UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

Automated Contingency Management in Unmanned Aircraft Systems

Presented by:
Hèctor Usach Molina

Supervised by:
Dr. Juan Antonio Vila Carbó

September 2019

Abstract

Technological development and scientific research are steadily enabling higher levels of automation in the global industry. In the aerospace sector, the operation of Unmanned Aircraft System (UAS) is a clear example. Given the huge potential of the UAS market, Civil Aviation Authorities are elaborating a new regulatory framework for the safe integration of UAS into the civil airspace. The general goal is ensuring that the operation of UAS has an Equivalent Level of Safety (ELOS) to that of manned aviation.

To meet the previous goal, this thesis advocates for increasing the level of automation of UAS operations by providing the automatic system on-board the aircraft with Automated Contingency Management (ACM) functions. ACM functions are designed to assist the pilot-in-command in case a contingency, and ultimately to fully replace the pilot if this is required by the situation (e.g. due to a Command and Control (C2) link loss) or if the pilot decides so. However, in order for automation to be safe, automated functions must be developed following safe design methodologies based on aerospace standards.

The thesis develops a technological solution that is based on three pillars: *a)* a software architecture for the automatic system on-board the aircraft that tries to autonomously adapt to contingencies while still achieving mission objectives; *b)* a novel Mission Plan specification than increases predictability in the event of a contingency; and *c)* a probabilistic risk model that ensures that the flight trajectory is optimal from the point of view of the risk exposure.

The different proposals are prototyped and validated using a simulation environment. The results obtained support the idea that an increase in the automation level of the aircraft can be an effective means towards the safe integration of UAS into the civil airspace. The proposed ACM functions are proved to reduce the operational risk in the event of a contingency, while ensure that the aircraft remains predictable, even without pilot intervention.

Resumen

El ritmo de desarrollo tecnológico actual y la investigación científica están permitiendo alcanzar mayores niveles de automatización en todos los sectores industriales. Uno de los ejemplos más representativos es el uso de aeronaves no tripuladas (UAS) en diferentes aplicaciones. Debido al gran potencial de este tipo de aeronaves, las Autoridades de Aviación Civil están desarrollando un nuevo marco regulatorio que permita integrarlas en el espacio aéreo civil de forma segura. El objetivo consiste en garantizar que la operación con UAS se realice con un nivel de seguridad equivalente al de la aviación tripulada convencional.

Para tratar de alcanzar este objetivo, esta tesis propone aumentar el nivel de automatización de un UAS dotando al sistema embarcado con la capacidad de Gestión Automática de Contingencias (ACM). La función del sistema ACM es la de asesorar al piloto en el momento en que se produce una contingencia en vuelo; y en última instancia, tomar el control total de la aeronave si la situación así lo requiere (por ejemplo, en caso de pérdida del enlace de Comunicación y Control (C2)) o si el piloto delega la resolución del conflicto al sistema automático. Para acreditar que las nuevas funciones no suponen un riesgo añadido para la operación, resultará determinante seguir metodologías de diseño seguro basadas en los estándares de la industria aeroespacial.

La tesis propone una solución tecnológica basada en tres pilares: *a)* una arquitectura software para el sistema automático a bordo de la aeronave que trate de adaptar la trayectoria de vuelo a la condición operacional del vehículo, equilibrando seguridad y robustez; *b)* una especificación de Plan de Misión novedosa

que permita aumentar la predictibilidad de la aeronave tras sufrir una contingencia; y *c)* un modelo de riesgo que permita determinar la ruta que minimiza el riesgo derivado de la operación.

Las diferentes propuestas realizadas en esta tesis se han implementado en un demostrador y se han validado en un entorno de simulación. Los resultados obtenidos apoyan la idea de que dotar al sistema embarcado de mayor grado de automatización puede ser un mecanismo viable hacia la integración segura de UAS en el espacio aéreo civil. En concreto, los resultados muestran que el sistema ACM propuesto es capaz de reducir el riesgo de la operación tras sufrir una contingencia y que, cuando esto ocurre, la respuesta de la aeronave sigue siendo predecible, incluso si el piloto no puede intervenir.

Resum

El ritme de desenvolupament tecnològic actual i la investigació científica estan permetent implementar majors nivells d'automatització a tots els àmbits de la indústria. Un dels exemples més representatius és l'ús d'aeronaus no tripulades (UAS) en diferents aplicacions. Vist el gran potencial d'aquest tipus d'aeronaus, les Autoritats d'Aviació Civil estan tractant de desenvolupar un nou marc regulador que permeta integrar-les en l'espai aeri civil de forma segura. Es tracta de garantir que l'operació d'un UAS es realitza amb un nivell de seguretat equivalent al de l'aviació tripulada convencional.

Per tal d'assolir aquest objectiu, aquesta tesi proposa augmentar el nivell d'automatització d'un UAS dotant el sistema embarcat amb la capacitat de Gestió Automàtica de Contingències (ACM). La funció del sistema ACM és assessorar el pilot quan ocorre una contingència en vol; i en última instància, prendre el control total sobre l'aeronau si és necessari (per exemple, en cas de pèrdua de l'enllaç de Comunicació i Control (C2)) o si el pilot delega la resolució del conflicte al sistema automàtic. Per tal d'acreditar que les noves funcions del sistema automàtic no comporten un risc afegit per a l'operació, resultarà determinant emprar metodologies de disseny segur d'acord amb els estàndards de la indústria aeroespacial.

La tesi proposa una solució tecnològica basada en tres pilars: *a)* una arquitectura software per al sistema automàtic a bord de l'aeronau que tracte d'adaptar la trajectòria de vol a la condició operacional del vehicle, equilibrant seguretat i robustesa; *b)* una especificació de Pla de Missió innovadora que permeta augmentar la predictibilitat de l'aeronau quan ocorre una contingència; i *c)* un

model de risc que permeta determinar la ruta que minimitza el risc derivat de l'operació.

Les distintes propostes realitzades en aquesta tesi s'han implementat sobre un demostrador i s'han validat en un entorn de simulació. Els resultats de la investigació recolzen la idea que dotar el sistema embarcat d'un major grau d'automatització pot ser un mecanisme adient per integrar els UAS en l'espai aeri civil de manera segura. En concret, els resultats indiquen que el sistema ACM és capaç de reduir el risc de l'operació quan ocorre una contingència i que en eixe cas, la resposta de l'aeronau segueix sent predicible, fins i tot si el pilot no hi pot intervenir.

Agraïments

Vull agrair la col·laboració de totes aquelles persones i institucions que han fet possible la realització d'aquesta tesi:

En primer lloc, vull fer constar que aquesta tesi ha estat co-financiada pel Fons Social Europeu 2014-2020 i pel programa VALI+d de la Generalitat Valenciana (expedient número ACIF/2016/197).

De forma molt especial, vull agrair a Joan Vila la seua dedicació al llarg de tots aquests anys. El seu recolçament i la seua tasca de direcció han estat fonamentals per dur a terme aquest treball.

També vull agrair a tots els membres del *Department of Unmanned Aircraft* del *German Aerospace Center – Institute of Flight Systems* la seua acollida durant la meua estada a Braunschweig; i en especial a Jörg Dittrich, director del departament; Florian Adolf, supervisor de la estada; i Christoph Torens, per la seua inestimable ajuda.

Finalment, vull agrair a la meua familia el seu suport i afecte (tot i el meu geni durant la recta final!): als meus pares, el millor exemple; a Pau i Ana, pels seus consells i comprensió; a Marc, el més bonic; i a Irene, que val el cel.

Unión Europea
Fondo Social Europeo
El FSE invierte en tu futuro

Contents

List of Figures

List of Tables

Glossary

3T Three-tier

ACAS Airborne Collision Avoidance System

ACM Automated Contingency Management

ADS-B Automatic Dependent Surveillance-Broadcast

AESA *Agencia Estatal de Seguridad Aérea* (Spanish Aviation Safety Agency)

AGL Above Ground Level

AIP Aeronautical Information Publication

AIRAC Aeronautical Information Regulation And Control

AIS Aeronautical Information Service

ALFURS Autonomy Levels For Unmanned Rotorcraft Systems

AMSL Above Mean Sea Level

ANP Actual Navigation Performance

AP/FD Autopilot/Flight Director

API Application Programming Interface

ATC Air Traffic Control

ATM Air Traffic Management

ATS Air Traffic Service

ATZ Aerodrome Traffic Zone

BBN Bayesian Belief Network

BRLOS Beyond Radio Line of Sight

BVLOS Beyond Visual Line of Sight

C2 Command and Control

C3 Communications, Command and Control

ConOps Concept of Operation

CPT Conditional Probability Table

CRC Cyclic Redundancy Check

CTL Computation Tree Logic

CTR Controlled Traffic Region

DAA Detect and Avoid

DLR *Deutsches Zentrum für Luft- und Raumfahrt* (German Aerospace Center)

EASA European Aviation Safety Agency

ELOS Equivalent Level of Safety

EoV Entity of Value

EPT Extended Path Terminator

ERP Emergency Response Plan

ETA Event Tree Analysis

Eurocontrol European Organisation for the Safety of Air Navigation

FAA Federal Aviation Administration

FD Flight Director

FL Flight Level

FMC Flight Management Computer

FMS Flight Management System

FSM Finite-State Machine

FTA Fault Tree Analysis

FTE Flight Technical Error

FTP Flight Termination Point

FTS Flight Termination System

GNSS Global Navigation Satellite System

GPS Global Positioning System

GUI Graphical User Interface

HALE High-Altitude Long-Endurance

HMI Human-Machine Interface

IAF Initial Approach Fix

ICAO International Civil Aviation Organization

IFR Instrument Flight Rules

IMA Integrated Modular Avionics

IMU Inertial Measurement Unit

INE *Instituto Nacional de Estadística* (Spanish Statistics Institute)

JARUS Joint Authorities for Rulemaking on Unmanned Systems

LIDAR Laser Imaging Detection and Ranging

LNAV Lateral Navigation

LOAT Levels of Automation Taxonomy

MAC Mid-Air Collision

MALE Medium-Altitude Long-Endurance

MBD Model-Based Design

MCP Mode Control Panel

MIDCAS Mid-Air Collision Avoidance System

MiPlEx Mission Planner and Execution

MMS Mission Management System

MOPS Minimum Operational Performance Standards

MTOW Maximum Take-Off Weight

NAA National Aviation Authority

NASA National Aeronautics and Space Administration

NATO North Atlantic Treaty Organization

NAVAID Navigation Aid

NEST Network Strategic Modelling Tool

NMAC Near Mid-Air Collision

NOTAM Notice To Airmen

NSE Navigation System Error

OOP Object-Oriented Programming

OS Operating System

PBN Performance-based Navigation

PDE Path Definition Error

RAIM Receiver Autonomous Integrity Monitoring

RCP Required Communication Performance

RLOS Radio Line of Sight

RNAV Area Navigation

RNP Required Navigation Performance

RPA Remotely Piloted Aircraft

RPAS Remotely Piloted Aircraft System

RPASP Remotely Piloted Aircraft System Panel

RWC Remain Well Clear

SAA See and Avoid

SAIL Specific Assurance and Integrity Level

SARPs Standards and Recommended Practices

SESAR Single European Sky ATM Research

SI *Système International (d'unités)* (International System of Units)

SKE Separation Kernel Emulator

SMMS Safe Mission Management System

SMV Symbolic Model Verifier

SORA Specific Operation Risk Assessment

SSR Secondary Surveillance Radar

TCAS Traffic alert and Collision Avoidance System

TLC Target Language Compiler

TLS Target Level of Safety

UA Unmanned Aircraft

UAS Unmanned Aircraft System

UASSG Unmanned Aircraft System Study Group

UDP User Datagram Protocol

UML Unified Modeling Language

UPV *Universitat Politècnica de València* (Technical University of Valencia)

USAL UAS Service Abstraction Layer

UTM UAS Traffic Management

V&V Validation and Verification

VFR Visual Flight Rules

VLOS Visual Line of Sight

VNAV Vertical Navigation

VOR Very High Frequency (VHF) Omni-Directional Range

WGS World Geodetic System

XML Extensible Markup Language

Chapter 1

Introduction

1.1 Problem statement

Day after day, technological development and scientific research are enabling higher levels of automation in the global industry. Manufacturing, transportation, energy production, health care, etc: almost every industrial sector exploits the automation technology available on the market today.

Aviation has certainly seen a gradual transfer from control functions traditionally performed by human operators to automated systems [32, 139, 143, 173]. Moreover, new automation technology has been adopted while maintaining or even increasing the *safety level*. Safety is paramount in aviation. For this reason, it must be accounted that besides there is a potential positive impact of automation on safety, this positive impact will necessarily be subject to more demanding (safety) requirements on the system design. That is, automated flight will only be safe if flight automation technology is developed following safe design methodologies based on certification standards. This work will analyze how to safely enable higher levels of automation in aviation from the point of view of the airborne software.

Current automation technology in aviation is able to automatize the full flight profile provided that the operation is performed as initially intended. However,

when some non-nominal condition occurs (e.g. adverse weather, a conflicting traffic or a technical issue with the aircraft), the human pilot is often required to handle the situation. One of the current research trends is thus introducing *Automated Contingency Management (ACM)* functions in aviation. Such functionality can play an important role to ensure safety in modern aviation: on one hand, ACM functions can assist the pilot-in-command after a *contingency* occurs, shortening the pilot's reaction time; on the other hand, ACM functions can even fully replace the pilot-in-command during the contingency handling if this is required by the situation or if the pilot decides so.

An important phase of contingency handling is *decision making*. Assisting the pilot in this process is key, since pilot's reaction time in case of an adverse event is a safety-critical parameter [175]. This is even more important in modern aircraft operating with a high degree of automation because, in these cases, pilots often play a supervisory role that has been credited to reduce their *situational awareness* [33, 137, 175]. Recovering the situational awareness after a pilot's distraction is difficult, all the more in the event of a contingency. So providing some degree of assistance to the pilot or even fully automating the contingency handling (with the pilot's approval) can notably increase the probability of success of a contingency handling strategy.

Besides a high degree of automation can also bring new hazards related to the automated functions, the increase of the automation level is a general trend in modern aircraft. For example, in recent years, the Airbus company developed the Autopilot/Flight Director (AP/FD) Traffic alert and Collision Avoidance System (TCAS) [9] which is able not only to make a decision in case of a collision threat, but also to execute the avoidance maneuver with the pilot's approval. A more controversial example would be the idea of introducing *single-pilot operations* [36]. Under this idea, some sort of ACM functionality should be able to assume full aircraft control in case of sudden pilot incapacitation.

But the most significative application exemplifying the need of higher levels of automation in aviation is the operation of Unmanned Aircraft System (UAS). Due to the nature of these aircraft, ACM functions are utterly important for ensuring an acceptable safety level. This is the main research topic of this work.

1.1.1 UAS operations and the need for contingency management

As it is well known, the number of applications relying on the use of UAS is increasing quickly [58, 63]. Typical examples of UAS missions include surveillance, image acquisition, or firefighting, among many others. Certainly, some of these missions have been traditionally flown using manned aircraft. However, there are good reasons for which the use of UASs may be beneficial for the operator. To start with, some of these missions often involve dirty, dangerous, or dull tasks (too long, or too tedious for a human crew) [34]. Another important reason is the reduced operational cost of a UAS operation as compared to a conventional one [71, 161]. But probably the most determining factor is *safety*. In this regard, it is necessary to account that, unlike manned aircraft, a UAS mishap does not necessarily have the potential to cause the loss of human lives. As a consequence, the *severity* of some risks or failure conditions can be lowered in a safety risk assessment. This work will analyze in detail safety aspects related to the UAS operation.

The increased number of UAS operations is challenging the Air Traffic Management (ATM) system, though. The ATM system must ensure safe and efficient aircraft operations through the provision of facilities and services in collaboration with all the involved actors [83]. Up to present day, the common approach for ensuring the safe operation of UAS consists of restricting the airspace volume where the mission has to be technically confined. Using operational restrictions and airspace segregation, civil aviation authorities guarantee that UAS remain well clear of all other traffic while airborne. However, given the huge potential of the UAS market, the industry is trying to make progress on the full insertion of UAS into the civil airspace.

To this aim, the International Civil Aviation Organization (ICAO) published the guidance material in Doc. 10019 AN/507. From this document, it can be extracted that "only unmanned aircraft that are remotely piloted could be integrated alongside manned aircraft in non-segregated airspace and at aerodromes" [82]. A Remotely Piloted Aircraft System (RPAS) is a subclass of UAS that excludes all those *autonomous* vehicles for which no human action is necessary after take-off. According to ICAO, the term RPAS includes the Remotely Piloted Aircraft (RPA), its associated remote pilot station(s), the required Command and Control (C2) links between the remote pilot and the RPA, plus any other components specified in the type design [82].

In addition, there is broad agreement that RPAS that aim to operate in non-segregated airspace need to [58, 63, 82, 143]: *1)* demonstrate an Equivalent Level of Safety (ELOS) to that of a manned aircraft, *2)* operate in compliance

with existing aviation regulations, and *3)* appear transparent to other airspace users. The primary goal is that the overall accident rate attained by the current ATM system is not increased with the introduction of equivalent civil RPAS.

In order to achieve the previous objective, it becomes necessary to mitigate the specific risks of RPAS operations as compared to manned aviation [94]. The obvious difference between manned and unmanned aircraft is that the pilot-in-command and the RPA are not co-located. Due to this fact, the following specific risks arise: *1)* reduced situational awareness of the remote pilot, and *2)* risk of losing the C2 link between the remote pilot and the RPA. In the former case, reduced situational awareness means that remote pilots, unlike pilots of manned aircraft in visual conditions, have reduced perception of environmental elements and events. This complicates piloting and decision-making tasks, especially during an abnormal or emergency state. In the latter case, the C2 link loss is a degradation or failure of the communication channel after which the RPA can no longer be controlled by the remote pilot. In this condition, the aircraft is considered to be "flying not under command" [82].

There exist three complementary approaches to mitigate the previous risks: *1)* setting operational restrictions on the RPAS mission, *2)* imposing certain functional requirements related to the on-board equipment, and *3)* by means of operational flight planning and the development of operations manuals with provisions for contingency handling.

Operational restrictions can be used to reduce the risk exposure. For example, geographical restrictions can be approved to limit the airspace volume where the operation is expected to take place. This is the primary risk mitigation measure applied to unmanned aircraft. However, it is contrary to the full RPAS integration objective considered in this work. Time restrictions can also be used to limit the RPAS operation to a time of day when manned aircraft do not routinely fly.

Functional requirements related with the on-board system can also be imposed to compensate for the absence of an on-board pilot. In general, these requirements exemplify the need for increased automatic or even autonomous flight capabilities on-board the RPAS. This is the main research hypothesis of this work. For example, a Detect and Avoid (DAA) capability is often required to counteract the reduced situational awareness of the remote pilot if the operation is to be performed in the vicinity of manned aircraft [82, 95]. Moreover, even though the RPAS category excludes completely autonomous vehicles, aviation authorities can require an *autonomous mode* to handle the C2 link loss in some circumstances [55, 82]. Note that this operational mode is not a desirable

flight condition, but it may be preferred over an out of control condition or even *flight termination*.

If we go one step further, the reader can easily realize that any combination of abnormal events or *contingencies* can happen along with the C2 link loss. When this occurs, some sort of *Automated Contingency Management (ACM)* function is necessarily required to handle the situation. Contingency Management refers to the ability to handle contingencies to maintain an acceptable level of safety during the entire flight. Automated Contingency Management implies that this ability is performed by the embedded software on-board the RPAS. This work will study the implications of the ACM functions on the design of the embedded system.

1.1.2 RPAS flight planning and replanning

As it was introduced before, in order to gain authorization to operate in non-segregated airspace, RPAS must adhere to airspace rules and procedures approved by civil aviation authorities. This includes, among other things, the use of *flight plans* for traffic planning and traffic control purposes. The flight plan concept used in manned aviation specifies information relative to an intended flight, such as the type of flight, flight rules, departure and arrival aerodromes, intended route, etc. This information is provided to Air Traffic Service (ATS) units so that the flight is conducted safely and efficiently. The flight plan file is standardized by ICAO in Doc. 4444 [83]. Both the European Organisation for the Safety of Air Navigation (Eurocontrol) in Europe and the Federal Aviation Administration (FAA) in the United States have adhered this model.

Due to the huge potential of flight planning in regards to contingency handling, RPAS *Mission Plans* will also be discussed in detail in this work. On this subject, we believe that conventional flight plans cannot capture the specificities of RPAS missions. To start with, most RPAS missions are quite different from typical "transport missions" between one origin and one destination. In general, it is difficult to define a "typical" RPAS profile [138, 144]. For this reason, RPAS flight planning should be as versatile as the RPAS operation is required to be.

In addition, ICAO Doc. 10019 AN/507 explicitly specifies that RPAS flight planning should include provisions for contingency handling [82]. In conventional aviation, alteration of the approved trajectory in flight time is unusual, so conventional flight plans do not cope with it. The only aspect that conventional flight plans take into consideration is the "alternate airport". However,

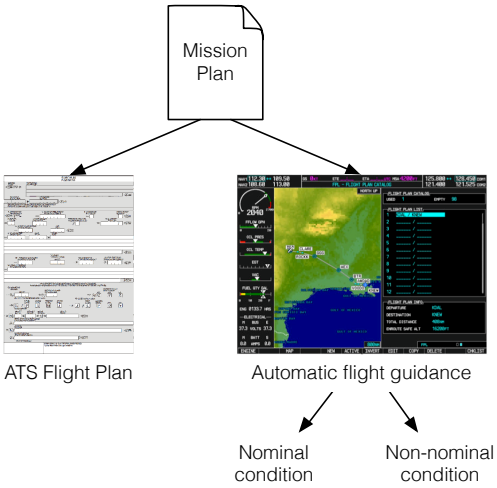


Figure 1.1: RPAS Mission Plan concept.

the aircraft route in case of some contingency is not specified any further, nor is it automated in flight plans of manned aviation: it is the responsibility of the pilot-in-command to make a decision and execute it.

When it comes to RPAS operations, due to the reduced situational awareness of the remote pilots, they rely to a much greater extent on pre-planning abnormal and emergency scenarios that may occur along the intended route of flight. Moreover, since the RPAS can eventually fly in a completely autonomous way and, under this condition, any alternate route must be still predictable, we believe that all the possible alternate plans need to be specified in a detailed manner pre-flight. In fact, ICAO Doc. 10019 AN/507 states that "procedures for the loss of the C2 link for RPA conducting controlled flights should be pre-approved by the Air Traffic Control (ATC) units involved in each portion of the flight planned route" [82]. In case that the RPAS is operated in semi-automatic or manual mode, alternate plans can also be used to suggest possible options to the remote pilot.

As a result, we propose deriving the RPAS *Mission Plan* concept as a generalization of the flight plan concept used in manned aviation. In our view, RPAS Mission Plans should serve additional purposes: *1)* to provide automatic flight guidance along all the phases of flight, and *2)* to specify RPAS behavior in case of abnormal flight conditions so as to be predictable and suitable for automatization in case of a C2 link loss, see Fig. 1.1

To meet the previous goals, RPAS Mission Plans should fulfill new requirements: *1) RPAS Mission Plans should be flexible to allow different mission profiles to be specified.* This accounts for the fact that RPAS missions have a wide variety of profiles. *2) RPAS Mission Plans should specify the flight segments to be covered in controlled and non-controlled areas.* This is because Mission Plans serve automation purposes too. In addition, RPAS can fly under non-conventional ATC services not included in controlled areas. One example is the National Aeronautics and Space Administration (NASA) proposal for the airspace below 400 ft known as UAS Traffic Management (UTM) [104]. Another example would be an ATC unit specifically for the operations area, similar to the one used to access the operations area in firefighting. *3) RPAS Mission Plans should allow for dynamic trajectory replanning.* RPAS missions usually have a preferred or nominal route, but contingencies or some other flight conditions may require the current plan to be abandoned and replaced by an alternate one. Since the remote pilot may be out of the control loop due to a C2 link loss, all allowed flight segments must be specified pre-flight. The action of replacing the current route with a new one in flight time will be termed mission *reconfiguration* or *replanning*.

Based on these requirements, this work will develop the formal specification of a RPAS Mission Plan than enables reconfiguration, contingency handling and higher levels of automation and pilot assistance.

1.2 Objectives and methodology

Based on the previous discussion, the general objective of this thesis is to analyze the feasibility of increasing the level of automation of aircraft operations by providing the automatic system on-board the aircraft with the ability to manage contingencies in an automatic or semi-automatic manner. Although this work is specially focus on *unmanned* aircraft, and particularly on RPAS, the proposed analysis is extensible to aircraft in general, including manned aircraft. In fact, we believe that Automated Contingency Management could be an interesting approach towards single-pilot operations, one of the current research trends in commercial aviation.

In order to achieve the main research objective, this work will analyze the safety risks derived of the intended operation, and will propose a technological solution that is intended to maintain an acceptable level of safety even under the effect of contingencies. To this aim, we will use the current flight automation technology of manned aviation as a baseline. In particular, the Flight

Management System (FMS), the automatic flight guidance and control system used in commercial airliners, is able to control the aircraft route as long as the aircraft is flying in a nominal condition. It is a mature and robust technology that is implemented on-board a wide variety of aircraft models operating all over the world; so we propose making this technology to evolve in order to suit specificities of unmanned aircraft.

Accordingly, this work will address the following specific research objectives:

RO1 To analyze the safety risks posed by RPAS operations. Based on the identified risks, to analyze the risk mitigation mechanisms required to ensure the safe insertion of RPAS in the ATM system.

RO2 To derive the software requirements of the automatic system on-board the RPAS.

RO3 To develop a Mission Plan specification that is adequate for defining the specificities of RPAS missions.

RO4 To design a high-level software architecture for the automatic system on-board the RPAS. The software architecture must address the software requirements of **RO2**. The automatic system must be capable of performing a RPAS mission defined using the Mission Plan specification of **RO3**.

RO5 To develop a prototype model of the software architecture of **RO4**.

RO6 To evaluate the effectiveness of the risk mitigation mechanisms implemented on the prototype system running in a simulation environment.

In addition, it is necessary to account that avionics applications like the one developed in this work are subject to rigorous development and verification standards. In particular, software design methodologies for critical software used in avionics must follow the guidelines and activities defined by the DO-178 standard [131]. The process of developing applications meeting the objectives of this standard can be time-consuming and expensive. It is not in the scope of this work to deal with the certification process. In any case, we will follow the design guidelines of this document as much as possible.

In this regard, one of the key issues of any software design methodology is the *deployment phase*. It deals with the run-time environment of the application and the process for porting the software model to this environment. The run-time environment for the avionics of the last generation of airliners (A350,

A380, B777, B767) is based on the Integrated Modular Avionics (IMA) concept, which is standardized by ARINC-653 [6]. IMA architectures structure the system as a network of *partitions*, where each partition provides protection and separation among applications running on the same hardware. Partitioning is one of the mechanisms allowed by the DO-178 standard to contain and isolate faults and potentially reduce the effort of the software verification process [131].

One of the research lines of the research group of the authors is the development of XtratuM: a hypervisor for real-time embedded systems that is compliant with the ARINC-653 standard [113]. This poses an additional research objectives for this thesis:

RO7 To deploy the prototype system of **RO5** to a run-time environment based on the XtratuM hypervisor.

In order to facilitate and accelerate the deployment process to XtratuM, we will develop a suit of tools and methods that allow to automatize this process.

1.3 Thesis outline

This thesis is organized in eight chapters including the present one (Chapter 1). Chapter 2 provides the background to this thesis, including a summary of current UAS regulation and a review of similar works in the literature. Chapter 3 introduces the Specific Operation Risk Assessment (SORA) methodology and describes the Concept of Operation (ConOps) that will be analyzed in this work. Chapter 4 develops an on-board Mission Management System software architecture with provisions for contingency handling. Chapter 5 introduces the concept of Reconfigurable Mission Plans for RPAS. Chapter 6 develops the probabilistic risk model that will be used to evaluate the proposal in this work. Chapter 7 validates the different contributions of this work. Finally, Chapter 8 summarizes the main findings and provides possible future lines of research.

In addition to the previous chapters, five appendices provide implementation details of the different prototype models developed during this work. In particular, Appendix A describes the formal design and verification of the proposed contingency management policy. Appendix B provides an overview of the Mission Management System implementation in Matlab/Simulink. Appendix C develops the algorithms for dynamic route reconfiguration. Appendix D describes a series of tools that allow to automatize the process of porting designs in Matlab/Simulink to a target system based on the XtratuM hypervisor. Finally, Appendix E provides the probabilistic risk model parameters.

Chapter 2

Background

2.1 Introduction

This chapter provides the background to the remaining research in this work. In particular, Sec. 2.2 outlines current UAS regulation; Sec. 2.3 shows current trends in UAS contingency management; Sec. 2.4 describes the automation taxonomy used in this work; Sec. 2.5 introduces some relevant software architectures for mission control; and finally Sec. 2.6 describes different RPAS Mission Plan specifications that can be found in the literature.

2.2 UAS regulation

Although the development of civil UAS is relatively recent, the use of unmanned aircraft date back to the mid-1800 [47]. The work in [150] provides a historical overview of the UAS sector. In regards to the legal framework applicable to these aircraft, international civil aviation is regulated according to the Chicago Convention of 7 December 1944 [84]. After this convention, ICAO became the United Nations' specialized agency in charge of developing the international Standards and Recommended Practices (SARPs). ICAO SARPs are grouped into 19 Annexes which, in conjunction, provide guidance to Con-

tracting States for developing their legally-enforceable national civil aviation regulations.

With respect to *unmanned aircraft*, Article 8 of the Chicago Convention already referred to "pilotless aircraft" [84]. But it was not until the introduction of Circular 328 AN/190 in 2011 when the term "unmanned aircraft" was adopted by ICAO [81]. In particular, Circular 328 AN/190, developed by the Unmanned Aircraft System Study Group (UASSG), used this term to refer to any type of aircraft without a pilot on board (including UAS and RPAS, where RPAS is a subset of UAS). This document was the first step towards the integration of UAS into the civil airspace.

Afterwards, the Remotely Piloted Aircraft System Panel (RPASP) continued the work by the UASSG and developed the "*Manual on Remotely Piloted Aircraft System (RPAS)*" (Doc. 10019 AN/507) [82]. This document, first published on 2015, extends Circular 328 AN/190 and explicitly specifies that "only unmanned aircraft that are remotely piloted could be integrated alongside manned aircraft in non-segregated airspace and at aerodromes" [82]. In addition, "in order for RPAS to be widely accepted, they will have to be integrated into the existing aviation system without negatively affecting manned aviation (e.g., safety or capacity reduction). If this cannot be achieved (e.g., due to intrinsic limitations of RPAS design), the RPA may be accommodated by being restricted to specific conditions or areas (e.g., Visual Line of Sight (VLOS), segregated airspace, or away from heavily populated areas). This will be one of the main guiding principles of this thesis.

In present days, it is considered that most of the existing SARPs are directly applicable to RPAS; but others will have to be revised, amended, or enhanced by the RPASP to define the manner in which RPAS must comply [82]. It is envisioned that this will be a gradual process, with new SARPs being adopted as technology and experience mature.

European regulation

The European Aviation Safety Agency (EASA) is the European agency which supports its Member States in implementing the ICAO standards; coordinates their auditing activities; and promotes common positions on matters addressed at global level. For this reason, the European Commission tasked EASA to

develop the regulatory framework for *drone*[1] operations in Europe. The founding work (Regulation (EC) No 216/2008, commonly referred to as *Basic regulation* [119]) was approved by the European Parliament on February 2008. However, its scope was limited to drones with a Maximum Take-Off Weight (MTOW) above 150 kg that are not used for military, customs, police, firefighting, search and rescue or experimental work. The remaining drone operations should therefore be regulated by dedicated national aviation legislation.

Consequently, several Member States developed their own regulation. Some common principles like categorization based on mass criteria or operational limitations like visual line of sight and altitude limitations were present in almost all cases. For example, the Spanish regulation defined three drone categories as a function of the aircraft MTOW [19]. A comparison of different national regulations can be found in [44]. The resulting situation was not satisfactory, though [44]. On one hand, European legislation was not harmonized along Member States and there was no obligation on mutual recognition of certificates. On the other hand, operations affecting multiple Member States would require multiple authorizations.

To overcome this, EASA developed the *Concept of Operation for Drones* [47]. The main idea behind this document is that regulation should be operation-centric, proportionate, risk- and performance-based, what is in line with the Roadmap for the integration of civil RPAS into the European Aviation System [58]. With this aim, EASA proposed to establish three *categories of operations* (named "open", "specific" and "certified") and their associated regulatory regime. An overview is shown in Table 2.1. In short, the *open category* is for low risk operations where safety is ensured through compliance with operational limitations, mass limitations, product safety requirements and a minimum set of operational rules. Authorization from a National Aviation Authority (NAA) is not required. The *specific category* is for medium risk operations and requires NAA authorization based on a risk assessment performed by the operator. A manual of operations must list the risk mitigation measures used to enforce the required safety level. Finally, the *certified category* is for large UAS flying in non-segregated airspace, the requirements for which are comparable to those for manned aviation. Safety objectives for this category must be derived from [94].

In order to develop the new regulatory framework according to the previous categories of operation, EASA opened two consultation processes on July 2015

[1] EASA uses the term "drone" to refer to any aircraft without a human pilot on board, whose flight is controlled either autonomously or under the remote control of a pilot on the ground or in another vehicle.

Table 2.1: EASA's Concept of Operation for drones.

Open category	Specific category	Certified category
$MTOW < 25$ kg; and $h < 120$ m; and in VLOS; and Outside reserved areas	$MTOW \geq 25$ kg; or $h \geq 120$ m; or BVLOS	Risks like manned aviation Size, complexity, kinetic energy
No certification	SORA	Full certification

and May 2017 (see A-NPA 2015-10 [44] and NPA 2017-05 [48]), the outcome of which were captured in the corresponding technical opinions (document [51] published on December 2015, and document [49] from February 2018). These documents are specially focused on the open and the specific categories; while the effort of this thesis goes into the specific and the certified categories. At the time these lines are written, the new regulatory framework is still pending to be adopted by the European Commission.

One of the main novelties of the first technical opinion [51] is that it introduced the concept of *Specific Operation Risk Assessment (SORA)*: the safety risk assessment process that the operator must perform to gain the operation authorization for the specific category. The SORA methodology was established by the Joint Authorities for Rulemaking on Unmanned Systems (JARUS) Working Group 6 on June 2017 [95]. Since UAS risk assessment and the determination of the corresponding mitigation measures is the major research topic of this work, the SORA process will be further discussed in Chapter 3.

Other regulatory approaches

In the United States, the FAA and the U.S. Department of Transportation are also taking an incremental approach towards safe integration of UAS into the National Airspace System [63]. Since 1990, civil operations were only authorized on a case-by-case basis and relying on airspace segregation. In present days, there exist two specific UAS rules: a first one on registration and marking requirements for UAS weighting between 0,55 and 55 pounds [60], and a second one enabling routine small UAS operations in VLOS [59]. A review of other regulatory approaches here omitted for brevity can be found in [158].

2.3 UAS contingency management

There is an important research effort behind the previous regulatory proposals. The main ATM research framework in Europe is the Single European Sky ATM Research (SESAR) work programme [144]. In document [142], SESAR identifies separation provision/collision avoidance, C2 link loss policies, and communication with ATM as the three main research areas that must be addressed before RPAS integration can be demonstrated to be safe. Separation provision/collision avoidance and C2 link loss policies are directly related with contingency management and will be discussed in detail in this work. Here we provide a short literature review of relevant works in these fields.

Separation provision/collision avoidance in UAS mainly relies on the DAA capability. Two of the main research initiatives behind the DAA capability are the Mid-Air Collision Avoidance System (MIDCAS) project [125] and the Airborne Collision Avoidance System (ACAS) X [62], an evolution of the current ACAS versions used in manned aviation. ACAS X includes four particular variations, named X_A, X_O, X_U and X_P, where ACAS X_U is the specific version for unmanned aircraft. The work in [111] describes the specificities and challenges to the ACAS X_U system. In all cases, the main research problem is reaching fully automated collision avoidance maneuvers using non-cooperative sensors. Although this work does not deal with the technical barriers of the DAA systems, we will allocate the DAA functionality into the on-board software architecture here studied. We will also assess the risk reduction achieved when a DAA system compliant with the operational performance required by SORA is available.

When a DAA system compliant with the required performance is not available, airspace segregation or advanced tactical separation mechanisms are necessary. An interesting approach can be found in [127, 128]. These works provide a methodology for airspace design and safe flight planning that is intended to maintain the level of safety in current civil airspace even after the insertion of RPAS.

Apart from the loss of separation condition, the other limiting factor towards the safe insertion of RPAS in civil airspace is the C2 link performance, or ensuring that the C2 link loss event does not significantly increase the risk entailed to third parties. Multiple works in the literature have addressed the problem of designing safe C2 link loss policies. As an example, the work in [99] presents a method for computing optimal lost-link policies for unmanned aircraft conducting surveillance alongside manned aircraft in a wildfire scenario. However, in view of full RPAS integration, it is essential to consider C2 link

loss policies from the point of view of the ATM system. In this regard, acceptable policies must ensure predictability and compliance with existing rules for manned aviation. The works in [52, 65, 105, 126] evaluate lost-link procedures assuming the interaction between the RPAS and ATC.

Another interesting approach in the field of safe and predictable C2 link loss policies is the work in [123]. It follows a similar approach to the one in this thesis as it assumes that predictability must necessarily rely on pre-planning contingency scenarios and subsequent reactions. However, the solution in [123] is based on the funneling strategy (similar to the *point merge* concept [43]) which introduces some degree of uncertainty in the aircraft route. Our proposal reduces the uncertainty as all possible flight segments will be explicitly specified pre-flight.

Apart from the previous contingencies, if we consider that the current approach for accommodating UAS operations in civil airspace is airspace segregation, the violation of the mission boundary limits condition must also be discussed in detail. The mission boundary limits violation is commonly prevented through the use of geo-awareness systems or *geofencing*. Several works in the literature have developed geo-awareness systems for UAS operations [31, 38, 74]. One of the most interesting approaches in this field is the work in [149], which applies *formal methods* to perform on-line monitoring checks. In this thesis, we will also exploit formal methods to develop the contingency management architecture under discussion.

2.4 Automation levels

As it was mentioned in the introductory part of this work, one of the key design questions in automation is defining the appropriate *automation level* of an automated system. Much research has been devoted in the literature to address the *levels of autonomy* of an automatic system and the interaction between the automatic system and a human operator [28, 42, 76, 98, 117, 141, 147, 152, 173]. The work by Sheridan [152] is one of the seminal ones that defined an autonomy scale. The Sheridan scale is based on ten automation levels, where each level defines a certain grade of interaction in terms of decision and action selection, see Table 2.2: "Level 1" means manual control and "Level 10" is fully automatic (i.e. *autonomous*) behavior.

Other works have defined automation scales for specific purposes. For example, the work in [141] describe the automation of driving tasks using a taxonomy with six levels of autonomy. Other scales are specific to UAS missions [28, 76,

Table 2.2: Sheridan Levels of automation of decision and action selection. Source: [152].

Level	Description
1	The computer offers no assistance: human must take all decisions and actions
2	The computer offers a complete set of decision/action alternative, or
3	Narrows the selection down to a few, or
4	Suggests one alternative
5	Executes that suggestion if the human approves, or
6	Allows the human a restricted time of veto before automatic execution, or
7	Executes automatically, then necessarily informs the human, and
8	Informs the human only if asked, or
9	Informs the human only if it, the computer, decides to
10	The computer decides everything, acts autonomously, ignoring the human

98]. Among them, the most widely used taxonomy is the Autonomy Levels For Unmanned Rotorcraft Systems (ALFURS) scale[2] [98]. In short, the ALFURS framework is a multidimensional scale that introduces eleven automation levels according to different functionalities: guidance, navigation and control functions. Note that Levels 3 and above include the ability to perform contingency management functions. UAS with higher automation levels are also capable of performing collaborative missions and group decision-making.

The SESAR programme has also developed a taxonomy scale specifically adapted to the ATM domain [147]. The SESAR Levels of Automation Taxonomy (LOAT) is grouped by four cognitive functions: information acquisition, information analysis, decision and action selection, and action implementation; and, for each cognitive function, there are a number of levels in relation to human performance: from manual task accomplishment ("Level 0") through to full automation ("Level 8").

Among all the possible automation scales, this work will refer to the Sheridan scale in Table 2.2 since it is clear and simple, it fits the purpose of this work, and it is widely accepted in the aviation industry.

[2]Note that, although the ALFURS framework is originally intended for unmanned rotorcraft, it can be used for unmanned aircraft in general too [98].

2.5 Software architectures for mission control

One of the main contributions of this work is the development of a Mission Management software architecture for UAS that provides higher levels of autonomy and mission control. Up to present day, multiple research initiatives are leading unmanned aircraft towards higher automation levels, see a survey in [98]. Among them, the most significant work for this thesis is the automated Mission Planner and Execution (MiPlEx) system developed at the German Aerospace Center (DLR) [2, 3, 39]. This architecture combines the main ideas of the behavior-based paradigm [21, 66] and the Three-tier (3T) architecture developed by NASA [20]. Due to a collaboration between the DLR and the author of this work, the MiPlEx framework will be considered as the reference software architecture for this thesis. This initial architecture will here be adapted to account for RPAS operations in integrated airspace and will be enhanced to perform Automated Contingency Management functions.

The term *Automated Contingency Management* was first introduced by NASA in a research project in collaboration with Impact Technologies LLC and Georgia Institute of Technology [148, 159, 160]. NASA's Automated Contingency Management (ACM) concept is designed to improve the reliability and survivability of safety-critical aerospace systems. However, the NASA approach to ACM differs from the one presented in this work in its focus on control optimization techniques rather than on trajectory replanning. An interesting extension to this approach is the work in [108] where human-machine interface considerations in contingency management are discussed.

Some research works have already addressed the problem of introducing Automated Contingency Management functions into unmanned aircraft. Among them, the most interesting approach can be found in [122, 124]. The authors of these works develop a software architecture for performing UAS missions in non-segregated airspace, called UAS Service Abstraction Layer (USAL), that is conceptually close to the approach in this thesis. For example, contingency management functions rely on two separated software components: a first one that evaluates the system state (here named *Safety Monitor*; *Health Monitor* in [122, 124]), and a second one that performs decision-making upon the occurrence of a contingency (the *Contingency Manager*). However, we will extend the proposal in the cited works in several aspects:

1. We will study the implications of separating monitoring and decision-making from the point of view of the software development process. Such analysis is not provided in the cited works.

2. As a result of the previous analysis, we will identify the Safety Monitor as the safety-critical component of the architecture, so we will assign it the ability to command the flight termination action. In the cited works, the Flight Termination System is engaged by the Contingency Manager.

3. We will detail the decision-making process behind the Contingency Manager. No details are provided in the cited works. Moreover, we will optimize this decision-making process from the point of view of the risk exposure; and we will verify this process using formal methods.

The introduction of a Safety Monitor component like the one in this thesis is also suggested in [70]. The goal of the referenced work, however, is to expand the operational range and raise the autonomy level, rather than contingency handling. Regarding the use of autonomous systems in defense, the compilation work done by the North Atlantic Treaty Organization (NATO) [173] provides a good discussion on the definition of the levels of automation that can be introduced into an aircraft and their associated risks. It covers the challenging legal, ethical, policy, operational, and technical issues of autonomous systems from a multidisciplinary point of view.

2.6 Flight plans and Mission plans

Two of the most widely used models for defining UAS missions are: *a)* a declaration with the list of waypoints of a mission, and *b)* a behavior-based description of the flight procedures of the mission. The first approach consists of setting a number of waypoints and associated commands to define the mission route. This is usually done through a Graphical User Interface (GUI). Navigation commands are used to specify movements to and around waypoints. Payload-related commands are used for setting options like the camera trigger distance or setting a servo value. A survey of autopilots using this type of specification can be found in [23]. On the other hand, the behavior-based paradigm is based on using a set of behaviors, which are a high level description of the flight procedures of a mission plan [1, 15, 97]. Behaviors are structured in a hierarchical way: complex behaviors can be built on top of lower-level ones. Such mission plans are usually system-specific, so they cannot be easily generalized.

Novel control interfaces are also trying to improve the human-machine interface in regards to the trajectory planning. As an example, the work in [72] allow the define the intended actions or trajectories by drawing instructions on a

control screen. Other approach relies on speech and gestures to control the aircraft behavior in real-time [162].

Certainly, none of the previous approaches assumes a controlled airspace where a number of well defined flight procedures are required by navigation charts or databases. In this regard, the works in [145, 146, 155] are more in line with the proposal in this thesis: they aim to enhance the level of automation of a UAS operating in controlled airspace using a Mission Plan specification with semantically richer constructs to enable the definition of more complex flight plans and new UAS-specific features. The authors propose some Area Navigation (RNAV) leg types extensions for complex paths, as well as some control structures for repetitive and conditional behavior. The Mission Plan is formally specified using the Extensible Markup Language (XML). Although these proposals have the same goal as this thesis and share the approach of extending current navigation concepts and technologies, they concentrate on complex routing and do not address in detail the topic of dynamically reconfiguring a flight plan and dealing with contingencies.

A critical aspect when reconfiguring the intended path in controlled airspace is that any route change must be previously agreed with the corresponding ATC unit. The trajectory negotiation process between the airspace user (the remote pilot or the RPA operating autonomously) and ATC is not dealt in detail in this work. The reader is referred to the works in [93, 101, 138] for a detailed discussion on this topic. Of particular interest is the work in [138]: it develops a Human-Machine Interface (HMI) that supports the trajectory management process and communication of trajectory parameters between the ATC unit and the remote pilot. We assume that an equivalent functionality will be available to support the dynamic reconfiguration tools developed in this work.

Chapter 3

Contingency management in UAS

3.1 Introduction to risk assessment in UAS

According to ICAO, safety is “the state in which the possibility of harm to persons or of property damage is reduced to, and maintained at or below, an acceptable level through a continuing process of hazard identification and safety risk management" [89]. Therefore, the fundamental goal of *risk management* is to ensure that the effects of hazards in the system are removed when possible, or mitigated to an acceptable risk level otherwise, to ultimately demonstrate safety. In manned aviation, the applicant can use any *risk management methodology* that is suitable for this purpose. The guidance material in ICAO Doc. 9859 establishes a general methodology that is based on the following steps, schematized in Fig. 3.1:

1. System description
2. Identification of threats and hazards
3. Identification of harms

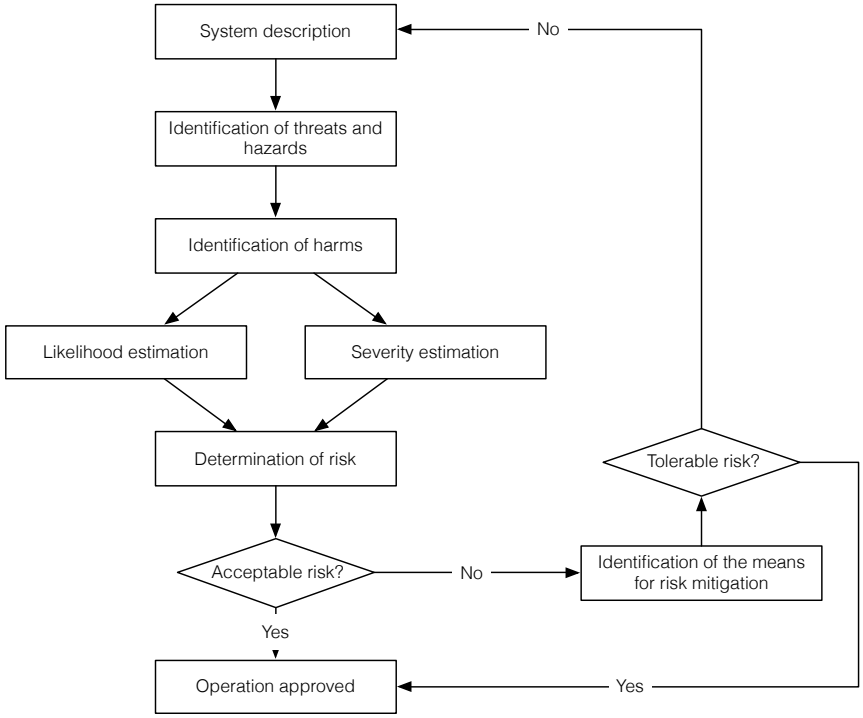


Figure 3.1: Safety risk assessment process.

4. Determination of risk
5. Determination of the acceptable level of risk
6. Identification of the means for risk mitigation

In unmanned aircraft, the risk management process depends on the EASA's *risk categories*. In the *Open category*, the risk is reduced through operational restrictions, product safety requirements and a minimum set of operational rules. Compliance with these rules is assumed to provide an acceptable level of safety, so authorization from a NAA is not required. In the *Specific category*, there exists an increased risk for third parties, so a NAA must determine the acceptability of the proposed operation based on the safety risk assessment process performed by the operator. In the *Certified category*, the risk level is comparable to that of manned aviation, so the authorization is subject to a full certification process.

Performing a risk assessment process is a sensitive and complex task, though. In order to facilitate and harmonize the safety analysis of UAS operations, the Working Group 6 of the JARUS initiative developed the Specific Operation Risk Assessment (SORA) methodology. The SORA methodology basically particularizes the risk assessment steps in Fig. 3.1 to evaluate the risks involved with the operation of UASs of any class and size and for any type of operation [95]. Although it is specially intended for UAS operating within the Specific category, it may be used as an acceptable means of compliance with safety objectives for the Certified category as well [95].

In short, SORA develops an holistic risk model to identify the hazards, threats and relevant threat and harm barriers applicable to a given UAS operation. The SORA model is also used to evaluate the effectiveness of these barriers at reducing the risk entailed for third parties, to ultimately demonstrate whether the proposed operation can be conducted in a safe manner. In this chapter, we will exploit the SORA risk model to identify the relevant aspects of a *contingency management* scheme for UAS. The resulting scheme is considered to set the basis for the research conducted in this thesis. Therefore, we will first discuss the SORA process in some detail; and then, in Sec. 3.2, we will follow the SORA process to characterize the proposed *contingencies*, as well as the possible responses to these contingencies.

3.1.1 System description

The first step of a risk assessment is a description of the system under analysis. In general, the system description should identify the intended functionality, the operational environment or the relevant system interfaces. To do so, the applicant should collect relevant technical, operational and human information related to the use of this system. In the case of a UAS operation, this information is defined by the Concept of Operation (ConOps). According to the SORA methodology, the ConOps should describe at least the type of activity to be performed, MTOW of the UAS, intended flight altitude, airspace class where the operation takes places and time schedule [95]. When it is considered relevant for the operation, the ConOps should also provide technical information related with the UAS such as aircraft structure and performance, software and systems, and relevant functions such as navigation, communication, or command and control. The ConOps can also include safety features identified in further steps of the risk assessment process, like the implemented risk mitigation barriers that will be introduced later on.

3.1.2 Identification of threats and hazards

ICAO Doc. 9859 defines a *hazard* as an event or condition with the potential to cause death, injuries, or damage to equipment or structures [89]. According to the SORA model, the only hazard related to the UAS operation is the "UAS operation out of control", i.e. an operation being conducted outside of the approved limits. For example, an out of control condition occurs when the UAS violates the limits of a segregated area, or if it infringes an altitude restriction. In general, this term is considered to refer to situations in which any of the following conditions hold [95]:

- The outcome of the situation highly relies on providence; or
- The outcome of the situation could not be handled by a contingency procedure; or
- When there is grave and imminent danger of fatalities.

By definition, hazards are caused by potential *threats* when the appropriate *threat barriers* are not in place, or when they result ineffective. The SORA methodology groups all the possible threats of a UAS operation in the following five categories:

1. Technical issue with the UAS,
2. Human error,
3. Aircraft on collision course,
4. Adverse operating conditions, and
5. Deterioration of external systems supporting the UAS operation.

3.1.3 Identification of harms

Once the possible hazards of a particular operational scenario have been identified, the next step is to estimate the credible worst case *harm* that can occur. In the SORA approach, the most severe harms that are supposed to occur after the UAS operation is out of control are classified in the following categories[1]:

[1] Other possible consequences like the environmental damage or the financial impact of an accident are not modeled by SORA, although they could be assessed by means of this methodology as well [95].

1. Fatal injuries to third parties on the ground,
2. Fatal injuries to third parties in the air (i.e., catastrophic mid-air collision with manned aircraft), and
3. Damage to critical infrastructures.

3.1.4 Determination of risk

There exist multiple definitions of "risk" in the literature. One of the most widely used definitions (and the one accepted in SORA) is "the combination of the frequency (probability) of an occurrence and its associated level of severity" [140]. Therefore, in order to assess the risk posed by a hazard, it is necessary to determine two parameters: the severity of the expected outcome and its probability of occurrence.

On one hand, the severity levels associated with the identified hazards are often grouped into qualitative tables. For example, the DO-178 standard establishes five levels of severity ranging from "catastrophic" to "no safety effect" [131]. Then, each possible severity level has an associated list of potential outcomes. A comparison of different severity classification schemes commonly used in the aviation industry can be found in [169], including specific definitions for UAS. In this case, the SORA approach is considered a conservative approach since it is only focused on the "catastrophic" consequences of the "UAS operation out of control", being it fatal injuries to third parties on the ground or in the air, or damage to critical infrastructure.

On the other hand, the probability of occurrence of an event is the average time the event occurs within a particular time interval (usually per hour of operation). In this case, probabilities can be assessed in a quantitative or qualitative manner. However, qualitative ranges like "frequent", "occasional", "remote" events, etc., should be defined in terms of quantitative ranges too. For example, a remote event is one having an average probability between $1 \cdot 10^{-5}$ and $1 \cdot 10^{-7}$ occurrences per hour of operation, for instance. The work in [171] compares different qualitative definitions used in the aviation industry.

Quantitative vs. Qualitative risk models

In the first version of the SORA document, the following risk model was presented as an approach to estimate the likelihood of occurrence of each of the expected harm categories:

$$P_{harm} = P_{ooc}\ P_{strike/ooc}\ P_{harm/strike} \tag{3.1}$$

Where, P_{ooc} is the probability of having the UAS operation out of control; $P_{strike/ooc}$ is the conditional probability of striking a person, an aircraft, or a critical infrastructure (hereinafter, a *third party* for brevity), if the UAS operation is out of control; and $P_{harm/strike}$ is the conditional probability of causing the harm to the third party, if the strike actually occurs. Each of these components should be assessed individually to obtain an estimate of the likelihood of occurrence of a harm category.

A delicate question is whether quantitative models are adequate for risk assessment processes or not. Quantitative methods are powerful tools but are subject to several limitations [95]. To start with, the individual risk perception is often communicated in a qualitative manner. This is because quantitative risks in the form of probability and severity can be hard to understand by the general public. Moreover, the accuracy of quantitative models depends on the uncertainties introduced in the risk model: when it comes to model complex, real processes, mathematical models are by force affected by some degree of uncertainty. The UAS operation is a remarkable example since:

1. It involves interactions between several actors, like Unmanned Aircraft (UA), remote pilot, other traffic, the ATS provider, etc.;
2. It is subject to external factors like human reactions, or weather conditions, among others, that are difficult to model; and
3. Some model parameters like component failure rates, human error rates, or external event rates, are hard to assess or the data is not available.

In short, complexity and uncertainty can make quantitative methods unfeasible in most cases, or the expected model performance might not worth the increased workload of the modeling approach. It is for this reason that JARUS opted for applying a qualitative methodology to the SORA risk assessment; although a complementary quantitative analysis is required in some circumstances [95].

3.1.5 Determination of the acceptable level of risk

It is acknowledged that an absolute level of safety is unrealistic, or a very expensive goal. For this reason, safety risk assessment processes are based on the concept of *risk tolerability*. By risk tolerability, a safety risk can conceptually be assessed as acceptable, tolerable or intolerable depending on the combination of risk probability and severity. According to ICAO [89]:

- Safety risks assessed as intolerable are unacceptable under any circumstances. When a risk of such magnitude is encountered, immediate mitigation action is required. The mitigation action should be effective at reducing the likelihood of occurrence and/or severity level of the hazard outcome so that the updated risk falls within the tolerable region. If this is unfeasible, then the operation should be aborted, or the proposed system description should be revisited (see Fig. 3.1).
- Safety risks assessed as tolerable are acceptable if appropriate mitigation strategies are implemented. In some cases, tolerable risks are acceptable even if no mitigation strategies are implemented, if a cost-benefit analysis justifies that the cost of implementing the corrective actions overcomes the expected benefit (ALARP[2] principle).
- Safety risks assessed as acceptable require no further action in the safety assessment.

Safety risk tolerability is often represented using a safety risk assessment matrix like in Table 3.1. In this matrix, acceptable, tolerable, and intolerable risks are represented in green, yellow, and red, respectively, depending on the tolerability criteria set by the corresponding regulation.

In the aviation industry, the acceptable level of safety is generally defined in terms of the probability of an aircraft accident occurring per flight hour. This parameter is set on the basis of the Target Level of Safety (TLS) required by the corresponding airworthiness regulation. In manned aviation, the TLS is specified according to the aircraft type and complexity. For example, the CS/-FAR 25.1309 standard for large aeroplanes requires the probability of having a catastrophic failure to be extremely improbable (where "extremely improbable" is defined as a likelihood of occurrence of $1 \cdot 10^{-9}$ or less) [46]. By contrast, the CS/FAR 23.1309 standard for normal, utility aeroplanes requires the TLS to be between $1 \cdot 10^{-6}$ and $1 \cdot 10^{-9}$, depending on the type certificate [61]. These

[2] As Low As Reasonably Practicable

Table 3.1: Safety risk assessment matrix example. Acceptable risks are represented in green, tolerable risks in yellow, and intolerable risks in red.

Risk probability	Risk severity				
	Catastrophic	Hazardous	Major	Minor	No effect
Frequent	5A	5B	5C	5D	5E
Occasional	4A	4B	4C	4D	4E
Remote	3A	3B	3C	3D	3E
Improbable	2A	2B	2C	2D	2E
Extremely improbable	1A	1B	1C	1D	1E

quantitative probabilities are defined taking into account the operational context, complexity, accident statistics, etc.

In the case of UAS, current regulation aims at maintaining the accident rate close to that of manned aircraft. Moreover, a principle of risk equivalence between the different risk categories of UAS is to be retained [95]. However, as it was discussed before, SORA is a qualitative methodology, so it does not specify the acceptable safety level in terms of the TLS but using the Specific Assurance and Integrity Level (SAIL) concept. The SAIL represents the level of confidence that the UAS operation will stay under control [95]. This way, based on the assigned SAIL, SORA determines a series of operational safety objectives to be complied with, the activities that might support compliance with these objectives, as well as the evidences that indicates these objectives have been satisfied. In this case, the operational safety objectives represent *threat* and *harm barriers*, which are introduced next.

3.1.6 Identification of the means for risk mitigation

When it is necessary to reduce the risk to an acceptable level, risk mitigation measures shall be incorporated. Risk mitigation aims at:

1. Eliminating the risk; or
2. Mitigating the risk, if elimination is not feasible; or
3. Cope with it, if neither elimination nor mitigation is feasible.

The goal is to lower the severity and/or the probability of a hazard's projected consequence by incorporating defenses or preventive controls [89]. Therefore, there exist two complementary approaches to risk mitigation:

1. Reduce the likelihood of the hazard by incorporating *threat barriers*. A threat barrier aims at reducing the likelihood that a threat can cause a hazard by either: *a)* preventing the threat from developing into the hazard; or *b)* reducing the likelihood of the threat. In this way, threat barriers affect the component P_{ooc} of the risk model in Eq. (3.1).

2. Reduce the consequences of the hazard by incorporating *harm barriers*. A harm barrier aims at reducing the likelihood and/or the severity of the consequences of the hazard, assuming that the hazard has actually occurred. In this way, harm barriers affect the components $P_{strike/ooc}$ and $P_{harm/strike}$ of the risk model in Eq. (3.1).

The identification of the appropriate risk mitigation measure is a difficult task that should be performed on the basis of a detailed system analysis. Fault trees and event trees are often useful in identifying the hazard's contributing factors since an effective risk mitigation measure will have to modify one or more of these factors. In the case of SORA, a list of threat and harm barriers has been collected based on the experience of many experts in the field [95]. The proposed list of threat and harm barriers are gathered in Tables 3.2 and 3.3, respectively. Note that these barriers are grouped based on the threat or harm they help to avoid; so some of them may be repeated in the tables. For example, a threat barrier like the use of safe design methodologies and the compliance with aerospace standards can lower the likelihood of experiencing a technical issue with the UAS, thus reducing the component P_{ooc}. Similarly, a harm barrier like the deployment of an emergency parachute may reduce the impact energy transferred to a third party on the ground, thus reducing the component $P_{harm/strike}$.

Table 3.2: Threat barriers provided in SORA. Source: [95]

Threat category	Threat barrier
Technical issue with the UAS	Ensure the operator is competent and/or proven
	UAS manufactured by competent and/or proven entity (e.g. industry standards)
	UAS maintained by competent and/or proven entity (e.g. industry standards)

	UAS developed to authority recognized design standards (e.g. industry standards)
	C3 link performance is appropriate for the operation
	UAS is designed considering system safety and reliability
	Inspection of the UAS to ensure consistency to the ConOps
	Operational procedures are defined, validated and adhered to
	Remote crew trained and able to control the abnormal situation
	Safe recovery from technical issue
Human error	Operational procedures are defined, validated and adhered to
	Remote crew trained and current and able to control the abnormal situation
	Multi crew coordination
	Adequate resting times are defined and followed
	Automatic protection of critical flight functions (e.g. envelope protection)
	Safe recovery from Human Error
	A Human Factors evaluation has been performed and the HMI found appropriate for the mission
Adverse operating conditions	Operational procedures are defined, validated and adhered to
	The remote crew is trained to identify critical environmental conditions and to avoid them
	Environmental conditions for safe operations defined, measurable and adhered to
	UAS designed and qualified for adverse environmental conditions (e.g. adequate sensors, DO-160 qualification)
Deterioration of external systems supporting the operation	Procedures are in-place to handle the deterioration of external systems supporting UAS operation
	The UAS is designed to manage the deterioration of external systems supporting UAS operation
	External services supporting UAS operations are adequate to the operation
Aircraft on collision course	Strategic conflict management
	External tactical mitigation (e.g. ATC, UTM)
	Internal tactical mitigation (e.g. DAA)

The *Agencia Estatal de Seguridad Aérea* (Spanish Aviation Safety Agency) (AESA) guidelines on the SORA methodology [8] goes beyond the JARUS initiative since it not only identifies the above threat and harm barriers, but also includes a series of mandatory requirements that the operator must ad-

Table 3.3: Harm barriers provided in SORA. Source: [95]

Harm category	Harm barrier
Fatal injuries to third parties on the ground	An ERP is in place, operator validated, and effective Effects of ground impact are reduced (e.g. emergency parachute, shelter, etc.) Technical containment is in place and effective
Injuries to third parties in the air	Providence
Damage to critical infrastructure	An ERP is in place, operator validated, and effective UAS equipped with obstacle avoidance capability Effects of ground impact are reduced (e.g. emergency parachute, shelter, etc.) Specific operation profile designed with consideration to critical infrastructure

dress to gain authorization to operate the UAS in the Spanish territory. These requirements are gathered in Table 3.4.

The last step in the safety assessment process is the verification of the robustness of the proposed barriers. In the SORA approach, the level of robustness of a mitigation action depends on its level of integrity (i.e. the safety gain provided by the mitigation) and on its level of assurance (the proof that the claimed safety gain has been achieved). In general, the required level of robustness of each mitigation increases with the assigned SAIL, although it can be adapted by the corresponding aviation authority. If all the proposed barriers meet the required robustness level, the proposed operation will be considered to meet the required level of safety and will therefore be approved by the corresponding NAA; otherwise, a new application with a modified ConOps will have to be presented.

Table 3.4: Risk mitigation (RM) requirements for UAS operations. Source: [8]

RM1	The UAS shall be equipped with a Flight Termination System in all cases. The operator shall describe the Flight Termination System in the Emergency Response Plan.
RM2	The UAS shall be equipped with a system capable of reducing the energy upon impact if operating in a populated environment or over gathering of people.
RM3	The operator shall have the means to carry out bi-directional communications with the corresponding aeronautical stations in the frequency bands required by reguation.
RM4	The operator shall demonstrate an adequate knowledge of the language or languages used in the communications between the controller and the aircraft, as well as the required phraseology when operating in controlled airspace.
RM5	Equipment must be in place to ensure that the aircraft operates according to the approved limitations, including the area to which the operation needs to be technically contained.
RM6	Lightning or an appropriate paint scheme shall be accounted to increase UAS detectability and conspicuity. Navigation lights and strobe lights will be required when operating in night hours.
RM7	The UAS shall be equipped with a forward-facing camera if the operation is carried out BVLOS of the operator.
RM8	UAS that aim to operate in controlled airspace shall be equipped with a Mode S transponder, unless the operation is performed in VLOS of the operator and the aircraft MTOW is below 25 kg.
RM9	The operator shall have the means to monitor the UAS position at all times while in-flight.

3.2 Contingency Management approach

Before proceeding further, it is necessary to clarify some concepts of the SORA semantic model. According to this model, a UAS may be operating in a *normal state* (nominal), in an *abnormal state* or in an *emergency state*, see Fig. 3.2:

- When operating in the nominal state, the UAS is assumed to be performing the nominal mission which is defined as a planned sequence of operational procedures (not necessarily *standard* operational procedures[3]).
- In an abnormal situation, the UAS is in an undesirable state where "it is no longer possible to continue the flight using normal procedures, but

[3]Most UAS missions aim at performing some payload-related task within an operations area, and the flight procedures performed in this area are not necessarily standard operational procedures (i.e. procedures defined in navigation charts) but UAS-specific procedures. This will be further discussed in Chapter 5.

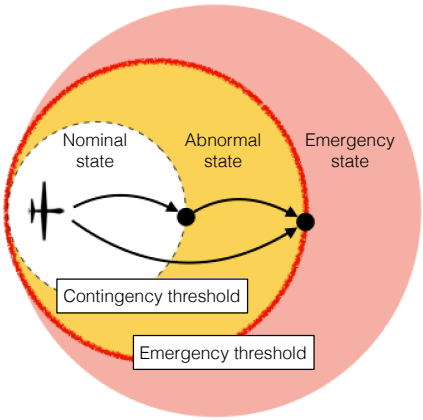


Figure 3.2: UAS operating conditions. Note that thresholds do not necessarily represent geographical areas.

the safety of the aircraft or persons on board or on the ground is not in danger" [95].

- Finally, an emergency is declared when the UAS is an unrecoverable state where there is imminent risk for third parties.

In the SORA semantic model, both nominal and abnormal conditions are considered conditions where the operation is under control; while the emergency situation is an out of control condition, i.e. the hazard identified in the SORA model.

In this approach, we consider that an abnormal situation is produced after a *contingency* occurs. Contingencies are unforeseen events with the potential of causing an emergency. As an example, the works in [16, 17, 75, 172] identified several important contingencies in UAS operations. Some of them are common to manned aviation –like the loss of separation with a transient aircraft or the loss of control–, while others are specific to UAS. Among them, the most remarkable example is the loss of performance of the C2 link. In the same manner, we consider that an *emergency event* is the degeneration of an abnormal situation that may occur if appropriate risk mitigation barriers are not in place, or if they result ineffective in mitigating the effect of the contingency.

The source of contingencies are *faults*. These include both system component faults and human faults. Component faults occur when some aircraft component –such as an Inertial Measurement Unit (IMU), a Global Navigation

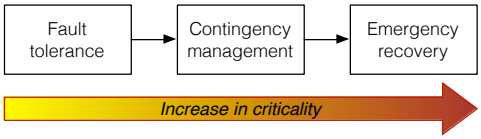


Figure 3.3: Framework of techniques for keeping safety in UAS. Source: [157].

Satellite System (GNSS), or a barometric system– fails. Human faults refer to piloting errors, ATC errors, and any other faults related to inappropriate aircraft operation. There is a causal relation between faults and contingencies. For example, a GNSS fault could cause inaccuracy in position determination, thus resulting in an infringement of the operational volume where the operation needs to be technically contained. Another example is the loss of control, which may be due to a faulty IMU. Faults usually degenerate into contingencies after some short period of time. In the same way, contingencies may degenerate into emergencies after some given time too. For this reason, early fault detection will be key for effective risk management.

As a result, increasing the level of safety in an aircraft is usually accomplished using two complementary techniques: fault tolerance and contingency management, see Fig. 3.3. On one hand, *fault tolerance* is the ability to continue operating in a normal condition in the event of a failure. Therefore, the main goal of fault tolerance is to prevent that simple faults develop into contingencies. A number of fault tolerant methods exist in the literature, including fault-tolerant control and redundancy [157]. Faults affecting critical system components can be tolerated using redundancy. Full redundancy means replicating a component with exactly the same functionality and performance. Graceful degradation is the ability to maintain a limited or degraded functionality when some component fails. For example, GNSS navigation could be replaced with dead-reckoning when the GNSS fails, though at the cost of position accuracy. Fault-tolerant control consists on adapting the controller to a faulty plant; but this is out of the scope of this work.

On the other hand, *contingency management* is the ability to respond to contingencies for the purpose of mitigating safety risks. In particular, contingency management follows a twofold objective: the primary goal is to prevent the abnormal state from developing into an emergency; while the secondary goal is to attempt to recover the nominal condition, if this is considered feasible. To do so, contingency management does not attempt to adapt the aircraft to the situation, like fault tolerance does. Rather, it relies on the execution of *contingency procedures*: flight procedures that define an aircraft trajectory that is

expected to minimize the impact of the contingency being fronted. Therefore, contingency management is considered to mitigate safety risks at a mission level: after a contingency occurs, the current reference trajectory will be replaced with a new one that is based on safety concerns. For example, after a loss of separation with a transient aircraft, contingency management allows to replan the mission to perform an avoidance maneuver; if the separation minima is restored, then the previous mission objective may be resumed afterwards, or a different mission may be planned instead. In general, the response to a contingency will be defined from a predefined set of contingency procedures that *reconfigure* the intended mission in different ways, as established in a *contingency plan*.

Along with fault tolerance and contingency management techniques, there exists a third barrier for enforcing safety which is the *emergency recovery*, see Fig. 3.3. Emergency recovery relies on the execution of *emergency procedures*, which assumes that the emergency situation has already occurred. Emergency procedures aim at limiting the escalating effect of the hazard being fronted by ceasing the flight as soon as possible. In manned aircraft, the emergency procedure is generally an emergency landing in which the pilot tries to reach the nearest landing site (at best, an aerodrome, but not necessarily). In this way, the pilot tries to minimize the safety impact to onboard people, and also to third parties on the ground; although some damage to the aircraft is expected.

