \times \overrightarrow{v_{12}}$;
    $[s_1, s_2] = s\_point(\overrightarrow{v_1}, \overrightarrow{v_2}, s, path, \psi)$;
    $O = center(s_1, s_2, r_1, r_2, v_1, v_2)$;
    $pc = circle\_points(v_{12}, s_1, s_2, O, R, ds)$;
    $[points_{ref}, S_{tot}, i_{val}, flag_{val}] = build\_path1(V_{min}, V_{max}, \frac{\mathrm{d}V}{\mathrm{d}S}, path_{ref}(end,:)$
    $, path(i : i+2, :), \psi(i : i+3), \gamma(i : i+3), v_1, v_2, s_1, s_2, ds, flag, pc)$;
  else
    $[\overrightarrow{v_1}] = vector\_v(path(i : i+1, :))$;
    $[points_{ref}, S_{tot}, i_{val}, flag_{val}] = build\_path2(V_{min}, V_{max}, \frac{\mathrm{d}V}{\mathrm{d}S}, path_{ref}(end,:)$
    $, path(i : i+2, :), \psi(i : i+3), \gamma(i : i+3), v_1, ds, flag)$;
  end if
  $path_{ref} = [path_{ref};points_{ref}]$;
  $S = [S\ S_{tot}]$;
  $flag = flag_{val}$;
  $i = i_{val}$;
end while

In relation to the Algorithm 7, once a first general trajectory and velocity profile are generated, the code attempts to find possible errors in the construction (given due to overlapping arcs) and study whether the new trajectory collides or not utilising the Collision Toolbox. At the beginning, the load factors are set at low values to create big radii of gyration, although it is sure that configuration will cause error, in order that the algorithm finds the optimal solution by means of optimising the vector $factors$. Therefore, the code analyses which the problematic waypoints are and adds to their respective $factors$ parameters the *precision* value, increasing the load factor in those points with excessive radius of gyration (these positions are defined by the vector *locations*) and trying to avoid that the arcs overlap. Every time a building error or collision is found, the whole code generates a new trajectory with the improved $factors_{new}$ vector; hence, only when the new path is perfectly well defined, the algorithm ends and the solution is obtained.

Finally, the yaw and flight path angles of the new trajectory are calculated with the function *new_angles*. Also, the time variation of the yaw angle ($\frac{\mathrm{d}\psi}{\mathrm{d}t}$) is obtained, which is used by the guiding code to apply the proper control actions.

Algorithm 7 Smooth 3D Path Planning (S-3D-PP)

define $path, \gamma, \psi, V_{min}, V_{max}, V_0, \frac{\mathrm{d}V}{\mathrm{d}S}, ds, n_0, n_{max}, precision, obstacles$
$factors = zeros(size(path, 1) - 1)$
$solution = 0$
repeat
  $\left[path_{ref}, S, s_v\right] = S\_3D\_PSD(path, \gamma, \psi, V_{min}, V_{max}, V_0, \frac{\mathrm{d}V}{\mathrm{d}S}, ds, n_0, n_{max}, factors)$;
  $[error, locations] = error\_finding(path_{ref}, s_v)$;
  $[collision, locations] = collision\_finding(path_{ref}, obstacles)$;
  if *error* ==**true then**
    $[factors_{new}] = modification\_factors(factors, locations, precision)$;
  else if *collision* ==**true then**
    $[factors_{new}] = modification\_factors(factors, locations, precision)$;
  else if (*error* **and** *collision*) ==**false then**
    $solution == 1$
  end if
  $factors = factors_{new}$;
until $solution == 1$
$\left[\psi_{new}, \gamma_{new}, \frac{\mathrm{d}\psi}{\mathrm{d}t}\right] = new\_angles(path_{ref})$;
return $Final\_trajectory\_parameters \rightarrow wp = \left[path_{ref}\ \psi_{new}\ \gamma_{new}\ \frac{d\psi}{dt}\right]$

In the figure 6.16, the final result of the algorithm is shown. The parameters utilised in this case are given in the expressions 6.13.

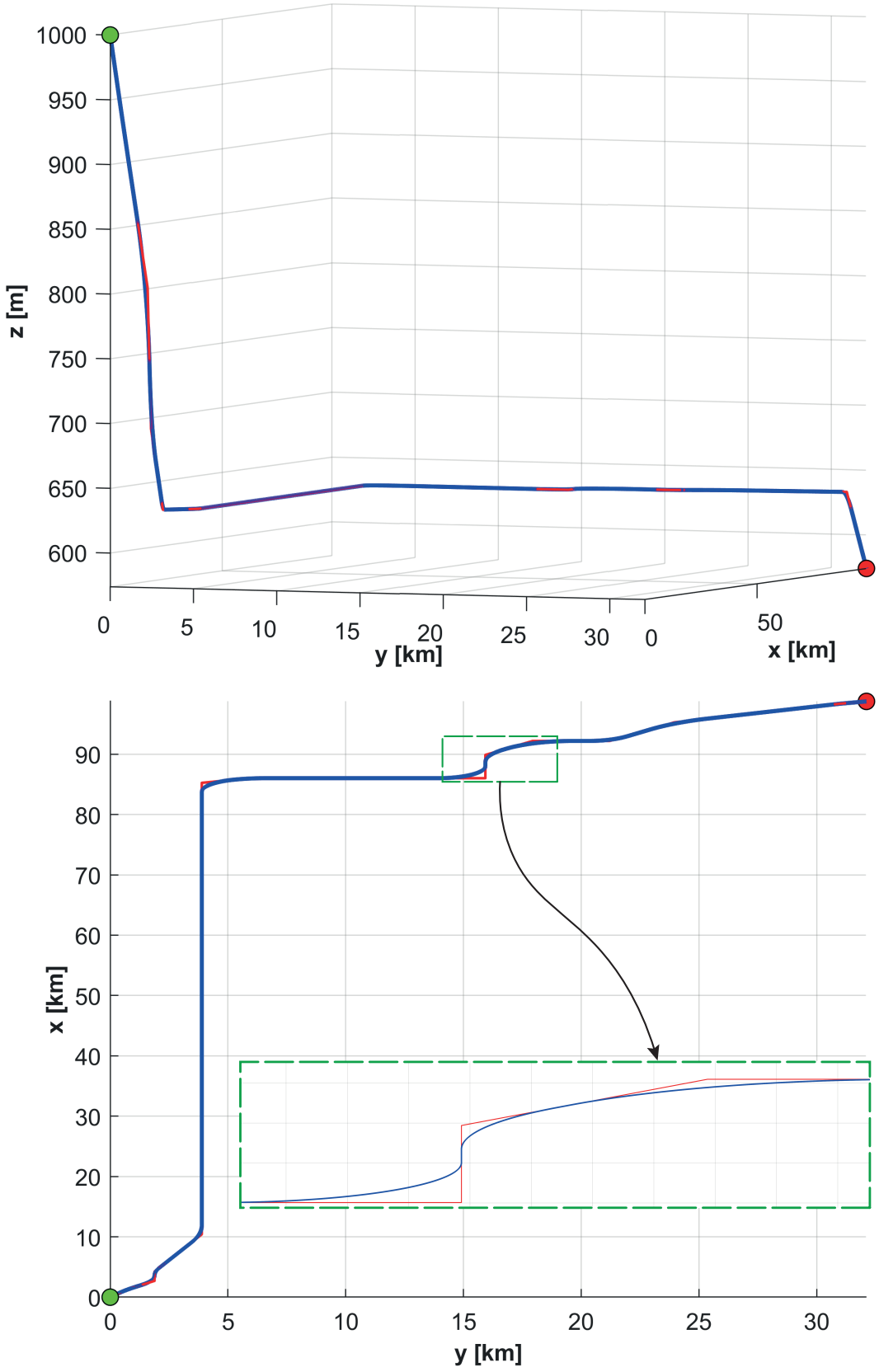

Figure 6.16: Final trajectory obtained by the Smooth 3D path planning algorithm: RR-ACD path (blue), final trajectory (red), initial point (green dot), final point (red dot).

$$\begin{cases} V_{min} = 180\frac{m}{s} \\ V_{max} = 250\frac{m}{s} \end{cases} \quad \begin{cases} V_0 = 200\frac{m}{s} \\ \frac{\mathrm{d}V}{\mathrm{d}S} = 5\frac{m/s}{km} \end{cases} \quad \begin{cases} ds = 20m \\ n_0 = 1.075 \end{cases} \quad \begin{cases} n_{max} = 5.500 \\ precision = 1 \cdot 10^{-2} \end{cases} \tag{6.13}$$

The velocity profile can be seen in the image 6.17, showing the idea that has been attempted to apply at the dynamic trajectory: diminishing the speed during curves and descents and increasing it at straight lines in order to minimise the flight time of the path and make it easier to follow. As it has been mentioned, the deceleration is slightly greater in most of the cases than the acceleration, seeing that the aeroplane is hoped to reduce speed at a lower time than if it tries to increment it, due to drag forces.

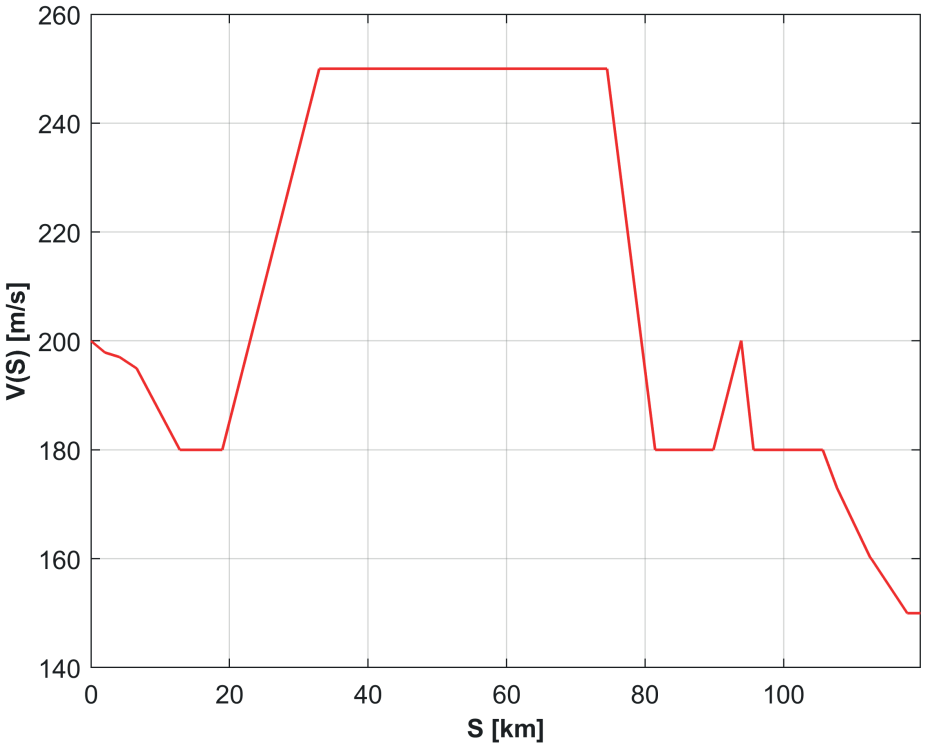

Figure 6.17: Velocity profile in function of the curvilinear total distance S.

And last but not least, the yaw angle time variation result is presented in the figure 6.18, in which no curve presents an excessive angular rate. The velocity profile has been adapted to the need of diminish this time variation as much as possible, so that before the aircraft enters into a strong curve, the speed is fairly decreased. The sudden changes in the graph shows the start of a curve, and therefore, after these changes, if the value does not remain constant, it means that still the velocity is being reduced.

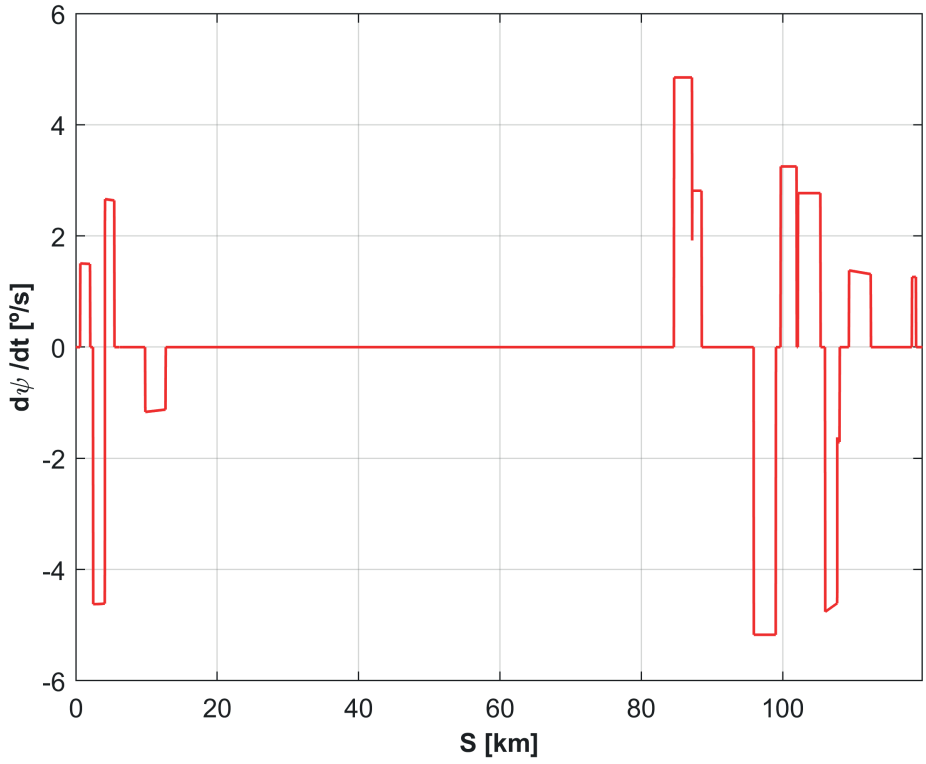

Figure 6.18: Time variation of the yaw angle in function of the curvilinear total distance S.

6.6 Dropping point estimation

In order to estimate the ideal dropping point from the final reference trajectory, obtained from the RR-ACD algorithm and modified by the Smooth 3D Path Planning one, the mathematical model of the bombs (see Chapter 3) based on [6] is used. In this case, the code employed is fairly similar to the one utilised to linearize the model of the aircraft, taking the same equations, although supposing no rotation in the x-axis of the aeroplane ($\phi = 0°, p = 0°/s$).

The Algorithm 8 shows the procedure to obtain the possible dropping points and the best one of them, together with all trajectories described by the bombs and their flight parameters at each case. First of all, two arbitrary distances d_{min_1} and d_{min_2} are defined: the first one will be the maximum distance between a point of the reference trajectory and the final destination (q_f, which now it is modified, setting its z-coordinate to the ground level value), therefore, it fixes the initial index ($i_{initial}$) of the path vector ($path_{ref}$) from which the algorithm starts; the second one sets the final index of the path vector that the algorithm analyses. Both parameters are determined in function of the bomb dynamics. In addition, the integration time is defined.

Then, the algorithm extracts the initial conditions from $path_{ref}$ and assigns them to the vector $pmts$ that is going to contain the flight parameters of the bombs and the time. The trajectory of both bombs are supposed to be the same, starting from the similar point. Afterwards, a while loop starts to calculate the flight param-

eters at each time step until the bombs reach the ground, instant at which the loop ends. The impact distance is calculated by means of a bilinear interpolation (see [3]), which interpolates the ground elevation from the closest four elevation points to the bombs at each instant of time[14], giving as result an approximation of the elevation at those coordinates.

At the end of the while loop, the parameters are saved and their positions in the general vector $pmts_v$ in the positions vector $positions_v$. If the final point of the trajectory of the bombs exceeds the distance d_{min_2}, the loop ends and the index of $path_{ref}$ that contains the best dropping point option and its deviation in the impact from q_f are calculated.

Algorithm 8 Dropping Point Calculator

define $path_{ref}, ground\_elevation, q_f, d_{min_1}, d_{min_2}, geometry\_bomb, dt$
$positions_v = pmts_v = []$
$[i_{initial}] = position\_finding(d_{min_1})$
for $i = i_{initial} : -1 : 0$ **do**
  $[u, v, w, \theta, \psi, q, r, x, y, z] = initial\_conditions(path_{ref}(i, :), geometry\_bomb);$
  $pmts = [0, u, v, w, \theta, \psi, q, r, x, y, z]\,;$
  $z_{impact} = q_f(3);$
  while $z > z_{impact}$ **do**
    $\left[\dot{u}, \dot{v}, \dot{w}, \dot{\theta}, \dot{\psi}, \dot{q}, \dot{r}, \dot{x}, \dot{y}, \dot{z}\right] = aerodynamics([u, v, w, \theta, \psi, q, r, x, y, z]);$
    $[u, v, w, \theta, \psi, q, r, x, y, z] = integrator(\left[\dot{u}, \dot{v}, \dot{w}, \dot{\theta}, \dot{\psi}, \dot{q}, \dot{r}, \dot{x}, \dot{y}, \dot{z}\right], dt);$
    $pmts = [pmts; [pmts(end, 1) + dt, u, v, w, \theta, \psi, q, r, x, y, z]]\,;$
    $z_{impact} = bilinear\_interpolation(ground\_elevation, x, y);$
  end while
  $positions_v = [positions_v; [size(pmts_v, 1) + 1, size(pmts_v, 1) + size(pmts, 1)]]\,;$
  $pmts_v = [pmts_v; pmts]\,;$
  if $(|pmts(end, 9)| > |q_f(1)| + d_{min_2})$ **or** $(|pmts(end, 10)| > |q_f(2)| + d_{min_2})$ **then**
    break
  end if
end for
$[index_{optimal}, deviation] = closest\_point(pmts_v, positions_v, q_f)$
return $index_{optimal}, deviation$

The parameters used in the simulation of the mission example are the following ones:

$$d_{min_1} = 1500m, \quad d_{min_2} = 100m, \quad dt = 2 \cdot 10^{-3}s \tag{6.14}$$

[14]When the bombs overflow the final destination q_f, the impact altitude is set as the one of the point q_f.

The results of the algorithm of the present case are shown in the figure 6.19. A very little deviation is obtained from the simulation (5.74m), so that the whole procedure accomplished to define a trajectory, which guides to the aircraft to a possible dropping point, has been successful, since obstacles are avoided and the target is reached accurately. However, this is the ideal case in the real simulation, in which the airplane follows exactly the reference; consequently, if the control is not correct or the perturbations are excessive for the system, the objective of the mission could be accomplished with a very low precision.

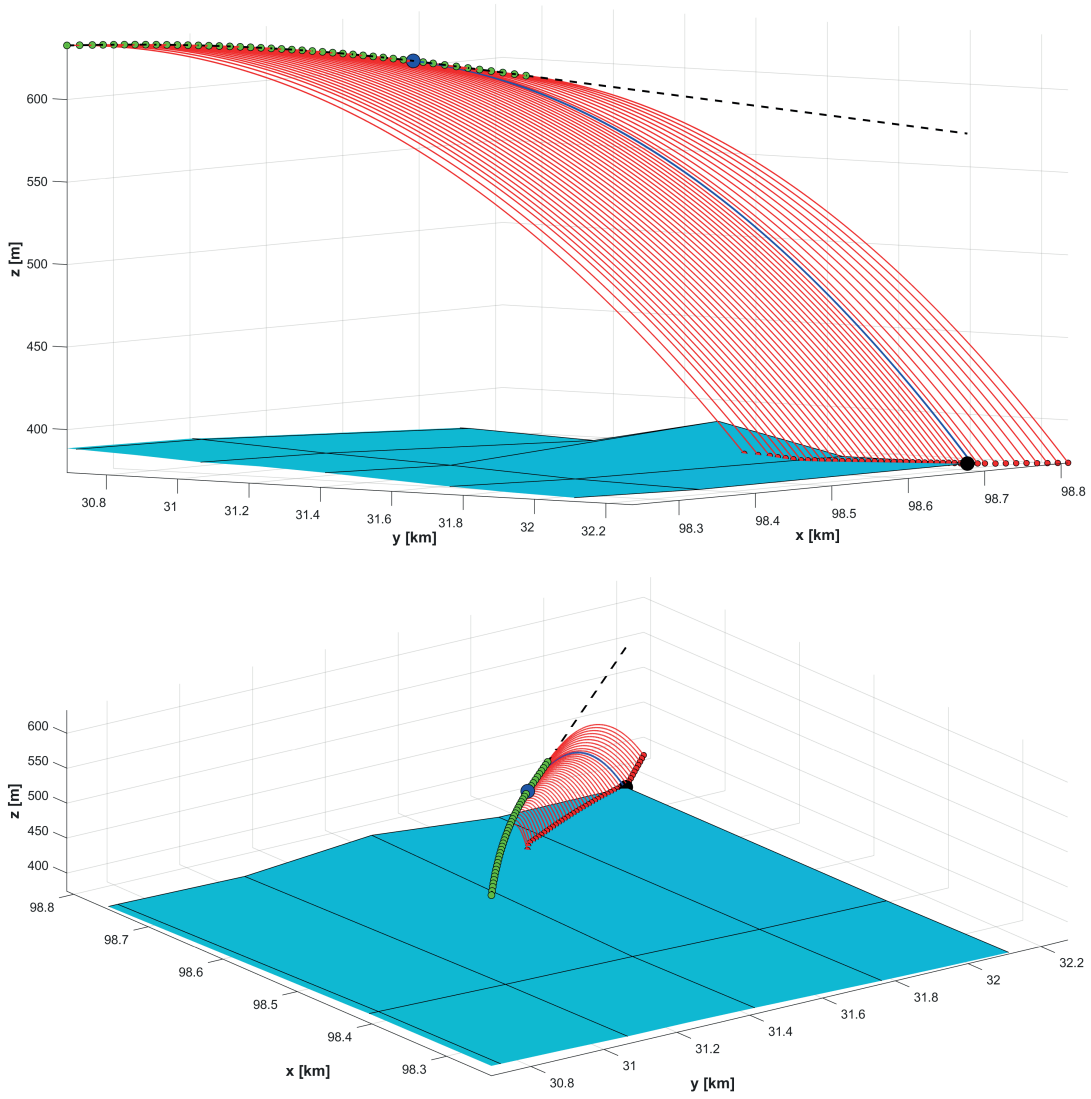

Figure 6.19: Dropping options and final selection from different perspectives: reference path (black, discontinuous), dropping optional points (green dots), impact points (red dots), mission objective (black dot), selected dropping point (blue dot), free-falling paths of the bombs (red), chosen bomb trajectory (blue).

In the group of figures 6.20, the flight parameters of the bombs during the simulation time until the impact on the ground are shown. It can be observed how the velocity is increased and pitch angle decreased due to the free-falling; in addition, the rotational movement in the z-axis is dampened.

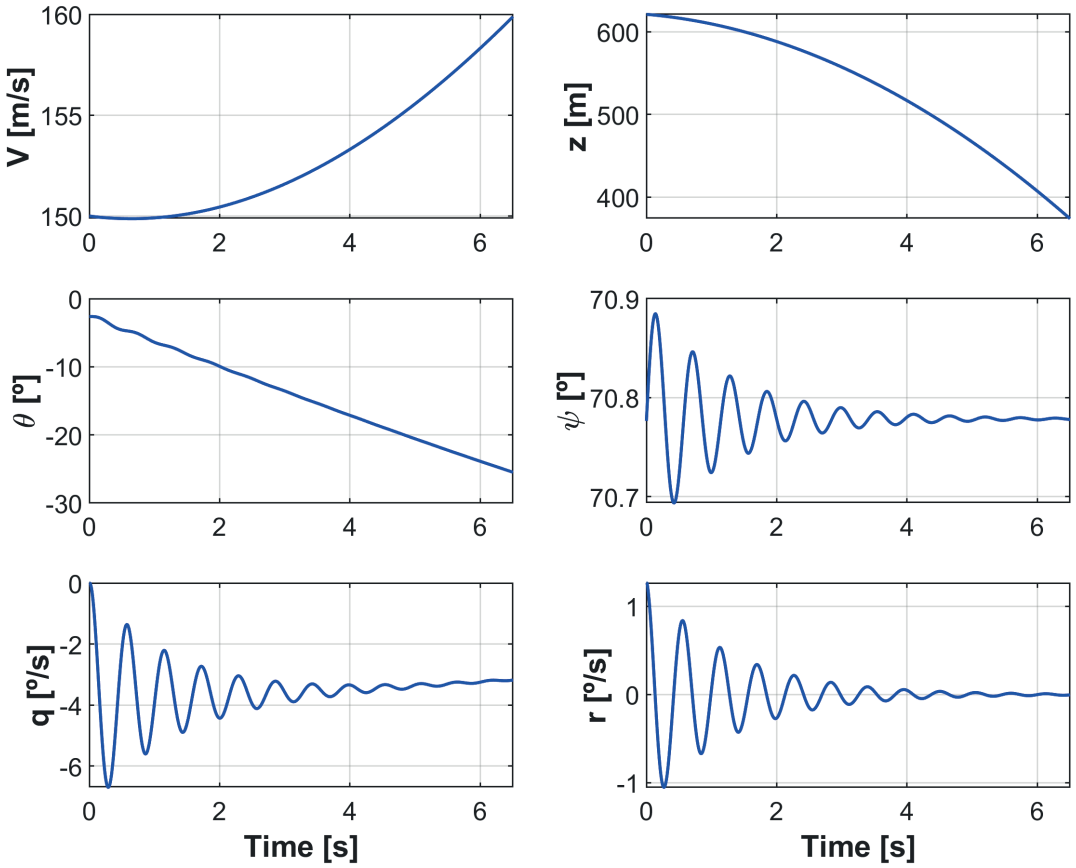

Figure 6.20: Flight parameters of the bombs in the selected trajectory to the target.

Finally, a brief analysis is conducted in wind condition in order to determine whether the wind velocity can difficult the proper impact of the bombs and vary significantly the wind parameters with respect to the ones at base conditions.

Due to the fact that the bombs trajectory is very short (that is why the dropping point is set at low heights above the ground) and they are very heavy, the wind modifies slightly the final result; hence, the wind velocity does not have a serious impact on the dropping path of the bombs. In the table 6.1, an example of the deviations at the different velocity values are presented, denoting that the error due to wind is very low. In addition, the optimal indexes of $path_{ref}$ are the same as the base case at the low wind simulations; only when the wind is very intense, the index changes, but just at one level. However, as it can be seen in the figure 6.21, the flight parameters (except for the coordinates) are noticeably modified, although it hardly influences the impact owing to the short flight time.

The reference case will be set at wind conditions of velocity $V = 15m/s$, direction $\psi = 45°$ and elevation $\theta = 0°$, since they are the mean conditions of the wind model that is going to be applied into the simulation. However, two different references could be taken, seeing that the simulations are conducted at wind and non-wind conditions (see Chapter 7), but as it is seen below, the difference in the dropping path between the data with $V = 0m/s$ and $V = 15m/s$ is almost negligible, so that the results with $V = 15m/s$ are used in all cases.

Deviations				
V_{wind} (m/s)	$V_{direction}$ (°)	$i_{optimal}$	Optimal error (m)	Base deviation (m)
0	0	5966	5.75	5.75
15	45	5966	7.59	7.59
30	45	5966	9.01	9.01
45	45	5966	11.56	11.56
60	45	5965	10.84	17.69

Table 6.1: Example of deviations in the impact of M117 bombs depending on the wind speed.

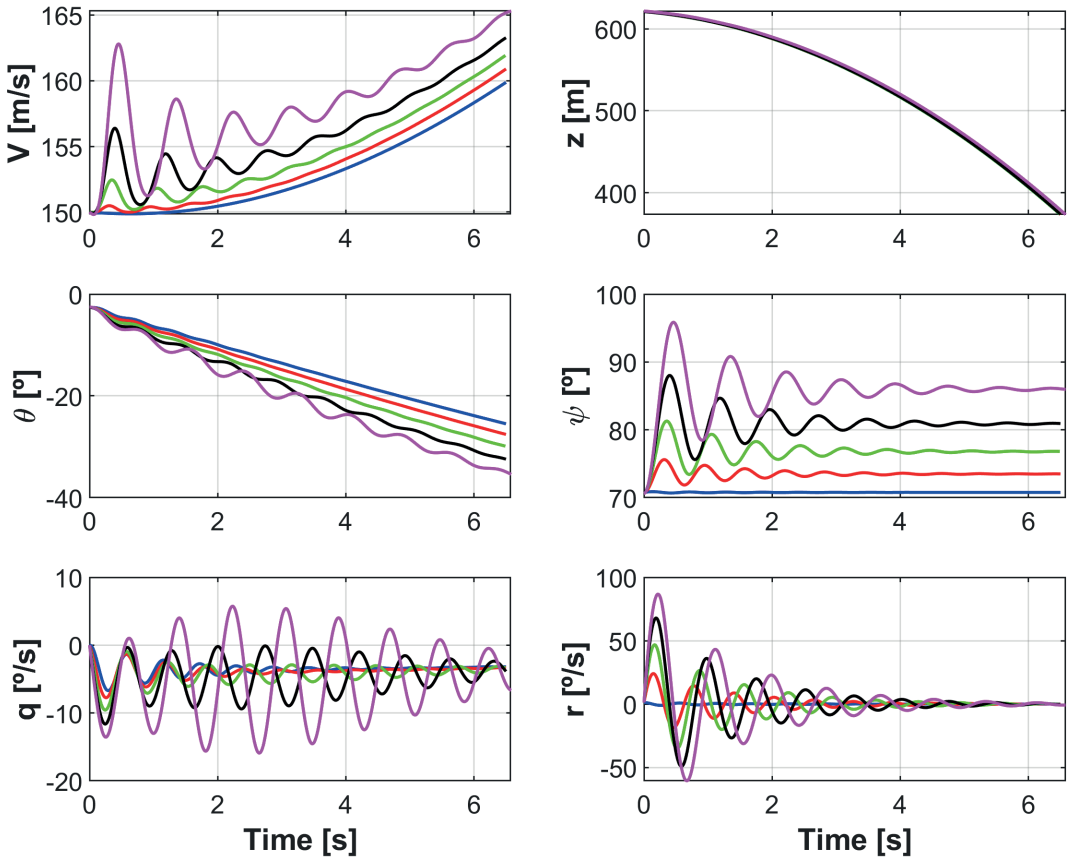

Figure 6.21: Flight parameters of the bombs in the base trajectory to the target at wind direction 45°and values: $0m/s$ (blue), $15m/s$ (red), $30m/s$ (green), $45m/s$ (black), $60m/s$ (magenta).

Despite the fact that the optimal dropping position has been found, if the control system of the aircraft is not adequate enough or there are strong perturbations, the impact of the objective could be achieved with a high deviation. Hence, the Algorithm 8 is also applied to the real simulations in the Chapter 7 in order to find the best points to release the bombs, analysing the distance from these points to the reference dropping position.

7 | Results

In this chapter, the results of the simulations accomplished under the different conditions are going to be analysed, determining whether the controllers could be valid and which one would be more appropriate in each case.

The mathematical model of the aircraft is linearized at a velocity $V = 215m/s$ and an altitude $z = 787m$, which are the mean values of the expected minimum and maximum of each variable[1].

Firstly, the response at non-wind conditions and constant mass is going to be conducted, studying the results in the simplest case without perturbations; secondly, the wind model defined in the section 3.1.7 is applied, creating a normal distributed wind with a mean value in the modulus of 15 m/s and in the direction of 45°, together with a constant mass of the whole system, and finally, the complete response is studied, in which the wind model and the variable fuel mass $m_{fuel}(t)$ according to the explanation in the section 3.1.5. At each graph shown in this chapter, the blue color will be associated with LQR results, red one with MPC and green with the reference variables at each case.

Afterwards, an analysis about the dropping and impact points of the bombs at each different case is conducted. In addition, the flight time and total fuel mass consumption are determined.

7.1 Response at non-wind conditions

This is the simplest case, since no perturbations are introduced in the system: the wind velocity is null and the Mach of divergence (M_{div}) is not surpassed, so that the wave drag is zero ($CD_{wave} = 0$).

In the figure 7.1, the velocities and altitudes during the flight of each control method are compared. The discontinuous green lines are the references that the guidance algorithm (see Algorithm 3) extracts from the reference dynamic path, giving to the aircraft the required velocity and altitude at each time step.

As it can be observed, both controllers have not been able to track the reference velocity properly during the whole trajectory, however, the MPC offers more ade-

[1] Although the minimum reference velocity achieves 150 m/s, it is only achieved at the end of the path, so that the minimum is considered as 180 m/s.

quate results than LQR, since the deviation is lower due to the prediction capability of the MPC.

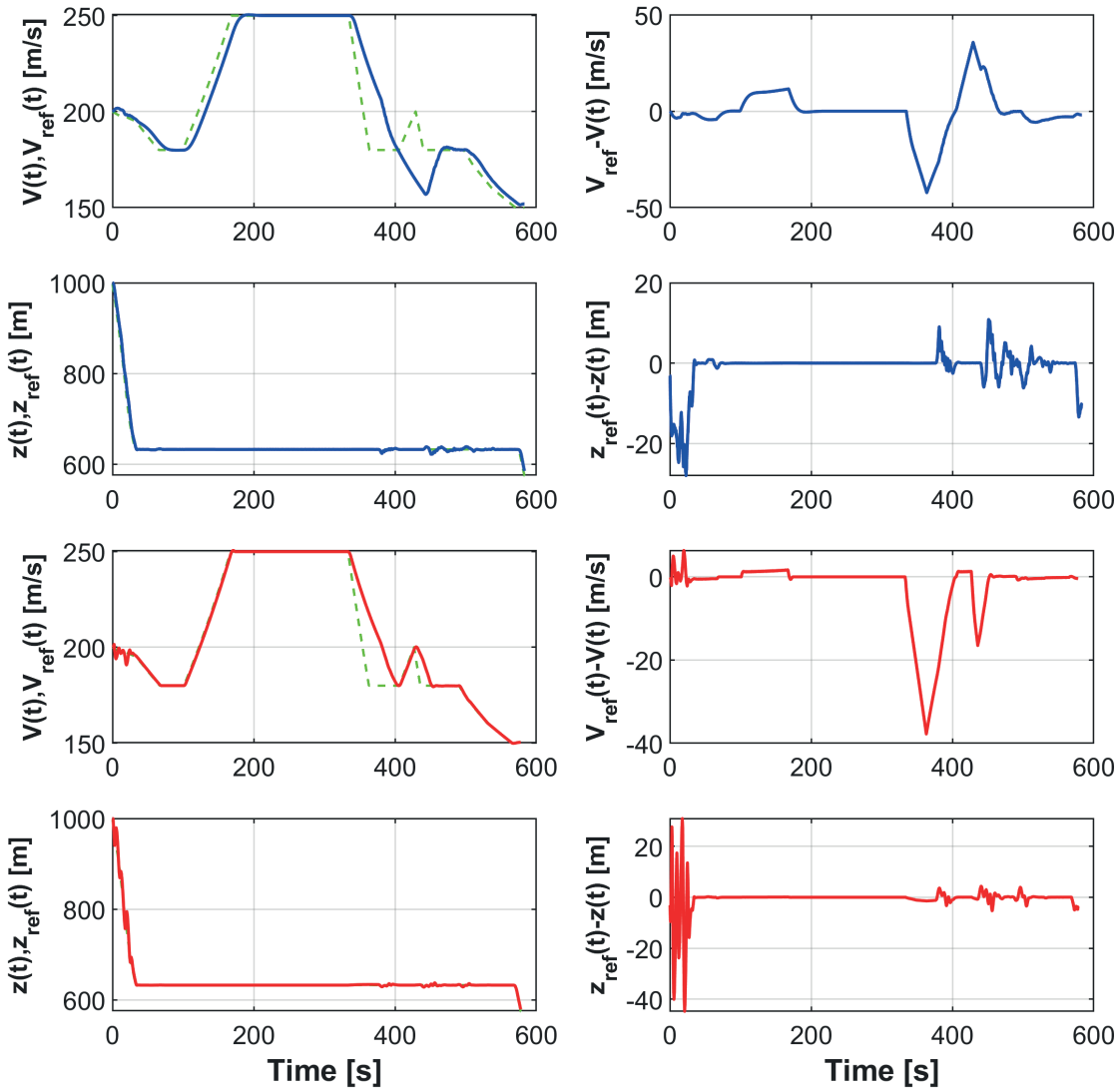

Figure 7.1: Reference tracking of velocity and altitude: LQR (blue), MPC (red), reference at each case (green, discontinuous).

Regarding the altitude, there also errors in the tracking and above all in the areas where the aeroplane must turn: due to the fact that the model is linearized at trimmed conditions in the vertical plane motion (see Section 3.3.1), the system and control matrices consider the influence of the elevator in turns as negligible and the roll angle variation does not have impact on the altitude modification, giving a wrong idea about the real dynamics of the aircraft at these parts of the flight (longitudinal and lateral-directional motions are coupled). In addition, these deviations in the flight level are increased when the aeroplane needs to descend and also make turns, seeing that modifications in the flight path angle and roll angle at the same time are very difficult to manage owing to the non-linear character of the system.

In this way, the differences with respect to the reference altitude is higher at the initial and final parts of the path. The deviations of the MPC results during the

initial descent are greater than the ones of LQR, nevertheless, the error has been reduced for the rest of the trajectory in the case of the MPC.

In the group of graphs 7.2 and 7.3, the flight parameters of angle of attack (α), sideslip angle (β), Euler angles (ϕ, θ, ψ) and angular body velocities (p, q, r) are shown.

When the aircraft is turning, the angle of attack presents greater oscillations in the LQR results than the MPC ones, although in the initial descent the MPC results present acute variations in this angle, producing negative values of α that could lead to stall in the real behaviour of the aeroplane. However, despite the initial values of angle of attack by means of the MPC controller, the rest of values of both simulations are acceptable. In this way, here the variations in the altitude with respect to the reference one can be observed, seeing that strong fluctuations in the altitude parameter produce fast modifications in the angle of attack as well.

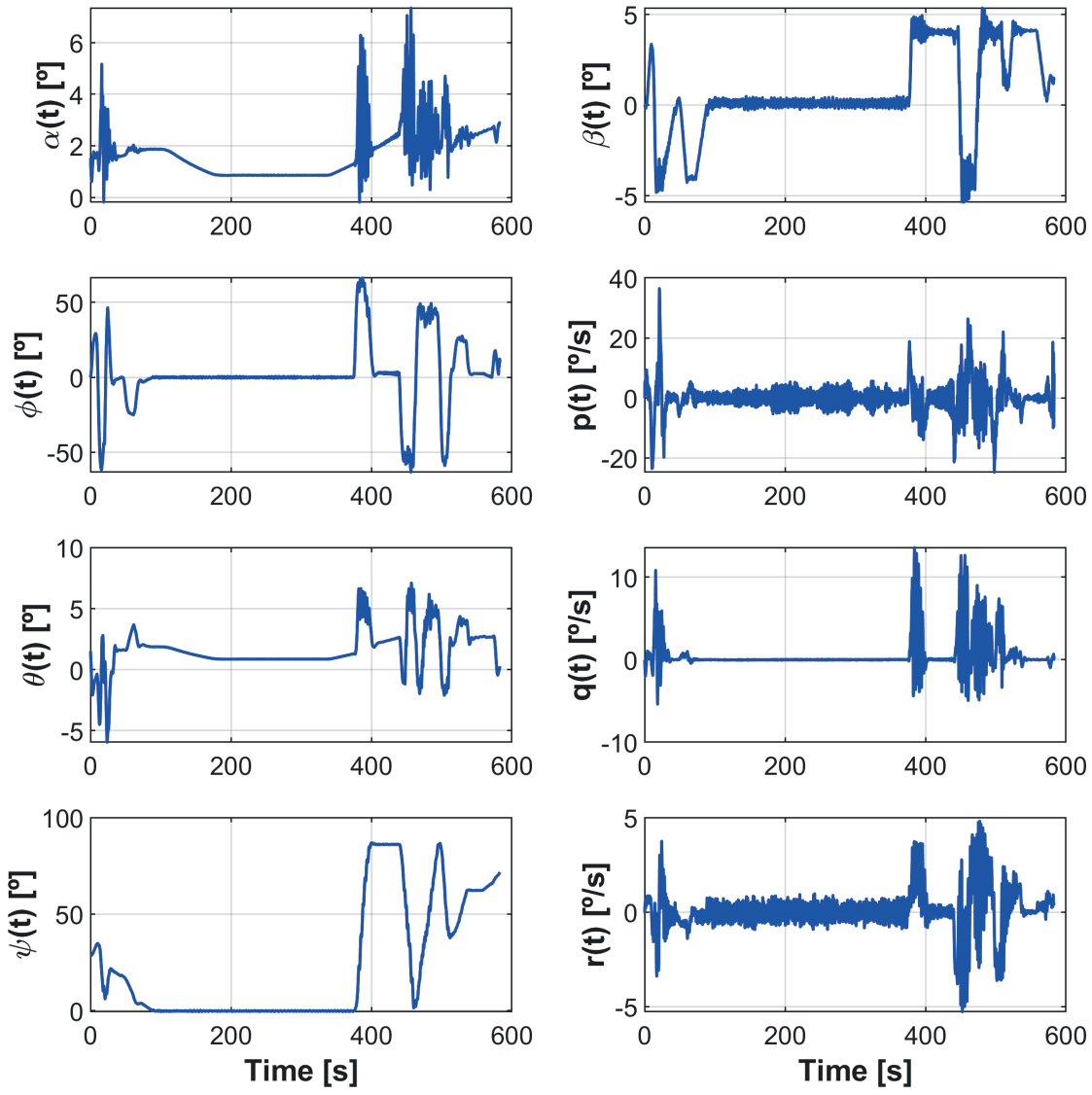

Figure 7.2: Principal flight parameters by LQR control.

Furthermore, the error in altitude can be also detected in the excessive variations of the pitch angle and angular velocity q, therefore, MPC results show great values at the beginning and ones closer to zero in the rest of the simulation, while LQR ones do not oscillate too much at the descent but during the curves the behaviour is more aggressive.

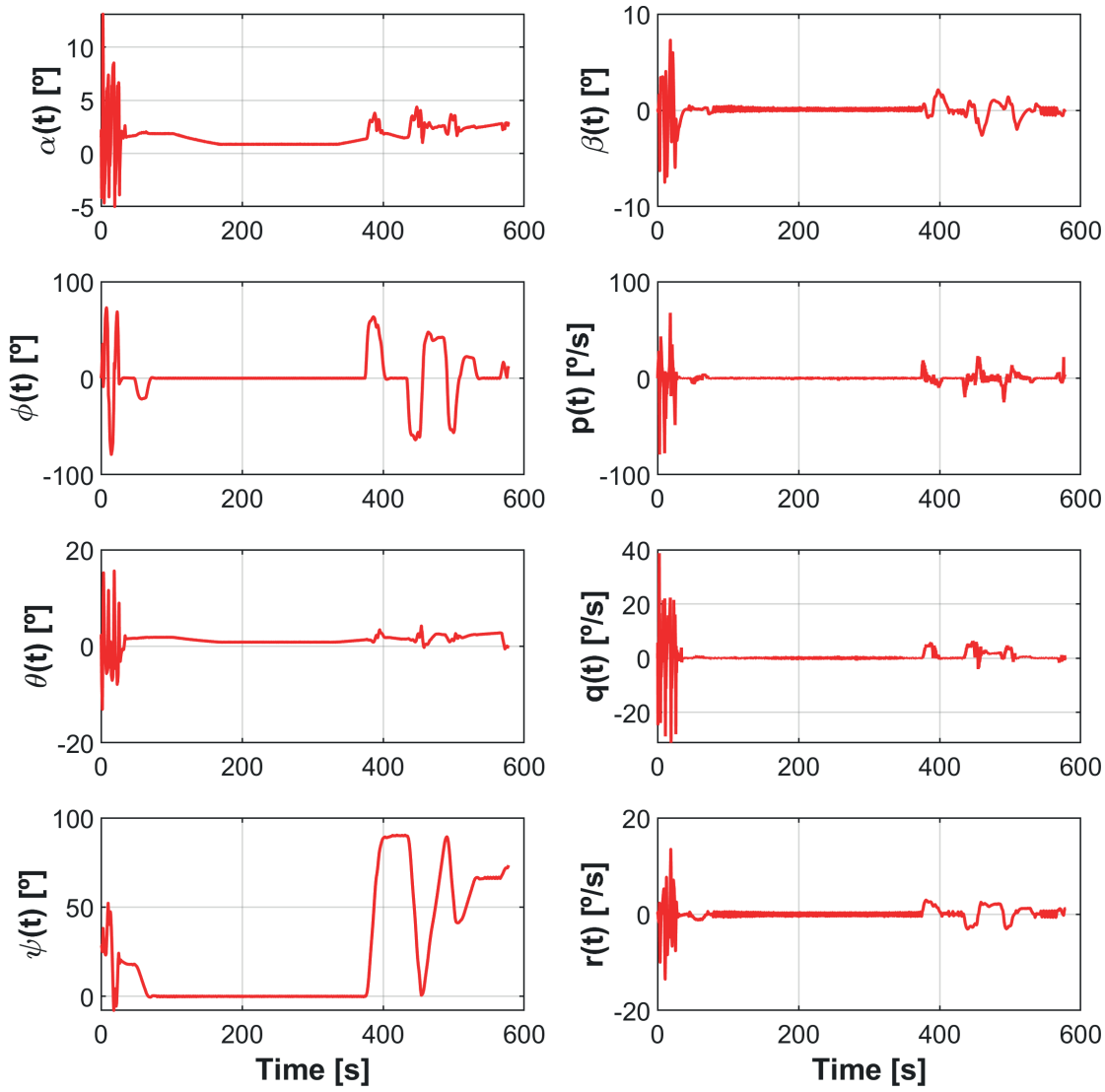

Figure 7.3: Principal flight parameters by MPC control.

The same thing happens in the sideslip angle, at the initial descent MPC results are more oscillatory, but at constant flight level LQR provides poor results in terms of oscillations and size of the values (nevertheless, this is not a huge problem, since a negative β does not give as a result stall, in contrast to α). However, it is important to notice that neither LQR nor MPC do get an optimal response along the complete trajectory: for instance, in the zone around time $t = 200s$, in which there is a long straight trajectory (see figure 7.5), the sideslip angle should be zero, since there is non-wind conditions; but, in both cases, this angle oscillates with short amplitudes due to the fact that the controllers are not able to achieve the ideal exact position of the aircraft.

Regarding roll and yaw angles, the results obtained are acceptable in both of the controllers, since the values are the expected ones and the limits in roll angle are respected. In relation to the angular velocities p and r, it happens the same as before, the MPC gives more aggressive results at the initial descent and the LQR at the curves.