In the case of UAS, there exists a specific emergency procedure which is the *flight termination* procedure. Flight termination is the intentional process to end the flight and ground the UAS expeditiously, in a somewhat controlled manner, for example by deploying a parachute, or using a self-destruct device. Upon performing the flight termination procedure, it is expected that the UA may suffer loss or damage, but no additional hazard will be created to persons or property on the ground. As a result, this option becomes crucial in the safety management process because it is an accepted means to minimize the consequences associated with a UAS mishap [8, 50, 118].

A summary of the risk mitigation techniques that are available for keeping safety in UAS is shown in Table 3.5. Note that, from a SORA model point of view, both fault tolerance and contingency management are considered threat barriers since they aim at preventing threats (in this case, faults and contingencies) from developing into hazards; while emergency recovery is a harm barrier since it implies that the hazard has already occurred, and thus it aims at reducing the safety impact of this condition. Note also that these techniques complement each other: when some fault is not tolerated and becomes a failure, then contingency management measures are needed; if contingency

Table 3.5: Summary of techniques for keeping safety in UAS.

	Fault tolerance	Contingency management	Emergency recovery
State coverage	Normal	Abnormal	Emergency
Mitigation approach	Threat barrier	Threat barrier	Harm barrier
Procedure category	Nominal procedure	Contingency procedure	Emergency procedure
Primary goal	Prevent contingency	Prevent emergency	Limit hazard severity
Secondary goal	–	Recover nominal condition	–

management is unfeasible or it becomes ineffective, then emergency recovery is a last resort for enforcing safety. For example, system faults can be tolerated to some extent by the use of redundancy, but human errors and inappropriate aircraft operation can only be handled through contingency management.

Another important aspect is how the different techniques impact the ATC. In general, fault tolerance does not affect the aircraft trajectory, or the resulting effect is small (for example, reduced aircraft performance); so no significant impact on ATC is expected. By contrast, contingency management and emergency recovery imply that the current trajectory is replaced with a new one. If the trajectory replanning involves flight segments performed in controlled areas, then coordination with the corresponding ATC unit is a must. For this reason, they are assumed to have a greater, negative effect on the ATC performance [64].

The main focus of this work is contingency management, and specially Automated Contingency Management (ACM). In this section, we will follow the steps of the SORA risk analysis process in Sec. 3.1 to identify the main aspects of the contingency management scheme used in this work. In particular, we will describe the proposed ConOps, we will enumerate the main contingencies for this ConOps, and we will also identify the contingency procedures that are required to handle these contingencies. These aspects are detailed next.

3.2.1 System description

In order to provide a broad vision of the contingency management problem in UAS, this work is not focused on a particular type of operation. Rather, the proposed ConOps describes a wide range of flight profiles with the following general common features:

- The UAS operation is to be performed Beyond Visual Line of Sight (BVLOS) of the remote pilot. Possible VLOS segments are not discarded, although they are not significant for the whole mission.
- The UAS operation is to be performed under Instrument Flight Rules (IFR). It is generally accepted that UAS operating BVLOS may comply with IFR requirements, but not with Visual Flight Rules (VFR) [57, 95, 174]. Therefore, when airspace requirements impose compliance with VFR, airspace segregation will be necessary.
- The UAS operation may enter in controlled airspace. The operation may also take-off or land at a controlled airport. Therefore, coordination with the corresponding ATC authority is a must. Additionally, the UAS can fly under non-conventional ATC services not included in controlled areas. One example is the NASA proposal for the airspace below 400 ft AGL known as UTM [104]. Another example would be an ATC unit that acts specifically at the operations area, similar to the one used to coordinate the operations in a firefighting.
- The UAS operation is to take place out of urban areas.

As an example, one possible mission described by this ConOps consists of a route going from a departure site (an airport or simply a take-off area) to an operations area; a set of maneuvers within this area; and finally a route to the destination site (which is not necessarily the departure site). The operations area can be a segregated area where the UAS performs some specific work related to the payload (like surveillance, image recording, etc.). This phase of flight is expected to be much more dynamic than the en-route phase as it requires higher pilot intervention: in general, the exact route or the flight procedures performed in this phase will not be anticipated pre-flight. It is then similar in many ways to a military mission.

Due to the inherent complexity of the proposed ConOps, it is assumed that UA models capable of flying these missions will be comparable to manned aircraft in terms of size and complexity. Mission range, or airspace requirements

Table 3.6: IAI Super Heron specifications. Source: [171]

Property	Value
Length	8, 5 m
Wingspan	16, 6 m
MTOW	2.425 lb
Payload weight	551 lb
Powerplant	115 hp
Cruise speed	125 kt
Rate of climb	650 ft/min
Endurance	52 h
Range	190 NM
Ceiling	30.000 ft

Figure 3.4: Super Heron HF model in X-Plane.

with respect to the on-board equipment (e.g. Table 3.4) make it difficult to remain within the MTOW limit of the open category, see Table 2.1. Different Tactical, Medium-Altitude Long-Endurance (MALE) and High-Altitude Long-Endurance (HALE) fixed-wing vehicles that may be suitable for the proposed ConOps, as well as some of their performance parameters, can be found in [171]. As a representative model, the IAI Super Heron UA will be used for demonstration purposes in this work, see Table 3.6 and Fig. 3.4.

In addition, this work assumes that the UAS will be remotely piloted by an operator, called remote pilot, who is responsible for the safe flight. This is required by ICAO, as "only unmanned aircraft that are remotely piloted could be integrated alongside manned aircraft in non-segregated airspace and at aerodromes" [82]. The remote pilot will be located at a remote pilot station. In

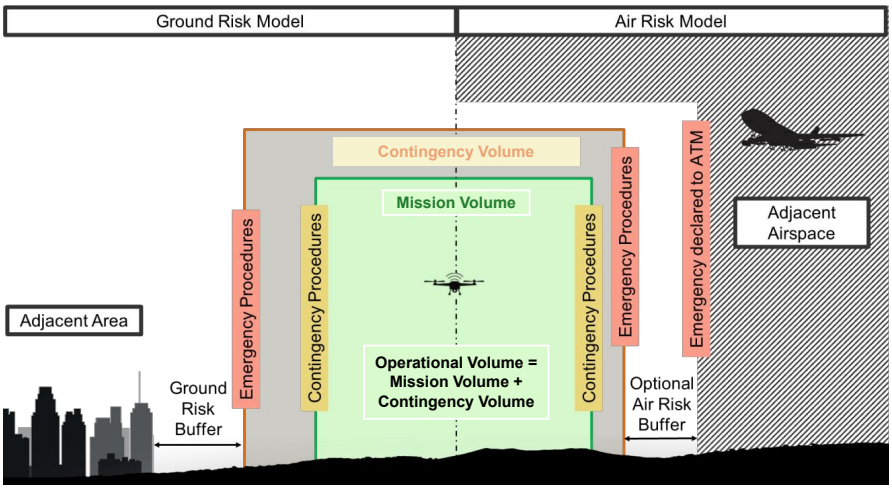


Figure 3.5: Graphical representation of SORA semantic model. Source: [96]

compliance with ICAO Doc. 10019, a remote pilot station will just manage the flight of a single unmanned aircraft; and in case that more than one station is used during the flight, we assume that the handover will be safe and effective [82]. Finally, the communication between the remote pilot station and the unmanned aircraft will be conducted through a C2 data link that will be further characterized in this section. In conclusion, from now on, the UAS will be denoted as RPAS, which includes the RPA, the remote pilot station(s) and the C2 link.

3.2.2 Identification of threats and hazards

In compliance with the SORA model, this works assumes that the only hazard related with the UAS operation (in this case, a RPAS operation) is the "out of control" condition. In order to characterize this flight condition, we will make use of the SORA semantic model: according to this model, the RPAS operation is to be technically contained within a given *operational volume*. In this case, "the operational volume is defined as including both the *flight geography* (i.e. the UA flight path under normal operations) and the *contingency volume* (i.e. the projected UA flight path under abnormal conditions handled through contingency procedures)" [96]. This work will rename the flight geography volume as the *mission volume* for convenience. A graphical representation is shown in Fig. 3.5.

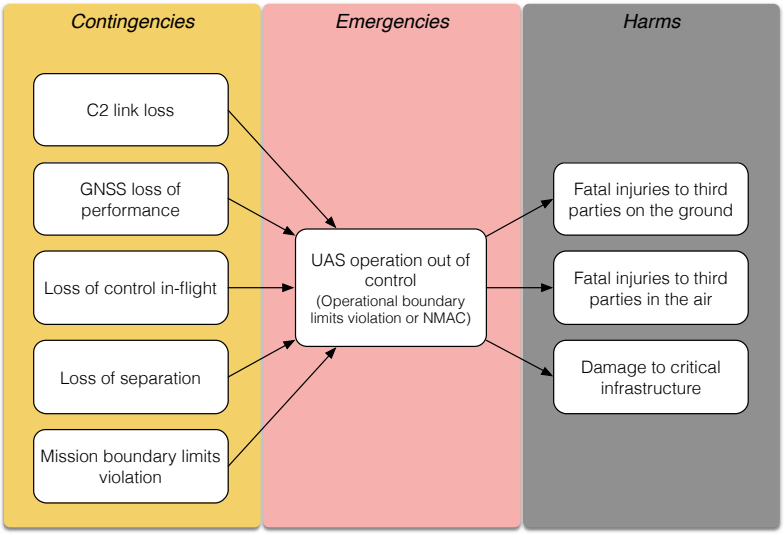


Figure 3.6: Scope of the proposed contingency management framework.

Therefore, "an out of control operation means that the UA is flying out of this operational volume, potentially leading to harm to third parties in the air or on the ground" [95]. In this work, we will extend this definition so that it explicitly specifies that Near Mid-Air Collisions (NMACs) represent out of control conditions too.

As it was motivated in the introductory part of this section, the out of control condition represents an emergency situation that is potentially caused by the occurrence of a *contingency*. In this work, we will consider five *contingency events* that are directly related with the occurrence of the out of control condition. They are the *C2 link loss*, the *GNSS loss of performance*, the *loss of control in-flight*, the *loss of separation* with a transient aircraft and the *mission boundary limits violation*, see Fig. 3.6.

Note that *mission boundary limits violation* and the *loss of separation* conditions can be considered as direct precursors of the *operational boundary limits violation* and the NMAC conditions, respectively; while the remaining contingencies are expected to have a strong contribution to the occurrence of an emergency. The possible effects of these events, their causal relationship as well as their potential consequences are summarized in Table 3.7, and are further discussed next.

Table 3.7: Possible effects, contributing factors and potential outcomes of the proposed contingencies.

Contingency event	Possible effects	Contributing factors	Potential outcomes
C2 link loss	– Inability to command or control aircraft from ground – Inability to monitor aircraft position	– Technical errors – Interferences (e.g. screening terrain, adverse weather, jamming) – Operation out of range	– Mission boundary limits violation – Loss of separation
GNSS loss of performance	– Aircraft position cannot be determined with the required accuracy – Inability to fly desired trajectory	– Technical errors (e.g. satellite error, receiver malfunction) – Weak satellite geometry – Signal propagation error	– Mission boundary limits violation – Loss of separation – Loss of control in-flight
Loss of control in-flight	– Undesired flight trajectory – Unstable control	– Technical errors – Ineffective aircraft control (e.g. operation out of the flight envelope) – Adverse weather	– Mission boundary limits violation – Loss of separation – Ground impact
Loss of separation	– Collision with a transient aircraft	– Human errors (e.g. surveillance, decision-making, piloting) – Degraded navigation	– NMAC – MAC
Mission boundary limits violation	– Aircraft exits assigned mission volume	– Human errors (e.g. incorrect mission definition) – Technical errors – Ineffective aircraft control – Degraded navigation – Adverse weather	– Operational boundary limits violation – Loss of separation – Ground impact

C2 link loss

The Command and Control (C2) link is the data link between the remote pilot and the unmanned aircraft for the purpose of managing the flight in real-time. It may be considered functionally equivalent to the control wires or the databus between the cockpit and the control surfaces of a manned aircraft. Therefore, the C2 link plays a key role in the safety of a RPAS operation, and must be addressed in the certification process. In some implementations, the C2 link is also referred to as Communications, Command and Control (C3) link, if it includes provisions for communication with ATC. However, this will not be considered in this analysis.

The C2 link may be designed for operations in direct Radio Line of Sight (RLOS) or Beyond Radio Line of Sight (BRLOS). The key difference is that C2 links designed for BRLOS operations are based on satellite systems and/or terrestrial networks, so they are considered to introduce an appreciable delay in the communication process. In both cases, the C2 link performance must comply with the Required Communication Performance (RCP) set by regulation. The RCP defines the safety performance of the C2 link in terms of the communication transaction time (the maximum time for the completion of the communication transaction), and in terms of continuity, availability, and integrity [91]. The RCP is set on the basis of the type certificate (and thus on the type of operation) for each different phase of flight in the operation [82]. The RCP may also depend on the different operational modes of the RPAS. For example, operations under manual control are expected to require a higher communication performance than in automatic control. Security requirements are another relevant aspect for the design of the C2 link, although they will not be discussed in this work.

In order to comply with the RCP, redundant configurations may be required. In these cases, dissimilar redundancy is often recommended to increase the robustness of the configuration. However, state of the art C2 link technology may be insufficient to comply with the RCP under all operating conditions, even using redundancy [82]. This is because of the nature of the communication channel, which may be subject to occasional degradation or even loss of the C2 link. In these cases, design constraints or operational mitigations may be required to meet an acceptable performance in terms of safety.

In this work, *C2 link loss* contingency is considered to be any situation in which the remote pilot cannot intervene in the flight's trajectory due to the degradation or total loss of the communication channel. Possible causes include unintended interferences like screening terrain, ocean wave effects, meteorolog-

ical conditions, or human activities, among others. Intentional interferences (e.g. jamming), operation out of range, or failures of the technical equipment are other sources of degradation. This way, threat categories related with the *C2 link loss* event include technical issues with the RPAS, deterioration of external systems supporting the RPAS operation, and adverse operating conditions. These effects may be encountered in a temporary, intermittent or even in a permanent manner.

Another important aspect is that the C2 link mode of failure is not always a fail-stop failure (i.e. a "clean" failure): the C2 link may experience a degradation in which repetitive or intermittent unavailabilities or delays in the transaction time occur. This is considered a "byzantine failure" which causes a distributed system to have imperfect information on whether a component has failed or not. Byzantine failures are the most difficult failure modes to be handled. In the case of the C2 link data link, the byzantine failure may corrupt the transmitted data, resulting in incorrect instructions for the unmanned aircraft or incorrect information displayed to the remote pilot. Instructions can be even received without error, but with a delay greater than the RCP allows. In these cases, there will be a time period beyond which continued flight in this manner may not be considered acceptable, and therefore the C2 link should be declared as being lost.

For simplicity, this work will assume that the failure mode of the C2 link will always be a fail-stop (the simplest failure mode). This implies that there exists a data link health monitoring function that is always reliable, and that this function is able to convert a byzantine failure into a fail-stop failure. This is not unrealistic, though. Erroneous messages can be detected and removed using integrity mechanisms like Cyclic Redundancy Checks (CRCs). In the case of triple redundant data links, voting algorithms can be implemented to detect and isolate the faulty channel. A more conservative option could be declaring the C2 link as being lost whenever a link degradation is detected, and then check if the RCP has been restored after a given period of time.

GNSS loss of performance

In manned aircraft, the navigation capability is performed by a number of sensors. Depending on the selected sensors, the navigation function can be classified as *inertial navigation* or *radio navigation*. Inertial navigation relies on accelerometers and gyroscopes onboard the aircraft to calculate the position by *dead reckoning*. Radio navigation relies on a network of external transmitters and the corresponding receivers onboard the aircraft to derive

the aircraft position from the external radio frequency signal. The advantage of dead reckoning is that the navigation capability is independent of external equipment and signals; however, the navigation accuracy decreases with time since navigation errors are accumulated. In radio navigation, the accuracy basically depends on the relative position and/or distance between transmitter and receiver. In addition, external transmitters can be located on ground or in satellites. Ground-based transmitters, called Navigation Aids (NAVAIDs), are only available in certain coverage areas in the vicinity of the main routes and airdromes. By contrast, Global Navigation Satellite Systems (GNSSs) provide continuous worldwide navigation.

Due to the better accuracy of the GNSS with respect to any ground-based system [30], plus the fact that there exists a wide variety of compact, light-weight and inexpensive GNSS receivers in the market, the GNSS can be considered the main navigation sensor in unmanned aircraft. In short, GNSS navigation is based on a constellation of satellites whose position is known to high accuracy. Each satellite broadcast a timing signal and a data message with their orbital parameters. GNSS receivers use these signals to compute the *pseudo-range* (the time delay between signal transmission and reception) from each satellite in view. Using the pseudo-range of at least four satellites, it is possible to compute the aircraft position in three dimensions.

In aerospace applications, the Required Navigation Performance (RNP) of the GNSS system is prescribed in terms of accuracy, integrity, availability and continuity [77]. The accuracy of the GNSS solution depends on the precision of the pseudo-range measurements and on the relative position of the satellites being used to compute the solution (the so called geometry): satellites that are evenly spaced and around the user's horizon provide better geometries than if they were clustered together. Accuracy can be enhanced by integrating GNSS measurements and inertial data using Kalman filters [69]. *Differential techniques* are another approach to improve accuracy [30]. Integrity, availability and continuity requirements can be addressed using *augmentation techniques* [87]. These techniques allow to measure the Actual Navigation Performance (ANP), which is the maximum navigation error. The most common aircraft-based augmentation system is the Receiver Autonomous Integrity Monitoring (RAIM) system, which is able to detect and remove erroneous satellite signals when redundant data is available.

Even with augmentation systems, failures and errors can still degrade the system performance. The *GNSS loss of performance* occurs when the ANP is insufficient to perform the intended operation, probably because the RPA position is unknown, inaccurate, or lacks integrity. It is considered one of the

most critical contingencies because it has an impact on all of the remaining abnormal conditions considered in this work. Moreover, it compromises the effectiveness of some of the contingency management options, like *geofencing* or the emergency landing.

The GNSS loss of performance may be due to a number of factors occurring at all levels of the GNSS positioning process: from message generation, to broadcast and signal propagation, through to reception and processing at the receiver. As an example, consider the lack of satellite coverage or a poor satellite signal. Inaccuracy in the satellite position estimation, signal masking or multipath interference are other sources of error [26]. Even with optimal coverage and geometries, the GNSS receiver may also fail, either at a hardware or software level. For this reason, some of the most stringent navigation specifications require dual or triple redundant GNSS receivers [86]. In this case, we also assume that alternative means for navigation like dead-reckoning will be equipped as a backup.

Loss of control in-flight

Loss of control in-flight (hereinafter simply *loss of control*) refers to situations where either the remote pilot or the FMS are unable to control the aircraft, resulting in an unexpected deviation from the intended trajectory. Loss of control is the biggest single cause of fatal accidents [172]. This may well be because it is one of the most complex contingencies, involving numerous contributing factors that act individually or, more often, in combination. These factors include technical failures, ineffective aircraft control and adverse weather conditions.

Technical failures contributing to loss of control include structural damage of the aircraft, loss of function or malfunction of the actuators, and loss of power. Software errors related with the FMS are another important source of technical errors that will be further discussed in this work.

Ineffective aircraft control occurs when the aircraft enters a flight regime outside the flight envelope limits, close to the stall warning condition. This may occur due to a software error of the FMS too. For example, the guidance system provides the autopilot with wrong commands like targets that cannot be achieved. In some cases, the autopilot can prevent this situation by setting a flight envelope protection that puts some limits on attitude and speed targets. However, actuator failures or extreme environmental conditions can make these preventive actions ineffective. Human factors also play a key role in this case. For example, the remote pilot can perform a manual maneuver

that infringes the flight envelope limits. This may be due to lower situational awareness, because of poor piloting skills, or due to some problem derived from the remote pilot station. In this case, we presuppose that the HMI is adequate for the operation [138], that the remote crew is trained to avoid distraction during the flight, and that there exists some monitoring function that is able to provide a stall warning alert before the contingency occurs.

With respect to adverse weather conditions affecting the loss of control, it is possible to assume that the RPAS will be qualified to operate within some given environmental limits (e.g. visibility, turbulence, icing, etc.), and that the operation will not be conducted unless these conditions hold. If weather conditions change during the flight, then the remote pilot will have the means to identify the situation (e.g. using sensors or through a report of the corresponding ATC/UTM service) so that she or he can take an appropriate action.

Loss of separation and Near Mid-Air Collision

The *loss of separation* contingency and its degeneration into the *Near Mid-Air Collision (NMAC)* emergency are two events related with the encounter with a conflicting traffic in a given airspace volume. According to ICAO, the conflict management approach in UAS should parallel the layered approach for manned aircraft [82]. This approach relies on three layers, named *strategic conflict management*, *separation provision* and *collision avoidance* [88], which are already identified as threat barriers in SORA, see Table 3.2. The three layers are schematized in Fig. 3.7, and are discussed next.

The *strategic conflict management* is the first layer of conflict management. It involves strategic actions prior to the operation, including airspace organization and management; demand and capacity balancing; and traffic synchronization [88]. By strategic conflict management, all aircraft in the airspace adhere to rules and procedures of the airspace. For example, as it is well known, the airspace is organized in different airspace volumes, each of which is assigned to one of the seven *airspace classes* in ICAO Annex 11 [78]: classes A–E are referred to as *controlled airspace*, while F and G are *uncontrolled airspace*. In order to gain authorization to operate in these volumes, a series of requirements related with the onboard equipment and the training of the personnel need to be addressed. For example, transponder mode S is required to operate in controlled airspace, see **RM8** in Table 3.4. Requirements **RM3** to **RM7** in this table also apply in this case.

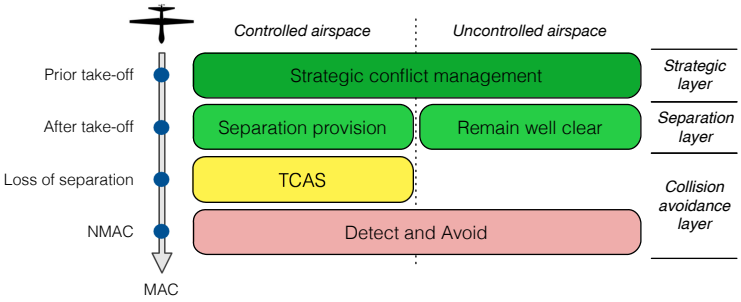


Figure 3.7: Conflict management layers in UAS.

One of the most representative and effective strategic mechanisms used to lower the risk of UAS operations is the use of *operational restrictions* of time or space, and its publication in the corresponding Notice To Airmen (NOTAM). By this means, it is possible to reduce the aircraft density in a given airspace volume, to ultimately segregate manned and unmanned aircraft. Moreover, compliance with these operational restrictions can be enforced by using geo-awareness systems, as it will be further discussed below.

When strategic measures are not effective at reducing the risk to an acceptable level, or if full RPAS integration is desired, then tactical mitigation layers are required. The first tactical layer is *separation provision*, see Fig. 3.7. It refers to the tactical process of keeping aircraft away from hazards (in this case, other traffics, but in general terrain and obstacles too) by at least the *separation minima* [88]. The separation minima is the minimum distance designed to ensure that the intended safety level for a proposed airspace meets the required standard [90]. Depending on the airspace class, separation provision relies on ATC (controlled airspace) or the pilot-in-command (uncontrolled airspace). In controlled airspace, the separation minima is prescribed based on the provisions of ICAO Doc. 4444 [83]. In uncontrolled airspace, separation minima is a subjective term because it is basically defined by visual lookout. For this reason, separation provision in uncontrolled airspace is simply referred to as *remain well clear*.

The *loss of separation* contingency occurs when the distance between two aircraft in flight violates the separation minima, or if they fail at remaining well clear. This may occur due to a human error, if the corresponding ATC unit does not detect the potential conflict, or if she or he provide an ineffective clearance. The pilot-in-command can also fail at performing the ATC clearance, or at remaining well clear. Other sources of error include technical issues

related with the automatic control function, or degradation of the GNSS signal resulting in an increased navigation error.

Once the *loss of separation* contingency has occurred, the *collision avoidance* layer comes into play, see Fig. 3.7. Collision avoidance is the conflict management layer directly related with contingency management. It relies on airborne sensors for detecting and tracking threats. Once a threat is detected, a quick maneuver deviating the aircraft from the current flight path is to be performed to provide separation with the collision threat.

The limiting factor related with this layer is that airborne sensors are imperfect and noisy, resulting in uncertainty in the current positions and velocities of the aircraft involved. In addition, variability in pilot behavior and aircraft dynamics make it difficult to predict where the intruder aircraft will be in the future. For this reason, false alerts and skipping true alerts are an important issue in collision avoidance systems that requires an important tradeoff between safety and performance.

Depending on the detection mechanism, a threat can be classified as *collaborative* or *non-collaborative*. Collaborative threats transmit information about its position using a Secondary Surveillance Radar (SSR) transponder. The advantage of such threats is that the effectiveness of the detection mechanism is not dependent on meteorological conditions nor the type and size of the aircraft encountered. However, collaborative threats imply that the intruder aircraft is equipped with a transponder, and it is not possible to assume that all aircraft in a given airspace volume will be carrying one.

Non-collaborative threats do not broadcast any direct information about its position because they do not carry a transponder, or because it is switched off. Alternative means for detecting non-collaborative threats basically rely on optical techniques like video, LIDAR or thermal imaging. The advantage of these techniques is that they are not dependent on the equipment of the encountered threat. By contrast, they are almost ineffective in instrument meteorological conditions. Moreover, the detection mechanism must be robust enough to detect aircraft of different sizes and shapes, and which fly at very different speeds and conditions (e.g. from general aviation to balloons, gliders, etc), what is a delicate issue. In general, small vehicles like small RPAs will be very difficult to detect, even if equipped with strobe lights.

Another key difference between cooperative and non-cooperative sensors is that, most of the times, non-cooperative sensors operate over a much shorter range than cooperative sensors (usually in direct line of sight) [106]. For this

reason, we consider that collision avoidance can be performed at two complementary levels, each one at a different time horizon. The first level is a collaborative level which mainly relies on the TCAS equipment, while the second level is non-collaborative and basically depends on the DAA capability. TCAS is a robust technology that is equipped in almost all commercial airliners. It is considered the main collision avoidance mechanism when operating in controlled airspace, where all aircraft are required to be equipped with a transponder. This way, after a loss of separation, the TCAS provides a resolution advisory suggesting the pilot a vertical maneuver (climb/descend) and a vertical speed that is expected to avoid the hazard. In most TCAS implementations, the resolution advisory is triggered minutes before the point of closest approach[132], and the actual avoidance maneuver is to be performed manually by the pilot-in-command.

If the TCAS-induced maneuver results ineffective, or if the transponder is unable to detect the threat, then the NMAC emergency will occur. To illustrate, SORA defines the NMAC condition as proximity of 500 ft horizontally and ±100 ft vertically [95]. Therefore, after the NMAC condition, the Mid-Air Collision (MAC) is imminent and only a last turn maneuver performed seconds before the point of closest approach can prevent the accident from occurring. In manned aircraft, this maneuver is to be performed manually by the pilot based on the See and Avoid (SAA) paradigm. In unmanned aircraft, due to the fact that the remote pilot as a reduced situational awareness, the SAA principle is often ineffective, specially when operating BVLOS. To overcome this, unmanned aircraft can be required to carry a DAA system, which goes beyond SAA as it implies that the detection function can be performed automatically onboard the UAS. This will be considered the main collision avoidance mechanism when operating in uncontrolled airspace.

The DAA system is one of the most demanding systems in terms of performance, though. According to SORA, UAS that aim to operate integrated alongside manned aircraft must be equipped with a DAA system compliant with the RTCA SC-228 or EUROCAE WG-105 MOPS standard [95]. Up to now, several industry and research approaches are trying to achieved the required system performance. Examples are MIDCAS, ACAS or FLARM. For example, ACAS-Xu, the ACAS version for UAS [111] use the SSR Mode S to detect cooperative traffics, but also have electro-optical sensors that can detect non-cooperative threats [174]. In addition, it includes provisions for carrying out the recommended resolution advisory automatically, what is required by ICAO in Doc. 10019 [82] (though the remote pilot still retains the ability to override the proposed action). However, it is generally understood that state-

of-the-art technology is still unable to reach the minimum performance under all operating conditions. For this reason, if a DAA system is required by regulation but the required performance is considered unfeasible, then airspace segregation will be necessary.

Mission boundary limits violation and Operational boundary limits violation

One of the most encouraging measures proposed by EASA for risk mitigation in RPAS is setting some boundary limits on the RPAS operation. The aim is to prevent the RPAS from accidentally going out of its operational area or flying over dangerous or prohibited areas. In this regard, we envision two slightly different problems, represented in Fig. 3.8:

- Setting boundaries or contention barriers to the area where the operation is to be technically contained and taking the proper measures to enforce these boundaries (Fig. 3.8 (a)). The RPAS shall not fly out of these zones.
- Setting boundaries or contention barriers to locations where flight may be restricted by regulation or raise safety concerns (Fig. 3.8 (b)). The RPAS shall not fly into these zones.

In both cases, the boundaries may include horizontal and also vertical limits. The SORA addresses the requirement of setting such boundaries to the operation [95]. In fact, it defines the *operational volume* as an airspace volume composed of the *mission volume* and the *contingency volume*, see Fig. 3.5. The mission volume encloses the intended flight path performed during the nominal condition. The contingency volume is a protection volume that accounts for the area required to perform an avoidance maneuver before violating the operational boundary. Therefore, the *mission boundary limits violation* is considered a contingency that can degenerate into the *operational boundary limits violation* emergency (the out of control condition identified in SORA).

The definition of the operational boundary limits differs depending on whether the airspace volume is controlled or uncontrolled. In the first case, the airspace volume will be structured as defined in the appropriate national Aeronautical Information Publications (AIPs) using airways, departure and arrival procedures, etc. When flying these standard procedures, the operational volume is defined as a containment area around the intended flight path. Historically, this containment area was defined based on the navigation accuracy of the ground-based NAVAID being used. For example, an instrument flight procedure designed for VOR navigation has a containment area that increases with the distance to the ground station [85].

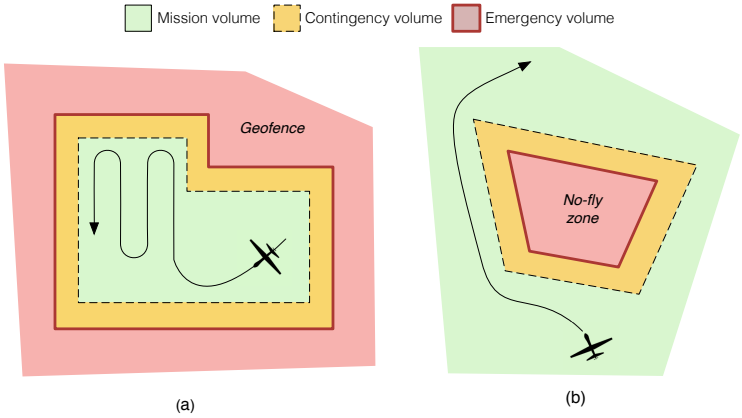


Figure 3.8: Geofencing and no-fly zones.

With the adoption of the GNSS navigation and the implementation of the ICAO Performance-based Navigation (PBN) concept, the definition of the containment area evolved. In PBN, there exist different navigation specifications which define the RNP with independence of the navigation sensors being used [86]. This way, each navigation specification specifies a containment area within which the aircraft shall remain 95% of the time. For example, a flight procedure defined using a RNP-1 navigation specification has a containment area of 1 NM on either side of the intended trajectory. It is assumed that RPAS that aim to operate in non-segregated airspace shall adhere to these rules.

When the airspace volume is uncontrolled, the aircraft trajectory is not subject to routes in the AIP. In this case, the operational boundaries are in general the limits of the uncontrolled volume. Specific limits for a given operation can also be set by segregating a defined piece of the airspace volume. As it was mentioned earlier in this section, this type of operational restriction is one of the key mechanisms for enforcing safe separation between manned and unmanned aircraft. ATC authorities should segregate these volumes and publish them in NOTAMs. RPAS are often required to monitor and enforce these boundaries using geo-awareness systems, what is called *geofencing*, see **RM5** in Table 3.4. Geofencing is specially useful when flying within the operations area, where the flight path cannot be always anticipated pre-flight. However, if the whole mission takes place in segregated airspace, the boundary limits may also include the path from the aerodrome to and from the operations area.

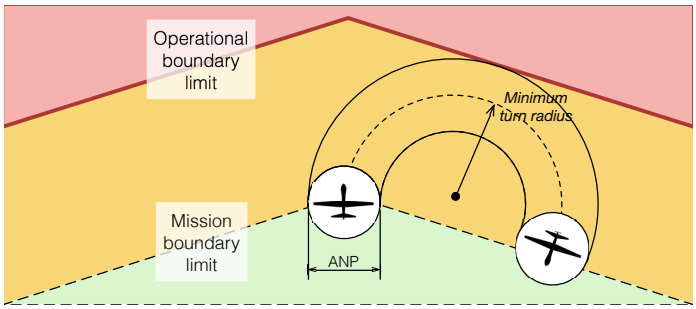


Figure 3.9: Mission boundary limits as a function of the aircraft performance and the navigation performance.

Once the operational boundary limits have been specified, the mission boundary limits are set based on the last chance to perform an avoidance maneuver before violating the operational boundary. This limit strongly depends on the aircraft performance and on the navigation performance, see Fig. 3.9 for instance.

Although in principle the use of geofencing and compliance with standard flight procedures should prevent the aircraft from infringing a restricted or prohibited area, human errors, technical errors or adverse operating conditions could deviate the aircraft from the intended trajectory. For this reason, the declaration of explicit *no-fly zones* is also suggested. There exist three contributing factors that can cause the *mission boundary limits violation* contingency and subsequent emergency: the Path Definition Error (PDE), the Navigation System Error (NSE) and the Flight Technical Error (FTE) [86, 95], described next.

The PDE occurs when the defined path does not correspond to the desired path, or when the defined path is inconsistent with the airspace structure. This is generally due to human errors. For example, the defined path may differ from the intended path if the navigation database in not up to date, or if the pilot makes a mistake when selecting a waypoint in the Flight Management Computer (FMC). The defined path may also be inconsistent with the airspace structure if an altitude target specified in the flight plan exceeds the vertical limits of the operational volume, for instance.

The NSE is the difference between the aircraft's estimated position and actual position, so it depends on the navigation accuracy of the sensors being used. In RPAS operations, where the GNSS is assumed to be the main navigation

sensor, the NSE is mainly affected by the loss of performance of the GNSS signal, a contingency that has already been discussed in this work.

Finally, the FTE refers to the ability to follow the defined path in a manual or automatic manner. Similar to the loss of control event, technical issues with the UAS, human errors or adverse operating conditions are threat categories with the potential of deviating the aircraft from the intended path. In order to limit the impact of this contributing factor, requirement **RM9** in Table 3.4 requires the aircraft position to be monitored at all times, so that the remote pilot can take corrective actions if required.

3.2.3 Identification of harms

This work assumes that the most severe consequences derived from a UAS operation out of control are the three harm categories identified in SORA, i.e. to cause fatal injuries to third parties on the ground; fatal injuries to third parties in the air; or damage to critical infrastructures [95]. For brevity, we will refer to these harm categories as the Entities of Value (EoV) at risk.

3.2.4 Determination of risk

In order to analyze how the proposed contingencies can develop into an emergency, and how the emergency can cause the above harms, a probabilistic risk model will be developed in this work. The proposed model, which will be presented in Chapter 6, is based on Eq. (3.1).

3.2.5 Determination of the acceptable level of risk

The determination of the acceptable risk level is a matter of the corresponding NAA, so it is considered out of the scope of this work. In this case, we will simply exploit the risk model in Chapter 6 to evaluate the risk reduction achieved when performing the proposed contingency management policy.

3.2.6 Identification of the means for risk mitigation

Once all the proposed contingencies and emergencies have been characterized, as well as the possible outcomes of these conditions, the last step is to identify the possible means for reducing the risk posed by the intended ConOps. As it was mentioned in the introductory part of this section, the response to a contingency must be defined from a predefined, limited list of options, called

Table 3.8: List of proposed contingency and emergency procedures.

Contingency procedures	Emergency procedures
Revert to manual control	Flight termination
Collision avoidance	
Loiter	
Climb to regain signal	
Land at designated landing site	

contingency procedures, that reconfigure the UAS mission at different levels; while the response to an emergency is normally *flight termination*. In general, the list of procedures is dependent on the contingency events under consideration: if different event are handled, then the list of procedures should be reevaluated, as should the decision logic. Next, we identify the contingency procedures that will be considered in this work as a response to the proposed contingencies. They are summarized in Table 3.8.

As a general rule, it is agreed that contingency procedures for UAS should mirror those of the manned aviation [25, 55, 82, 138]. The most common contingency procedure in manned aviation is the automatic autopilot disengagement to *revert to manual control*. By this means, the response to a contingency is delegated to the pilot-in-command. When it comes to UAS, reverting to manual control is a sensitive option because of the lower situational awareness of the remote pilot, as well as for the risk of experiencing a C2 link loss. For this reason, we consider that this option should be limited to contingencies whose resulting condition is difficult to handle in an automatic manner. For example, we believe that automatic handling of the loss of control would imply an automatic stall recovery function that might not be considered, in general, a safe procedure. The effectiveness of reverting to manual control after a loss of control depends on factors like the nature of the source of the contingency, the ability of the remote pilot when attempting to recover the nominal condition, or the aircraft height at the moment of experiencing the contingency.

Another common contingency procedure is the *collision avoidance*. In this case, we consider that this option will be necessary to respond to contingencies like the loss of separation and its degeneration into the NMAC condition, or to avoid a mission boundary threshold.

Other common contingency options are to start a *loiter* waiting for a given clearance, and *land at a designated landing site*. In particular, the ICAO

guidelines for contingency management in RPAS identifies two possible landing alternatives: land at the *nearest* appropriate designated landing site, or a direct return to the departure aerodrome [82]. In this work, we will develop a contingency management scheme that allows to select any of these alternatives in a flexible manner.

The last contingency procedure that will be considered in this work is another option identified in ICAO Doc. 10019: *climbing to altitude to attempt to regain the C2 link* [82], which is also identified in [64]. In this case, we will generalize and rename this option as *climb to altitude to regain the signal*, since we consider that it might be effective at regaining not only the C2 link signal, but also the GNSS signal after the GNSS signal loss of performance.

In summary, we assume that the different contingency procedures gathered in Table 3.8 will be effective at mitigating the effect of the proposed contingencies; and that these procedures are representative of the possible options available for UAS of any size and complexity.

3.3 Automated Contingency Management

Up to now, this work has identified some of the most representative contingencies for RPAS, as well as the possible responses to these contingencies. The next relevant design issue is how to carry out the mapping between contingency events and contingency procedures. At first sight, the answer to this question may seem relatively straightforward. For example, after the loss of separation, it seems obvious that collision avoidance is the right option. Far be it from this case. In general, the selection of one given contingency option depends on multiple factors, like the system state, the aircraft position or the airspace class where the RPAS is flying at the moment of experiencing the contingency. For example, collision avoidance could not be considered a safe procedure if the loss of separation occurs when the GNSS signal has also failed. In the same way, multiple contingency procedures could be considered effective for mitigating the effect of a given contingency at some given point of the mission, and the decision has relevant implications on the safety process. In fact, the work in [124] affirms that many UAS accidents are directly imputable to pilot errors when trying to manage an unexpected contingency.

It is also important to consider the most specific contingency in RPAS, which is the C2 link loss. Proper handling of this event is strictly required by ICAO for operating in non-segregated airspace [82]. Moreover, assuming that any combination of contingencies can happen in conjunction with C2 link loss,

some sort of Automated Contingency Management (ACM) ability is necessarily required to handle the situation when the remote pilot is not in the control loop. ACM functions imply that the detection of contingencies, the decision-making process for selecting the response to a contingency, as well as the execution of the selected response is performed by the on-board system in an automatic or semiautomatic way. Next chapter develops the software architecture of an on-board Mission Management System (MMS) with provisions for ACM.

Chapter 4

On-board Mission Management System software architecture

4.1 Introduction

In a conventional airliner, the Flight Management System (FMS) is the on-board system responsible for the automatic flight guidance and control of the aircraft. It performs the guidance actions based on the directives of a flight plan. From a high level point of view, this system can be represented as the two-layer scheme represented in Fig. 4.1, described next:

- The FMC is the top component of the architecture. It is responsible for the Strategic Operation, which means that it can handle the entire route described by the flight plan. It usually includes the Lateral Navigation (LNAV) mode and the Vertical Navigation (VNAV) mode.
- The Flight Director (FD) is the bottom component of the architecture. It is responsible for the Tactical Operation, which means that it controls the current maneuver by programming the autopilot and auto-throttle modes for the speed, course, and altitude control. It can be commanded by the upper layer or directly by the pilot-in-command through the Mode Control Panel (MCP) when the FMC is not engaged.

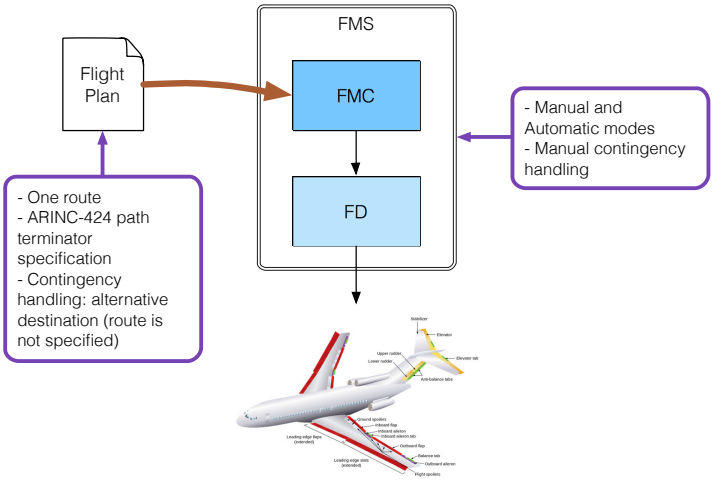


Figure 4.1: Flight Management System for conventional aviation.

Table 4.1: Operational modes in a conventional FMS.

Operational mode	Responsible agent	
	Routing decisions	Guidance actions
Manual	Pilot-in-command	Pilot-in-command
Automatic	Pilot-in-command	Automatic system

In this scheme, the interaction between the pilot-in-command and the automatic system can be characterized as a function of two factors: *a)* who decides the route to be followed and makes strategic decisions (i.e. in the long term) so that the flight is conducted safely and efficiently, and *b)* who is in charge of the guidance actions at the tactical level (i.e. in the short term), like flight level changes, turns, or speed selections. Based on these factors, a conventional FMS is considered to provide two *operational modes*, outlined in Table 4.1 and described next:

- The *Manual mode* is the lowest level of automation, corresponding to a Sheridan Level 1 in Table 2.2. In this mode, all decisions (in the short term and in the long term) are taken by the pilot-in-command. In other words, the FMC is disengaged and the pilot exercises direct control over the aircraft by either using the yoke or by setting the proper control targets in the FD.

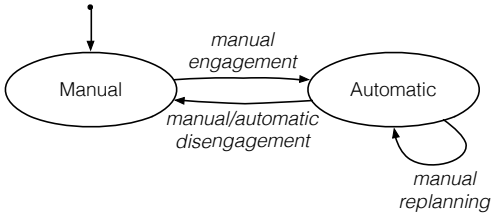


Figure 4.2: Operational modes and transitions between modes in a conventional FMS.

- The *Automatic mode* represents the classical operation using the FMC. In this mode, the pilot-in-command programs a route in the FMC, and then the FMC decides about the commands that will be delivered to the FD to follow the desired route. This corresponds to Sheridan Levels 2 to 6. In other words, the route selection is made by the pilot-in-command and guidance actions are performed by the automatic system. In any case, this mode must also include some capability to override the actions of the automatic system using manual control or to abort some decision of the automatic system and reverting to manual control if necessary. For example, if the pilot decides an automatic landing, the pilot is allowed to abort the maneuver before a deadline if something goes wrong.

The previous operational modes and the transitions between these modes can be represented as a state automaton like in Fig. 4.2. One important aspect to this machine is how transitions occur. This is discussed next:

- Transitions *from manual to automatic* mode shall only occur upon the pilot's decision to engage the FMC. This is a safety mechanism to prevent inadvertent behavior of the automatic system. Another important remark is that this transition requires the pilot act by deciding on the route to be followed once the automatic mode is engaged, and how the transition to this route is to be performed. If no suitable route exists, then a stabilizer mode that maintains course, altitude and speed is often engaged.

- Transitions *from automatic to manual* mode can occur for three reasons: programmed disengagements, pilot decisions, or as a contingency handling strategy. Programmed disengagements occur when a transition to manual mode is pre-programmed in the middle of a sequence of automatic maneuvers, or at the end of this sequence. A disengagement can also be forced by pilot decision at any time; this can be done by simply acting on any flight control. Finally, the automatic (unexpected) autopilot disengagement can also occur before the autopilot control becomes unstable,

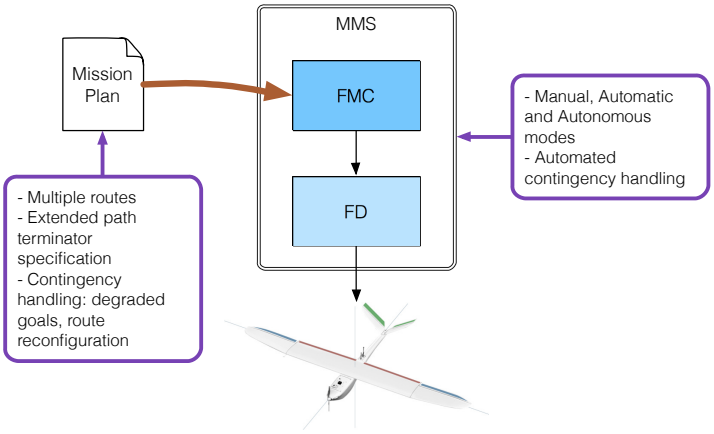


Figure 4.3: Mission Management System for unmanned aircraft.

preventing *loss of control* conditions, or as a response to abnormal conditions that cannot be safely automatized. This is the nominal contingency handling strategy in manned aviation.

When the aircraft is a UAS, and specifically a RPAS, the pilot-in-command is a remote pilot, and the on-board FMS is renamed as Mission Management System (MMS), what emphasizes the difference between flight plan and mission plan (e.g. that the RPAS is intended to perform some payload related-task, that there exist multiple possible routes, etc., see Chapter 5). As a starting point, it can be assumed that the MMS follows the same scheme as the FMS in Fig. 4.1, and that it provides the same operational modes described above. Following this premise, the research group of the authors developed a MMS software architecture that allows to fly elementary, static Mission Plans in a manual or automatic manner [68, 167]. This initial architecture will be briefly introduced in Sec. 4.2.

The previous system is able to perform the intended mission as long as the RPAS is flying in a nominal condition. However, it is generally understood that *the MMS of a RPAS operating in the specific or certified category should provide an increased level of automation with respect to a FMS for manned aviation* to counteract the lower *situational awareness* of the remote pilot, as well as the likelihood of experiencing a C2 link loss, to ultimately reach the TLS required by regulation. As a response, this work proposes to increase the RPAS automation level by providing the initial MMS in Sec. 4.2 with the ACM functions described in Chapter 3, see Fig. 4.3.

Table 4.2: Operational modes in the proposed MMS.

Operational mode	Responsible agent	
	Routing decisions	Guidance actions
Manual	Remote pilot	Remote pilot
Automatic	Remote pilot	Automatic system
Autonomous	Automatic system	Automatic system

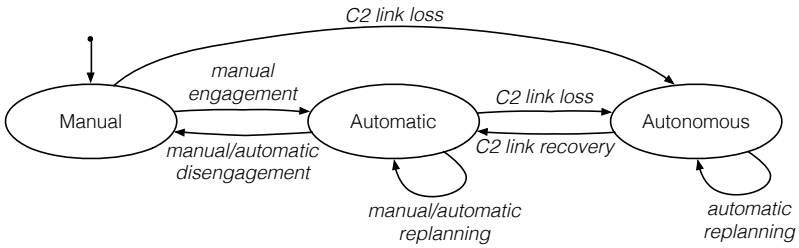


Figure 4.4: Operational modes and transitions between modes in the proposed MMS.

ACM functions imply that the state machine in Fig. 4.2 must be extended to provide the MMS with an additional operational mode, the *Autonomous mode*, which represents the highest level of automation of a UAS (corresponding to a Sheridan Level 10). This mode, which does not exist in conventional FMSs, is intended for completely automating the aircraft guidance when all communication links are down or some other serious contingency prevents the pilot to take control of the aircraft. In other words, the on-board system assumes decisions in the short term and in the long term because all further contingencies must also be handled in this mode. Note that, although the autonomous operation is not a desirable flight condition, it can solve an out-of-control situation perhaps in a better or more efficient way than flight termination.

The resulting operational modes for the proposed MMS are gathered in Table 4.2. The resulting state automaton is also represented in Fig. 4.4. Note that, in this automaton, transitions *from manual or automatic to autonomous* mode are always triggered by the loss of the C2 link; and that the C2 link recovery triggers the *autonomous to automatic* transition.

In this chapter, we will describe the transformation of the initial MMS in Sec. 4.2 into a Safe Mission Management System (SMMS) that handles contingencies. In particular, Sec. 4.3 presents the software architecture of the

proposed SMMS; and Sec. 4.4 discusses safety aspects relating to the software development of this system.

4.2 Initial Mission Manager architecture design

As it was introduced before, the Mission Management System (MMS), or simply *Mission Manager*, is the core system for performing the automatic guidance and control of the RPAS. Its functionality is based on the definition of a *Mission Plan* that basically specifies the RPAS route and payload actions. The initial architecture for this system is based on the ideas of the Three-tier (3T) architecture [20]. In short, a 3T architecture separates the intelligent control problem into three interacting layers named *Deliberative layer*, *Sequencing layer*, and *Reactive layer*. In this case, the 3T concept has been applied from a flight guidance and control perspective, so the three layers have been renamed as *Path Planner*, *Guidance System*, and *Flight Director*, respectively, for clarity. They are schematized in Fig. 4.5. Note that, in this scheme, the *Path Planner* in combination with the *Guidance System* provides a functionality that is equivalent to the one of the FMC of a conventional FMS.

4.2.1 Path Planner

The *Path Planner* is the highest level component of the architecture. It can be considered as an abstract object with the ability to provide a reference trajectory for the Guidance System. As it is shown in Fig. 4.5, there exist multiple instances of Path Planners in the Mission Manager, but there is only one active instance at a time. A particularly important instance of the Path Planner is the "Mission Planner" which determines the reference trajectory based on the directives of a Mission Plan. In the initial MMS, the Mission Plan is specified as a sequence of *flight legs* that implement the ARINC 424 *path terminators* [5]. Thus, the role of the Mission Planner is to provide each leg to the Guidance System in a sequential manner. In parallel to the Mission Planner, there exist some other *Task Specific Planners* for special tasks, such as the exploration of unknown terrain. These planners usually provide deviations on the nominal path planning policy for a short period of time. In the initial version, the remote pilot can select the required Task Specific Planner at will.

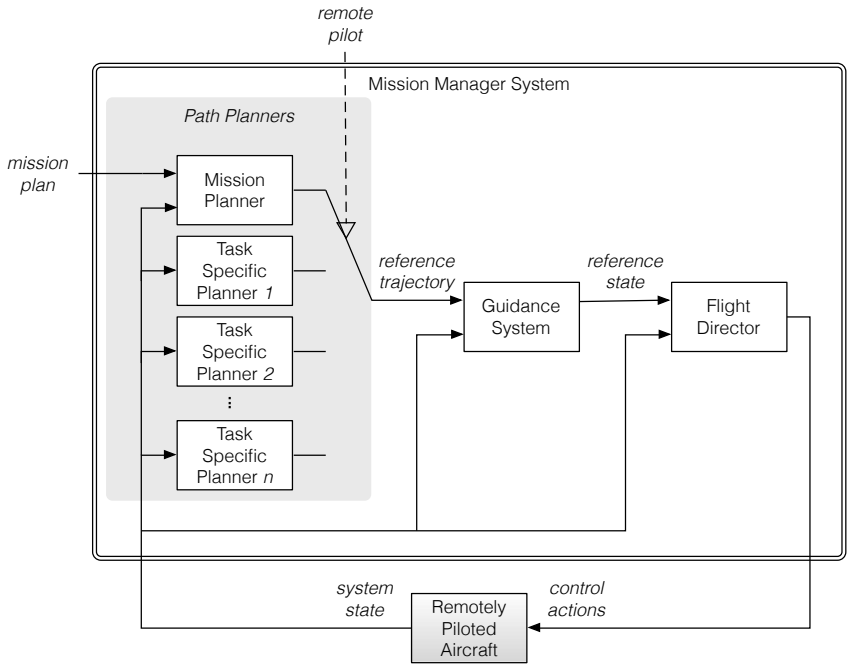


Figure 4.5: The initial Mission Manager architecture is structured into three layers: the Path Planners, the Guidance System, and the Flight Director.

4.2.2 Guidance System

The *Guidance System* determines how to fly the reference trajectory provided by the active Path Planner and then activates the appropriate control modes of the Flight Director. To do so, the Guidance System uses a library of elemental maneuvers in the lateral plane (LNAV) and in the vertical plane (VNAV). LNAV maneuvers include straight maneuvers (with constant heading, with constant course, etc.) and turn maneuvers (with constant radius, with constant turn rate, etc). VNAV maneuvers include flight level maneuvers and climb/descent maneuvers (at constant speed, at constant vertical speed, etc). Thus, each time a new reference trajectory is received, the Guidance System plans a suitable sequence of maneuvers. The sequence of maneuvers in the LNAV plane is independent of the sequence of maneuvers in the VNAV plane.

Once the list of elemental maneuvers has been planned, the Guidance System activates the LNAV maneuvers and the VNAV maneuvers for carrying out the plan in a sequential manner. Only one LNAV maneuver and one VNAV

maneuver can be active at a time. According to these active maneuvers, an interpreter activates the appropriate control modes of the Flight Director. The interpreter also computes the target values (the reference state) for the selected modes using different guidance algorithms. For example, the control mode for flying a turn with a constant radius is the heading control mode; and the algorithm that computes the target heading for this mode is based on the "carrot-chasing" algorithm in [40].

Each control mode is flown until some target event occurs. For example, the turn with a constant radius can be flown until the RPAS reaches a given waypoint, or until a given time-out occurs, for instance. When this target event is triggered, the Guidance System selects the next maneuver in the sequence. When the sequence of maneuvers is completed, the Guidance System notifies the Path Planner so that the high-level component will provide a new reference trajectory.

4.2.3 Flight Director

Finally, the *Flight Director* implements the control loops of the autopilot control modes. For example, the autopilot has the "heading control" mode, the "altitude control" mode, the "vertical speed control" mode, etc. Recall that the Flight Director can be commanded not only by the upper layers of the architecture, but also by the remote pilot directly, depending on the operational mode engaged.

4.3 Safe Mission Manager architecture design

In order to transform the previous MMS into a Safe Mission Management System (SMMS), the initial architecture in Fig. 4.5 will be extended with new software modules to perform ACM functions. Moreover, some of the components in Fig. 4.5 will be also internally redesigned to enable them to execute *contingency handlers*. Finally, in order to support the execution of these handlers, a novel Mission Plan specification that deals with contingency handling will also be designed. Next, we introduce the architectural components of the SMMS, while the Mission Plan specification is to be discussed in Chapter 5.

To start with, the proposed SMMS will be designed under the hypothesis that the RPAS is equipped with a Flight Termination System (FTS) that is able to terminate the flight expeditiously, and that this measure is an effective mechanism for enforcing safety. The need for this mechanism is supported by

a number of aviation stakeholders [50, 55, 82, 118, 173]. They also indicate that this action is to be triggered manually by the remote pilot as well as autonomously by the on-board system. This latter aspect has an important implication on the system design:

Corollary 1 *A list of predefined conditions for the automatic activation of the FTS shall be specified in the embedded software.*

Based on the hypothesis that the flight termination action is an effective mechanism for enforcing safety, the ability to engage the FTS becomes a safety-critical function. Critical software in aerospace is subject to strict Validation and Verification (V&V) processes defined by the DO-178 standard [131]. According to this document, a software component that cannot be completely verified at the design phase should not adversely affect safety. This poses an additional requirement on the design of the SMMS:

Corollary 2 *In order to enable an extensive testing strategy, the list of predefined conditions for the automatic activation of the FTS shall be hardcoded in the embedded software.*

Commanding the flight termination action is a drastic decision, though. According to the safety framework in Fig. 3.3, less extreme contingency management options could also be attempted for mitigating the risk inherent to contingencies in some circumstances. In these cases, we consider that the hardcoding of policies is not strictly necessary because there is still a flight termination mechanism that can be thoroughly tested. In addition, we consider that having flexibility to specify less critical aspects of the contingency management policy would be beneficial to the final user.

Accordingly, we advocate separating ACM functions into two separate software components, named the *Safety Monitor* and *Contingency Manager*. They are responsible for the strategic decision-making process, but each one has a different impact on safety, as discussed next:

- The role of the *Safety Monitor* is to check system behavior for unsafe states; and when an unsafe state is detected, to take the critical decision of whether a contingency management option is feasible, or whether the flight termination action is required instead. As it was represented in Fig. 3.3, this decision depends on the criticality of the resulting state. This way, the Safety Monitor manages the two *safety thresholds* in Fig. 3.2:

When the contingency threshold is exceeded, the criticality is such that there is still a safety margin for attempting a contingency management option. The resolution of this state will be delegated to the Contingency Manager. But if the contingency option fails and the emergency threshold is surpassed, or if there is a low probability of successfully handling the contingency, the Safety Monitor will command the FTS to ensure that safety is not further compromised.

- When contingency management is a plausible option, the *Contingency Manager* should react and *plan* the appropriate response for reducing the probability of infringing the emergency threshold (and thus the probability of flight termination), and ultimately attempting to recover the nominal condition of the RPAS. As it was mentioned in Sec. 3.2, contingency management works at a mission level. For this reason, we describe the role of the Contingency Manager as to settle the most convenient *mission goal* to be achieved at a given flight condition. When no contingency has been declared, the mission goal is simply the *nominal goal*. This goal can be achieved by performing the intended mission. After a contingency occurs, the Contingency Manager can interrupt the current mission execution to set a different, degraded goal. A degraded goal can be achieved by performing a *contingency procedure* that replaces the current reference trajectory of the RPAS with a new one that is more suitable for the abnormal state being faced. Once a selection has been made, the Contingency Manager will notify the Mission Manager to execute the selected option.

Defining a contingency management policy for dealing with contingencies is a complex matter, though. In some cases, there could exist multiple contingency procedures that could be effective for mitigating the effect of a given contingency, and the optimal solution might depend on multiple factors. The extent to which this policy can be customized by the final user is a design issue that will be discussed later in this section.

In summary, we believe that the proposed differentiation between Safety Monitor and Contingency Manager provides an interesting tradeoff between safety and robustness: the Safety Monitor can enforce safety at any time during the mission, even when everything else fails; and the Contingency Manager enhances robustness of the system by providing solutions to abnormal states. The implementation of the Safety Monitor must be hardcoded at the design phase so that extensive testing can be performed; in contrast, the verification of the Contingency Manager is subject to user modification. Safety aspects re-

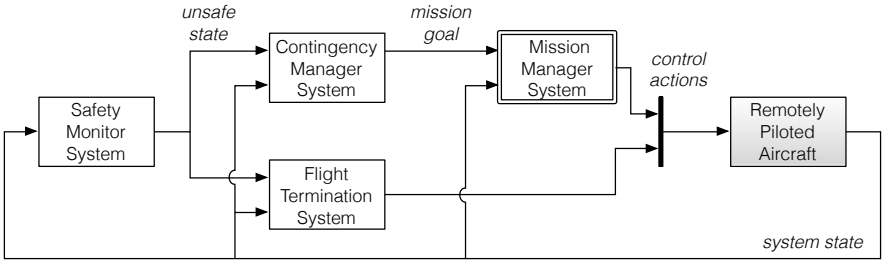


Figure 4.6: Safe Mission Manager architecture with four major components: Safety Monitor, Contingency Manager, Mission Manager and Flight Termination Systems.

lating to the software development of this architecture will be further discussed in Sec. 4.4.

High-level architectural view

The resulting software architecture of the proposed SMMS is schematized in Fig. 4.6. It shows four major software components: the Safety Monitor, the Contingency Manager, the Mission Manager and the Flight Termination Systems. In this scheme, the Safety Monitor, the Contingency Manager and the Flight Termination Systems extend the initial architecture in Sec. 4.2; while the Mission Manager must also be internally redesigned to enable it to execute the contingency procedures by introducing *contingency handlers*. Note that the "Mission Manager System" box in Fig. 4.6 gathers all the sub-components that were depicted in Fig. 4.5.

Based on Fig. 4.6, the execution of the proposed ACM scheme will be composed of the following steps:

1. *Monitoring the system behavior to detect and diagnose contingencies*: this step will be performed by the Safety Monitor;
2. *Deciding on a policy for dealing with contingencies*: step will be performed at two levels by the Safety Monitor and by the Contingency Manager; and
3. *Executing the corresponding policy*: this step will be performed by the Mission Manager or the Flight Termination System, depending on the selected policy.

The sub-sections below discuss the different components in more detail; while a prototype implementation of this architecture is presented in Appendix B.

4.3.1 Safety Monitor

Detecting unsafe states and triggering alarms is considered a safety critical functionality. An interesting approach for this task is the use of formal specifications to derive *safety monitors*. The advantage of this methodology is that it enables very efficient monitors that can be verified because they are automatically derived from a formal specification. This technique has been an important research topic for checking software and hardware behavior in embedded systems [107]. In this sense, the DLR proposed using similar techniques for system health management and the detection of unsafe conditions on UAS [4]. As a result of a cooperation between DLR and the author of this thesis, here we adopt the concept of safety monitoring to monitor the system behavior and diagnose contingencies. Next, we discuss how to integrate monitoring into the system architecture.

An aircraft is a distributed system consisting of a large number of independent subsystems. Critical system components are usually required to self-monitor, to perform fault detection and to report their faults. This is the case for the GNSS subsystem, where RAIM systems are prescribed. Another example is the ACAS, which is able to detect collision threats autonomously.

However, safety monitoring should be performed not only inside each subsystem but at the system level as well. This requires having access to the global system state because an unsafe situation can be formally specified as a predicate on the system state. We understand as a "global" state the aggregation of the states of a set of subcomponents of a distributed system [114]. As an example, consider the following unsafe condition: "the distance to the airport is greater than the mileage allowed by the reserve fuel remaining". Checking this predicate involves knowledge as to the airport location, the aircraft position and the remaining fuel. Even if the subsystems that estimate these state variables are failure free and are not reporting any alarm, if these data are processed at a system level, then an unsafe situation can hold. In general, all contingencies derived from inappropriate aircraft operation involve the state of several system components.

Thus, online safety monitoring requires some centralized, high level component that coordinates all the distributed monitors and performs diagnoses at the system level. Performing this task implies that some knowledge about the

normal behavior of the system is presented to the real-time reasoning. This knowledge can be typically developed using model-based or data-based techniques [37]: in model-based techniques, it is derived from theoretical models, while in data-based techniques, it is inferred from empirical experiments of fault-free operation.

One of the main problems is the reliability of the detection mechanisms. This refers to the probability of reporting false alarms or skipping true alarms. The main problem in creating a robust system is that sensors are imperfect and noisy. This results in uncertainty in the state determination. For this reason, the use of deterministic logic to define alarm conditions on the state variables does not guarantee reliable detection. Some proposed techniques for dealing with uncertainty are fuzzy logic [100], stochastic alarm detection techniques [18] and Markov decision processes [103]. Consequently, the Safety Monitor should also manage alarm thresholds.

In this work, we propose to model the centralized Safety Monitor as a Finite-State Machine (FSM) because this provides a number of advantages: on one hand, and as will be introduced later, the use of FSMs will ease the application of formal methods for the verification of this model; on the other hand, FSMs can also be used to represent probabilistic state automatons to account for the reliability of the detection mechanisms [130]. In this case, the proposed automaton model representing the Safety Monitor component is shown in Fig. 4.7. It starts at a nominal state where there is no evidence that a contingency has occurred. Therefore, within this state, the RPAS is flying the nominal mission in a manual or automatic manner. As represented in Fig. 3.2, the Safety Monitor manages two safety thresholds. When there is evidence that the contingency threshold has been exceeded, the Safety Monitor triggers a *contingency event*. Such events make the system shift into an abnormal state where a given contingency option can be planned. In this state, the Contingency Manager will be enabled. When there is evidence that the emergency threshold has been exceeded, the Safety Monitor raises an *emergency event*. Such events make the system evolve to a state where the nominal condition is considered to be unrecoverable, so the only feasible action is flight termination. Once in an emergency state, the FTS will be enabled.

When an abnormal is entered, a *recovery event* can bring the system back to the nominal state if the contingency procedure turns out to be effective; otherwise, the system will remain in the same state and further actions can be planned. In addition, subsequent contingencies may also occur. However, we believe effective handling of nested alerts is highly unfeasible in most cases. For this reason, the approach for dealing with these conditions will be *prevention*:

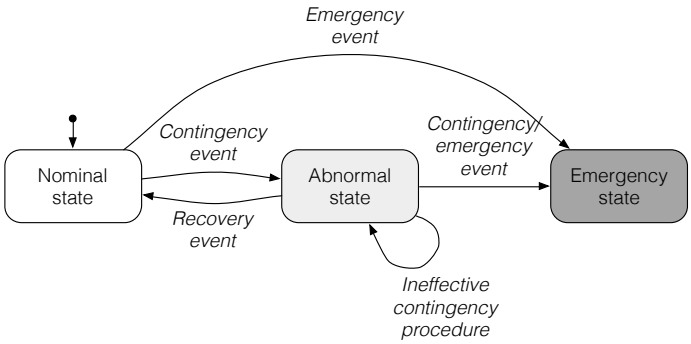


Figure 4.7: Generic model of a centralized Safety Monitor for RPAS.

avoiding, whenever possible, the occurrence of new contingency events by using a safe and conservative design of the contingency procedures; otherwise, nested contingency events will lead to the emergency state. Note that, according to Corollary 2, the FSM in Fig. 4.7 must be hardcoded in the embedded software.

4.3.2 Contingency Manager

Once an abnormal state is entered, the next step is to decide on a policy for dealing with the resulting flight condition. The role of the *Contingency Manager* is to select a mission goal that tries to partially complete the mission, probably in some degraded form, while maintaining safety. Once a selection has been made, if the RPAS is operated in manual or automatic mode, the Contingency Manager informs the remote pilot about this goal. The pilot can decide between accepting the proposed contingency handling or changing to manual mode. If the RPAS is operated in autonomous mode, the Contingency Manager instructs the Mission Manager directly to execute the selected goal.

In order to select the most convenient mission goal at a specific flight condition, the Contingency Manager must address the strategic decision-making problem of balancing the rewards with the risks associated with each possible option to maximize the probability of success, see Fig. 4.8. In this case, we consider that the main reward for selecting a mission goal associated with a contingency procedure is to try to avoid the negative outcomes of performing the flight termination action. Among the negative outcomes of flight termination, we envision the economic impact of not performing the intended mission, plus the risk of damaging (or even losing) the vehicle and the payload it may be carrying as well. On the other side, we consider that the risk of not performing instant

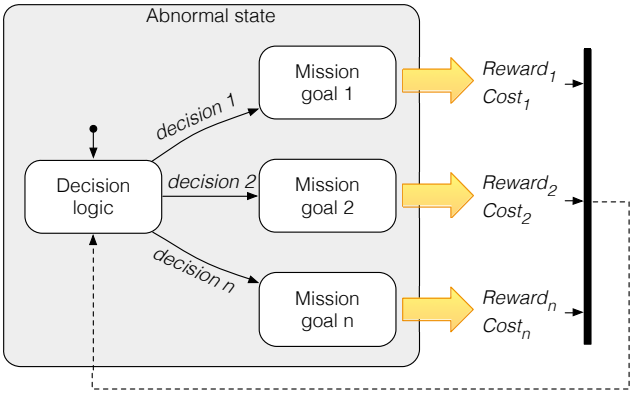


Figure 4.8: Decision logic for finding a feasible mission goal after a contingency occurs.

fight termination is to continue operating in some degraded mode, with reduced safety margins, what certainly increases the risk entailed for third parties.

In general, factors influencing the strategic decision-making process in Fig. 4.8 not only include the contingency event reported by the Safety Monitor, but also other state variables. For example, the Contingency Manager could make different decisions about how to handle the loss of separation condition depending on whether the RPAS is flying in controlled or uncontrolled airspace. The decision is even more complex when several concurrent events are reported by the Safety Monitor at the same time. For this reason, we believe that the Contingency Manager also needs to have access to the global system state, as it was represented in Fig 4.6. In this sense, one of the key state variables influencing the selection of a given contingency option is the aircraft position at the time the contingency occurs; and specially the relative position with respect to certain mission locations like landing sites, holding points, or flight termination areas. This will be demonstrated further on this work.

As a result, we propose to model the decision logic for selecting a suitable mission goal as a FSM too. We call the specification of this automaton the *Contingency Plan*. The design of this plan is a sensitive task since it determines the behavior of the RPAS after a contingency occurs. We consider that the design of this plan is subject to a series of constraints:

- It must be deterministic to ensure a predictable behavior of the RPAS, even without remote pilot intervention [35, 82].

- It must be consistent with current airspace regulation, like the rules of the air [79] or the ICAO guidelines for contingency management [80, 82].
- The aim of this plan should be to reduce the time of flight of the RPA experiencing the contingency [82].

An interesting design issue is who is the actor that should specify the Contingency Plan, and what is the level of customization allowed in the design of this plan. In other words, it is necessary to define to what extent should the Contingency Plan be hardcoded into the embedded software, and to what extent should it be open to user modification. Two opposing trends affect this issue. One one hand, having flexibility to specify this plan on a mission basis (thus avoiding the hardcoding of policies) can make the remote pilot handle a contingency scenario in a more responsive way. On the other hand, the specification of a state automaton with lots of states and transitions is a difficult and critical task, so it must be verified; in addition, the specification and verification of a safe state automation is an ability that probably exceeds the scope of the RPAS operator.

To overcome this, we envision the following tradeoff, schematized in Fig. 4.9: the specification of the Contingency Plan will be assigned to the system developer so that it is hardcoded into the embedded software; but if the contingency procedures have certain *configuration parameters* that allow to define *how* they must be executed, then these parameters will be specified by the remote pilot in the Mission Plan pre-flight.

As an example, assume that "land at a designated landing site" is one of the possible contingency options, and that the configuration parameters of this option include: *1)* the list of suitable landing sites, and *2)* the routes towards these sites. Then, in the proposed approach, the remote pilot will not be able to specify what state leads to the selection of this contingency option, but she or he will be able to specify the possible landing sites, as well as the possible routes for reaching them. A Contingency Plan example mapping the contingency events in Sec. 3.2.2 with the contingency procedures in Sec. 3.2.6 is developed in the Appendix A using formal methods; while the design of a Mission Plan specification with provisions for contingency handling will be discussed in Chapter 5.

Note that the proposed solution has an important implication on the system verification process: the verification of the Contingency Manager will be subject to the verification the Mission Plan at operation time. In other words,

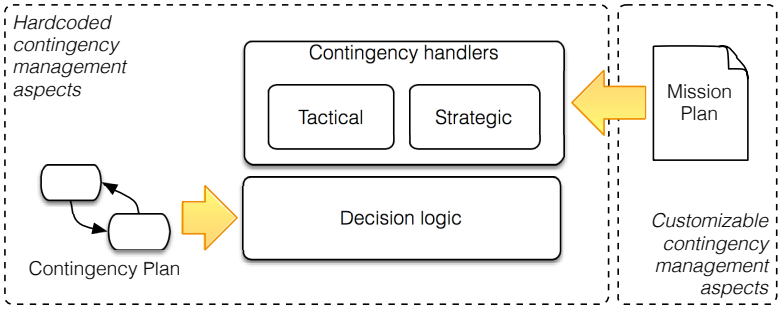


Figure 4.9: Contingency management specification scheme.

the decision made by the Contingency Manager will only be safe if the Mission Plan specification is correct.

4.3.3 Mission Manager

As was mentioned before, the Mission Manager is the software component with the ability to perform the automatic flight guidance and control of the aircraft. For this reason, once the Contingency Manager defines the most convenient mission goal, it is the Mission Manager who should manage the route that achieves this goal. In the initial architecture in Sec. 4.2, the Mission Manager was capable of processing and flying a single, conventional route. However, it did not include provisions for contingency handling, like the automatic replanning capability. In order to give the Mission Manager the ability to (re-)configure the route based on the mission goal settled by the Contingency Manager, it is necessary to internally redesign it by providing the corresponding *contingency handlers*.

Contingency handlers are a special case of the Path Planners in Fig. 4.5 that generate the contingency procedure's reference trajectory. Therefore, contingency handlers override the Path Planner guidance used during the nominal condition with some specific guidance based on safety concerns. The advantages of the 3T concept is that, although new Path Planning policies must be introduced into the model, the remaining layers of the Mission Manager (i.e. Guidance System and Flight Director, see Fig. 4.5) can basically remain unaltered, what reduces the development effort.

The number of contingency handlers to be embedded on the architecture depends on the required contingency procedures that have been identified in the

system risk assessment. In general, contingency procedures can be classified into three categories according to their impact on the planned route:

- *Strategic contingency procedures* are suitable when the initial route is no longer feasible and thus a new mission has to be planned. The new reference path is often constructed with a more conservative design, with limited turns, vertical speeds, etc. This mechanism protects against exceeding the flight envelope during an abnormal state (thus causing a nested alert), but also results in reduced aircraft performance due to the contingency being faced. One example of these procedures might be flying towards the alternative landing site.

- *Tactical contingency procedures* are flight procedures that deviate the aircraft from the intended route temporarily, though the original mission may be resumed afterwards once the effect of the contingency has been mitigated. In contrast to strategical contingency procedures, tactical ones often demand high flight performances, like in a traffic avoidance maneuver.

- *Operational mode commands* complement the strategic and tactical procedures since they do not affect the planned route in a direct manner, but the operational mode in which the route is to be performed. For example, revert to manual control does not necessarily imply that the initial route is no longer feasible, but that it cannot be flown using automatic control. In a similar way, after the C2 link loss, the operational mode automatically evolves to *Autonomous mode* (Fig. 4.4), meaning that the route cannot be flown using manual control.

We consider that some of the contingency procedures identified in Sec. 3.2.6 can be performed either at a strategic or at a tactical level, see Table 4.3. For example, the "loiter" procedure can be performed at a tactical level, meaning that the holding maneuver is started at the time the mission goal is selected by the Contingency Manager (irrespective of the current position of the RPAS). However, it can be performed at a strategic level too: to do so, it is necessary to define a loiter point, and the route towards this point. This way, the holding procedure will not be started until the loiter point is reached (when the strategic phase is over). This work is specially focused on strategic contingency procedures, since they are related with the Mission Plan specification in Chapter 5.

Based on the strategic/tactical differentiation, contingency handlers can also be classified into two classes according to their alternative guidance method:

Table 4.3: Proposed contingency procedures by category.

Procedure	Category
Revert to manual control	Operational mode command
Avoidance maneuver	Tactical
Loiter	Strategic & Tactical
Climb to regain signal	Strategic & Tactical
Land at designated landing site	Strategic
Flight termination	Strategic & Tactical

a) contingency handlers relying on the Mission Planner, and *b)* contingency handlers relying on a Task Specific Planner. All the contingency procedures that can be performed at a tactical level require to provide the Mission Manager with a dedicated Task Specific Planner. This is because the resulting reference trajectory strongly depends on the type of contingency being faced. For example, the reference trajectory of a traffic avoidance maneuver is generated by the "Collision Avoidance Planner". By contrast, contingency procedures performed at a strategic level do not require a specific Path Planner be introduced but rather an alternative route definition be provided to the Mission Planner. In this sense, the proposed Mission Plan specification in Chapter 5 will be capable of (re-)defining the route in accordance with the mission goal selected by the Contingency Manager. How to perform smooth transitions between the guiding actions of different Path Planners is an interesting issue that exceeds the scope of this discussion.

4.3.4 Flight Termination

As was motivated before, this work assumes that the RPAS will be equipped with a FTS that is capable of safely bringing the vehicle back to the ground in case of severe contingencies or emergencies. An interesting design issue is *how* to perform this action. In general, the most convenient flight termination mechanism depends on the RPA model and performance. For example, the work in [157] presents a survey of current and future technologies and procedures for performing aerodynamic and ballistic terminations. Alternatives are self-destruct systems that allow an in-flight destruction to be performed without the loss of human lives [41]. However, this latter option may not be allowed in RPAS operating in the certified category [25, 153]. In this work, we will simply assume that the selected FTS is effective at performing the flight

termination action, and that the completion of this action does not entail a risk for third parties.

This latter assumption may be controversial, though. Even if the FTS is effective at reducing the impact energy, a mid-size RPA (or the resulting debris) falling to the ground clearly poses a risk to third parties in the vicinity of the impact area. In order to mitigate the residual risk, this work puts the focus not on *how* to perform the flight termination, but on *where* to perform it. In this sense, we consider that the execution of this action should be performed, as long as possible, in dedicated areas called Flight Termination Points (FTPs). These points and the surrounding, protected area should be located in unpopulated areas or over the sea, and away from critical infrastructures [25, 65]. For this reason, we consider that these areas should be specified in the Mission Plan, and should be segregated by ATC. In summary, the flight termination procedure should be preceded, whenever possible, by a strategic phase for reaching the most convenient FTP.

4.4 Safety aspects relating to the software development

For the Mission Manager to be considered a *safe* system, it is necessary not only to provide it with ACM functions, but also develop it following a reliable software development process [164]. The software project for aerospace applications like the one presented in this work shall demonstrate an adequate level of confidence in safety to comply with the aerospace standards for certification [118, 131]. The reference manual for the avionics industry is the DO-178 standard. It defines an explicit correlation between the severity of system hazards and the scrutiny to which that system is subjected. In particular, it establishes five *software levels* (Level A to E) related to the effect of five failure conditions (Catastrophic to No safety effect). Several approaches have particularized the definition of these effects to the case of UASs, see a comparison in [169]. The relevant aspect is that, in DO-178, each level has a number of objectives that must be met in the software development process, so the verification effort increases with the software level of a component.

In regards to UAS, the EASA Concept of Operation states that system hazards are operation-centric [47]. According to this regulatory framework, unmanned aircraft operating in the open category are not subject to certification because the impact on safety of a software error is low: in this category, the aircraft is operated in VLOS and below 150 m, so a dedicated remote pilot is assumed to be present at all times of the operation. Accordingly, the remote pilot can

take control of the vehicle at any point in the mission, and specifically after a contingency occurs. It can therefore be reasonably justified that the embedded software has no effect on the operation in terms of safety. However, large UAS operating in the specific category, and ultimately in the certified category, can eventually operate BVLOS of the remote pilot. In these cases, if the C2 link is lost, the software is essentially replacing the remote pilot; and, for this reason, it becomes safety-critical and must be verified [164].

In this section, we discuss the impact on safety of each architectural component of the proposed SMMS. We also propose the use of partitioning as a means of fault contention, and we allocate the different software components to a partitioning scheme that allows the software level of some architectural components to be downgraded. Finally, we propose the use of formal methods to facilitate the analysis and verification of the critical software.

4.4.1 Preliminary software level determination

Based on the safety framework in Fig. 3.3, it is possible to ensure safety as long as the RPAS has the ability to command the flight termination action in an expeditious manner, at any time and under any condition of the RPAS. In the ACM scheme proposed in this chapter, the Safety Monitor is the software component with the ability to command this action autonomously, without the collaboration of any other system. Accordingly, in the safety assessment, a software error causing the loss of function of either the Safety Monitor or the FTS will have catastrophic effects. For this reason, these two systems will be considered the hardcore components for maintaining safety and should be assigned the highest software level.

In the case of the Contingency Manager and the Mission Manager components, the loss of any of these systems basically means that the flight path of the vehicle cannot be controlled. According to [75], this is thought to have hazardous effects on the operation of the RPAS as long as the vehicle is able to initiate a flight termination procedure; otherwise, such fault would have catastrophic effects too. Consequently, assuming that the hardcore components of the architecture are able to safely terminate the flight, the Contingency Manager and the Mission Manager could be assigned a software level "B".

The problem that emerges is that, according to the DO-178 standard, software components with common modes of failure cannot have different software levels [131]. That is, if a software error occurs at some component, and this error affects other functional components, then all these components should be as-

signed the software level associated with the most severe failure condition. In the case of the SMMS in Fig. 4.6, if a serial implementation is conceived, then the loss of one component like the Mission Manager could cause a total system loss; and, as a result, all the components in the architecture should be assigned the same (highest) software level, which adds to the software development effort.

4.4.2 Architectural strategies for fault contention

To overcome this, the DO-178 standard proposes some architectural choices that can limit the impact of failures to ultimately demonstrate that sufficient independence between software components with respect to their failure modes exists [131]. If this is achieved, it is possible to separate safety-critical functions to independent modules with independent failure modes. Consequently, it is possible to downgrade the software level of some components by means of an appropriate architectural choice.

One of the possible architectural choices is *redundancy*. Redundant configurations mitigate hazards by replicating system components in different processors. So if a fault (either a software fault or a hardware fault) causes a system malfunction, the affected component can be replaced by a backup copy that provides the same functionality, but one executed on different hardware. The use of *dissimilar*, redundant components is required to avoid software development errors in this case. Redundant systems are common in aviation. Dual and triple redundancy is often required in critical aircraft systems [46, 86]. However, having dedicated hardware for each replicated application increases the system complexity, as well as development costs. Moreover, in the case of UAS, the reduced size and weight restrictions make redundancy hard to implement.

Another architectural choice that can limit the impact of failures is *partitioning*. Partitioned architectures, called Integrated Modular Avionics (IMA) architectures in aerospace [6], provide protection and separation among applications running on the same hardware. This way, failures occurring in one partition are not propagated to other partitions.

The support for IMA architectures is defined by ARINC-650 and ARINC-651 documents that specify general purpose hardware and software standards, and by ARINC-653, which specifies the Application Programming Interface (API) [6]. As it was introduced in Sec. 1.2, one of the goals of this work is to validate the XtratuM hipervisor as an execution environment for avionics

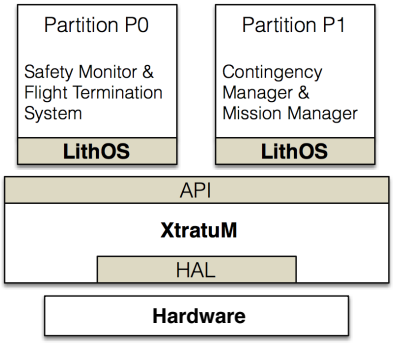


Figure 4.10: Proposed partitioning scheme for the SMMS.

applications. The XtratuM hipervisor provides support for IMA architectures [113], and is compliant with the ARINC-653 API through LithOS, a guest real-time Operating System (OS) [112]. Next paragraphs discuss how to exploit the IMA concept to allocate the different SMMS components to different XtratuM partitions.

Definition of the partitioning scheme

The definition of the partitioning scheme is a design issue. The simplest solution would be to allocate all the software components to a single partition. However, this does not exploit the IMA concept and implies that a fault occurring at some component can produce a total system loss. The opposite is allocating each function to a separate partition, but this unreasonably increases the system complexity. To overcome this, we propose an intermediate solution based on allocating software components according to their impact on overall safety.

Accordingly, the proposed partitioning scheme will be composed of two partitions, represented in Fig. 4.10. In this scheme, partition P0 allocates the software components that can contribute to failure conditions with catastrophic consequences; these are the Safety Monitor and the Flight Termination systems. Most of the software verification effort will fall on this partition. By contrast, partition P1 allocates the components whose failure is considered to have hazardous effects in the safety assessment, i.e. the Contingency Manager and the Mission Manager systems.

The advantage of this partitioning scheme is that faults occurring at a component allocated to partition P1 will not be propagated to any component in partition P0. In other words, a software error affecting partition P1 will not have catastrophic consequences because partition P0 is still able to perform the flight termination action (even if P1 fails). As a result, it is possible to downgrade the software level of the Contingency Manager and the Mission Manager from level "A" (in case of a serial implementation) to level "B" and reduce the number of verification objectives.

4.4.3 Prototyping and deployment to XtratuM

A prototype model of the SMMS software architecture presented in this chapter will be developed in Appendix B. The proposed model will be developed using the Model-Based Design (MBD) methodology in Matlab/Simulink. MBD is a design methodology for embedded systems that can be used to analyze, simulate, specify and deploy software algorithms using block diagrams and state charts [129]. Some remarkable advantages of this methodology are:

- It allows rapid prototyping;
- It introduces testing from the early stages of the development process, reducing development costs and effort; and
- It allows to automatically generate, deploy and verify the embedded code from the design model.

It is to be noted that MBD is supported by the last version of the DO-178 document (named version C) through the specific supplement DO-331 [133]. This supplement identifies how the guidance in DO-178C may be modified when software is developed using MBD. For example, it describes how simulation can be used to satisfy some certification objectives in place of traditional testing, and how model coverage analysis should be performed to check the structural coverage achieved during model simulations [92, 156]. Although this work does not intend to follow the certification activities to a full extent[1], simulation tests will be used in Chapter 7 to validate the functional behavior of the prototype under study.

In addition, when using MBD, the source code for the embedded system is generated from the design model. Code generation can be performed manually

[1]Further details about certification activities using MBD can be found in [156].

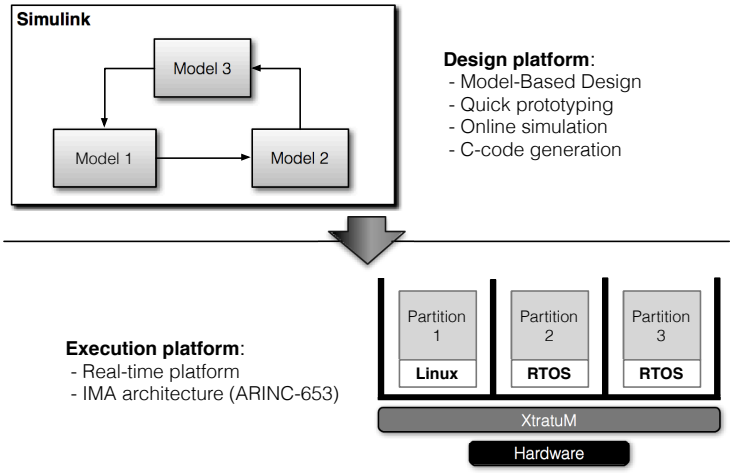


Figure 4.11: Automatic deployment process from Simulink to XtratuM.

but also automatically. Automatic code generation is a helpful tool for reducing coding errors. Matlab&Simulink provide automatic code generation tools named Matlab Coder, Simulink Coder and Embedded Coder. In Appendix D, these tools will be exploited to automatize the process of porting Simulink models to XtratuM. The proposed procedure, schematized in Fig. 4.11, allows both:

1. To identify the desired partitioning scheme from the Simulink model; and
2. To generate the application code for an ARINC-653 compliant system in an automatic manner.

Therefore, once the SMMS model is designed and tested using Matlab&Simulink, the automatic code generation tools in Appendix D will be able to identify the desired partitioning scheme (Fig. 4.10) from the model and to generate the application code for XtratuM based on this partitioning scheme.

4.4.4 Formal methods for software verification

From the above discussion, fault contention mechanisms and reliable software design methodologies must be followed to assure that critical software is of high quality and error-free. Practices like *defensive programming* are also suggested for reducing software complexity and ultimately limiting the chance of introducing errors. For example, it is possible to limit the use of a programming language to a subset; this avoids the use of structures that could lead to non-deterministic behaviors. But the key mechanism for error prevention is the use of verification methods that ensure that errors entered into the software lifecycle get detected. In the previous version of the DO-178 standard (named version B), the verification process mostly relies on generating a large set of test cases for different steps in the development process. However, the coverage of these tests cannot demonstrate the total absence of errors in the code [163]. Another shortcoming is that safety-related requirements are often difficult to test following such verification strategy [29, 120].

The last version of the DO-178 standard (version C) introduces the use of *formal methods* through the specific supplement DO-333 [134]. A formal method is defined as a formal analysis carried out on a formal model [131]. A formal model is a system description expressed with a formal specification language, i.e. with precise syntax and formal semantics. Such models can be useful in several phases of the development process, such as for the simulation of the system behavior or the reasoning over such system representation, among others. The most extended use is formal verification, though, which is the aim of the DO-333 supplement and of this work as well.

A formal specification makes it possible for system properties to be defined in a precise, consistent and complete way [116]. When used for verification, properties are deduced from the system requirements, and in the case of safety-critical system, from the safety requirements. Then, formal methods can be used to verify these properties against the model to reveal design errors or model inconsistencies, and ultimately to provide verification evidence for the certification process. One of the advantages of formal methods is that safety-critical properties can be checked more easily than with a conventional testing strategy [29, 120, 163].

The DO-333 supplement allows three classes of formal methods to satisfy certification objectives: theorem proving, abstract interpretation and model checking. This latter method is the focus of this work, while the remaining ones are

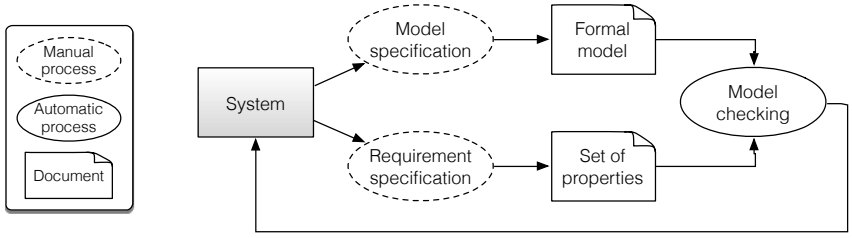


Figure 4.12: Model checking process. Source: [120].

omitted for brevity. In model checking, the model is represented as a FSM[2], and the properties are formalized using temporal logic. Both the model and the properties are deduced manually from the system that is being verified. Then, the entire state space of the FSM is analyzed to check the validity of the formal properties, with the advantage that this analysis is fully automated. If a property is not satisfied, a counter-example is generated, and the model or the property can be refined and analyzed again. The resulting process is schematized in Fig. 4.12.

The application of model checking techniques to safety-critical avionics systems has increased in recent years [29, 163, 170]. In Appendix A, we study how to exploit this method in the software development of the ACM system presented in this work. In particular, we apply formal techniques to the design and verification of a particular contingency management policy, including the Safety Monitor model and the Contingency Plan.

[2]Recall that this is one of the reasons why we opted for modeling the Safety Monitor and the Contingency Manager using FSMs, see Figures 4.7 and 4.8.

Chapter 5

Reconfigurable Mission Plans

5.1 Definition

Reconfigurable Mission Plans are based on the idea that an RPAS has a preferred or *nominal route*, though it is possible to modify this route at flight time to respond to contingencies or pilot decisions. The alternate routes that result from changing the nominal route must be deduced from the specifications in the Mission Plan. To do so, all possible deviations from the nominal route should be specified in advance in the Mission Plan. The aim is to achieve completely predictable behavior, especially when the RPAS is in autonomous mode. In manual mode, the remote pilot can always override the Mission Plan if necessary.

As an example, consider a mission where the current goal is reaching an airfield via a preferred route. If the C2 link is lost, the Mission Plan can be reconfigured so the new mission goal can be to regain the C2 link signal, for instance. If the C2 link is restored, the original goal can be resumed, perhaps by using a different route. This way, the notion that a mission goal can be achieved through different alternative routes is another feature of Reconfigurable Mission Plans.

From the above discussion, a Reconfigurable Mission Plan is defined as a Mission Plan that describes a nominal route and a number of alternative routes to

be flown in case of contingency. The specification of a Reconfigurable Mission Plan includes the set of all the possible *goals* of a mission and the associated routes that can be used to achieve these goals. One of these goals will be designated as the *nominal goal*; it defines the intended behavior if no contingency occurs. When some contingency occurs, the Mission Plan also defines alternative goals and one or more routes to manage the contingency. Alternative goals make use of different contingency handlers, such as the ones proposed in the contingency management scheme in Chapter 3. In manual or automatic mode, it is the remote pilot who decides on following the Mission Plan directions. In these modes, the Contingency Manager simply plays an advisory role. In autonomous mode, the Contingency Manager always uses the Mission Plan specifications to guide the aircraft.

From the implementation point of view, the flight path of a Reconfigurable Mission Plan route is structured into *segments*. A segment is a sequence of *legs* that correspond to a specific phase of flight or to a flight procedure (e.g. departure, arrival, etc). The Mission Plan comprises the set of all possible segments that can be used in a mission: departure procedures, routes, payload related procedures, landing procedures, etc. The set of all possible routes in a Reconfigurable Mission Plan can be deduced from a graph that results from the union of these segments. This graph will be called *Mission Graph*.

The Mission Graph allows to configure optimal routes that are effective to achieve a given goal. An optimal route is one that minimizes the associated *cost* while being effective for the intended goal. In this work, the cost of a route will be defined in terms of the risk posed to third parties, either on ground or in the air. Therefore, the aim of a Reconfigurable Mission Plan is to provide the route that reduces the risk of the RPAS operation.

The formal description of a Reconfigurable Mission Plan and all its components is introduced below using Graph Theory concepts. Then, Sec. 5.2 presents a formal specification of all the components of a Reconfigurable Mission Plan using Unified Modeling Language (UML) models. Finally, Sec. 5.3 introduces graph search algorithms of the Graph Theory to find minimum cost routes in the Mission Graph.

5.1.1 Waypoints

A waypoint is the most elementary object in a Mission Plan. It identifies a *point* over the earth surface using its WGS-84 coordinates. These points are used by other Mission Plan objects, like goals, routes, etc. In PBN, all fixes in a flight procedure are specified as waypoints and not by their relative position to some ground NAVAID.

The waypoint declaration of a Reconfigurable Mission Plan usually specifies points with constant longitude and latitude, similar to how they are declared in a navigation database. In addition, a Reconfigurable Mission Plan includes a variant of the waypoint class which is the *variable waypoint* class. This class is used to specify waypoints whose coordinates may vary depending on aircraft performance, and for this reason their coordinates do not appear in the initial waypoint declaration of the Mission Plan. These types of points will be used by some leg types in Sec. 5.1.3.

5.1.2 Mission goals

A mission goal represents a kind of "motivation" that leads the RPAS behavior and trajectory. Examples of such goals are flying to a location, landing, loitering, etc. In a Reconfigurable Mission Plan, the proposed goal types are gathered in Table 5.1. The proposed list is mostly derived from the ICAO recommendations on contingency handling: it includes the four strategic contingency procedures in Table 4.3, plus the "fly-over" goal (which will be used to define route constraints). Once in flight, there is only one active goal at a time, which is selected by the remote pilot or the Contingency Manager, depending on the system configuration.

Formally, a goal has an *associated location* (a loiter point, an airport, etc.) that must be reached to achieve the goal, and an *enabled procedure* that is flown upon reaching this location. The aforementioned Table 5.1 also shows the associated locations and procedures for the proposed goals. For example, the "loiter" goal is associated with a loiter point; reaching a loiter point allows the RPAS to perform a holding procedure.

One of the mission goals specified in a Reconfigurable Mission Plan is designated as the *nominal goal*. This goal describes the primary intended RPAS mission and it is the default goal to be achieved.

A goal can consist either of a single goal or a sequence of sub-goals or *stages*. For example, a goal may have an initial stage to fly to the operations area, a

Table 5.1: Goal types in the proposed Reconfigurable Mission Plan.

Goal type	Associated location	Enabled procedure
Fly-over	Waypoint	None
Loiter	Loiter point	Hold position
Regain signal	Waypoint	Climb trying to regain the C2 link or the GNSS signal
Land	Airport IAF	Approach procedure
Flight termination	Flight termination point	Flight termination action

stage to perform the payload task inside the operations area, a stage to exit the operations area and a final stage to land at some given location. Stages can be also used to meet route constraints. For example, imagine accomplishing a "land" goal requires flying a standard arrival procedure with some required intermediate waypoints. This could be modeled as a sequence of "fly-over" stages followed by a "land" stage.

The *associated location* of a goal can be specified as a single point or as a set of alternative points. For example, a "land" stage may have a set of alternative Initial Approach Fixes (IAFs); each IAF corresponding to a different landing procedure to be chosen according to the airport configuration. Similarly, the operations area may have several entry points or several exit points that the pilot can select. In general, only one of the listed points can be enabled at a time: the decision maker agent (usually the pilot) must choose between the different options according to the current state or condition.

5.1.3 Legs

As it was introduced at the beginning of this section, a Reconfigurable Mission Plan route is structured into segments. A segment is defined as a sequence of legs. Legs are the most basic maneuvers in a segment. In conventional airliners, legs are specified in the FMC using the *path terminator* concept of ARINC-424. ARINC-424 is a standard that defines the format of navigation databases [5]. The standard defines a number of entities (like waypoints, airports, airways, NAVAIDs, path terminators, etc), their attributes and their format. ARINC-424 path terminators have been adopted by ICAO in Doc. 8168 OPS/611 [85] to describe the basic flight maneuvers of RNAV flight procedures.

Path terminators provide higher flexibility and richness of behaviors than simple "waypoint navigation" used by most Mission Plan proposals. A path termi-

Table 5.2: Extended Path and Terminator codes.

Path		Terminator	
Code	Description	Code	Description
S	Scanning pattern	L	Lap limit
		T	Time limit

nator is defined using two letters, the first one describing the type of *path*, and the second, the *termination condition*. For example, the most common path terminator is the *track to fix*, coded as "TF". This coding defines a maneuver that the FMC must implement as navigating a *path* that is the shortest orthodromic track between two fixes. The *terminator* indicates that the maneuver ends when the target fix is reached. But path terminators provide additional path types (like arcs with constant radius, paths with constant heading, paths with constant course, etc), and additional types of terminators (reaching an altitude, manual termination, etc).

The advantage of using path terminators is that it allows flight procedures to be defined using PBN standards [86] and, ultimately, provides compatibility with commercial FMC technologies. Path terminators were originally intended to describe airliner maneuvers. We propose to extend this concepts to RPAS. This requires to introduce new path terminators to describe RPAS specific maneuvers in a way that is close to the ARINC-424 standard. Our proposal is a resulting set of path terminators referred to as Extended Path Terminators (EPTs). New features introduced by EPTs are:

1. *New types of paths and terminations.* The proposed paths and terminators are gathered in Table 5.2, though more options are possible. In this case, examples of new paths include the "S-path" to describe scanning patterns around a geographical area; while examples of new terminators are the "L-terminator" (used to specify a maximum number of laps) or the "T-terminator" (used to specify a given time limit). According to this, some examples of new EPTs would be: SL (scan for a number of laps), ST (scan for a given time) or SM (scan to manual termination).

2. *Combinations of paths and terminators not included in the standard.* Examples of new combinations of existing paths and terminators not defined in the standard include: RA (helicoidal ascents or descents to an altitude), or RM (orbit until pilot intervention), for instance.

On the other hand, EPTs also restrict some features defined by the ARINC-424 standard: since RPAS navigation is mostly based on GNSS, path terminators based on the use of ground NAVAIDs will be restricted in practice.

Table 5.3 summarizes the EPTs used in this approach. It includes 18 EPT codings: 11 out of the 14 standard path terminators defined for RNAV operations [86], plus 7 extended procedures for RPAS.

As this table shows, each EPT requires a number of *definition parameters*: some of them are compulsory, while others are optional. For example, the *course to altitude* (CA) needs the following parameters: the reference course, the altitude limit, and optionally the speed limit and the required vertical path angle.

An important remark regarding the definition parameters of FM, VM, RM and SM procedures is that, although their termination condition is the manual intervention of the remote pilot, they include a limit value which specifies a time-out condition. This is an alternative termination condition or safety mechanism for preventing indefinitely behaviors in case that the remote pilot is out of the control loop due to a C2 link loss.

Another important remark is that path terminators can lead the plane to a well defined waypoint or to a variable waypoint. The termination point is a waypoint in the case that the path terminator specifies a fix as the terminator. Conversely, if the path terminator specifies an altitude or some other termination condition, then the termination point is a variable waypoint since different planes can reach that altitude (for instance) at different points depending on their performance.

5.1.4 Segments

A *segment* is defined as a sequence of legs that correspond to a phase of flight or to a flight procedure. As stated in the Mission Plan definition, the Mission Plan specifies the set of all possible segments of a mission.

Every segment is tagged with a segment *type*. Six types of segments have been defined according to the phases of flight identified by Eurocontrol for RPAS missions [54]: departure, en-route, operation (i.e. performing the payload task in the operations area), ingress/egress (transitions between en-route and operation), and arrival.

The specification of the sequence of legs is subject to the following constraints:

Table 5.3: Extended Path Terminators and definition parameters.

Description	coding	waypoint1	waypoint2	fly-over	course	path length	altitude1	altitude2	speed limit	radius	turn direction	limit value	vertical angle
Course to altitude	CA				✓				O			1	O
Course to fix	CF	✓		O	✓		O	O	O				O
Direct to fix	DF	✓		O			O	O	O				
Fix to an altitude	FA	✓			✓				O			1	O
Fix to manual	FM	✓			✓		O	O	O			6	
Initial fix	IF	✓					O	O	O				
Radius to fix	RF	✓	2			✓	O	O	O		✓		O
Track to fix	TF	✓		O		✓	O	O	O				O
Heading to altitude	VA				3				O			1	O
Heading to intercept	VI				3		O	O	O				
Heading to manual	VM				3		O	O	O			6	
Radius to altitude	RA	2							O	✓	✓	1	O
Radius to lap number	RL	2			4		O	O	O	✓	✓	5	
Radius to manual	RM	2					O	O	O	✓	✓	6	
Radius to time	RT	2					O	O	O	✓	✓	6	
Scan to lap number	SL	✓	✓		7		O	O	O	✓		5	
Scan to manual	SM	✓	✓		7		O	O	O	✓		6	
Scan to time	ST	✓	✓		7		O	O	O	✓		6	

✓– Required
O – Optional
1 – Altitude limit (at or above)
2 – Arc center
3 – Heading not course
4 – Reference bearing
5 – Lap limit
6 – Time limit
7 – Initial course
Shaded – Not applicable field

Table 5.4: Valid initial and final legs per segment type.

Segment type	Initial leg	Final leg
Departure	IF	CF, DF, RF, TF
En-route	IF	TF
Ingress	IF	CF, DF, RF, TF
Operations	IF	CF, DF, RF, TF
Egress	IF	CF, DF, RF, TF
Arrival	IF	CF, RF, TF

Definition 1 (Well formed segment) *A segment is said to be* well formed *iff its sequence of legs meets the following rules derived from the ARINC-424 standard*[1]*:*

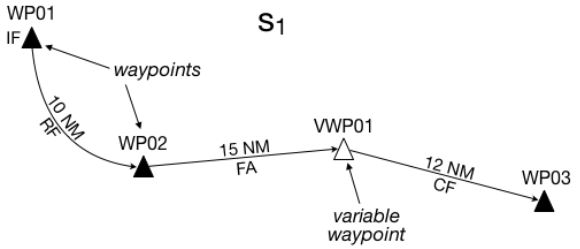
WFS1 *Permitted legs per segment type.* Not all EPT types are permitted in every segment type. For example, en-route segments can be composed of legs coded as IF and TF only. Similarly, EPTs for performing scanning patterns are limited to operations segments.

WFS2 *Permitted beginning and ending leg types.* The ARINC-424 standard [5] defines a table of valid initial and final path terminators. Table 5.4 is an extension of the ARINC-424 proposal considering the new phases of flight proposed by Eurocontrol for RPAS [54].

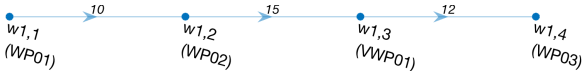
WFS3 *Permitted leg sequences.* The ARINC-424 standard [5] also defines a table of valid sequences of legs. The general rule is that RF and TF legs should be preceded by legs whose termination point is a waypoint. For example, the CA/TF sequence is not allowed because the TF requires a previous fix for defining the flight path. Another prohibited sequence is DF/RF because the resulting flight path is not predictable. We have derived an equivalent table of permitted EPT sequences that has been omitted for brevity.

Regarding Table 5.4, it must be noted that the initial leg of every segment type is coded as IF. This initial leg must be associated with a waypoint with well defined coordinates (not a variable waypoint), except in departure segments. In such segments, the starting condition is any point along the runway, so this point must be identified using a variable waypoint. The *de facto* starting leg in departures will be the second leg which must be coded as CA, CF, VA, or

[1] They are also the basis for the corresponding rules in ICAO Doc. 8168 OPS/611 [85].



(a) Navigation chart view.



(b) Graph view.

Figure 5.1: Example of segment representation. Note that, in this case, the cost associated with the segment is expressed in terms of path length.

VI [85]; but the IF leg is still required to support the definition of the segment, and particularly the *segment graph*, which will be introduced below. Note also that the permitted final legs in Table 5.4 are always EPTs ending at a fix; this will allow segments to be connected at well defined waypoints (not at variable waypoints), thus making the flight procedure more deterministic.

From the point of view of Graph Theory, a well formed segment can be modeled as a *directed path* [14]. A directed path (sometimes called *dipath*) is a sequence of edges that connects a sequence of nodes, and where all edges have the same direction. In our case, graph nodes represent the *termination points* of the legs of the segment (either a well defined waypoint or a variable waypoint), and directed edges are the *paths* between these points. Figure 5.1 compares a segment graph with its traditional representation of navigation charts.

Segment nodes will be named $w_{i,j}$, where i is the segment identifier, and j is the identifier of the segment leg (or more precisely its associated termination point). Notation $w_{1,1} \rightarrow w_{1,2}$ expresses that there is a directed edge from node $w_{1,1}$ to node $w_{1,2}$. This means that waypoint $w_{1,2}$ is *reached* (navigated) right after $w_{1,1}$ in segment 1.

By definition of directed path, the set of nodes of a segment are are totally ordered. As a result, a segment node can be reached from any node that precedes it in the ordering. If we define the in-degree of a node as the number of ingoing edges, the in-degree of all segment nodes is 1, except for the first

node (called the source node), that is 0. In the same way, if we define the out-degree of a node as the number of outgoing edges in that node, the out-degree of all segment nodes is 1, except for the last node (called the sink node), that is 0.

In order to optimize the *route configuration* in a Reconfigurable Mission Plan, the edges of a segment graph are *weighted*, with the weight representing the cost of flying between two consecutive nodes. In general, the cost could be defined in terms of the resulting path length (as in Fig. 5.1), or the resulting flight time, or the fuel consumption, etc. However, in this work, the cost will be measured in terms of the risk it entails for third parties (either on ground or onboard other aircraft) when the RPAS flies the given segment. How to compute this metric is a problem that will be discussed in Chapter 6.

5.1.5 Routes

As it was introduced at the beginning of this section, Reconfigurable Mission Plans allow to specify all the different routes that an aircraft can fly. One among all the possible routes must be declared as the *nominal route*. This route is the only one that is statically declared in the Mission Plan. The remaining alternative routes are *dynamically* derived from the segments declared in the Mission Plan. The set of all the possible routes is described by the *Mission Graph*. The Mission Graph is defined as the *union* of all the segments of the Mission Plan.

The union operator defines the conditions that allow to fly from one segment to another. The *necessary condition* for setting a path between two segments is that they shall have a waypoint in common. This is not a *sufficient condition*, though. In order for the resulting path to be consistent, it is also necessary to account for the position of the common waypoint in the sequence of legs of each segment, and for the phase of flight of the segments under consideration. Therefore, the definition of the union operator is similar to the notion of union in Graph Theory, but it should be particularized to cope with some restrictions on the path construction:

Definition 2 (Segment union) *Given the graph of two well formed segments s_a and s_b, the* union *of s_a with s_b, denoted as $s_a \cup s_b$, is the directed graph that results out of performing the following two operations:*

Table 5.5: Permitted phase of flight transitions.

		Next segment (s_b)					
		Departure	*En-route*	*Ingress*	*Operations*	*Egress*	*Arrival*
Current seg. (s_a)	*Departure*	–	✓	✓	✓	–	✓
	En-route	–	✓	✓	✓	–	✓
	Ingress	–	–	–	✓	✓	–
	Operations	–	✓	–	✓	✓	✓
	Egress	–	✓	✓	✓	–	✓
	Arrival	–	–	–	–	–	–

SU1 *Performing the disjoint union of both segment graphs.* This implies that, in the resulting graph, the nodes of s_a and the nodes of s_b will have no elements in common (even though they may reference the same waypoints).

SU2 *Creating additional edges between nodes associated with the same waypoint, iff the following conditions hold:*

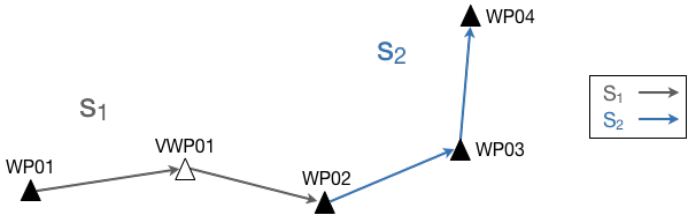
SU2.1 The waypoint is not associated with the termination point of the first leg of s_a.

SU2.2 The waypoint is not associated with the termination point of the last leg of s_b.

SU2.3 The phase of flight transition from s_a to s_b is permitted by Table 5.5.

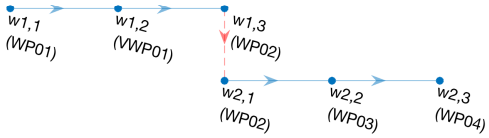
The new edges created when condition **SU2** holds will be called *transition edges* because they allow to fly between segments. An important remark is that, since the nodes connected by a transition edge are geographically co-located, the cost of flying a transition edge will be zero.

To illustrate this definition, imagine the two segments s_1 and s_2 in Fig. 5.2a; assume that both of them correspond to the en-route phase. The graph obtained after performing the disjoint union of these segments (operation **SU1**) is shown in Fig. 5.2b. As it can be observed, the resulting graph is not connected: there are two subgraphs, each one representing the graph of a segment, but they have no elements in common. In order to connect these graphs, it is nec-



(a) Navigation chart view.

(b) Disjoint union of segments s_1 and s_2 (operation **SU1**).



(c) Union of segments s_1 and s_2 (operations **SU1** and **SU2**).

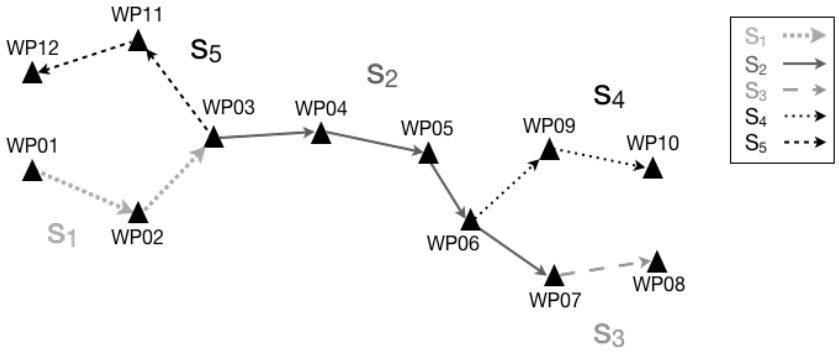
Figure 5.2: Example of segment union. Leg codings, path distances, and edge weights are omitted for simplicity.

essary to perform operation **SU2**. The resulting graph is shown in Fig. 5.2c. As it can be observed, there exists an additional edge from node $w_{1,3}$ to node $w_{2,1}$ (depicted as a dashed line) since both nodes are associated with the same waypoint (WP02) and conditions **SU2.1**, **SU2.2** and **SU2.3** hold. This way, it is possible to fly from s_1 to s_2 through this transition edge.

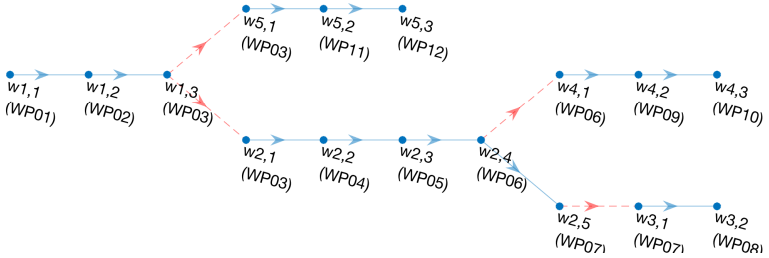
An important property of the segment union is that the union operator is not commutative:

$$s_a \cup s_b \neq s_b \cup s_a \tag{5.1}$$

In the above example, the union of s_2 with s_1 ($s_2 \cup s_1$) does not allow to fly from s_2 to s_1 through the common waypoint WP02 because conditions **SU2.1** and **SU2.2** fail in this case. Therefore, in order to obtain the Mission Graph,



(a) Navigation chart view.



(b) Graph view.

Figure 5.3: Example of Mission Graph.

it is necessary to perform the segment union of each segment declared in the Mission Plan with all the remaining segments.

Another important property of the segment union is that, although the graph of a segment is totally ordered, the graph that results out of performing the segment union might be partially ordered. This might occur if the resulting graph has multiple source nodes and/or multiple sink nodes; or if it presents *cycles*. To illustrate this, imagine a more complex example like the one in Fig. 5.3a. In this case, there are 5 segments which connect 12 waypoints. The Mission Graph that results out of performing the union of all these segments is presented in Fig. 5.3b. As it can be observed, it presents one source ($w_{1,1}$) but three sinks ($w_{3,2}$, $w_{4,3}$, and $w_{5,3}$), so it is not possible to define a total order.

Once the Mission Graph has been defined, we can define mission *routes* based on the concept of *reachability* between nodes of the Mission Graph:

Definition 3 (Node reachability) *Let $w_{a,i}$ and $w_{b,j}$ be two nodes of the Mission Graph. Node $w_{b,j}$ is said to be* reachable *from node $w_{a,i}$, denoted as $w_{a,i} \Rightarrow w_{b,j}$, iff there is* at least *one directed path from $w_{a,i}$ to $w_{b,j}$ in the Mission Graph.*

Accordingly, a route between two nodes $w_{a,i}$ and $w_{b,j}$ of the Mission Graph exists iff $w_{a,i} \Rightarrow w_{b,j}$. The resulting route is a directed path in the Mission Graph that can traverse the nodes of different segments. The set of all the nodes traversed by a route is totally ordered. The node in which the directed path reaches a segment is called the *entry point* of that segment in this route. The node in which the directed path leaves a segment is called the *exit point* of that segment in this route.

As a result, a route can be specified as the sequence of the segments traversed by a directed path in the Mission Graph, along with the entry point and the exit point of each segment in the sequence. This will be denoted as:

$$r = \langle \text{WP0} \xrightarrow{s_0} \text{WP1} \xrightarrow{s_1} \text{WP2} \ldots \rangle \tag{5.2}$$

Where WP0 and WP1 are the entry and exit points of s_0, and WP1 and WP2 are the entry and exit points, respectively, of s_1. For example, a possible route in the Mission Graph example of Fig. 5.3 is:

$$r_0 = \langle \text{WP01} \xrightarrow{s_1} \text{WP03} \xrightarrow{s_2} \text{WP06} \xrightarrow{s_4} \text{WP10} \rangle \tag{5.3}$$

Finally, a route is considered to be *effective* for achieving a specific mission goal if it traverses all the associated locations of that goal. For example, if we assume that a mission goal for the mission example in Fig. 5.3 is declared as "fly over waypoint WP05; then perform the landing procedure associated to waypoint WP10", then r_0 in Eq. (5.3) will be effective in achieving this because it traverses both waypoints in the required order.

5.1.6 Mission boundaries

In addition to the route information, the Mission Plan allows mission boundaries to be specified. A mission boundary is an airspace volume with well-specified limits which are monitored so as to produce a contingency if the limits are close to being trespassed. Therefore, the reason the specification of these volumes is entered into the Mission Plan is to detect the mission bound-

ary limits violations event while in-flight. As it was discussed in Sec. 3.2, there are two main *types* of boundaries: no-fly zones, and geofenced areas. In no-fly zones, the contingency is produced when the aircraft gets close to entering this area. In geofenced areas, the contingency is produced when the aircraft gets close to exiting this area.

5.2 Specification

At present time, we have not addressed the subject of designing a specific language or syntax to formally specify Mission Plans other than UML. Nevertheless, UML is a powerful representation that allows to derive a different specification or representation through a code generation process. This representation can be generated for a specific target system. For example, in our case we test Mission Plans on a Mission Manager prototype in Matlab/Simulink; so we need to translate the Mission Plan specification in UML to the Matlab language. This process can be automatized using a code generator. However, we will always refer throughout this paper to the original UML specification.

This section presents the specification of all the elements of a Reconfigurable Mission Plan using UML models. In accordance to UML, the naming convention will be as follows: Classes and interfaces will be denoted as `ClassName`, i.e, capitalizing the first letter of each word; while attributes and objects will be denoted as `attributeName`, always starting with a lowercase. Therefore, the UML model of the `ReconfigurableMissionPlan` class is shown in Fig. 5.4. The structural components of this class are detailed next:

- Attribute `waypoint` is the declaration of a set of relevant waypoints used in other Mission Plan attributes. The specification of this attribute is described in Sec. 5.2.1.
- Attribute `missionGoal` is the definition of the set of all mission goals that a mission may have, one of which is the nominal one. The specification of this attribute is described in Sec. 5.2.2.
- Attribute `segment` is the set of all possible segments that a mission may have. The specification of this attribute is described in Sec. 5.2.4. Before that, attribute `leg` of a `segment` object is described in Sec. 5.2.3
- The possible routes allowed in the Mission Plan are derived from the previous set of segments. One of these routes is the `nominalRoute`, which is the only route statically declared in the Mission Plan. The specification

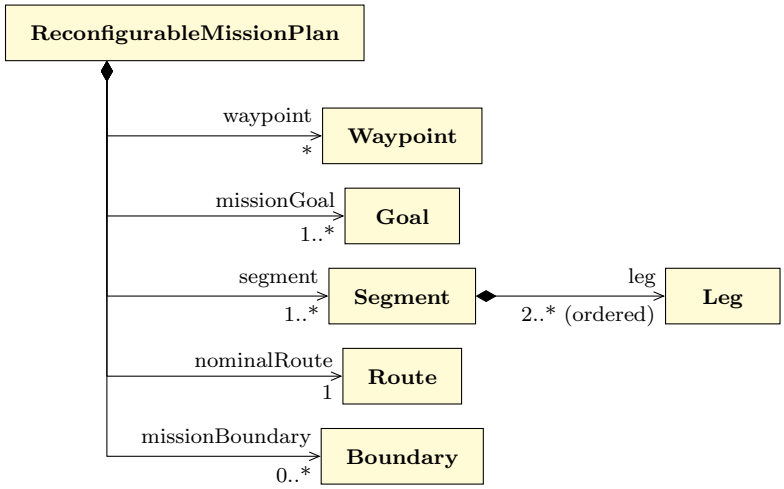


Figure 5.4: Structure diagram of a Reconfigurable Mission Plan object.

of a `route` object is described in Sec. 5.2.5. The remaining routes will be dynamically specified, as will be shown in Sec. 5.3.

- Attribute `missionBoundary` is a set of relevant airspace volumes that a mission may have. The specification of this attribute is described in Sec. 5.2.6.

5.2.1 Waypoints

The UML model of a waypoint is shown in Fig. 5.5 using the abstract class `Point`. This class has two subclasses or specializations: the `Waypoint` class, and the `VariableWaypoint` class. Both subclasses present two attributes: an `id`(entifier) and a `position`. The attribute `position` is specified using a `Coordinates2d` object class for expressing latitude and longitude coordinates in decimal degrees. The difference between each subclass is that in the `Waypoint` class, the `position` attribute is constant (emphasized using capital letters), while in the `VariableWaypoint` class, the coordinates of the position can vary.

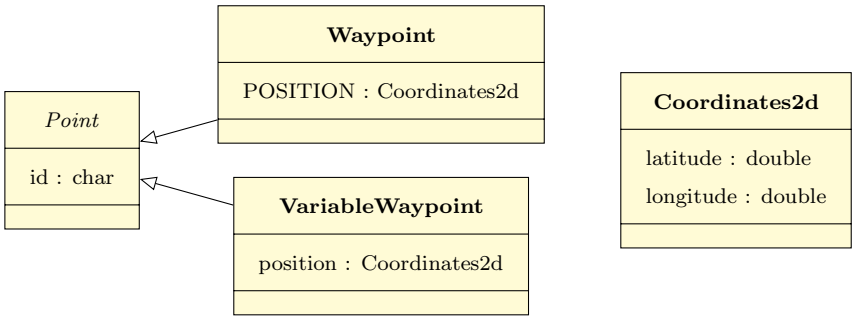


Figure 5.5: Structure diagram of a Waypoint object.

5.2.2 Mission goals

The UML model of the `Goal` class is shown in Fig. 5.6. Their attributes are an `id`(entifier), a `nominal` boolean to indicate if it is the nominal goal, and a sequence (ordered set) of `stage` objects. The UML's composition operator represents a parent-child relationship between a goal and the list of stages with a strong lifecycle dependency. The parent-child relationship means that the goal object has exclusive ownership over the stage object; and the lifecycle dependency states that if a goal object is deleted, then all its stages will also be deleted.

The `Stage` class is labeled with a `type` attribute, which is one of the enumerated values in Table 5.1. Stages have an associated `location` that has to be reached. Recall that several alternative points can be associated with one stage, representing different alternatives, so they are mutually exclusive. In this case, one of them has to be selected as the `enabledVariant`.

Stages also have an `enabledProcedure` that is performed upon reaching the associated location. The `ProcedureAttributes` class is an abstract class that provides flexibility to specify how this procedure is to be performed. For example, the `loiterAttributes` subclass (omitted from the UML diagram for brevity) include the `turnRadius`, the `turnDirection` and a `timeOut` for specifying the maximum allowed holding time, among others. `ProcedureAttributes` may also be used to specify payload-related commands, like engaging the videocamera, dropping a given item, etc., using the `payloadCommand` subclass.

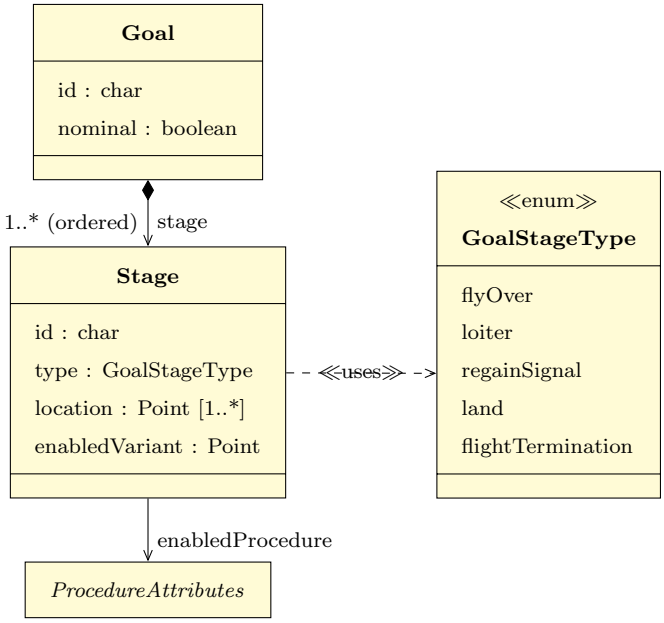


Figure 5.6: Structure diagram of a Goal object.

5.2.3 Legs

The UML model of legs shown in Fig. 5.7 is derived from the discussion in Sec. 5.1.3. There is an abstract class `Leg`, and the different EPTs are subclasses or specializations of this class. Each EPT has its own set of attributes, which basically corresponds to the parameters of Table 5.3.

In addition, the abstract class `Leg` has an attribute common to all EPTs, which is the `terminationPoint`. When the termination condition of the EPT is a fix (IF, RF, etc), the termination point is the `Waypoint` (fix) specified as the terminator of the EPT (attribute *waypoint1* in Table 5.3). Otherwise, this attribute is a `VariableWaypoint`. Since every leg has an associated termination point, there is a bijective relationship between legs and their corresponding termination waypoints.

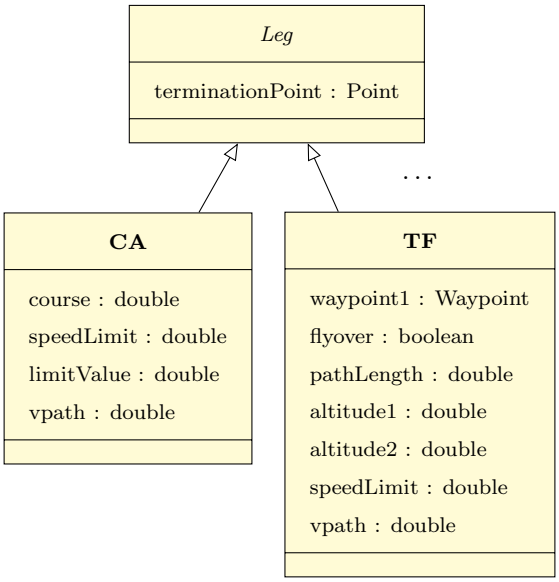


Figure 5.7: Structure diagram of two of the possible specializations of a Leg object.

5.2.4 Segments

The UML model of the `Segment` class is shown in Fig. 5.8. There are two basic attributes: the segment's `id`(entifier) and the segment's `type`. The attribute `leg` is the sequence of legs of the segment, where each leg is a subclass of the abstract class `Leg`.

In addition, the attribute `containmentArea` represents the protected volume that encloses the defined flight path. This attribute describes the navigation performance or the maximum error allowed to fly the segment. As shown in Fig. 5.8, the containment area is modeled as an interface that can be realized in two ways: by means of a `NavigationSpec` object class (a PBN navigation specification that defines the allowed cross track error); or by defining an airspace volume using a `Boundary` object class. For example, a `NavigationSpec` can be an RNP-1 specification that defines a containment area of 1 NM at each side of the intended flight path. `Boundary` objects define the containment area by its geographical limits, as explained in Sec. 5.2.6.

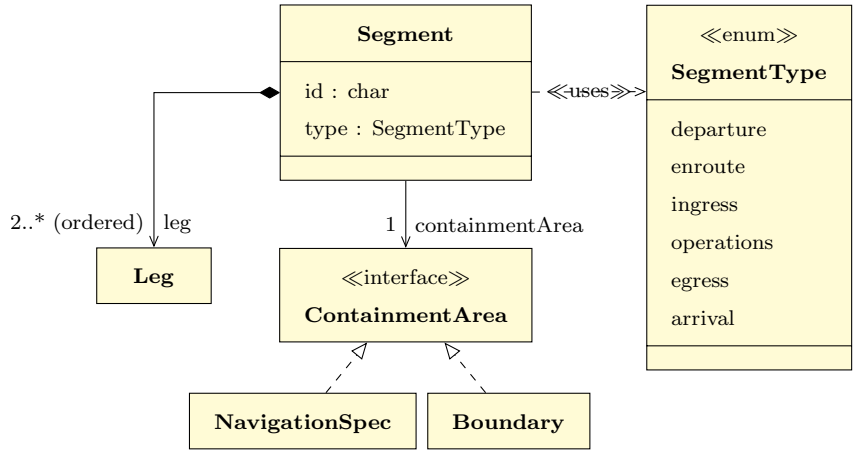


Figure 5.8: Structure diagram of a Segment object.

5.2.5 Routes

The UML model of the *Route* class is shown in Fig. 5.9. The attribute `goal` specifies the mission goal for which the route is intended. The attribute `routeSegment` specifies the flight path of the route as a sequence of `RouteSegment` objects. Each `RouteSegment` is associated with a `segment` instance of the Mission Plan. In addition, the `RouteSegment` also specifies the `entryPoint` and the `exitPoint` of this segment. These attributes are specified using the `Point` class. Note that all `entryPoints` and `exitPoints` are instances of class `Waypoint`, except the entry point of a departure segment, which is an instance of class `VariableWaypoint`.

The UML's direct association between the `RouteSegment` and the `Segment` class has an important implication on the route definition: as depicted in Fig. 5.4, the ownership of the `segment` objects of a route is the `ReconfigurableMissionPlan` class, not the `Route` class itself. In other words: routes are built using segments defined by the Mission Plan. For this reason, a segment used in the nominal route can also be used in another route.

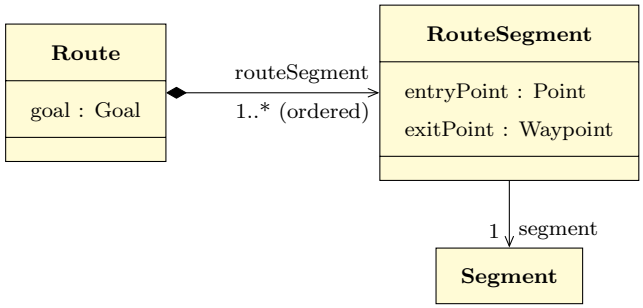


Figure 5.9: Structure diagram of a Route object.

5.2.6 Mission boundaries

The UML model of the `Boundary` class is shown in Fig. 5.10. According to this figure, a `Boundary` class has two possible specializations or subclasses: circular boundaries and polygonal boundaries. The specification of a circular boundary requires a `waypoint` object be provided and the corresponding `radius` attribute defined; while a polygonal boundary is formed from three or more `waypoint` objects in some given order.

`Boundary` objects may also have *vertical limits*: an upper limit, a lower limit, or a given altitude window. Attribute `type` is used to indicate whether it is a no-fly zone or a geofenced area. Finally, `boundary` objects may be associated with a list of *entry* and *exit points*. These points are waypoints that allow an ingress/egress segment to be connected with an operations segment that is geofenced.

5.3 Dynamic, risk-based route configuration

As introduced in this chapter, Reconfigurable Mission Plans provide two types of routes: static and dynamic routes. Static routes are statically declared in the Mission Plan pre-flight. By contrast, dynamic routes are not statically declared, but rather generated at flight time as a function of the Mission Graph, the current position, and the active mission goal. The problem in dynamically (re-)configuring a Mission Plan route can be stated as the problem of *finding the lowest cost route that is effective for achieving the active mission goal from the current position of the RPAS.*

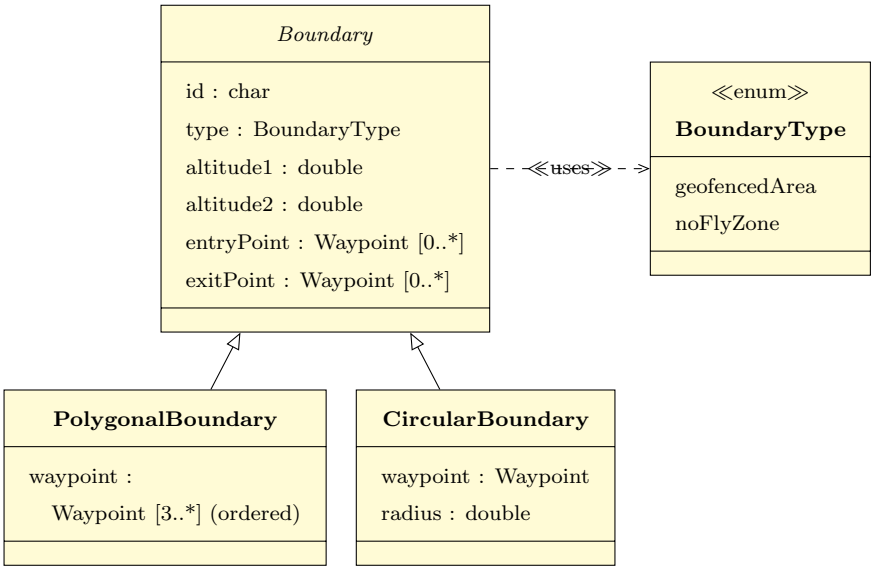


Figure 5.10: Structure diagram of a Boundary object.

As an example, let us return the mission example in Fig. 5.3. Assume that the nominal goal is declared as "fly over waypoint WP05; then land at WP10", and that the nominal route is r_0 in Eq. (5.3). There are also two alternate goals for performing the "flight termination" at waypoints WP08 and WP12, for instance. Now, once in flight time, the RPAS is flying the nominal route, somewhere in between waypoints WP02 and WP03, and a contingency occurs. If the decision maker agent sets the new goal type to "flight termination" (for instance), then the following two routes will be considered to be effective for achieving this goal:

$$r_1 = \langle \text{WP02} \xrightarrow{s_1} \text{WP03} \xrightarrow{s_5} \text{WP12} \rangle$$
$$r_2 = \langle \text{WP02} \xrightarrow{s_1} \text{WP03} \xrightarrow{s_2} \text{WP07} \xrightarrow{s_3} \text{WP08} \rangle$$

If the cost of flying a route r_i is $\bar{c}(r_i)$, and we assume that $\bar{c}(r_1) < \bar{c}(r_2)$, then the optimal route would be r_1 in this case. However, if the contingency had occurred after reaching WP03, then waypoint WP12 would have not been reachable, so the only suitable route in the Mission Graph would have been r_2.

As it was mentioned before, this work proposes to express the cost of a route in terms of the risk entailed for third parties, and particularly, in terms of

the likelihood of causing fatal injuries to people on ground and in the air. In order to compute a risk metric, a probabilistic risk model will be developed in Chapter 6.

Once the cost of all legs in the Mission Plan has been computed, the next problem is finding a route that allows some given type of goal to be achieved. As deduced in Sec. 5.1.5, a mission route is a directed path that connects one given source with one given destination. In this case, the source represents the initial or current position of the RPAS, and the destination is a location that is associated with the target goal. When this goal is composed of multiple sub-goals, the route should connect the origin with the destination, with the difference that this route must traverse all the required intermediate positions.

Since initial position, destination position, and possible intermediate positions are associated with nodes of the Mission Graph, the problem of dynamically configuring a Reconfigurable Mission Plan route can be solved optimally by addressing the Graph Theory's *shortest path problem*: the problem of finding the path between nodes that minimizes the sum of the weights of its edges. There exist multiple algorithms that solve the shortest path problem in the literature: Bellmand-Ford, Breadth-first search, etc. For the particular case of a Mission Graph (a weighted, directed graph, where all weights are positive), Dijkstra's algorithm is suggested [14]. In Appendix C, we present a series of tools that allow to find an optimal path in a Mission Graph. They are based on the Dijkstra's algorithm implementation in Matlab, and rely on the following two major steps in the general case:

1. Locating all the nodes of the Mission Graph that are associated with the target goal, and
2. Finding the shortest path in the Mission Graph that connects the source node with all the required nodes.

Chapter 6

Probabilistic Risk Assessment Framework

6.1 Probabilistic Risk Model

The probabilistic risk assessment methodology that will be used to validate the results obtained in this work is based on the SORA risk model introduced in Chapter 3. The proposed risk model was basically represented in Eq. (3.1) which is showed here again for handiness:

$$P_{harm} = P_{ooc}\ P_{strike/ooc}\ P_{harm/strike} \qquad \text{(3.1 revisited)}$$

As it was already mentioned, this model estimates the probability of causing a harm to an EoV (being it a fatal injury to third parties on the ground, to third parties in the air, or damage to a critical infrastructure) as a function of three factors: the probability of being out of control (P_{ooc}), the conditional probability of striking the EoV if the RPAS is out of control ($P_{strike/ooc}$), and the conditional probability of causing the given harm if the strike has actually occurred ($P_{harm/strike}$).

For convenience, the previous model will be rearranged so that it is expressed as a function of the probability of impact rather than the probability of being out of control. In the sequence of events of a RPAS accident, the "impact" event is an intermediate condition between the out of control event and the event of striking an EoV. As an example, imagine that the RPAS is out of control due to the loss of the C2 link. In this condition, the RPAS may impact on the ground; and if this occurs, then it might strike third parties on the ground. In the same manner, once out of control, an impact between a RPAS and a transient aircraft may also occur; if this happens, then it would strike third parties in the air if the transient aircraft is a manned aircraft. Having this in mind, the probability of impact P_{impact} can be expressed as a function of the probability of being out of control and the conditional probability of having an impact given the out of control condition $P_{impact/ooc}$ as follows:

$$P_{impact} = P_{ooc}\ P_{impact/ooc} \tag{6.1}$$

In the same manner, the probability of strike can be redefined as follows:

$$P_{strike/ooc} = P_{impact/ooc}\ P_{strike/impact} \tag{6.2}$$

As a result, the initial risk model can be also expressed as:

$$P_{harm} = P_{impact}\ P_{strike/impact}\ P_{harm/strike} \tag{6.3}$$

However, the Target Level of Safety (TLS) used to express the safety objective in a risk assessment is usually expressed not as a probability but as an expected number of occurrences per flight hour. Therefore, Eq. (6.3) can be rewritten in terms of rate of occurrence as follows:

$$\lambda_{harm} = \lambda_{impact}\ P_{strike/impact}\ P_{harm/strike} \tag{6.4}$$

Where λ_{harm} is the rate at which the harm under analysis occurs (per flight hour), and λ_{impact} is the rate at which the impact event is expected to occur (also per flight hour). In general, Eq. (6.4) expresses an instant risk as the different terms involved in this equation can vary along space and time. For example, the probability of striking a person depends on the population density in the vicinity of the RPAS position; in addition, the population density in some given area may also vary depending on whether the operation takes places

during the day, or at night hours, for instance. Therefore, if the RPAS is expected to follow a planned trajectory $r = r(t)$, $t \in [a, b]$, $a < b$, where $r(t)$ is a curve C between two points $r(a)$ and $r(b)$, then the instant risk at any point of the flight path is expressed as follows:

$$\lambda_{harm}(r) = \lambda_{impact}(r)\ P_{strike/impact}(r)\ P_{harm/strike}(r) \tag{6.5}$$

The aim of this work is to assess the risk posed by a RPAS flying a given trajectory, as well as to compare the risk of different trajectories. In order to compute the overall risk along a defined flight path, it is necessary to perform the line integral of Eq.(6.5) along the curve C between $r(a)$ and $r(b)$:

$$\Lambda_{harm} = \oint_C \lambda_{harm}(r)\ ds = \int_a^b \lambda_{harm}\left(r(t)\right)\ \|r'(t)\|\ dt \tag{6.6}$$

Where ds is an elementary arc length and $\lambda_{harm}(r)$ can be viewed as a scalar field $\lambda_{harm} : \mathbb{R}^n \to \mathbb{R}$ containing C. Note that Eq. (6.6) is expressed in terms of occurrences per hour of operation along a specified distance (i.e. $[\mathrm{s}^{-1} \cdot \mathrm{m}]$ in SI). Then, the average risk along this trajectory in terms of occurrences per flight hour is given by:

$$\bar{\lambda}_{harm} = \frac{\Lambda_{harm}}{L(C)} \tag{6.7}$$

Where $L(C) = \oint_C ds$ is the length of the curve C between $r(a)$ and $r(b)$ (i.e. the length of the planned trajectory). Next, Eq. (6.6) will be particularized to assess the risk of causing fatal injuries to third parties on the ground (hereinafter *ground risk*), and to third parties in the air (hereinafter *air risk*). Due to practical reasons, the third harm category identified by SORA (causing a damage to critical infrastructures) will not be assessed in this work.

6.2 Ground risk model

In order to derive the ground risk model from Eq. (6.6) (denoted as Λ_G), it is necessary to develop an impact model (term λ_{impact} in Eq. (6.5)), a strike model (term $P_{strike/impact}$), and a harm model ($P_{harm/strike}$). The proposed models for these terms are based on widely accepted models in the literature.

A review on some of these models can be found in [168]; while the approach followed in this work is discussed next.

6.2.1 Impact model

In the case of the ground risk model, the impact model provides the rate at which a ground impact occurs (λ_{impact}). In the literature, this term is often assumed to be constant and is either estimated based on historical accident data, component failure data, and expert judgement [27, 110], or deduced from the TLS [22, 73, 171]. However, there is limited data on UAS accidents, so models are often subject to a high degree of uncertainty [168]. Moreover, the aim of this work is to model the contribution of the different out of control conditions in Sec. 3.2.2 to the probability of occurrence of a ground impact. For example, the impact model should quantify to what extent the loss of the C2 link increases the likelihood of experiencing a ground impact. For this reason, the use of a system/functional approach to determine the mishap rate might be more suitable in this case. Possible modeling approaches include Fault Tree Analysis (FTA) and Event Tree Analysis (ETA), among others. In recent years, Bayesian Belief Networks (BBNs) are also adopted to this effect [10, 16, 168], providing a number of advantages.