In the figure 7.4, the control actions that take place along the simulation times are shown. The control conducted by the MPC is very aggressive during the descent, while for the rest of the path the LQR controller commands more demanding actions to proceed with the manoeuvres and diminish the error in the tracking. As it has been mentioned, during the long straight part of the trajectory (around time $t = 200s$), both controllers are not able to achieve the optimal position of the aircraft and avoid the oscillatory motion, hence, the ailerons and rudder are continuously activated in slight amplitudes.

In relation to the initial descent, the error in altitude tracking of the MPC can be understood by means of the analysis of the graph of throttle lever control: while the LQR attempts to diminish the thrust gradually to follow the reference in velocity, but also in altitude, the MPC fails in reducing the thrust and creates a problematic throttle lever signal, so that in order to track the velocity and altitude references, the system tries to use the elevator, giving as a result the acute signals of the figure. However, from the end of the descent, the MPC manages properly the coordinated use of throttle lever and elevator to achieve the required altitude and speed.

In addition, the error in altitude of LQR is also caused by wrong control actions, in this case, the aggressiveness in the use of the elevator at the curves leads to poor tracking in the flight level.

It is also necessary to highlight that in the case of the ailerons deflection, at no time the maximum and minimum values are employed to follow the reference, which will be useful when perturbations are given, since more ailerons deflection than this one is possible.

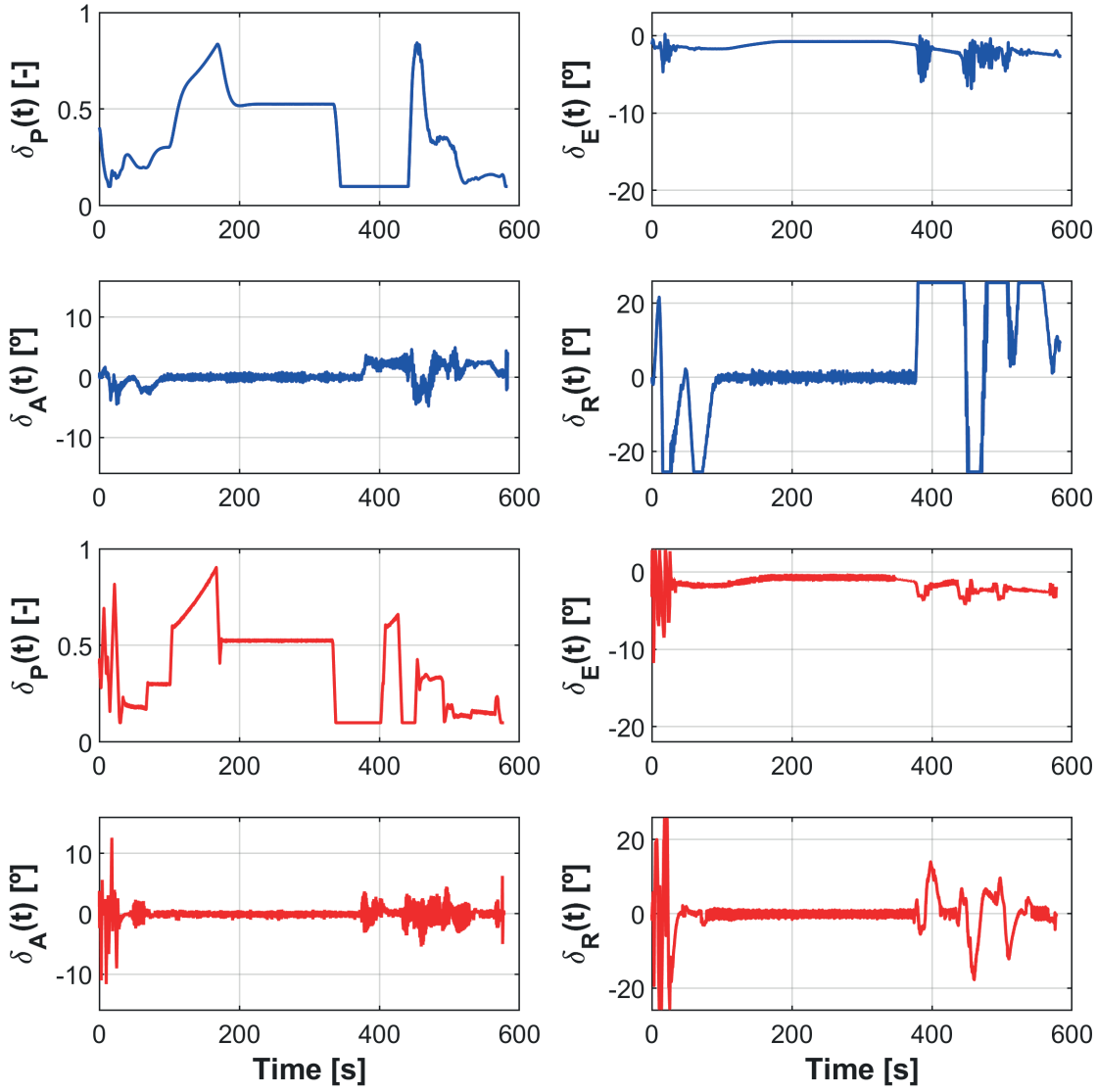

Figure 7.4: Control actions: LQR (blue), MPC (red).

In the figure 7.5, the 3D trajectories can be observed. Here, the errors in descent (MPC) and curves (LQR) can be appreciated, together with the really satisfactory tracking in the XY plane of both controllers due to the efficacy of LQR and MPC in following the roll and yaw angles from the guidance algorithm.

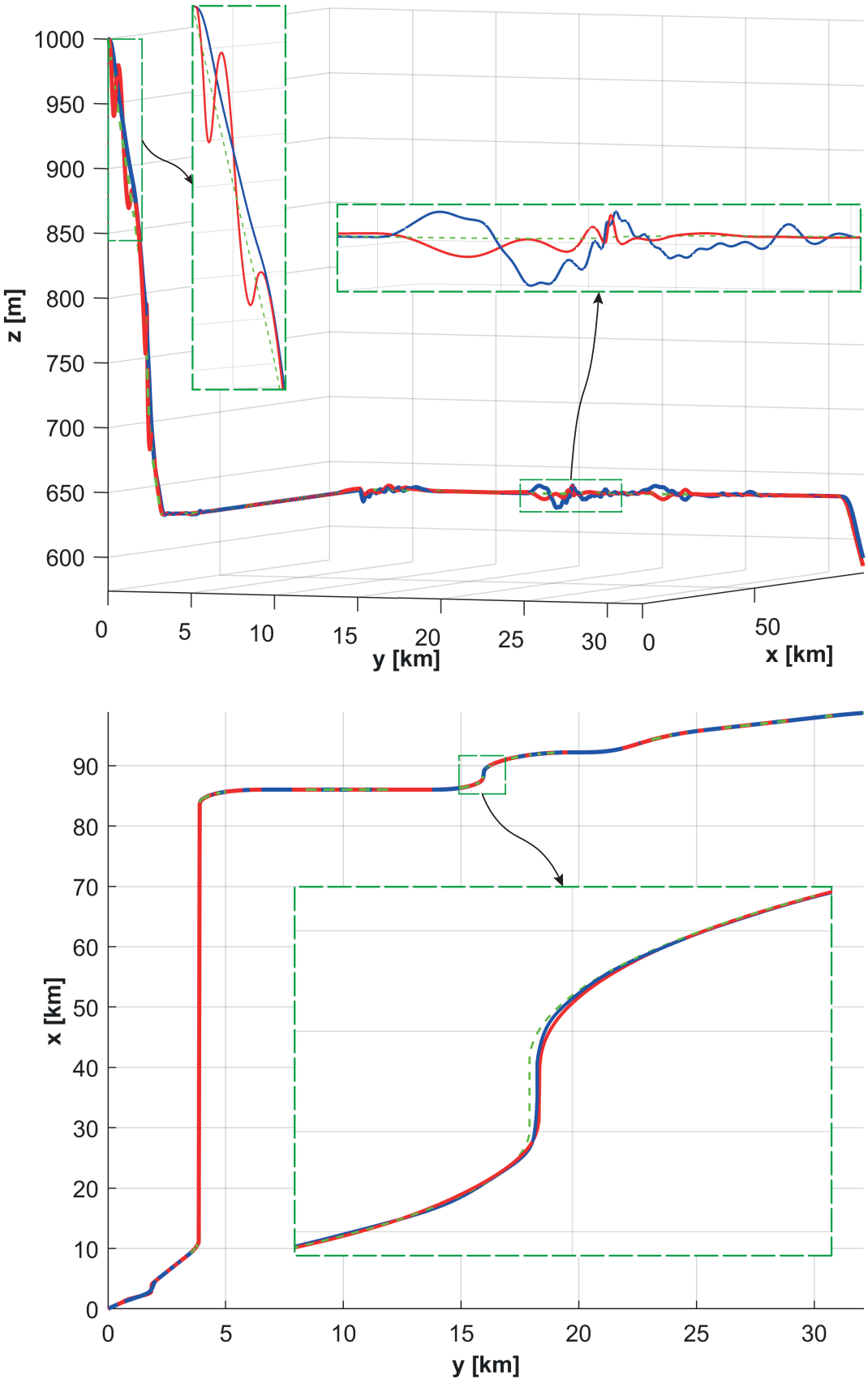

Figure 7.5: Final trajectory in two perspectives: LQR (blue), MPC (red), reference at each case (green, discontinuous).

Finally, the flight parameters of the bombs for the optimal free-falls, which give the best possible impact at each case, are shown in the image 7.6. As it can be seen, the dropping in MPC results is conducted at the reference velocity of bomb release ($V = 150m/s$), while the one at LQR case is accomplished at a greater speed, but also at a higher altitude. This fact, makes possible that the M117s impact on the ground in a briefer time in the MPC case than in LQR one, since the range of the parabolic motion is shorter. Furthermore, the initial yaw angle is not the same in both cases, leading to a different direction of the bombs trajectory.

In addition, the variation rate of the yaw angle is lower in the MPC case, due to the fact that the pitch angle of the aircraft at the dropping instant is higher in the LQR results than in MPC ones, which interferes with the motion in z-axis and creates greater oscillations. In the case of the angular velocity q, the results are similar in both cases.

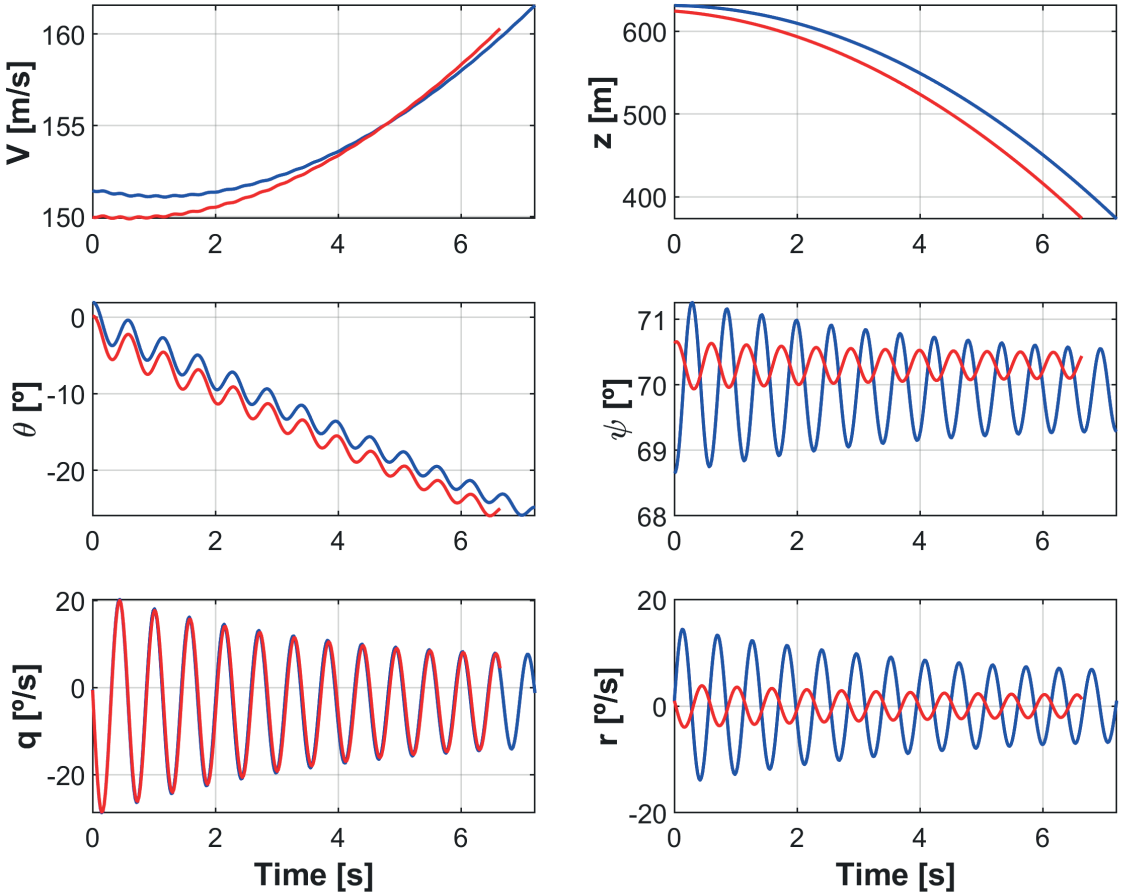

Figure 7.6: Trajectory of the M117 bombs from the optimal dropping point: LQR (blue), MPC (red).

7.2 Response with wind model

In this case, the simulations are conducted in the case of wind perturbations based on the normal distributed model. As before, compressibility effects are not

given, since the maximum velocity achieved does not exceed the Mach of divergence.

The wind velocities at each axis can be observed in the figure 7.7, where the mean values and standard deviations of modulus and direction and elevation angles of the wind model are respected, creating the result of the figure. The values of the figure are also utilised in the next section.

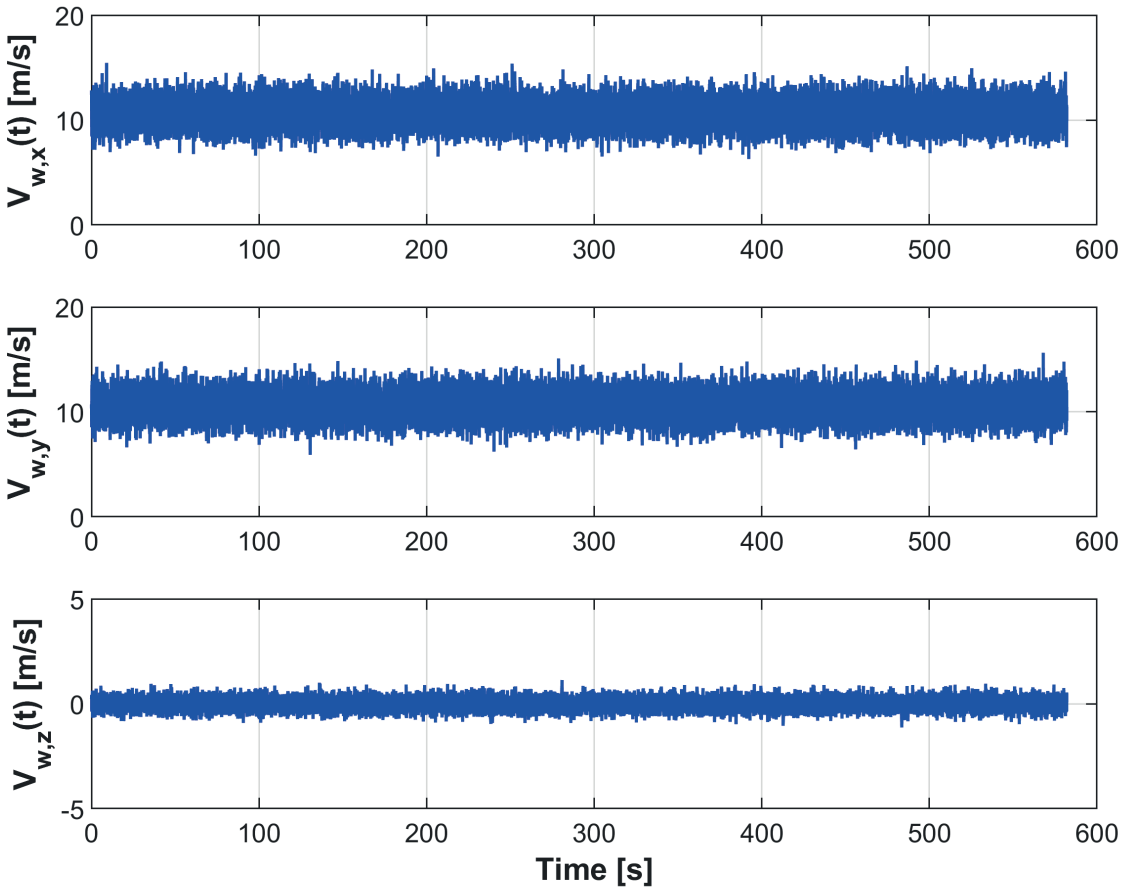

Figure 7.7: Wind velocities in each axis.

The speed and altitude parameters, shown in the figure 7.8, are a bit distinct from the ones of the previous figure 7.1. Firstly, the velocity tracking is conducted more unsatisfactorily in the MPC case (the reference is not followed as well as before, from time $t = 350s$), when the LQR results present lower deviations than in the previous case.

Secondly, attending to the reference altitude, it is followed more inadequately as well. Now, the LQR is not able to track the altitude for the descent properly, and the aircraft makes much more error; on the other hand, the MPC offers a more acceptable solution in this part of the path than in the previous case of the figure 7.1, although the error continues being a bit high. However, in the curves the LQR control gives a more favourable response, producing less error; but, in the moment before the dropping, the error in MPC is still lower.

On the basis of the results, the wind seems to affect positively some flight characteristics at different sections of the path in each case, but in general terms, it

contributes to worsen the reference tracking, above all in curves. The LQR gives a bit more adequate response than MPC one, since the error in velocity is comparable, but during turns, the error in altitude is acuter in the MPC results.

This lead to the fact that the weigh parameters selected for LQR control generates a robuster response when the aircraft faces perturbations, while the MPC is not able to manage appropriately the control actions to reduce the error, despite the fact that the MPC prediction capabilities make easier to find the best control procedure for each case, when the LQR does not optimise that control options (the gains are initially defined for the whole simulation).

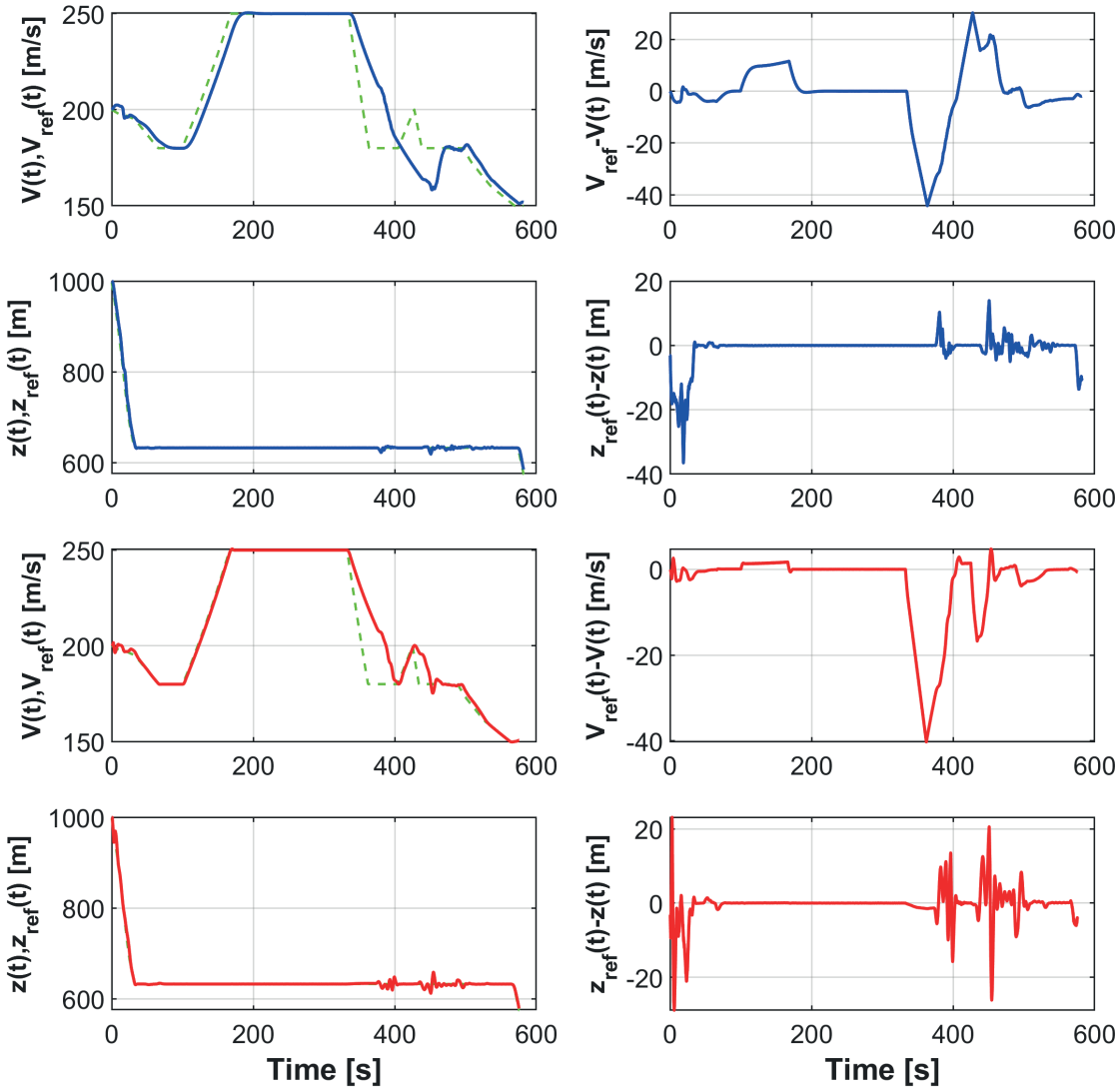

Figure 7.8: Reference tracking of velocity and altitude: LQR (blue), MPC (red), reference at each case (green, discontinuous).

In relation to the flight parameters (see figures 7.9 and 7.10), the fluctuations are increased in amplitude in all variables, which is normal due to the wind. However, in the case of the angle of attack, the number of times that it enters into the negative area is greater in both cases, enlarging the probability of stall in the real behaviour of the aircraft. The MPC controller could produce more failures in this sense.

With regard to the sideslip angle, the oscillations of the signal are greater than the ones in non-wind conditions and higher values are obtained. During the long straight section of the path, the signal oscillates with respect to a value distinct from zero, which demonstrates the impact of the wind speed over the aircraft motion.

Attending to the pitch angle and angular velocity q, the motion is more aggressive in the MPC case, because the elevator control action is still too demanding. However, the parameters that are fundamental in the XY plane motion (ϕ, ψ, p, r) present a bit lower fluctuations in the MPC case [2], so the guidance in (x,y) coordinates is slightly better than the LQR one.