Bayesian Belief Networks

A Bayesian Belief Network (BBN) is a probabilistic graphical model described by a directed acyclic graph. In this graph, nodes represent variables and edges represent the conditional dependencies between these variables. Each node variable is associated with a Bayesian probability which describes the state of knowledge of this variable. The probability of a node depends on the node's parent probabilities, and is expressed with a Conditional Probability Table (CPT). A CPT specifies the conditional probability of a variable for all the possible combinations of the parent nodes' states. Then, the probability calculation and propagation is performed using the Bayes' theorem.

One of the advantages of BBNs is that CPTs can be expressed with both qualitative and quantitative data simultaneously [16]. This is specially useful in models with high uncertainty as in the problem under study. Another advantage of the Bayesian approach is that it can be used to perform probabilistic inference. Therefore, an initial assumption regarding one node can be replaced by a perceived *evidence* regarding this node and then, the model automatically updates the probabilities of all its child nodes based on the presence of such

evidence [10]. In practice, this capability could be used to update the probability of a node given the real-time state of the system. For example, the probability of a ground impact can be updated depending on whether the C2 link is loss or alive, for instance.

Ground impact BBN model

In view of the previous advantages, this work advocates the use of the BBN approach to develop the ground impact model λ_{impact}. The proposed BBN model is represented in Fig. 6.1. As it can be observed, the sink node represents the probability of a ground impact P_{impact}, and the remaining nodes describe the sequence of events between the initiating factors and the expected outcome. In this case, the proposed sequence of events is based on the contingency event description of Sec. 3.2.2, and is discussed next in a bottom-up manner.

The "ground impact" node probability depends on the combined likelihood of experiencing the "loss of control" condition and the "boundary violation" condition, two of the contingency events considered in this work. Both contingencies can be caused by an "inappropriate guidance", e.g. a guidance command that is not suitable for the current state of the aircraft (because it exceeds the flight envelope limits, because it is not consistent with the approved Mission Plan, etc). In addition, the "boundary violation" can also result from a "navigation error" like the loss of performance of the GNSS (other of the proposed contingencies). The "inappropriate guidance" event probability is based on the combined effect of an "autopilot malfunction" (including loss of function and malfunction) and "pilot ineffectiveness". The human pilot is considered to be "ineffective" when she or he takes a wrong guidance decision, or when a correct decision is badly executed (e.g. selection of an inappropriate control mode, poor piloting skills, etc). The origin of an "autopilot malfunction" or a "pilot ineffectiveness" condition may be the use of incorrect navigation information caused by a "navigation error". Finally, the pilot may also be "ineffective" when she or he is not in the control loop due to the "C2 link loss".

An important remark regarding the previous model is that it outputs the probability of the occurrence of the ground impact event P_{impact}, not the failure rate λ_{impact}. In order to derive λ_{impact} from P_{impact}, it is necessary to assume a given probability distribution function. As in similar approaches in the literature [16, 115], this work assumes that the ground impact event follows a Poisson distribution: a discrete probability distribution commonly used to model rare events which occur at a constant rate, independently of the time

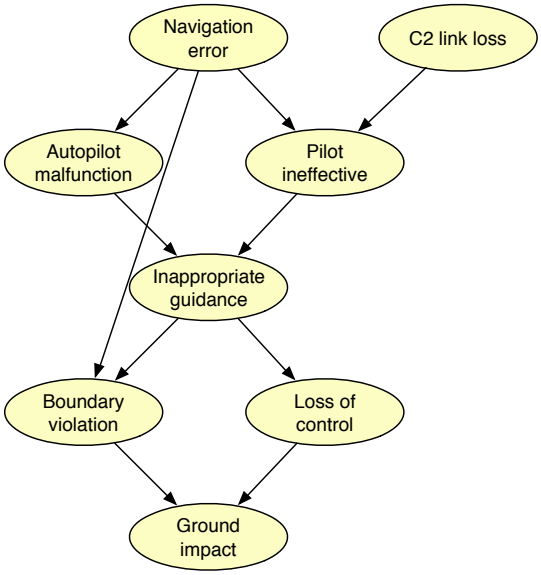


Figure 6.1: Ground impact BBN model.

since the last event. The probability mass function of a Poisson distribution is expressed as follows:

$$p(x;\lambda;t) = \frac{e^{-\lambda t}\left(\lambda t\right)^{x}}{x!} \quad \text{for } x = 0, 1, 2, \ldots \tag{6.8}$$

Where x is the number of times an event occurs in a time interval t, and λ is the expected time rate for the events to happen (also called shape parameter). If we consider that λ is the ground impact rate expressed in terms of occurrences per hour of operation, and we consider the timeframe to be one hour, then the probability of impact is given by:

$$P_{impact} = 1 - p(0;\lambda_{impact};1) = 1 - e^{-\lambda_{impact}} \tag{6.9}$$

Therefore:

$$\lambda_{impact} = -\ln\left(1 - P_{impact}\right) \tag{6.10}$$

6.2.2 Strike model

The strike model represents the conditional probability that an impact at a specific location strikes a person. In the literature, this factor is commonly modeled as follows [10, 22, 27, 73, 110, 151]:

$$P_{strike/impact}(r) = \rho_G(r)\ LA \tag{6.11}$$

Where $\rho_G(r)$ is the population density at the impact point, and LA is the lethal area of the airborne platform. On one hand, the population density depends on the geographical area where the operation takes place. Census data are often used to estimate this value [10, 27, 109, 171]. As it was mentioned before, the population density is to some extent a dynamic term: daily, weekly, and seasonal changes may occur. A review of dynamic population models can be found in [168]. However, this work assumes that the population distribution remains static for simplicity.

On the other hand, the lethal area represents the area where pedestrians may be struck after the RPAS crashes into the ground. This area basically depends on the RPAS model and the pedestrian model (mainly on their sizes), but also on the nature of the accident (or crash mode). Two crash modes are often considered [50]: vertical free fall and unpremeditated, gliding descent. The vertical free fall mode is usually associated with loss of control conditions where the aircraft crashes at high velocity. The lethal area for this mode is commonly modeled as follows [27, 109, 151]:

$$LA = \pi\ \left(\frac{max(w_{ua}, L_{ua})}{2} + R_p\right)^2 \tag{6.12}$$

Where w_{ua} is the UA wingspan, L_{ua} is the UA length; and R_p is the radius of an average person. By contrast, the unpremeditated descent mode assumes that the RPAS cannot maintain flight and glides to the ground at maximum lift-to-drag ratio. The crash area for this mode is modeled as follows [10, 27, 73, 110]:

$$LA = (w_{ua} + 2R_p)\ \left(L_{ua} + \frac{H_p}{\tan(\gamma)} + 2R_p\right) \tag{6.13}$$

Where H_p is the height of an average person, and γ is the glide angle. Therefore, LA is a constant parameter for any crash mode. Based on the ground

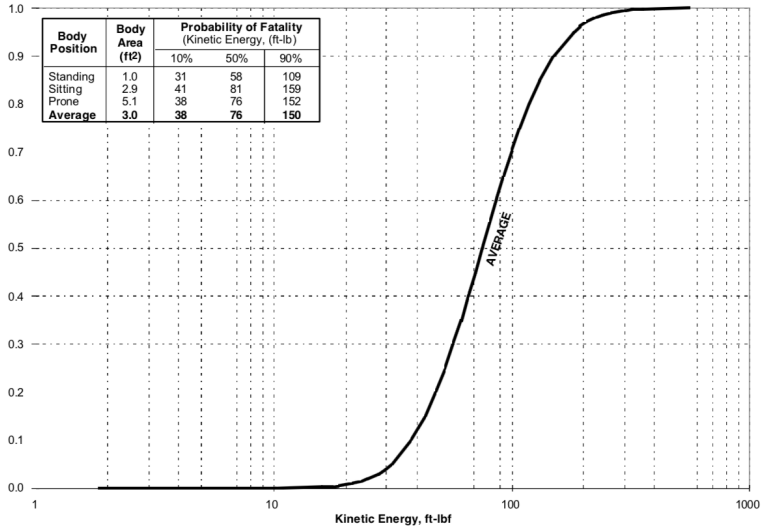


Figure 6.2: Probability of fatality from kinetic energy impact. Source: [135]

impact model in Fig. 6.1, it could be possible to assign the vertical free fall mode to the loss of control condition, and the unpremeditated descent mode to the boundary violation condition. However, for simplicity, this work assumes that both ground impact modes result in a vertical free fall impact so that the impact location is close to the point where the initiating failure has occurred.

6.2.3 Harm model

The harm caused to a person after a strike depends on multiple factors, including type and design of the RPA (e.g. size, fragility, exposure to rotating parts), conditions at the point of impact (e.g. speed, position), or secondary effects like explosions, etc. [168]. One of the widely accepted approaches in the literature models the possible harm as a function of the transfer of kinetic energy on impact [12, 22, 73, 135], see Fig. 6.2. However, it is reasonable to expect that any direct impact of the RPAS models considered in this work (i.e. mid-size RPAs operating above 500 ft) will clearly cause the instant death of the people involved in the accident. Therefore:

$$P_{casualty/strike}(r) = 1 \tag{6.14}$$

Another question is whether people are directly exposed to the harm or if there exists some kind of sheltering that is able to absorb the impact energy to some extent. Examples of sheltering structures could be buildings, cars, trees, etc. Sheltering is often modeled using a sheltering factor S_F so that [22, 73, 110, 171]:

$$P_{casualty/strike}(r) = 1 - S_F(r) \tag{6.15}$$

$S_F(r)$ is often higher in high density areas where city structures are considered to reduce the exposure of people to the harm. However, for simplicity this work assumes the worst case scenario where no shelter exists, so $S_F = 0$.

6.2.4 Data source

Based on the previous discussion, the ground risk model results as follows:

$$\Lambda_G = LA \int_a^b \lambda_{impact}\left(r(t)\right)\ \rho_G\left(r(t)\right)\ \|r'(t)\|\ dt \tag{6.16}$$

Next step is to populate the previous model with the corresponding data. To start with, the lethal area LA requires to specify the RPA dimensions and the average person model. In this case, the IAI Super Heron model will be used as an example, whose performance data can be found in Table 3.6; while the average person is usually modeled as a cylinder of height $H_p = 1,75$ m and radius $R_p = 0,25$ m. With respect to the ground impact event rate λ_{impact}, it is necessary to specify the CPTs for all the event nodes in Fig. 6.1. As previously stated, this requires to collect historical failure data as well as data from experts in that subject. In addition, it could be possible to model λ_{impact} as a function of the reference trajectory $r(t)$. For example, some operational areas may have a higher probability of experiencing C2 link losses or navigation errors due to the surrounding terrain. This way, it would be necessary to specify the CPTs for all the different regions. However, for simplicity this work assumes that λ_{impact} is independent of the aircraft trajectory. The preliminary CPTs used in this work are gathered in Appendix E.1. Finally, the population distribution ρ_G is usually defined using the census data at a given region. In this case, we have accessed the Spanish census data from *Instituto Nacional de Estadística* (Spanish Statistics Institute) (INE)[1] and we have processed it using the ArcGis

[1] Available online at https://www.ine.es/censos2011_datos/cen11_datos_inicio.htm (last accessed on October 2018).

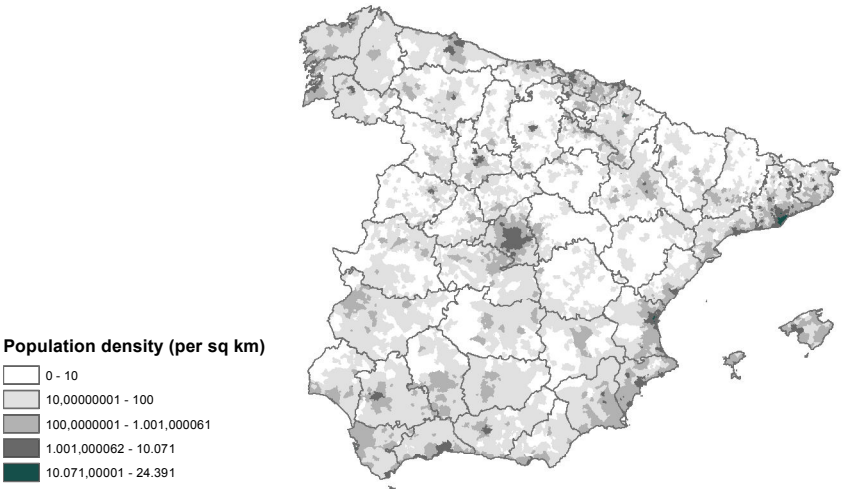


Figure 6.3: Population density in Spain (excluding Canary Islands) based on census data from INE.

software. The resulting data has been converted to a raster image where each pixel represents a cell of size 1×1 km, see Fig. 6.3.

The fact that the census data is provided as a gridded data has an important implication on the integration of Eq. (6.16). Due to this fact, the instant ground risk can just be evaluated at each of the cells of the grid, not along the continuous route $r(t)$. Therefore, in order to perform the line integral along $r(t)$, it is necessary to discretize the model as follows:

$$\Lambda_G = LA \sum_{k=1}^{N} \mu_k \; \lambda_{impact_k} \; \rho_{G_k} \; V_k \; \Delta t_k \tag{6.17}$$

Where N is the number of cells in the grid; λ_{impact_k}, ρ_{G_k} and V_k are the ground impact event rate, the population density, and the aircraft velocity in cell k, respectively (assuming they remain constant within a cell); Δt_k is the time interval within cell k; and μ_k is the membership function, defined as follows:

$$\mu_k = \begin{cases} 1 & \text{route crosses cell } k \\ 0 & \text{otherwise} \end{cases} \tag{6.18}$$

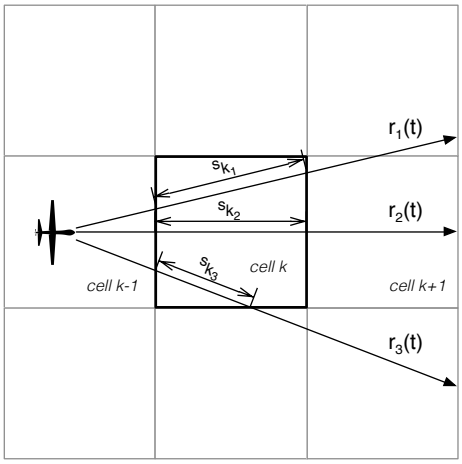


Figure 6.4: Distance travelled within a grid cell for different aircraft trajectories. Source: Drone icon by Anthony Lui from the Noun Project

Note that the time spent within each cell Δt_k depends on how the flight path crosses the cell and on the aircraft velocity, see Fig. 6.4. In order to simplify the analysis, this work assumes that the distance travelled within each cell s_k is the same in all cells visited by the route, e.g. $\bar{s} = 1$ km. This way, the time spent within a cell is simply $\Delta t_k = \bar{s}/V_k$, and the discrete model results as follows:

$$\Lambda_G = LA \sum_{k=1}^{N} \mu_k \ \lambda_{impact_k} \ \rho_{G_k} \ \bar{s} \tag{6.19}$$

Finally, note also that, in order to compute the average ground risk along a specified route, the route length $L(C)$ in Eq. (6.7) must also be approximated in accordance with the previous assumption:

$$L(C) = \sum_{k=1}^{N} \mu_k \ \bar{s} \tag{6.20}$$

6.2.5 Model limitations

Although the proposed model is considered to be a comprehensive approach to estimate the risk entailed to third parties in the vicinity of a UAS operation, it is important to account for the following model weaknesses:

- In order to compute the average risk along a given flight segment, it is necessary to perform the line integral in Eq. (6.17). This requires prior knowledge of the expected flight trajectory. However, when the intended route is described using EPTs, the actual flight path for some legs cannot be easily anticipated in all cases:
  - In EPTs starting and/or ending at a variable waypoint (e.g. CA, FA, VA, etc), the actual flight path depends on the aircraft performance or on external parameters like the wind condition. In these cases, a trajectory planning function should be used to estimate the resulting flight path for a given aircraft model and flight condition.
  - In EPTs with manual termination condition, the actual flight path depends on an external event that cannot be anticipated pre-flight. For this reason, it is not possible to perform the line integral in Eq. (6.17), even with trajectory planning functions, so a different risk estimation approach should be used in this case. Sec. 6.4 provides an alternative risk model that overcomes this issue.
- With respect to the modeling approach, the proposed model assumes that, once the initiating failure occurs, the resulting ground impact location will be in the vicinity of the current RPAS position because of the vertical free fall crash mode assumption. In particular, the model considers that the impact point will be located in the same raster cell than the one where the failure occurred. A more detailed model would have to take into account the uncertainty in the impact location for the different crash modes (i.e. vertical free fall or gliding descent). Therefore, an impact trajectory model defining the impact location distribution should be developed.
- Finally, the accuracy of the model strongly depends on the uncertainty of the input data. In particular, data related with the reliability of the UAS, or with sub-system components, is often difficult to collect or is inaccurate due to the fact that UASs are still an emerging technology. Population density data is also a sensitive parameter. In this case, the Spanish census data is considered an accurate source data. However,

when it comes to model sheltering structures, more detailed models or gross assumptions would have to be taken into consideration.

6.3 Air risk model

As in the case of the ground risk, deriving the air risk model Λ_A from Eq. (6.6) requires to develop an impact model (term λ_{impact} in Eq. (6.5)), a strike model (term $P_{strike/impact}$), and a harm model ($P_{harm/strike}$). The proposed approach to develop these terms is discussed next.

6.3.1 Impact model

When it comes to the air risk model, λ_{impact} refers to the rate of occurrence of a MAC between two aircraft. In general, the probability of a MAC depends on multiple factors related with the operational scenario, e.g. airspace structure and flight rules, traffic density in the area, etc. In literature, most of the approaches for modeling the rate of occurrence of this event are based on the Maxwell molecule formulation for predicting the collision frequency of gas molecules [11, 109, 115]. Based on this theory, the rate at which a MAC may occur is proportional to the traffic density. However, this theory assumes that the air traffic behaves randomly in airspace, omitting airspace rules and structure. For this reason, this model does not adequately represent traffics operating in controlled airspace. A refinement of this model is provided in [171], where the same theory is applied but taking into consideration aircraft density variations on airways and on flight levels. Finally, two specific air risk models for operations in controlled airspace can be found in [53, 90].

In any case, none of the previous approaches is able to capture the contribution of the different out of control conditions considered in this work to the probability of occurrence of a MAC accident. For this reason, this work advocates for developing a specific BBN model to determine the MAC rate λ_{impact} from the analysis of the undesirable event sequence like in the ground impact risk. In this case, the sequence of events should model the conflict management layers identified in Sec. 3.2.2, mainly separation assurance and collision avoidance. Considering that the implementation and effectiveness of these layers vary from controlled to uncontrolled airspace, we propose to develop two separated impact models: one for controlled airspace and other for uncontrolled airspace, which are described next.

Mid-air collision BBN model in controlled airspace

The proposed mid-air collision BBN model for flight segments performed in controlled airspace is represented in Fig. 6.5. The output node of this model is the "MAC" node which has an associated probability P_{impact}. According to the graph in this figure, the sequence of events leading to this flight condition depends on the "separation error" and on the "collision avoidance error". The "separation error" in controlled airspace is considered to occur when both "strategic separation" and "tactical separation" fail. "Strategic separation error" basically refers to the failure of the procedural separation mechanism, while "tactical separation error" involves the ATC surveillance capability. The "tactical separation error" node probability depends on the combined likelihood of the corresponding ATC unit being "ineffective" and the pilot in command of the RPAS performing an "inappropriate guidance". ATC is ineffective when a possible conflict is not detected, or when ATC provides an incorrect clearance. This node probability certainly depends on the "traffic density"[2] in the area. In the air risk model for controlled airspace, "inappropriate guidance" refers to conditions where the ATC clearance is not correctly executed by pilot in command of the RPAS (either the automatic/autonomous system or the remote pilot). In this case, the probability of experiencing an "inappropriate guidance" depends on the same sequence of events than in the ground impact BBN model described in Sec. 6.2.1.

Once the "separation error" occurs, collision avoidance layers can still prevent the MAC from occurring. In controlled airspace, it is assumed that aircraft will be equipped with a transponder. Therefore, collision avoidance can be performed at two levels with a different time horizon. At a first level, TCAS can trigger a traffic alert/resolution advisory. This alert is considered the main contingency management mechanism for preventing MACs when operating in controlled airspace, and is usually triggered with a time horizon of minutes before the point of closest approach. The effectiveness of this layer depends on the remote pilot because it is assumed that she or he must still approve or reject the resolution advisory. If the TCAS alert results "ineffective", then the NMAC condition will occur.

After this happens, a second collision avoidance mechanism can still reduce the probability of a MAC impact by performing an evasion maneuver seconds after the point of closest approach. This maneuver may be either a SAA-based

[2]Note that, in Fig. 6.5, the "traffic density" node has a rectangular shape instead of an ellipse. This notation emphasizes that this node is not a probabilistic node, but a decision node, i.e. a node representing an input variable of the model. In other words, the traffic density is considered to be known at a given airspace volume.

Figure 6.5: Mid-air collision BBN model in controlled airspace.

maneuver performed by the remote pilot, or a DAA-based maneuver performed by the automatic system (if a DAA system is equipped onboard the RPAS). At this time, it is not possible to assume that the RPAS will be equipped with such technology since state-of-the-art DAA systems are still unable to comply with the Minimum Operational Performance Standards (MOPS) required by SORA. In any case, a "DAA error" may occur if the on-board sensors are unable to detect the conflicting traffic. SAA may be "ineffective" when the remote pilot has a reduced situational awareness, or when the pilot is not in the control loop due to the "C2 link loss".

In summary, the MAC condition is expected to occur after a NMAC if all the collision avoidance mechanisms have resulted ineffective. As in the ground impact model, this work assumes that the MAC event follows a Poisson distribution so λ_{impact} can be deduced from P_{impact} using Eq. (6.10).

Mid-air collision BBN model in uncontrolled airspace

The proposed mid-air collision BBN model for flight segments performed in uncontrolled airspace is represented in Fig. 6.6. As in the BBN model for controlled airspace, the output node is the "MAC" node which has an associated probability P_{impact}. In this case, Eq. (6.10) also applies. However, as it can be observed in the figure, the sequence of events leading to this flight condition differs when flying in uncontrolled airspace. To start with, separation provision is independent of the ATC service. In this case, the main separation mechanism is the definition of the mission boundaries and the use of geofencing to enforce these boundaries. However, as it was discussed in Sections 3.2.2 and 6.2.1, "boundary violation" may occur due to "inappropriate guidance" or because of a "navigation error". Then, once the "boundary violation" occurs, the likelihood of experiencing a "separation error" increases with the "traffic density" in the area.

Even if the RPAS flies within the specified boundaries, other traffics may also be encountered in the same operational volume. This may be either because the operational area is not a segregated area or due to the fact that other aircraft are involved in the same RPAS mission (e.g. a firefighting mission). For this reason, the remote pilot is required to "remain well clear" of other aircraft at all times. However, the remote pilot may fail at remaining well clear because she or he performs an "inappropriate guidance". In addition, the model assumes that the likelihood of the remote pilot failing at remaining well clear increases with the "traffic density" because of the increased pilot workload.

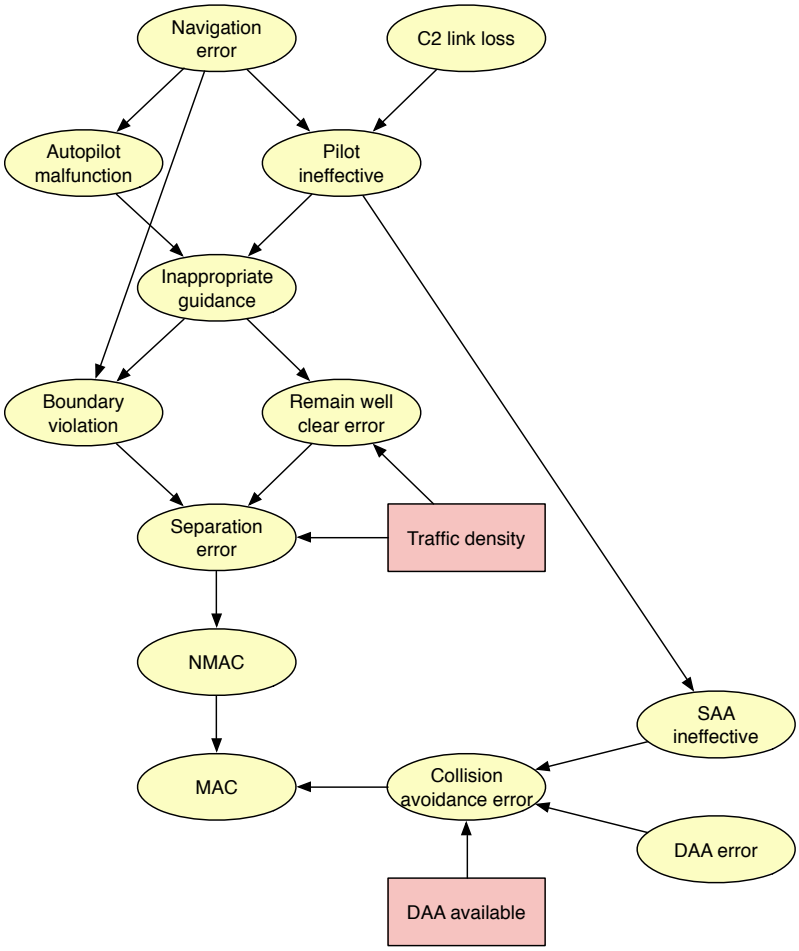


Figure 6.6: Mid-air collision BBN model for uncontrolled airspace.

The other key difference when operating in uncontrolled airspace is that aircraft are not required to be equipped with a transponder. Therefore, it is not possible to assume that an intruder aircraft will be a cooperative traffic, what makes the TCAS layer inoperative. As a result, after a "separation error" occurs, the "NMAC" condition is assumed to happen, and the only feasible collision avoidance mechanism is the SAA or DAA maneuver. This is one of the factors that certainly increases the operational risk when flying in uncontrolled airspace.

6.3.2 Strike model

The strike model represents the conditional probability that an impact between two aircraft strikes a person in the air. In the case of a RPAS operation, an impact is expected to cause a strike only if the transient aircraft is a manned aircraft. Therefore, the strike model should account for the ratio between manned and unmanned aircraft in the vicinity of the RPAS operating area. For simplicity, this work assumes that all mid-air collisions involve a manned aircraft as long as the RPAS is not performing a formation flight with other RPAs. This way, all impacts are supposed to result in a strike, so the conditional probability is expressed as follows:

$$P_{strike/impact} = \rho_A(r) \tag{6.21}$$

Where $\rho_A(r)$ is the number of people onboard a given aircraft. In order to estimate this term, it is necessary to characterize the aircraft flying in the airspace volume where the operation takes place. For example, it is possible to assume that most aircraft flying a controlled airway will be airliners, while most aircraft flying in uncontrolled airspace will be general aviation aircraft.

6.3.3 Harm model

The harm model determines the likelihood of causing fatal injuries to people on board the collided aircraft once the strike between the RPAS and the manned aircraft has occurred. As in the case of the ground risk model, the possible outcomes depend on the type of RPA, but also on the strike geometry, or the impact energy, among others. In this case, the operational scenarios considered in this work (i.e. mid-size RPAs operating at relatively high speeds) presumes that, if struck, the other aircraft cannot continue a safe flight and landing. Therefore, all strikes are supposed to result in a casualty:

$$P_{casualty/strike} = 1 \tag{6.22}$$

6.3.4 Data source

Based on the previous discussion, the proposed air risk model results as follows:

$$\Lambda_A = \int_a^b \lambda_{impact}\left(r(t)\right)\ \rho_A\left(r(t)\right)\ \|r'(t)\|\ dt \tag{6.23}$$

Next step is to populate the previous model with the corresponding data. As it can be observed, the two model parameters in Eq. (6.23) are λ_{impact} and ρ_A. According to the proposed impact model, λ_{impact} varies along the aircraft trajectory $r(t)$ as a function of the airspace class where the operation takes place (basically on whether it is controlled or not) and the aircraft density in each operational volume. In this case, the airspace class is an evidence for this model since it is implicit in the route specification. Regarding the traffic density, it is necessary to account for its spacial and temporal variation along the different flight segments. For example, the majority of traffic is expected to be concentrated in airways and in the vicinity of airports; in addition, the lower flight levels of an airway are usually less congested than the upper levels; similarly, the traffic density is normally higher at daytime than during the night. The problem that arises is how to collect and represent this data.

In the United States, there exists an extensive flight data collection of cooperative traffics that can be used to estimate the average traffic density at different regions and at altitudes; see Fig. 6.7 as an example. However, similar data has not been collected in Europe, or it has not been released to the general public. To date, the most representative data collection is supplied by the Network Strategic Modelling Tool (NEST) software by Eurocontrol, a simulation tool for network capacity planning and airspace design [56]. This tool provides a default dataset comprising 31.626 real cooperative flights operated in Europe during AIRAC cycle 1307 (effective 27 June 2013). Although this dataset does not include traffics flying in uncontrolled airspace, we will use these data as a basis to estimate the required traffic density values, as explained next.

The goal is to obtain a traffic density estimation for each operational volume of a given route $r(t)$. To do so, the route will be divided in the different flight legs that it is composed of. Then, for each flight leg, the traffic density will be estimated as a function of the number of flights crossing its associated termination point using the NEST dataset, following the next steps:

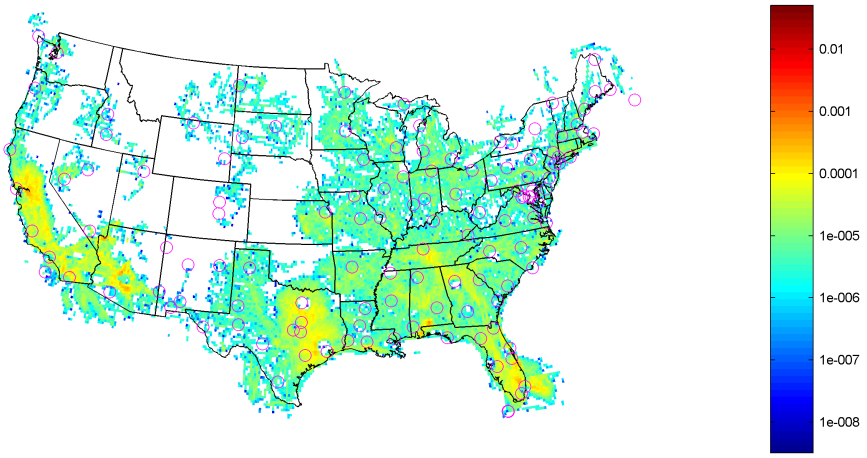


Figure 6.7: Average traffic density between 5000 ft AGL and 10.000 ft AMSL in aircraft per NM^3 for the periods 1–7 December 2007 and 1–7 June 2008 in the United States. Source: [102]

1. Collect the number of traffics flying over a given waypoint. For example, based on the available data, 340 flights flew over waypoint SOPET on 18 July 2013, see Fig. 6.8.

2. Average the number of flights flying over a given waypoint along the entire AIRAC cycle. For example, the average number of flights that flew over waypoint SOPET during cycle 1307 was 321,75.

3. Compute the average rate at which traffics fly over a given waypoint per day, assuming all traffics fly at daytime (i.e. 16 hours per day). In this example, the average rate at which traffics fly over waypoint SOPET is 20,11 aircraft/hour.

4. Assign a qualitative traffic rate category. In this work, we propose to assign the corresponding category according to the defined intervals in Table 6.1. In this case, waypoint SOPET is considered to have a "high" traffic rate.

5. Assume that the traffic rate category at a waypoint is representative of the traffic density in the vicinity of this waypoint. Therefore, it is expected to encounter a "high" traffic density in the vicinity of waypoint SOPET.

Table 6.1: Traffic rate category definition.

Traffic rate category	Traffic rate interval (aircraft per hour)
Low	< 5
Medium	5 to 15
High	> 15

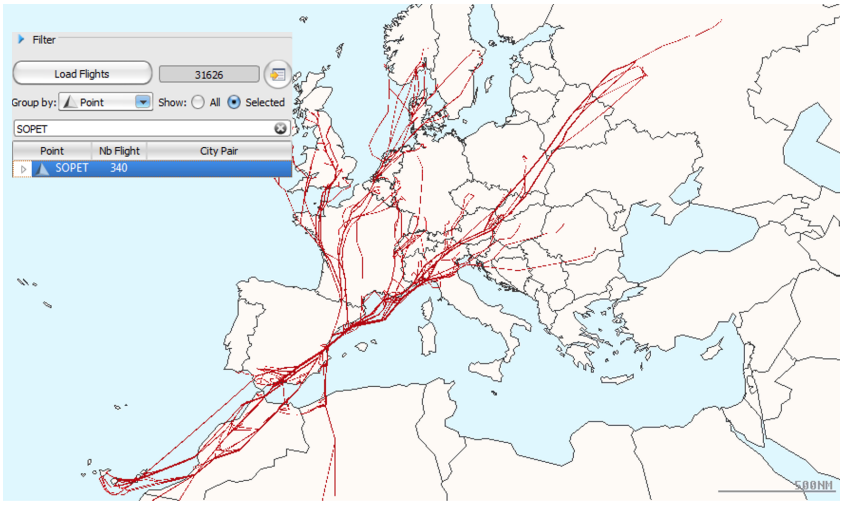


Figure 6.8: NEST screenshot showing traffics flying over waypoint SOPET on 18 July 2013.

6. Repeat for all the waypoints in the specified route. If traffic data for a given route waypoint is not available in the dataset, assume a "medium" traffic rate (density) for this waypoint.

Then, the CPTs for all the event nodes in Figures 6.5 and 6.6 must be specified for all the possible traffic density categories. As in the ground risk model, conditional probabilities in these CPTs should be deduced from historical accident data or from opinions from experts in the subject. In this case, the preliminary CPTs that will be used in this work are presented in Appendices E.2 and E.3.

Finally, the second model parameter in Eq. (6.23) is the number of people onboard the manned aircraft involved in the MAC, ρ_A. For simplicity, this work assumes that the most probable intruder aircraft when flying in controlled airspace is a short-to-medium-range airliner like a Boeing 737 or an

Airbus A320 (two of the world's most successful commercial airliners [154]), with an estimated capacity of $\rho_A = 180$ passengers. By contrast, when flying in uncontrolled airspace, the intruder aircraft is assumed to be a general aviation aircraft like a Cessna 172 or a Piper PA-28 Cherokee, with an estimated capacity of $\rho_A = 4$ passengers.

6.3.5 Model limitations

The proposed air risk model is considered to be a representative approach to model traffic encounters in UAS operations. However, one of the most remarkable drawbacks with this model is to obtain accurate input data, and particularly traffic density data. Therefore, we envision that, in order to exploit air risk model results to a greater extent, it would be necessary to process traffic density data in a more accurate way:

- Traffic density should be estimated for different altitude layers. This way, it could be possible to demonstrate that certain altitude levels are safer than others because they reduce the likelihood of encountering other traffics, for instance.
- The input dataset should also include traffics operating in uncontrolled airspace. Otherwise, a different traffic density model for these flight segments should be conceived. As an example, a traffic density model for uncontrolled airspace could weight the distance to uncontrolled airdromes, among other parameters.
- Traffic density data should be averaged not only over one month but over a broader period of time. This way, input data could capture seasonal traffic changes (e.g. winter/summer).

In addition, the drawback regarding how to perform the line integral in flight legs where the path is not fully defined (Sec. 6.2.5) is also encountered in this case. Due to practical reasons, implementation of the previous aspects is left to next stages of the research.

6.4 Probabilistic Risk Model for nondeterministic paths

As it was introduced before, the flight path of EPTs with manual termination (FM, VM, RM, SM) cannot be anticipated pre-flight. In these cases, the probabilistic risk model in Sec. 6.1 cannot be applied. A possible workaround for this issue is to estimate the risk entailed for third parties not using the expected trajectory but the maximum (worst-case) time of flight allowed for this leg and the aircraft velocity:

$$\Lambda_{harm} = \bar{\lambda}_{harm}\ V\ \Delta t_{max} \tag{6.24}$$

Where $\bar{\lambda}_{harm}$ is the average risk within the geographical area where the aircraft is expected to be contained when flying this leg, assuming that all possible positions have an equal probability; V is the aircraft velocity for this leg; and Δt_{max} is the maximum time limit defined for this leg (one of the definition parameters in Table 5.3).

6.5 Risk-based, cost estimation of Reconfigurable Mission Plan routes

Chapter 5 introduced the concept of Reconfigurable Mission Plans. In summary, a Reconfigurable Mission Plan specifies different mission goals and different routes to achieve these goals. The set of all the possible routes is described by the Mission Graph. In this graph, the edges are weighted, with the weight representing the cost of flying between two consecutive nodes. This allows to optimize the route configuration process. As it was introduced in Sec. 5.1.4, this work advocates for defining the cost of flying a route in terms of the risk entailed to third parties (either on ground or onboard other aircraft). Based on this idea, the cost of an edge of the Mission Graph can be computed using the ground risk model and the air risk model as follows:

$$c_k = \Lambda_{G_k} + \Lambda_{A_k} \tag{6.25}$$

Where c_k is the cost assigned to an edge k of the Mission Graph, and Λ_{G_k} and Λ_{A_k} are the risk entailed to third parties on the ground and in the air, respectively, when flying this edge. Therefore, Λ_{G_k} and Λ_{A_k} will be computed using Equations (6.17) and (6.23) in the general case, or Eq. (6.24) if the expected path is not predictable; and c_k is the total risk of this leg expressed

in terms of expected number of fatal injuries per hour of operation along a nautical mile (i.e. $\left[\mathrm{h}^{-1} \cdot \mathrm{NM}\right]$). Then, the cost of flying a route r_i is simply the cumulative cost of all the edges of crossed by r_i:

$$\bar{c}(r_i) = \sum_{k=1}^{N} c_k \tag{6.26}$$

Where N is the number of edges crossed by the specified route. Finally, the average cost of this route is given by:

$$\bar{\lambda} = \frac{\bar{c}(r_i)}{L(C)} \tag{6.27}$$

Where $\bar{\lambda}$ is expressed in terms of expected number of fatal injuries per hour of operation (and thus can be compared with the TLS required by regulation). These expressions will be used in Chapter 7 to calculate the mission risk assessment of a demonstration mission.

Chapter 7

Validation results

7.1 Introduction

This chapter presents the validation results of the prototype application developed in this work. The validation results aim at demonstrating the following two major research questions:

- Demonstrating that the proposed system allows to keep an acceptable level of safety, even if the RPAS operates in an autonomous or degraded condition.
- Demonstrating that the proposed system is able to reduce the probability of performing the flight termination action after a contingency happens.

By contrast, it is important to note that this validation does not refer to the V&V stage of the software development process which is prescribed by most methodologies like the MBD.

Therefore, validation results are organized as follows. Firstly, Sec. 7.2 illustrates the process of designing and specifying Reconfigurable Mission Plans with a realistic RPAS mission example. This section also discusses the mission risk assessment for this case study considering six different operational conditions. Then, based on these results, we will measure the effectiveness of the

proposed contingency management policy in mitigating the risk after a contingency happens. Afterwards, Sec. 7.3 simulates the execution of the proposed demonstration mission in the simulation environment. In particular, we will first simulate the execution of the proposed mission performed in a nominal condition; and then we will simulate two contingency scenarios to illustrate the on-board contingency management capability.

7.2 Mission definition and risk assessment

The proposed demonstration mission describes a representative RPAS operation in which the RPAS has to perform some direct observations over the Albufera natural park in Spain. In this case study, the operations area is defined as the boundary of this natural park, which coincides with the protected area F15B of the Spanish Aeronautical Information Service (AIS)[1]. This area is located within the Controlled Traffic Region (CTR) of the València Airport (ICAO code LEVC), so the mission will require special permission from ATS authorities. For any reason, the planned mission departs from the uncontrolled airport of Teruel (LETL) and the mission ends in the controlled airport of Castellón (LECH). Alternative landing sites are LETL and the Requena aerodrome (LERE). There are three no-fly zones in the vicinity of this mission: the nuclear plant LEP138, the military zone LED65, and the Aerodrome Traffic Zone (ATZ) around LEVC. The overall picture is presented in Fig. 7.1. The proposed Reconfigurable Mission Plan will be designed in compliance with the flight charts and airspace information available in the AIS as described next.

7.2.1 Mission specification

Reconfigurable Mission Plans are formally specified as UML object diagrams and, as previously said, we have not addressed the subject of designing a specific language or syntax to formally specify Mission Plans other than UML. This way, the high-level view of the Reconfigurable Mission Plan object diagram for the demonstration mission is shown in Fig. 7.2. As it can be observed, it includes a declaration of 61 waypoints used by other components of the Mission Plan; a declaration of 16 mission goals; a declaration of 23 segments used to build the routes of a mission; a declaration of one static route (the nominal route); and a declaration of 21 mission boundaries. Next, we describe the specification of some of the objects that conform this Reconfigurable Mission Plan object.

[1] Available online at https://ais.enaire.es/aip/ (last accessed on June 2018).

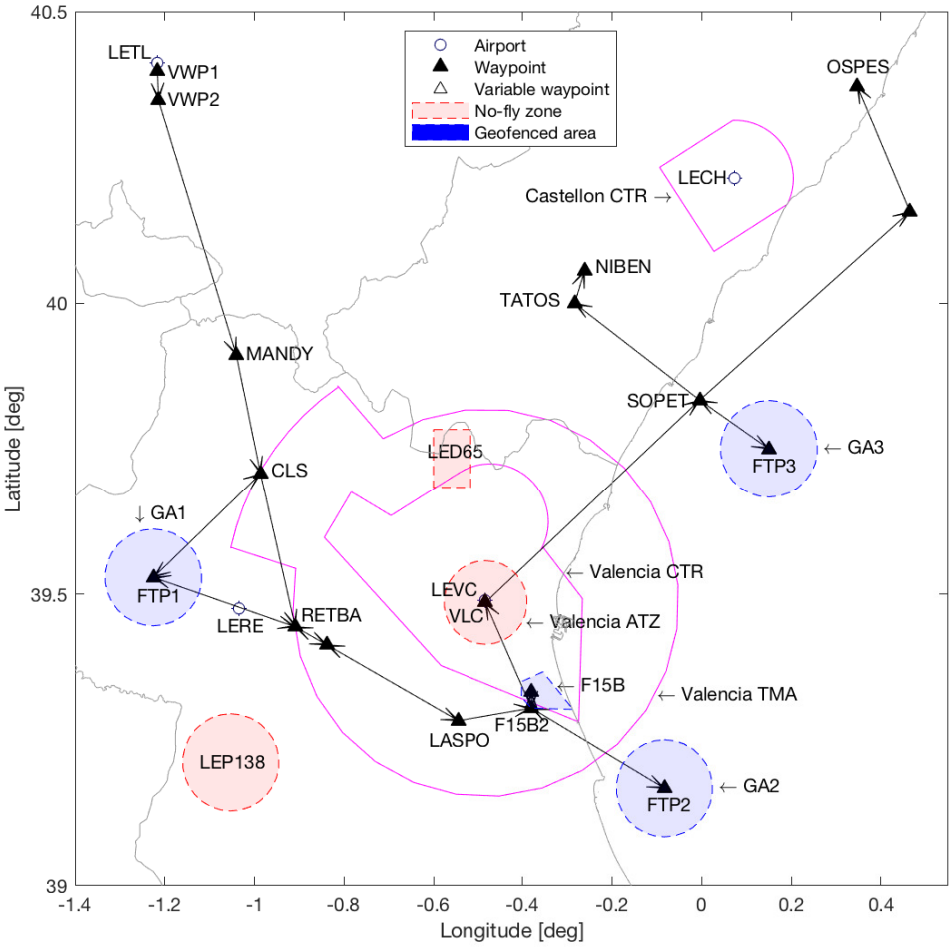


Figure 7.1: Demonstration mission: navigation chart view. Note that some departure and arrival segments have been omitted for clarity.

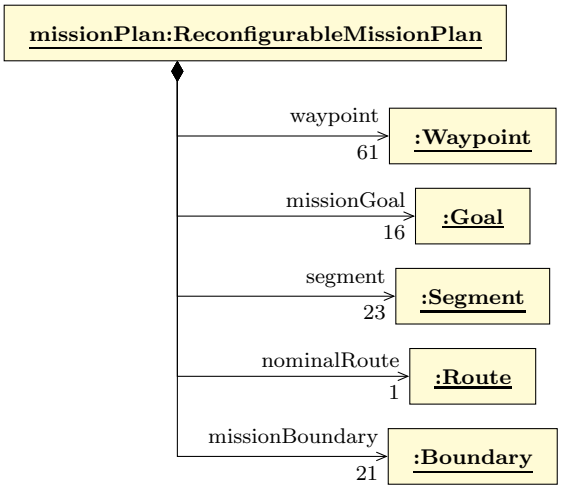


Figure 7.2: Demonstration mission: specification of the Reconfigurable Mission Plan.

To start with, Fig. 7.3 shows the specification of the `boundary` object describing the operations area. As shown, the operations area is geofenced and has a polygonal shape outlined by five waypoints. A lower vertical limit has been specified for this area to comply with the flight restriction area F15B (due to highly sensitive fauna). An upper limit has also been set to avoid conflicts with upper traffic. The entry and the exit point of this area is the same waypoint F15B2. Then, the specification of the remaining `boundary` objects for this mission are specified in a similar manner, so they here are omitted for brevity.

The segments declared in this Mission Plan include segments in controlled and uncontrolled areas. Based on the AIS charts, the nominal route will be composed of the following seven segments, represented in Fig. 7.1:

s_1: Departure segment s_1 describes the departure phase from LETL runway 18 to MANDY, the first waypoint of the en-route phase. The flight path is constructed as a sequence of three legs, coded as IF, CA, and DF (where the IF simply supports the graph definition).

s_2: En-route segment s_2 traverses the lower AIS route R29 from MANDY to RETBA, and then the RNAV airway M871 from RETBA to LASPO. All legs are coded as IF and TF, as required by rule **WFS1**, see Fig. 7.4.

s_3: Ingress segment s_3 connects LASPO with waypoint F15B2, the entry point of the operations area.

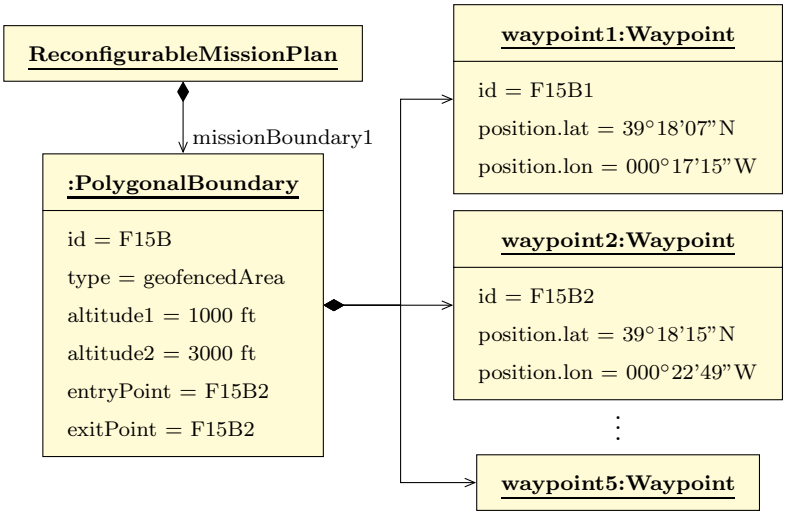


Figure 7.3: Demonstration mission: specification of the operations area.

s_4: Operations segment s_4 is linked to the operations area F15B. The flight path of s_4 is specified using three legs coded as IF, FM, and DF, see Fig. 7.5. One of these legs has a manual termination condition as the mission task has to be performed manually by the remote pilot. The maximum time slot for this leg is 1200 seconds. Afterwards, a DF leg is used to direct the aircraft towards the exit point of this area.

s_5: Egress segment s_5 connects the exit point F15B2 with waypoint VLC. It is flown through a corridor for VFR traffic that the Valencia CTR enables to cross the CTR from North to South.

s_6: En-route segment s_6 flies airway B26 from VLC to SOPET.

s_7: Arrival segment s_7 describes the standard arrival procedure SOPET1S from SOPET to waypoint NIBEN, the IAF for LECH runway 06.

The resulting nominal route is specified as $r_0 = \langle \text{VWP1} \xrightarrow{s_1} \text{MANDY} \xrightarrow{s_2} \text{LASPO} \xrightarrow{s_3} \text{F15B2} \xrightarrow{s_4} \text{F15B2} \xrightarrow{s_5} \text{VLC} \xrightarrow{s_6} \text{SOPET} \xrightarrow{s_7} \text{NIBEN} \rangle$, where "VWP" denotes a variable waypoint. The UML diagram of this route is schematized in Fig. 7.6 as a composition of 7 `routeSegment` objects that are associated with the previous segments. The goal in this figure will be detailed afterwards.

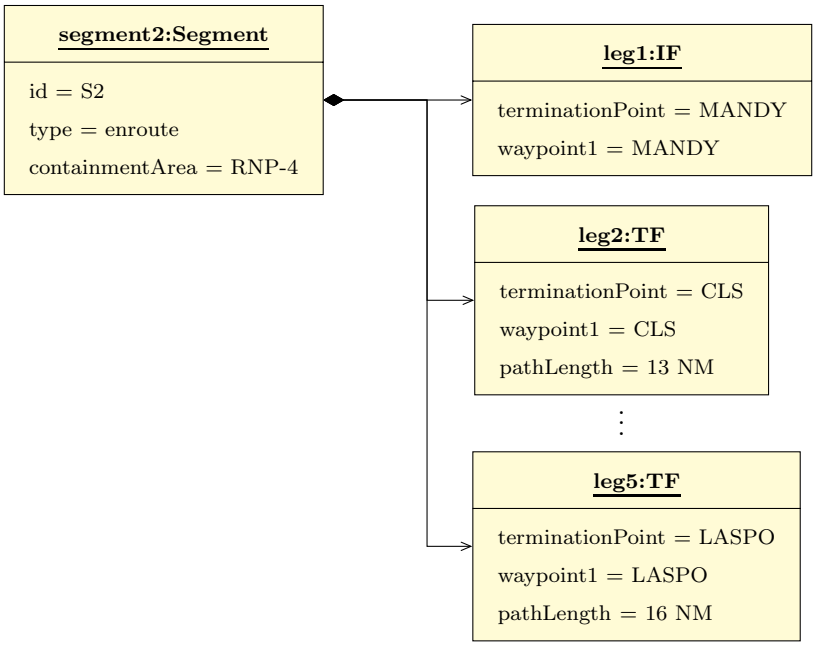


Figure 7.4: Demonstration mission: specification of the en-route segment s_2.

Along with the previous segments, the proposed Reconfigurable Mission Plan also declares additional segments for alternative routings. In short, there is an alternate departure segment s_8 for departing in the opposite runway direction of LETL; however, this segment will not be used in the assumed airport configuration. There is also an arrival segment s_9 towards OSPES, the IAF for the alternate airport configuration of LECH; as well as other arrival segments towards the alternative landing sites ($s_{10} \dots s_{13}$). With respect to the en-route segments, s_{15} is used to connect the operations area with the emergency landing site LERE. This segment goes below airway M871 following a dedicated flight corridor in uncontrolled airspace (because M871 is a single direction airway). The remaining segments ($s_{16} \dots s_{23}$) connect the nominal route with the restricted areas named as GA1, GA2 and GA3 in Fig. 7.3 in both directions. These areas are defined as safe areas where the RPAS must fly either to perform climbs trying to regain the C2 link signal or the Global Positioning System (GPS) signal, or to perform the flight termination action. This way, these areas must be located in unpopulated areas, and must be segregated for safety reasons.

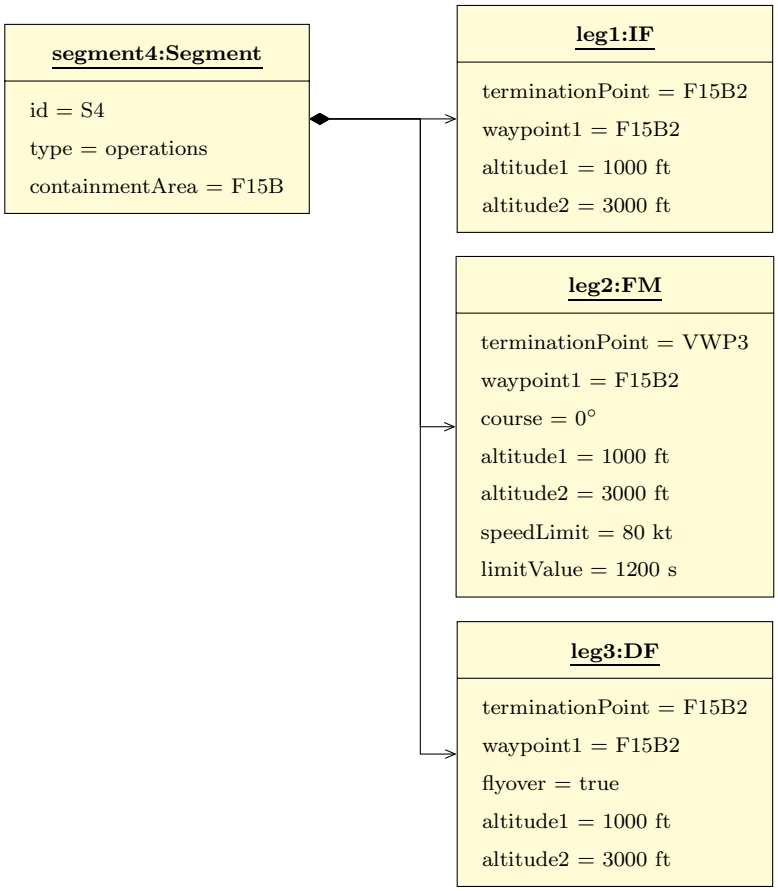


Figure 7.5: Demonstration mission: specification of the operations segment s_4.

With respect to the mission goals, based on the previous discussion, the nominal goal can be stated as: "to perform the manual task in the operations area and then land at LECH". Accordingly, the nominal goal is specified as a sequence of the two stages represented in Fig. 7.7. The first stage is a "fly-over" stage to fly over the termination point of the FM leg in s_4. This way, the first stage will be considered to be completed when the remote pilot ends the manual control. The second stage is a "land" stage in which the associated location is the set of IAFs of LECH (waypoints NIBEN and OSPES). For any reason, the current enabled variant is assumed to be NIBEN. The remaining alternate goals are all single staged, including:

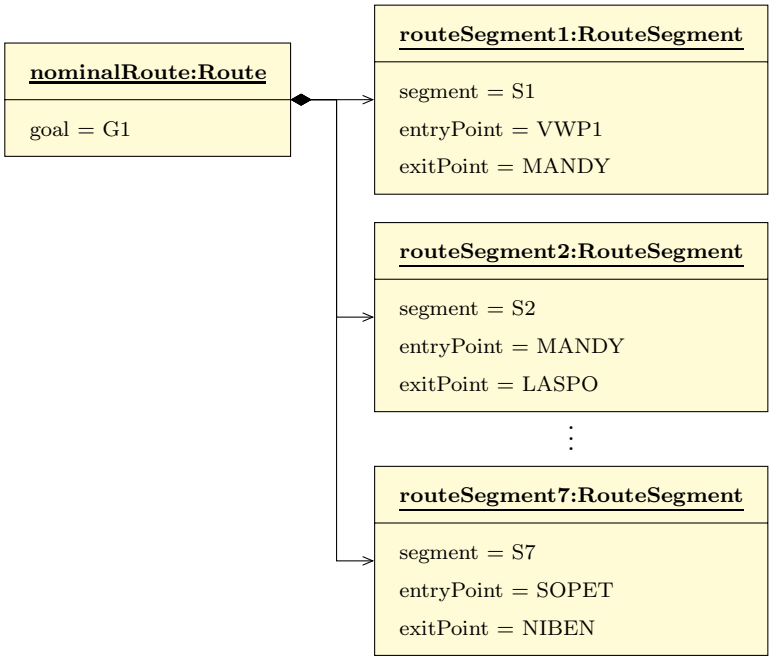


Figure 7.6: Demonstration mission: specification of the nominal route.

- Six "loiter" goals associated to waypoints MANDY, CLS, RETBA, F15B2 (see Fig. 7.8), SOPET and NIBEN.
- Three "regain signal" goals for performing climbs trying to regain the lost signal at waypoints FTP1, FTP2, and FTP3 (see Fig. 7.9).
- Three "land" goals for performing the landing at LETL (associated locations LETL18IAF and LETL36IAF), at LERE (LERE12IAF and LERE30IAF), and at LECH (NIBEN and OSPES).
- Three "flight termination" goals for performing the flight termination action at waypoints FTP1, FTP2, and FTP3.

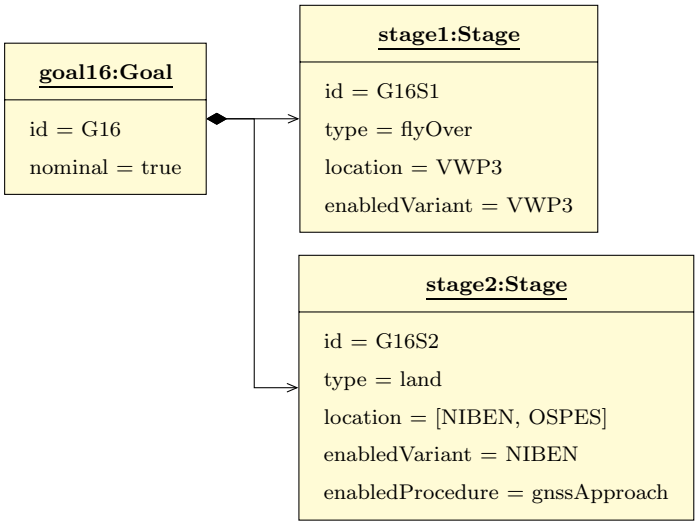


Figure 7.7: Demonstration mission: specification of the nominal goal.

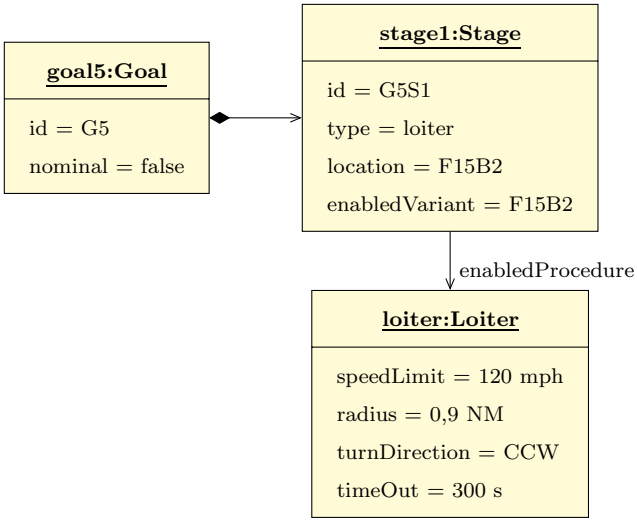


Figure 7.8: Demonstration mission: specification of a "loiter" goal.

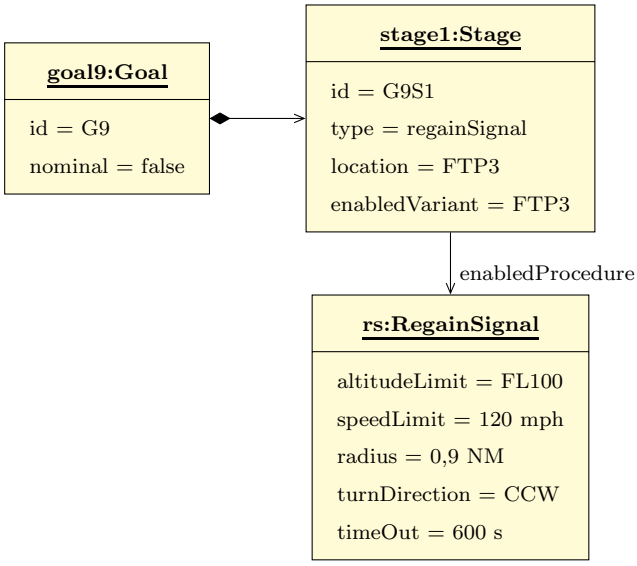


Figure 7.9: Demonstration mission: specification of a "climb to regain signal" goal.

7.2.2 Mission risk assessment

A risk assessment will be performed for this mission based on the risk models of Chapter 6. Recall that these models estimate the expected number of fatal injuries to third party people per hour of operation along a nautical mile; and that, in this work, this safety metric will be used to assign the edges' weights of the Mission Graph. So first, the Mission Graph for this mission will be automatically generated from the previous Mission Plan specification using the algorithm in Listing C.1. The resulting graph is depicted in Fig. 7.10. The correspondence between the node names in this figure and their associated waypoints is presented in Table 7.1. As it can be observed, the Mission Graph is a connected, directed graph with cycles. It has 2 source nodes ($w_{1,1}$ and $w_{8,1}$, each one associated with a runway direction at the departure site); and 6 sinks ($w_{7,3}$, $w_{9,3}$, $w_{10,3}$, $w_{11,4}$, $w_{12,2}$ and $w_{13,5}$, associated with the different "land" goals). In this figure, transition edges are plotted with red dashed lines, and the nominal route is highlighted using a solid line defining the following path:

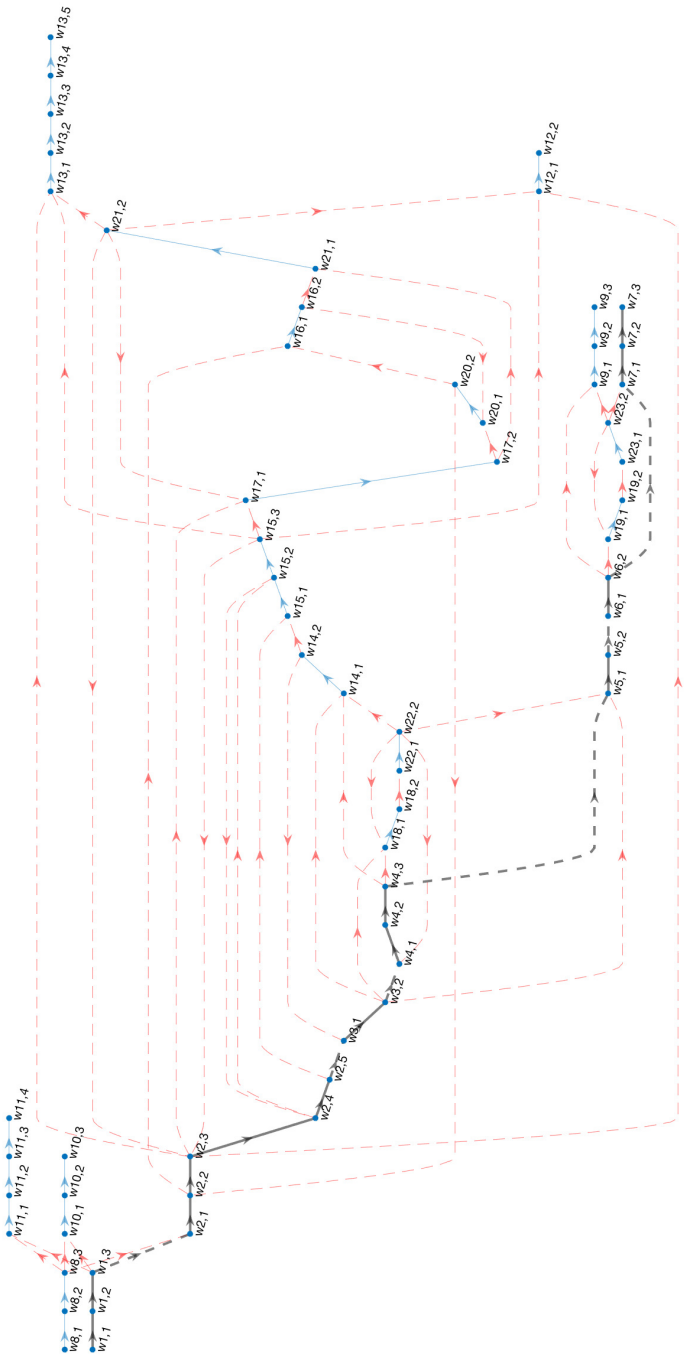


Figure 7.10: Demonstration mission: Mission Graph. Note that nominal route is highlighted in solid line; edge weights are omitted for clarity; and transition edges are plotted with red dashed lines.

Table 7.1: Demonstration mission: correspondence between nodes of the Mission Graph and their associated waypoints.

Node	Waypoint	Node	Waypoint	Node	Waypoint	Node	Waypoint
$w_{1,1}$	VWP1	$w_{6,2}$	SOPET	$w_{11,4}$	LETL18IAF	$w_{17,2}$	FTP1
$w_{1,2}$	VWP2	$w_{7,1}$	SOPET	$w_{12,1}$	RETBA	$w_{18,1}$	F15B2
$w_{1,3}$	MANDY	$w_{7,2}$	TATOS	$w_{12,2}$	LERE30IAF	$w_{18,2}$	FTP2
$w_{2,1}$	MANDY	$w_{7,3}$	NIBEN	$w_{13,1}$	RETBA	$w_{19,1}$	SOPET
$w_{2,2}$	CLS	$w_{8,1}$	VWP4	$w_{13,2}$	LERE30IAF	$w_{19,2}$	FTP3
$w_{2,3}$	RETBA	$w_{8,2}$	VWP5	$w_{13,3}$	LEREAUX1	$w_{20,1}$	FTP1
$w_{2,4}$	MOPIR	$w_{8,3}$	MANDY	$w_{13,4}$	LEREAUX2	$w_{20,2}$	CLS
$w_{2,5}$	LASPO	$w_{9,1}$	SOPET	$w_{13,5}$	LERE12IAF	$w_{21,1}$	FTP1
$w_{3,1}$	LASPO	$w_{9,2}$	LECHAUX1	$w_{14,1}$	F15B2	$w_{21,2}$	RETBA
$w_{3,2}$	F15B2	$w_{9,3}$	OSPES	$w_{14,2}$	LASPO	$w_{22,1}$	FTP2
$w_{4,1}$	F15B2	$w_{10,1}$	MANDY	$w_{15,1}$	LASPO	$w_{22,2}$	F15B2
$w_{4,2}$	VWP3	$w_{10,2}$	LETLAUX1	$w_{15,2}$	MOPIR	$w_{23,1}$	FTP3
$w_{4,3}$	F15B2	$w_{10,3}$	LETL36IAF	$w_{15,3}$	RETBA	$w_{23,2}$	SOPET
$w_{5,1}$	F15B2	$w_{11,1}$	MANDY	$w_{16,1}$	CLS		
$w_{5,2}$	VLC	$w_{11,2}$	LETLAUX1	$w_{16,2}$	FTP1		
$w_{6,1}$	VLC	$w_{11,3}$	LETLAUX2	$w_{17,1}$	RETBA		

$$\begin{aligned} r_0 = \langle & w_{1,1} \to w_{1,2} \to w_{1,3} \to w_{2,1} \to w_{2,2} \to w_{2,3} \to w_{2,4} \to w_{2,5} \to \dots \\ & w_{3,1} \to w_{3,2} \to w_{4,1} \to w_{4,2} \to w_{4,3} \to w_{5,1} \to w_{5,2} \to w_{6,1} \to \dots \\ & w_{6,2} \to w_{7,1} \to w_{7,2} \to w_{7,3} \rangle \end{aligned} \tag{7.1}$$

Then, the mission risk assessment producing the edges' weights for this graph will be performed using the algorithm in Listing C.2. In this case study, the risk will be measured for 6 different operational conditions (named as **OC1** to **OC6**), outlined in Table 7.2 and described next:

- In **OC1**, the RPAS is operating under a nominal state (i.e. there is no evidence of a contingency event having occurred). The RPAS is not equipped with a DAA system.
- In **OC2**, the RPAS is operating in an autonomous mode (i.e. there is a C2 link loss evidence). The RPAS is not equipped with a DAA system.

Table 7.2: Operational conditions evaluated in the mission risk assessment.

Operational condition	Contingency evidence observed	DAA equipped
OC1	None	No
OC2	C2 link loss	No
OC3	GNSS loss of performance	No
OC4	None	Yes
OC5	C2 link loss	Yes
OC6	GNSS loss of performance	Yes

- In **OC3**, the RPAS is operating in a degraded navigation state (i.e. there is a GNSS loss of performance evidence). The RPAS is not equipped with a DAA system.
- In **OC4**, the RPAS is operating under a nominal state. The RPAS is equipped with a DAA system compliant with the RTCA SC-228 or the EUROCAE WG-105 MOPS standard.
- In **OC5**, the RPAS is operating in an autonomous mode. The RPAS is equipped with a DAA system compliant with the RTCA SC-228 or the EUROCAE WG-105 MOPS standard.
- In **OC6**, the RPAS is operating in a degraded navigation state. The RPAS is equipped with a DAA system compliant with the RTCA SC-228 or the EUROCAE WG-105 MOPS standard.

The risk assessment results for each leg in each of these operational conditions are presented in Tables 7.3 to 7.8. The results in these tables show the path length (L), the ground risk (Λ_G), the air risk (Λ_A), the total risk (c), and the average risk ($\bar{\lambda}$) for all the edges of the Mission Graph (except the transition edges, whose cost is zero by definition). The ground risk and the air risk components are computed using Equations (6.19) and (6.23), except for the edge ID 8 (associated with the FM leg in segment s_4), where risk components are computed using Eq. 6.24 (see Sec. 6.4). The total risk c is computed using Eq. (6.25). Note that this parameter is the one that is assigned to the edges' weights. Finally, the average risk $\bar{\lambda}$ is computed using Eq. (6.27) considering a single edge.

Table 7.3: Mission risk assessment results for operational condition **OC1**.

Edge ID	Source node	Target node	L [NM]	Λ_G [h^{-1}NM]	Λ_A [h^{-1}NM]	c [h^{-1}NM]	$\bar{\lambda}$ [h^{-1}]
1	$w_{1,1}$	$w_{1,2}$	3	3,11e-05	1,54e-03	1,57e-03	5,22e-04
2	$w_{1,2}$	$w_{1,3}$	27	4,37e-05	1,38e-02	1,39e-02	5,14e-04
3	$w_{2,1}$	$w_{2,2}$	13	1,08e-05	5,89e-03	5,90e-03	4,54e-04
4	$w_{2,2}$	$w_{2,3}$	16	2,83e-05	3,63e-03	3,65e-03	2,28e-04
5	$w_{2,3}$	$w_{2,4}$	4	3,51e-05	1,81e-03	1,85e-03	4,62e-04
6	$w_{2,4}$	$w_{2,5}$	16	1,33e-04	7,25e-03	7,39e-03	4,62e-04
7	$w_{3,1}$	$w_{3,2}$	8	4,22e-04	4,10e-03	4,52e-03	5,65e-04
8	$w_{4,1}$	$w_{4,2}$	30	5,80e-04	1,54e-02	1,59e-02	5,31e-04
9	$w_{4,2}$	$w_{4,3}$	2	1,19e-03	1,02e-03	2,21e-03	1,11e-03
10	$w_{5,1}$	$w_{5,2}$	12	3,95e-03	3,07e-03	7,02e-03	5,85e-04
11	$w_{6,1}$	$w_{6,2}$	30	2,66e-03	1,02e-02	1,29e-02	4,29e-04
12	$w_{7,1}$	$w_{7,2}$	16	7,53e-04	7,25e-03	8,01e-03	5,00e-04
13	$w_{7,2}$	$w_{7,3}$	4	8,55e-05	1,36e-03	1,45e-03	3,61e-04
14	$w_{8,1}$	$w_{8,2}$	3	3,11e-05	1,54e-03	1,57e-03	5,22e-04
15	$w_{8,2}$	$w_{8,3}$	33	1,11e-04	1,69e-02	1,70e-02	5,16e-04
16	$w_{9,1}$	$w_{9,2}$	29	0	9,86e-03	9,86e-03	3,40e-04
17	$w_{9,2}$	$w_{9,3}$	14	3,86e-05	4,76e-03	4,80e-03	3,43e-04
18	$w_{10,1}$	$w_{10,2}$	25	3,38e-05	1,28e-02	1,28e-02	5,13e-04
19	$w_{10,2}$	$w_{10,3}$	3	3,18e-05	1,54e-03	1,57e-03	5,23e-04
20	$w_{11,1}$	$w_{11,2}$	25	3,38e-05	1,28e-02	1,28e-02	5,13e-04
21	$w_{11,2}$	$w_{11,3}$	12	7,67e-05	6,15e-03	6,22e-03	5,19e-04
22	$w_{11,3}$	$w_{11,4}$	3	4,03e-06	1,54e-03	1,54e-03	5,13e-04
23	$w_{12,1}$	$w_{12,2}$	1	1,26e-05	5,12e-04	5,25e-04	5,25e-04
24	$w_{13,1}$	$w_{13,2}$	1	1,26e-05	5,12e-04	5,25e-04	5,25e-04
25	$w_{13,2}$	$w_{13,3}$	3	1,43e-05	1,54e-03	1,55e-03	5,17e-04
26	$w_{13,3}$	$w_{13,4}$	14	3,32e-05	7,17e-03	7,20e-03	5,15e-04
27	$w_{13,4}$	$w_{13,5}$	3	8,73e-06	1,54e-03	1,55e-03	5,15e-04
28	$w_{14,1}$	$w_{14,2}$	8	4,22e-04	6,15e-03	6,57e-03	8,21e-04
29	$w_{15,1}$	$w_{15,2}$	16	1,33e-04	1,23e-02	1,24e-02	7,77e-04
30	$w_{15,2}$	$w_{15,3}$	4	3,51e-05	1,02e-03	1,06e-03	2,65e-04
31	$w_{16,1}$	$w_{16,2}$	15	4,12e-05	7,68e-03	7,72e-03	5,15e-04
32	$w_{17,1}$	$w_{17,2}$	16	4,35e-05	8,19e-03	8,24e-03	5,15e-04
33	$w_{18,1}$	$w_{18,2}$	16	5,82e-04	8,19e-03	8,78e-03	5,49e-04
34	$w_{19,1}$	$w_{19,2}$	9	0	4,61e-03	4,61e-03	5,12e-04

35	$w_{20,1}$	$w_{20,2}$	15	4,12e-05	1,15e-02	1,16e-02	7,71e-04
36	$w_{21,1}$	$w_{21,2}$	16	4,35e-05	4,10e-03	4,14e-03	2,59e-04
37	$w_{22,1}$	$w_{22,2}$	16	5,82e-04	8,19e-03	8,78e-03	5,49e-04
38	$w_{23,1}$	$w_{23,2}$	9	0	4,61e-03	4,61e-03	5,12e-04

Table 7.4: Mission risk assessment results for operational condition **OC2**.

Edge ID	Source node	Target node	L [NM]	Λ_G [h^{-1}NM]	Λ_A [h^{-1}NM]	c [h^{-1}NM]	$\bar{\lambda}$ [h^{-1}]
1	$w_{1,1}$	$w_{1,2}$	3	4,89e-05	2,43e-03	2,48e-03	8,26e-04
2	$w_{1,2}$	$w_{1,3}$	27	6,88e-05	2,19e-02	2,19e-02	8,12e-04
3	$w_{2,1}$	$w_{2,2}$	13	1,70e-05	9,32e-03	9,33e-03	7,18e-04
4	$w_{2,2}$	$w_{2,3}$	16	4,46e-05	5,73e-03	5,78e-03	3,61e-04
5	$w_{2,3}$	$w_{2,4}$	4	5,52e-05	2,87e-03	2,92e-03	7,30e-04
6	$w_{2,4}$	$w_{2,5}$	16	2,09e-04	1,15e-02	1,17e-02	7,30e-04
7	$w_{3,1}$	$w_{3,2}$	8	6,63e-04	6,48e-03	7,14e-03	8,93e-04
8	$w_{4,1}$	$w_{4,2}$	30	9,12e-04	2,43e-02	2,52e-02	8,40e-04
9	$w_{4,2}$	$w_{4,3}$	2	1,87e-03	1,62e-03	3,49e-03	1,74e-03
10	$w_{5,1}$	$w_{5,2}$	12	6,22e-03	4,86e-03	1,11e-02	9,23e-04
11	$w_{6,1}$	$w_{6,2}$	30	4,19e-03	1,61e-02	2,03e-02	6,77e-04
12	$w_{7,1}$	$w_{7,2}$	16	1,18e-03	1,15e-02	1,27e-02	7,91e-04
13	$w_{7,2}$	$w_{7,3}$	4	1,35e-04	2,15e-03	2,28e-03	5,71e-04
14	$w_{8,1}$	$w_{8,2}$	3	4,89e-05	2,43e-03	2,48e-03	8,26e-04
15	$w_{8,2}$	$w_{8,3}$	33	1,75e-04	2,67e-02	2,69e-02	8,15e-04
16	$w_{9,1}$	$w_{9,2}$	29	0	1,56e-02	1,56e-02	5,37e-04
17	$w_{9,2}$	$w_{9,3}$	14	6,07e-05	7,52e-03	7,59e-03	5,42e-04
18	$w_{10,1}$	$w_{10,2}$	25	5,32e-05	2,02e-02	2,03e-02	8,12e-04
19	$w_{10,2}$	$w_{10,3}$	3	5,01e-05	2,43e-03	2,48e-03	8,26e-04
20	$w_{11,1}$	$w_{11,2}$	25	5,32e-05	2,02e-02	2,03e-02	8,12e-04
21	$w_{11,2}$	$w_{11,3}$	12	1,21e-04	9,72e-03	9,84e-03	8,20e-04
22	$w_{11,3}$	$w_{11,4}$	3	6,34e-06	2,43e-03	2,44e-03	8,12e-04
23	$w_{12,1}$	$w_{12,2}$	1	1,98e-05	8,10e-04	8,29e-04	8,29e-04
24	$w_{13,1}$	$w_{13,2}$	1	1,98e-05	8,10e-04	8,29e-04	8,29e-04
25	$w_{13,2}$	$w_{13,3}$	3	2,25e-05	2,43e-03	2,45e-03	8,17e-04
26	$w_{13,3}$	$w_{13,4}$	14	5,22e-05	1,13e-02	1,14e-02	8,13e-04
27	$w_{13,4}$	$w_{13,5}$	3	1,37e-05	2,43e-03	2,44e-03	8,14e-04
28	$w_{14,1}$	$w_{14,2}$	8	6,63e-04	9,72e-03	1,04e-02	1,30e-03
29	$w_{15,1}$	$w_{15,2}$	16	2,09e-04	1,94e-02	1,96e-02	1,23e-03

30	$w_{15,2}$	$w_{15,3}$	4	5,52e-05	1,62e-03	1,67e-03	4,19e-04
31	$w_{16,1}$	$w_{16,2}$	15	6,49e-05	1,21e-02	1,22e-02	8,14e-04
32	$w_{17,1}$	$w_{17,2}$	16	6,84e-05	1,30e-02	1,30e-02	8,14e-04
33	$w_{18,1}$	$w_{18,2}$	16	9,16e-04	1,30e-02	1,39e-02	8,67e-04
34	$w_{19,1}$	$w_{19,2}$	9	0	7,29e-03	7,29e-03	8,10e-04
35	$w_{20,1}$	$w_{20,2}$	15	6,49e-05	1,82e-02	1,83e-02	1,22e-03
36	$w_{21,1}$	$w_{21,2}$	16	6,84e-05	6,48e-03	6,55e-03	4,09e-04
37	$w_{22,1}$	$w_{22,2}$	16	9,16e-04	1,30e-02	1,39e-02	8,67e-04
38	$w_{23,1}$	$w_{23,2}$	9	0	7,29e-03	7,29e-03	8,10e-04

Table 7.5: Mission risk assessment results for operational condition **OC3**.

Edge ID	Source node	Target node	L [NM]	Λ_G $[\mathrm{h}^{-1}\mathrm{NM}]$	Λ_A $[\mathrm{h}^{-1}\mathrm{NM}]$	c $[\mathrm{h}^{-1}\mathrm{NM}]$	$\bar{\lambda}$ $[\mathrm{h}^{-1}]$
1	$w_{1,1}$	$w_{1,2}$	3	5,45e-05	1,85e-03	1,90e-03	6,34e-04
2	$w_{1,2}$	$w_{1,3}$	27	7,67e-05	1,66e-02	1,67e-02	6,18e-04
3	$w_{2,1}$	$w_{2,2}$	13	1,90e-05	5,91e-03	5,93e-03	4,56e-04
4	$w_{2,2}$	$w_{2,3}$	16	4,97e-05	3,64e-03	3,68e-03	2,30e-04
5	$w_{2,3}$	$w_{2,4}$	4	6,15e-05	1,82e-03	1,88e-03	4,70e-04
6	$w_{2,4}$	$w_{2,5}$	16	2,33e-04	7,28e-03	7,51e-03	4,69e-04
7	$w_{3,1}$	$w_{3,2}$	8	7,39e-04	4,92e-03	5,66e-03	7,08e-04
8	$w_{4,1}$	$w_{4,2}$	30	1,02e-03	1,85e-02	1,95e-02	6,49e-04
9	$w_{4,2}$	$w_{4,3}$	2	2,06e-03	1,23e-03	3,29e-03	1,65e-03
10	$w_{5,1}$	$w_{5,2}$	12	6,93e-03	3,69e-03	1,06e-02	8,85e-04
11	$w_{6,1}$	$w_{6,2}$	30	4,67e-03	1,02e-02	1,49e-02	4,97e-04
12	$w_{7,1}$	$w_{7,2}$	16	1,32e-03	7,28e-03	8,60e-03	5,37e-04
13	$w_{7,2}$	$w_{7,3}$	4	1,50e-04	1,36e-03	1,51e-03	3,79e-04
14	$w_{8,1}$	$w_{8,2}$	3	5,45e-05	1,85e-03	1,90e-03	6,34e-04
15	$w_{8,2}$	$w_{8,3}$	33	1,95e-04	2,03e-02	2,05e-02	6,21e-04
16	$w_{9,1}$	$w_{9,2}$	29	0	9,89e-03	9,89e-03	3,41e-04
17	$w_{9,2}$	$w_{9,3}$	14	6,76e-05	4,77e-03	4,84e-03	3,46e-04
18	$w_{10,1}$	$w_{10,2}$	25	5,87e-05	1,54e-02	1,54e-02	6,18e-04
19	$w_{10,2}$	$w_{10,3}$	3	5,52e-05	1,85e-03	1,90e-03	6,34e-04
20	$w_{11,1}$	$w_{11,2}$	25	5,87e-05	1,54e-02	1,54e-02	6,18e-04
21	$w_{11,2}$	$w_{11,3}$	12	1,33e-04	7,38e-03	7,52e-03	6,26e-04
22	$w_{11,3}$	$w_{11,4}$	3	6,99e-06	1,85e-03	1,85e-03	6,18e-04
23	$w_{12,1}$	$w_{12,2}$	1	2,18e-05	6,15e-04	6,37e-04	6,37e-04
24	$w_{13,1}$	$w_{13,2}$	1	2,18e-05	6,15e-04	6,37e-04	6,37e-04

25	$w_{13,2}$	$w_{13,3}$	3	2,48e-05	1,85e-03	1,87e-03	6,24e-04
26	$w_{13,3}$	$w_{13,4}$	14	5,76e-05	8,61e-03	8,67e-03	6,19e-04
27	$w_{13,4}$	$w_{13,5}$	3	1,52e-05	1,85e-03	1,86e-03	6,20e-04
28	$w_{14,1}$	$w_{14,2}$	8	7,39e-04	7,38e-03	8,12e-03	1,02e-03
29	$w_{15,1}$	$w_{15,2}$	16	2,33e-04	1,48e-02	1,50e-02	9,38e-04
30	$w_{15,2}$	$w_{15,3}$	4	6,15e-05	1,23e-03	1,29e-03	3,23e-04
31	$w_{16,1}$	$w_{16,2}$	15	7,23e-05	9,23e-03	9,30e-03	6,20e-04
32	$w_{17,1}$	$w_{17,2}$	16	7,62e-05	9,85e-03	9,92e-03	6,20e-04
33	$w_{18,1}$	$w_{18,2}$	16	1,02e-03	9,85e-03	1,09e-02	6,79e-04
34	$w_{19,1}$	$w_{19,2}$	9	0	5,54e-03	5,54e-03	6,15e-04
35	$w_{20,1}$	$w_{20,2}$	15	7,23e-05	1,38e-02	1,39e-02	9,28e-04
36	$w_{21,1}$	$w_{21,2}$	16	7,62e-05	4,92e-03	5,00e-03	3,12e-04
37	$w_{22,1}$	$w_{22,2}$	16	1,02e-03	9,85e-03	1,09e-02	6,79e-04
38	$w_{23,1}$	$w_{23,2}$	9	0	5,54e-03	5,54e-03	6,15e-04

Table 7.6: Mission risk assessment results for operational condition **OC4**.

Edge ID	Source node	Target node	L [NM]	Λ_G [h^{-1}NM]	Λ_A [h^{-1}NM]	c [h^{-1}NM]	$\bar{\lambda}$ [h^{-1}]
1	$w_{1,1}$	$w_{1,2}$	3	3,11e-05	1,54e-08	3,11e-05	1,04e-05
2	$w_{1,2}$	$w_{1,3}$	27	4,37e-05	1,38e-07	4,39e-05	1,63e-06
3	$w_{2,1}$	$w_{2,2}$	13	1,08e-05	5,89e-08	1,09e-05	8,37e-07
4	$w_{2,2}$	$w_{2,3}$	16	2,83e-05	3,63e-08	2,84e-05	1,77e-06
5	$w_{2,3}$	$w_{2,4}$	4	3,51e-05	1,81e-08	3,51e-05	8,78e-06
6	$w_{2,4}$	$w_{2,5}$	16	1,33e-04	7,25e-08	1,33e-04	8,30e-06
7	$w_{3,1}$	$w_{3,2}$	8	4,22e-04	4,10e-08	4,22e-04	5,27e-05
8	$w_{4,1}$	$w_{4,2}$	30	5,80e-04	1,54e-07	5,80e-04	1,93e-05
9	$w_{4,2}$	$w_{4,3}$	2	1,19e-03	1,02e-08	1,19e-03	5,93e-04
10	$w_{5,1}$	$w_{5,2}$	12	3,95e-03	3,07e-08	3,95e-03	3,29e-04
11	$w_{6,1}$	$w_{6,2}$	30	2,66e-03	1,02e-07	2,66e-03	8,87e-05
12	$w_{7,1}$	$w_{7,2}$	16	7,53e-04	7,25e-08	7,53e-04	4,71e-05
13	$w_{7,2}$	$w_{7,3}$	4	8,55e-05	1,36e-08	8,55e-05	2,14e-05
14	$w_{8,1}$	$w_{8,2}$	3	3,11e-05	1,54e-08	3,11e-05	1,04e-05
15	$w_{8,2}$	$w_{8,3}$	33	1,11e-04	1,69e-07	1,11e-04	3,37e-06
16	$w_{9,1}$	$w_{9,2}$	29	0	9,86e-08	9,86e-08	3,40e-09
17	$w_{9,2}$	$w_{9,3}$	14	3,86e-05	4,76e-08	3,86e-05	2,76e-06
18	$w_{10,1}$	$w_{10,2}$	25	3,38e-05	1,28e-07	3,40e-05	1,36e-06
19	$w_{10,2}$	$w_{10,3}$	3	3,18e-05	1,54e-08	3,18e-05	1,06e-05

20	$w_{11,1}$	$w_{11,2}$	25	3,38e-05	1,28e-07	3,40e-05	1,36e-06
21	$w_{11,2}$	$w_{11,3}$	12	7,67e-05	6,15e-08	7,67e-05	6,40e-06
22	$w_{11,3}$	$w_{11,4}$	3	4,03e-06	1,54e-08	4,04e-06	1,35e-06
23	$w_{12,1}$	$w_{12,2}$	1	1,26e-05	5,12e-09	1,26e-05	1,26e-05
24	$w_{13,1}$	$w_{13,2}$	1	1,26e-05	5,12e-09	1,26e-05	1,26e-05
25	$w_{13,2}$	$w_{13,3}$	3	1,43e-05	1,54e-08	1,43e-05	4,77e-06
26	$w_{13,3}$	$w_{13,4}$	14	3,32e-05	7,17e-08	3,32e-05	2,37e-06
27	$w_{13,4}$	$w_{13,5}$	3	8,73e-06	1,54e-08	8,74e-06	2,91e-06
28	$w_{14,1}$	$w_{14,2}$	8	4,22e-04	6,15e-08	4,22e-04	5,27e-05
29	$w_{15,1}$	$w_{15,2}$	16	1,33e-04	1,23e-07	1,33e-04	8,30e-06
30	$w_{15,2}$	$w_{15,3}$	4	3,51e-05	1,02e-08	3,51e-05	8,77e-06
31	$w_{16,1}$	$w_{16,2}$	15	4,12e-05	7,68e-08	4,13e-05	2,75e-06
32	$w_{17,1}$	$w_{17,2}$	16	4,35e-05	8,19e-08	4,36e-05	2,72e-06
33	$w_{18,1}$	$w_{18,2}$	16	5,82e-04	8,19e-08	5,82e-04	3,64e-05
34	$w_{19,1}$	$w_{19,2}$	9	0	4,61e-08	4,61e-08	5,12e-09
35	$w_{20,1}$	$w_{20,2}$	15	4,12e-05	1,15e-07	4,14e-05	2,76e-06
36	$w_{21,1}$	$w_{21,2}$	16	4,35e-05	4,10e-08	4,35e-05	2,72e-06
37	$w_{22,1}$	$w_{22,2}$	16	5,82e-04	8,19e-08	5,82e-04	3,64e-05
38	$w_{23,1}$	$w_{23,2}$	9	0	4,61e-08	4,61e-08	5,12e-09

Table 7.7: Mission risk assessment results for operational condition **OC5**.

Edge ID	Source node	Target node	L [NM]	Λ_G [h^{-1}NM]	Λ_A [h^{-1}NM]	c [h^{-1}NM]	$\bar{\lambda}$ [h^{-1}]
1	$w_{1,1}$	$w_{1,2}$	3	4,89e-05	2,43e-08	4,89e-05	1,63e-05
2	$w_{1,2}$	$w_{1,3}$	27	6,88e-05	2,19e-07	6,90e-05	2,56e-06
3	$w_{2,1}$	$w_{2,2}$	13	1,70e-05	9,32e-08	1,71e-05	1,32e-06
4	$w_{2,2}$	$w_{2,3}$	16	4,46e-05	5,73e-08	4,46e-05	2,79e-06
5	$w_{2,3}$	$w_{2,4}$	4	5,52e-05	2,87e-08	5,52e-05	1,38e-05
6	$w_{2,4}$	$w_{2,5}$	16	2,09e-04	1,15e-07	2,09e-04	1,31e-05
7	$w_{3,1}$	$w_{3,2}$	8	6,63e-04	6,48e-08	6,63e-04	8,29e-05
8	$w_{4,1}$	$w_{4,2}$	30	9,12e-04	2,43e-07	9,12e-04	3,04e-05
9	$w_{4,2}$	$w_{4,3}$	2	1,87e-03	1,62e-08	1,87e-03	9,34e-04
10	$w_{5,1}$	$w_{5,2}$	12	6,22e-03	4,86e-08	6,22e-03	5,18e-04
11	$w_{6,1}$	$w_{6,2}$	30	4,19e-03	1,61e-07	4,19e-03	1,40e-04
12	$w_{7,1}$	$w_{7,2}$	16	1,18e-03	1,15e-07	1,18e-03	7,40e-05
13	$w_{7,2}$	$w_{7,3}$	4	1,35e-04	2,15e-08	1,35e-04	3,36e-05
14	$w_{8,1}$	$w_{8,2}$	3	4,89e-05	2,43e-08	4,89e-05	1,63e-05

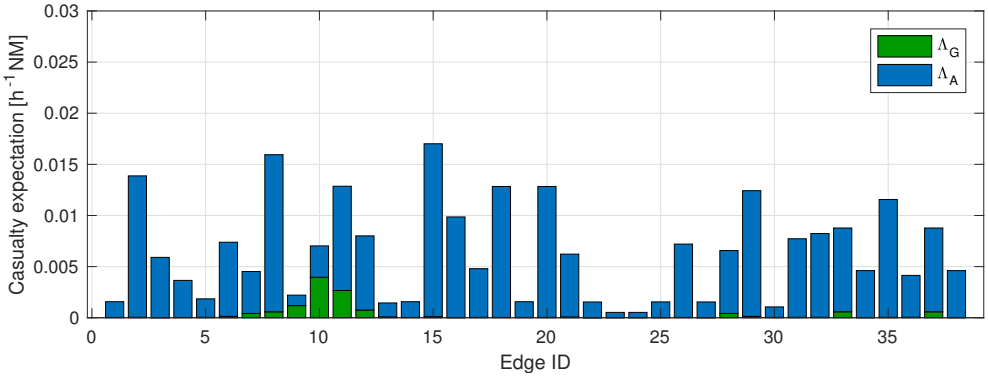
15	$w_{8,2}$	$w_{8,3}$	33	1,75e-04	2,67e-07	1,75e-04	5,30e-06
16	$w_{9,1}$	$w_{9,2}$	29	0	1,56e-07	1,56e-07	5,37e-09
17	$w_{9,2}$	$w_{9,3}$	14	6,07e-05	7,52e-08	6,07e-05	4,34e-06
18	$w_{10,1}$	$w_{10,2}$	25	5,32e-05	2,02e-07	5,34e-05	2,14e-06
19	$w_{10,2}$	$w_{10,3}$	3	5,01e-05	2,43e-08	5,01e-05	1,67e-05
20	$w_{11,1}$	$w_{11,2}$	25	5,32e-05	2,02e-07	5,34e-05	2,14e-06
21	$w_{11,2}$	$w_{11,3}$	12	1,21e-04	9,72e-08	1,21e-04	1,01e-05
22	$w_{11,3}$	$w_{11,4}$	3	6,34e-06	2,43e-08	6,36e-06	2,12e-06
23	$w_{12,1}$	$w_{12,2}$	1	1,98e-05	8,10e-09	1,98e-05	1,98e-05
24	$w_{13,1}$	$w_{13,2}$	1	1,98e-05	8,10e-09	1,98e-05	1,98e-05
25	$w_{13,2}$	$w_{13,3}$	3	2,25e-05	2,43e-08	2,25e-05	7,51e-06
26	$w_{13,3}$	$w_{13,4}$	14	5,22e-05	1,13e-07	5,23e-05	3,74e-06
27	$w_{13,4}$	$w_{13,5}$	3	1,37e-05	2,43e-08	1,38e-05	4,59e-06
28	$w_{14,1}$	$w_{14,2}$	8	6,63e-04	9,72e-08	6,64e-04	8,29e-05
29	$w_{15,1}$	$w_{15,2}$	16	2,09e-04	1,94e-07	2,09e-04	1,31e-05
30	$w_{15,2}$	$w_{15,3}$	4	5,52e-05	1,62e-08	5,52e-05	1,38e-05
31	$w_{16,1}$	$w_{16,2}$	15	6,49e-05	1,21e-07	6,50e-05	4,33e-06
32	$w_{17,1}$	$w_{17,2}$	16	6,84e-05	1,30e-07	6,85e-05	4,28e-06
33	$w_{18,1}$	$w_{18,2}$	16	9,16e-04	1,30e-07	9,16e-04	5,73e-05
34	$w_{19,1}$	$w_{19,2}$	9	0	7,29e-08	7,29e-08	8,10e-09
35	$w_{20,1}$	$w_{20,2}$	15	6,49e-05	1,82e-07	6,51e-05	4,34e-06
36	$w_{21,1}$	$w_{21,2}$	16	6,84e-05	6,48e-08	6,85e-05	4,28e-06
37	$w_{22,1}$	$w_{22,2}$	16	9,16e-04	1,30e-07	9,16e-04	5,73e-05
38	$w_{23,1}$	$w_{23,2}$	9	0	7,29e-08	7,29e-08	8,10e-09

Table 7.8: Mission risk assessment results for operational condition **OC6**.

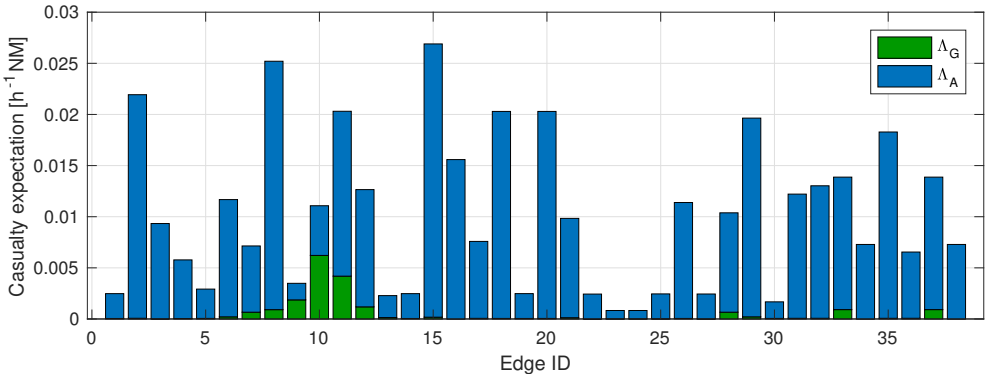
Edge ID	Source node	Target node	L [NM]	Λ_G $[\mathrm{h}^{-1}\mathrm{NM}]$	Λ_A $[\mathrm{h}^{-1}\mathrm{NM}]$	c $[\mathrm{h}^{-1}\mathrm{NM}]$	$\bar{\lambda}$ $[\mathrm{h}^{-1}]$
1	$w_{1,1}$	$w_{1,2}$	3	5,45e-05	1,86e-08	5,45e-05	1,82e-05
2	$w_{1,2}$	$w_{1,3}$	27	7,67e-05	1,67e-07	7,69e-05	2,85e-06
3	$w_{2,1}$	$w_{2,2}$	13	1,90e-05	5,91e-08	1,90e-05	1,46e-06
4	$w_{2,2}$	$w_{2,3}$	16	4,97e-05	3,64e-08	4,97e-05	3,11e-06
5	$w_{2,3}$	$w_{2,4}$	4	6,15e-05	1,82e-08	6,15e-05	1,54e-05
6	$w_{2,4}$	$w_{2,5}$	16	2,33e-04	7,28e-08	2,33e-04	1,45e-05
7	$w_{3,1}$	$w_{3,2}$	8	7,39e-04	4,95e-08	7,39e-04	9,24e-05
8	$w_{4,1}$	$w_{4,2}$	30	1,02e-03	1,86e-07	1,02e-03	3,39e-05
9	$w_{4,2}$	$w_{4,3}$	2	2,06e-03	1,24e-08	2,06e-03	1,03e-03

10	$w_{5,1}$	$w_{5,2}$	12	6,93e-03	3,71e-08	6,93e-03	5,77e-04
11	$w_{6,1}$	$w_{6,2}$	30	4,67e-03	1,02e-07	4,67e-03	1,56e-04
12	$w_{7,1}$	$w_{7,2}$	16	1,32e-03	7,28e-08	1,32e-03	8,25e-05
13	$w_{7,2}$	$w_{7,3}$	4	1,50e-04	1,36e-08	1,50e-04	3,75e-05
14	$w_{8,1}$	$w_{8,2}$	3	5,45e-05	1,86e-08	5,45e-05	1,82e-05
15	$w_{8,2}$	$w_{8,3}$	33	1,95e-04	2,04e-07	1,95e-04	5,91e-06
16	$w_{9,1}$	$w_{9,2}$	29	0	9,89e-08	9,89e-08	3,41e-09
17	$w_{9,2}$	$w_{9,3}$	14	6,76e-05	4,77e-08	6,77e-05	4,83e-06
18	$w_{10,1}$	$w_{10,2}$	25	5,87e-05	1,55e-07	5,89e-05	2,36e-06
19	$w_{10,2}$	$w_{10,3}$	3	5,52e-05	1,86e-08	5,53e-05	1,84e-05
20	$w_{11,1}$	$w_{11,2}$	25	5,87e-05	1,55e-07	5,89e-05	2,36e-06
21	$w_{11,2}$	$w_{11,3}$	12	1,33e-04	7,43e-08	1,33e-04	1,11e-05
22	$w_{11,3}$	$w_{11,4}$	3	6,99e-06	1,86e-08	7,01e-06	2,34e-06
23	$w_{12,1}$	$w_{12,2}$	1	2,18e-05	6,19e-09	2,18e-05	2,18e-05
24	$w_{13,1}$	$w_{13,2}$	1	2,18e-05	6,19e-09	2,18e-05	2,18e-05
25	$w_{13,2}$	$w_{13,3}$	3	2,48e-05	1,86e-08	2,48e-05	8,28e-06
26	$w_{13,3}$	$w_{13,4}$	14	5,76e-05	8,66e-08	5,77e-05	4,12e-06
27	$w_{13,4}$	$w_{13,5}$	3	1,52e-05	1,86e-08	1,52e-05	5,06e-06
28	$w_{14,1}$	$w_{14,2}$	8	7,39e-04	7,43e-08	7,40e-04	9,24e-05
29	$w_{15,1}$	$w_{15,2}$	16	2,33e-04	1,49e-07	2,33e-04	1,45e-05
30	$w_{15,2}$	$w_{15,3}$	4	6,15e-05	1,24e-08	6,15e-05	1,54e-05
31	$w_{16,1}$	$w_{16,2}$	15	7,23e-05	9,28e-08	7,24e-05	4,83e-06
32	$w_{17,1}$	$w_{17,2}$	16	7,62e-05	9,90e-08	7,63e-05	4,77e-06
33	$w_{18,1}$	$w_{18,2}$	16	1,02e-03	9,90e-08	1,02e-03	6,38e-05
34	$w_{19,1}$	$w_{19,2}$	9	0	5,57e-08	5,57e-08	6,19e-09
35	$w_{20,1}$	$w_{20,2}$	15	7,23e-05	1,39e-07	7,25e-05	4,83e-06
36	$w_{21,1}$	$w_{21,2}$	16	7,62e-05	4,95e-08	7,63e-05	4,77e-06
37	$w_{22,1}$	$w_{22,2}$	16	1,02e-03	9,90e-08	1,02e-03	6,38e-05
38	$w_{23,1}$	$w_{23,2}$	9	0	5,57e-08	5,57e-08	6,19e-09

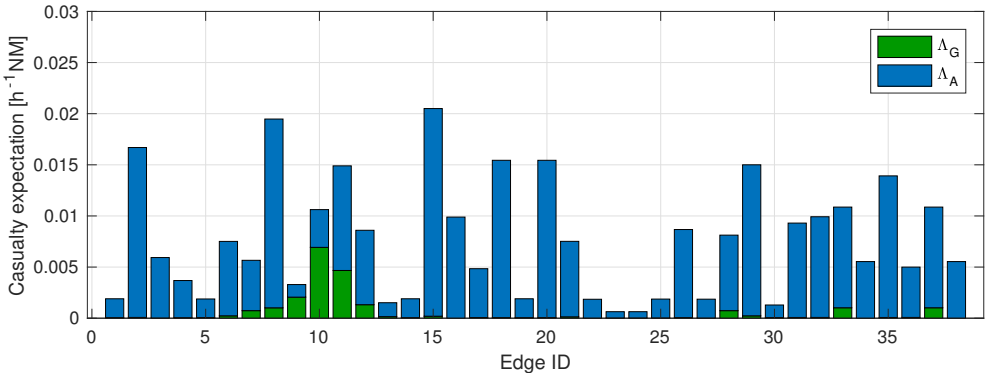
The ground risk and the air risk components for each edge in each operational condition are also graphically represented in Figures 7.11a to 7.11f. As it can be observed, the air risk component is the main contribution to the total risk whenever a DAA system is not equipped on the RPAS (see Figures 7.11a to 7.11c). However, this risk component can be almost entirely removed if a DAA system is equipped and it complies with the MOPS required by SORA. When it comes to the ground risk, it becomes a determining factor when overflying high population density areas like the metropolitan area of València (see



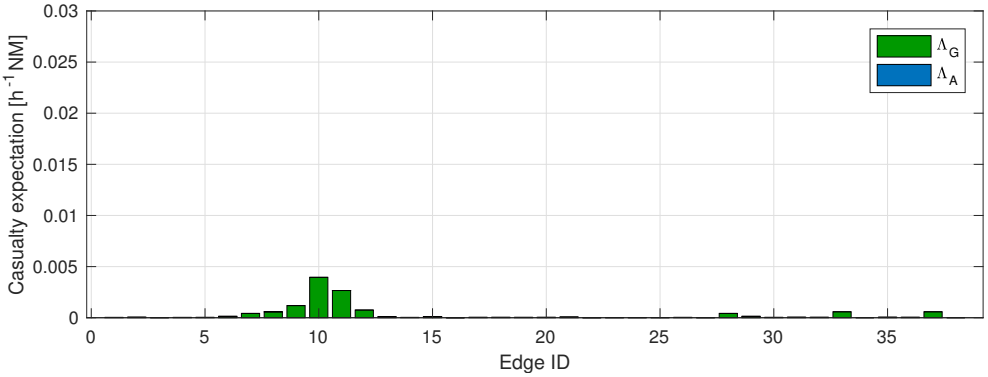
(a) Operational condition **OC1**.



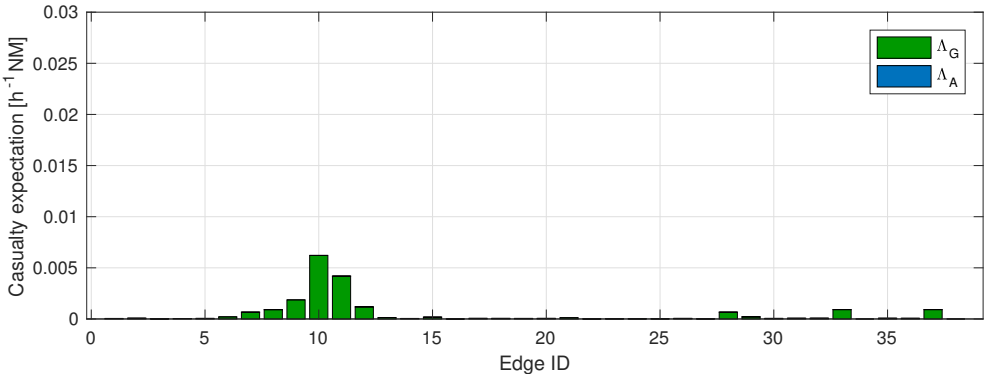
(b) Operational condition **OC2**.



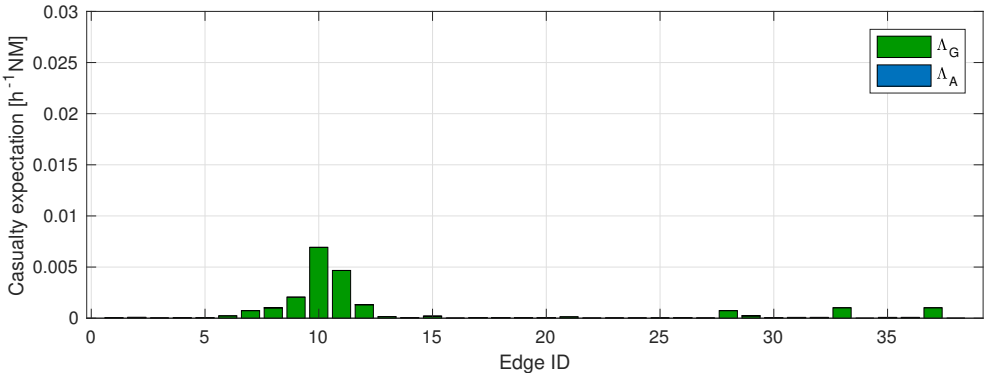
(c) Operational condition **OC3**.



(d) Operational condition **OC4**.



(e) Operational condition **OC5**.



(f) Operational condition **OC6**.

Figure 7.11: Mission risk assessment results: ground risk and air risk components.

Table 7.9: Overall results of the mission risk assessment.

Operational condition	Average $\bar{\lambda}_G$ $[\mathrm{h}^{-1}]$	Average $\bar{\lambda}_A$ $[\mathrm{h}^{-1}]$	Average $\bar{\lambda}$ $[\mathrm{h}^{-1}]$
OC1	3,71e-05	4,80e-04	5,17e-04
OC2	5,84e-05	7,59e-04	8,18e-04
OC3	6,47e-05	5,59e-04	6,24e-04
OC4	3,71e-05	4,80e-09	3,71e-05
OC5	5,84e-05	7,59e-09	5,84e-05
OC6	6,47e-05	5,62e-09	6,48e-05

edge IDs 10 and 11, i.e. legs ending at waypoints VLC and SOPET in Fig. 7.1). Note also that the ground risk component over the sea is considered to be zero because of the lack of data regarding marine traffic (see edge IDs 16, 34 and 38).

A summary of the risk assessment results is provided in Table 7.9. In particular, this table shows the average ground risk, the average air risk, and the average total risk (all of them expressed in terms of expected number of fatal injuries per hour of operation) for each operational condition under analysis. As it can be observed, the average air risk ($\bar{\lambda}_A$) is one order of magnitude higher than the average ground risk ($\bar{\lambda}_G$) in operational conditions **OC1** to **OC3**; while, $\bar{\lambda}_G$ prevails in **OC4** to **OC6**, where the DAA system is equipped.

Table 7.9 also allows to quantify the effect of contingencies on the proposed mission. For example, this table shows that the C2 link loss condition increases the total risk ($\bar{\lambda}$) by $58,22\%$ with respect to the nominal condition; while the degraded navigation condition increases it by $20,70\%$ (both considering that the DAA is not available in the system). By contrast, if a DAA system is equipped on the RPAS, the increase in risk when the C2 link is lost is similar to the previous case, but this time the navigation error increases the risk to a greater extent ($74,66\%$). This increased effect is due to the fact that the navigation error has a greater impact over the ground risk than over the air risk, plus the fact that the ground risk prevails over the air risk when the DAA system is available.

Finally, recall that the aim of the previous risk assessment is not to evaluate the risk along the entire mission in one specific operational condition. Conversely, the aim is to compute the risk associated to each leg in different flight conditions so that the dynamic route configuration algorithm can find the min-

imum cost route at some point of the mission execution. This way, the Mission Graph in Fig. 7.10 will be filled with the corresponding weights in flight time depending on the current operational condition of the RPAS.

7.2.3 Static route configuration analysis

Based on the previous results, it is possible to assess the risk posed by the RPAS when it flies some given mission route. Table 7.10 shows the cumulative risk when flying the nominal route in Eq. (7.1) under operational conditions **OC1** to **OC6**. Recall that the cumulative risk $\bar{c}$ is computed using Eq. (6.26) along the proposed route, i.e. by adding the edges' weights of the different edges traversed by this route; while the average risk $\bar{\lambda}$ is computed using Eq. (6.27). As an example, the cumulative risk when no contingency has occurred and no DAA system is equipped on the RPAS (**OC1**) is $\bar{c} = 8,62 \cdot 10^{-2}$ h^{-1}NM; although it can be reduced down to $\bar{c} = 9,92 \cdot 10^{-3}$ h^{-1}NM by means of the DAA capability (**OC4**). Considering that the estimated path length for this route is $L = 174$ NM, the average risk in these conditions is $\bar{\lambda} = 4,76 \cdot 10^{-4}$ h^{-1} and $\bar{\lambda} = 5,48 \cdot 10^{-5}$ h^{-1}, respectively.

Table 7.10: Cumulative risk and average risk when the RPAS flies the nominal route in the different operational conditions.

Operational condition	$\bar{c}$ $[h^{-1}NM]$	$\bar{\lambda}$ $[h^{-1}]$
OC1	8,62e-02	4,76e-04
OC2	1,36e-01	7,53e-04
OC3	1,02e-01	5,62e-04
OC4	9,92e-03	5,48e-05
OC5	1,56e-02	8,62e-05
OC6	1,74e-02	9,60e-05

Note that the cumulative risk for operational conditions **OC2**, **OC3**, **OC5** and **OC6** has been provided for completeness, although the obtained results are not significative: it is not expected to perform the entire mission in such degraded conditions.

7.2.4 Dynamic route configuration analysis

In the general case, the above results could be used to support the decision of whether the proposed mission meets an acceptable safety level or not. For example, we could assume than an average risk level above $\bar{\lambda} = 1 \cdot 10^{-5}$ fatalities per hour of operation is not acceptable. Therefore, we can assume that the proposed mission will not be approved unless the RPAS is equipped with a DAA system (what is also required by SORA). Consequently, operational conditions **OC1** to **OC3** will not be further discussed in this work. Moreover, one of the main contributions of this work is the idea that the previous static route can be reconfigured in flight time as a response to contingencies, thus lowering the resulting risk to a greater extent.

In order to analyze how dynamic route configuration can be used to minimize the operational risk in a contingency scenario, we will first perform a *goal reachability analysis*. This analysis exploits the `getPathToGoalsByType` procedure in Listing C.3 to check what goal types are achievable from each node in the Mission Graph, and what the cumulative risk associated with the route to achieve these goals is in each case. Then, based on the results of this analysis, we will assess how the new goal type defined by the Contingency Plan allows to reduce the overall risk after a contingency happens.

Goal reachability analysis

The results of the goal reachability analysis for operational conditions **OC4**, **OC5** and **OC6** are presented in Tables 7.11, 7.12 and 7.13, respectively. These tables show, for each node in the Mission Graph and for each goal type in a Reconfigurable Mission Plan[2], what is the optimal goal (in terms of risk) in the Mission Plan, what is the cumulative risk ($\bar{c}$) of the route that allows to achieve the optimal goal from the given node, and what is the average risk of this route ($\bar{\lambda}$).

[2]Note that "fly over" goals are not evaluated because this goal type is used to specify intermediate route constraints, not as a response to contingencies.

Table 7.11: Goal reachability analysis results for operational condition **OC4**.

Node	Goal type											
	Loiter			Regain signal			Land			Flight termation		
	Goal ID	$\bar{c}$ [$h^{-1}NM$]	$\bar{\lambda}$ [h^{-1}]	Goal ID	$\bar{c}$ [$h^{-1}NM$]	$\bar{\lambda}$ [h^{-1}]	Goal ID	$\bar{c}$ [$h^{-1}NM$]	$\bar{\lambda}$ [h^{-1}]	Goal ID	$\bar{c}$ [$h^{-1}NM$]	$\bar{\lambda}$ [h^{-1}]
$w_{1,1}$	1	7,50e-05	2,50e-06	7	1,27e-04	2,19e-06	12	1,27e-04	2,11e-06	13	1,27e-04	2,19e-06
$w_{1,2}$	1	4,39e-05	1,63e-06	7	9,61e-05	1,75e-06	12	9,57e-05	1,68e-06	13	9,61e-05	1,75e-06
$w_{1,3}$	1	0	–	7	5,22e-05	1,86e-06	12	5,18e-05	1,73e-06	13	5,22e-05	1,86e-06
$w_{2,1}$	1	0	–	7	5,22e-05	1,86e-06	12	5,18e-05	1,73e-06	13	5,22e-05	1,86e-06
$w_{2,2}$	2	0	–	7	4,13e-05	2,75e-06	12	4,09e-05	2,41e-06	13	4,13e-05	2,75e-06
$w_{2,3}$	3	0	–	7	4,36e-05	2,72e-06	12	1,26e-05	1,26e-05	13	4,36e-05	2,72e-06
$w_{2,4}$	3	3,51e-05	8,77e-06	7	7,87e-05	3,93e-06	12	4,77e-05	9,53e-06	13	7,87e-05	3,93e-06
$w_{2,5}$	3	1,68e-04	8,39e-06	7	2,11e-04	5,87e-06	12	1,80e-04	8,59e-06	13	2,11e-04	5,87e-06
$w_{3,1}$	5	4,22e-04	5,27e-05	8	1,00e-03	4,18e-05	12	1,02e-03	2,77e-05	14	1,00e-03	4,18e-05
$w_{3,2}$	5	0	–	8	5,82e-04	3,64e-05	12	6,02e-04	2,08e-05	14	5,82e-04	3,64e-05
$w_{4,1}$	5	0	–	8	2,35e-03	4,89e-05	12	2,37e-03	3,88e-05	14	2,35e-03	4,89e-05
$w_{4,2}$	5	1,19e-03	5,93e-04	8	1,77e-03	9,83e-05	12	1,79e-03	5,77e-05	14	1,77e-03	9,83e-05
$w_{4,3}$	5	0	–	8	5,82e-04	3,64e-05	12	6,02e-04	2,08e-05	14	5,82e-04	3,64e-05
$w_{5,1}$	5	0	–	9	6,61e-03	1,30e-04	11	7,45e-03	1,20e-04	15	6,61e-03	1,30e-04
$w_{5,2}$	4	2,66e-03	8,87e-05	9	2,66e-03	6,83e-05	11	3,50e-03	7,00e-05	15	2,66e-03	6,83e-05
$w_{6,1}$	4	2,66e-03	8,87e-05	9	2,66e-03	6,83e-05	11	3,50e-03	7,00e-05	15	2,66e-03	6,83e-05
$w_{6,2}$	4	0	–	9	4,61e-08	5,12e-09	11	8,39e-04	4,19e-05	15	4,61e-08	5,12e-09
$w_{7,1}$	4	0	–	–	–	–	11	8,39e-04	4,19e-05	–	–	–
$w_{7,2}$	6	8,55e-05	2,14e-05	–	–	–	11	8,55e-05	2,14e-05	–	–	–
$w_{7,3}$	6	0	–	–	–	–	11	0	–	–	–	–
$w_{8,1}$	1	1,42e-04	3,95e-06	7	1,95e-04	3,04e-06	12	1,94e-04	2,94e-06	13	1,95e-04	3,04e-06

$w_{8,2}$	1	1,11e-04	3,37e-06	7	1,63e-04	2,68e-06	12	1,63e-04	2,59e-06	13	1,63e-04	2,68e-06	
$w_{8,3}$	1	0	–	7	5,22e-05	1,86e-06	12	5,18e-05	1,73e-06	13	5,22e-05	1,86e-06	
$w_{9,1}$	4	0	–	–	–	–	–	–	–	–	–	–	
$w_{9,2}$	–	–	–	–	–	–	–	–	–	–	–	–	
$w_{9,3}$	–	–	–	–	–	–	–	–	–	–	–	–	
$w_{10,1}$	1	0	–	–	–	–	–	–	–	–	–	–	
$w_{10,2}$	–	–	–	–	–	–	–	–	–	–	–	–	
$w_{10,3}$	–	–	–	–	–	–	–	–	–	–	–	–	
$w_{11,1}$	1	0	–	–	–	–	10	1,15e-04	2,87e-06	–	–	–	
$w_{11,2}$	–	–	–	–	–	–	10	8,08e-05	5,39e-06	–	–	–	
$w_{11,3}$	–	–	–	–	–	–	10	4,04e-06	1,35e-06	–	–	–	
$w_{11,4}$	–	–	–	–	–	–	10	0	–	–	–	–	
$w_{12,1}$	3	0	–	–	–	–	12	1,26e-05	1,26e-05	–	–	–	
$w_{12,2}$	–	–	–	–	–	–	12	0	–	–	–	–	
$w_{13,1}$	3	0	–	–	–	–	12	1,26e-05	1,26e-05	–	–	–	
$w_{13,2}$	–	–	–	–	–	–	12	0	–	–	–	–	
$w_{13,3}$	–	–	–	–	–	–	–	–	–	–	–	–	
$w_{13,4}$	–	–	–	–	–	–	–	–	–	–	–	–	
$w_{13,5}$	–	–	–	–	–	–	–	–	–	–	–	–	
$w_{14,1}$	5	0	–	7	6,33e-04	1,44e-05	12	6,02e-04	2,08e-05	13	6,33e-04	1,44e-05	
$w_{14,2}$	3	1,68e-04	8,39e-06	7	2,11e-04	5,87e-06	12	1,80e-04	8,59e-06	13	2,11e-04	5,87e-06	
$w_{15,1}$	3	1,68e-04	8,39e-06	7	2,11e-04	5,87e-06	12	1,80e-04	8,59e-06	13	2,11e-04	5,87e-06	
$w_{15,2}$	3	3,51e-05	8,77e-06	7	7,87e-05	3,93e-06	12	4,77e-05	9,53e-06	13	7,87e-05	3,93e-06	
$w_{15,3}$	3	0	–	7	4,36e-05	2,72e-06	12	1,26e-05	1,26e-05	13	4,36e-05	2,72e-06	
$w_{16,1}$	2	0	–	7	4,13e-05	2,75e-06	12	9,74e-05	3,04e-06	13	4,13e-05	2,75e-06	
$w_{16,2}$	2	4,14e-05	2,76e-06	7	0	–	12	5,61e-05	3,30e-06	13	0	–	
$w_{17,1}$	3	0	–	7	4,36e-05	2,72e-06	12	9,96e-05	3,02e-06	13	4,36e-05	2,72e-06	
$w_{17,2}$	2	4,14e-05	2,76e-06	7	0	–	12	5,61e-05	3,30e-06	13	0	–	

$w_{18,1}$	5	0	–	8	5,82e-04	3,64e-05	12	1,77e-03	2,90e-05	14	5,82e-04	3,64e-05
$w_{18,2}$	5	5,82e-04	3,64e-05	8	0	–	12	1,18e-03	2,63e-05	14	0	–
$w_{19,1}$	4	0	–	9	4,61e-08	5,12e-09	11	8,39e-04	2,21e-05	15	4,61e-08	5,12e-09
$w_{19,2}$	4	4,61e-08	5,12e-09	9	0	–	11	8,39e-04	2,89e-05	15	0	–
$w_{20,1}$	2	4,14e-05	2,76e-06	7	0	–	12	8,23e-05	2,57e-06	13	0	–
$w_{20,2}$	2	0	–	7	4,13e-05	2,75e-06	12	4,09e-05	2,41e-06	13	4,13e-05	2,75e-06
$w_{21,1}$	3	4,35e-05	2,72e-06	7	0	–	12	5,61e-05	3,30e-06	13	0	–
$w_{21,2}$	3	0	–	7	4,36e-05	2,72e-06	12	1,26e-05	1,26e-05	13	4,36e-05	2,72e-06
$w_{22,1}$	5	5,82e-04	3,64e-05	8	0	–	12	1,18e-03	2,63e-05	14	0	–
$w_{22,2}$	5	0	–	8	5,82e-04	3,64e-05	12	6,02e-04	2,08e-05	14	5,82e-04	3,64e-05
$w_{23,1}$	4	4,61e-08	5,12e-09	9	0	–	11	8,39e-04	2,89e-05	15	0	–
$w_{23,2}$	4	0	–	9	4,61e-08	5,12e-09	11	8,39e-04	4,19e-05	15	4,61e-08	5,12e-09

Table 7.12: Goal reachability analysis results for operational condition **OC5**.

Node	Goal type											
	Loiter			Regain signal			Land			Flight termation		
	Goal ID	$\bar{c}$ $[\mathrm{h}^{-1}\mathrm{NM}]$	$\bar{\lambda}$ $[\mathrm{h}^{-1}]$	Goal ID	$\bar{c}$ $[\mathrm{h}^{-1}\mathrm{NM}]$	$\bar{\lambda}$ $[\mathrm{h}^{-1}]$	Goal ID	$\bar{c}$ $[\mathrm{h}^{-1}\mathrm{NM}]$	$\bar{\lambda}$ $[\mathrm{h}^{-1}]$	Goal ID	$\bar{c}$ $[\mathrm{h}^{-1}\mathrm{NM}]$	$\bar{\lambda}$ $[\mathrm{h}^{-1}]$
$w_{1,1}$	1	1,18e-04	3,93e-06	7	2,00e-04	3,45e-06	12	1,99e-04	3,32e-06	13	2,00e-04	3,45e-06
$w_{1,2}$	1	6,90e-05	2,56e-06	7	1,51e-04	2,75e-06	12	1,51e-04	2,64e-06	13	1,51e-04	2,75e-06
$w_{1,3}$	1	0	–	7	8,21e-05	2,93e-06	12	8,15e-05	2,72e-06	13	8,21e-05	2,93e-06
$w_{2,1}$	1	0	–	7	8,21e-05	2,93e-06	12	8,15e-05	2,72e-06	13	8,21e-05	2,93e-06
$w_{2,2}$	2	0	–	7	6,50e-05	4,33e-06	12	6,44e-05	3,79e-06	13	6,50e-05	4,33e-06
$w_{2,3}$	3	0	–	7	6,85e-05	4,28e-06	12	1,98e-05	1,98e-05	13	6,85e-05	4,28e-06
$w_{2,4}$	3	5,52e-05	1,38e-05	7	1,24e-04	6,19e-06	12	7,50e-05	1,50e-05	13	1,24e-04	6,19e-06
$w_{2,5}$	3	2,64e-04	1,32e-05	7	3,33e-04	9,24e-06	12	2,84e-04	1,35e-05	13	3,33e-04	9,24e-06
$w_{3,1}$	5	6,63e-04	8,29e-05	8	1,58e-03	6,58e-05	12	1,61e-03	4,35e-05	14	1,58e-03	6,58e-05
$w_{3,2}$	5	0	–	8	9,16e-04	5,73e-05	12	9,47e-04	3,27e-05	14	9,16e-04	5,73e-05
$w_{4,1}$	5	0	–	8	3,70e-03	7,70e-05	12	3,73e-03	6,11e-05	14	3,70e-03	7,70e-05
$w_{4,2}$	5	1,87e-03	9,34e-04	8	2,78e-03	1,55e-04	12	2,81e-03	9,08e-05	14	2,78e-03	1,55e-04
$w_{4,3}$	5	0	–	8	9,16e-04	5,73e-05	12	9,47e-04	3,27e-05	14	9,16e-04	5,73e-05
$w_{5,1}$	5	0	–	9	1,04e-02	2,04e-04	11	1,17e-02	1,89e-04	15	1,04e-02	2,04e-04
$w_{5,2}$	4	4,19e-03	1,40e-04	9	4,19e-03	1,07e-04	11	5,51e-03	1,10e-04	15	4,19e-03	1,07e-04
$w_{6,1}$	4	4,19e-03	1,40e-04	9	4,19e-03	1,07e-04	11	5,51e-03	1,10e-04	15	4,19e-03	1,07e-04
$w_{6,2}$	4	0	–	9	7,29e-08	8,10e-09	11	1,32e-03	6,60e-05	15	7,29e-08	8,10e-09
$w_{7,1}$	4	0	–	–	–	–	11	1,32e-03	6,60e-05	–	–	–
$w_{7,2}$	6	1,35e-04	3,36e-05	–	–	–	11	1,35e-04	3,36e-05	–	–	–
$w_{7,3}$	6	0	–	–	–	–	11	0	–	–	–	–
$w_{8,1}$	1	2,24e-04	6,22e-06	7	3,06e-04	4,78e-06	12	3,05e-04	4,63e-06	13	3,06e-04	4,78e-06

$w_{8,2}$	1	1,75e-04	5,30e-06	7	2,57e-04	4,22e-06	12	2,57e-04	4,07e-06	13	2,57e-04	4,22e-06
$w_{8,3}$	1	0	–	7	8,21e-05	2,93e-06	12	8,15e-05	2,72e-06	13	8,21e-05	2,93e-06
$w_{9,1}$	4	0	–	–	–	–	–	–	–	–	–	–
$w_{9,2}$	–	–	–	–	–	–	–	–	–	–	–	–
$w_{9,3}$	–	–	–	–	–	–	–	–	–	–	–	–
$w_{10,1}$	1	0	–	–	–	–	–	–	–	–	–	–
$w_{10,2}$	–	–	–	–	–	–	–	–	–	–	–	–
$w_{10,3}$	–	–	–	–	–	–	–	–	–	–	–	–
$w_{11,1}$	1	0	–	–	–	–	10	1,81e-04	4,51e-06	–	–	–
$w_{11,2}$	–	–	–	–	–	–	10	1,27e-04	8,47e-06	–	–	–
$w_{11,3}$	–	–	–	–	–	–	10	6,36e-06	2,12e-06	–	–	–
$w_{11,4}$	–	–	–	–	–	–	10	0	–	–	–	–
$w_{12,1}$	3	0	–	–	–	–	12	1,98e-05	1,98e-05	–	–	–
$w_{12,2}$	–	–	–	–	–	–	12	0	–	–	–	–
$w_{13,1}$	3	0	–	–	–	–	12	1,98e-05	1,98e-05	–	–	–
$w_{13,2}$	–	–	–	–	–	–	12	0	–	–	–	–
$w_{13,3}$	–	–	–	–	–	–	–	–	–	–	–	–
$w_{13,4}$	–	–	–	–	–	–	–	–	–	–	–	–
$w_{13,5}$	–	–	–	–	–	–	–	–	–	–	–	–
$w_{14,1}$	5	0	–	7	9,96e-04	2,26e-05	12	9,47e-04	3,27e-05	13	9,96e-04	2,26e-05
$w_{14,2}$	3	2,64e-04	1,32e-05	7	3,33e-04	9,24e-06	12	2,84e-04	1,35e-05	13	3,33e-04	9,24e-06
$w_{15,1}$	3	2,64e-04	1,32e-05	7	3,33e-04	9,24e-06	12	2,84e-04	1,35e-05	13	3,33e-04	9,24e-06
$w_{15,2}$	3	5,52e-05	1,38e-05	7	1,24e-04	6,19e-06	12	7,50e-05	1,50e-05	13	1,24e-04	6,19e-06
$w_{15,3}$	3	0	–	7	6,85e-05	4,28e-06	12	1,98e-05	1,98e-05	13	6,85e-05	4,28e-06
$w_{16,1}$	2	0	–	7	6,50e-05	4,33e-06	12	1,53e-04	4,79e-06	13	6,50e-05	4,33e-06
$w_{16,2}$	2	6,51e-05	4,34e-06	7	0	–	12	8,82e-05	5,19e-06	13	0	–
$w_{17,1}$	3	0	–	7	6,85e-05	4,28e-06	12	1,57e-04	4,75e-06	13	6,85e-05	4,28e-06
$w_{17,2}$	2	6,51e-05	4,34e-06	7	0	–	12	8,82e-05	5,19e-06	13	0	–

$w_{18,1}$	5	0	–	8	9,16e-04	5,73e-05	12	2,78e-03	4,56e-05	14	9,16e-04	5,73e-05
$w_{18,2}$	5	9,16e-04	5,73e-05	8	0	–	12	1,86e-03	4,14e-05	14	0	–
$w_{19,1}$	4	0	–	9	7,29e-08	8,10e-09	11	1,32e-03	3,47e-05	15	7,29e-08	8,10e-09
$w_{19,2}$	4	7,29e-08	8,10e-09	9	0	–	11	1,32e-03	4,55e-05	15	0	–
$w_{20,1}$	2	6,51e-05	4,34e-06	7	0	–	12	1,29e-04	4,05e-06	13	0	–
$w_{20,2}$	2	0	–	7	6,50e-05	4,33e-06	12	6,44e-05	3,79e-06	13	6,50e-05	4,33e-06
$w_{21,1}$	3	6,85e-05	4,28e-06	7	0	–	12	8,82e-05	5,19e-06	13	0	–
$w_{21,2}$	3	0	–	7	6,85e-05	4,28e-06	12	1,98e-05	1,98e-05	13	6,85e-05	4,28e-06
$w_{22,1}$	5	9,16e-04	5,73e-05	8	0	–	12	1,86e-03	4,14e-05	14	0	–
$w_{22,2}$	5	0	–	8	9,16e-04	5,73e-05	12	9,47e-04	3,27e-05	14	9,16e-04	5,73e-05
$w_{23,1}$	4	7,29e-08	8,10e-09	9	0	–	11	1,32e-03	4,55e-05	15	0	–
$w_{23,2}$	4	0	–	9	7,29e-08	8,10e-09	11	1,32e-03	6,60e-05	15	7,29e-08	8,10e-09

Table 7.13: Goal reachability analysis results for operational condition **OC6**.

Node	Goal type											
	Loiter			Regain signal			Land			Flight termation		
	Goal ID	$\bar{c}$ $[h^{-1}NM]$	$\bar{\lambda}$ $[h^{-1}]$	Goal ID	$\bar{c}$ $[h^{-1}NM]$	$\bar{\lambda}$ $[h^{-1}]$	Goal ID	$\bar{c}$ $[h^{-1}NM]$	$\bar{\lambda}$ $[h^{-1}]$	Goal ID	$\bar{c}$ $[h^{-1}NM]$	$\bar{\lambda}$ $[h^{-1}]$
$w_{1,1}$	1	1,31e-04	4,38e-06	7	2,23e-04	3,84e-06	12	2,22e-04	3,70e-06	13	2,23e-04	3,84e-06
$w_{1,2}$	1	7,69e-05	2,85e-06	7	1,68e-04	3,06e-06	12	1,67e-04	2,94e-06	13	1,68e-04	3,06e-06
$w_{1,3}$	1	0	–	7	9,14e-05	3,27e-06	12	9,05e-05	3,02e-06	13	9,14e-05	3,27e-06
$w_{2,1}$	1	0	–	7	9,14e-05	3,27e-06	12	9,05e-05	3,02e-06	13	9,14e-05	3,27e-06
$w_{2,2}$	2	0	–	7	7,24e-05	4,83e-06	12	7,15e-05	4,21e-06	13	7,24e-05	4,83e-06
$w_{2,3}$	3	0	–	7	7,63e-05	4,77e-06	12	2,18e-05	2,18e-05	13	7,63e-05	4,77e-06
$w_{2,4}$	3	6,15e-05	1,54e-05	7	1,38e-04	6,89e-06	12	8,33e-05	1,67e-05	13	1,38e-04	6,89e-06
$w_{2,5}$	3	2,94e-04	1,47e-05	7	3,71e-04	1,03e-05	12	3,16e-04	1,51e-05	13	3,71e-04	1,03e-05
$w_{3,1}$	5	7,39e-04	9,24e-05	8	1,76e-03	7,34e-05	12	1,80e-03	4,85e-05	14	1,76e-03	7,34e-05
$w_{3,2}$	5	0	–	8	1,02e-03	6,38e-05	12	1,06e-03	3,64e-05	14	1,02e-03	6,38e-05
$w_{4,1}$	5	0	–	8	4,10e-03	8,54e-05	12	4,13e-03	6,77e-05	14	4,10e-03	8,54e-05
$w_{4,2}$	5	2,06e-03	1,03e-03	8	3,08e-03	1,71e-04	12	3,12e-03	1,01e-04	14	3,08e-03	1,71e-04
$w_{4,3}$	5	0	–	8	1,02e-03	6,38e-05	12	1,06e-03	3,64e-05	14	1,02e-03	6,38e-05
$w_{5,1}$	5	0	–	9	1,16e-02	2,27e-04	11	1,31e-02	2,11e-04	15	1,16e-02	2,27e-04
$w_{5,2}$	4	4,67e-03	1,56e-04	9	4,67e-03	1,20e-04	11	6,14e-03	1,23e-04	15	4,67e-03	1,20e-04
$w_{6,1}$	4	4,67e-03	1,56e-04	9	4,67e-03	1,20e-04	11	6,14e-03	1,23e-04	15	4,67e-03	1,20e-04
$w_{6,2}$	4	0	–	9	5,57e-08	6,19e-09	11	1,47e-03	7,35e-05	15	5,57e-08	6,19e-09
$w_{7,1}$	4	0	–	–	–	–	11	1,47e-03	7,35e-05	–	–	–
$w_{7,2}$	6	1,50e-04	3,75e-05	–	–	–	11	1,50e-04	3,75e-05	–	–	–
$w_{7,3}$	6	0	–	–	–	–	11	0	–	–	–	–
$w_{8,1}$	1	2,50e-04	6,93e-06	7	3,41e-04	5,33e-06	12	3,40e-04	5,15e-06	13	3,41e-04	5,33e-06

$w_{8,2}$	1	1,95e-04	5,91e-06	7	2,86e-04	4,70e-06	12	2,86e-04	4,53e-06	13	2,86e-04	4,70e-06	
$w_{8,3}$	1	0	–	7	9,14e-05	3,27e-06	12	9,05e-05	3,02e-06	13	9,14e-05	3,27e-06	
$w_{9,1}$	4	0	–	–	–	–	–	–	–	–	–	–	
$w_{9,2}$	–	–	–	–	–	–	–	–	–	–	–	–	
$w_{9,3}$	–	–	–	–	–	–	–	–	–	–	–	–	
$w_{10,1}$	1	0	–	–	–	–	–	–	–	–	–	–	
$w_{10,2}$	–	–	–	–	–	–	–	–	–	–	–	–	
$w_{10,3}$	–	–	–	–	–	–	–	–	–	–	–	–	
$w_{11,1}$	1	0	–	–	–	–	10	1,99e-04	4,98e-06	–	–	–	
$w_{11,2}$	–	–	–	–	–	–	10	1,40e-04	9,35e-06	–	–	–	
$w_{11,3}$	–	–	–	–	–	–	10	7,01e-06	2,34e-06	–	–	–	
$w_{11,4}$	–	–	–	–	–	–	10	0	–	–	–	–	
$w_{12,1}$	3	0	–	–	–	–	12	2,18e-05	2,18e-05	–	–	–	
$w_{12,2}$	–	–	–	–	–	–	12	0	–	–	–	–	
$w_{13,1}$	3	0	–	–	–	–	12	2,18e-05	2,18e-05	–	–	–	
$w_{13,2}$	–	–	–	–	–	–	12	0	–	–	–	–	
$w_{13,3}$	–	–	–	–	–	–	–	–	–	–	–	–	
$w_{13,4}$	–	–	–	–	–	–	–	–	–	–	–	–	
$w_{13,5}$	–	–	–	–	–	–	–	–	–	–	–	–	
$w_{14,1}$	5	0	–	7	1,11e-03	2,52e-05	12	1,06e-03	3,64e-05	13	1,11e-03	2,52e-05	
$w_{14,2}$	3	2,94e-04	1,47e-05	7	3,71e-04	1,03e-05	12	3,16e-04	1,51e-05	13	3,71e-04	1,03e-05	
$w_{15,1}$	3	2,94e-04	1,47e-05	7	3,71e-04	1,03e-05	12	3,16e-04	1,51e-05	13	3,71e-04	1,03e-05	
$w_{15,2}$	3	6,15e-05	1,54e-05	7	1,38e-04	6,89e-06	12	8,33e-05	1,67e-05	13	1,38e-04	6,89e-06	
$w_{15,3}$	3	0	–	7	7,63e-05	4,77e-06	12	2,18e-05	2,18e-05	13	7,63e-05	4,77e-06	
$w_{16,1}$	2	0	–	7	7,24e-05	4,83e-06	12	1,71e-04	5,33e-06	13	7,24e-05	4,83e-06	
$w_{16,2}$	2	7,25e-05	4,83e-06	7	0	–	12	9,81e-05	5,77e-06	13	0	–	
$w_{17,1}$	3	0	–	7	7,63e-05	4,77e-06	12	1,74e-04	5,29e-06	13	7,63e-05	4,77e-06	
$w_{17,2}$	2	7,25e-05	4,83e-06	7	0	–	12	9,81e-05	5,77e-06	13	0	–	

$w_{18,1}$	5	0	–	8	1,02e-03	6,38e-05	12	3,10e-03	5,08e-05	14	1,02e-03	6,38e-05
$w_{18,2}$	5	1,02e-03	6,38e-05	8	0	–	12	2,08e-03	4,62e-05	14	0	–
$w_{19,1}$	4	0	–	9	5,57e-08	6,19e-09	11	1,47e-03	3,87e-05	15	5,57e-08	6,19e-09
$w_{19,2}$	4	5,57e-08	6,19e-09	9	0	–	11	1,47e-03	5,07e-05	15	0	–
$w_{20,1}$	2	7,25e-05	4,83e-06	7	0	–	12	1,44e-04	4,50e-06	13	0	–
$w_{20,2}$	2	0	–	7	7,24e-05	4,83e-06	12	7,15e-05	4,21e-06	13	7,24e-05	4,83e-06
$w_{21,1}$	3	7,63e-05	4,77e-06	7	0	–	12	9,81e-05	5,77e-06	13	0	–
$w_{21,2}$	3	0	–	7	7,63e-05	4,77e-06	12	2,18e-05	2,18e-05	13	7,63e-05	4,77e-06
$w_{22,1}$	5	1,02e-03	6,38e-05	8	0	–	12	2,08e-03	4,62e-05	14	0	–
$w_{22,2}$	5	0	–	8	1,02e-03	6,38e-05	12	1,06e-03	3,64e-05	14	1,02e-03	6,38e-05
$w_{23,1}$	4	5,57e-08	6,19e-09	9	0	–	11	1,47e-03	5,07e-05	15	0	–
$w_{23,2}$	4	0	–	9	5,57e-08	6,19e-09	11	1,47e-03	7,35e-05	15	5,57e-08	6,19e-09

As an example, assume that no contingency evidence is observed (so Table 7.11 applies) and that the RPAS is flying the ingress segment s_3 towards waypoint F15B2 (corresponding to node $w_{3,2}$ in the Mission Graph). In this state, the optimal "land" goal in terms of risk is the mission goal with ID equal to 12 (a "land" goal associated with airdrome LERE). The route that connects waypoint F15B2 with the arrival procedure to LERE has a cumulative risk $\bar{c} = 6,02 \cdot 10^{-4}$ h^{-1}NM, and an average risk $\bar{\lambda} = 2,08 \cdot 10^{-5}$ h^{-1}. Note that this goal has been identified as the optimal "land" goal because the route towards the other possible aerodromes (LETL and LECH, see Fig. 7.1) have a greater cumulative risk.