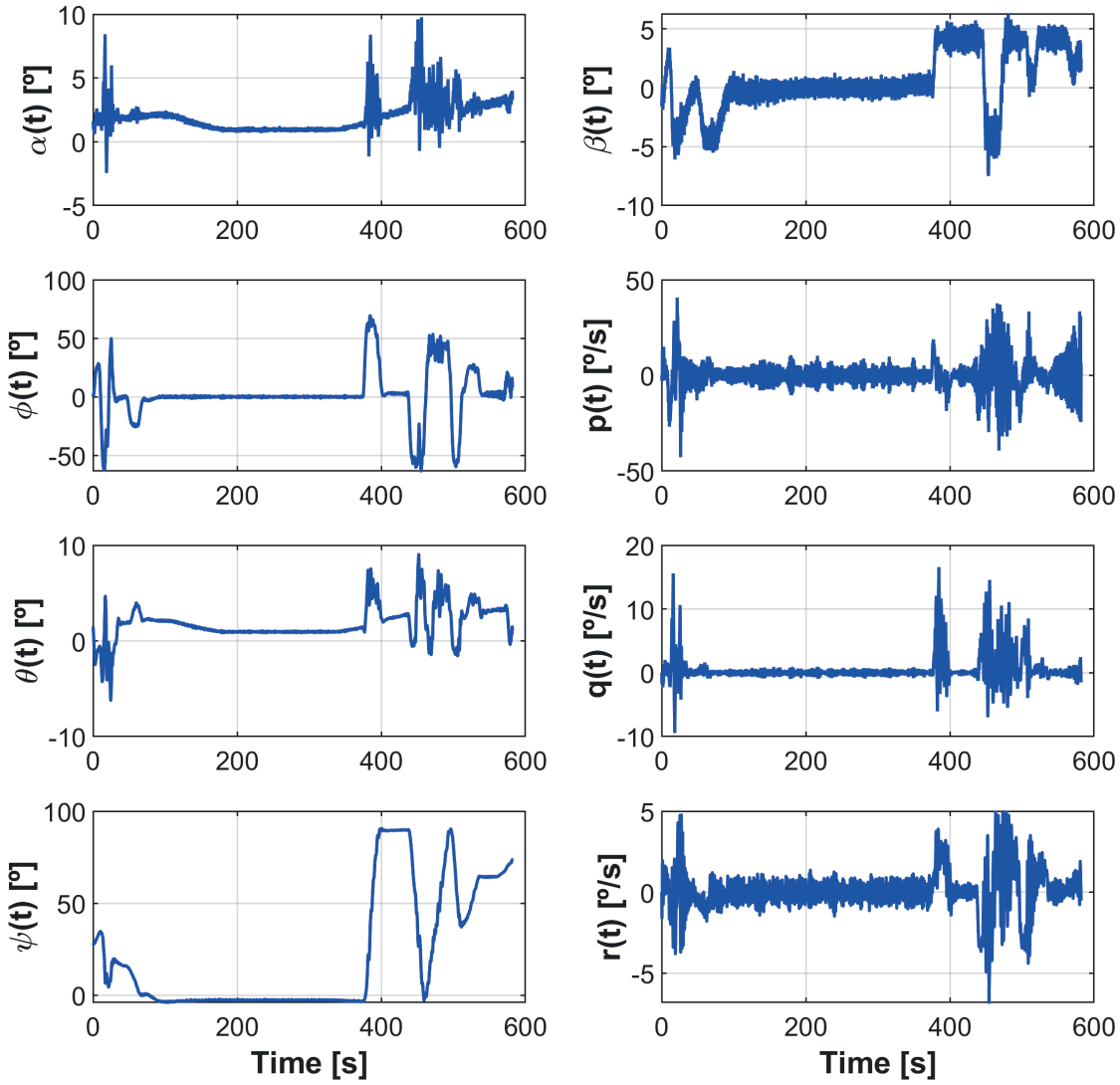

Figure 7.9: Principal flight parameters by LQR control.

[2] p and r oscillate with a lower amplitude.

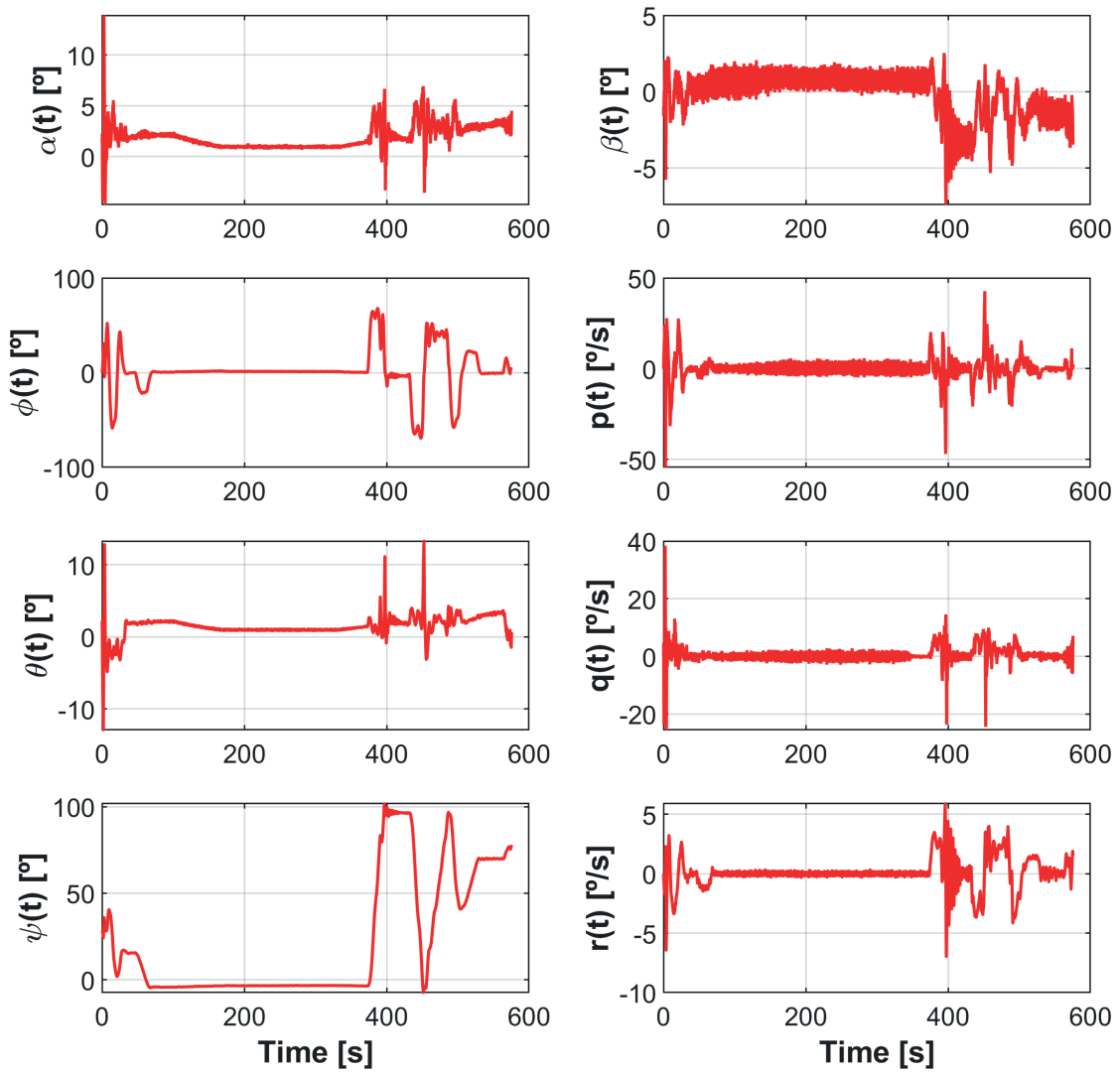

Figure 7.10: Principal flight parameters by MPC control.

In the image 7.11, the differences among control actions from LQR and MPC can be appreciated: in this case, the control signals are very similar among them, although the MPC ones present fluctuations with slightly greater amplitude and the LQR is more aggressive in the utilisation of the throttle lever (maximum thrust is achieved) and rudder.

Regarding the MPC results, for the descent, the controller diminishes the thrust in order to not depend on the elevator to reduce the velocity as much as before, but it is still a problem, therefore, the altitude tracking is not conducted properly during this stage of the trajectory.

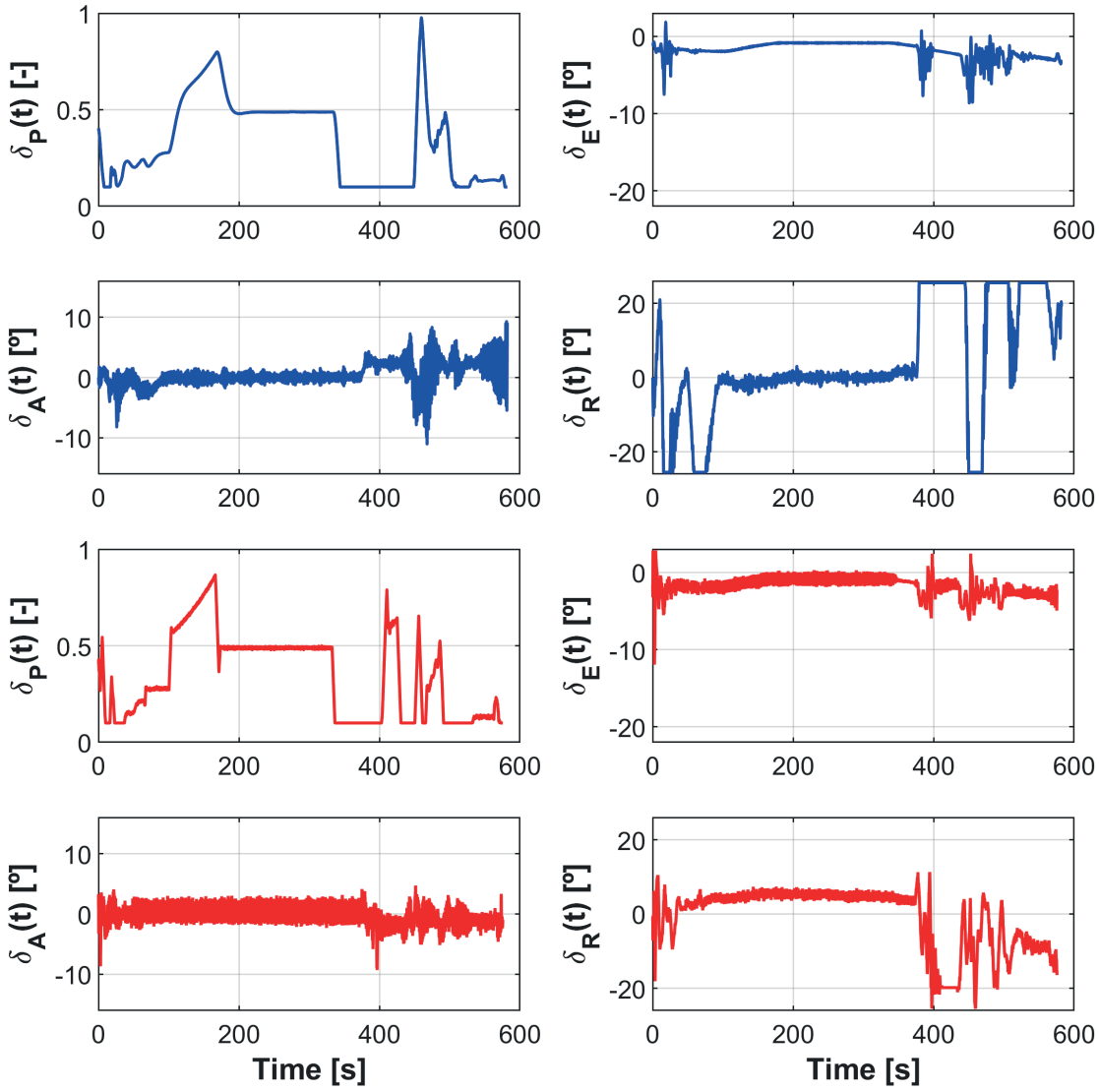

Figure 7.11: Control actions: LQR (blue), MPC (red).

In the figure 7.12, the trajectories show the fact that MPC altitude tracking is not as appropriate as the LQR one, obtaining higher fluctuations in the path. However, in this case the response from MPC has been more satisfactory in the initial descent than in the non-wind conditions. In relation to the reference tracking in the XY plane, greater deviations are observed due to the fact that the increment of the error in the reference altitude following complicates the tracking in x and y coordinates, since the elevator deflections intervenes in the lateral-directional motion and disrupt the task of the ailerons and rudder.

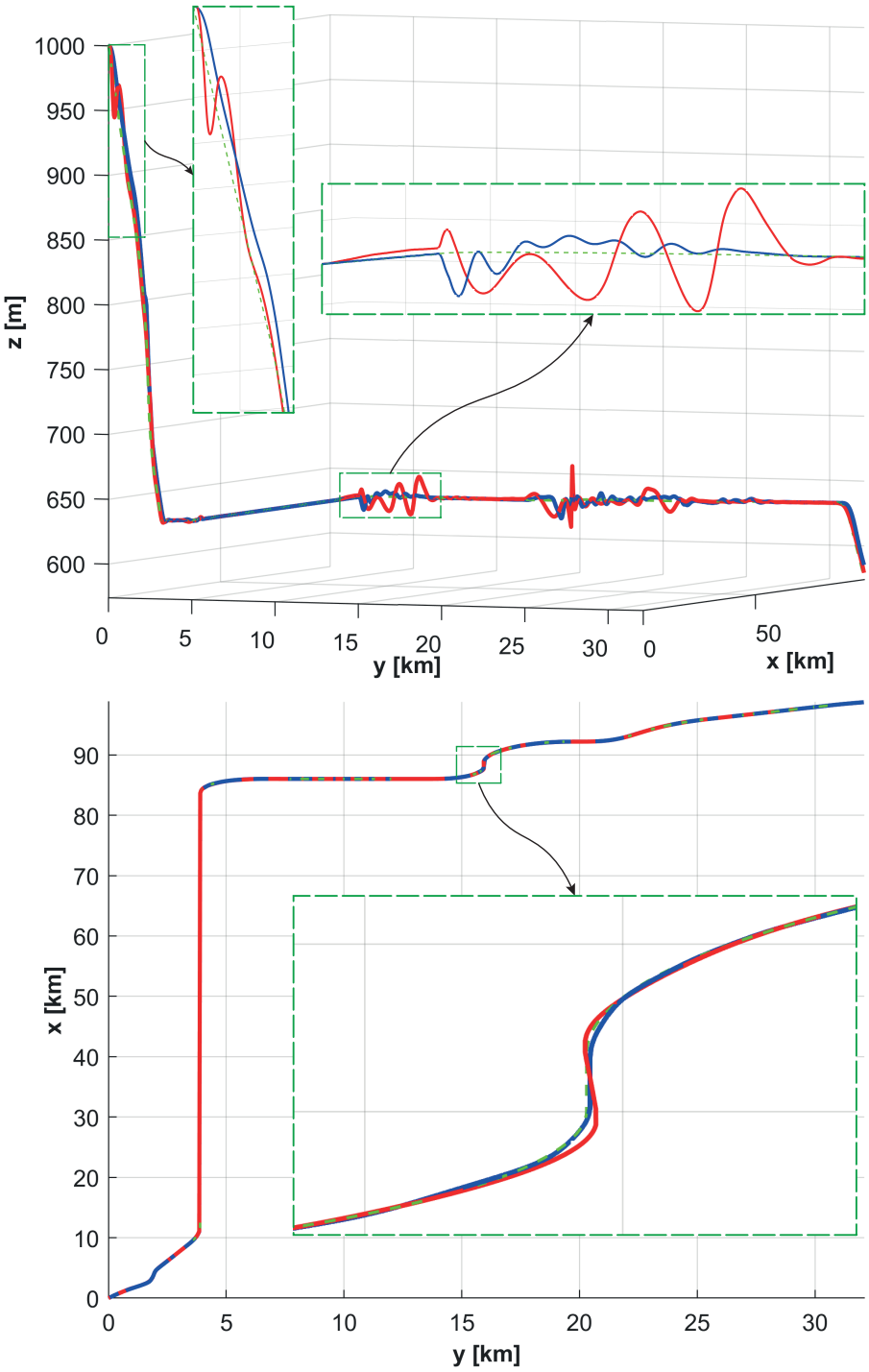

Figure 7.12: Final trajectory in two perspectives: LQR (blue), MPC (red), reference at each case (green, discontinuous).

In relation to the M117s flight parameters, shown in the graphs 7.13, the results are very similar to the ones obtained in the figure 7.6, although new oscillations are found in all variables, except for the altitude. The wind contributes to modify all the

variables and create different non-damped fluctuations. Now, the angular velocity r does not tend to zero, and the angular speed q is not the same in LQR and MPC cases. As before, the initial speed and altitude at the LQR case are greater, which will lead to a distinct optimal dropping point selection.

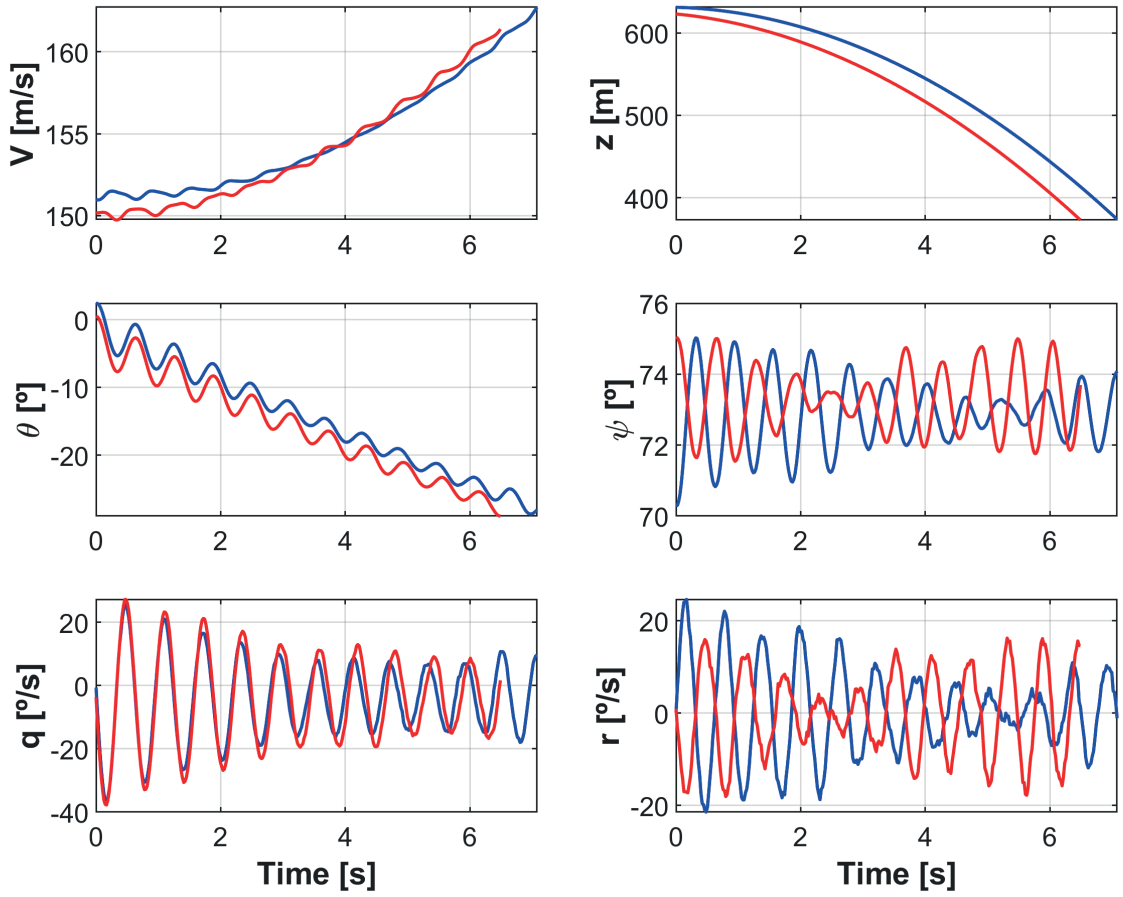

Figure 7.13: Trajectory of the M117 bombs from the optimal dropping point: LQR (blue), MPC (red).

7.3 Complete response: wind model and variable mass

In this third section, the complete response of the system is studied. It includes the wind model application and the fuel mass consumption along the time, so that it is the closest simulation to the real behaviour of the aeroplane. Nevertheless, neither compressibility effects are produced during the flight, so that only wind perturbations are given.

In the figure 7.14, the different fuel mass amount for the whole simulation time are shown. As it can be observed, and basing on the throttle lever actions (see figure 7.18), the rate of descent in fuel mass is greater at high speed sections of the path, since a higher thrust is needed to fulfil the requirements, while if areas with curves are traversed by the aircraft, the consumption is lower.

The fuel consumption is very similar between both simulations, although there is a small region in which both curves are distinct: in the area between time $t = 400s - 500s$, the reference velocity is achieved at the MPC case, when it is not at the LQR one (see figure 7.15).

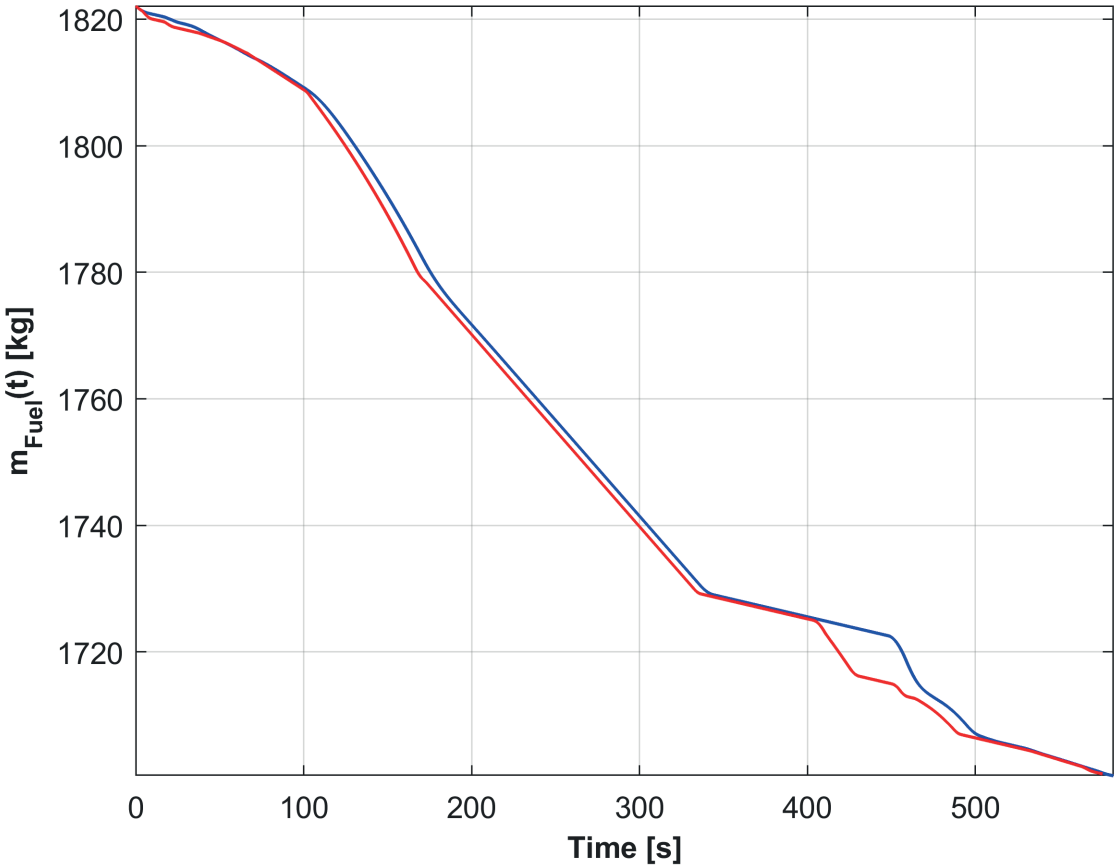

Figure 7.14: Fuel mass variation: LQR (blue), MPC (red).