In the same way, the most convenient "flight termination" goal from this node is the mission goal with ID equal to 14 (a "flight termination" goal associated with the termination point FTP2). The route that connects waypoint F15B2 with FTP2 has a cumulative risk $\bar{c} = 5,82 \cdot 10^{-4}$ h^{-1}NM, and an average risk $\bar{\lambda} = 3,64 \cdot 10^{-5}$ h^{-1}. Finally, the optimal "loiter" goal from the same node $w_{3,2}$ is the mission goal with ID equal to 5, a mission goal associated with waypoint F15B2. Since the RPAS is currently flying towards these waypoint, the route to achieve this mission goal from node $w_{3,2}$ has a zero cost.

An important remark regarding the results in these tables is that not all goal types can be achieved from every node in the Mission Graph. As an example, consider the node $w_{7,2}$. It is associated with waypoint TATOS, a waypoint flown during the arrival procedure to LECH runway 06. From this node, the only reachable goal types are "loiter" goals and "land" goals, see Tables 7.11, 7.12 or 7.13. In this case, the "loiter" goal is associated with the arrival procedure's holding pattern, and the "land" goal is associated with the final approach procedure to LECH. However, "flight termination" goals cannot be reached from this node because there is not route connecting the arrival procedure with a flight termination point. Table 7.14 shows the number of nodes in the Mission Graph where each goal type is currently achievable.

Table 7.14: Nodes of the Mission Graph where each goal type is achievable.

Goal type	Nodes where goal type is achievable [%]
Loiter	80,3
Regain signal	67,2
Land	85,2
Flight termination	67,2

Table 7.15: Overall results of the goal reachability analysis for operational condition **OC4**.

Goal type	Average $\bar{c}$ $[\mathrm{h}^{-1}\mathrm{NM}]$	Average $\bar{\lambda}$ $[\mathrm{h}^{-1}]$	Worst case node
Loiter	1,90e-04	2,01e-05	$w_{5,2}$
Regain signal	5,33e-04	1,65e-05	$w_{5,1}$
Land	6,43e-04	1,79e-05	$w_{5,1}$
Flight termination	5,33e-04	1,65e-05	$w_{5,1}$

Table 7.16: Overall results of the goal reachability analysis for operational condition **OC5**.

Goal type	Average $\bar{c}$ $[\mathrm{h}^{-1}\mathrm{NM}]$	Average $\bar{\lambda}$ $[\mathrm{h}^{-1}]$	Worst case node
Loiter	2,98e-04	3,16e-05	$w_{5,2}$
Regain signal	8,38e-04	2,59e-05	$w_{5,1}$
Land	1,01e-03	2,82e-05	$w_{5,1}$
Flight termination	8,38e-04	2,59e-05	$w_{5,1}$

Table 7.17: Overall results of the goal reachability analysis for operational condition **OC6**.

Goal type	Average $\bar{c}$ $[\mathrm{h}^{-1}\mathrm{NM}]$	Average $\bar{\lambda}$ $[\mathrm{h}^{-1}]$	Worst case node
Loiter	3,32e-04	3,50e-05	$w_{5,2}$
Regain signal	9,33e-04	2,88e-05	$w_{5,1}$
Land	1,13e-03	3,13e-05	$w_{5,1}$
Flight termination	9,33e-04	2,88e-05	$w_{5,1}$

Tables 7.15, 7.16 and 7.17 characterize the routes provided by the goal reachability analysis. In particular, these tables show the cumulative risk of the average route to achieve each goal type, the average risk of this average route, and the worst case condition (i.e. the node where the cumulative risk is maximum) for each operational condition under analysis. For example, the cumulative risk of the average route to achieve a "loiter" goal in **OC4** is $\bar{c} = 1,84 \cdot 10^{-4}$ $\mathrm{h}^{-1}\mathrm{NM}$. This number has been computer by averaging the cumulative risks over all the nodes in Table 7.11. In the same way, the average risk of the average route to achieve this goal type is $9,03 \cdot 10^{-6}$ h^{-1}; and the worst case node is node $w_{5,1}$.

Similar results can be extracted from the remaining goal types and operational conditions under analysis.

The previous safety metrics can be used to refine some aspect of a particular mission design. For example, if the average risk posed by routes for achieving the "flight termination" goal is considered to be above some given safety threshold, then new segments or additional flight termination points can be introduced in the mission specification. However, such type of analysis will not be performed in this case study.

Effectiveness of the Contingency Plan

Based on the results of the goal reachability analysis, it is possible to measure the risk reduction achieved when executing the contingency management policy described in the Contingency Plan. Next, we will assess the effectiveness of the Contingency Plan model developed in Appendix A, and implemented onboard the prototype application of Appendix B, specifically for C2 link loss and GNSS loss of performance conditions.

In order to measure the risk reduction achieved when executing these policies, it is necessary to define a risk reference, or *base risk*. In this case, we will define the base risk as the risk posed by the operation when the mission is not replanned after a contingency happens. In particular, the base risk will be the cumulative risk posed by the route that attempts to achieve the nominal goal from a given node of the Mission Graph when operating in a degraded operational condition. For example, the base risk when flying towards node $w_{2,1}$ in a C2 link loss condition is the cumulative risk of the route that attempts to achieve the nominal goal from this node (i.e. basically the nominal route, but removing the departure segment) when operating in **OC5**. Once the base risk is defined, the new operational risk is simply the risk of attempting to achieve the new goal defined by the Contingency Plan, i.e. the cumulative risk posed by the route that achieves this goal from the current node of the Mission Graph.

The results showing the effectiveness of the C2 link loss policy and the GNSS signal loss of performance policy are presented in Tables 7.18 and 7.19, respectively. These tables show, for each node in the Mission Graph, what is the base risk in the corresponding operational condition; what is the contingency management policy proposed by the Contingency Plan, as well as the type of policy (i.e. if it is a strategic goal, a tactical goal, or an operational mode

restriction); and finally the effectiveness of this policy, i.e. the risk reduction with respect to the base risk.

Table 7.18: Effectiveness of the contingency management policy for C2 link loss conditions.

Node ID	Node	Base risk $[h^{-1}NM]$	C2 link loss policy	Policy type	Risk mitigation effectiveness [%]
1	$w_{1,1}$	1,56e-02	Land	Strategic	-98,7
2	$w_{1,2}$	1,56e-02	Land	Strategic	-99,0
3	$w_{1,3}$	1,55e-02	Land	Strategic	-99,5
4	$w_{2,1}$	1,55e-02	Land	Strategic	-99,5
5	$w_{2,2}$	1,55e-02	Land	Strategic	-99,6
6	$w_{2,3}$	1,54e-02	Land	Strategic	-99,9
7	$w_{2,4}$	1,54e-02	Land	Strategic	-99,5
8	$w_{2,5}$	1,52e-02	Land	Strategic	-98,1
9	$w_{3,1}$	1,24e-02	Regain signal	Strategic	-87,2
10	$w_{3,2}$	1,17e-02	Regain signal	Strategic	-92,2
11	$w_{4,1}$	1,45e-02	Regain signal	Strategic	-74,5
12	$w_{4,2}$	1,36e-02	Regain signal	Strategic	-79,5
13	$w_{4,3}$	1,17e-02	Regain signal	Strategic	-92,2
14	$w_{5,1}$	1,17e-02	Regain signal	Strategic	-11,3
15	$w_{5,2}$	5,51e-03	Regain signal	Strategic	-24,0
16	$w_{6,1}$	5,51e-03	Regain signal	Strategic	-24,0
17	$w_{6,2}$	1,32e-03	Regain signal	Strategic	-100,0
18	$w_{7,1}$	1,32e-03	Land	Strategic	0
19	$w_{7,2}$	1,35e-04	Land	Strategic	0
20	$w_{7,3}$	0	Land	Strategic	0
21	$w_{8,1}$	1,29e-02	Land	Strategic	-97,6
22	$w_{8,2}$	1,29e-02	Land	Strategic	-98,0
23	$w_{8,3}$	1,27e-02	Land	Strategic	-99,4
24	$w_{9,1}$	–	Flight termination	Tactical	–
25	$w_{9,2}$	–	Flight termination	Tactical	–
26	$w_{9,3}$	–	Flight termination	Tactical	–
27	$w_{10,1}$	–	Flight termination	Tactical	–
28	$w_{10,2}$	–	Flight termination	Tactical	–
29	$w_{10,3}$	–	Flight termination	Tactical	–
30	$w_{11,1}$	–	Land	Strategic	–
31	$w_{11,2}$	–	Land	Strategic	–
32	$w_{11,3}$	–	Land	Strategic	–

33	$w_{11,4}$	–	Land	Strategic	–
34	$w_{12,1}$	–	Land	Strategic	–
35	$w_{12,2}$	–	Land	Strategic	–
36	$w_{13,1}$	–	Land	Strategic	–
37	$w_{13,2}$	–	Land	Strategic	–
38	$w_{13,3}$	–	Flight termination	Tactical	–
39	$w_{13,4}$	–	Flight termination	Tactical	–
40	$w_{13,5}$	–	Flight termination	Tactical	–
41	$w_{14,1}$	1,30e-02	Land	Strategic	-92,7
42	$w_{14,2}$	1,24e-02	Land	Strategic	-97,7
43	$w_{15,1}$	1,28e-02	Land	Strategic	-97,8
44	$w_{15,2}$	1,26e-02	Land	Strategic	-99,4
45	$w_{15,3}$	1,26e-02	Land	Strategic	-99,8
46	$w_{16,1}$	1,28e-02	Regain signal	Strategic	-99,5
47	$w_{16,2}$	1,27e-02	Regain signal	Strategic	-100,0
48	$w_{17,1}$	1,28e-02	Regain signal	Strategic	-99,5
49	$w_{17,2}$	1,27e-02	Regain signal	Strategic	-100,0
50	$w_{18,1}$	1,36e-02	Regain signal	Strategic	-93,2
51	$w_{18,2}$	1,26e-02	Regain signal	Strategic	-100,0
52	$w_{19,1}$	1,32e-03	Regain signal	Strategic	-100,0
53	$w_{19,2}$	1,32e-03	Regain signal	Strategic	-100,0
54	$w_{20,1}$	1,28e-02	Regain signal	Strategic	-100,0
55	$w_{20,2}$	1,27e-02	Land	Strategic	-99,5
56	$w_{21,1}$	1,27e-02	Regain signal	Strategic	-100,0
57	$w_{21,2}$	1,26e-02	Land	Strategic	-99,8
58	$w_{22,1}$	1,26e-02	Regain signal	Strategic	-100,0
59	$w_{22,2}$	1,17e-02	Regain signal	Strategic	-92,2
60	$w_{23,1}$	1,32e-03	Regain signal	Strategic	-100,0
61	$w_{23,2}$	1,32e-03	Regain signal	Strategic	-100,0

Table 7.19: Effectiveness of the contingency management policy for GNSS loss of performance conditions.

Node ID	Node	Base risk $[h^{-1}NM]$	GNSS signal loss policy	Policy type	Risk mitigation effectiveness [%]
1	$w_{1,1}$	1,74e-02	Loiter	Strategic	-99,2
2	$w_{1,2}$	1,73e-02	Loiter	Strategic	-99,6
3	$w_{1,3}$	1,72e-02	Loiter	Strategic	-100,0

4	$w_{2,1}$	1,72e-02	Loiter	Strategic	-100,0
5	$w_{2,2}$	1,72e-02	Loiter	Strategic	-100,0
6	$w_{2,3}$	1,72e-02	Loiter	Strategic	-100,0
7	$w_{2,4}$	1,71e-02	Loiter	Strategic	-99,6
8	$w_{2,5}$	1,69e-02	Loiter	Strategic	-98,3
9	$w_{3,1}$	1,38e-02	Loiter	Strategic	-94,6
10	$w_{3,2}$	1,31e-02	Loiter	Strategic	-100,0
11	$w_{4,1}$	1,61e-02	Loiter	Strategic	-100,0
12	$w_{4,2}$	1,51e-02	Loiter	Strategic	-86,4
13	$w_{4,3}$	1,31e-02	Loiter	Strategic	-100,0
14	$w_{5,1}$	1,31e-02	Loiter	Strategic	-100,0
15	$w_{5,2}$	6,14e-03	Loiter	Strategic	-24,0
16	$w_{6,1}$	6,14e-03	Loiter	Strategic	-24,0
17	$w_{6,2}$	1,47e-03	Loiter	Strategic	-100,0
18	$w_{7,1}$	1,47e-03	Loiter	Strategic	-100,0
19	$w_{7,2}$	1,50e-04	Loiter	Strategic	0
20	$w_{7,3}$	0	Loiter	Strategic	–
21	$w_{8,1}$	1,44e-02	Loiter	Strategic	-98,3
22	$w_{8,2}$	1,44e-02	Loiter	Strategic	-98,6
23	$w_{8,3}$	1,42e-02	Loiter	Strategic	-100,0
24	$w_{9,1}$	–	Loiter	Strategic	–
25	$w_{9,2}$	–	Revert to manual	Op. mode restriction	–
26	$w_{9,3}$	–	Revert to manual	Op. mode restriction	–
27	$w_{10,1}$	–	Loiter	Strategic	–
28	$w_{10,2}$	–	Revert to manual	Op. mode restriction	–
29	$w_{10,3}$	–	Revert to manual	Op. mode restriction	–
30	$w_{11,1}$	–	Loiter	Strategic	–
31	$w_{11,2}$	–	Revert to manual	Op. mode restriction	–
32	$w_{11,3}$	–	Revert to manual	Op. mode restriction	–
33	$w_{11,4}$	–	Revert to manual	Op. mode restriction	–
34	$w_{12,1}$	–	Loiter	Strategic	–
35	$w_{12,2}$	–	Revert to manual	Op. mode restriction	–
36	$w_{13,1}$	–	Loiter	Strategic	–
37	$w_{13,2}$	–	Revert to manual	Op. mode restriction	–
38	$w_{13,3}$	–	Revert to manual	Op. mode restriction	–
39	$w_{13,4}$	–	Revert to manual	Op. mode restriction	–
40	$w_{13,5}$	–	Revert to manual	Op. mode restriction	–
41	$w_{14,1}$	1,45e-02	Loiter	Strategic	-100,0
42	$w_{14,2}$	1,38e-02	Loiter	Strategic	-97,9

43	$w_{15,1}$	1,43e-02	Loiter	Strategic	-97,9
44	$w_{15,2}$	1,40e-02	Loiter	Strategic	-99,6
45	$w_{15,3}$	1,41e-02	Loiter	Strategic	-100,0
46	$w_{16,1}$	1,42e-02	Loiter	Strategic	-100,0
47	$w_{16,2}$	1,42e-02	Regain signal	Strategic	-100,0
48	$w_{17,1}$	1,43e-02	Loiter	Strategic	-100,0
49	$w_{17,2}$	1,42e-02	Regain signal	Strategic	-100,0
50	$w_{18,1}$	1,51e-02	Loiter	Strategic	-100,0
51	$w_{18,2}$	1,41e-02	Regain signal	Strategic	-100,0
52	$w_{19,1}$	1,47e-03	Loiter	Strategic	-100,0
53	$w_{19,2}$	1,47e-03	Regain signal	Strategic	-100,0
54	$w_{20,1}$	1,42e-02	Regain signal	Strategic	-100,0
55	$w_{20,2}$	1,41e-02	Loiter	Strategic	-100,0
56	$w_{21,1}$	1,42e-02	Regain signal	Strategic	-100,0
57	$w_{21,2}$	1,41e-02	Loiter	Strategic	-100,0
58	$w_{22,1}$	1,41e-02	Regain signal	Strategic	-100,0
59	$w_{22,2}$	1,31e-02	Loiter	Strategic	-100,0
60	$w_{23,1}$	1,47e-03	Regain signal	Strategic	-100,0
61	$w_{23,2}$	1,47e-03	Loiter	Strategic	-100,0

Note that, in some nodes like nodes $w_{9,1}$, $w_{9,2}$, $w_{10,1}$ or $w_{10,2}$, the base risk cannot be estimated because the nominal goal is not reachable from these nodes. When this occurs, the effectiveness of the proposed policy cannot be measured, although an effective risk mitigation level is expected.

In order to illustrate the previous results, we will analyze two representative contingency scenarios, named as **CS1** and **CS2** (which will also be simulated in execution time later on):

- In **CS1**, the RPAS is flying the nominal route, and is currently located in the ingress segment before arriving to the operations area, somewhere between waypoints LASPO and F15B2, see Fig. 7.12. The source node is therefore node $w_{3,1}$. At some point of this leg, the GNSS signal experiences a degradation, so the RPAS starts flying in a degraded navigation mode. According to Table 7.19, the base risk for this condition is $\bar{c}_0 = 1,38 \cdot 10^{-2}$ h^{-1}NM (the risk posed by the route represented using a blue dotted line in the previous figure). According to the previous table, the proposed contingency management policy for this condition is to replan the mission and perform a strategic "loiter". As it can be extracted from Table 7.13, the optimal "loiter" goal from this node is the

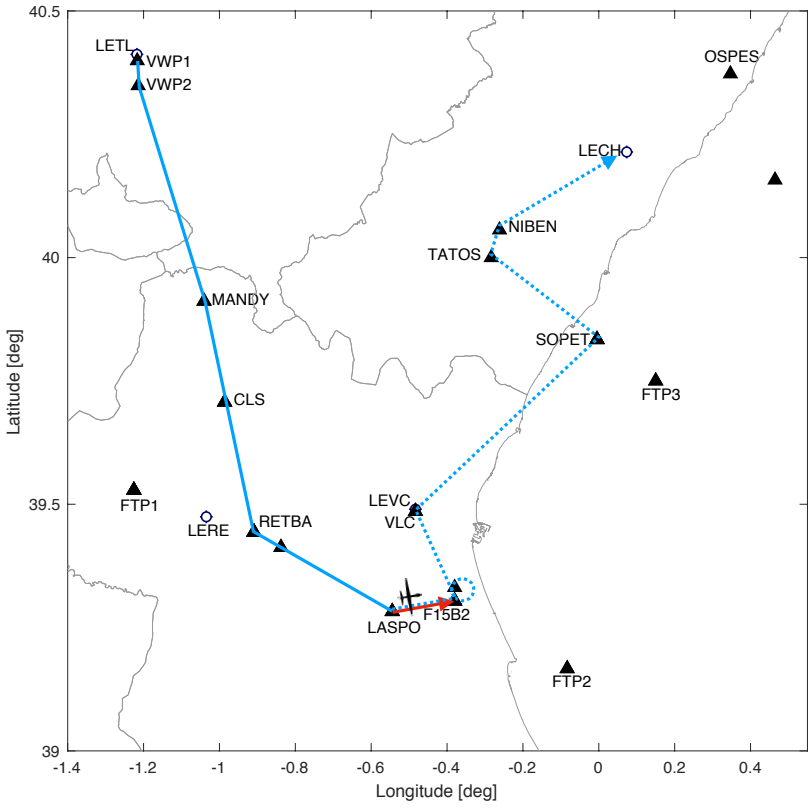


Figure 7.12: Dynamic route configuration in the contingency scenario **CS1**. Note that the blue dotted line represents the static (nominal) route, while the red solid line is the dynamic (contingency) route.

mission goal ID equal to 5, a mission goal associated to waypoint F15B2. The dynamic route that achieves this goal is described by the following sequence of nodes, represented in Fig. 7.12 using a red solid line:

$$r_{\text{CS1}} = \langle w_{3,1} \rightarrow w_{3,2} \rangle \tag{7.2}$$

As a result, if the remote pilot executes the proposed contingency option, the operational risk will be reduced by $94,6\%$ with respect to the base risk, see Table 7.19.

Table 7.20: Overall results of the contingency management policy for C2 link loss conditions.

Policy	Nodes where policy is triggered [%]
Regain signal	37,8
Land	47,5
Flight termination	14,7

- In **CS2**, the RPAS is flying the nominal route, it has already completed the payload-related task within the operations area, and is currently flying towards the destination site, somewhere between waypoints VLC and SOPET, in segment s_6, see Fig. 7.13. The source node is therefore node $w_{6,1}$. At some point of this leg, the C2 link experiences a degradation and is declared as being lost, so the mission should be replanned. According to Table 7.18, the base risk for this condition is $\bar{c}_0 = 5,51 \cdot 10^{-3}$ h^{-1}NM (the risk posed by the route represented using a blue dotted line in the previous figure). In this condition, the proposed contingency management policy is to attempt to regain the C2 link signal. As it can be extracted from Table 7.12, the optimal "regain signal" goal from the current node is the mission goal ID equal to 9, a mission goal associated to waypoint FTP3. The dynamic route that achieves this goal is described by the following sequence of nodes, represented using a red solid line:

$$r_{\text{CS2}} = \langle w_{6,1} \rightarrow w_{6,2} \rightarrow w_{19,1} \rightarrow w_{19,2} \rangle \tag{7.3}$$

  As a result, if the suggested option is performed, the operational risk is expected to be reduced by $24,0\%$ with respect to the base risk, see Table 7.18.

Finally, the overall results of the proposed contingency management policies are gathered in Tables 7.20 and 7.21. In particular, these tables show the number of times each possible contingency option is triggered in each case. As an example, after the loss of the C2 link, "regain signal" is the optimal policy at $37,8\%$ of the nodes of the Mission Graph; "land" is the optimal option at $47,5\%$ of the nodes; and "flight termination" at $14,7\%$. Note, however, that nodes where "flight termination" is the optimal option are nodes associated with arrival segments steering towards inactive runway thresholds. As a result, it is possible to affirm that, in practice, the proposed policy reduces the probability of performing the "flight termination" action to zero.

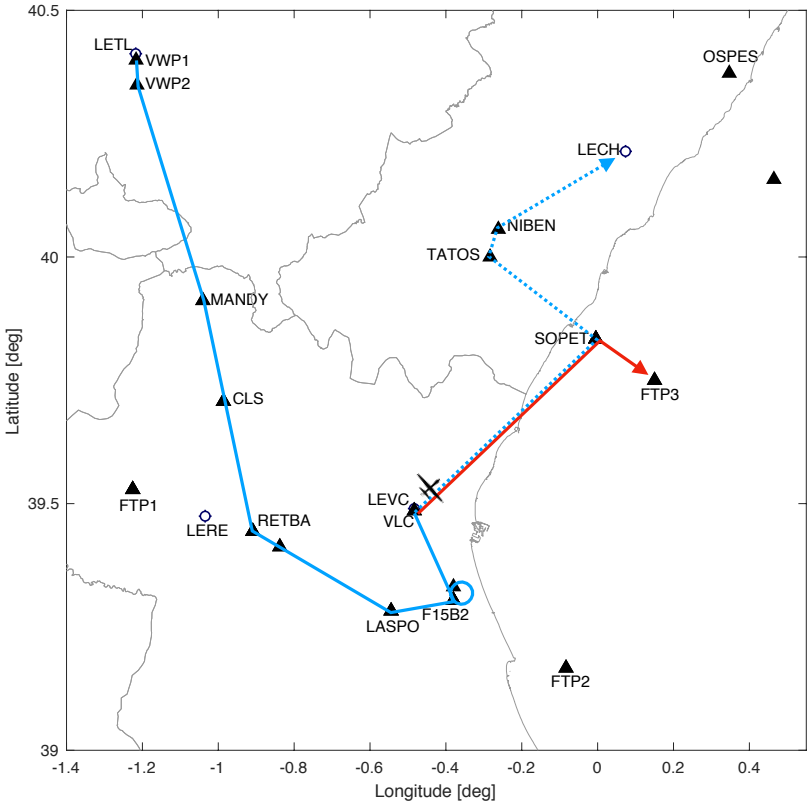


Figure 7.13: Dynamic route configuration in the contingency scenario **CS2**. Note that the blue dotted line represents the static (nominal) route, while the red solid line is the dynamic (contingency) route.

Table 7.21: Overall results of the contingency management policy for GNSS signal loss conditions.

Policy	Nodes where policy is triggered [%]
Loiter	67,2
Regain signal	13,1
Revert to manual	19,7

7.3 Flight simulation results

Once the mission risk assessment for the demonstration mission is performed, we will simulate the execution of this mission using the SMMS prototype described in Appendix B running in the simulation environment described in the same appendix. Firstly, we will simulate the mission execution assuming that the RPAS is operating in a nominal condition during the entire flight. This will exhibit the normal functioning of the prototype. Afterwards, we will exploit the contingency injection mechanism in Fig. B.3b to simulate the two contingency scenarios that were introduced above (**CS1** and **CS2**). This will illustrate the mission reconfiguration concept and the ACM capability of the SMMS prototype in simulation time.

In this case, the three simulations will be conducted under the following conditions:

- The IAI Super Heron model will be used as a demonstration UAS.
- The mission will be performed under visual meteorological conditions and light wind.
- We will assume that the different separation mechanisms are effective at separating traffics in the vicinity of the RPAS, so no loss of separation condition will be simulated.

In addition, the departure segment from airport LETL and the final approach to LECH will not be simulated for brevity.

7.3.1 Nominal operation

The flight simulation results of the demonstration mission performed under a nominal condition are gathered in Figures 7.14 to 7.24. In particular, Fig. 7.14 shows the contingency injection signal sent to the SMMS application. As it can be observed, no contingency is injected in this scenario, so the mission is expected to be conducted in a nominal condition.

Apart from the manual contingency injection mechanism, the Safety Monitor component can still detect abnormal states autonomously based on the system state and the simulation data. In this case, Fig. 7.15 shows the active state of the centralized Safety Monitor component is always the nominal operation state; so no abnormal conditions were detected during the mission execution.

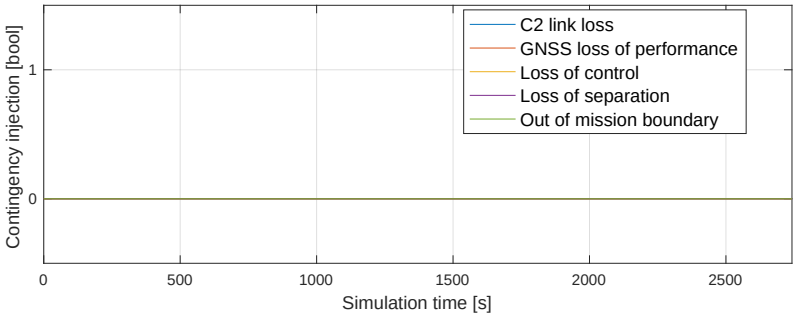


Figure 7.14: Flight simulation results, nominal scenario: Contingency injection.

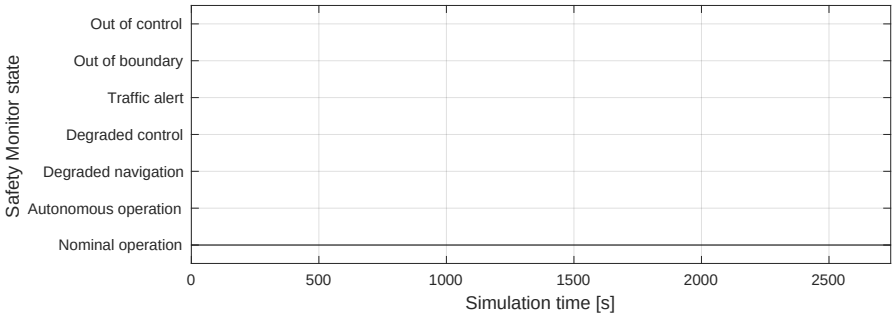


Figure 7.15: Flight simulation results, nominal scenario: Safety Monitor state.

Then, Fig. 7.16 shows the strategic goal selection performed during this mission. In particular, it compares which of the mission goals of the Reconfigurable Mission Plan is suggested by the Contingency Plan at each time (signal "Contingency Plan advisory" of Fig. 7.16); which of the mission goals is manually selected by the remote pilot through the mission control panel of Fig. B.3a (signal "Strategic goal armed"); and which of these is being processed by the Mission Manager component ("Strategic goal engaged"). As it can be observed, the Contingency Plan always suggests the nominal goal (which is the mission goal number 16, see Fig. 7.7) because no abnormal states were detected by the Safety Monitor. In addition, the remote pilot has not manually selected a mission goal other than the nominal goal. Therefore, the strategic goal processed by the Mission Manager component is also the nominal goal.

In the proposed SMMS implementation, apart from the previous goal selection, the remote pilot also has to select the mission goal that will be engaged in case

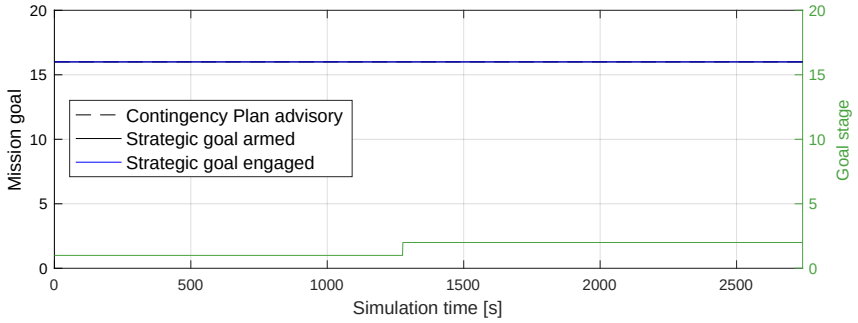


Figure 7.16: Flight simulation results, nominal scenario: Mission goal selection.

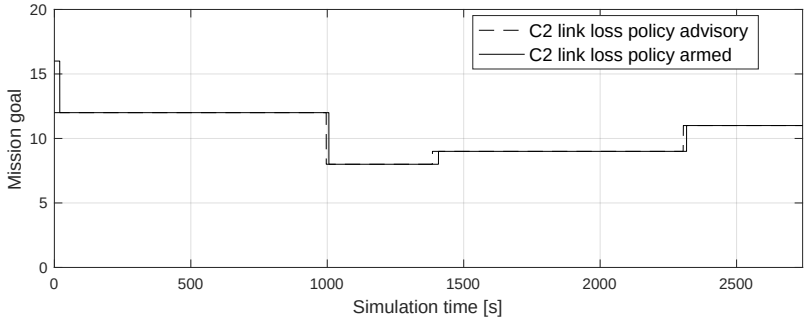


Figure 7.17: Flight simulation results, nominal scenario: C2 link loss policy selection.

that the C2 link is lost. This goal is called the *C2 link loss policy armed* in the SMMS. In any case, the Contingency Manager suggests the remote pilot which of the mission goals of the Reconfigurable Mission Plan is the most convenient C2 link loss policy at each time of the mission execution (based on the directives of the Contingency Plan). The remote pilot can select the suggested policy or any other option at will using the control panel of Fig. B.3a (although the selected option is to be agreed with the corresponding ATC unit).

Fig. 7.17 compares the C2 link loss policy suggested by the Contingency Manager (signal "C2 link loss policy advisory") with the C2 link loss policy actually armed by the remote pilot. Fig. 7.18 shows a screenshot of the Diagnostic Viewer in Simulink where the Contingency Manager provides the remote pilot a corresponding advisory. As it can be observed in Fig. 7.17, the remote pilot always follows the Contingency Plan directives. Note that there is a certain time gap between the time the Contingency Manager suggests a given policy

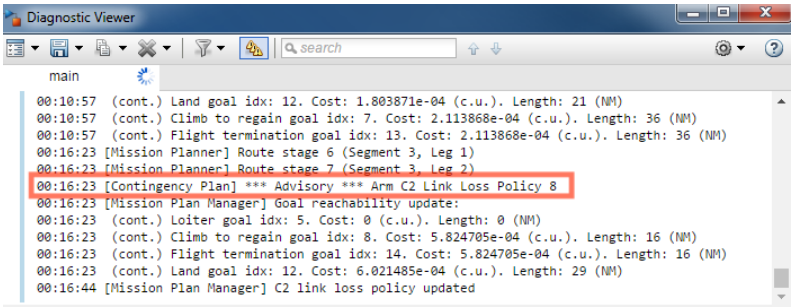


Figure 7.18: Detail of the Diagnostic Viewer window in Simulink showing a Contingency Plan advisory.

and the time the pilot accepts it. This gap depends on the pilot alertness and reaction time. Although these aspects are safety-critical, a further discussion on human factors exceeds the scope of this work. It is also interesting to analyze the C2 policy advisory provided by the Contingency Plan. As it can be observed, it varies between mission goals 12, 8, 9 and 11, depending on the RPAS position in the nominal route[3]:

1. The initial C2 link loss policy advisory is the mission goal equal to 12 (a "land" goal associated with the alternate airdrome LERE). It is the optimal option until the RPAS reaches waypoint LASPO in segment s_2 (time $t = 983$ s).

2. Then, the C2 link loss policy advisory switches to the mission goal number 8 (a "climb to regain signal" goal associated with waypoint FTP2). It is the optimal option until the RPAS leaves the operations area through waypoint F15B2 (time $t = 1\,385$ s).

3. Afterwards, the C2 link loss policy advisory becomes the mission goal number 9 (a "climb to regain signal" goal associated with waypoint FTP3). It is the optimal option until the RPAS reaches waypoint SOPET in segment s_6 (time $t = 2\,305$ s).

4. Once the RPAS starts the arrival segment in waypoint SOPET, the C2 link loss policy advisory is the mission goal equal to 11 (a "land" goal associated with the main destination site LECH). It remains as the optimal policy until the end to the mission execution.

[3]Note that the following results are consistent with the C2 link loss policy analysis presented in Table 7.18.

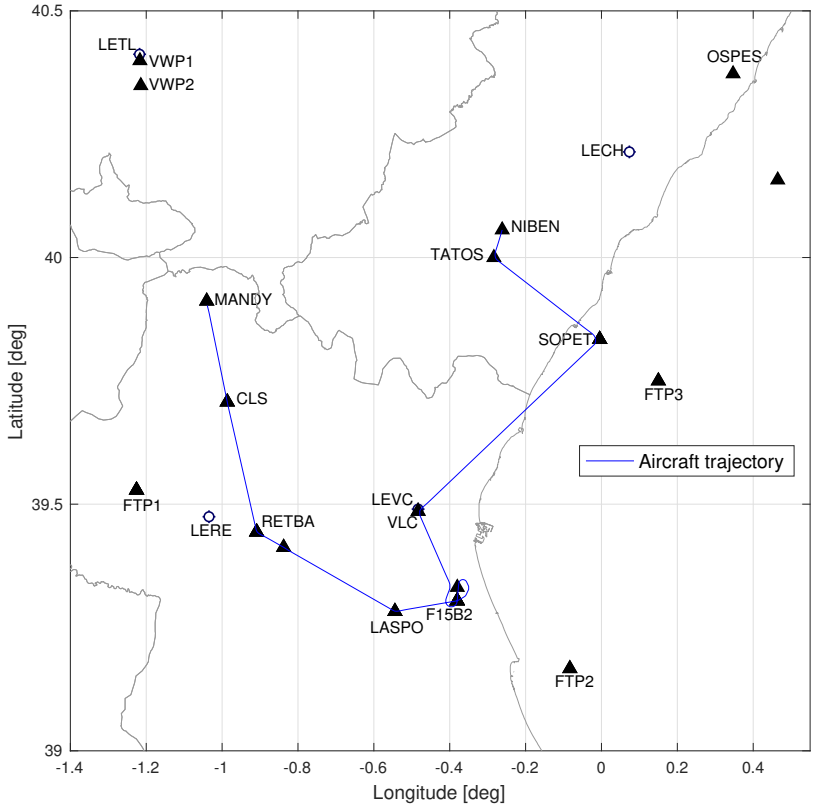


Figure 7.19: Flight simulation results, nominal scenario: Aircraft trajectory.

Then, the flight performance data that result out of performing the strategic goal engaged using the SMMS prototype are graphically represented in Figures 7.19 to 7.25. In particular, Figures 7.19 to 7.21 describe the LNAV profile and Figures 7.22 to 7.25 the VNAV profile. Each of these figures is briefly discussed next.

To start with, Fig. 7.19 shows the actual trajectory followed by the RPAS to achieve the nominal goal. Note that it corresponds to the nominal route described in Sec. 7.2.1 (although it starts from waypoint MANDY, the first waypoint of the en-route phase); then, it traverses en-route waypoints CLS, RETBA, MOPIR and LASPO; it enters the operations area through waypoint F15B2; it performs a manual overfly over the Albufera's Natural Park; it exits the operations area through the exit point F15B2; and it flies towards the destination airport LECH following the en-route waypoints VLC, SOPET, and

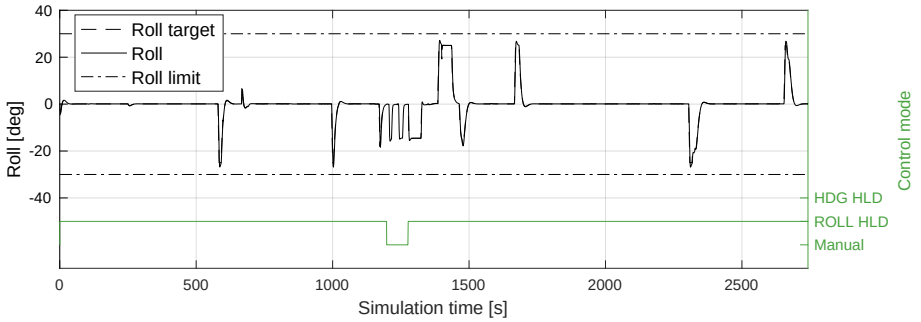


Figure 7.20: Flight simulation results, nominal scenario: Roll angle.

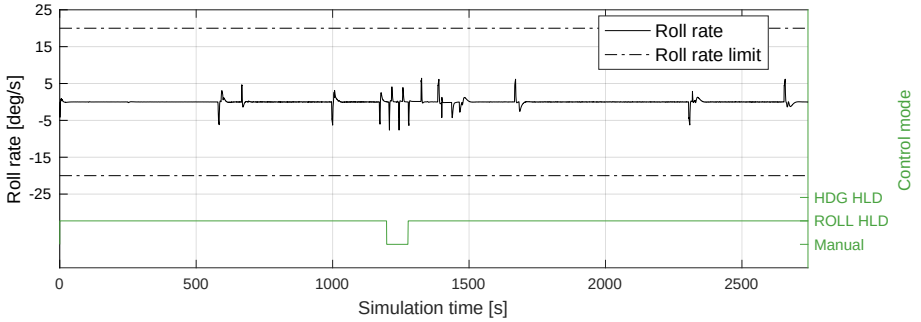


Figure 7.21: Flight simulation results, nominal scenario: Roll rate.

the arrival waypoints TATOS and NIBEN. The total flight duration was 45 min and 42 s.

Fig. 7.20 shows the roll angle of the RPAS, along with the control mode engaged in the LNAV profile. As it can be observed, the active control mode is always the roll hold (`ROLL HLD`) mode, except during the manual segment in the operations area, where the control system is disengaged by the remote pilot to perform the FM leg. Note that the roll hold mode is effective at following the control target provided by the guidance system; and that the roll values are always within the roll limits considered for the aircraft model under study.

The last relevant variable related with the LNAV profile is the roll rate, which is plotted in Fig. 7.21. As it can be observed, the design of the SMMS prototype allows to maintain the roll rate values within the roll rate limits considered

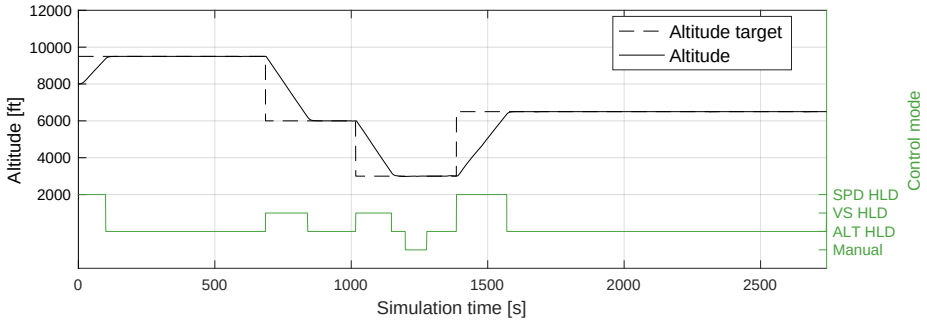


Figure 7.22: Flight simulation results, nominal scenario: Aircraft altitude.

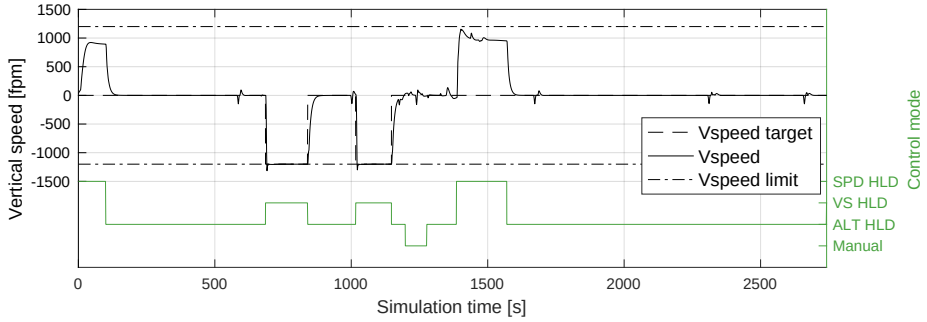


Figure 7.23: Flight simulation results, nominal scenario: Vertical speed.

for the aircraft model under study; what allows to reduce the probability of experiencing a loss of control in-flight.

With respect to the VNAV profile, Fig. 7.22 shows the RPAS altitude during the flight simulation, along with the altitude target and the corresponding control mode engaged. As it can be observed, the initial altitude target is FL95, the lowest altitude level of airway R29. Note that the aircraft reaches the target altitude using the airspeed hold mode (`SPD HLD`), the control mode use for performing ascents in the en-route phase. Once waypoint MOPIR in airway M871 is reached, a descent to 6000 ft is commanded. In this case, the descent is performed using the vertical speed hold mode (`VS HLD`). Afterwards, the RPAS exits controlled airspace and descends to 3000 ft just before entering the operations area. This altitude is maintained during all the operations segment. Finally, once the operations segment is completed, the aircraft climbs

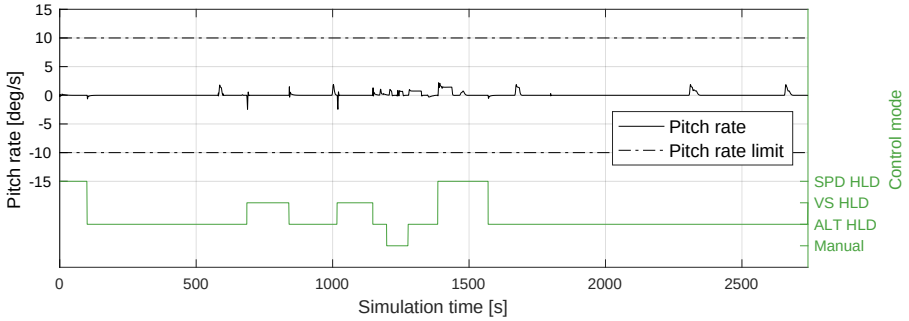


Figure 7.24: Flight simulation results, nominal scenario: Pitch rate.

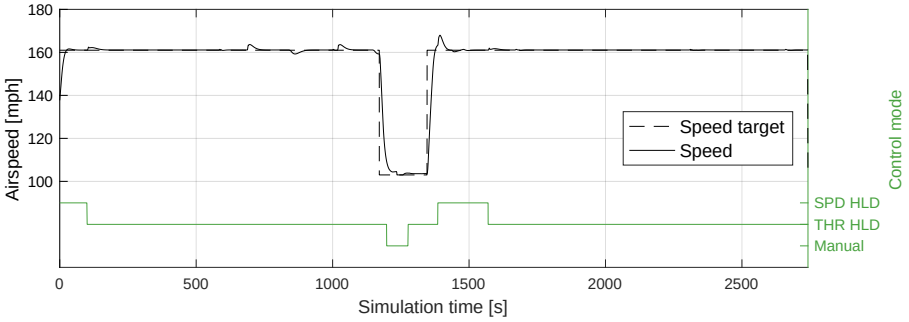


Figure 7.25: Flight simulation results, nominal scenario: Indicated airspeed.

to 6500 ft to overly the València CTR and fly towards the destination airport LECH.

Fig. 7.23 shows the vertical speed of the RPAS, along with the vertical speed target, the vertical speed limit and control mode engaged at each time. Note that the `VS HLD` mode is only engaged for performing descents; and that the vertical speed value is always within the specified safety margins, independently of the active control mode.

The pitch rate data is plotted in Fig. 7.24. It is to be noted that the Flight Director module is able to maintain the roll rate values within the roll rate limits of the aircraft model under study, what helps ensuring the aircraft stability and control.

Finally, the speed profile is shown in Fig. 7.25. As it can be observed, the aircraft maintains a constant indicated airspeed of 160 mph during all the mission, except during the operations segment, where the airspeed is reduced down to 100 mph to perform the mission task. Note also that the speed is normally controlled using the auto-throttle mode (`THR HLD`), except during the ascents, where the `SPD HLD` mode is used.

7.3.2 Contingency scenario CS1

The same mission is performed again but this time the contingency scenario **SC1** will be simulated: a GNSS loss of performance event will be injected right after the RPAS reaches waypoint LASPO, when it is flying the ingress segment towards the operations area[4]. The flight simulation results of this scenario are gathered in Figures 7.26 to 7.34. To start with, Fig. 7.26 shows the contingency injection signal sent to the SMMS application. Note that the GNSS loss of performance injection signal is triggered at time $t = 1\,015$ s. The Safety Monitor component is then notified of the event triggering and enters the "Degraded navigation" state, see Fig. 7.27.

Since the Degraded navigation state is catalogued as an abnormal state, the Contingency Manager reacts and suggests the remote pilot the most convenient response. In this case, the Contingency Plan dictates that the optimal option (i.e. the contingency option that minimizes the risk entailed for third parties) corresponds the strategic goal number 5 (see Fig. 7.28): a "loiter" goal associated with waypoint F15B2 (the entrance to the operations area)[5]. After a short reaction time, the remote pilot accepts the suggested policy and arms the mission goal number 5 using the mission control panel (signal "Strategic goal armed" of Fig. 7.28). Given that the SMMS is being operated in the automatic mode, the mission goal armed by the remote pilot is automatically engaged by the automatic system (note that "Strategic goal armed" and "Strategic goal engaged" signals are superimposed in Fig. 7.28).

As a result, the initial mission is replanned and a new route to achieve the engaged goal is configured by the SMMS using the graph search tools in Appendix C. In this case, the new route is described by the sequence of nodes in Eq. (7.2). The resulting aircraft trajectory is represented in Fig. 7.29. As it can

[4]Note that, although a GNSS loss of performance condition is simulated, the actual flight data processed by the SMMS prototype will not be degenerated or corrupted in the simulation environment for simplicity. This way, we will simply analyze how the SMMS reacts to such degraded condition, but the flight performance observed will not be representative of the assumed flight condition.

[5]Note that the specification of this goal was shown in Fig. 7.8. Note also that this response is in line with the GNSS loss of performance policy that was analyzed off-line in Table 7.19.

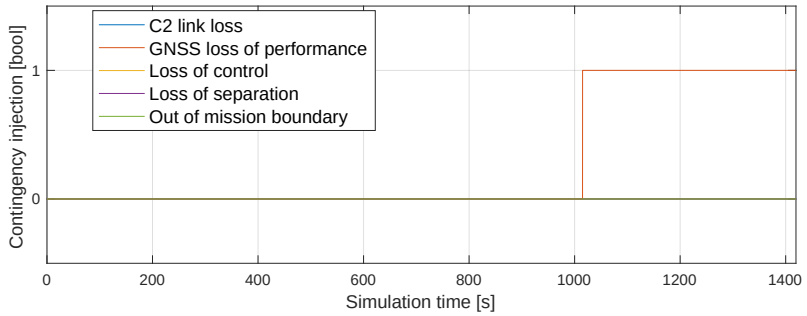


Figure 7.26: Flight simulation results, contingency scenario **CS1**: Contingency injection.

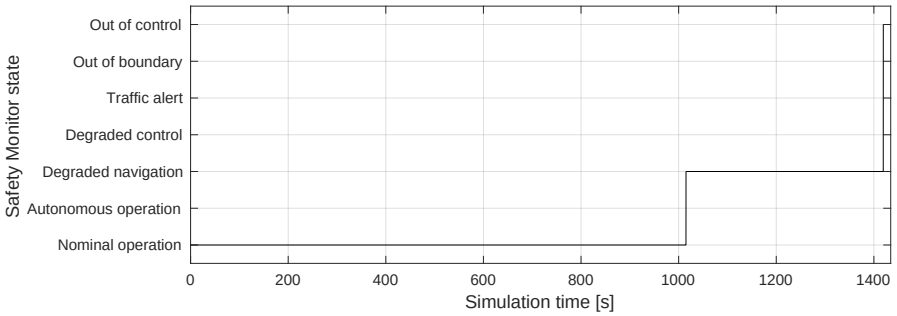


Figure 7.27: Flight simulation results, contingency scenario **CS1**: Safety Monitor state.

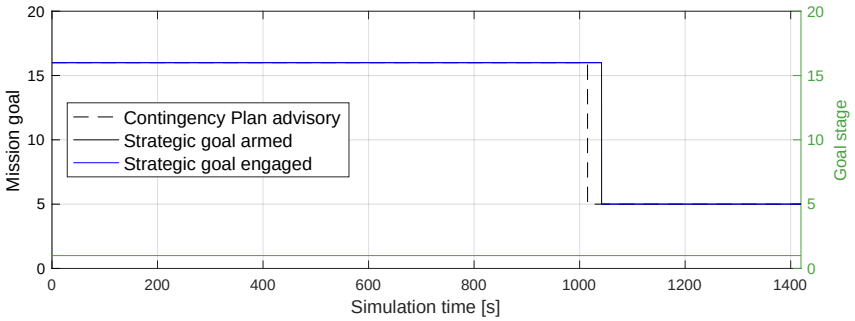


Figure 7.28: Flight simulation results, contingency scenario **CS1**: Mission goal selection.

be observed, the route reconfiguration avoids entering the operations area in the current degraded state, which could have caused an unsafe condition. Con-

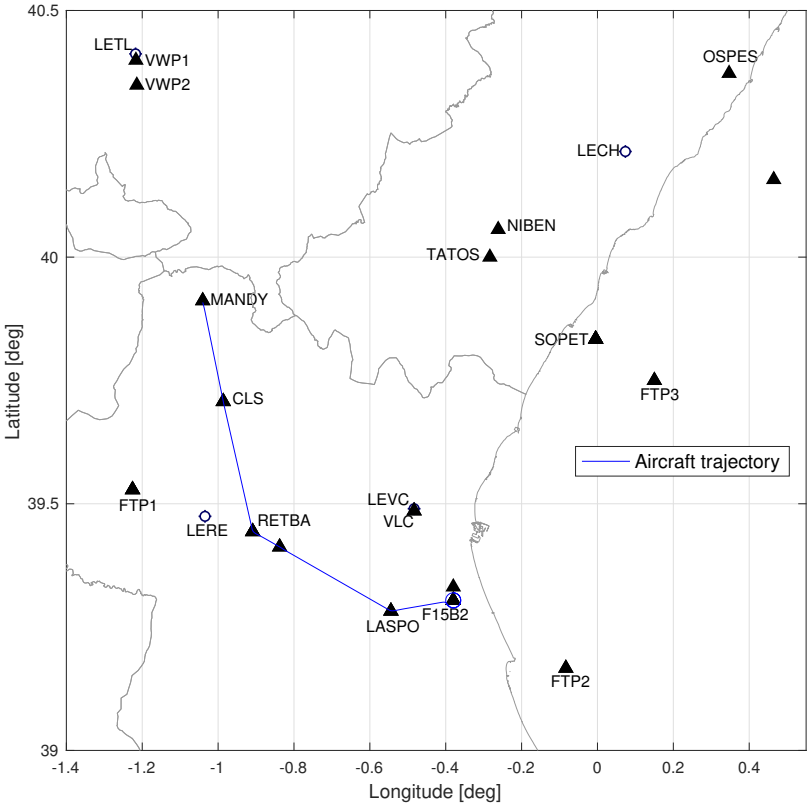


Figure 7.29: Flight simulation results, contingency scenario **CS1**: Aircraft trajectory.

versely, once the RPAS reaches the new target destination (waypoint F15B2), it performs the flight procedure enabled by the mission goal engaged: a loiter maneuver. In fact, it can be noted how the RPAS orbits around waypoint F15B2 in Fig. 7.29.

One important aspect regarding this enabled procedure is that it was specified using a time-out limit of 5 min, see Fig. 7.8. This implies that the remote pilot has necessarily to take a further action before the time limit occurs; otherwise, the mission goal will be considered to be ineffective, and the SMMS will take corresponding actions (in this approach, flight termination). In this scenario, we have opted for simulating this latter safety-critical condition: the GNSS performance is not recovered after performing the holding maneuver (because we have not deactivated the contingency injection signal, see Fig. 7.26), the remote pilot does not provide any alternative contingency option and, after

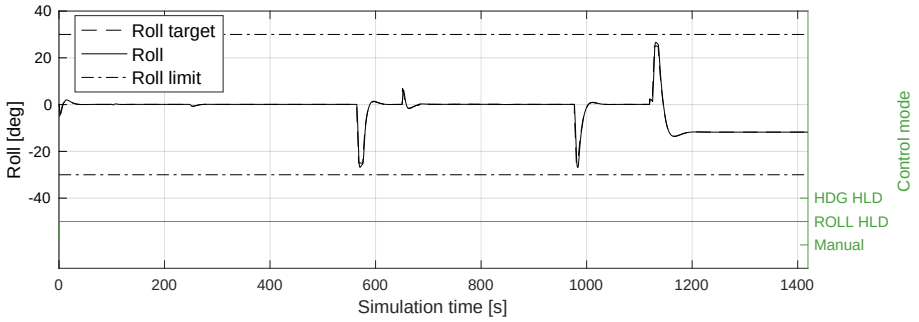


Figure 7.30: Flight simulation results, contingency scenario **CS1**: Roll angle.

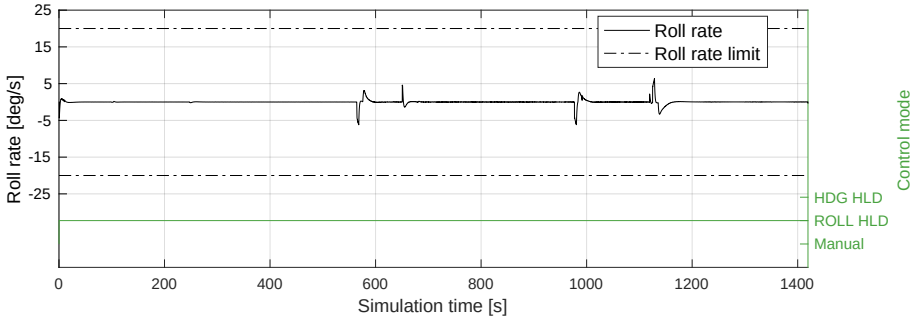


Figure 7.31: Flight simulation results, contingency scenario **CS1**: Roll rate.

the time-out occurs, the Safety Monitor enters the "out of control" state and triggers the Flight Termination System (see Fig. 7.27, time $t = 1\,419$ s). Note that, when the SMMS prototype runs in the simulation environment, the activation of the Flight Termination System simply stops the current simulation, see Sec. B.8. The total flight duration was 23 min and 39 s.

The remaining flight performance data are represented in Figures 7.30 to 7.35 for completeness. In particular, these figures show the roll angle of the RPAS, the roll rate, the altitude profile, the vertical speed, the pitch rate and the indicated airspeed during the simulation. As an example, note how the airspeed is reduced down to 120 mph in Fig. 7.35 once the RPAS starts the loitering maneuver, as it is specified by the corresponding procedure attributes of Fig. 7.8. Note also that the figure describing the C2 link loss policy armed was omitted for brevity since it is not relevant in this scenario.

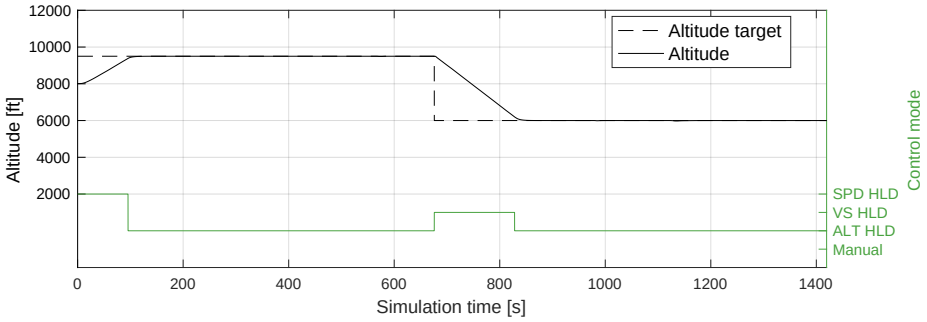


Figure 7.32: Flight simulation results, contingency scenario **CS1**: Aircraft altitude.

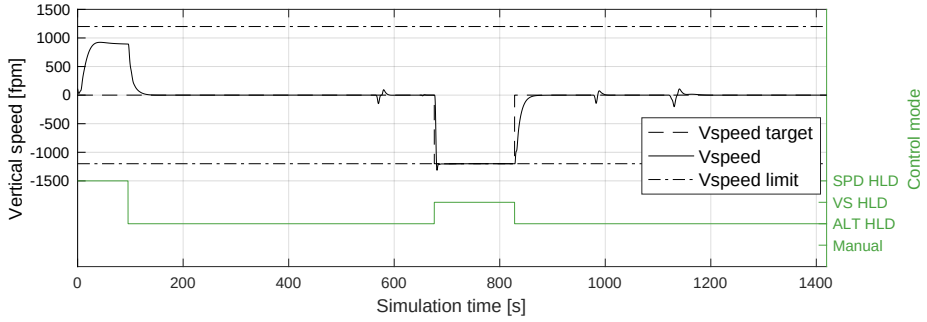


Figure 7.33: Flight simulation results, contingency scenario **CS1**: Vertical speed.

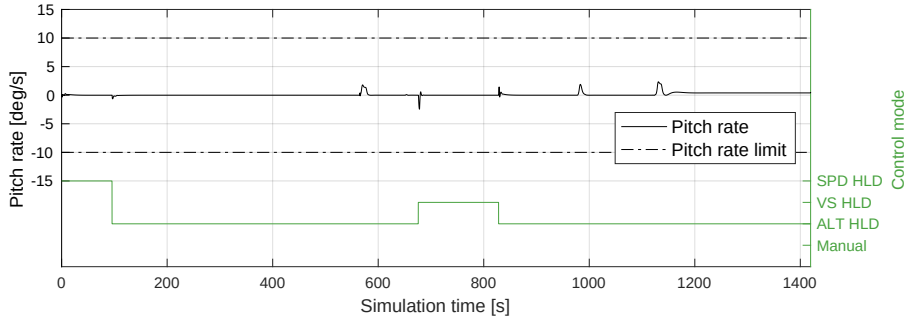


Figure 7.34: Flight simulation results, contingency scenario **CS1**: Pitch rate.

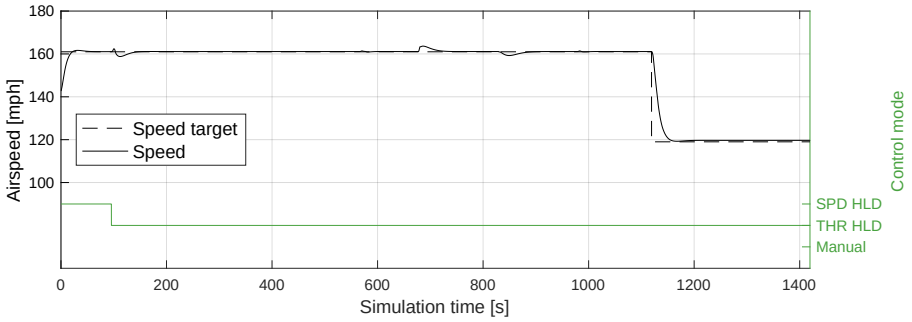


Figure 7.35: Flight simulation results, contingency scenario **CS1**: Indicated airspeed.

7.3.3 Contingency scenario CS2

The last validation results of this work correspond to the simulation of the contingency scenario **CS2**, i.e. a C2 link loss event will be injected while the RPAS is flying the nominal route of Fig. 7.6, specifically when it is located somewhere between waypoints VLC and SOPET, in segment s_6. The flight simulation results of this scenario are gathered in Figures 7.36 to 7.45.

To start with, Fig. 7.36 shows that the C2 link loss signal is injected at time $t = 1\,864$ s. The flight data before this happens are similar to those of the nominal case in Sec. 7.3.1, so they are here omitted for brevity. Once the contingency is injected, the Safety Monitor receives the corresponding contingency signal and enters the "Autonomous operation" state, see Fig. 7.37.

Since this state is catalogued as an abnormal state, the Contingency Manager must provide the most convenient response for this degraded flight condition. In this case, the C2 link loss policy that was armed at the time the contingency occurred is the mission goal number 9, see Fig. 7.38. Note that the Contingency Manager suggested this policy at time $t = 1\,393$ s and the remote pilot accepted it at time $t = 1\,410$ s (see "C2 link loss policy advisory" and "C2 link loss policy armed" signal signals in Fig. 7.39).

As a result, right after the C2 link is lost, the SMMS disengages the nominal goal and switches to the C2 link loss policy approved by the remote pilot: the mission goal number 9. It corresponds to a "climb to regain signal" goal associated with waypoint FTP3[6]. Once this goal is engaged, the SMMS computes a route to achieve this goal dynamically using the Reconfigurable Mission Plan

[6]Note that the specification of this goal was shown in Fig. 7.9. Note also that this response is in line with the C2 link loss policy that was analyzed off-line in Table 7.18.

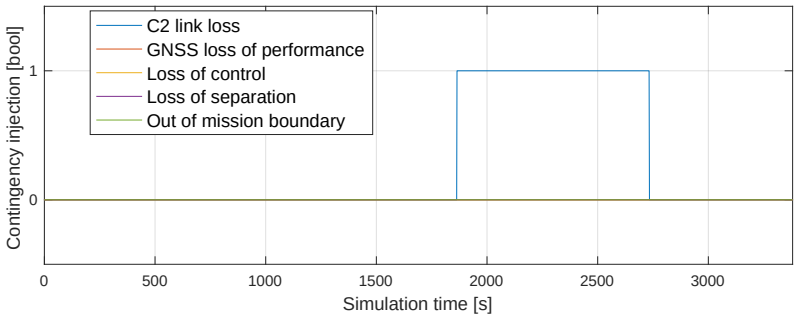


Figure 7.36: Flight simulation results, contingency scenario **CS2**: Contingency injection.

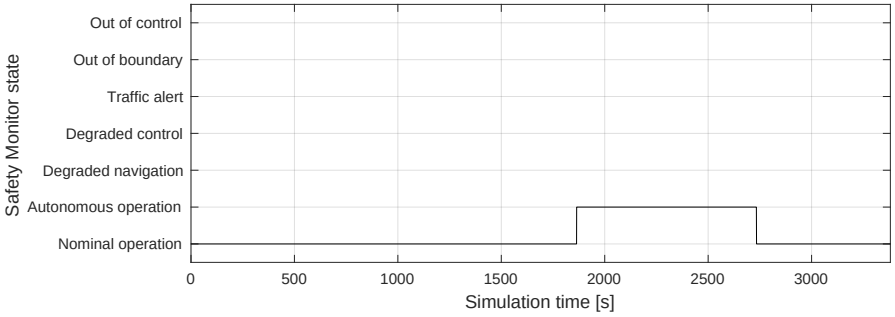


Figure 7.37: Flight simulation results, contingency scenario **CS2**: Safety Monitor state.

data, and specifically the Mission Graph and the graph search tools in Appendix C. In this case, the new route is described by the sequence of nodes of Eq. (7.3).

The resulting aircraft trajectory is represented in Fig. 7.40. As it can be observed, once the RPAS reaches waypoint SOPET, it does not start the arrival segment towards the destination site LECH; rather, it deviates towards waypoint FTP3 (a safe area located over the sea) what reduces the risk entailed for third parties. Once this waypoint is reached, the SMMS performs the flight procedure associated to the current goal: a climb maneuver trying to regain the C2 link signal. We have implemented a Path Planner that generates the procedure's reference trajectory as an helicoidal ascent around the target waypoint. In addition, we allow to specify some of the procedure's attributes in the Reconfigurable Mission Plan. For example, as it was shown in Fig. 7.9, the target altitude for this procedure is FL100 and the time slot allowed to

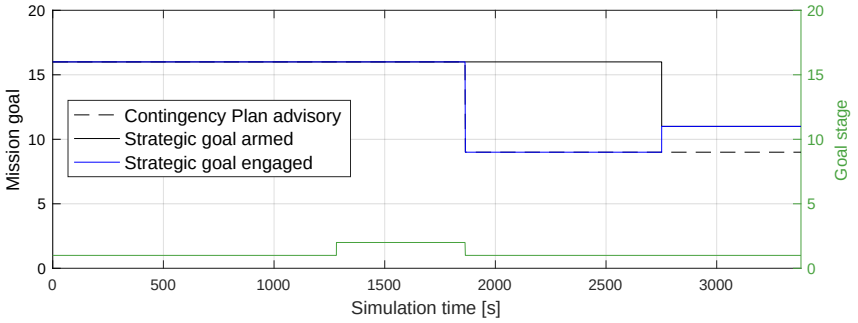


Figure 7.38: Flight simulation results, contingency scenario **CS2**: Mission goal selection.

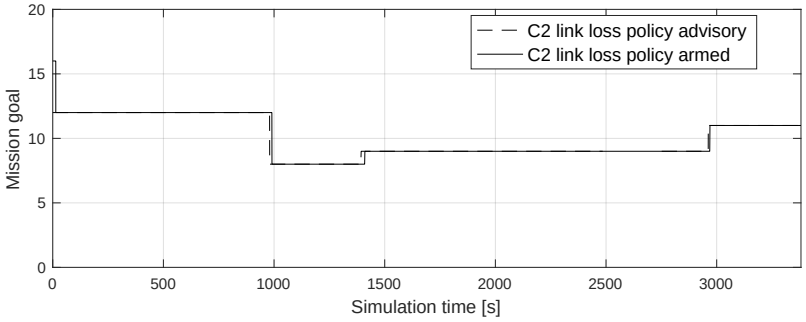


Figure 7.39: Flight simulation results, contingency scenario **CS2**: C2 link loss policy selection.

try to regain the signal is 10 min in this case. The resulting vertical profile is represented in Fig. 7.43.

Unlike the previous case, this time we have opted for simulating that the "regain signal" is effective at recovering the C2 link before the time limit occurs: as it can be observed in Fig. 7.36, the contingency injection signal is deactivated at time $t = 2,733$ s, so the C2 link is assumed to be restored. As a result, the Safety Monitor returns to the "Nominal operation" state (Fig. 7.37), and the RPAS remains loitering, waiting for a remote pilot action. As it is shown in Fig. 7.38, the pilot decision is to engage the mission goal number 11, a "land" goal associated with the main landing site LECH. The route to achieve this goal allows to the RPAS to safely reach the intended destination. The total flight duration was 56 min and 21 s (appreciably longer than in the nominal case).

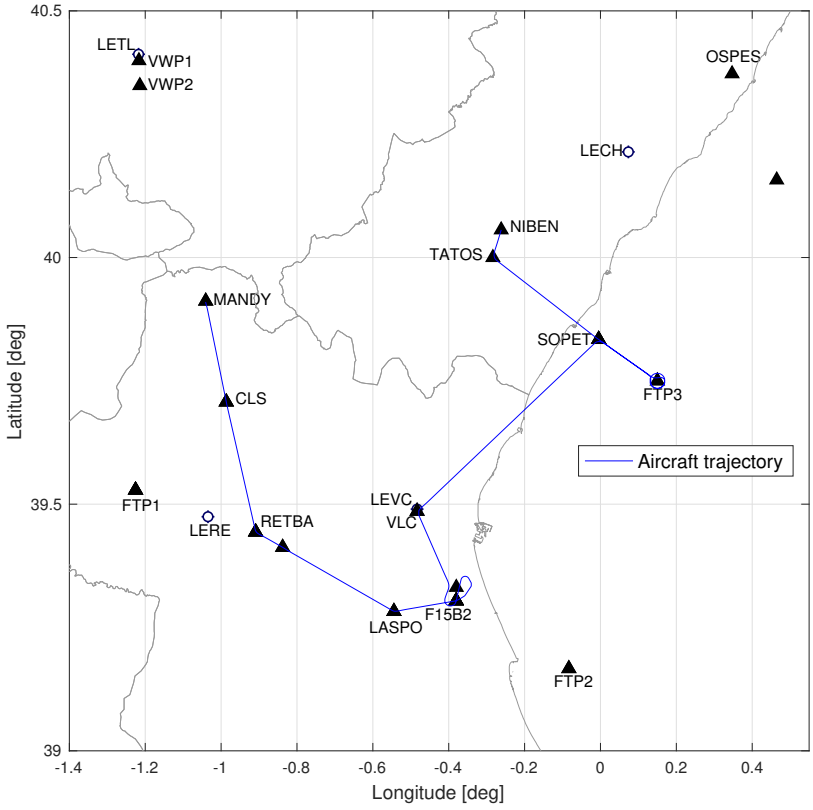


Figure 7.40: Flight simulation results, contingency scenario **CS2**: Aircraft trajectory.

For completeness, the remaining flight simulation data is also represented in Figures 7.41 to 7.46, including the roll angle data, the roll rate, the altitude profile, the vertical speed, the pitch rate and the indicated airspeed.

In summary, the flight simulation results showed how the proposed SMMS prototype is able to perform Automated Contingency Management functions in simulation time; how these functions provide dynamic routes that minimize the risk entailed for third parties in the event of a contingency; and how the remote pilot can participate in the decision-making process to increase the predictability of the RPAS.