The reference tracking of velocity and altitude, flight parameters and control actions from figures 7.15-7.18, offer similar results to the ones as the respective graphs from the previous section, although in this case, at the end of the path some differences can be found in most of the parameters. Due to the fact that the trajectory does not take a huge time to be completed, the mass variation is not very important and does not affect considerably to the dynamics of the system.

On the other hand, a much longer reference path could have been proposed in order to analyse how far the system could arrive without taking excessive errors, however, the large non-linearity of the mathematical model of the aeroplane makes impracticable this option, introducing many non-modelled perturbations in the system that the controller can not manage. In addition, the objective of this project is to build an application which is able to accomplish the proposed mission, so that determining the maximum mass consumption allowed is not the principal finality of the work.

In this way, this trajectory offers the possibility of observing the increment in fluctuations in all variables, seeing that the variation in mass is not taken into ac-

count in the linearization, and ensure that the aircraft follows all references without impermissible errors.

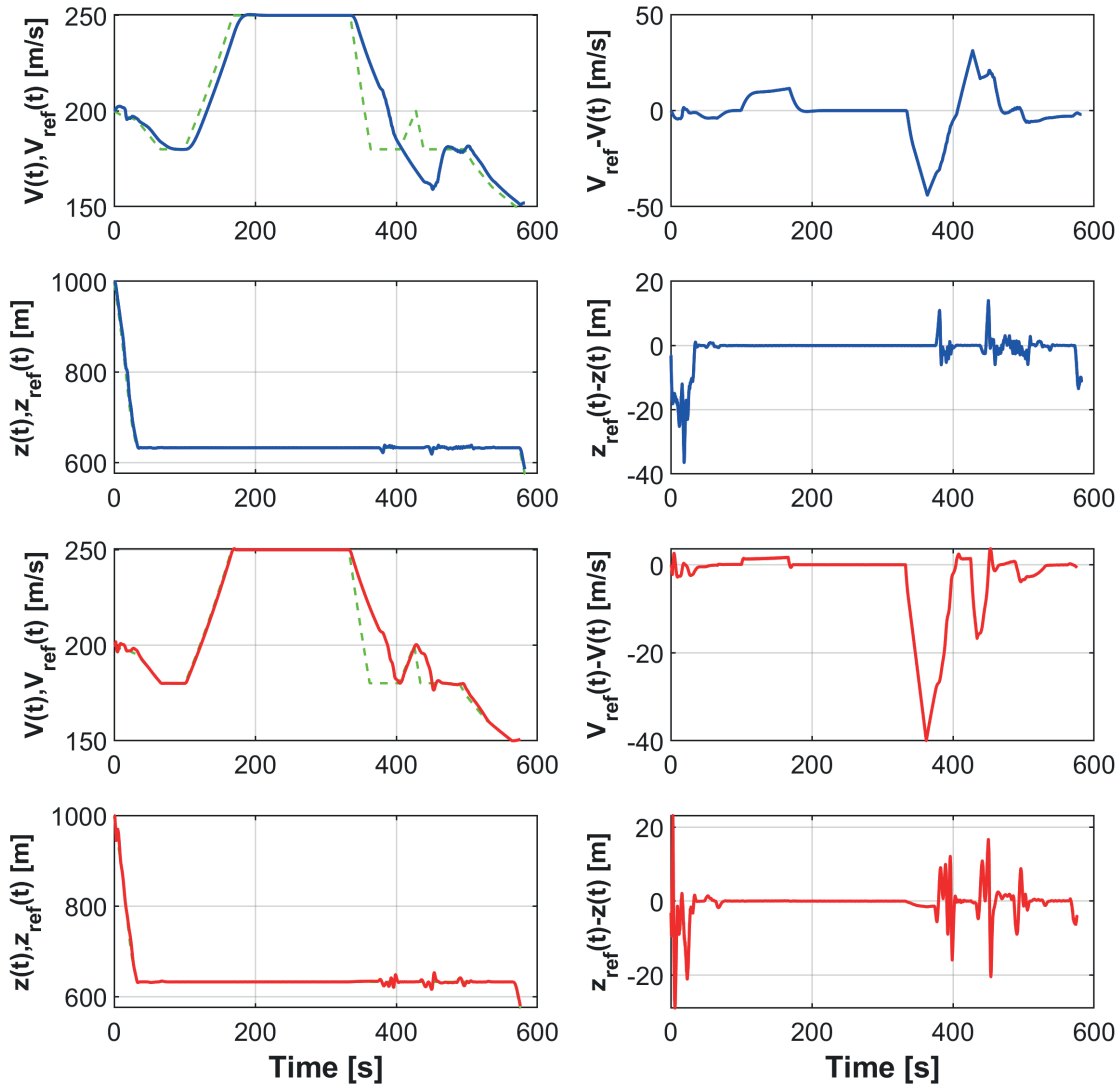

Figure 7.15: Reference tracking of velocity and altitude: LQR (blue), MPC (red), reference at each case (green, discontinuous).

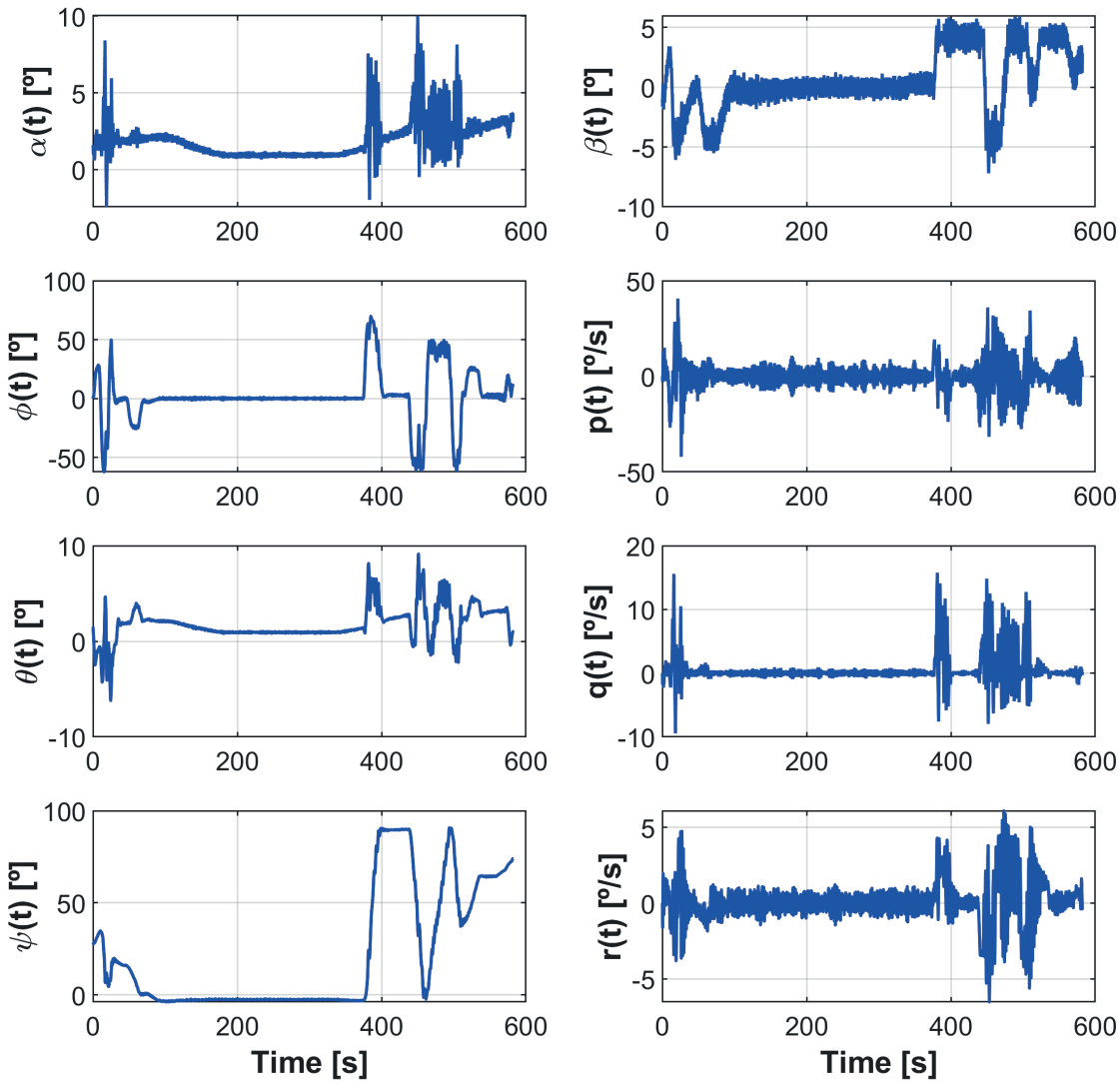

Figure 7.16: Principal flight parameters by LQR control.

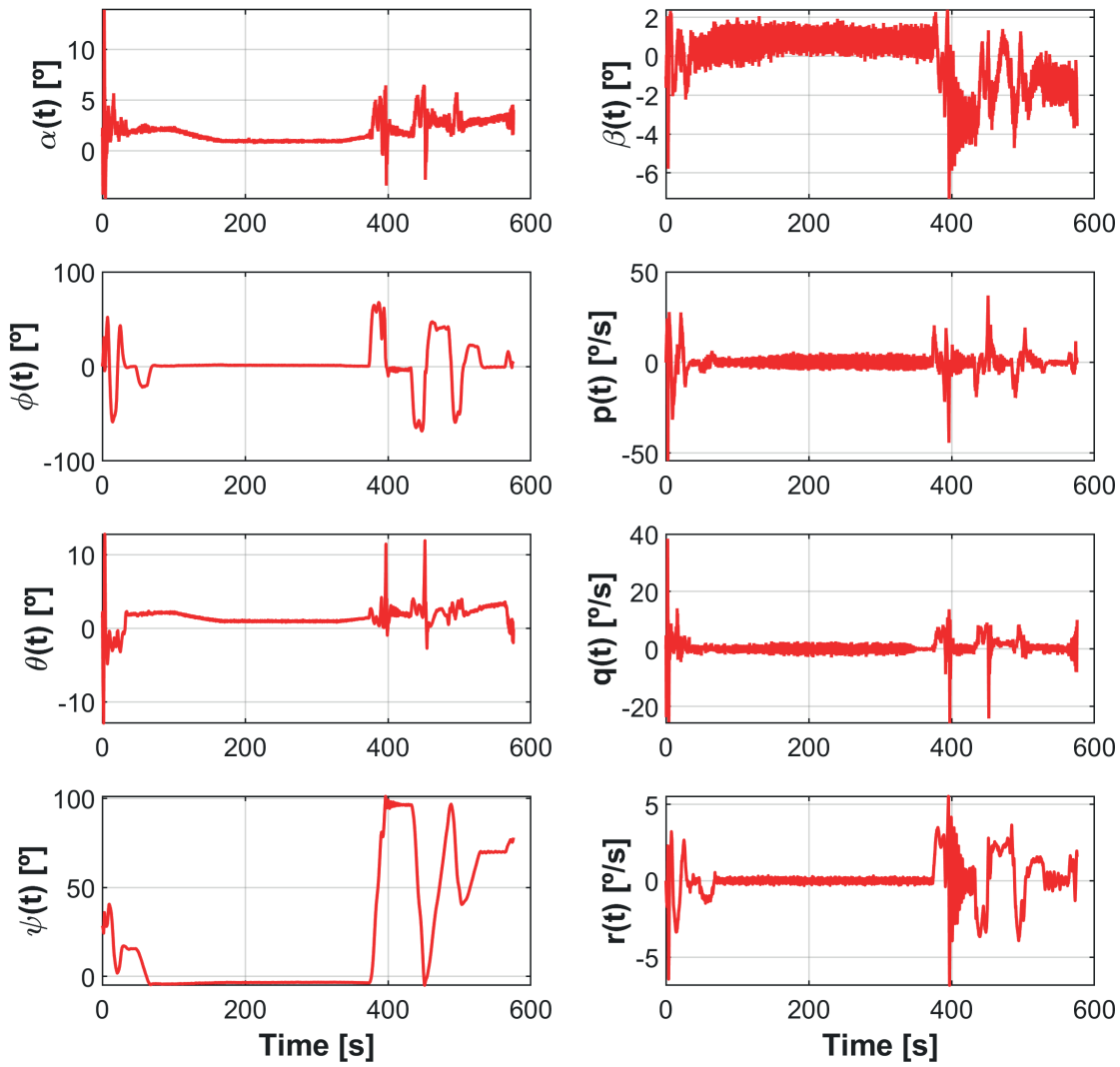

Figure 7.17: Principal flight parameters by MPC control.

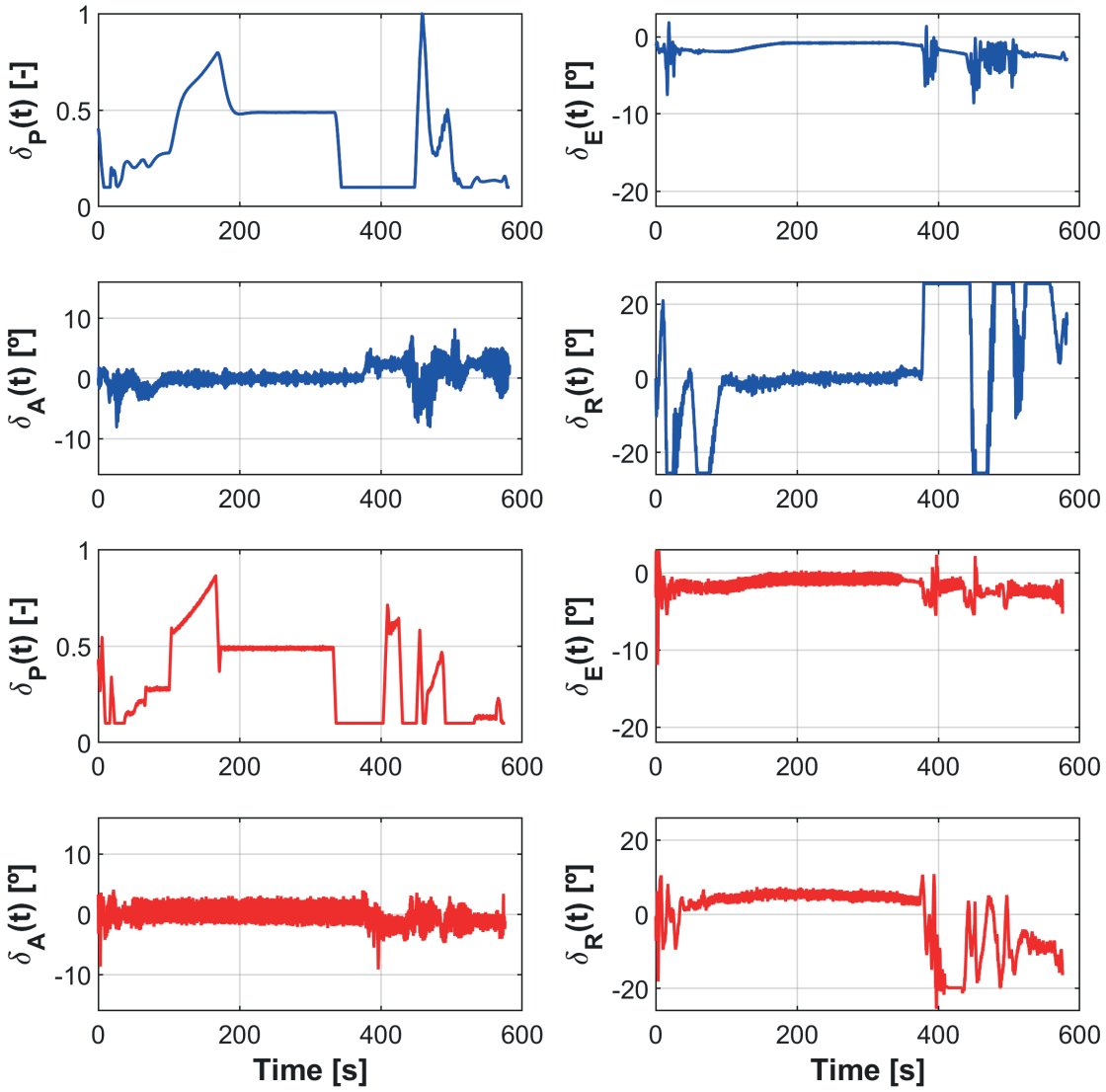

Figure 7.18: Control actions: LQR (blue), MPC (red).

The final trajectories utilising both control methods can be observed in the figure 7.19, which are very similar to the ones from figure 7.12, but including tiny variations at the end of the path.

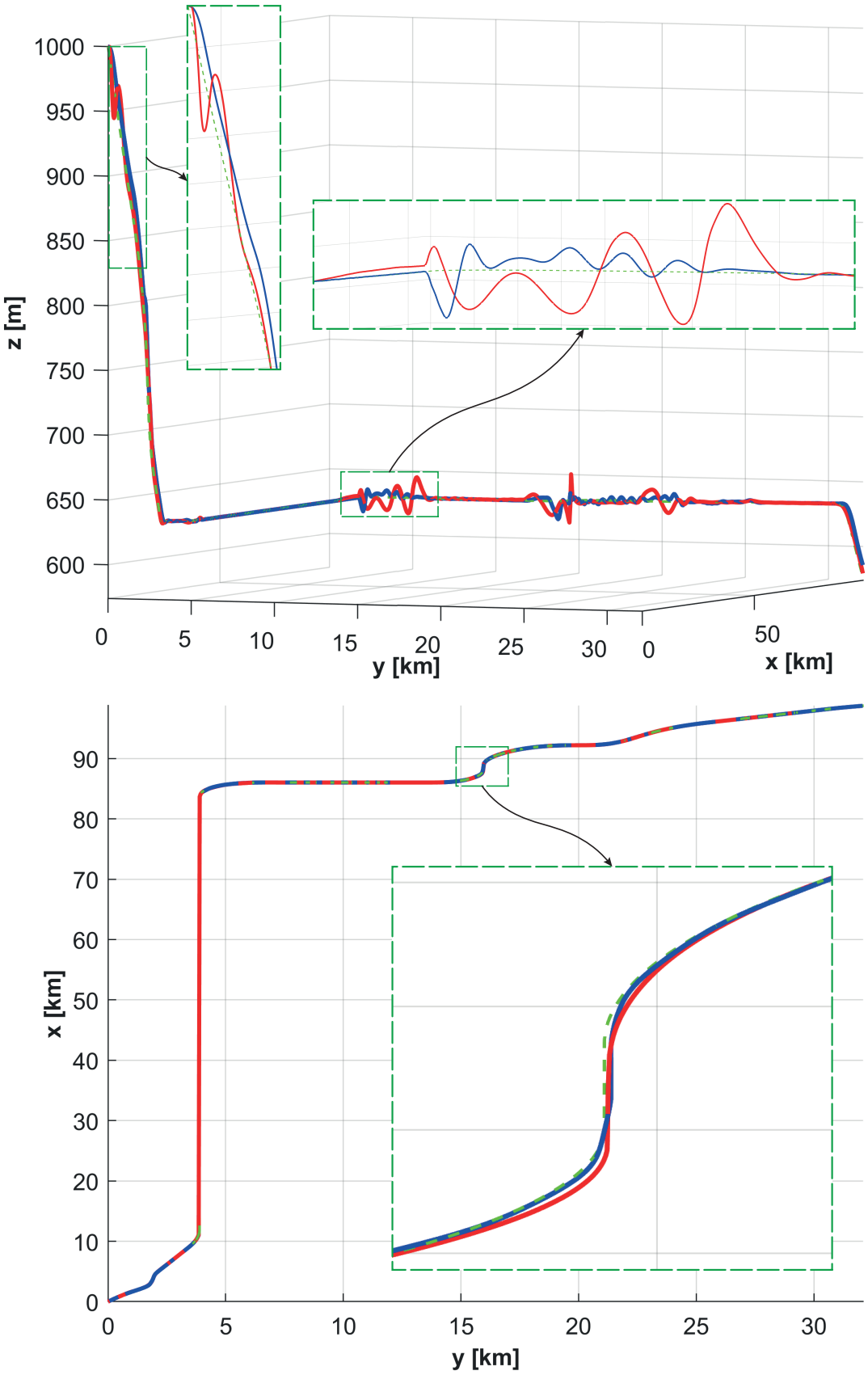

Figure 7.19: Final trajectory in two perspectives: LQR (blue), MPC (red), reference at each case (green, discontinuous).

This case, in which the wind model and the variable mass are taken into account, has been simulated in FlightGear, demonstrating that the aircraft acts as it was expected and there are no collisions with the ground throughout the flight, when LQR or MPC are utilised. In the figure 7.20, 8 different views have been taken at distinct parts of the flight: the first two pictures, correspond to the initial descent, at which the city of Seoul can be observed; the third and fourth ones, show the behaviour of the aircraft with the first mountains in the path, which are perfectly avoided at a very low distance to the ground; pictures 5,6 and 7 represent the attitude of the F-86 Sabre when it is turning at different curves, and finally, the last picture shows how the aircraft is approaching to Pyonggang, which is the objective to be bombarded.

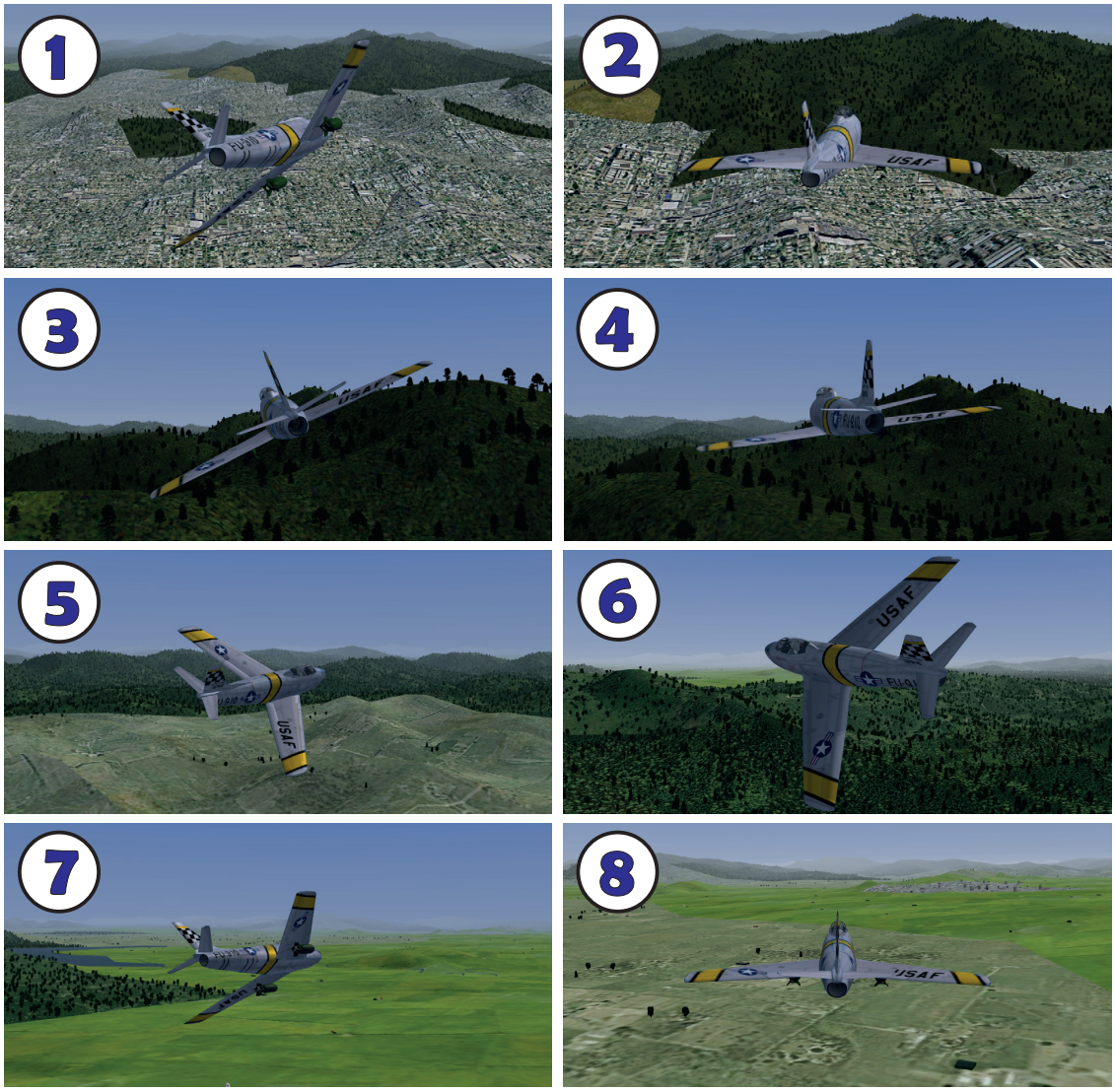

Figure 7.20: Different perspectives during the flight from FlightGear simulation.