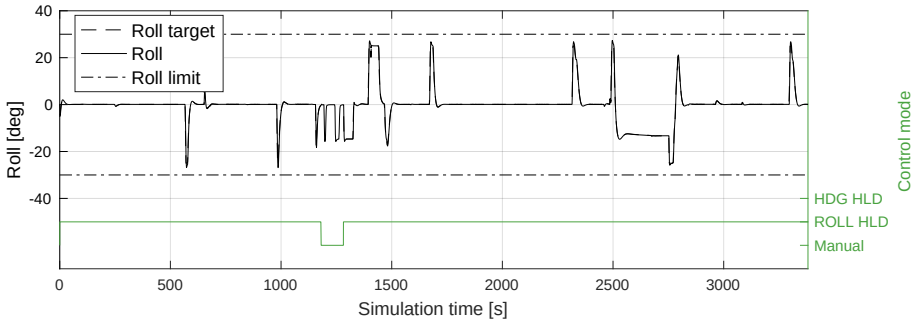


Figure 7.41: Flight simulation results, contingency scenario **CS2**: Roll angle.

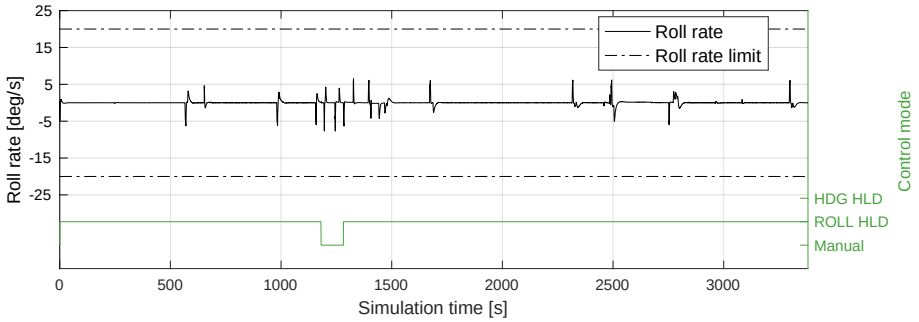


Figure 7.42: Flight simulation results, contingency scenario **CS2**: Roll rate.

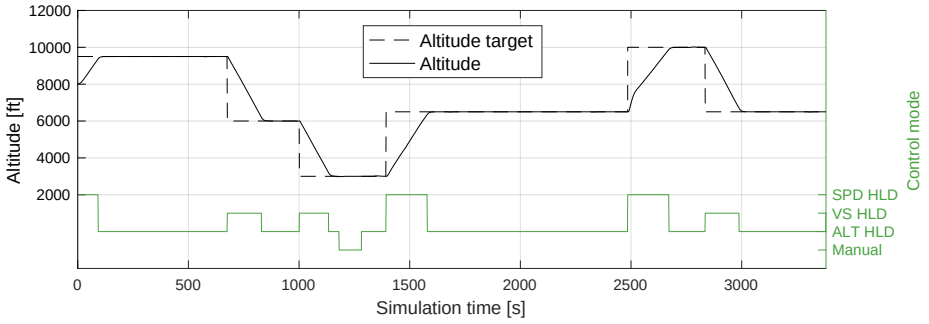


Figure 7.43: Flight simulation results, contingency scenario **CS2**: Aircraft altitude.

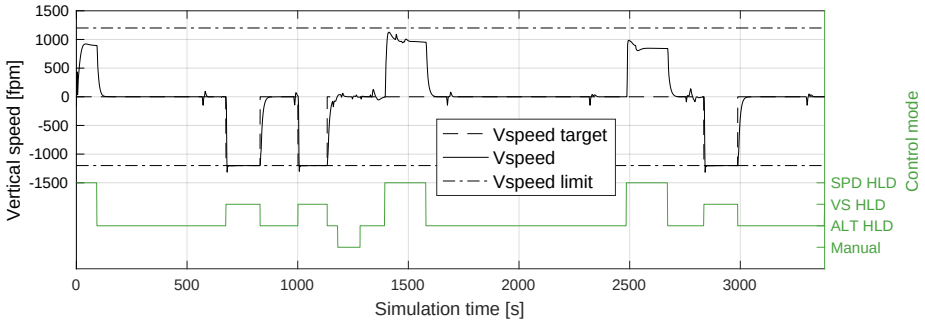


Figure 7.44: Flight simulation results, contingency scenario **CS2**: Vertical speed.

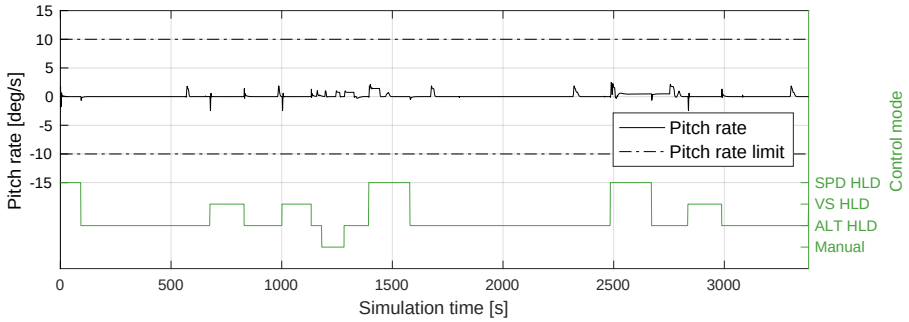


Figure 7.45: Flight simulation results, contingency scenario **CS2**: Pitch rate.

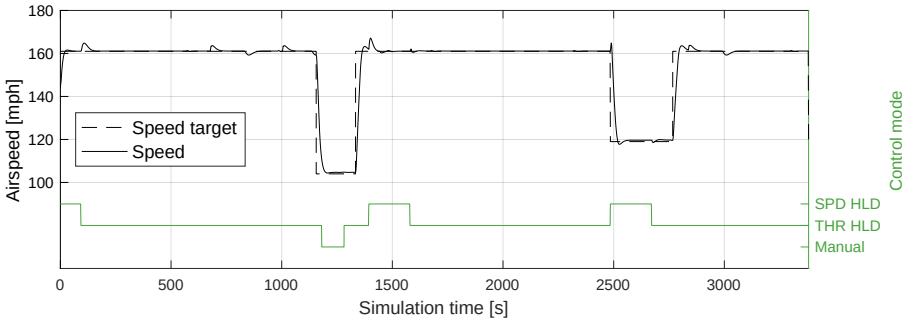


Figure 7.46: Flight simulation results, contingency scenario **CS2**: Indicated airspeed.

Chapter 8

Conclusions and future work

8.1 Conclusions

This work has analyzed the feasibility of increasing the level of automation in aircraft in general, and in unmanned aircraft in particular, without compromising the safety levels of the current ATM system. The proposal in this work consisted of providing the automatic system on-board the aircraft with the ability to perform ACM functions.

The principal contributions of this thesis are:

- A software architecture for the automatic system on-board a UAS that tries to adapt autonomously to abnormal states before commanding the flight termination action.
- A novel Mission Plan specification that overcomes the limitations of the conventional flight plans for describing RPAS missions and that ensures predictability in the event of a contingency.
- A probabilistic risk model that is consistent with the SORA analysis and that can be used to minimize the risk exposure during the operation.

- A suit of tools and methods for safely enabling higher levels of automation in aviation.

The proposed software architecture provides an interesting balance between safety and robustness. Results indicate that the Contingency Manager can often find a feasible mitigation action that reduces the need of performing flight termination. Moreover, the Contingency Manager is an effective tool to assist the remote pilot in finding the most convenient mitigation action, what should reduce the pilot's reaction time after the contingency occurs.

One of the most sensitive issues concerning the design of the software architecture was the decision of hardcoding the contingency management policy into the embedded software to the detriment of user flexibility (although the final user will be free to specify the way contingency procedures are executed using the Mission Plan). This decision was made in the interest of safety, since the possibility of performing an exhaustive testing strategy is essential for system safety.

Concerning the testing strategy, the use of formal methods for model checking was proposed in this work. Results have demonstrated the correctness of the proposed contingency management policy before the implementation phase. In addition, the use of partitioning as a means for fault contention was also proposed. In this regard, the different software components of the architecture were allocated to a particular partitioning configuration that is expected to reduce the software verification effort.

The novel Mission Plan specification provides the final user with some level of customization in the definition of the contingency management scheme. The proposed Mission Plan supports the ACM functions performed by the on-board system as it allows detailed specification of all the possible routes the aircraft can fly. This improves predictability and increases the level of automation by enabling automatic reconfiguration of the intended plan. The formal specification of the Reconfigurable Mission Plan is adequate for describing the specificities of a RPAS mission.

In order to ensure that the dynamic reconfiguration of the intended plan is optimal from the point of view of the risk exposure, a probabilistic risk model is proposed. One of the main novelties of the proposed model is it is consistent with the SORA approach to risk analysis. The risk model must be supplied with a number of input parameters such as aircraft model, population density or traffic density, among others; the degree of uncertainty about these parameters will determine the trustworthiness of the results obtained.

In conclusion, results support the idea of safely enabling higher levels of automation in aviation by introducing ACM functions into the on-board system. Results demonstrate that the proposed contingency management functions are an effective mechanism for minimizing the operational risk of an aircraft experiencing a contingency, and that the resulting aircraft behavior is still deterministic and consistent with airspace regulation, even without pilot intervention. This should facilitate the safe integration of RPAS into civil airspace.

8.2 Future work

Future lines of research can be categorized in the following three areas:

- *To address the mission reconfiguration problem as a multi-objective optimization problem.* During this research, an algorithm for automatic mission reconfiguration has been develop. The proposed algorithm exploits Graph Theory and a probabilistic risk model to provide optimal routes from the point of view of the risk exposure. However, the algorithm does not take into account the resulting route length; and more importantly, it does not check if the proposed route can be completed with the current fuel capacity or battery level. Therefore, future work is to reformulate the mission reconfiguration problem so that the output route minimizes the risk while is still flyable according to the aircraft endurance.

- *To analyze human factors when flying Reconfigurable Mission Plans.* This includes two particular sub-problems: *1)* to develop a proper HMI between the remote pilot and the RPA that facilitates the selection of mission goals and their associated alternative routings; and *2)* to define communication protocols between the remote pilot and the ATC authority, and between the RPA and the ATC authority in regards to the trajectory negotiation process. In this latter aspect, the work in [138] should be used as a baseline.

- *To apply artificial intelligence to the decision-making process.* Intelligent systems go beyond the autonomous capability analyzed in this work as they are able to generate their own goals based on internal motivation and self-learning methods without human support. As an example, an intelligent UAS could avoid flying towards certain areas that have been proved to have a higher risk of losing the C2 link. In other words, an intelligent system could anticipate hazardous conditions to be encountered during the operation, and could prevent them in a more efficient way than it is done today.

Bibliography

[1] F. Adolf and M. M. Carneiro. "Behavior-based High Level Control of a VTOL UAV". In: *AIAA Infotech @ Aerospace Conference*. AIAA. Seattle, Washington, 2009, pp. 1–13. DOI: 10.2514/6.2009-1977.

[2] F. Adolf and F. Thielecke. "A Sequence Control System for Onboard Mission Management of an Unmanned Helicopter". In: *AIAA Infotech @ Aerospace, AIAA SciTech*. AIAA. Rohnert Park, California, 2007, pp. 2769–2780. DOI: 10.2514/6.2007-2769.

[3] F. Adolf, F. Andert, S. Lorenz, L. Goormann, and J. Dittrich. "An Unmanned Helicopter for Autonomous Flights in Urban Terrain". In: *Advances in Robotics Research: Theory, Implementation, Application*. Ed. by T. Kröger and F. Wahl. Vol. 9. Springer, Berlin, Heidelberg, 2009, pp. 275–285.

[4] F. Adolf, P. Faymonville, B. Finkbeiner, S. Schirmer, and C. Torens. "Stream Runtime Monitoring on UAS". In: *Runtime Verification. Lecture Notes in Computer Science*. Ed. by S. Lahiri and G. Reger. Vol. 10548. Springer, Cham, 2017. DOI: 10.1007/978-3-319-67531-2_3.

[5] Aeronautical Radio, Inc. *ARINC specification 424-15. Navigation System Data Base*. 2000.

[6] Aeronautical Radio, Inc. *ARINC specification 653-1. Avionics Application Software Standard Interface*. 2003.

[7] Agencia Estatal de Seguridad Aérea. *Apéndice O: Medios aceptables de cumplimiento relativos a los requisitos de los equipos para la operación con RPAS según el Real Decreto 552/2014*. 2018.

[8] Agencia Estatal de Seguridad Aérea. *Apéndice S: Guía sobre el contenido del estudio aeronáutico de seguridad*. 2018.

[9] Airbus Media. *EASA certifies new "Autopilot/Flight Director" TCAS mode for A380*. Available online: https://www.airbus.com/newsroom/press-releases/en/2009/08/easa-certifies-new-autopilot-flight-director-tcas-mode-for-a380.html (Last accessed on 24 June 2019).

[10] E. Ancel, F. M. Capristan, J. V. Foster, and R. C. Condotta. "Real-time risk assessment framework for Unmanned Aircraft System (UAS) Traffic Management (UTM)". In: *17th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*. AIAA. Denver, Colorado, 2017, p. 3273. DOI: 10.2514/6.2017-3273.

[11] J. N. Anno. "Estimate of human control over mid-air collisions". In: *Journal of Aircraft* 19.1 (1982), pp. 86–88.

[12] J. B. Baeker, J. D. Collins, and J. M. Haber. "Launch risk analysis". In: *Journal of Spacecraft and Rockets* 14.12 (1977), pp. 733–738. DOI: 10.2514/3.27993.

[13] C. Baier and J.-P. Katoen. *Principles of model checking*. The MIT press, 2008.

[14] J. Bang-Jensen and G. Z. Gutin. *Digraphs: Theory, Algorithms and Applications*. 2nd ed. Springer, 2008.

[15] M. Barbier and E. Chanthery. "Autonomous mission management for unmanned aerial vehicles". In: *Aerospace Science and Technology* 8.4 (2004), pp. 359–368. DOI: 10.1016/j.ast.2004.01.003.

[16] L. C. Barr, R. L. Newman, E. Ancel, C. M. Belcastro, J. V. Foster, J. Evans, and D. H. Klyde. "Preliminary risk assessment for small Unmanned Aircraft Systems". In: *17th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*. AIAA. Denver, Colorado, 2017, p. 3272. DOI: 10.2514/6.2017-3272.

[17] C. M. Belcastro, R. L. Newman, J. Evans, D. H. Klyde, L. C. Barr, and E. Ancel. "Hazards Identification and Analysis for Unmanned Aircraft System Operations". In: *17th AIAA Aviation Technology, Integration, and Operations Conference, AIAA AVIATION Forum*. AIAA. Denver, Colorado, 2017, p. 3269. DOI: 10.2514/6.2017-3269.

[18] K. Blin, M. Akian, F. Bonnans, E. Hoffman, C. Martini, and K. Zeghal. "A Stochastic Conflict Detection Model Revisited". In: *18th Applied Aerodynamics Conference*. AIAA. Denver, Colorado, 2000, pp. 4270–4279. DOI: 10.2514/6.2000-4270.

[19] Boletín Oficial del Estado. "Real Decreto-ley 8/2014, de 4 de julio, de aprobación de medidas urgentes para el crecimiento, la competitividad y la eficiencia." 2014.

[20] R. P. Bonasso, R. J. Firby, E. Gat, D. Kortenkamp, D. P. Miller, and M. G. Slack. "Experiences with an architecture for intelligent, reactive agents". In: *Journal of Experimental & Theoretical Artificial Intelligence* 9.2-3 (1997), pp. 237–256. DOI: 10.1080/095281397147103.

[21] R. Brooks. "A robust layered control system for a mobile robot". In: *IEEE Journal on Robotics and Automation* 2.1 (1986), pp. 14–23. DOI: 10.1109/JRA.1986.1087032.

[22] D. A. Burke. "System Level Airworthiness Tool: A Comprehensive Approach to Small Unmanned Aircraft System Airworthiness". PhD thesis. Raleigh, North Carolina: North Carolina State University, 2010.

[23] H. Chao, Y. Cao, and Y. Chen. "Autopilots for small fixed-wing unmanned air vehicles: A survey". In: *International Conference on Mechatronics and Automation*. IEEE. Harbin, China, 2007, pp. 3144–3149. DOI: 10.1109/ICMA.2007.4304064.

[24] A. Cimatti, E. M. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani, and A. Tacchella. "NuSMV 2: An OpenSource Tool for Symbolic Model Checking". In: *Computer Aided Verification*. Ed. by E. Brinksma and K. G. Larsen. Springer, Berlin, Heidelberg, 2002, pp. 359–364. DOI: 10.1007/3-540-45657-0_29.

[25] Civil Air Navigation Services Organisation. *Air Navigation Service Provider (ANSP) Considerations for RPAS Operations*. 2014.

[26] R. A. Clothier, B. P. Williams, and K. J. Hayhurst. "Modelling the risks remotely piloted aircraft pose to people on the ground". In: *Safety Science* 101 (2018), pp. 33–47. ISSN: 0925-7535. DOI: 10.1016/j.ssci.2017.08.008.

[27] R. A. Clothier, R. A. Walker, N. Fulton, and D. A. Campbell. "A casualty risk analysis for Unmanned Aerial System (UAS) operations over inhabited areas". In: *12th Australian International Aerospace Congress (AIAC12)*. Melbourne, Australia, 2007.

[28] B. T. Clough. "Metrics, schmetrics! How the heck do you determine a UAV's autonomy anyway?" In: *Performance Metrics for Intelligent Systems (PerMIS) Conference*. Gaithersburg, Maryland, 2002.

[29] D. Cofer and S. P. Miller. *Formal Methods: Case Studies for DO-333*. NASA Langley Research Center. Hampton, Virginia, 2014.

[30] R. Collinson. *Introduction to Avionics Systems*. 3rd ed. Springer, 2011.

[31] S. D'Souza, A. Ishihara, B. Nikaido, and H. Hasseeb. "Feasibility of varying geo-fence around an unmanned aircraft operation based on vehicle performance and wind". In: *35th Digital Avionics Systems Conference (DASC)*. IEEE/AIAA. Sacramento, California, 2016, pp. 1–10. DOI: 10.1109/DASC.2016.7777987.

[32] K. Dalamagkidis, K. P. Valavanis, and L. A. Piegl. *On integrating Unmanned Aircraft Systems into the National Airspace System: Issues, Challenges, Operational Restrictions, Certification, and Recommendations*. 2nd ed. Vol. 54. Springer, 2012. DOI: 10.1007/978-94-007-2479-2.

[33] L. Damilano, G. Guglieri, F. Quagliotti, and I. Sale. "FMS for Unmanned Aerial Systems: HMI Issues and New Interface Solutions". In: *Journal of Intelligent & Robotic Systems* 65 (2011), pp. 27–42. DOI: 10.1007/s10846-011-9567-3.

[34] F. De Florio. *Airworthiness: An Introduction to Aircraft Certification and Operations*. 3rd ed. Butterworth-Heinemann, 2016.

[35] M. T. DeGarmo. *Issues concerning integration of unmanned aerial vehicles in civil airspace*. MITRE Center for Advanced Aviation System Development. 2004.

[36] S. Deutsch and R. W. Pew. *Single Pilot Commercial Aircraft Operation*. BBN Technologies. Cambridge, Massachusetts, 2005.

[37] A. Dheedan. "Distributed Online Safety Monitor Based on Multi-Agent System and AADL Safety Assessment Model". In: *Distributed networks: Intelligence, security, and applications*. Ed. by Q. A. Memon. CRC Press, 2008, pp. 317–345.

[38] E. T. Dill, S. D. Young, and K. J. Hayhurst. "SAFEGUARD: An assured safety net technology for UAS". In: *35th Digital Avionics Systems Conference (DASC)*. IEEE/AIAA. Sacramento, California, 2016, pp. 1–10. DOI: 10.1109/DASC.2016.7778009.

[39] J. S. Dittrich, A. Bernatz, and F. Thielecke. "Intelligent Systems Research using a small autonomous rotorcraft testbed". In: *2nd AIAA Unmanned Unlimited Conference, Workshop and Exhibit*. AIAA. San Diego, California, 2003, pp. 6561–6572. DOI: 10.2514/6.2003-6561.

[40] G. J. Ducard. *Fault-tolerant flight control and guidance systems*. 1st ed. London: Springer-Verlag, 2009.

[41] C. M. Eaton, E. K. P. Chong, and A. A. Maciejewski. "Multiple-scenario Unmanned Aerial System control: a systems engineering approach and review of existing control methods". In: *Aerospace* 3.1 (2015). DOI: 10.3390/aerospace3010001.

[42] M. R. Endsley and D. B. Kaber. "Level of automation effects on performance, situation awareness and workload in a dynamic control task." In: *Ergonomics* 42.3 (1999), pp. 462–492. DOI: 10.1080/001401399185595.

[43] Eurocontrol Experimental Centre. *Point Merge Integration of Arrival Flows Enabling Extensive RNAV Application and Continuous Descent - Operational Services and Environment Definition*. 2nd ed. Brétany-sur-Orge, France, 2010.

[44] European Aviation Safety Agency. *Advance Notice of Proposed Amendment 2015-10: Introduction of a regulatory framework for the operation of drones*. 2015.

[45] European Aviation Safety Agency. *Certification Specification: CS VLA - Subpart B*. 2016.

[46] European Aviation Safety Agency. *Certification Specifications for Large Aeroplanes CS-25 Amendment 5*. 2008.

[47] European Aviation Safety Agency. *Concept of Operations for Drones: A risk based approach to regulation of unmanned aircraft*. 2015.

[48] European Aviation Safety Agency. *Notice of Proposed Amendment 2017-05 (A): Introduction of a regulatory framework for the operation of drones. Unmanned aircraft system operations in the open and specific category*. 2017.

[49] European Aviation Safety Agency. *Opinion No 01/2018: Introduction of a regulatory framework for the operation of unmanned aircraft systems in the open and specific categories*. 2018.

[50] European Aviation Safety Agency. *Policy Statement Airworthiness Certification of Unmanned Aircraft Systems (UAS)*. 2009.

[51] European Aviation Safety Agency. *Technical opinion: Introduction of a regulatory framework for the operation of drones*. 2015.

[52] European Organisation for the Safety of Air Navigation. *Eurocontrol Air Traffic Management Guidelines for Global Hawk in European Airspace*. 2010.

[53] European Organisation for the Safety of Air Navigation. *Eurocontrol Specification for ATM Surveillance System Performance (Volume 2 Appendices)*. 1.1. 2015.

[54] European Organisation for the Safety of Air Navigation. *Eurocontrol Specification for the application of the Flexible Use of Airspace (FUA)*. 2009.

[55] European Organisation for the Safety of Air Navigation. *Eurocontrol Specifications for the Use of Military Remotely Piloted Aircraft as Operational Air Traffic Outside Segregated Airspace*. 2nd ed. 2012.

[56] European Organisation for the Safety of Air Navigation. *NEST User Guide*. 1.6. 2013.

[57] European Organisation for the Safety of Air Navigation. *UAS ATM Flight Rules*. 1.1. 2018.

[58] European RPAS Steering Group. *Roadmap for the integration of civil Remotely Piloted Aircraft Systems into the European Aviation System*. European Commission. 2013.

[59] Federal Aviation Administration. *14 CFR PART 107: Small Unmanned Aircraft Systems*. 2016.

[60] Federal Aviation Administration. *14 CFR Parts 1, 45, 47, 48, 91, and 375: Registration and Marking Requirements for Small Unmanned Aircraft; Final Rule*. 2015.

[61] Federal Aviation Administration. *23.1309-lE: System safety analysis and assessment for Part 23 airplanes*. U.S. Department of Transportation. 2011.

[62] Federal Aviation Administration. *Concept of Operations for the Airborne Collision Avoidance System X*. 1st ed. 2012.

[63] Federal Aviation Administration. *Integration of Civil Unmanned Aircraft Systems (UAS) in the National Airspace System (NAS) Roadmap*. 2nd ed. 2018.

[64] L. Fern, C. Rorie, and J. Shively. "UAS contingency management: the effect of different procedures on ATC in civil airspace operations". In: *14th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*. AIAA. Atlanta, Georgia, 2014, p. 2414. DOI: 10.2514/6.2014-2414.

[65] M. Finke. "Defining RPAS emergency procedures for controllers, remote pilots and automatic on-board systems". In: *Deutscher Luft- und Raumfahrtkongress (DLRK)*. Braunschweig, Germany, 2016.

[66] C. Flanagan, D. Toal, and R. Strunz. "Subsumption Architecture for the Control of Robots". In: *Polymodel 16: Applications of Artificial Intelligence*. Sunderland, United Kingdom, 1995, pp. 150–158.

[67] B. Fons, H. Usach, J. Vila, and A. Crespo. "Development of Integrated Modular Avionics Applications based on Simulink and XtratuM". In: *Data Systems In Aerospace (DASIA) Conference*. 2013.

[68] B. Fons. "Plataforma para Diseño y Ejecución de Aplicaciones de Aviónica". MA thesis. Universitat Politècnica de València, 2013.

[69] B. Fons. "Sistema de navegación basado en integración de INS y GPS sobre teléfono móvil". Bachelor thesis. València, Spain: Universitat Politècnica de València, 2011.

[70] A. Frey and T. Hanti. "Expanding the operational range of UAS with an onboard supervisory instance". In: *34th Digital Avionics Systems Conference (DASC)*. IEEE/AIAA. Prague, Czech Republic, 2015, pp. 1–12. DOI: 10.1109/DASC.2015.7311437.

[71] GRA, Incorporated. *Benefit-Cost Analyses for Integration of Unmanned Aircraft Systems into Civilian Aviation Applications*. Jenkintown, Pennsylvania, 2014.

[72] M. Gianni, G. Gonnelli, A. Sinha, M. Menna, and F. Pirri. "An Augmented Reality approach for trajectory planning and control of tracked vehicles in rescue environments". In: *2013 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE. 2013, pp. 1–6. DOI: 10.1109/SSRR.2013.6719360.

[73] F. Grimsley. "Equivalent safety analysis using casualty expectation approach". In: *AIAA 3rd Unmanned Unlimited Technical Conference, Workshop and Exhibit*. AIAA. Chicago, Illinois, 2004, p. 6428. DOI: 10.2514/6.2004-6428.

[74] T. Gurriet and L. Ciarletta. "Towards a generic and modular geofencing strategy for civilian UAVs". In: *International Conference on Unmanned Aircraft Systems (ICUAS)*. Arlington, Virginia, 2016, pp. 540–549. DOI: 10.1109/ICUAS.2016.7502603.

[75] K. J. Hayhurst, J. M. Maddalon, P. S. Miner, G. N. Szatkowski, M. L. Ulrey, M. P. DeWalt, and C. R. Spitzer. *Preliminary considerations for classifying hazards of unmanned aircraft systems*. NASA/TM-2007-214539. National Aeronautics and Space Administration. 2007.

[76] H.-M. Huang, E. Messina, and J. Albus. *Autonomy levels for Unmanned Systems (ALFUS) Framework - Volume II: Framework Models*. National Institute of Standards and Technology (NIST). 2007.

[77] International Civil Aviation Organization. *Annex 10 to the Convention on International Civil Aviation: Aeronautical telecommunications*. 5th ed. ICAO. Montréal, Canada, 1996.

[78] International Civil Aviation Organization. *Annex 11 to the Convention on International Civil Aviation: Air Traffic Services*. 13th ed. ICAO. Montréal, Canada, 2001.

[79] International Civil Aviation Organization. *Annex 2 to the Convention on International Civil Aviation: Rules of the Air*. 10th ed. ICAO. Montréal, Canada, 2005.

[80] International Civil Aviation Organization. *Annex 6 to the Convention on International Civil Aviation: Operation of Aircraft*. 8th ed. ICAO. Montréal, Canada, 2001.

[81] International Civil Aviation Organization. *Cir 328 AN/190: Unmanned Aircraft Systems (UAS)*. ICAO. Montréal, Canada, 2011.

[82] International Civil Aviation Organization. *Doc. 10019, AN/507: Manual on Remotely Piloted Aircraft Systems (RPAS)*. 1st ed. ICAO. Montréal, Canada, 2015.

[83] International Civil Aviation Organization. *Doc. 4444, ATM/501: Procedures for Air Navigation Services: Air Traffic Management*. 16th ed. ICAO. Montréal, Canada, 2016.

[84] International Civil Aviation Organization. *Doc. 7300/9, Convention on International Civil Aviation*. 9th ed. ICAO. Montréal, Canada, 2009.

[85] International Civil Aviation Organization. *Doc. 8168, OPS/611: Procedures for Air Navigation Services: Aircraft Operations*. 5th ed. ICAO. Montréal, Canada, 2006.

[86] International Civil Aviation Organization. *Doc. 9613, AN/937: Performance-based Navigation (PBN) Manual*. 4th ed. ICAO. Montréal, Canada, 2013.

[87] International Civil Aviation Organization. *Doc. 9849 AN/457: Global Navigation Satellite System (GNSS) Manual*. 1st ed. ICAO. Montréal, Canada, 2005.

[88] International Civil Aviation Organization. *Doc. 9859, AN/458: Global Air Traffic Management Operational Concept*. 1st ed. ICAO. Montréal, Canada, 2005.

[89] International Civil Aviation Organization. *Doc. 9859, AN/474: Safety Management Manual (SMM)*. 3rd ed. ICAO. Montréal, Canada, 2013.

[90] International Civil Aviation Organization. *Doc. 9869, AN/953: Manual on airspace planning methodology for the determination of separation minima*. 1st ed. ICAO. Montréal, Canada, 1998.

[91] International Civil Aviation Organization. *Doc. 9869: Performance-based Communication and Surveillance (PBCS) Manual*. 2nd ed. ICAO. Montréal, Canada, 2017.

[92] S. Jacklin. "Certification of Safety-Critical Software Under DO-178C and DO-278A". In: *AIAA Infotech @ Aerospace*. AIAA. Garden Grove, California, 2012, pp. 1–13. DOI: 10.2514/6.2012-2473.

[93] M. R. C. Jackson. "Role of avionics in trajectory based operations". In: *27th Digital Avionics Systems Conference (DASC)*. IEEE/AIAA. 2008, 3.A.1–1–3.A.1–9. DOI: 10.1109/DASC.2008.4702792.

[94] Joint Authorities for Rulemaking of Unmanned Systems Working Group 6. *AMC RPAS.1309: Safety Assessment of Remotely Piloted Aircraft Systems*. 2015.

[95] Joint Authorities for Rulemaking of Unmanned Systems Working Group 6. *JARUS guidelines on Specific Operations Risk Assessment (SORA)*. 1st ed. 2017.

[96] Joint Authorities for Rulemaking of Unmanned Systems Working Group 6. *JARUS guidelines on Specific Operations Risk Assessment (SORA)*. 2nd ed. 2019.

[97] M. Kao, G. Weitzel, X. Zheng, and M. Black. "A simple approach to planning and executing complex AUV missions". In: *Symposium on Autonomous Underwater Vehicle Technology*. IEEE. 1992, pp. 95–102. DOI: 10.1109/AUV.1992.225188.

[98] F. Kendoul. "Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems". In: *Journal of Field Robotics* 29.2 (2012), pp. 315–378. DOI: 10.1002/rob.20414.

[99] Y. Kim, M. J. Kochenderfer, J. Grana, J. Bono, and D. Wolpert. "Optimal lost-link policies for unmanned aircraft". In: *34th Digital Avionics Systems Conference (DASC)*. IEEE/AIAA. Prague, Czech Republic, 2015, pp. 1–13. DOI: 10.1109/DASC.2015.7311430.

[100] E. Kiyak and F. Caliskan. "Application of Fuzzy Logic in Aircraft Sensor Fault Diagnosis". In: *International Journal of Systems Applications, Engineering & Development* 6 (2012).

[101] J. Klooster, S. Torres, D. Earman, M. Castillo-Effen, R. Subbu, L. Kammer, D. Chan, and T. Tomlinson. "Trajectory synchronization and negotiation in Trajectory Based Operations". In: *29th Digital Avionics Systems Conference (DASC)*. IEEE/AIAA. 2010, 1.A.3–1–1.A.3–11. DOI: 10.1109/DASC.2010.5655536.

[102] M. J. Kochenderfer, J. K. Kuchar, L. P. Espindle, and J. D. Griffith. *Uncorrelated Encounter Model of the National Airspace System Version 1.0*. Massachusetts Institute of Technology, Lincoln Laboratory. 2008.

[103] M. J. Kochenderfer, J. E. Holland, and J. P. Chryssanthacopoulos. "Next-Generation Airborne Collision Avoidance System". In: *Lincoln Laboratory Journal* 19.1 (2012).

[104] P. Kopardekar. "Safely enabling UAS operations in low-altitude airspace". In: *Unmanned Aerial Systems Traffic Management (UTM) Convention*. National Aeronautics and Space Administration. Moffett Field, California, 2015.

[105] B. Korn and A. Udovic. "File and Fly: Procedures and techniques for integration of UAVs in controlled airspace". In: *25th Congress of International Council of the Aeronautical Sciences (ICAS)*. Hamburg, Germany, 2006.

[106] A. R. Lacher, D. R. Maroney, and A. D. Zeitlin. "Unmanned aircraft collision avoidance: technology assessment and evaluation methods". In: *ATM R&D Seminar*. Barcelona, Spain, 2007.

[107] M. Leucker and C. Schallhart. "A Brief Account of Runtime Verification". In: *The Journal of Logic and Algebraic Programming* 78.5 (2009). DOI: 10.1016/j.jlap.2008.08.004.

[108] J. Li and G. Vachtsevanos. "Human-machine interface: A framework for contingency management of complex aerospace systems". In: *IEEE AUTOTESTCON*. IEEE. National Harbor, Maryland, 2015, pp. 80–86. DOI: 10.1109/AUTEST.2015.7356470.

[109] C. Lum and B. Waggoner. "A risk based paradigm and model for Unmanned Aerial Systems in the National Airspace". In: *AIAA Infotech @ Aerospace*. AIAA. St. Louis, Missouri, 2011, p. 1424. DOI: 10.2514/6.2011-1424.

[110] C. Lum, K. Gauksheim, C. Deseure, J. Vagners, and T. McGeer. "Assessing and estimating risk of operating Unmanned Aerial Systems in populated areas". In: *11th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*. AIAA. Virginia Beach, Virginia, 2011, p. 6918. DOI: 10.2514/6.2011-6918.

[111] G. Manfredi and Y. Jestin. "An introduction to ACAS Xu and the challenges ahead". In: *35th Digital Avionics Systems Conference (DASC)*. IEEE/AIAA. Sacramento, California, 2016, pp. 1–9. DOI: 10.1109/DASC.2016.7778055.

[112] M. Masmano, Y. Valiente, P. Balbastre, I. Ripoll, A. Crespo, and J. Metge. "LithOS: a ARINC-653 guest operating for XtratuM". In: *12th Real-Time Linux Workshop*. Nairobi, Kenia, 2010.

[113] M. Masmano, I. Ripoll, A. Crespo, and J. Metge. "XtratuM: a hypervisor for safety critical embedded systems". In: *11th Real-Time Linux Workshop*. Dresden, Germany, 2009.

[114] F. Mattern. "Virtual Time and Global States of Distributed Systems". In: *Parallel and Distributed Algorithms*. North-Holland, 1989, pp. 215–226.

[115] T. McGeer, L. R. Newcome, and J. Vagners. "Quantitative risk management as a regulatory approach to civil UAVs". In: *International Workshop on UAV Certification*. Paris, France, 1999.

[116] V. B. Mišić and D. M. Velašević. "Formal specifications in software development: An overview". In: *The Yugoslav Journal of Operations Research* 7.1 (1997), pp. 79–96.

[117] NATO Standardization Agency. *STANAG 4586: Standard Interfaces of UA Control System (UCS) for NATO UA Interoperability*. 4th ed. 2017.

[118] NATO Standardization Agency. *STANAG 4671: Unmanned Aerial Vehicles Systems Airworthiness Requirements (USAR)*. 2009.

[119] Official Journal of the European Union. "Regulation (EC) No 216/2008 of 20/02/2008 on common rules in the field of civil aviation and establishing a European Aviation Safety Agency, and repealing Council Directive 91/670/EEC, Regulation (EC) No 1592/2002 and Directive 2004/36/E". 2008.

[120] R. Oliveira. "Formal Specification and Verification of Interactive Systems with Plasticity: Applications to Nuclear-Plant Supervision". PhD thesis. Université Grenoble Alpes, 2015.

[121] J. Park, H. Oh, and M. Tahk. "UAV collision avoidance based on geometric approach". In: *SICE Annual Conference*. IEEE. Tokyo, Japan, 2008, pp. 2122–2126. DOI: 10.1109/SICE.2008.4655013.

[122] E. Pastor, P. Royo, E. Santamaria, M. P. Batlle, C. Barrado, and X. Prats. "An Architecture to Automate UAS Operations in Non-segregated Airspace". In: *1st International Conference on Application and Theory of Automation in Command and Control Systems*. Barcelona, Spain, 2011, pp. 5–16.

[123] E. Pastor, P. Royo, M. P. Batlle, X. Prats, and C. Barrado. "Evaluating technologies and mechanisms for the automated/autonomous operation of UAS in non-segregated airspace". In: *1st SESAR Innovation Days*. Toulouse, France, 2011.

[124] E. Pastor, P. Royo, E. Santamaria, X. Prats, and C. Barrado. "In-Flight Contingency Management for Unmanned Aerial Vehicles". In: *Journal of Aerospace Computing, Information, and Communication* 9.4 (2012), pp. 144–160. DOI: 10.2514/1.55109.

[125] J. Pellebergs. "The MIDCAS project". In: *27th International Congress of the Aeronautical Sciences* (2012).

[126] M. Pérez-Batlle, R. Cuadrado, C. Barrado, P. Royo, and E. Pastor. "Real-time Simulations to Evaluate RPAS Contingencies in Shared Airspace". In: *Proceedings of the 5th SESAR Innovation Days*. Bologna, Italy, 2015.

[127] J. Pérez-Castán, F. G. Comendador, A. Rodríguez-Sanz, R. M. Arnaldo-Valdés, and J. Torrecilla. "Conflict-resolution algorithms for RPAS in non-segregated airspace". In: *Aircraft Engineering and Aerospace Technology* 91.2 (2019), pp. 366–372. DOI: 10.1108/AEAT-01-2018-0024.

[128] J. Pérez-Castán, F. G. Comendador, A. Rodríguez-Sanz, I. A. Cabrera, and J. Torrecilla. "RPAS conflict-risk assessment in non-segregated airspace". In: *Safety Science* 111 (2019), pp. 7–16. DOI: 10.1016/j.ssci.2018.08.018.

[129] B. Potter. *Complying with DO-178C and DO-331 using Model-Based Design*. MathWorks, Inc. 2012.

[130] M. O. Rabin. "Probabilistic automata". In: *Information and Control* 6.3 (1963), pp. 230–245. DOI: 10.1016/S0019-9958(63)90290-0.

[131] Radio Technical Commission for Aeronautics. *DO-178C/ED-12C Software Considerations in Airborne Systems and Equipment Certification*. Washington, D.C., 2011.

[132] Radio Technical Commission for Aeronautics. *DO-185B Minimum Operational Performance Standards for Traffic Alert and Collision Avoidance System II (TCAS II)*. Washington, D.C., 2008.

[133] Radio Technical Commission for Aeronautics. *DO-331/ED-218 Model-Based Development and Verification Supplement to DO-178C and DO-278A*. Washington, D.C., 2011.

[134] Radio Technical Commission for Aeronautics. *DO-333/ED-216 Formal Methods Supplement to DO-178C and DO-278A*. Washington, D.C., 2011.

[135] Range Commanders Council. *Document 323-99. Range safety criteria for Unmanned Air Vehicles*. 1999.

[136] L. R. Ribeiro and N. M. R. Oliveira. "UAV autopilot controllers test platform using Matlab/Simulink and X-Plane". In: *IEEE Frontiers in Education Conference (FIE)*. IEEE. Arlington, Virginia, 2010, S2H–1–S2H–6. DOI: 10.1109/FIE.2010.5673378.

[137] C. L. Robertson. *Determining appropriate levels of automation: FITS SRM Automation Management Research*. FAA-Industry Training Standards (FITS). 2010.

[138] R. Román Cordón. "Future intensive use of UASs for civil and military applications in non-segregated airspace–GCS". PhD thesis. Madrid, Spain: Universidad Politécnica de Madrid, 2018.

[139] R. Roosien and T. van Birgelen. *Introducing automation in aviation: Lessons learned for self-driving vehicles*. Netherlands Aerospace Centre. 2017.

[140] SAE International. *ARP4754A: Guidelines For Development of Civil Aircraft and Systems*. 2010.

[141] SAE International. *J3016: Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems*. 2014.

[142] SESAR Joint Undertaking. *Demonstrating RPAS integration in the European aviation system: A summary of SESAR drone demonstration project results*. 2016.

[143] SESAR Joint Undertaking. *European ATM Master Plan: the roadmap for delivering high performing aviation for Europe*. 3rd ed. 2015.

[144] SESAR Joint Undertaking. *SESAR Concept of Operations at a Glance*. 2nd ed. 2011.

[145] E. Santamaria, C. Barrado, and E. Pastor. "An Event Driven Approach for Increasing UAS Mission Automation". In: *AIAA Infotech @ Aerospace*. AIAA. Seattle, Washington, 2009, pp. 1–21. DOI: 10.2514/6.2009-2044.

[146] E. Santamaria, C. Barrado, E. Pastor, P. Royo, and E. Salami. "Reconfigurable automated behavior for UAS applications". In: *Aerospace Science and Technology* 23.1 (2012), pp. 372–386. DOI: 10.1016/j.ast.2011.09.005.

[147] L. Save, B. Feuerberg, and E. Avia. "Designing human-automation interaction: a new level of automation taxonomy". In: *Human Factors: a view from an integrative perspective*. 2012, pp. 43–56.

[148] A. Saxena, M. E. Orchard, B. Zhang, G. Vachtsevanos, L. Tang, Y. Lee, and Y. Wardi. "Automated Contingency Management for Propulsion Systems". In: *European Control Conference (ECC)*. IEEE. Kos, Greece, 2007, pp. 3515–3522.

[149] S. Schirmer, C. Torens, and F. Adolf. "Formal Monitoring of Risk-based Geofences". In: *AIAA Infotech @ Aerospace, AIAA SciTech*. AIAA. Kissimmee, Florida, 2018. DOI: 10.2514/6.2018-1986.

[150] B. I. Scott. *The Law of Unmanned Aircraft Systems: An Introduction to the Current and Future Regulation Under National, Regional and International Law*. Kluwer Law International, 2016.

[151] A. V. Shelley. "A model of human harm from a falling Unmanned Aircraft: Implications for UAS regulation". In: *International Journal of Aviation, Aeronautics, and Aerospace* 3.3 (2016), p. 1. DOI: 10.15394/ijaaa.2016.1120.

[152] T. B. Sheridan and W. L. Verplank. *Human and computer control of undersea teleoperators*. Massachusetts Institute of Technology. 1978.

[153] R. J. Shively, A. Hobbs, B. Lyall, and C. Rorie. *Human performance considerations for Remotely Piloted Aircraft Systems (RPAS)*. Remotely Pilot Aircraft Systems Panel (RPASP). 2015.

[154] O. Smith. "10.000 and counting: the most successful jet aircraft of all time". In: *The Telegraph* (2018).

[155] M. Solé, E. Pastor, M. Pérez, E. Santamaria, and C. Barrado. "Dynamic Flight Plan Design for UAS Remote Sensing Applications". In: *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*. AIAA. Orlando, Florida, 2010, pp. 1–23. DOI: 10.2514/6.2010-418.

[156] C. R. Spitzer, U. Ferrell, and T. Ferrell, eds. *Digital Avionics Handbook*. 3rd ed. CRC Press LLC, 2014. ISBN: 9781439868614.

[157] R. Stansbury, T. Wilson, and W. Tanis. "A Technology Survey of Emergency Recovery and Flight Termination Systems for UAS". In: *AIAA Infotech @ Aerospace, AIAA SciTech*. AIAA. Seattle, Washington, 2009, pp. 2038–2045. DOI: 10.2514/6.2009-2038.

[158] C. Stöcker, R. Bennett, F. Nex, M. Gerke, and J. Zevenbergen. "Review of the Current State of UAV Regulations". In: *Remote Sensing* 9.5 (2017). DOI: 10.3390/rs9050459.

[159] L. Tang, G Kacprzynski, K. Goebel, J. Reimann, M. E. Orchard, A. Saxena, and B. Saha. "Prognostics in the Control Loop". In: *AAAI Fall Symposium on Artificial Intelligence for Prognostics*. AAAI. Arlington, Virginia, 2007.

[160] L. Tang, A. Saxena, M. E. Orchard, G. J. Kacprzynski, G. Vachtsevanos, and A. Patterson-Hine. "Simulation-based Design and Validation of Automated Contingency Management for Propulsion Systems". In: *Aerospace Conference*. IEEE. Big Sky, Montana, 2007, pp. 1–11.

[161] S. Taraglio, V. Nanni, and D. Taurino. "A possible solution to introduce UAVs into non segregated areas". In: *1st International Conference on Application and Theory of Automation in Command and Control Systems*. Barcelona, Spain, 2011, pp. 125–132.

[162] G. Taylor, B. Purman, P. Schermerhorn, G. Garcia-Sampedro, R. Hubal, K. Crabtree, A. Rowe, and S. Spriggs. "Multi-modal interaction for UAS control". In: *SPIE 9468, Unmanned Systems Technology XVII*. 2015. DOI: 10.1117/12.2180020.

[163] C. Torens and F. Adolf. "Using Formal Requirements and Model-Checking for Verification and Validation of an Unmanned Rotorcraft". In: *AIAA Infotech @ Aerospace, AIAA SciTech*. AIAA. 2015, pp. 1645–1657. DOI: 10.2514/6.2015-1645.

[164] C. Torens, F. Adolf, and L. Goormann. "Certification and Software Verification Considerations for Autonomous Unmanned Aircraft". In: *Jour-*

nal of Aerospace Information Systems 11.10 (2014), pp. 649–664. DOI: 10.2514/1.I010163.

[165] H. Usach, B. Fons, J. Vila, and A. Crespo. "An Autopilot Testbed for IMA (Integrated Modular Avionics) Architectures". In: *IFAC Proceedings Volumes* 46.19 (2013), pp. 435–440. DOI: 10.3182/20130902-5-DE-2040.00076.

[166] H. Usach, J. Vila, A. Crespo, and P. Yuste. "Automatic Deployment of an RPAS Mission Manager to an ARINC-653 Compliant System". In: *Journal of Intelligent & Robotic Systems* 92 (2018), pp. 587–598. DOI: 10.1007/s10846-017-0694-3.

[167] H. Usach. "Integridad y tolerancia a fallos en sistemas de aviónica". MA thesis. Universitat Politècnica de València, 2014.

[168] A. Washington, R. A. Clothier, and J. Almeida da Silva. "A review of unmanned aircraft system ground risk models". In: *Progress in Aerospace Sciences* 95 (2017), pp. 24–44. DOI: 10.1016/j.paerosci.2017.10.001.

[169] A. Washington, R. A. Clothier, and B. P. Williams. "A Bayesian approach to system safety assessment and compliance assessment for Unmanned Aircraft Systems". In: *Journal of Air Transport Management* 62 (2017), pp. 18–33. DOI: 10.1016/j.jairtraman.2017.02.003.

[170] M. Webster, N. Cameron, M. Jump, and M. Fisher. "Towards certification of autonomous unmanned aircraft using formal model checking and simulation". In: *AIAA Infotech @ Aerospace*. AIAA. Garden Grove, California, 2012, pp. 1–15. DOI: 10.2514/6.2012-2573.

[171] R. E. Weibel. "Safety considerations for operation of Unmanned Aerial Vehicles in the National Airspace System". M.Sc. Thesis. Cambridge, Massachusetts: Massachusetts Institute of Technology, 2005.

[172] G. Wild, J. Murray, and G. Baxter. "Exploring civil drone accidents and incidents to help prevent potential air disasters". In: *Aerospace* 3.3 (2016). DOI: 10.3390/aerospace3030022.

[173] A. P. Williams and P. D. Scharre, eds. *Autonomous Systems: Issues for Defense Policymakers*. Norfolk, Virginia: NATO Supreme Allied Command Transformation, 2015.

[174] I. A. Wilson. "Integration of UAS in existing Air Traffic Management systems connotations and consequences". In: *2018 Integrated Communications, Navigation, Surveillance Conference (ICNS)*. IEEE. Herndon, Virginia, 2018. DOI: 10.1109/ICNSURV.2018.8384851.

[175] S. Wood. *Flight Crew Reliance on Automation*. Civil Aviation Authority-Safety Regulation Group. 2004.

[176] fentISS. *XtratuM Hypervisor Emulator: XtratuM Emulator (SKE) Start Guide*. 2015.

Appendix A

Formal design and verification of the contingency management policy: a case study

A.1 Introduction

This appendix illustrates the use of model checking to verify the correctness of a specific contingency management policy. Based on the architectural choice in Sec. 4.3, the contingency management policy proposed in this work depends on the Safety Monitor state automaton and on the Contingency Plan. In addition, the decisions made by the Contingency Plan rely on some parameters specified in the Mission Plan, see Fig. 4.9. Accordingly, in this appendix we will develop a specification for the Safety Monitor, the Contingency Plan and the Mission Plan models, and we will identity some properties with which these models must comply. Then, we will translate both the models and the properties into a formal language and use the NuSMV model checking tool [24] to analyze the formal model with regards to the properties. The resulting process will ensure that the proposed design reaches a certain level of quality before it is implemented in the prototype application of Appendix B.

A.2 Specification of the Safety Monitor model

In this section, the generic Safety Monitor state machine in Fig. 4.7 will be particularized to cope with the contingencies and emergencies identified in Sec. 3.2.2. In short, the proposed contingencies or fault hypothesis include:

1. *C2 link loss*
2. *GNSS loss of performance*
3. *Loss of control in-flight*
4. *Loss of separation*
5. *Mission boundary limits violation*

In addition, the previous contingencies can potentially cause the following emergency conditions:

1. *NMAC*
2. *Operational boundary limits violation*

Then, the Safety Monitor must diagnose each of the previous events and decide whether the resulting state is to be handled by the Contingency Manager or by the Flight Termination System. In this case study, we determine that the occurrence of one single contingency results in an abnormal state; abnormal states will be addressed by the Contingency Manager. Conversely, any combination of nested contingencies or the occurrence of an emergency event results in an emergency state; emergency states require instant flight termination. The resulting centralized Safety Monitor FSM is shown in Fig. A.1.

The proposed FSM has seven states: the nominal state (S_1), five abnormal states (S_2 to S_6, one per contingency under study) and one emergency state (S_7, representing all possible out of control conditions). With respect to the abnormal states, *Autonomous operation* (S_2) is entered after the C2 link loss; *Degraded navigation* (S_3) is entered after the GNSS loss of performance; *Degraded control* (S_4) is entered after the loss of control in-flight; *Traffic alert* (S_5) is entered after the loss of separation; and *Out of mission volume* (S_6) is entered after the mission boundary limits violation.

Transitions between these states are labeled as g_{ij}, where i is the initial state and j is the resulting state. Note that transitions where $i > j$ represent *recovery*

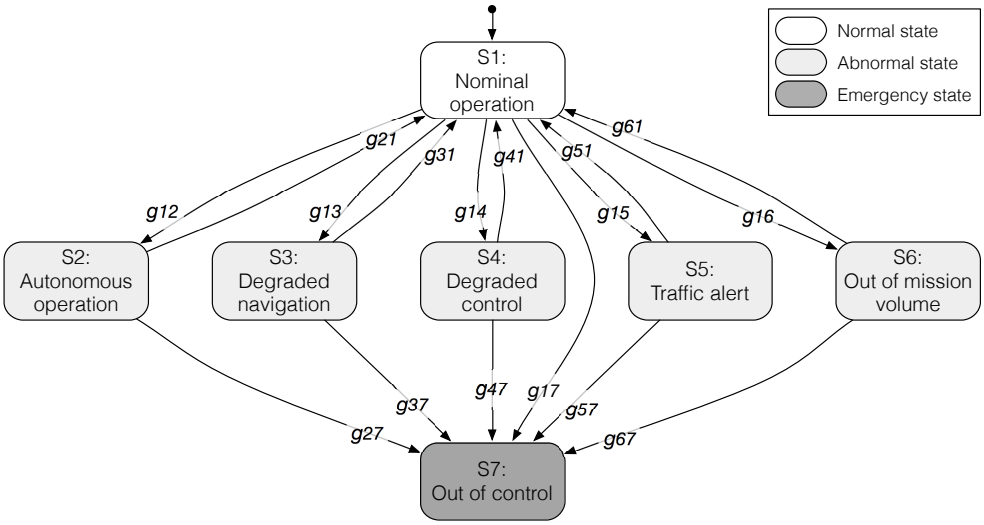


Figure A.1: Centralized Safety Monitor model for the prototype application.

events: events that make the system to evolve from an abnormal state to the nominal state. By contrast, the emergency state is an unrecoverable state: when this state is entered, no transition can make the system to evolve to a different state.

Therefore, some important model requirements or properties related to the proposed Safety Monitor state machine are stated next:

SM1 *There should always be a transition for reaching the* out of control *state in one step.* In other words, the Safety Monitor should always be able to trigger the flight termination action.

SM2 *The out of control state shall be a final state* (i.e. one that has no successors). By definition, emergency states are unrecoverable states [95].

SM3 *A contingency state shall not be reachable from another contingency state in one step.* In other words, nested contingencies will not be handled by the Contingency Manager.

A.3 Specification of the Contingency Plan model

The Contingency Plan specifies the decision logic that should map an abnormal state with the most convenient contingency procedure in a deterministic manner. In this case study, the abnormal states that will be studied are those identified by the Safety Monitor; and the possible contingency procedures are those gathered in Table 3.8 (see Sec. 3.2.6).

The specification of the Contingency Plan is a complex and sensitive task. Although in some cases the decision logic can be straightforward (for example, in a degraded control mode, reverting to manual control is probably the only plausible option), most of the times there could exist multiple contingency procedures with the potential of mitigating the effect of a contingency. For example, after the loss of the C2 link, land at a designated landing site, climb trying to regain the signal and flight termination are acceptable options [82]. In these cases, the decision logic should try to determine the most convenient alternative.

Moreover, it is necessary to account that the most convenient alternative could be defined in terms of the one with the highest probability of success, or in terms of the one that minimizes the time of flight, or the risk exposure, etc; so the problem of designing the Contingency Plan becomes a multi-objective optimization problem. In this case study, for simplicity, we will develop a Contingency Plan design where the "only" optimized variable is the risk entailed for third parties. In future stages of the research, a multi-objective approach will be carried out. As a result, we will exploit the risk models in Chapter 6 to support the decision making problem.

To start with, in order to ensure a safe and correct decision logic, it is necessary to account that each of the proposed contingency procedures can be executed under specified conditions only:

CP1 Revert to manual control *shall not be executed in the autonomous operation state*. This seems evident since the remote pilot is out of the control loop in such degraded condition.

CP2 Collision avoidance *shall not be executed in the degraded navigation state nor in the degraded control state*. This is because of the high flight performance required to perform the evasion maneuver.

CP3 Loiter *shall not be executed in the degraded control state.* In this degraded mode, the inability to perform automatic control makes this option unfeasible.

CP4 Climb to regain signal *shall only be executed in the autonomous state or in the degraded navigation state.* This contingency option is senseless for other abnormal states.

CP5 Land *shall only be executed in the autonomous operation state.* On one hand, performing an automatic landing requires a high flight performance, so this option is unfeasible in degrade navigation or degraded control modes. On the other hand, this option is senseless for loss of separation or mission boundary violation conditions; so autonomous operation is the only remaining abnormal state.

CP6 Flight termination *can be executed in every flight condition.* This is a safety requirement that was motivated in Sec. 4.3.

Based on these requirements, we envision the following decision logic:

- In the *Autonomous operation* state, the ICAO guidelines for contingency management specify that the most suitable options are *climb to regain the signal*, *land* at a designated landing site, or *flight termination* [82]. Among these options, we propose prioritizing climb and land procedures over flight termination since flight termination should be the last resort when no other option is feasible. Then, we will address the decision making problem by exploiting the Reconfigurable Mission Plan concept: since the possible routes to achieve each of the previous goals are specified in the Mission Plan, it is possible to analyze what goal types are reachable from the current position of the RPAS, and what is the risk entailed when flying the route to achieve these goals. Therefore, we suggest the following decision logic, schematized in Fig. A.2:
  - When both climb and land goals are reachable from the current position of the RPAS, the most convenient option will be the one whose associated route has the minimum cost (i.e the one that minimizes the risk entailed for third parties).
  - When only one among climb and land is reachable from the current position of the RPAS, then this option will be selected.

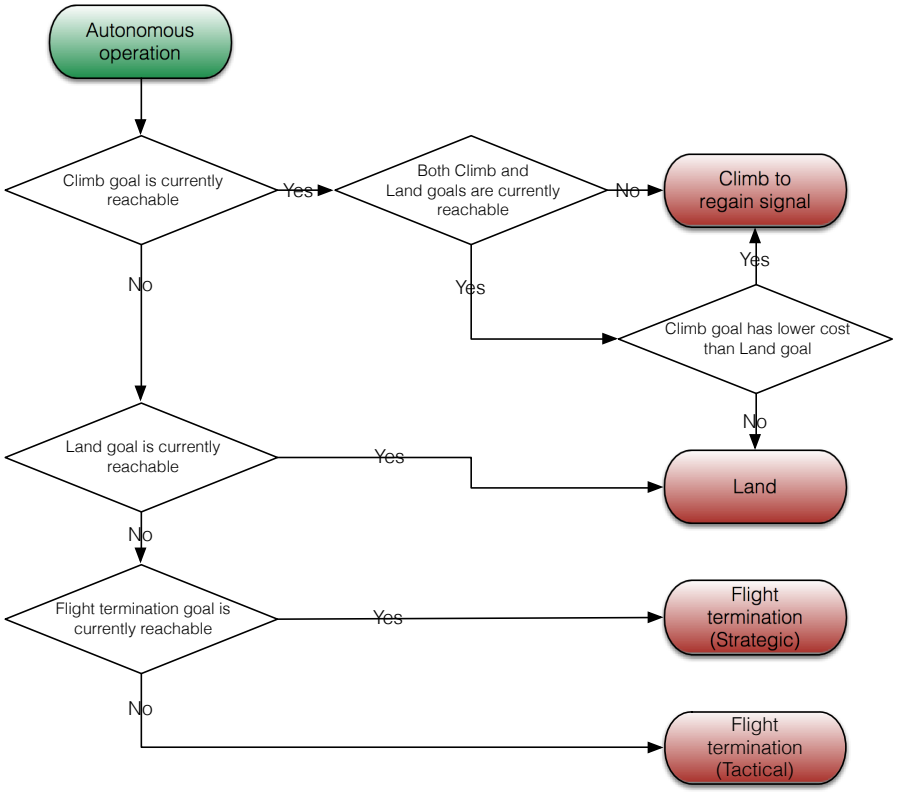


Figure A.2: Autonomous operation's decision logic.

- If neither climb nor land goals are reachable from the current position of the RPAS, then *flight termination* will be used as a backup. When this occurs:
  - If there exists a route that allows to achieve a Flight Termination Point from the current position of the RPAS, then the flight termination action will be preceded by a strategic phase where the RPAS tries to reach this safe area;
  - Otherwise, instant flight termination will be carried out.

- In the *Degraded navigation* state, if the RPAS is flying in controlled airspace, PBN specifications require to disengage the autopilot and *revert to manual control* [86]. Otherwise, we consider that *climb to regain the signal* or *loitering* are the most convenient options. As in the *Autonomous*

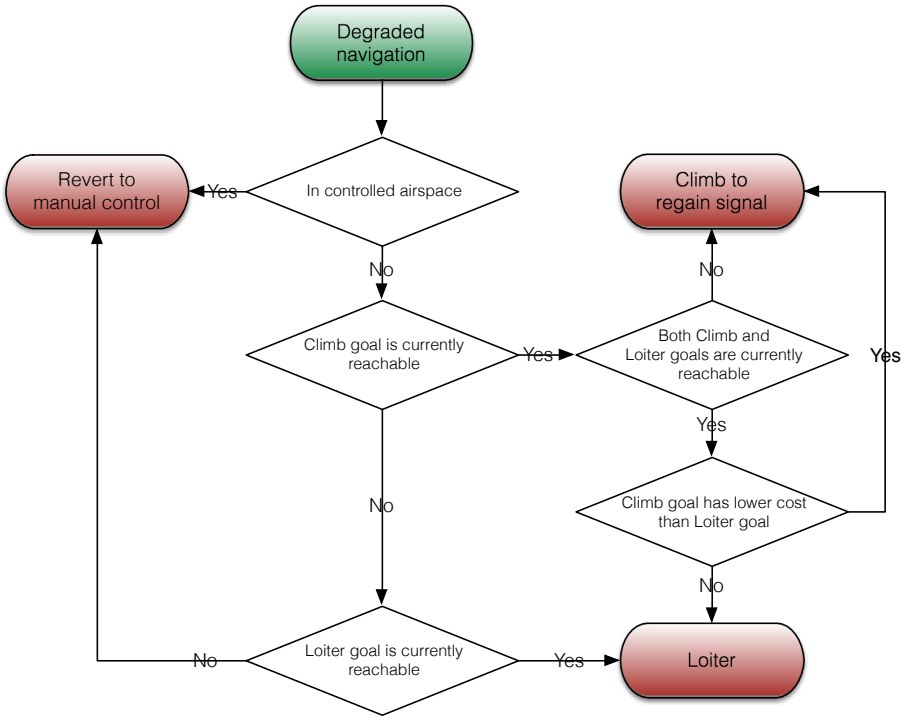


Figure A.3: Degraded navigation's decision logic.

operation problem, the possible routes to achieve these goal types are also specified in the Mission Plan, so we propose the following decision logic, schematized in Fig. A.3:

  - When both goal types are reachable from the current position of the RPAS, the most convenient option will be the one whose associated route has the minimum cost (i.e the one that minimizes the risk for third parties).
  - When only one of these goals is reachable from the current position of the RPAS, then this option will be selected.
  - If none of these goal types are reachable from the current position of the RPAS, then *revert to manual control* will be used as a backup.

- In the *Degraded control* state, we consider that *reverting to manual control* is the only feasible option, see Fig. A.4. This procedure should be per-

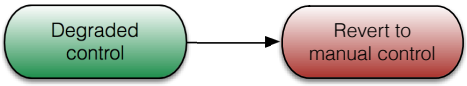


Figure A.4: Degraded control's decision logic.

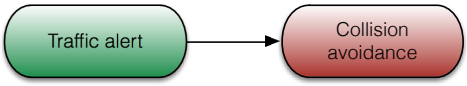


Figure A.5: Traffic alert's decision logic.

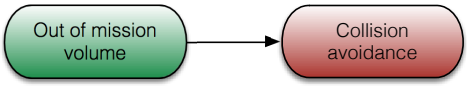


Figure A.6: Out of mission volume's decision logic.

formed without resulting in a transient that requires exceptional piloting skill or alertness from the RPA crew [45].

- In the *Traffic alert* state, the RPAS shall start a *collision avoidance* maneuver to regain the separation minima, see Fig. A.5. If this is achieved, the original mission can resumed afterwards.
- In the *Out of mission volume* state, the RPAS shall start a *collision avoidance* maneuver to go back inside the mission limits, see Fig. A.6. If this is achieved, the original mission can be automatically resumed afterwards.

A.4 Specification of the Mission Plan model

Although Chapter 5 develops the detailed model of a Reconfigurable Mission Plan, it is not the aim of this section to verify the formal model of a Reconfigurable Mission Plan. Rather, we are simply interested in modeling those Reconfigurable Mission Plan attributes that are directly related with the decision-making process of the Contingency Plan. In this regard, from the above discussion, the Mission Plan can be viewed as object that provides the following parameters:

1. Whether the airspace class is controlled or not;

2. Whether a given type of mission goal is reachable from the current position of the RPAS; and

3. In case that a certain goal type is reachable, the risk entailed for third parties when the RPAS flies the route to achieve this goal.

A.5 Formal specification and verification of the policy

In order to verify the proposed contingency management policy using the model checking techniques, it is necessary to translate the previous specifications into a formal language. In this case, we will use the Symbolic Model Verifier (SMV) language, and NuSMV as the model checking tool. The proposed implementation is composed of four modules: the `safetyMonitor` module, the `contingencyPlan` module, the `missionPlan` module and a `main` module that instantiates the previous ones. An extract is shown in Listing A.1. Note that the order of module definitions in the input file is not relevant in SMV [24].

In short, all SMV modules in Listing A.1 start with a declaration of the state variables using the `VAR` command. This is followed by the specification of the FSM that describes the behavior of each model. In SMV, the FSM is declared using the `ASSIGN` command, followed by the initial value `init(state)` and the list of transitions `next(state)`. The last part of each module specifies the properties with which the model must comply. In this case, the properties will be expressed using Computation Tree Logic (CTL) formulae, where `CTLSPEC` defines the start of a CTL expression, and "`->`" is the logical implication operator. The remaining notation is described in [24]. Next paragraphs provide the most relevant details of the different modules.

Safety Monitor module

The extract of the `safetyMonitor` module is shown in lines 1 to 32 of Listing A.1. In particular, lines 9 to 23 show the implementation of some transitions of the FSM in Fig. A.1 (specifically transitions from the `nominal` and `autonomousOperation` states). Then, lines 28 to 32 show the implementation of some model properties identified in Sec. A.4 using the SMV language. For example, line 28 specifies that if a `contingencyEvent` occurs when flying in a `nominal` state, then the next state shall be an `abnormalState`; lines 30 to 32 specify that the `outOfControl` state shall be always reachable, that it must be reachable in one step, and that it is a final state. Note that these prop-

erties have been specified using symbolic declarations like `contingencyEvent`, or `abnormalState`, which are also shown in the extract.

Contingency Plan module

The extract of the `contingencyPlan` module is shown in lines 36 to 76 of Listing A.1. Note that this module is invoked with two input parameters: the current state of the `safetyMonitor` module (variable `smState`) and the `missionPlan` instance (variable `mp`). These are the variables on which the decision logic of the Contingency Plan relies. Then, the FSM describing the proposed logic is shown in lines 39 to 67. For example, the `contingencyProc` selected during the `autonomousOperation` can be either `climb`, `land`, or `flightTermination`, see lines 45 to 50: the decision depends on which goals are currently reachable, and on the expected level of risk of each feasible option. The last part of this extract shows the preconditions for activating each `contingencyProcedure`, see lines 68 to 76. In this case, they have been formalized using CTL formulae of the form `AG(s | AX !p)`; meaning that each occurrence of condition `p` (the selection of a mission goal) is preceded by condition `s` (the required state condition) [13].

Mission Plan module

The extract of the `missionPlan` module is shown in lines 80 to 100 of Listing A.1. The implementation of this module basically consists on the declaration of the Mission Plan variables related with the decision making process: *1)* whether the RPAS is flying `inControlledAirspace`; *2)* which of the mission goals in a Reconfigurable Mission Plan are currently reachable (boolean variables `climbGoalIsReachable`, `loiterGoalIsReachable`, `landGoalIsReachable` and `flightTerminationGoalIsReachable`); and *3)* what is the risk associated to each of the possible goals (`climbRisk`, `loiterRisk`, `landRisk` and `flightTerminationRisk`). Note that the risk associated to each goal type is defined in a scale from 1 to 4, where this index represents the relative position in a risk scale. For example, value 1 is assigned to the option with the lowest risk, while value 4 is assigned to the option with the highest risk.

```
########################################################
The transition relation is total: No deadlock state exists
########################################################
-- specification AG ((state = nominal & event = c2LinkLoss) -> AX state = autonomousOp) IN sm is true
-- specification AG ((state = nominal & event = gnssLossOfPerformance) -> AX state = degradedNav) IN sm is tru
e
-- specification AG ((state = nominal & event = lossOfControl) -> AX state = degradedControl) IN sm is true
-- specification AG ((state = nominal & event = lossOfSeparation) -> AX state = trafficAlert) IN sm is true
-- specification AG ((state = nominal & event = missionBoundaryViolation) -> AX state = outOfMissionVolume) IN
 sm is true
-- specification AG ((state = autonomousOp & event = c2LinkRcv) -> AX state = nominal) IN sm is true
-- specification AG (((state = autonomousOp & contingencyEvent) & event != c2LinkLoss) -> AX state = outOfCont
rol) IN sm is true
-- specification AG ((state = degradedNav & event = gpsRcv) -> AX state = nominal) IN sm is true
-- specification AG (((state = degradedNav & contingencyEvent) & event != gnssLossOfPerformance) -> AX state =
 outOfControl) IN sm is true
-- specification AG ((state = degradedControl & event = controlRcv) -> AX state = nominal) IN sm is true
-- specification AG (((state = degradedControl & contingencyEvent) & event != lossOfControl) -> AX state = out
OfControl) IN sm is true
-- specification AG ((state = trafficAlert & event = trafficSeparated) -> AX state = nominal) IN sm is true
-- specification AG (((state = trafficAlert & contingencyEvent) & event != lossOfSeparation) -> AX state = out
OfControl) IN sm is true
-- specification AG ((state = outOfMissionVolume & event = inMissionBoundary) -> AX state = nominal) IN sm is
true
-- specification AG (((state = outOfMissionVolume & contingencyEvent) & event != missionBoundaryViolation) ->
AX state = outOfControl) IN sm is true
-- specification AG ((state = nominal & contingencyEvent) -> AX abnormalState) IN sm is true
-- specification AG ((abnormalState & recoveryEvent) -> AX state = nominal) IN sm is true
-- specification AG (emergencyEvent -> AX state = outOfControl) IN sm is true
-- specification AG (EF state = outOfControl) IN sm is true
-- specification AG (state = nominal -> EF state != nominal) IN sm is true
-- specification AG (state = autonomousOp -> EF state != autonomousOp) IN sm is true
-- specification AG (state = trafficAlert -> EF state != trafficAlert) IN sm is true
-- specification AG (state = outOfMissionVolume -> EF state != outOfMissionVolume) IN sm is true
-- specification AG (state = degradedNav -> EF state != degradedNav) IN sm is true
-- specification AG (state = degradedControl -> EF state != degradedControl) IN sm is true
-- specification AG (state = outOfControl -> !(EF state != outOfControl)) IN sm is true
-- specification AG (!inAutonomousOp | AX contingencyProc != toManual) IN cm is true
-- specification AG ((!inDegradedNav & !inDegradedControl) | AX contingencyProc != avoidance) IN cm is true
-- specification AG (!inDegradedControl | AX contingencyProc != loiter) IN cm is true
-- specification AG (((!inDegradedControl & !inTrafficAlert) & !inOutOfMissionVolume) | AX contingencyProc !=
climb) IN cm is true
-- specification AG ((((!inDegradedNav & !inDegradedControl) & !inTrafficAlert) & !inOutOfMissionVolume) | AX
contingencyProc != land) IN cm is true
-- specification AG (EF contingencyProc = flightTermination) IN cm is true
```

Figure A.7: Verification results in the NuSMV console.