Regarding the optimal trajectories of the M117 bombs, the results are practically identical to the ones from the graphs 7.13, since the dropping point and the flight conditions at each case are almost equal.

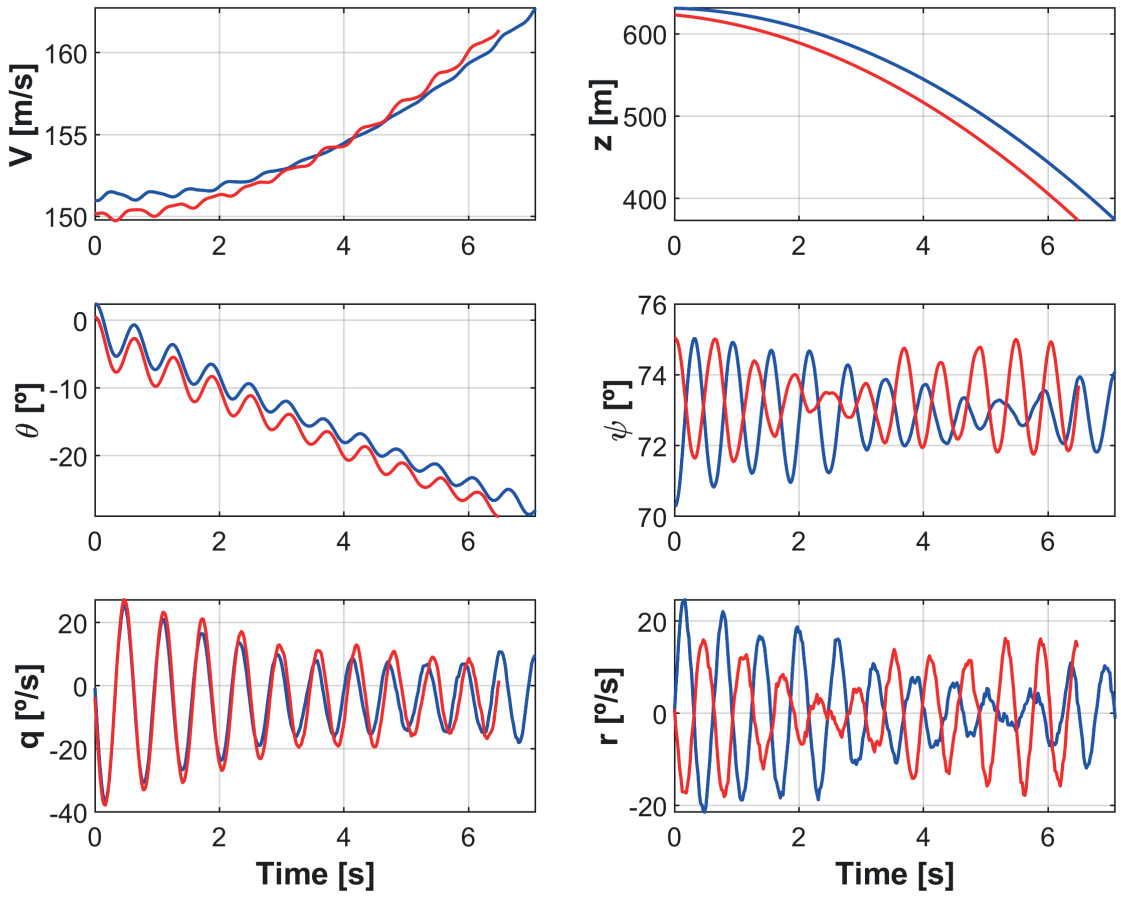

Figure 7.21: Trajectory of the M117 bombs from the optimal dropping point: LQR (blue), MPC (red).

7.4 Analysis of impact, time and mass

In the end, a study of the precision at the impact and the difference from the reference is conducted in order to determine the best control method for the mission in terms of accuracy. Moreover, the flight time and total mass consumption are analysed at each case.

The coordinates of the optimal dropping points and the impact ones are given by the table 7.1. In the table 7.2, the deviations in dropping with respect to the reference one and the exact deviation from the objective at each case are shown: in all cases, the MPC controller provides more adequate results in terms of accuracy and deviation from the reference release point, which is due to the fact that the reference tracking of the path in the XY plane and following of the reference velocity are conducted more adequately by means of the MPC controller. However, the differences among LQR and MPC results is not truly high in the impact, so that, despite the different initial dropping conditions, an optimal trajectory is found that reaches the objective with not too much error.

Coordinates			
Case		Optimal dropping point (x,y,z) (km)	Impact point (x,y) (km)
Reference		(98.444,31.207,0.621)	(98.761,32.113)
Non-wind, m_{cte}	LQR	(98.404,31.096,0.631)	(98.774,32.107)
	MPC	(98.434,31.180,0.625)	(98.768,32.107)
Wind, m_{cte}	LQR	(98.412,31.117,0.631)	(98.770,32.110)
	MPC	(98.440,31.203,0.623)	(98.759,32.108)
Wind, $m(t)$	LQR	(98.415,31.124,0.631)	(98.765,32.109)
	MPC	(98.435,31.204,0.623)	(98.760,32.108)

Table 7.1: Coordinates of the dropping and impact points.

Deviations			
Case		Deviation in dropping (m)	Deviation in impact (m)
Reference		-	7.59
Non-wind, m_{cte}	LQR	117.78	21.26
	MPC	28.96	15.05
Wind, m_{cte}	LQR	95.69	16.27
	MPC	6.42	6.95
Wind, $m(t)$	LQR	88.13	11.56
	MPC	5.57	7.84

Table 7.2: Deviations in the release (from reference case) and in the impact of the bombs.

But the issue comes when the real flight of the aircraft is conducted, because the aircraft can not throw the bomb at the optimal points of table 7.1, due to the fact that the aeroplane can face different non-planned conditions[3] each time and the coordinates do not longer serve: the aircraft needs to drop the bombs at the optimal reference point, or at least as closest as possible; consequently, the high deviations in the dropping points of LQR results from the reference one make the MPC controller the clear winner.

In addition, in this case the wind direction has resulted to have a positive impact over the bomb dropping and impact, reducing the errors. This could make non sense, since the higher the perturbations, the poorer the accuracy; nevertheless, the bomb trajectory is very sensitive to the initial conditions, so that the introduction of perturbations could have made possible that the dropping conditions are more favourable.

Finally, analysing the flight time and the fuel consumption of the complete response, given in the table 7.3, the MPC shows more favourable results as well, providing the best conditions for the mission, since it can be fulfilled in a shorter time and losing less fuel mass. Although, the fuel consumption is fairly similar in both cases and the difference in total time is less than 10 seconds; therefore, this fact is not as relevant as the dropping point deviation from the reference.

[3]For example, the wind model could be very different from the real wind conditions, or the temperature of the atmosphere is very distinct from the reference one.

Time and Fuel mass			
Case		Simulation time (s)	Fuel consumption (kg)
Non-wind, m_{cte}	LQR	583.128	-
	MPC	577.836	-
Wind, m_{cte}	LQR	582.015	-
	MPC	575.610	-
Wind, $m(t)$	LQR	582.099	121.671
	MPC	575.694	121.370

Table 7.3: Simulation time and fuel consumption.

8 | Conclusions

In this project, a complete autonomous navigation system has been designed, which is able to create a suitable trajectory for an F-86 Sabre between an initial point and the chosen destination and generate the required signals for the control surfaces and the throttle lever to conduct such path. The objectives consisted in achieving that the application built in this present work makes possible to bombard successfully an objective in a mission example, contained in the historical period of the Korean War, with two 750-lb Demolition Bombs M 117, while some imposed constraints or restrictions are avoided. Basing on the results obtained in the Chapters 6 and 7, it can be affirmed that the finality of this work has been fulfilled.

In the first place, a mathematical model has been created for the aircraft and the demolition bombs, which can provide the introduction of the allowed control actions and analyse the attitude and trajectory during the flight of both. In the case of the Sabre, the possible control actions have been simplified, such as the horizontal stabilizer deflection or the utilisation of air brakes during the flight, since they are considered as not necessary to fulfil the mission; regarding the bombs, no controls are used, so that a free-fall motion is obtained. All geometrical and physical parameters and motion equations have been extracted from specific bibliography and, each one of the surmises or hypothesis applied at the models has been substantiated on previous related works and documents as well; therefore, the mathematical models of the whole system (aircraft and bombs) and the individual one of the bombs have been built according to their real behaviours, or at least, as closer as possible to these ones. Furthermore, a wind model has been generated, which attempts to adapt to the real wind behaviour at the Korean peninsula and at the flight altitudes of the mission.

Afterwards, an study of different control methods is accomplished, from which the two best options in terms of quality of results and complexity are selected: Linear Quadratic Regulator (LQR) and Linear Time-Invariant Model Predictive Control (MPC). The LQR controller is designed by means of the division of the system into four distinct subsystems, offering a more adequate control solution than whether the whole system is analysed, while in the MPC case, no subdivisions are necessary and the complete model is inserted into the controller, making the control design more comfortable. Both control methods have given satisfactory solutions in relation to reference tracking and the mission finality of bombarding the objective, although some differences are found.

In usual conditions, LQR provides a robust response and makes the system fairly stable, due to the fact that the optimal gains for the closed loop are calculated

from the beginning; while a Linear Time-Invariant MPC controller is able to predict the future events at each iteration, finding the best control option at each case and, not setting one initial gain from the start as LQR does, so that normally the system behaves more satisfactorily in terms of aggressiveness whether it is needed and robustness when it faces perturbations.

However, these methods are prepared for systems with a high linearity and conditions in which the variables do not vary excessively from the equilibrium point; so that, in this case, when the mathematical model of the aircraft is very non-linear and it must conduct curves at important load factors, a lot of non-modelled perturbations come to the controllers and their effectiveness is in danger. In addition, LQR and MPC are very sensitive to the selection of weigh parameters, giving as a result unreasonable control signals that lead to instability or poor control actions that do not produce an adequate following of the reference variables. Hence, since the problem is very complex and many combinations of weigh parameters can be conducted, their selection can be easily wrongly done, or at least not as adequate as possible, creating the possibility that LQR could give more acceptable results than MPC, although it is contrary to what was expected.

In this way, in the final simulation with wind model and variable fuel mass, the LQR control strategy has resulted in a more adequate response of the system when it comes to reference trajectory tracking, though the reference velocity is followed slightly better in the MPC case. Therefore, LQR has demonstrated greater robustness in the presence of perturbations, above all in curves, when the model becomes more non-linear and the motion is not longer in the vertical plane. However, none of these controllers have provoked that the aircraft enters into restricted areas and have achieved the correct guidance of the aircraft to the destination, in order to the bombs impact properly. Consequently, both control methods could theoretically complete the mission objectives.

Nevertheless, despite the fact that LQR has given a more favourable response of the system, regarding the results of deviations from reference dropping point, impacts, flight time and fuel mass consumption lead to the MPC as the best option. In all cases, the bombs released in LQR cases are released from the reference point further than in MPC ones, producing a higher error in the impact as well. Therefore, since the obstacle imposed are respected in both cases of control method and the dynamic path is followed (although with some errors), the MPC would be more suitable in the case of application to the aircraft's system.

In order to obtain improved results in MPC cases, a Linear Adaptive MPC should have been utilised instead, giving the opportunity of introducing a clearer idea about real non-linear behaviour of the model to the controller. Though, this method is more complex to implement than the Time-invariant option, so that this second one has been preferred in order to make manageable the difficulty of the problem.

In addition, it is important to mention the noticeable fluctuations in the control action signals, which could lead to structural fatigues in the engine and control surfaces and mechanical problems in the real mission. Besides, negative angles of attack are achieved along the simulations with both controllers, giving the opportunity to the aircraft of entering into stall regime. Therefore, an optimisation of these

controllers should be conducted in this sense in order to avoid this kind of issues.

Furthermore, the guidance algorithm is very relevant over the results obtained, because if a classical proportional guidance law had been employed instead of one based on forces, the trajectories would have been much more oscillatory or even could have lead to a failure. Consequently, the success in the simulation is also a prove that the guidance algorithm is correctly implemented and the chosen one works.

Finally, the different algorithms used for the dynamic path generation have provided an appropriate reference trajectory and velocity profile that fully respects the restrictions of the proposed mission and affords acceptable reference parameters for the aircraft to follow. Nevertheless, the configuration of the RR-ACD algorithm needs to set some weigh parameters for the reward function, which, in some cases, it makes hard to find an adequate solution, needing many attempts until achieve such goal. Moreover, neither the optimal solution is found, since it is not the shortest possible distance that fulfils the physical and dynamic constraints: in order to get the best dynamic path option, another algorithm could have been implemented to find the optimal weigh parameters configuration and some variation in the neighbourhood function should be introduced to facilitate to the RR-ACD code more appropriate next available waypoints. Regarding the chosen velocity profile, a shorter range of velocities should have been selected, since the controllers are not able to follow the reference speed in many cases and sometimes, the aircraft can not decrease the velocity so fast even with ideal control actions.

However, despite the limitations of the algorithm, the code gives a satisfactory solution and fulfils successfully the requirements of this project.

Bibliography

[1] Federation of American Scientists. *NORTH AMERICAN F-86H Sabre*. Dec. 1999. URL: http://www.fas.org/nuke/guide/usa/airdef/f-86h.htm.

[2] Cameron Beccario. *Earth Null School*. 2020. URL: https://earth.nullschool.net/.

[3] The Super Computing Blog. *Coding Bilinear Interpolation*. 2011. URL: http://supercomputingblog.com/graphics/coding-bilinear-interpolation/.

[4] Javier Carballeira Morado and Vicente Mata Amela. "Rigid Body Kinematics". In: *Mechanics*. 2019, pp. 1–70.

[5] History.com Editors. *Korean War*. May 2020. URL: https://www.history.com/topics/korea/korean-war.

[6] Eugene L Fleeman. *Tactical Missile Design Second Edition*. 2006, pp. 21–80.

[7] Flylib.com. *F-86 v. MiG-15 and Boyd s OODA Loop*. 2005. URL: https://flylib.com/books/en/4.536.1.72/1/.

[8] A. García Pérez. *Estadística Básica*. Ed. by UNED. Spain: Grafo, S.A., 2013, pp. 122–126.

[9] Garg Gp Capt Atul, Linda Rezawana Islam, and Chowdhury Tonoy. "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).

[10] Francisco Rafael Gavilán Jiménez. "Sistemas de Control y Guiado para Vehículos Aéreos No Tripulados". PhD thesis. Universidad de Sevilla, 2012, pp. 89–164.

[11] N. Leonard Wener. *Measurement of aircraft moments of inertia*. Paris: Armed services technical information agency arlington hall station arlington 12, virginia, 1959, pp. 1–11.

[12] M. R. Napolitano. *Aircraft Dynamics: From Modeling to Simulation*. John Wiley & Sons, Inc., 2012, pp. 135–267. ISBN: 9780470626672.

[13] José Pedro Magraner Rullan. "Ecuaciones de la dinámica de un avión rígido". In: *Universitat Politècnica de València* 01 (2020), pp. 1–57.

[14] José Pedro Magraner Rullan, Marcos Carreres Talens, and Pedro Martí Gómez-Aldaraví. "CD0 estimation (low subsonic)". In: *Universitat Politècnica de València* (2019), pp. 1–13.

[15] José Pedro Magraner Rullan, Marcos Carreres Talens, and Pedro Martí Gómez-Aldaraví. *Flight Mechanics*. Universitat Politècnica de València, 2018, pp. 1–190. DOI: 10.1007/978-981-10-8721-9_2.

[16] MathWorks. *lqr*. 2020. URL: https://www.mathworks.com/help/control/ref/lqr.html.

[17] MathWorks. *Understanding Model Predictive Control, Part 4: Adaptive, Gain-Scheduled, and Nonlinear MPC*. 2020. URL: https://www.mathworks.com/videos/understanding-model-predictive-control-part-4-adaptive-gain-scheduled-and-nonlinear-mpc-1530606851674.html.

[18] meteoblue. *Climate Seoul*. 2020. URL: https://www.meteoblue.com/en/weather/historyclimate/climatemodelled/seoul_republic-of-korea_1835848.

[19] Ian Moir, Allan Seabridge, and Malcolm Jukes. *Civil Avionics Systems*. 2013, pp. 365–404. ISBN: 9781118536704. DOI: 10.1002/9781118536704.

[20] "4 - Overview of Digital Signal Processing Algorithms". In: *DSP Software Development Techniques for Embedded and Real-Time Systems*. Ed. by Robert Oshana. Embedded Technology. Burlington: Newnes, 2006, pp. 59–121. ISBN: 978-0-7506-7759-2. DOI: https://doi.org/10.1016/B978-075067759-2/50006-5.

[21] Michael Peck. *Jet Fighter Death Match: Russia's MiG-15 vs. America's F-86 Sabre (Who Wins?)* Oct. 2019. URL: https://nationalinterest.org/blog/buzz/jet-fighter-death-match-russias-mig-15-vs-americas-f-86-sabre-who-wins-88101.

[22] James P. Pegram and William A. Anemaat. "Preliminary estimation of airplane moments of inertia using CAD solid modeling". In: *SAE Technical Papers* May 2000 (2000). ISSN: 26883627. DOI: 10.4271/2000-01-1700.

[23] Picuino. *Controlador PID*. 2020. URL: https://www.picuino.com/es/arduprog/control-pid.html.

[24] Quora. *Was the Mig 15 design a copy of the f86 sabre?* Jan. 2016. URL: https://www.quora.com/Was-the-Mig-15-design-a-copy-of-the-f86-sabre.

[25] RC4ever. *FLY BY WIRE*. 2020. URL: http://airguardian.net/projects/fly-by-wire-project/.

[26] Élodie Roux. *Turbofan and Turbojet Engines Database Handbook*. Éditions Élodie Roux, 2007, p. 242. ISBN: 9782952938006.

[27] Franklin Samaniego et al. "Recursive rewarding modified adaptive cell decomposition (RR-MACD): A dynamic path planning algorithm for UAVs". In: *Electronics (Switzerland)* 8.3 (2019), pp. 1–21. ISSN: 20799292. DOI: 10.3390/electronics8030306.

[28] Franklin Samaniego et al. "Smooth 3d path planning by means of multiobjective optimization for fixed-wing UAVs". In: *Electronics (Switzerland)* 9.1 (2020). ISSN: 20799292. DOI: 10.3390/electronics9010051.

[29] H. Samet and A. Kochut. "Octree approximation an compression methods". In: *Computer Society* (2002), pp. 1–9. DOI: 10.1109/TDPVT.2002.1024101.

[30] Adam Schneider. *GPSVisualizer*. 2019. URL: https://www.gpsvisualizer.com/elevation.

[31] John Snyder. *Great Circles on Mercator's Chart*. 2007. URL: http://demonstrations.wolfram.com/GreatCirclesOnMercatorsChart/.

[32] P. B. Sujit, Srikanth Saripalli, and J. B. Sousa. "An evaluation of UAV path following algorithms". In: *2013 European Control Conference, ECC 2013* 1 (2013), pp. 1–6. DOI: 10.23919/ecc.2013.6669680.

[33] USAF. "F-86F-1 thru -20". In: *Standard aircraft characteristics*. 1956, pp. 1–6.

[34] USAF. *Flight manual USAF series F-86F aircraft*. 1960, pp. 177–192.

[35] USAF. *Handbook maintenance instructions F-86F*. 1954, pp. 1–575.

[36] Wikipedia. *Battle of White Horse*. 2020. URL: https://en.wikipedia.org/wiki/Battle_of_White_Horse.

[37] Wikipedia. *Great-circle distance*. 2020. URL: https://en.wikipedia.org/wiki/Great-circle_distance.

[38] Wikipedia. *Korean War*. June 2020. URL: https://en.wikipedia.org/wiki/Korean_War.

[39] Wikipedia. *M117 bomb*. Mar. 2020. URL: https://en.wikipedia.org/wiki/M117_bomb.

[40] Wikipedia. *Yeouido Airport*. 2020. URL: https://en.wikipedia.org/wiki/Yeouido_Airport.

[41] F. S. Withington. "Op 1280". In: *Navy Department Bureau of Ordenance Washington 25, D.C.* 1280.August (1955), pp. 3–10, 95–99.

[42] Kemin Zhou and John C. Doyle. *Essentials of Robust Control*. Ed. by Pearson Education. Vol. 8. 3. Prentice Hall, 1999, pp. 1–10.

A | Model system parameters

A.1 Non-linear dynamic model of the complete system

A.1.1 Non-dimensional radii of gyration

Radii of gyration					
$R_x(-)$	0.322	$R_y(-)$	0.346	$R_z(-)$	0.464

Table A.1: Radii of gyration used in the inertia matrix calculation of the aircraft.

A.1.2 Mass centers and moments of inertia of the bomb

$$x_{CG_{cone}} = \frac{3l_n}{4} = 0.381m \tag{A.1a}$$

$$x_{CG_{cyl}} = \frac{l_{b_1} + l_n}{2} = 0.838m \tag{A.1b}$$

$$\begin{aligned} x_{CG_{tcone}} &= l_{b_1} + \frac{\frac{(d_b - d_{b_b})(l_b - l_{b_1})}{4}\frac{(l_b - l_{b_1})}{3} + \frac{d_b}{2}(l_b - l_{b_1})\frac{(l_b - l_{b_1})}{2}}{\frac{(d_b - d_{b_b})(l_b - l_{b_1})}{4} + \frac{d_b}{2}(l_b - l_{b_1})} = \\ &= l_{b_1} + \frac{\frac{(d_b - d_{b_b})(l_b - l_{b_1})}{3} + d_b(l_b - l_{b_1})}{(3d_b - d_{b_b})} = 1.607m \end{aligned} \tag{A.1c}$$

$$x_{CG_b} = \frac{x_{CG_{cone}} m_{cone} + x_{CG_{cyl}} m_{cyl} + x_{CG_{tcone}} m_{tcone}}{m_b} = \mathbf{0.784\ m} \tag{A.1d}$$

$$I_{cone} = \begin{pmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{pmatrix} = m_{cone} \begin{pmatrix} \frac{3d_b^2}{40} & 0 & 0 \\ 0 & \frac{3d_b^2}{80} + \frac{3}{80}l_n^2 & 0 \\ 0 & 0 & \frac{3d_b^2}{80} + \frac{3}{80}l_n^2 \end{pmatrix} = \\ = \begin{pmatrix} 0.954 & 0 & 0 \\ 0 & 1.213 & 0 \\ 0 & 0 & 1.213 \end{pmatrix} kg \cdot m^2 \tag{A.2a}$$

$$I_{cyl} = \begin{pmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{pmatrix} = m_{cyl} \begin{pmatrix} \frac{d_b^2}{8} & 0 & 0 \\ 0 & \frac{d_b^2}{16} + \frac{1}{12}(l_{b_1} - l_n)^2 & 0 \\ 0 & 0 & \frac{d_b^2}{16} + \frac{1}{12}(l_{b_1} - l_n)^2 \end{pmatrix} = \\ = \begin{pmatrix} 5.571 & 0 & 0 \\ 0 & 12.457 & 0 \\ 0 & 0 & 12.457 \end{pmatrix} kg \cdot m^2 \tag{A.2b}$$

$$r(x) = \frac{(d_b - d_{b_b})}{2(l_b - l_{b_1})}x - \frac{(d_b - d_{b_b})(x_{CG_{tcone}} - l_b)}{2(l_b - l_{b_1})} + \frac{d_{b_b}}{2} \Rightarrow \\ \begin{cases} I_{x,tcone} = \frac{3m}{(l_b - l_{b_1})(d_b^2 + d_{b_b}^2 + d_b d_{b_b})} \int_{(x_{CG_{tcone}} - l_b)}^{(x_{CG_{tcone}} - l_{b_1})} (x^2(r(x)^2 + \frac{1}{2}r(x)^4))dx \\ I_{y,tcone} = \frac{3m}{(l_b - l_{b_1})(d_b^2 + d_{b_b}^2 + d_b d_{b_b})} \int_{(x_{CG_{tcone}} - l_b)}^{(x_{CG_{tcone}} - l_{b_1})} (x^2(r(x)^2 + \frac{1}{4}r(x)^4))dx \end{cases} \tag{A.2c}$$

$$I_{tcone} = \begin{pmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{pmatrix} = \begin{pmatrix} 0.419 & 0 & 0 \\ 0 & 0.387 & 0 \\ 0 & 0 & 0.387 \end{pmatrix} kg \cdot m^2 \tag{A.2d}$$

$$\sum_{i=1}^{3} m_i(x_{CG_b} - x_{CG_i})^2 = m_{cone}(x_{CG_b} - x_{CG_{cone}})^2 + m_{cyl}(x_{CG_b} - x_{CG_{cyl}})^2 + \\ + m_{tcone}(x_{CG_b} - x_{CG_{tcone}})^2 = 26.646 kg \cdot m^2 \tag{A.2e}$$

$$I_b = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \sum_{i=1}^{3} m_i(x_{CG_b} - x_{CG_i})^2 & 0 \\ 0 & 0 & \sum_{i=1}^{3} m_i(x_{CG_b} - x_{CG_i})^2 \end{pmatrix} kg \cdot m^2 \\ + I_{cone} + I_{cyl} + I_{tcone} = \begin{pmatrix} \mathbf{6.944} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{40.703} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{40.703} \end{pmatrix} \boldsymbol{kg \cdot m^2} \tag{A.2f}$$

A.1.3 Mass, mass center and inertia matrix of the complete system

$$m_0 = m_{a_0} + 2(m_b + m_r) = \mathbf{7578.21}\boldsymbol{kg} \tag{A.3a}$$

$$x_{CG_0} = \frac{x_{CG_a} m_{a_0} + 2x_{CG_{ba},x} m_b + 2x_{CG_r,x} m_r}{m} = \mathbf{2.50}\boldsymbol{m} \tag{A.3b}$$

$$\begin{aligned} I_0 = I_a + 2I_b + 2m_b \begin{pmatrix} x_{CG_{ba},y}^2 + x_{CG_{ba},z}^2 & 0 & 0 \\ 0 & (x_{CG_0} - x_{CG_{ba},x})^2 & 0 \\ 0 & 0 & (x_{CG_0} - x_{CG_{ba},x})^2 \end{pmatrix} + \\ + 2m_r \begin{pmatrix} x_{CG_r,y}^2 + x_{CG_r,z}^2 & 0 & 0 \\ 0 & (x_{CG_0} - x_{CG_r,x})^2 & 0 \\ 0 & 0 & (x_{CG_0} - x_{CG_r,x})^2 \end{pmatrix} = \\ = \begin{pmatrix} 3.316m_a + 3703.73 & 0 & 0 \\ 0 & 3.910m_a + 87.18 & 0 \\ 0 & 0 & 27.833m_a + 87.18 \end{pmatrix} kg \cdot m^2 = \\ \begin{pmatrix} \mathbf{26050.35} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{26436.79} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{187654.60} \end{pmatrix} \boldsymbol{kg \cdot m^2} \end{aligned} \tag{A.3c}$$

A.2 Linearized model of the complete system

$$A \cdot \Delta x = \begin{pmatrix} 0 & 2.19 \cdot 10^{-2} & 0 & -1.00 & 0 & 215 & 0 & 0 & 0 & 0 & 0 \\ -4.60 \cdot 10^{-5} & -1.10 \cdot 10^{-2} & 0 & 5.41 \cdot 10^{-2} & 0 & -9.81 & 0 & 0 & -4.71 & 0 & 0 \\ 0 & 0 & -1.88 \cdot 10^{-1} & 0 & 9.81 & 0 & 0 & 4.44 & 0 & -214.02 & 0 \\ 9.17 \cdot 10^{-4} & -5.80 \cdot 10^{-2} & 0 & -2.22 & 0 & -2.15 \cdot 10^{-1} & 0 & 0 & 212.88 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.00 & 0 & 2.19 \cdot 10^{-2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.00 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.00 & 0 \\ 0 & 0 & -1.75 \cdot 10^{-1} & 0 & 0 & 0 & 0 & -2.36 & 0 & 7.62 \cdot 10^{-1} & 0 \\ -3.22 \cdot 10^{-6} & 6.00 \cdot 10^{-3} & 0 & -2.26 \cdot 10^{-1} & 0 & -2.42 \cdot 10^{-19} & 0 & 0 & -3.27 & 0 & 0 \\ 0 & 0 & 3.32 \cdot 10^{-2} & 0 & 0 & 0 & 0 & -6.20 \cdot 10^{-3} & 0 & -2.10 \cdot 10^{-1} & 0 \\ -2.58 \cdot 10^{-5} & 1.12 \cdot 10^{-2} & 0 & 5.30 \cdot 10^{-3} & 0 & -9.81 & 0 & 0 & -3.72 \cdot 10^{-2} & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta z \\ \Delta u \\ \Delta v \\ \Delta w \\ \Delta \phi \\ \Delta \theta \\ \Delta \psi \\ \Delta p \\ \Delta q \\ \Delta r \\ \Delta V \end{pmatrix} \tag{A.4}$$

$$B \cdot \Delta u = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3.166 & 1.40 \cdot 10^{-1} & 0 & 0 & 8.70 \cdot 10^{-3} & -4.70 \cdot 10^{-3} & -5.43 \cdot 10^{-2} & -93.04 \\ 0 & 0 & 0 & 4.77 & -8.96 \cdot 10^{-2} & 1.17 \cdot 10^{-1} & 0 & 0 \\ 2.18 \cdot 10^{-4} & -35.74 & 0 & 0 & 9.39 \cdot 10^{-2} & 5.09 \cdot 10^{-2} & 2.22 & -2.04 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 51.65 & 2.12 & 8.35 \cdot 10^{-2} & 1.54 \cdot 10^{-1} & 0 & 0 \\ 3.42 \cdot 10^{-4} & -56.65 & 0 & 0 & -9.36 \cdot 10^{-4} & -5.07 \cdot 10^{-4} & 2.26 \cdot 10^{-1} & -2.27 \cdot 10^{-18} \\ 0 & 0 & 1.62 \cdot 10^{-2} & -1.17 & 1.58 \cdot 10^{-2} & -2.92 \cdot 10^{-2} & 0 & 0 \\ 3.16 & -6.43 \cdot 10^{-1} & 0 & 0 & 1.07 \cdot 10^{-2} & 5.80 \cdot 10^{-3} & -5.50 \cdot 10^{-3} & -93.06 \end{pmatrix} \begin{pmatrix} \Delta \delta_P \\ \Delta \delta_E \\ \Delta \delta_A \\ \Delta \delta_R \\ V_{x_w} \\ V_{y_w} \\ V_{z_w} \\ CD_{wave} \end{pmatrix} \tag{A.5}$$

B | Control parameters

B.1 Eigenvalues of the system matrix

$$det(A - I \cdot \lambda) = 0 \Rightarrow \begin{cases} \lambda_1 = 0 + 0i \\ \lambda_2 = 0 + 0i \\ \lambda_3 = -0.19 + 2.81i \\ \lambda_4 = -0.19 - 2.81i \\ \lambda_5 = -2.36 + 0i \\ \lambda_6 = -2.74 + 6.92i \\ \lambda_7 = -2.74 - 6.92i \\ \lambda_8 = -5.10 \cdot 10^{-3} + 0i \\ \lambda_9 = -4.80 \cdot 10^{-3} + 7.40 \cdot 10^{-2}i \\ \lambda_{10} = -4.80 \cdot 10^{-3} - 7.40 \cdot 10^{-2}i \\ \lambda_{11} = -2.30 \cdot 10^{-3} + 0i \end{cases} \tag{B.1}$$

B.2 LQR parameters

$$\begin{cases} Q_{1z} = \begin{pmatrix} 1 \cdot 10^{-6} & 0 & 0 & 0 & 0 & 0 \\ 0 & 4 \cdot 10^{-4} & 0 & 0 & 0 & 0 \\ 0 & 0 & 100 & 0 & 0 & 0 \\ 0 & 0 & 0 & 14.59 & 0 & 0 \\ 0 & 0 & 0 & 0 & 131.31 & 0 \\ 0 & 0 & 0 & 0 & 0 & 6 \cdot 10^{10} \end{pmatrix}, Q_{1V} = \begin{pmatrix} 4 \cdot 10^{-4} & 0 \\ 0 & 1 \cdot 10^{6} \end{pmatrix} \\ Q_{21} = \begin{pmatrix} 4.05 \cdot 10^{-1} & 0 & 0 & 0 \\ 0 & 4.05 \cdot 10^{-1} & 0 & 0 \\ 0 & 0 & 1 \cdot 10^{13} & 0 \\ 0 & 0 & 0 & 1 \cdot 10^{13} \end{pmatrix}, Q_{22} = \begin{pmatrix} 3.65 & 0 & 0 & 0 \\ 0 & 8.21 & 0 & 0 \\ 0 & 0 & 1 \cdot 10^{9} & 0 \\ 0 & 0 & 0 & 1 \cdot 10^{10} \end{pmatrix} \end{cases} \tag{B.2a}$$

$$\begin{cases} R_{1z} = 1.46 \cdot 10^{15}, R_{1V} = 2.00 \cdot 10^{10} \\ R_{21} = \begin{pmatrix} 2.92 \cdot 10^{12} & 0 \\ 0 & 1.34 \cdot 10^{13} \end{pmatrix}, R_{22} = \begin{pmatrix} 6.57 \cdot 10^{8} & 0 \\ 0 & 2.36 \cdot 10^{8} \end{pmatrix} \end{cases} \tag{B.2b}$$

$$\begin{cases} K_{1z} = \begin{pmatrix} -9.70 \cdot 10^{-3} & -4.18 \cdot 10^{-4} & 6.20 \cdot 10^{-3} & -1.57 & -5.46 \cdot 10^{-2} & 6.40 \cdot 10^{-3} \end{pmatrix} \\ K_{1V} = \begin{pmatrix} 6.68 \cdot 10^{-2} & -7.10 \cdot 10^{-3} \end{pmatrix} \\ K_{21} = \begin{pmatrix} -1.85 & 1.29 \cdot 10^{-2} \\ -6.00 \cdot 10^{-3} & -8.64 \cdot 10^{-1} \end{pmatrix} \\ K_{22} = \begin{pmatrix} 1.77 \cdot 10^{-1} & 1.22 \cdot 10^{-1} & -1.23 & 1.89 \cdot 10^{-1} \\ 1.33 \cdot 10^{-2} & -3.15 & -9.98 \cdot 10^{-2} & 6.50 \end{pmatrix} \end{cases} \tag{B.2c}$$

B.3 MPC parameters

$$\begin{cases} Weights_{output\ variables} = \begin{pmatrix} 2.50 \cdot 10^{-1} & 0 & 0 & 0 & 1 & 0 & 5 \cdot 10^{-1} & 0 & 30 & 0 & 2.50 \cdot 10^{-1} \end{pmatrix} \\ Weights_{manipulated\ variables} = \begin{pmatrix} 0 & 2 & 1 \cdot 10^{-2} & 1.25 \cdot 10^{-1} \end{pmatrix} \\ Weights_{manipulated\ variables\ rate} = \begin{pmatrix} 1 \cdot 10^{-3} & 3 & 1 \cdot 10^{-1} & 1 \cdot 10^{-1} \end{pmatrix} \\ ECR = 1 \cdot 10^{9} \end{cases} \tag{B.3}$$

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Design and Simulation of the guidance and control system for an F-86 Sabre

Project specification

Author: Álvaro Ortiz Moya
Tutor: Sergio García-Nieto Rodríguez

Aerospace Engineering Bachelor's Degree

Escuela Técnica Superior de Ingeniería del Diseño
Universitat Politècnica de València
Spain, 15th September 2020

Contents

1 | Objectives

In this project specification, the requirements to the execution of the project are detailed. There will be 5 functional groups, which have a total of different 11 work units and define the procedure to follow, in order to achieve the goals of the project, in the areas of mathematical models, control, guidance, dynamic path planning and result analysis. In addition, some conditions or necessities of materials must be satisfied. If the project planning follows the requirements of this project specification properly, the project will be conducted successfully and their objectives will be satisfied.

The required functional groups of work units are the following ones:

1. Development of the mathematical models.
2. Design of the controllers for the system.
3. Design of the guidance system.
4. Generation of the dynamic path.
5. Analysis of the navigation system quality.

2 | Conditions of materials and works

In this chapter, the technical conditions, which are going to be applied to the project, are detailed. In this way, the materials and work units must fulfil required characteristics and pass quality controls and tests. Although the functional group 4 (Generation of the dynamic path) is the unique one that does not need that the previous work units are completed, all the others work units does; therefore, this chapter has organised the work units chronologically.

2.1 Description

In this section, the 5 functional groups are detailed, explaining the procedure done at each case properly. In addition, the conditions of materials are explained as well.

2.1.1 Development of the mathematical models

This functional group encompasses the mathematical approach of a reliable model of the whole system, composed of the F-86 Sabre and two M117 bombs, but also the linearization of the complete system and the state-space model building. The work units of the group are: the implementation of the complete system model, the implementation of the M117 bomb model and, the linearization of the complete system, trimmed point calculation and state-space model generation.

Complete system model

A deep investigation about the dimensions and weights of the F-86F-20 model of the Sabre and the bomb M117 is conducted. The inertia matrices of both will be calculated and combined in order to develop the real inertia matrix of the complete system. In addition, the actuators and the limitations in the control actions of the aircraft will be defined.

The flight mechanics model is employed in order that the whole motion of the system can be analysed. Furthermore, the aerodynamic coefficients of the aircraft will be calculated and interpolated, defining all the aspects of the flight mechanics model.

Finally, a suitable wind model will be generated, based on the meteorological conditions of the mission area. The system and wind models are implemented in Matlab, in such way that the system model is ready for being linearized in the code and the wind model parameters are defined before the simulations of the Work unit 10.

M117 bomb model

In this work unit, the flight mechanics model is also applied; although, in this case, with some simplifications due to the bomb geometry and configuration. Moreover, the aerodynamic coefficients are computed as well. Then, the mathematical model is implemented in Matlab, specifically, in the dropping point estimation algorithm from Work unit 9.

State-space model

Once the Work unit 1 is done, the 11 selected equations that define the system behaviour will be linearized, which is conducted by means of a linearization algorithm. Firstly, state and control variables are defined; afterwards, the group of equations are derived from each one of the variables and with respect to a symbolic linearization point, then, a matrix of symbolic coefficients is obtained, which are substituted by the equilibrium point parameters to obtain the real matrix of coefficients.

In order to determine the equilibrium point, the aircraft will be trimmed in a vertical position motion and in function of the imposed linearization parameters (initial yaw angle, average velocity and average altitude). At the end, the matrices that compose the linearized state-space model are obtained from the real matrix of coefficients. All this procedure is accomplished in Matlab.

2.1.2 Design of the controllers for the system

In this group, the analysis of the possible control techniques applicable to the problem is performed, together with the selection of the two most adequate options. In addition, the design of the chosen controllers is conducted In this way, the work units of this functional group are: study of control methods alternatives and selection of the most appropriate two of them and, design of the controllers to the compliance of the specifications required by the contractor

Study of alternatives and selection of two of them

So as to obtain the most satisfactory controller options to the actual issue, an investigation about the current control methodologies must be performed. In this way, different controllers are examined to determine whether they could be valid or not on the basis of the following characteristics: robustness when facing perturbations and non-modelled dynamics, design simplicity, implementation easiness and

proper reference tracking. The two most adequate contemplated options will be chosen.

Design of controllers

In this work unit, the two most adequate controllers from the previous one are developed. The design of the controllers is completely based on the investigation performed before, although the final structure and required parameters must be determined by the designer (in this case, the engineer). However, the selection of those parameters is not aleatory, it must ensure the compliance of different requirements from the contractor.

Therefore, these controllers must be implemented in Simulink and be able to guarantee: adequate reference tracking if the dynamics of the aircraft allows it; suitable control actions (not excessive ones that can lead to instability); avoiding flight conditions which could produce dangerous conditions in the real flight, and proper responses to perturbations.

2.1.3 Design of the guidance system

As it has been done in the previous functional group, an initial analysis that explores the different guidance law options and the selection of the most appropriate alternative is performed. Afterwards, the chosen guidance method is implemented in Simulink by means of developing the needed algorithm. The work units of this group are: study of guidance law alternatives and selection of the most adequate option and, design and implementation of the guidance algorithm.

Study of alternatives and selection of the most adequate option

In this work unit, distinct alternatives of guidance laws are proposed, therefore, an investigation about the principal types of guidance law (proportional and modern options) is performed, extracting the required information from thesis and papers in relation to the issue.

The selected guidance law must provide to the controller the reference variables of altitude, velocity and roll and yaw angles by means of the dynamic path parameters, defined in the Work unit 9. In addition, the guidance method has to transmit to the controller proper references in those 4 state variables: the parameters must be adjusted in order that the control actions are not excessive, but also not too much weak, since the reference tracking would be fairly poor.

Design of the guidance algorithm

The most suitable guidance law is implemented in Simulink as a block that receives the current flight parameters of the aircraft as an input and sends in the output the references in altitude, velocity and roll and yaw angles.

The guidance algorithm must select a point from the reference dynamic path and evaluate the roll and yaw angle needed to perform a change in course angle. The selection of this point has to depend on the velocity of the aircraft, from which the coordinates (x,y) are obtained. The altitudes and velocities to be tracked are got from the closest point of the reference trajectory to the current position of the aircraft.

2.1.4 Generation of the dynamic path

In this case, the mission of the project is fully explained and the problem is defined. To solve the mission requirements, while respecting the constraints imposed, different algorithms are utilised to create a suitable dynamic path and optimise it in order to facilitate to the navigation to the aircraft. Furthermore, the dropping point estimation method is developed. The work units are the following ones: definition of the problem and explanation of the mission example that must be conducted and, design and implementation of the algorithms which intervene in the dynamic path planning (Recursive Rewarding Adaptive Cell Decomposition and 3D Smooth Path Planning), together with the required code needed to define the optimal dropping point at the reference trajectory.

Definition of the problem

In this work unit, the mission is defined and the constraints and restricted areas, through which the aircraft can not go, are specified. These restrictions and the possibility of collision of a 3D space with them, are introduced in the Matlab code of the RR-ACD algorithm by means of the Collision Toolbox, which has been investigated in order to fully understand the capabilities of the Toolbox. In addition, the ground elevation of the mission area is computed based on the concept of great circle.

The problem definition must define all aspects of the mission and the options of collision objects to insert.

Design of algorithms

The RR-ACD algorithm is implemented in Matlab and generates the path solution to the mission, creating a trajectory that fulfils all the restrictions imposed. The algorithm must decompose initially the total 3D space in Cartesian coordinates and apply again other decomposition to each neighbour subspace, in order to find which ones are occupied and which ones are free, by means of an Adaptive Cell Decomposition methodology. Once the next available free spaces are defined, the Recursive Rewarding method evaluates the most acceptable one, specifying the next waypoint on the route.

Once the path is created, a soften process is applied. The 3D Smooth Path Planning must introduce radii of gyration at each vertex of the trajectory, so that

the aircraft can follow the trajectory with less difficulties. Moreover, the reference velocity profile has to be generated by this algorithm.

At the end, the dropping point estimation algorithm is performed. This one, will have to calculate each trajectory of the bombs thrown from the closest waypoints to the objective of impact (inside of a given range) and determine the most appropriate point to release the bombs.

2.1.5 Analysis of the navigation system quality

In this functional group, different simulations are made at distinct conditions of flight in order to determine whether the controllers and guidance system fulfil the conditions of the contractor and the mission requirements. The analysis is performed over the flight parameters, final trajectories and impacts of the bombs. The corresponding work units are: simulations at different conditions and analysis of the system capabilities and study of the impact of the M117 bombs at each simulation case

Simulations to the analysis of the system capabilities

In this work unit, different simulations are performed in order to evaluate the mission compliance, employing the chosen controllers from the Work unit 5 and the guidance system from Work unit 7. The simulations must be made at these conditions: non-wind, application of the wind model (from Work unit 1) and utilisation of wind model and consideration of variable mass due to the fuel consumption. This is due to the fact that the controllers must be tested in different situations, so as to prove the robustness when the system faces perturbations and non-modelled dynamics, ensuring that the aircraft can perform the dynamic path without extreme errors or entering into restricted areas.

In addition, a simulation in FlightGear will be conducted, in order to show graphically the estimated motion of the aircraft in attitude, velocity and altitude.

Study of the impact of the M117 bombs

Once the simulations have been made, the dropping point estimation algorithm from the Work unit 9 is applied. The purpose of this application consists in determining which control method is the most satisfactory one in terms of accuracy in the bomb impact and deviation in the dropping point from the reference one.

2.1.6 Materials

In the project, computer equipment (hardware) and programmes (software) are employed to generate and analyse the autonomous navigation system of the F-86 Sabre.

Hardware

The hardware used corresponds to a laptop hp, in which the work units are conducted, such as design, calculations, data simulations and redaction of the project. The characteristics of the laptop hp are the following ones:

1. Model: HP Notebook
2. Processor: Intel Core i5-7200U CPU 2.50 GHz 2.70 GHz
3. RAM memory: 16 GB
4. Graphic card: AMD Radeon R7 M440
5. Operating system: Windows 10 Home
6. Architecture: 64 bits.

Software

The software includes the following programmes or toolboxes from Matlab:

1. Matlab R2019a: real and symbolic calculations.
2. Simulink R2019a: simulation of the system with block diagrams.
3. Aerospace Toolbox: blocks used in Simulink to the performance of simulations.
4. FlightGear: simulator of the aircraft motion.
5. Google Earth 2020: creation of maps and evaluation of coordinates.

2.2 Quality control

Once the different conditions of the work units have been described, a quality control must be applied at each one of them in order to guarantee the compliance of all requirements imposed that allow the satisfactory performance of the project.

2.2.1 Development of the mathematical models

Complete system model

The mathematical system model must approximate as much as possible the real behaviour of the aircraft with the bombs at a flight. In order to check possible confusions during the investigation of all required parameters, the non-linear model will be tested in Simulink in open loop configuration, in order to determine whether the system is well defined or not. Therefore, aspects of the aircraft related to characteristics such as maximum speed, the flight envelope or turn rates will be analysed and compared to the bibliography.

M117 bomb model

In the same way, the model of M117 bombs will analysed at the performance of the dropping point estimation algorithm (from the Work unit 9). The trajectory and flight parameters will be checked and possible errors in the model must be detected during the calculations.

State-space model

The linearized state-space model must be compared to bibliography in relation to flight mechanics in order to check if the matrices have sense or not. The state-space model does not take many simplifications that the usual flight mechanics model does, so that it has to be taken into account as well. The linearized state-space model will be tested in open loop structure to determine the result quality of the trimmed point calculation (if the system is stable when it starts at the calculated state and control variables from trimmed point, this calculation is perfect) and the validity of the system and control matrices configuration.

2.2.2 Design of the controllers for the system

Study of alternatives and selection of two of them

In this work unit, no control quality can be applied, since an investigation study is performed. Only, different sources must be analysed at each control method alternative in order to corroborate the information.

Design of controllers

The selected controllers must be adapted to the issue of the mission and fulfil the requirements from the previous section. In order to ensure that, different combination of controller parameters and distinct structures (if the structure can be varied, as in the LQR case) must be applied and tested in Simulink, so that the most appropriate possible combination is chosen. Therefore, the controllers, together with the guidance system, must be checked at different trajectories before testing the real dynamic path from the Work unit 9.

2.2.3 Design of the guidance system

Study of alternatives and selection of the most appropriate option

By the same token as in the Work unit 4, no quality control can be applied at this case, only a deep analysis of the different alternatives, employing distinct resources, is performed.

Design of the guidance algorithm

Together with the Work unit 5 checking, this one is tested as well. Different combination of the parameters of the guidance are proved in Simulink so as to obtain the most suitable guidance law for the aircraft. This is performed at different trajectories and velocities in order to guarantee the appropriate behaviour of the system in the actual mission simulations.

2.2.4 Generation of the dynamic path

Definition of the problem

In this work unit, some investigations to define the methodology of the algorithms utilised and the mission of the project are made. On the basis of those investigations, the constraints of the mission are detailed, implementing them in Matlab. Therefore, an analysis must be made so as to specify the restrictions of the 3D space building capabilities, together with the proper creation of collision elements in Matlab.

Design of algorithms

The algorithms must be proved in different configurations of constraints at the actual mission, ensuring their adequate functioning in the problem solving, but also specifying their limitations. The parameters used in the algorithms must be adjusted to obtain the most satisfactory possible result, so that multiple tests must be conducted to examine the algorithm outputs.

2.2.5 Analysis of the navigation system quality

Simulations to the analysis of the system capabilities

This work unit mainly consists in a whole quality control at different conditions of the autonomous navigation system that has been designed. The system must be tested in the actual mission to ensure its capabilities, so that an exam of control, guidance and dynamic path planning requirements must be passed. In case that the imposed necessities of the system are not fulfilled, other analysis of the distinct parameters used in controllers and algorithms must be performed.

Study of the impact of the M117 bombs

Finally, the most adequate control method is chosen in terms of quality of the bombs droppings. The system has to ensure the most satisfactory possible impact of the bombs in order to destroy the objective of the mission; hence, in the case that a bad result is obtained, a new analysis of parameters (similar to the one at the previous work unit) must be conducted.

2.2.6 Materials

The needed materials must perform properly and no failures have to be detected. In such case, the damaged material is repaired, or in the case that it is not possible, it is replaced by a new one.

3 | Conditions of execution

The objective of this chapter consists in regulating the execution of the work units, assigning the corresponding functions to the contractor and the engineer, together with the relations among them. All terms of execution of the project are defined, in which, in addition, the aspects of security, hygiene and health are specified. These are regularised by the Ministry of Work according to the Legislative Decree 486/1997, on April 14^{th}, which defines the minimum security and health conditions in workspaces.

The work contract is composed of the following documents:

1. Conditions fixed in the document of contract.
2. The project specification.
3. The budget of the project.
4. The project documentation (report, plans and rest of documents).

The instructions and orders from the work direction are result of exact interpretation of the report and specifications of the project. In addition, literal specifications prevail over the graphs.

3.1 Description

In this section, the execution conditions of the project will be detailed in each one of their possible aspects.

3.1.1 Conditions of workspace

The workspace must provide enough dimensions to keep all needed elements and allow a minimum mobility of the worker. The contractor must offer security conditions by means of applying distinct measures, from an emergency plan design to a protection plan against fires. In addition, the workspace must be kept in conditions of cleanliness and organisation.

Most of the project will be performed with the help of visualisation screens, therefore, their utilisation must follow the Legislative Decree 486/1997. The contractor has to ensure the compliance of such decree. Furthermore, the works which really need these elements, are characterised by long positions of the body during

the labour day, hence, it is also recommended that the contractor provides: seats, tables, high resolution screens, keyboards and protection glasses against the light from the screens.

The electrical installation must be managed by a business authorised by the Ministry of Industry or any other qualified organisation related to the corresponding region of the country.

Regarding the ambient conditions, defined by the Legislative Decree 486/1997, the temperature of the workspace must be in the range of 23-26°C in summer and 20-24°C in winter. Moreover, the humidity of the air must be kept between a 45% and a 65%. Furthermore, the equipment of the workspace can not produce additional heat which bothers to workers.

Finally, the conditions of illumination and noise should be the most adequate ones as far as possible. The illumination can be natural or artificial, although natural one would be preferred, and must be adapted to ensure the minimum level of light to not affect to workers. In the case of the noise, the workspace must be as quietly as possible, the levels of noise can exceed 55dB under no circumstances.

3.1.2 Work planning

The contractor will conduct the work units in the given range in the period required by the contract, fulfilling the partial periods for each work unit resolution. The planning of the works must follow the structure given by this present document, explained in the Chapter 2; however, if the technical circumstances entails any issue in the planning ruled by the project specification, the contractor can modify the structure of the works providing that the engineer agrees.

In addition, the contractor must inform to the engineer the start of the works with, at least, three labour days of anticipation.

3.1.3 Work enlargement owing to unexpected reasons

Due to unexpected reasons or sudden requirements, the project can be enlarged. The work units that are being developed, will not stop their execution and planning, although the contractor must perform the needed works urgently; while, the engineer will modify the project basis and its budget to adapt them to the new circumstances if it was necessary.

3.1.4 Extension due to causes of force majeure

In case of force majeure or choice of the contractor that could affect to the start of the works or the planning of the project, or even, suppose the cancellation of the project, an extension of the contract will be applied to fulfil its compliance (proving that the engineer agrees).

3.1.5 General conditions of works executions

The work will be strictly performed based on the project, its corresponding modifications and work orders from the engineer.

3.1.6 Faulty works

The contractor must utilise the required and optimal materials to the resolution of the work units and conduct the works in accordance with the project specification. This person will be responsible for the execution of the project and for a possible wrong procedure. In the case that the engineer or the contractor notices defects in the work materials, the contractor must provide new ones.

3.1.7 Materials origin

The contractor will provide by himself the materials to the project from the source or origin that he/she desires and decides as the most appropriate one, informing to the engineer of the decision and its fundamentals. In case that the engineer disagrees the origin selection of the materials, the contractor must find other one.

3.1.8 Additional tests

All additional required tests to develop the project must be paid by the contractor, repeating the works which do not offer enough guarantees. The final tests of analysis are included in the project, but the additional ones are not.

3.2 Quality control

All the conditions of execution of the project that have been mentioned must be proved along the resolution of the works. In case of misunderstanding or issues between the contractor and the workers (the engineer in this case), the complete group of conditions of execution must be examined and it has to be decided which conditions are not being satisfied properly and the procedure to solve the problems so as to continue the works.

4 | Final tests and adjustments

In this final chapter, the final tests and adjustments to the project are defined. In the case that the contractor and the engineer conclude that it is necessary to perform a modification of the project, increment the number of tests to ensure the capabilities of the system or improve the system due to possible errors, a revision of the project will be conducted by the engineer in order to satisfy the decision made by both and an additional budget will be made.

Perhaps, the results are adequate enough but they are not as satisfactory as the contractor hoped, or new events say that it is better to change any methodology. In this case, the configuration of the parameters at each part of the project is adjusted again in order to improve the results.

In addition, the number of test could be very low according to the contractor's new opinion. Then, the wind model is modified and the mission configuration is changed in order to create new conditions of testing and, consequently, demonstrate the acceptable operation of the system.

Finally, the results can be wrong or not as adequate as they must be. The errors in results can be due to many possibilities, which must be examined one by one. The errors can be owing to:

1. Model: the mathematical models are not valid and produce errors that the engineer has not noticed, giving wrong solutions which do not represent the real behaviour of the system
2. Control: the controllers do not provide an appropriate response of the system, making that the aircraft presents an excessive instability.
3. Guidance: the guidance law is not correct or it is not an acceptable option for the case.
4. Dynamic path planning: the dynamic trajectory of the aircraft is not realistic or the mission is wrongly planned.
5. Simulations: the simulations are not accomplished well, providing results that do not represent the previous work done properly.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Design and Simulation of the guidance and control system for an F-86 Sabre

Budget of the project

Author: Álvaro Ortiz Moya
Tutor: Sergio García-Nieto Rodríguez

Aerospace Engineering Bachelor's Degree

Escuela Técnica Superior de Ingeniería del Diseño
Universitat Politècnica de València
Spain, 15th September 2020

Contents

List of Tables

1 | Introduction

This document provides all the needed information about the budget of the present project "Design and Simulation of the guidance and control system for an F-86 Sabre". Therefore, the costs of analysis and design of the whole autonomous navigation system implemented for the aircraft are going to be exposed. This project has been divided into 11 work units, arranged in 5 functional groups as it follows:

1. Development of the mathematical models.
   (a) Working unit 1: Complete system model.
   (b) Working unit 2: M117 bomb model.
   (c) Working unit 3: State-space model.
2. Design of the controllers for the system.
   (a) Working unit 4: Study of alternatives and selection of two of them.
   (b) Working unit 5: Design of the controllers.
3. Design of the guidance system.
   (a) Working unit 6: Study of alternatives and selection of the most adequate option.
   (b) Working unit 7: Design of the guidance algorithm.
4. Generation of the dynamic path.
   (a) Working unit 8: Definition of the problem.
   (b) Working unit 9: Design of algorithms.
5. Analysis of the navigation system quality.
   (a) Working unit 10: Simulations to the analysis of the system capabilities.
   (b) Working unit 11: Study of the impact of the M117 bombs.

All costs, including equipment and personnel, of each one of these work units are calculated, specifying a single cost per work unit. At the end, the direct costs due to the project, industrial benefit and taxes are explained.

2 | Labour costs

In this chapter, the labour costs of the project are defined. In this case, it is going to be taken into account the salary of a technical aerospace engineer working at full-time. In addition to the gross salary, the social security contribution is added to the global costs, for which an office work with indefinite contract has been considered. The work characteristics, salaries, contributions and final cost per hour are shown in the tables 2.1 and 2.2.

Labour costs			
Gross salary (€/month)	Labour hours/day	Labour days/month	Annual salary (€)
2000	8	20	24000

Table 2.1: Gross labour costs of a technical aerospace engineer at full-time.

Full costs			
Social Security contribution (%)	Taxes to the Social Security (€)	Annual costs (€)	Final price (€/h)
30.90	7416	31416	16.36

Table 2.2: Total labour costs of a technical aerospace engineer at full-time.

3 | Material costs

In this section, the materials costs are estimated based on hours of utilisation of the goods in possession and computer licences needed to conduct the project. The total costs of these elements are given in the table 3.1.

Material costs		
Material		Price (€)
Matlab licenses	Laptop hp	750
	Matlab (annual)	800
	Simulink (annual)	1200
	Aerospace Toolbox (annual)	460
FlightGear license		0
Google Earth license		0

Table 3.1: Material costs of the goods and licenses.

Afterwards, the lifespan of these goods and computer licenses in possible labour hours is calculated, obtaining a final result of prices per hour of use, as it is seen in the table 3.2. The labour hours per day and month are based on the values presented in the table 2.1.

Material Costs					
Material	Price (€)	Lifespan (years)	Total available labour hours/year	Total available hours (h)	Final price (€/h)
Laptop hp	750	5	1920	9600	$7.81 \cdot 10^{-2}$
Matlab license	800	1	1920	1920	$4.17 \cdot 10^{-1}$
Simulink license	1200	1	1920	1920	$6.25 \cdot 10^{-1}$
Aerospace Toolbox license	460	1	1920	1920	$2.40 \cdot 10^{-1}$

Table 3.2: Material costs per labour hour of utilisation.

4 | Costs due to labour hours

In the following chapter, the total amount of labour hours at each work unit from all functional groups is itemised. In addition, the partial cost of each work unit is computed on the basis of the required labour and materials costs.

4.1 Estimation of labour hours

4.1.1 Development of the mathematical models

Work unit 1		
Tasks	Days	Labour hours
Dimensions and geometries	1	3
Flight mechanics model approach	1	2
Evaluation of aerodynamic coefficients	3	6
Wind model building	1	3
Implementation in Matlab	1	3
TOTAL		**17**

Table 4.1: Labour hours of the Work unit 1.

Work unit 2		
Tasks	Days	Labour hours
Flight mechanics model approach	1	1
Evaluation of aerodynamic coefficients	1	5
TOTAL		**6**

Table 4.2: Labour hours of the Work unit 2.

Work unit 3		
Tasks	Days	Labour hours
State and control variables definition	1	3
Linearization of the equations: implementation in Matlab	2	6
Trimmed point calculation: implementation in Matlab	1	4
System and control matrices building: implementation in Matlab	1	2
TOTAL		**16**

Table 4.3: Labour hours of the Work unit 3.

4.1.2 Design of the controllers for the system

Work unit 4		
Tasks	Days	Labour hours
List of alternatives of controllers	1	4
Investigation about similar projects with those alternatives	3	15
TOTAL		**19**

Table 4.4: Labour hours of the Work unit 4.

Work unit 5		
Tasks	Days	Labour hours
Preliminary study of LQR	1	5
Selection of the most adequate LQR structure	5	30
Determination of static gains of LQR	3	15
Preliminary study of MPC	3	13
Selection of the most adequate MPC configuration	2	10
Determination of weigh parameters of MPC	3	25
TOTAL		**98**

Table 4.5: Labour hours of the Work unit 5.

4.1.3 Design of the guidance system

Work unit 6		
Tasks	Days	Labour hours
List of alternatives of guidance laws	1	3
Investigation of similar projects with those alternatives	2	5
TOTAL		**8**

Table 4.6: Labour hours of the Work unit 6.

Work unit 7		
Tasks	Days	Labour hours
Determination of the guidance law parameters	2	10
Implementation of the guidance algorithm in Matlab	3	15
TOTAL		**25**

Table 4.7: Labour hours of the Work unit 7.

4.1.4 Generation of the dynamic path

Work unit 8		
Tasks	Days	Labour hours
Investigation about great circles	1	2
Historical investigation	1	2
Environment construction in Google Earth	1	2
Ground elevations calculations	3	14
Investigation about Matlab Collision Toolbox	1	4
Implementation of constraints in Matlab	2	10
TOTAL		**34**

Table 4.8: Labour hours of the Work unit 8.

Work unit 9		
Tasks	Days	Labour hours
Investigation about different Adaptive Cell Decomposition algorithms	1	3
Implementation of Recursive Rewarding Adaptive Cell Decomposition algorithm in Matlab	12	50
Implementation of 3D Smooth Path Planning algorithm in Matlab	6	25
Implementation of dropping point estimation algorithm in Matlab	2	5
TOTAL		**83**

Table 4.9: Labour hours of the Work unit 9.

4.1.5 Analysis of the navigation system quality

Work unit 10		
Tasks	Days	Labour hours
Realisation of simulations at non-wind conditions	1	2
Realisation of simulations at wind conditions	1	2
Realisation of simulations at wind and variable mass conditions	1	2
Comparison of results and problems checking in control actions and flight parameters at each case	2	5
TOTAL		**11**

Table 4.10: Labour hours of the Work unit 10.

Work unit 11		
Tasks	Days	Labour hours
Application of the Dropping Point Estimation algorithm to results of simulations	1	2
Determination of the most appropriate control method in impact accuracy and deviations from the reference	1	1
TOTAL		**3**

Table 4.11: Labour hours of the Work unit 11.

4.2 Decomposed prices

4.2.1 Development of the mathematical models

Work unit 1	
Labour hours (h)	17
Materials price (€/h)	Laptop hp ($7.81 \cdot 10^{-2}$) Matlab license ($4.17 \cdot 10^{-1}$)
Materials costs (€)	8.42
Salary price (€/h)	16.36
Personnel costs (€)	278.12
TOTAL	**286.54 €**

Table 4.12: Costs of the Work unit 1.

Work unit 2	
Labour hours (h)	6
Materials price (€/h)	Laptop hp ($7.81 \cdot 10^{-2}$) Matlab license ($4.17 \cdot 10^{-1}$)
Materials costs (€)	2.97
Salary price (€/h)	16.36
Personnel costs (€)	98.16
TOTAL	**101.13 €**

Table 4.13: Costs of the Work unit 2.

Work unit 3	
Labour hours (h)	16
Materials price (€/h)	Laptop hp ($7.81 \cdot 10^{-2}$) Matlab license ($4.17 \cdot 10^{-1}$)
Materials costs (€)	7.92
Salary price (€/h)	16.36
Personnel costs (€)	261.76
TOTAL	**269.68 €**

Table 4.14: Costs of the Work unit 3.

4.2.2 Design of the controllers for the system

Work unit 4	
Labour hours (h)	19
Materials price (€/h)	Laptop hp ($7.81 \cdot 10^{-2}$)
Materials costs (€)	1.48
Salary price (€/h)	16.36
Personnel costs (€)	310.84
TOTAL	**312.32 €**

Table 4.15: Costs of the Work unit 4.

Work unit 5	
Labour hours (h)	98
Materials price (€/h)	Laptop hp ($7.81 \cdot 10^{-2}$) Matlab license ($4.17 \cdot 10^{-1}$) Simulink license ($6.25 \cdot 10^{-1}$)
Materials costs (€)	109.77
Salary price (€/h)	16.36
Personnel costs (€)	1603.28
TOTAL	**1713.05 €**

Table 4.16: Costs of the Work unit 5.

4.2.3 Design of the guidance system

Work unit 6	
Labour hours (h)	8
Materials price (€/h)	Laptop hp ($7.81 \cdot 10^{-2}$)
Materials costs (€)	0.62
Salary price (€/h)	16.36
Personnel costs (€)	130.88
TOTAL	**131.50 €**

Table 4.17: Costs of the Work unit 6.

Work unit 7	
Labour hours (h)	25
Materials price (€/h)	Laptop hp ($7.81 \cdot 10^{-2}$) Matlab license ($4.17 \cdot 10^{-1}$) Simulink license ($6.25 \cdot 10^{-1}$)
Materials costs (€)	28.00
Salary price (€/h)	16.36
Personnel costs (€)	490
TOTAL	**518.00 €**

Table 4.18: Costs of the Work unit 7.

4.2.4 Generation of the dynamic path

Work unit 8	
Labour hours (h)	34
Materials price (€/h)	Laptop hp ($7.81 \cdot 10^{-2}$) Matlab license ($4.17 \cdot 10^{-1}$)
Materials costs (€)	16.83
Salary price (€/h)	16.36
Personnel costs (€)	556.24
TOTAL	**573.07 €**

Table 4.19: Costs of the Work unit 8.

Work unit 9	
Labour hours (h)	83
Materials price (€/h)	Laptop hp ($7.81 \cdot 10^{-2}$) Matlab license ($4.17 \cdot 10^{-1}$)
Materials costs (€)	41.09
Salary price (€/h)	16.36
Personnel costs (€)	1357.88
TOTAL	**1398.97 €**

Table 4.20: Costs of the Work unit 9.

4.2.5 Analysis of the navigation system quality

Work unit 10	
Labour hours (h)	11
Materials price (€/h)	Laptop hp ($7.81 \cdot 10^{-2}$) Matlab license ($4.17 \cdot 10^{-1}$) Simulink license ($6.25 \cdot 10^{-1}$) Aerospace Toolbox ($2.40 \cdot 10^{-1}$)
Materials costs (€)	14.96
Salary price (€/h)	16.36
Personnel costs (€)	179.96
TOTAL	**194.92 €**

Table 4.21: Costs of the Work unit 10.

Work unit 11	
Labour hours (h)	3
Materials price (€/h)	Laptop hp ($7.81 \cdot 10^{-2}$) Matlab license ($4.17 \cdot 10^{-1}$)
Materials costs (€)	1.49
Salary price (€/h)	16.36
Personnel costs (€)	49.08
TOTAL	**50.57 €**

Table 4.22: Costs of the Work unit 11.

5 | Partial budget

In this chapter, the a partial budget of the project is accomplished. In the table 5.1, the total cost of each work unit is calculated based on the needed quantities of all of them. In addition, the complete cost is computed.

Partial budget			
Work unit	Quantity	Price (€)	Total cost (€)
1	1	286.54	286.54
2	1	101.13	101.13
3	1	269.68	269.68
4	1	312.32	312.32
5	1	1713.05	1713.05
6	1	131.50	131.50
7	1	518.00	518.00
8	1	573.07	573.07
9	1	1398.97	1398.97
10	1	194.92	194.92
11	1	50.57	50.57
TOTAL			**5549.75**

Table 5.1: Partial budget with work units costs.

6 | Final budgets

Once the costs of each element have been calculated and a partial budget has been created, the following final budgets are computed in the table 6.1. These budgets are: material execution, investment and bid base.

Final budgets	
Concept	Total cost (€)
Functional group 1	657.35
Functional group 2	2025.37
Functional group 3	649.50
Functional group 4	1972.04
Functional group 5	245.49
TOTAL MATERIAL EXECUTION	**5549.75**
General expenses (15%)	832.46
Industrial benefit (6%)	332.99
TOTAL INVESTMENT	**6715.20**
IVA (21%)	1410.19
TOTAL BID BASE	**8125.39**

Table 6.1: Material execution, investment and bid base budgets.

Hence, the final costs of the project are summarised in the following budgets from the previous table 6.1:

1. TOTAL MATERIAL EXECUTION BUDGET: FIVE THOUSAND FIVE HUNDRED FORTY-NINE EUROS AND SEVENTY-FIVE CENTS
2. TOTAL INVESTMENT BUDGET: SIX THOUSAND SEVEN HUNDRED FIFTEEN EUROS AND TWENTY CENTS
3. TOTAL BID BASE BUDGET: EIGHT THOUSAND ONE HUNDRED TWENTY-FIVE EUROS AND THIRTY-NINE CENTS