Formal verification results

The SMV file in Listing A.1 can then be interpreted by NuSMV, which will check if the CTL specifications are satisfied by the model. When a specification is not satisfied, NuSMV provides a counter-example that demonstrates the falsity of a model property. In this case, the output of this program is shown in Fig. A.7. It shows that the transition relation is total, and that all specifications hold. In summary, the results demonstrate the correctness of the proposed policy before the implementation phase.

Listing A.1: Extract of the contingency management policy model in SMV language.

```
1  MODULE safetyMonitor
2  VAR
3       state : {nominal, autonomousOp, degradedNav, degradedControl, ...
            trafficAlert, outOfMissionVolume, outOfControl};
4       event : {c2LinkLoss, gnssLossOfPerformance, lossOfControl, ...
            lossOfSeparation, nmac, missionBoundaryViolation, ...
            perationalBoundaryViolation, c2LinkRcv, gpsRcv, controlRcv, ...
            trafficSeparated, inMissionBoundary};
```

```
5  DEFINE
6      contingencyEvent := (event = c2LinkLoss | event = ...
           gnssLossOfPerformance | event = lossOfControl | event = ...
           lossOfSeparation | event = missionBoundaryViolation);
7      abnormalState := (state = autonomousOp | state = degradedNav | ...
           state =  degradedControl | state = trafficAlert | state = ...
           outOfMissionVolume);
8  -- *** Lines omitted ***
9  ASSIGN -- Safety Monitor state automaton
10     init (state) := nominal;
11     next (state) :=
12         case
13             --In S1: Nominal operation
14             state = nominal & event = c2LinkLoss : autonomousOp; --g12
15             state = nominal & event = gnssLossOfPerformance : ...
                   degradedNav; --g13
16             state = nominal & event = lossOfControl : ...
                   degradedControl; --g14
17             state = nominal & event = lossOfSeparation : ...
                   trafficAlert; --g15
18             state = nominal & event = missionBoundaryViolation : ...
                   outOfMissionVolume; --g16
19             --In S2: Autonomous operation
20             state = autonomousOp & event = c2LinkRcv : nominal; --g21
21             state = autonomousOp & (event = lossOfSeparation | ...
                   event =
22             missionBoundaryViolation | event = ...
                   gnssLossOfPerformance | event =
23             lossOfControl) : outOfControl; --g27
24 -- *** Lines omitted ***
25             state != outOfControl & emergencyEvent : outOfControl; ...
                   --gX7
26             TRUE : state; --Otherwise
27         esac;
28 CTLSPEC AG (state = nominal & contingencyEvent -> AX ...
       abnormalState);
29 CTLSPEC AG (abnormalState & recoveryEvent -> AX state = nominal);
30 CTLSPEC AG (emergencyEvent -> AX state = outOfControl);
31 CTLSPEC AG (state = outOfControl -> ! EF state != outOfControl);
32 CTLSPEC AG EF (state = outOfControl);
33
34 -- ######################################################################
35 -- ######################################################################
36 MODULE contingencyPlan (smState, mp)
37 VAR
38     contingencyProc : {nominalGoal, avoidance, loiter, climb, land, ...
           flightTermination, toManual};
39 ASSIGN
40         init (contingencyProc) := nominalGoal;
41         next (contingencyProc) :=
42             case
43                 --In S1: Nominal operation
```

```
44                  smState = nominal & (contingencyProc = avoidance | ...
                        contingencyProc = toManual) : nominalGoal;
45                  --In S2: Autonomous operation
46                  smState = autonomousOp & mp.climbGoalIsReachable & ...
                        mp.landGoalIsReachable & mp.climbRisk <= ...
                        mp.landRisk : climb;
47                  smState = autonomousOp & mp.climbGoalIsReachable & ...
                        mp.landGoalIsReachable : land;
48                  smState = autonomousOp & mp.climbGoalIsReachable & ...
                        !mp.landGoalIsReachable : climb;
49                  smState = autonomousOp & !mp.climbGoalIsReachable & ...
                        mp.landGoalIsReachable : land;
50                  smState = autonomousOp & !mp.climbGoalIsReachable & ...
                        !mp.landGoalIsReachable : flightTermination;
51                  --In S3: Degraded navigation
52                  smState = degradedNav & mp.inControlledAirspace : ...
                        toManual;
53                  smState = degradedNav & !mp.inControlledAirspace & ...
                        mp.loiterGoalIsReachable & ...
                        mp.climbGoalIsReachable & mp.loiterRisk <= ...
                        mp.climbRisk : loiter;
54                  smState = degradedNav & !mp.inControlledAirspace & ...
                        mp.loiterGoalIsReachable & ...
                        mp.climbGoalIsReachable : climb;
55                  smState = degradedNav & !mp.inControlledAirspace & ...
                        mp.loiterGoalIsReachable & ...
                        !mp.climbGoalIsReachable : loiter;
56                  smState = degradedNav & !mp.inControlledAirspace & ...
                        !mp.loiterGoalIsReachable & ...
                        mp.climbGoalIsReachable : climb;
57                  smState = degradedNav & !mp.inControlledAirspace & ...
                        !mp.loiterGoalIsReachable & ...
                        !mp.climbGoalIsReachable : toManual;
58                  --In S4: Degraded control
59                  smState = degradedControl : toManual;
60                  --In S5: Traffic alert
61                  smState = trafficAlert : avoidance;
62                  --In S6: Out of mission volume
63                  smState = outOfMissionVolume : avoidance;
64                  --In S7: Out of control
65                  smState = outOfControl : flightTermination;
66                  TRUE : contingencyProc; --Otherwise
67              esac;
68  CTLSPEC AG ((! inAutonomousOp) | AX contingencyProc != toManual)
69  CTLSPEC AG ((! inDegradedNav & ! inDegradedControl) | AX ...
        contingencyProc !=
70  avoidance)
71  CTLSPEC AG ((! inDegradedControl) | AX contingencyProc != loiter)
72  CTLSPEC AG ((! inDegradedControl & ! inTrafficAlert & ! ...
        inOutOfMissionVolume) |
73  AX contingencyProc != climb)
74  CTLSPEC AG ((! inDegradedNav & ! inDegradedControl & ! ...
        inTrafficAlert & !
```

```
75  inOutOfMissionVolume) | AX contingencyProc != land)
76  CTLSPEC AG EF contingencyProc = flightTermination;
77
78  -- ###########################################################################
79  -- ###########################################################################
80  MODULE missionPlan
81  VAR
82      inControlledAirspace : boolean;
83      climbGoalIsReachable : boolean;
84      loiterGoalIsReachable : boolean;
85      landGoalIsReachable : boolean;
86      flightTerminationGoalIsReachable : boolean;
87      climbRisk : {1, 2, 3, 4};
88      loiterRisk : {1, 2, 3, 4};
89      landRisk : {1, 2, 3, 4};
90      flightTerminationRisk : {1, 2, 3, 4};
91  ASSIGN
92      next (inControlledAirspace) := {TRUE, FALSE};
93      next (climbGoalIsReachable) := {TRUE, FALSE};
94      next (loiterGoalIsReachable) := {TRUE, FALSE};
95      next (landGoalIsReachable) := {TRUE, FALSE};
96      next (flightTerminationGoalIsReachable) := {TRUE, FALSE};
97      next (climbRisk) := {1, 2, 3, 4};
98      next (loiterRisk) := {1, 2, 3, 4};
99      next (landRisk) := {1, 2, 3, 4};
100     next (flightTerminationRisk) := {1, 2, 3, 4};
101
102 -- ###########################################################################
103 -- ###########################################################################
104 MODULE main
105 VAR
106     mp : missionPlan;
107     sm : safetyMonitor;
108     cm : contingencyPlan (sm.state, mp);
```

Appendix B

On-board Mission Management System software architecture implementation

B.1 Introduction

The present appendix describes the prototyping of a SMMS that is able to fly Reconfigurable Mission Plans and performs ACM functions. In this case, the proposed SMMS follows the software architecture described in Chapter 4 and implements the contingency management policy discussed in Appendix A. The prototype application has been developed in Matlab&Simulink following the MBD methodology, has been tested in a simulation environment based on the X-Plane flight simulator, and will be deployed to an execution environment based on the XtratuM hypervisor using the automatic deployment tools in Appendix D.

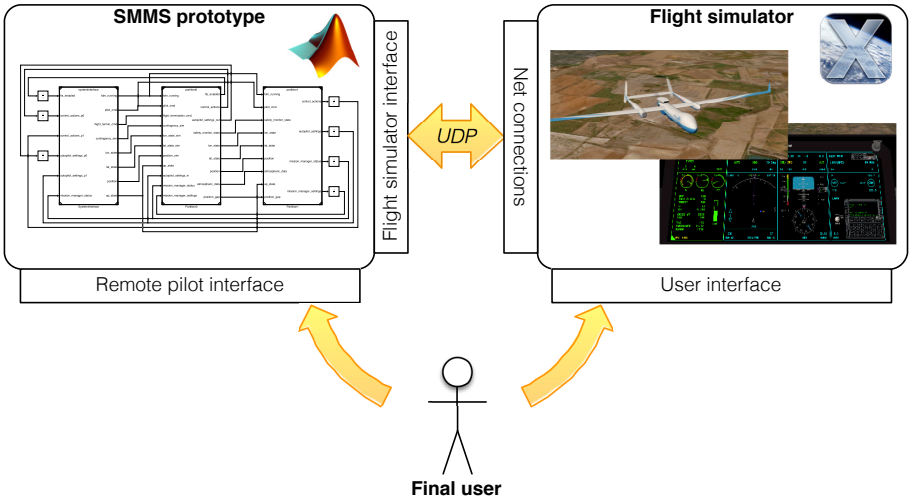


Figure B.1: Simulation environment for the SMMS prototype.

Simulation environment considerations

One of the advantages of the MBD methodology is that it allows to validate the design from the early stages of the development process. In this case, in order to validate the functional behavior of the SMMS prototype, we will execute the proposed system on a simulation environment based on the X-Plane simulator, see Fig. B.1. In this simulation environment, X-Plane provides the flight dynamic model of a wide variety of aircraft, including general aviation aircraft, airliners, military aircraft and also some unmanned aircraft. Among the available unmanned aircraft, we will use the IAI Super Heron model as a representative aircraft for performing the intended ConOps.

But the main reason for using X-Plane is that it provides full access to the simulator's property tree using User Datagram Protocol (UDP) communication. This enables reading and writing flight simulation data (such as the aircraft state, the control actions, etc.) from an external application throughout the so-called *datarefs*[1] [136].

Moreover, most of the aircraft models in X-Plane include their own autopilot system. This way, although in a complete system development, flight control

[1] A list of the X-Plane datarefs is available online at http://www.xsquawkbox.net/xpsdk/docs/DataRefs.html (last accessed on May 2019).

functions must be implemented in the application code, we will delegate the design of the autopilot to the developers of the flight simulation engine. This design decision allows to reduce the system complexity and to put all the development effort in the remaining system components. In conclusion, the outputs of the SMMS prototype will be the control targets for the X-Plane autopilot, not the control actions provided by the control loops.

Next sections describe the SMMS model in Matlab&Simulink with some detail.

B.2 Top-level SMMS model

To start with, it is necessary to account that, when a Simulink model is to be deployed on an XtratuM-based target using the automatic deployment tools in Appendix D, the following design constraint applies: Simulink blocks that you want to allocate to a same partition must be grouped into a same *referenced model* on the top-level view of the Simulink model, see Sec. D.2.1. Based on this premise, if we want the SMMS prototype to match the partitioning scheme represented in Fig. 4.10, the top-level view of the SMMS model must include two referenced models: a first one containing the Safety Monitor model and the Flight Termination System model, and a second one containing the Contingency Manager model and the Mission Manager model, see Sec. 4.4.2.

Additionally, in order for the SMMS prototype to be run in the simulation environment, it is necessary to include a third system partition that models the system interface. This includes the interface between the SMMS and the flight simulator, and the interface between the SMMS and the remote pilot (or final user), see Fig. B.1. Note that this supporting partition does not respond to a need of providing fault isolation but to a need of providing the required services to the application (in this case, Linux services for UDP communication).

The resulting high-level view of the proposed SMMS model is shown in Fig. B.2. It includes three referenced models representing three system partitions: `Partition0`, `Partition1` and `SystemInterface`. `Partition0` includes the `SafetyMonitor` and the `FlightTerminationSystem` models; `Partition1` the `ContingencyManager` and the `MissionManager` models; and `SystemInterface` the `FlightSimulatorInterface` and the `RemotePilotInterface` models. Next, we discuss some implementation details related to these models.

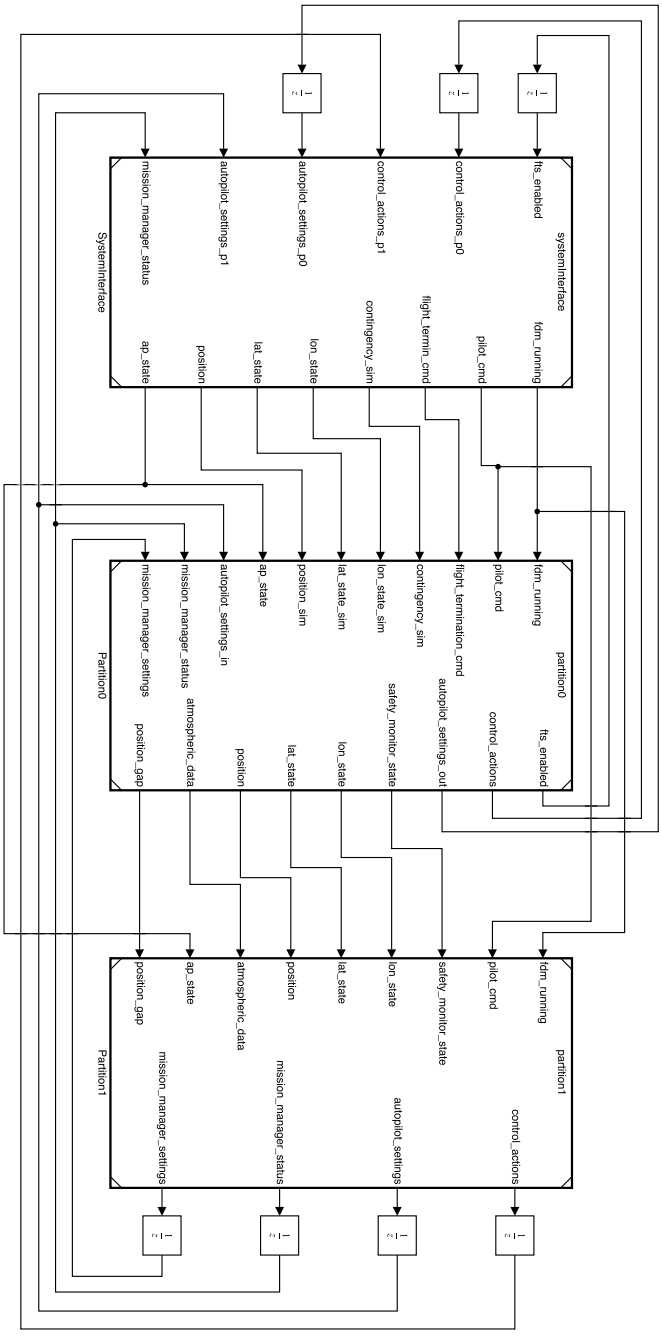


Figure B.2: High-level view of the Safe Mission Management System model in Simulink.

B.3 Flight simulator interface

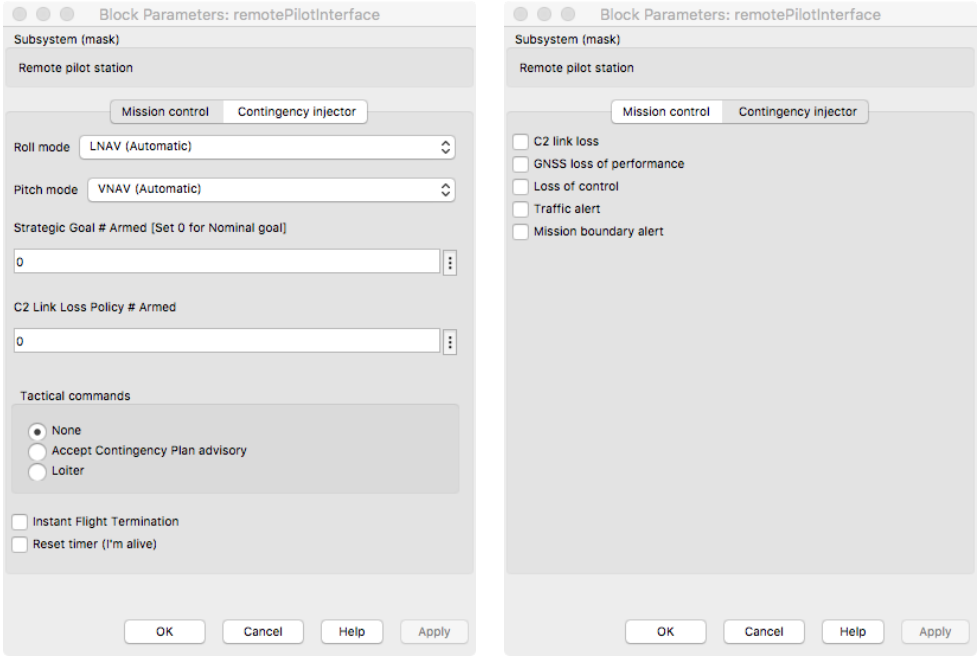
The `FlightSimulatorInterface` allows to connect the SMMS prototype to X-Plane using UDP sockets. In particular, we read up to 28 datarefs related with the flight dynamic model of the aircraft (e.g. aircraft position, altitude, airspeed, pitch, roll and yaw angles, etc.) and also with the world model (e.g. wind speed and direction); and we write up to 17 datarefs that allow to set the MCP provided by X-Plane (Fig. B.4) based on the SMMS directives.

B.4 Remote Pilot interface

In the proposed prototype application, the final user plays the role of the remote pilot. As it was represented in Fig. B.1, in order to control the mission execution, the final user has two separated and complementary system interfaces with the application: a first one implemented in Simulink (see Fig. B.3) and a second one provided by X-Plane (see Fig. B.4):

- On one hand, the remote pilot interface in Simulink (Fig. B.3) is used to provide the SMMS with high-level commands mostly related with the mission goal to be achieved. We have designed a GUI[2] that consists of two control panels: the *mission control* panel and the *contingency injector* panel:
  - The mission control panel allows the final user to define the operational mode for the roll mode and for the pitch mode, and also the strategic goal or the tactical goal to be armed in the SMMS. In particular, when the RPAS is operated in manual or automatic mode, the final user can set which of the goals of the Reconfigurable Mission Plan must be currently armed. This goal must be set on the "strategic goal # armed" field of Fig. B.3a. In addition, the final user can also set which of the goals of the Reconfigurable Mission Plan must be armed in case of C2 link loss. This goal must be set on the "C2 link loss policy # armed" field. The fact that the C2 link loss policy can be defined by the final user through the remote pilot interface provides flexibility to the final user and increases predictability once the RPAS starts operating autonomously. Nevertheless, the SMMS will suggest the remote pilot which C2 link loss

[2] As it can be observed in Fig. B.3, the proposed GUI is an elementary user interface that will be used for demonstration purposes only. The design of an appropriate remote pilot interface is a safety-critical problem that is out of the scope of this work. In this regard, the reader is referred to the work in [138] for a detailed discussion on this topic.



(a) Mission control panel. **(b)** Contingency injector panel.

Figure B.3: Remote pilot interface in Simulink.

Figure B.4: Remote pilot interface in X-Plane.

policy is the most convenient one at each point of the mission based on the directives of the Contingency Plan. In particular, we make use of the Diagnostic Viewer window in Simulink to print certain messages from the SMMS for the remote pilot.

Apart from the strategic commands, the final user can also provide tactical commands. Selecting a tactical command preempts the strategic goal execution. In this case, the final user can accept or reject the tactical goal suggested by the Contingency Plan after a contingency occurs, or trigger a tactical loitering procedure. In addition, the final user can also engage the Flight Termination System in a manual manner using the "instant flight termination" button. Note that, in compliance with the AESA requirement in [7], the instant flight termination signal is sent through an independent communication channel to the SMMS. Note also that, after the C2 link loss, none of the commands sent through the mission control panel will be received in the SMMS.

– The contingency injector panel allows to user to simulate the occurrence of the different contingencies considered in this work. This mechanism will be used for demonstration purposes in Chapter 7.

- On the other hand, the remote pilot interface in X-Plane (Fig. B.4) is used to provide tactical inputs to the Flight Director. When the RPAS is operated in manual mode, the remote pilot has full control over the MCP interface. When the RPAS is operated in automatic mode, the remote pilot can just set the altitude target, the speed target and the vertical speed target. When the RPAS is operated in autonomous mode, the remote pilot is out of the control loop so all buttons and switches in Fig. B.4 are disabled using a configuration file.

B.5 Safety Monitor System model

The `SafetyMonitor` model is shown in Fig. B.5. As it can be observed, the proposed model includes five distributed monitors, named `C2LinkMonitor`, `GnssMonitor`, `LossOfControlMonitor`, `TrafficMonitor` and `BoundaryMonitor`, and a centralized component (`CentralizedSafetyMonitor`) that coordinates the previous ones. As it can be deduced from the model names, each distributed monitor checks the state of a sub-system that is related with an abnormal or emergency state considered in this work. For example, the `C2LinkMonitor` checks C2 link state: when the C2 link is lost, it triggers the

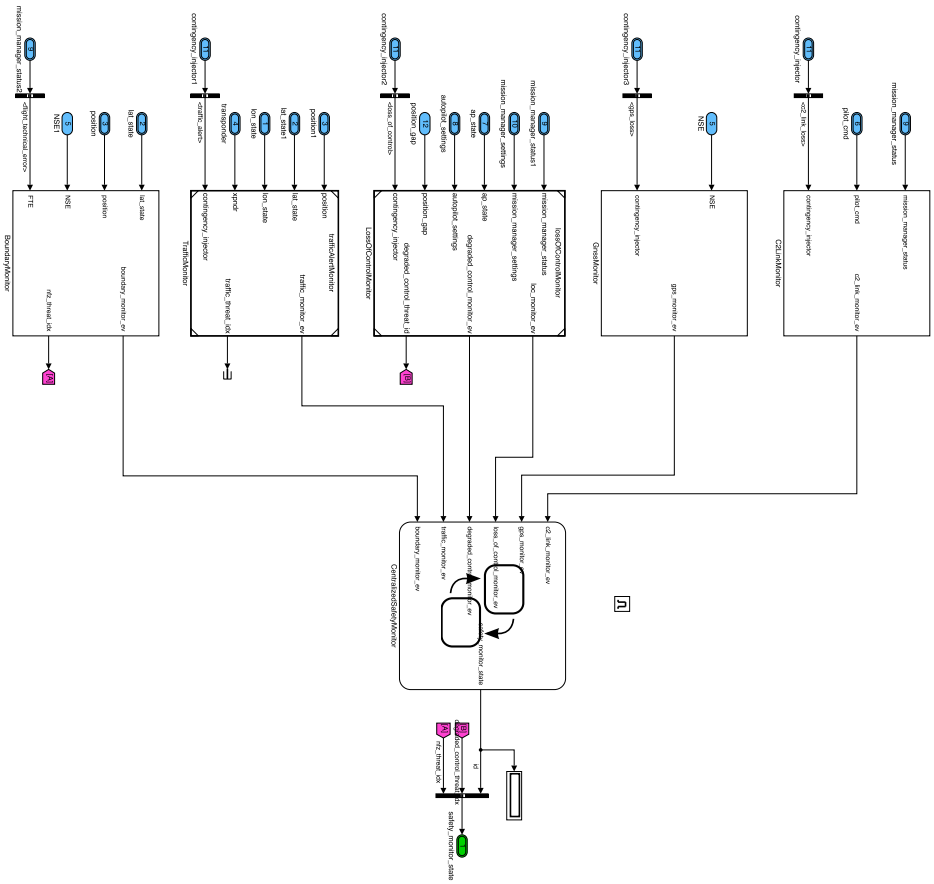


Figure B.5: Safety Monitor model in Simulink.

corresponding contingency event. In the same way, the `GnssMonitor` checks the GNSS performance and triggers the GNSS loss of performance event when specified conditions hold. The `TrafficMonitor` can trigger the loss of separation contingency but also the NMAC emergency, etc.

Event triggering depends on a monitoring function that checks specified conditions related with the unsafe state. For example, the function that monitors the loss of separation condition is based on the detection of the point of closest approach [121]. In this regard, we can supply the `TrafficMonitor` with synthetic traffic data or with real traffic data recorded with an ADS-B receiver located at the UPV. Another example is the monitoring function of the mission boundary violation condition which relies on the obstacle detection algorithm in [68]. Note that we can also simulate the occurrence of a given contingency using the contingency injection mechanism in Fig. B.3b.

Then, the `CentralizedSafetyMonitor` component coordinates all the distributed monitors and runs the FSM in Fig. A.1. As it can be observed, we have modeled the FSM using a Simulink's Stateflow chart. Depending on the active state in this FSM, the `ContingencyManager` model or the `FlightTerminationSystem` model will be engaged.

B.6 Contingency Manager System model

The `ContingencyManager` model is shown in Fig. B.6. In short, this model provides the active mission goal to `MissionManager` model and also defines different Mission Manager settings, like the current operational mode. With this aim, the model diagram in Fig. B.6 is organized as follows:

1. The `GoalReachability` block provides the current reachable goals in the Reconfigurable Mission Plan from the current position of the RPAS. This requires to make use of the Mission Graph object and the graph search tools described in Appendix C.

2. According to the goal reachability results and the current state of the `CentralizedSafetyMonitor` model, the `ContingencyPlan` block suggests the remote pilot the most convenient mission goal. In this case, the `ContingencyPlan` block implements the Contingency Plan design of Appendix A. This Plan is evaluated as follows:

    - Whenever the RPAS is operated in manual or automatic mode, the `ContingencyPlan` suggests the remote pilot which of the goals of

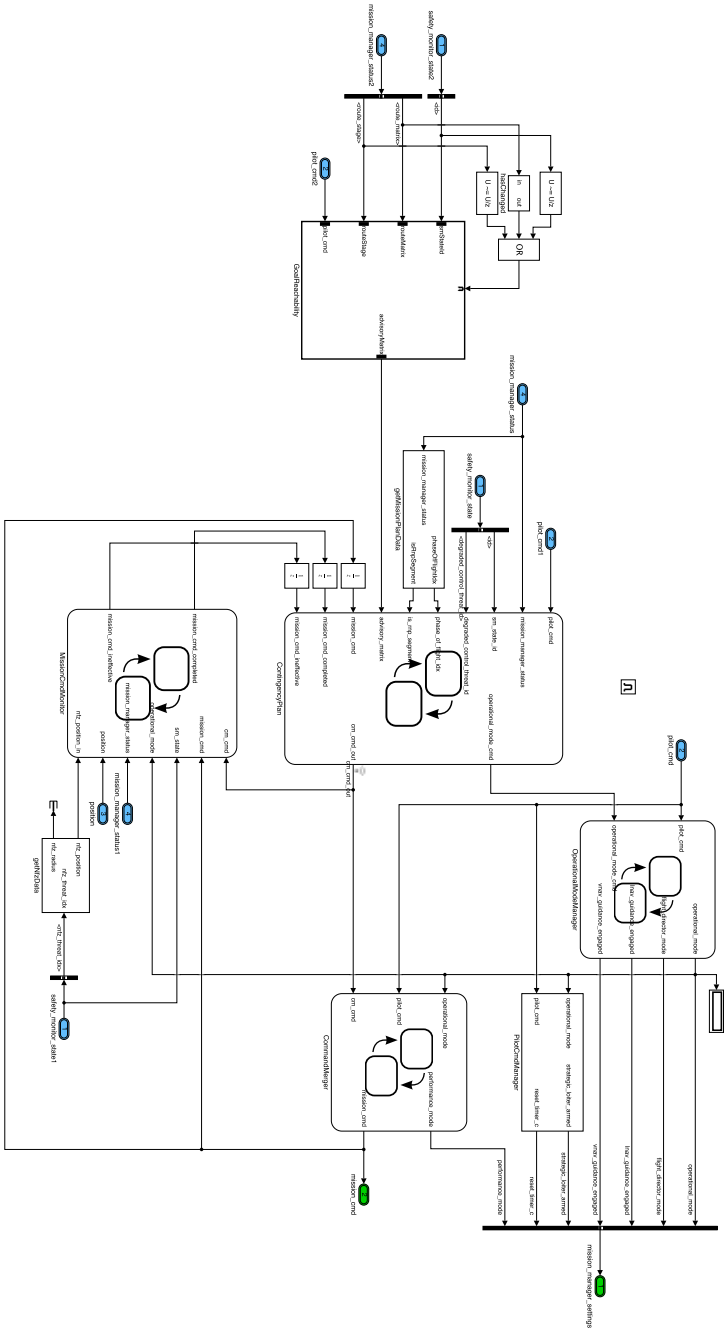


Figure B.6: Contingency Manager model in Simulink.

the Reconfigurable Mission Plan is the most appropriate response for the C2 link loss event. The remote pilot can accept this policy or arm a different one using the "C2 link loss policy armed" field of Fig. B.3a. Note that this goal will not be engaged unless the C2 link loss condition occurs.

- When the `CentralizedSafetyMonitor` enters an abnormal state:
  - If the RPAS is still operated in manual or automatic mode, the `ContingencyPlan` suggests the most convenient policy for this condition. If this policy is a strategic goal or a tactical goal, the remote pilot can accept it or revert to manual control using the mission control panel of Fig. B.3a. If the most convenient policy is actually reverting to manual control, the remote pilot is required to handle the situation.
  - If the RPAS is now operated in the autonomous mode, the `ContingencyPlan` provides the C2 link loss policy armed by the remote pilot.

3. The `OperationalModeManager` block defines the current operational mode depending on the remote pilot settings on the control panel of Fig. B.3a and on the previous Contingency Manager commands. Note that this block is modeled using a Stateflow chart that implements the FSM in Fig. 4.4.

4. The `CommandMerger` block receives the remote pilot commands and the `ContingencyPlan` directives and outputs the mission goal that will be actually sent to the `MissionManager` model based on the following logic: when the RPAS is operated in manual or automatic mode, the output mission goal is the one set by the remote pilot unless she or he accepts the Contingency Plan proposal; when the RPAS is operated in autonomous mode, the output mission goal is always the mission goal provided by the Contingency Plan because the remote pilot is out of the control loop (note, however, that this goal was specified by the remote pilot before the C2 link was lost).

5. Finally, the `MissionCmdMonitor` block evaluates the effectiveness of the output mission goal. In this regard, the output mission goal can be successfully completed or it can turn out to be ineffective (for example, if the lost signal is not recovered after completing the regain signal procedure). When this occurs, the `MissionCmdMonitor` rises a flag so that the `ContingencyPlan` block plans further actions.

B.7 Mission Manager System model

The `MissionManager` model is shown in Fig. B.7. As it can be observed, the internal design of this model is structured into the three layers of the 3T architecture: the `PathPlanner`, the `GuidanceSystem` and the `FlightDirector`. Each of these layers are further discussed next.

B.7.1 Path Planner

The `PathPlanner` model is shown in Fig. B.8. In short, this model is in charge of generating a suitable aircraft trajectory based on the mission goal defined by the Contingency Manager. It is composed of three main modules: the `GoalManager`, the `PathPlannerManager` and the `PathPlannerPolicies`. The `GoalManager` is in charge of handling the strategic goal selected by the `ContingencyManager`. In this respect, it tries to find a route in the Reconfigurable Mission Plan that is effective to achieve the strategic goal armed. If a feasible route is found, then the strategic goal will be engaged and the corresponding route will be processed by the underlying components of the architecture. If the selected goal is not achievable from the current position of the RPAS, then a flag will be risen so that the upper components of the architecture or the remote pilot handle the situation. Note that, in order to find the corresponding route, this module exploits the Mission Graph and the dynamic route configuration tools described in Appendix C.

In addition, the `GoalManager` is also responsible for monitoring the goal execution: it checks if a goal stage is reached; when the goal has multiple stages, then it sequences the different goal stages; and when all goal stages are completed, it rises a flag to notify it to the remaining system components.

Then, the `PathPlannerManager` is in charge of activating the path planning policy depending on the active mission goal and the current goal stage. Finally, the `PathPlannerPolicies` contain the different instances of path planners that generate the aircraft trajectory based on different criteria. For example, the `MissionPlanner` sequences the different legs of the route found by the `GoalManager`; the `LoiterPlanner` computes the reference trajectory of the holding procedure, etc. In all cases, the output reference trajectory is specified using one of the EPTs of Table 5.3.

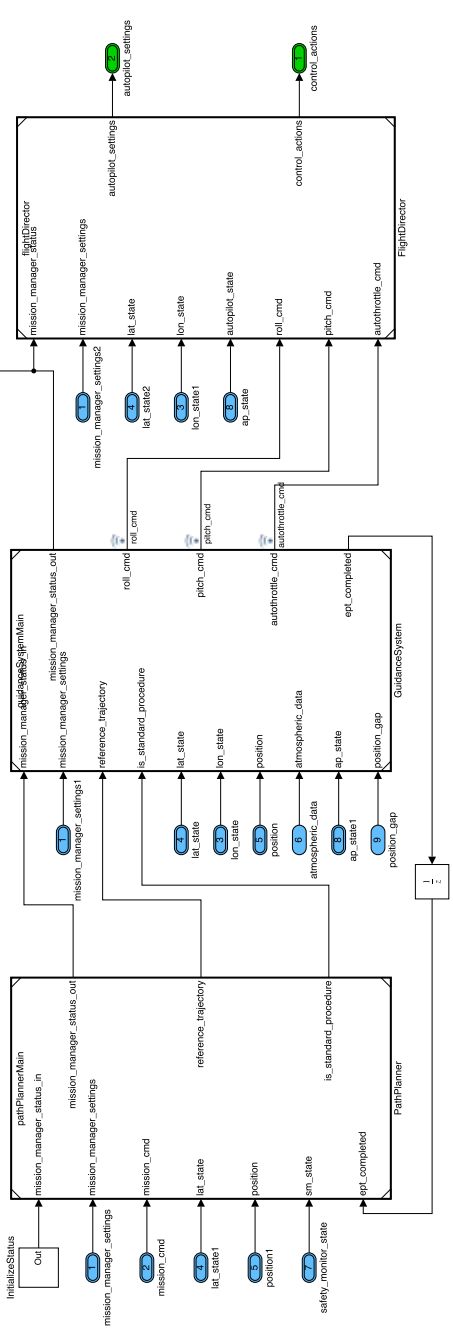


Figure B.7: High-level view of the Mission Manager model in Simulink.

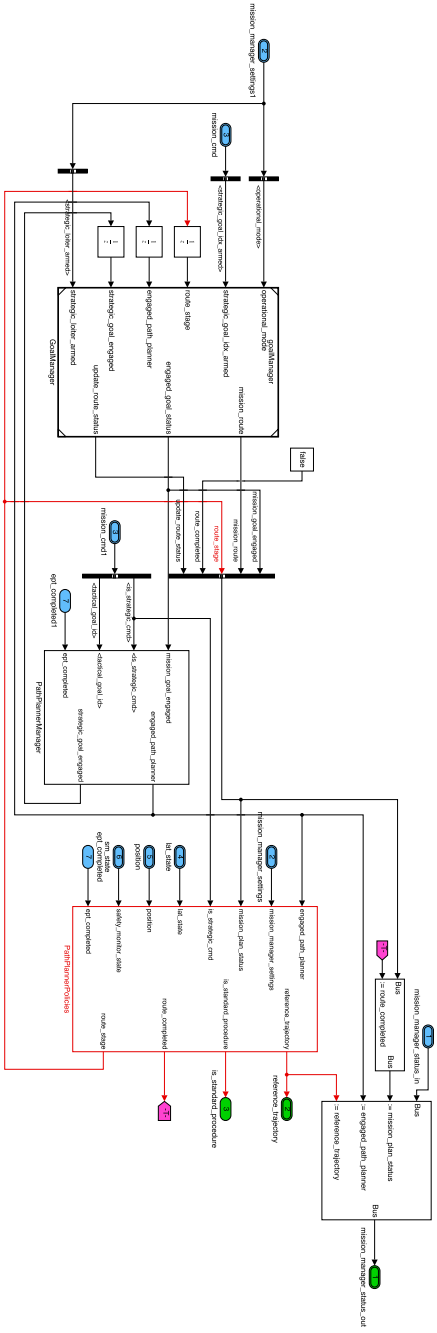


Figure B.8: Path Planner model in Simulink.

B.7.2 Guidance System

The `GuidanceSystem` model is shown in Fig. B.9. It is in charge of generating the reference values for the `FlightDirector` so that the aircraft trajectory is flown in an automatic manner. As it can be observed, this model contains three sub-modules: the `ManeuverPlanner`, the `GuidanceLaw` and the `Flight-EnvelopeProtection` function. Each time the `PathPlanner` provides a new reference trajectory, the `ManeuverPlanner` decomposes it into a sequence of elemental maneuvers in the LNAV plane and in the VNAV plane. Elemental maneuvers in the LNAV plane include straight segments (with constant heading or with constant course) and circular segments (with constant turn radius or with constant bank angle). Elemental maneuver in the VNAV plane include level-off segments and climb or descent segments (with constant speed, with constant vertical speed or with constant vertical flight path angle). Once the list of elemental maneuvers is planned, the `ManevuerPlanner` activates them in a sequential manner (but only one LNAV maneuver and one VNAV maneuver can be active at a time). An elemental maneuver is flown until a specified condition occurs. For example, a climb/descent maneuver is flown until the target altitude is reached. When all maneuvers in the maneuver buffer are completed, the module rises the `ept_completed` flag so that the `PathPlanner` layer reacts.

Then, `GuidanceLaw` module computes the roll command, the pitch command and the auto-throttle command based on the active LNAV/VNAV maneuver and the current aircraft state. In particular, the different commands specify the controlled variable, the target value for this variable and also the appropriate control mode of the `FlightDirector`. For example, when the active maneuver is straight maneuver with constant heading, the controlled variable is the aircraft heading, the target value is computed using a certain *guidance law*, and the control mode is the heading hold mode. As it was indicated in Sec. 4.2, we have implemented several guidance laws for the different elemental maneuvers, like the "carrot-chasing" algorithm in [40]. The control modes of the `FlightDirector` are basically the control modes of a conventional MCP: heading hold (`HDG_HLD`), speed hold (`SPD_HLD`), vertical-speed hold (`VS_HLD`), auto-throttle (`ATR`), etc.

Finally, the `FlightEnvelopeProtection` function is a safety mechanism that prevents that the target values computed by the `GuidanceLaw` module exceeds the flight envelope limits of the aircraft model under consideration. In this case, this module limits the output range of the roll, roll rate, speed and vertical speed based on the flight performance data of the Super Heron.

Figure B.9: Guidance System model in Simulink.

B.7.3 Flight Director

The `FlightDirector` model is shown in Fig. B.10. In general, this model should implement the control loops that provide the input values for the control surfaces of the aircraft (elevators, ailerons, rudder) and for the power plant. In this case, as it was mentioned before, we will delegate the flight control functions to the Flight Control System provided by X-Plane. Therefore, the role of the `FlightDirector` is simply to provide the interface between the `GuidanceSystem` of the SMMS and the autopilot settings of X-Plane.

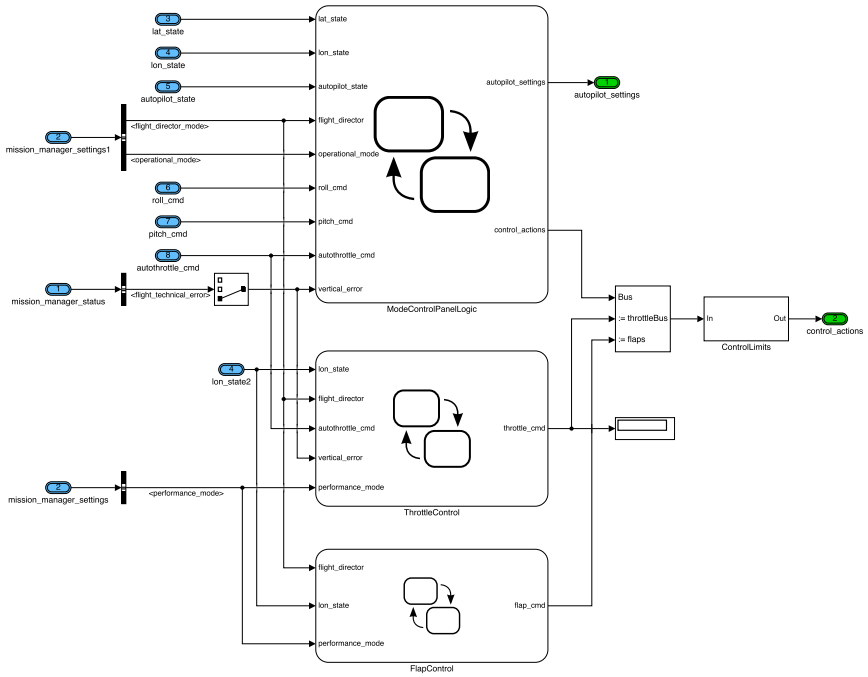


Figure B.10: Flight Director model in Simulink.

B.8 Flight Termination System model

In the prototype application, the activation of the `FlightTerminationSystem` model simply stops the simulation, so further implementation details are omitted for brevity.

Appendix C

Reconfigurable Mission Plan implementation and tools

C.1 Introduction

As a part of the demonstration application developed in this thesis, the Reconfigurable Mission Plan concept described in Chapter 5 has been prototyped in Matlab. The proposed implementation exploits the Matlab support for Object-Oriented Programming (OOP). Using OOP, the different classes that conform the Reconfigurable Mission Plan can be implemented following the UML models in Sec. 5.2 in a straightforward manner. Further implementation details are thus omitted for brevity.

Reconfigurable Mission Plan objects allow to deduce the route of a mission dynamically. Therefore, a number of tools for obtaining and analyzing dynamic routes have been also developed using Matlab. The proposed tools are organized in two main modules: a first one to derive the Mission Graph from a Reconfigurable Mission Plan object; and a second one to search the graph and find dynamic routes that are effective for achieving some given mission goal. In this case, these tools rely on the "Graph and Network Algorithms" Matlab toolkit. The following subsections provide more implementation details.

Listing C.1: Pseudocode for constructing the Mission Graph.

```
1  function G = getMissionGraph(missionObj)
2  G=digraph(); %Initialize digraph object
3  %1.- Perform disjoint union of segments (SU1)
4  for i=1:getNumSegments(missionObj)
5      [s,t,w]=getSegmentNodes(missionObj.segmentObj(i));
6      G=addedge(G,s,t,w);  %Add nodes to graph object
7  end
8  %2.- Create transition edges (SU2)
9  for i=1:getNumSegments(missionObj)
10     for j=1:getNumSegments(missionObj)
11         currentSegmentObj=missionObj.routeSegment(i);
12         nextSegmentObj=missionObj.routeSegment(j);
13         [s,t,w]=getTransitionEdgeNodes(currentSegmentObj, ...
               nextSegmentObj);
14         G=addedge(G,s,t,w); %Add nodes to graph object
15     end
16 end
17 end
```

C.2 Mission Graph construction

Constructing the Mission Graph object consists of performing the union of all the segments in a Reconfigurable Mission Plan object. The pseudocode of Listing C.1 performs this process based on rules **SU1** and **SU2** in Sec. 5.1.5. In short, the pseudocode that checks condition **SU1** (lines 4 to 7) reads the Mission Plan data structure `missionObj`. Then, it gets the nodes (`s`, `t`) and the edge weights (`w`) of all the segments in `missionObj` using the `getSegmentNodes` procedure (which is further described below). Finally, these nodes are added to the Mission Graph structure `G`. Note that the `addedge` function in line 6 is a Matlab function available from the "Graph and Network Algorithms" toolkit.

Then, the pseudocode that checks condition **SU2** (lines 9 to 16) shows a double `for` loop that iterates for every couple of segments in the `missionObj` and verifies if some pair of waypoints meet conditions **SU2.1**, **SU2.2** and **SU2.3** in Sec. 5.1.5 (procedure `getTransitionEdgeNodes`). If so, a transition edge is added to the digraph `G`. Recall that the associated weight of these edges is `w=0`.

One of the key aspects in the Mission Graph construction is computing the edge weights returned by the `getSegmentNodes` procedure. In this work, these weights represent the risk entailed to third parties defined as in Eq. (6.25). Procedure `getEdgeWeight` in Listing C.2 implements this equation based on the

risk models in Chapter 6. Note that this procedure is invoked by `getSegmentNodes` for every `legObj` in the `segmentObj` under analysis. Then, for each leg, `getEdgeWeight` computes the ground risk `Lambda_G` and the air risk `Lambda_A` as described next.

On one hand, the pseudocode that implements the ground risk model (lines 2 to 5) computes the membership function `membershipFcn` for the input leg according to the grid defined by the `populationDensityFile` (the raster image in Fig. 6.3). Implementation of `getMembershipFcn` relies on Matlab procedures `imline` and `createMask`, but further details are omitted for brevity. Then, `getGroundImpactRate` implements the BBN ground risk model in Fig. 6.1 using the "Bayes Net Toolbox" for Matlab[1]. This procedure returns the `groundImpactRate` in terms of occurrences per hour of operation according to the input model evidences defined in the data structure `groundRiskModelEvidence`. Finally, procedure `getGroundRisk` computes the instant risk at every cell of the raster file using the model parameters `groundRiskModelParams` and the `groundImpactRate`. Then, it computes the discrete line integral in Eq. (6.19) for all the cells identified by the `membershipFcn`.

On the other hand, the pseudocode that implements the air risk model (lines 6 to 16) defines the traffic density category `trafficDensityCat` for the input leg based on the `trafficDensityFile`, a XLS file containing the average traffic rate at different waypoints deduced from Eurocontrol's NEST dataset. Then, it computes the MAC rate (`macRate`) from the air impact BBN models in Figures 6.5 or 6.6, depending on whether the `airspaceClass` is controlled or not. Note that these models are also implemented using the "Bayes Net Toolbox" for Matlab. Afterwards, procedure `getAirRisk` implements the line integral in Eq. (6.23) to return the air risk (`Lambda_A`). Finally, the output edge weight (`w`) is computed in line 18.

C.3 Dynamic route configuration tools

Once the Mission Graph object `G` has been created, the next step for flying a Reconfigurable Mission Plan is to find the optimal route to achieve a given goal. In this case, this requires to solve the Graph Theory's shortest path problem. Next, we present a series of tools to find an optimal path in a Mission Graph using the Dijkstra's algorithm in Matlab. Once an optimal path has been

[1]Bayes Net Toolbox written by Kevin Murphy. Available online at https://www.cs.utah.edu/~tch/notes/matlab/bnt/docs/bnt_pre_sf.html (last accessed on March 2019).

Listing C.2: Pseudocode for setting a Mission Graph edge weight.

```
function w = getEdgeWeight(segmentObj,legObj,populationDensityFile, ...
    groundRiskModelParams,groundRiskModelEvidence, ...
    trafficDensityFile,airRiskModelParams, ...
    airRiskModelEvidenceControlled,airRiskModelEvidenceUncontrolled)
%1.- Get ground risk
membershipFcn = getMembershipFcn(legObj,populationDensityFile);
groundImpactRate = getGroundImpactRate(groundRiskModelEvidence)
Lambda_G = getGroundRisk(groundRiskModelParams,groundImpactRate, ...
    populationDensityFile,membershipFcn);
%2.- Get air risk
trafficDensityCat = getTrafficDensity(legObj,trafficDensityFile);
if isa(segmentObj.containmentArea, 'NavigationSpec')
    airspaceClass = 'Controlled';
    airRiskModelEvidence = airRiskModelEvidenceControlled;
else
    airspaceClass = 'Uncontrolled';
    airRiskModelEvidence = airRiskModelEvidenceUnontrolled;
end
macRate = getMacRate(trafficDensityCat,airspaceClass, ...
    airRiskModelEvidence);
Lambda_A = getAirRisk(airRiskModelParams,macRate,airspaceClass,legObj);
%3.- Compute edge weight
w = Lambda_G+Lambda_A;
end
```

found, it will be necessary to convert this path to a route specification object. This process will also be described afterwards.

C.3.1 Route search tools

As it was introduced in Sec. 5.3, the problem of finding a route in the graph consists of two major steps:

1. Locating all the nodes of the Mission Graph that are associated with the target goal, and
2. Finding a path in the Mission Graph that connects the source node with all the required nodes.

Based on this idea, we have implemented a Matlab procedure for finding dynamic routes that is structured into the following three levels:

Listing C.3: Pseudocode for finding the path to achieve one given goal type.

```
function [POut,cOut] = getPathToGoalsByType(G,missionObj, ...
    srcNode,trgGoalType)
goalObjArr=getGoalsByType(missionObj,trgGoalType);
for i=1:length(goalObjArr)
    [P,c]=getPathToGoal(G,srcNode,goalObjArr(i) );
    POut{i,:}=P; %Add path to output struct
    cOut=[cOut c]; %Add cost to output struct
end
[POut,cOut]=sort(POut,cOut); %Sort paths by cost
end
```

1. *Find the path for achieving one given goal type.* The highest level of this problem requires all the goals in the Mission Plan matching the target goal type be found (for example, all the "loiter" goals, or the "flight termination" goals, etc.), and then the path for achieving each of these goals is computed. The algorithm `getPathToGoalsByType` of Listing C.3 performs this task for a given Mission Graph `G` and a Mission Plan data structure `missionObj`. The remaining input arguments are the source node `srcNode` and the target goal type `trgGoalType`. The algorithm returns the paths for achieving these goals (`POut`), as well as the cost associated to these paths (`cOut`). These paths are computed by invoking the `getPathToGoal` procedure multiple times, as described next.

2. *Find the path for achieving one given mission goal.* This problem requires a path traversing all the associated locations of a given goal be found. The algorithm `getPathToGoal` of Listing C.4 performs this task for a given input goal `goalObj`. When the goal is a single stage goal, the problem is trivial and is solved as computing the path from the source node `srcNode` to a node associated with the target location. This path is computed using the `getPathToWaypoint` procedure, which will be introduced below. In multiple stage goals, we assume that the intermediate stages are to be flown in some given order, so the problem can be solved by invoking the `getPathToWaypoint` procedure between every two consecutive stage waypoints and appending this sub-path to the output path structure `POut`. The cumulative cost of the path (`cOut`) is then computed as the sum of the costs of each sub-path.

3. *Find the path towards one given waypoint.* The most elementary problem consists of finding *all* the nodes of the Mission Graph associated with a

Listing C.4: Pseudocode for finding the path to achieve one given goal.

```
function [POut,cOut] = getPathToGoal(G,srcNode,goalObj)
s0=srcNode; %Initialize source node
n=getNumStages(goalObj);
for i=1:length(n)
    trgWaypoint=goalObj.stage(i).enabledVariant;
    [P,c]=getPathToWaypoint(G,s0,trgWaypoint);
    if isempty(P)
        return;
    end
    POut=[POut P(1)]; %Append path to output struct
    cOut=cOut+c(1); %Update path cost
    s0=POut(end); %Update source node
end
end
```

Listing C.5: Pseudocode for finding the path towards one given waypoint.

```
function [POut,cOut] = getPathToWaypoint(G,srcNode,waypointObj)
trgNodeArr=getNodesInWaypoint(G,waypointObj);
for i=1:length(trgNodeArr)
    [P,c]=shortestpath(G,srcNode,trgNodeArr(i),'Method','positive')
    POut{i,:}=P; %Add path to output struct
    cOut=[cOut c]; %Add cost to output struct
end
[POut,cOut]=sort(POut,cOut); %Sort paths by cost
end
```

given waypoint, and then computing the path from a given source node to these target nodes. The algorithm `getPathToWaypoint` of Listing C.5 performs this task for a given input waypoint `waypointObj`. In this algorithm, the path between nodes is computed using the Matlab procedure `shortestpath`, setting the `'Method'` attribute to `'positive'`. This method implements Dijkstra's algorithm in Matlab. The `getPathToWaypoint` procedure returns the path from the source node to all these target nodes (`POut`) and sorts these paths according to their cost.

C.3.2 Specification of the Route object

The last step of the dynamic route configuration process is converting one path described as a sequence of nodes of the Mission Graph to a `route` object like in Fig. 5.9. This step requires all the segments traversed by the path be identified, as well as the points in which the path traverses from one segment to another segment (i.e. the entry points and the exit points of each segment, see Sec. 5.1.5). The advantage of using the disjoint union in the graph creation is that this task is straightforward, so details are here omitted for brevity.

Appendix D

Automatic deployment to XtratuM

D.1 Introduction

The deployment process deals with the process of porting designs from the design platform to the run-time environment. In this work, the design platform relies on Matlab/Simulink and the run-time environment relies on the XtratuM hipervisor.

D.1.1 XtratuM run-time environment

We propose to execute the prototype application of Appendix B over a target platform running XtratuM: a hypervisor for real-time embedded systems developed in our research group [113]. XtratuM is based on the partitioning concept standardized by ARINC-653 [6]. In a partitioned system, each partition integrates an application and an associated guest OS. The advantage is that different applications running on the same hardware are isolated from the temporal and spacial point of views. In this case, temporal isolation is achieved by implementing a fixed cyclic scheduler, what is consistent with the

ARINC-653 scheduling policy for partitions; while spatial isolation relies on fixed memory allocation.

XtratuM supports several real-time OSs. One of them is LithOS, an ARINC-653 compliant execution environment also developed in our research group [112]. In short, LithOS provides the services defined by the ARINC-653 specification, including partition management, process management, time management, inter-partition and intra-partition communication, and health monitoring. Regarding process management, LithOS implements the priority based scheduling policy for concurrent applications.

XtratuM hypervisor emulator

In order to accelerate the design of aerospace applications to be executed on top of XtratuM, the research group has developed an emulation environment of XtratuM, called Separation Kernel Emulator (SKE) [176]. The SKE allows to debug and validate the functional behavior of a partitioned application when a board running XtratuM natively is not yet available.

The XtratuM SKE runs as a Linux process that controls the execution of a set of processes that represent the partitioned system. In other words, in the emulation environment, LithOS partitions are executed as Linux processes, where LithOS is included as an internal OS of each of these processes.

The resulting emulation is functionally equivalent in all aspects to the native XtratuM except for time management: the SKE process in Linux implements its own clock which provides emulated time, not real-time; so real-time behavior cannot be validated. By contrast, the advantage is that a LithOS partition executed as a Linux process can benefit from services provided by Linux, such as sockets or other libraries that can be integrated into the test bench.

D.2 Automatic deployment tools

One of the advantages of the Matlab/Simulink design platform is the support for the Model-Based Design (MBD) methodology. MBD technologies usually provide automatic code generation tools to generate executable code from a symbolic or high level model. This is the case of the Simulink Coder, which is able to generate C/C++ code from Simulink models, Stateflow charts, and Matlab functions. Simulink Coder supports run-time execution targets like POSIX or ARINC-653 compliant systems (like XtratuM). As a result, it is

possible to generate the executable code of XtratuM applications from symbolic models in Simulink.

A series of tools have been designed to automatize the process of porting designs from Simulink models to XtratuM based on the following three steps:

1. Mapping Simulink abstractions to XtratuM abstractions,
2. Generating the application code, and
3. Configuring the XtratuM project directory.

Each of these steps is briefly described next; while further details can be found on previous works of the authors and members of the research group [67, 68, 165, 166].

D.2.1 Identification of the application partitioning scheme

The first stage of the porting process is related to the partitioning of the application. The goal is *mapping Simulink abstractions to XtratuM abstractions*. In other words, it is necessary to map Simulink blocks to XtratuM partitions, communication ports and channels, etc.

The proposed approach relies on the Simulink *referenced model* concept. Referenced models allow to include one model into another by referencing it. Based on this idea, Simulink blocks that you want to allocate to a same partition must be grouped into a same referenced model on the top-level diagram of the Simulink model. For example, the top-level diagram of the SMMS model in Fig. B.2 presents three referenced models, named `SystemInterface`, `Partition0` and `Partition1`. Therefore, the proposed SMMS prototype will be composed of three system partitions in XtratuM. Simulink blocks referenced by each of these models will be allocated to a different partition after the porting process.

In addition, referenced models have an interface that consists on a series of inputs, outputs and parameter arguments. Based on the same idea, input and output ports of referenced models on the top-level diagram will be mapped to XtratuM sampling ports. Simulink signals connecting these ports will represent communication channels. For example, the XtratuM partition allocating the `Partition0` model in the previous figure will have 11 input sampling ports and 9 output sampling ports; 6 of these output ports connect `Partition0` with `Partition1` through communication channels.

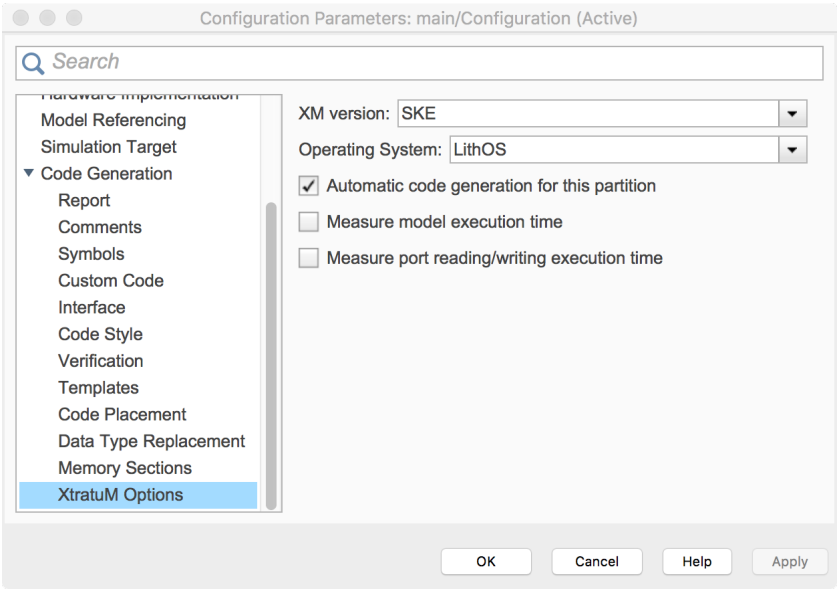


Figure D.1: XtratuM configuration window.

All these tasks are accomplished through a specific tool which *identifies* partitions, ports, and channels from the Simulink top-level model. Then, the last part of the partition configuration process is *setting the OS for each system partition*. In this case, this task is to be performed manually by the final user through a Simulink menu specially developed for this purpose, see Fig. D.1. Several guest real-time OSs are available in XtratuM, including LithOS, Partikle (POSIX type) and Linux. The partition can also be configured to run without OS. This is called a *bare* partition. If the selected OS supports multitasking and it has been enabled in the Simulink model, then the user is asked to enable it in the XtratuM application too.

D.2.2 Code generation

The next step in the porting process is code generation. This is mainly done through the automatic code generator tool called Simulink Coder that generates C/C++ code from Matlab, Simulink, and Stateflow blocks. Even though, there are some important aspects that need to be addressed:

- *Customized generated code.* Some Simulink generated code needs to be tuned according to the API of the target execution environment. In the

case of the SKE, this refers to the calls to the ARINC-653 services, including the sampling ports creation, the reading and writing mechanisms, the generation of some support files with constants definitions, or the way POSIX threads and semaphores are used, among others. Simulink Coder allows to customize the way the code is generated by means of Target Language Compiler (TLC) files. Therefore, we have created some TLC files and modified some of the ones provided by Simulink in order to make the generated code runnable on XtratuM.

- *Concurrency model.* The multitasking model must match the multitasking model of the target OS, so it has to be properly configured. Multitasking in LithOS partitions is implemented through the ARINC-653 process concept, whilst Linux and Partikle systems use POSIX threads. In bare applications (those running on XtratuM directly), there is no support for concurrent execution so no multitasking calls to the OS must be generated.

- *External libraries.* Simulink Coder assumes that the standard C libraries are available, so it generates calls to their functions or defined constants, waiting for the links to be resolved in compilation time. This issue is not relevant when the target platform is the SKE since it can have access to all underlying Linux services. However, some of these libraries are not included in bare version of XtratuM, so the user may have to provide the missing resources when required.

It is also worth noting that the automatic coding process performed by Simulink Coder is not certified, but in any case it helps the coding task very much, as long as the produced code is understandable, well structured, and does not make use of non approved language constructions for certification.

D.2.3 Configuration of the XtratuM project directory

The last part of the porting process is configuring the XtratuM project directory. XtratuM requires a number of makefiles and configuration files which are often difficult to generate. To overcome this, we have developed an automatic tool that performs this task using the information contained in the Simulink models. In particular, this tool generates the following project files:

- *Makefiles* describing the rules to compile the source code of each system partition. Although Simulink generates its own makefile, it is barely use-

ful because compiling applications for XtratuM requires some particular rules [113].

- *Hypervisor configuration files* defining system resources, and how they are allocated to each partition. This includes aspects like the number of partitions and their communication ports, or cyclic plan information. In the bare metal hypervisor version, it also includes information regarding memory allocation that must be supplied by the user at a later stage.
- *LithOS configuration files* specifying the maximum number of the different resources used by the partition (processes, events, semaphores, etc.). Each LithOS partition has its own configuration file.

Finally, the automatic tool also generates the structure of the project directory and merges there the different generated files required to compile the application, including source files, libraries, makefiles, and configuration files.

D.3 Safe Mission Management System deployment issues

Due to the inherent complexity of the SMMS prototype of Appendix B, a number of problems when exploiting the automatic deployment tools have made the initial goal of executing the prototype on top of XtratuM unfeasible. One of the most remarkable problems is that some of the Matlab toolboxes, Simulink blocks, or data types used in the implementation of either the Reconfigurable Mission Plan concept or the Safe Mission Manager architecture do not fully support code generation. For example, the Simulink Level-2 S-function block which allows to execute Matlab code from Simulink does not support code generation unless specific TLC files are provided. Due to practical reasons and time constraints, the creation of specific TLC files for each required model block and the complete deployment of the prototype model to XtratuM will be reported as future work.

Appendix E

Bayesian Belief Network impact model parameters

E.1 Ground impact model CPTs

- The CPT of node "Navigation error" in Fig. 6.1 is shown in Table E.1.
- The CPT of node "C2 link loss" in Fig. 6.1 is shown in Table E.2.
- The CPT of node "Autopilot malfunction" in Fig. 6.1 is shown in Table E.3.
- The CPT of node "Pilot ineffective" in Fig. 6.1 is shown in Table E.4.
- The CPT of node "Inappropriate guidance" in Fig. 6.1 is shown in Table E.5.
- The CPT of node "Boundary violation" in Fig. 6.1 is shown in Table E.6.
- The CPT of node "Loss of control" in Fig. 6.1 is shown in Table E.7.
- The CPT of node "Ground impact" in Fig. 6.1 is shown in Table E.8.

E.2 Mid-air collision model in controlled airspace CPTs

- The CPT of node "Navigation error" in Fig. 6.5 is shown in Table E.1.
- The CPT of node "C2 link loss" in Fig. 6.5 is shown in Table E.2.
- The CPT of node "Autopilot malfunction" in Fig. 6.5 is shown in Table E.3.
- The CPT of node "Pilot ineffective" in Fig. 6.5 is shown in Table E.4.
- The CPT of node "Inappropriate guidance" in Fig. 6.5 is shown in Table E.9.
- The CPT of node "ATC ineffective" in Fig. 6.5 is shown in Table E.10.
- The CPT of node "Tactical separation error" in Fig. 6.5 is shown in Table E.11.
- The CPT of node "Strategic separation error" in Fig. 6.5 is shown in Table E.12.
- The CPT of node "Separation error" in Fig. 6.5 is shown in Table E.13.
- The CPT of node "TCAS ineffective" in Fig. 6.5 is shown in Table E.14.
- The CPT of node "NMAC" in Fig. 6.5 is shown in Table E.15.
- The CPT of node "SAA ineffective" in Fig. 6.5 is shown in Table E.16.
- The CPT of node "DAA error" in Fig. 6.5 is shown in Table E.17.
- The CPT of node "Collision avoidance error" in Fig. 6.5 is shown in Table E.18.
- The CPT of node "MAC" in Fig. 6.5 is shown in Table E.19.

E.3 Mid-air collision model in uncontrolled airspace CPTs

- The CPT of node "Navigation error" in Fig. 6.6 is shown in Table E.1.
- The CPT of node "C2 link loss" in Fig. 6.6 is shown in Table E.2.
- The CPT of node "Autopilot malfunction" in Fig. 6.6 is shown in Table E.3.

- The CPT of node "Pilot ineffective" in Fig. 6.6 is shown in Table E.4.
- The CPT of node "Inappropriate guidance" in Fig. 6.6 is shown in Table E.9.
- The CPT of node "Boundary violation" in Fig. 6.6 is shown in Table E.20.
- The CPT of node "Remain well clear error" in Fig. 6.6 is shown in Table E.21.
- The CPT of node "Separation error" in Fig. 6.6 is shown in Table E.22.
- The CPT of node "NMAC" in Fig. 6.6 is shown in Table E.23.
- The CPT of node "SAA ineffective" in Fig. 6.6 is shown in Table E.24.
- The CPT of node "DAA error" in Fig. 6.6 is shown in Table E.17.
- The CPT of node "Collision avoidance error" in Fig. 6.6 is shown in Table E.18.
- The CPT of node "MAC" in Fig. 6.6 is shown in Table E.19.

Table E.1: CPT of node "Navigation error" in Figures 6.1, 6.5 and 6.6.

Navigation error	
F	T
9,9005e−1	9,9502e−3

Table E.2: CPT of node "C2 link loss" in Figures 6.1, 6.5 and 6.6.

C2 link loss	
F	T
3,6788e−1	6,3212e−1

Table E.3: CPT of node "Autopilot malfunction" in Figures 6.1, 6.5 and 6.6.

Navigation error	Autopilot malfunction	
	F	T
F	9,9999e−1	1,0000e−5
T	9,9899e−1	1,0100e−3

Table E.4: CPT of node "Pilot ineffective" in Figures 6.1, 6.5 and 6.6.

C2 link loss	Navigation error	Pilot ineffective	
		F	T
F	F	9,9900e−1	9,9950e−4
F	T	9,9800e−1	1,9985e−3
T	F	0	1
T	T	0	1

Table E.5: CPT of node "Inappropriate guidance" in Fig. 6.1.

Autopilot malfunc.	Pilot ineffective	Inappropriate guidance	
		F	T
F	F	1	0
F	T	0	1
T	F	0	1
T	T	0	1

Table E.6: CPT of node "Boundary violation" in Fig. 6.1.

Navigation error	Inappro. guidance	Boundary violation	
		F	T
F	F	1	0
F	T	0,25	0,75
T	F	0,25	0,75
T	T	0,05	0,95

Table E.7: CPT of node "Loss of control" in Fig. 6.1.

Inappro. guidance	Loss of control	
	F	T
F	1	0
T	9,9990e−1	9,9995e−5

Table E.8: CPT of node "Ground impact" in Fig. 6.1.

Loss of control	Boundary violation	Ground impact	
		F	T
F	F	1	0
F	T	9,9900e−1	9,9950e−4
T	F	0,1	0,9
T	T	0,1	0,9

Table E.9: CPT of node "Inappropriate guidance" in Figures 6.5 and 6.6.

Autopilot malfunc.	Pilot ineffective	Inappropriate guidance	
		F	T
F	F	1	0
F	T	0,8	0,2
T	F	0,99	0,01
T	T	0	1

Table E.10: CPT of node "ATC ineffective" in Fig. 6.5.

Traffic density	ATC ineffective	
	F	T
Low	9,9940e−1	5,9995e−4
Medium	9,9890e−1	1,0999e−3
High	9,9840e−1	1,5998e−3

Table E.11: CPT of node "Tactical separation error" in Fig. 6.5.

Traffic density	ATC ineffective	Inappro. guidance	Tactical separation error	
			F	T
Low	F	F	1	0
Low	F	T	9,9950e−1	5,0000e−4
Low	T	F	0	1
Low	T	T	0	1
Medium	F	F	1	0
Medium	F	T	9,9900e−1	1,0000e−3
Medium	T	F	0	1
Medium	T	T	0	1
High	F	F	1	0
High	F	T	9,9850e−1	1,5000e−3
High	T	F	0	1
High	T	T	0	1

Table E.12: CPT of node "Strategic separation error" in Fig. 6.5.

Strategic sep. error	
F	T
9,9005e−1	9,9502e−3

Table E.13: CPT of node "Separation error" in Fig. 6.5.

Strategic sep. error	Tactical sep. error	Separation error	
		F	T
F	F	1	0
F	T	1	0
T	F	1	0
T	T	0	1

Table E.14: CPT of node "TCAS ineffective" in Fig. 6.5.

Pilot ineffective	TCAS ineffective	
	F	T
F	9,9999e−1	1,0000e−5
T	0	1

Table E.15: CPT of node "NMAC" in Fig. 6.5.

Separation error	TCAS ineffective	NMAC	
		F	T
F	F	1	0
F	T	1	0
T	F	1	0
T	T	0	1

Table E.16: CPT of node "SAA ineffective" in Fig. 6.5.

Pilot ineffective	SAA error	
	F	T
F	9,9900e−1	9,9950e−4
T	0	1

Table E.17: CPT of node "DAA error" in Figures 6.5 and 6.6.

DAA error	
F	T
9,9999e−1	1,0000e−5

Table E.18: CPT of node "Collision avoidance error" in Figures 6.5 and 6.6.

DAA available	SAA ineffective	DAA error	Collision avoidance error	
			F	T
F	F	F	1	0
F	F	T	1	0
F	T	F	0	1
F	T	T	0	1
T	F	F	1	0
T	F	T	0,95	0,05
T	T	F	1	0
T	T	T	0	1

Table E.19: CPT of node "MAC" in Figures 6.5 and 6.6.

NMAC	Collision av. error	MAC	
		F	T
F	F	1	0
F	T	1	0
T	F	1	0
T	T	0	1

Table E.20: CPT of node "Boundary violation" in Fig. 6.6.

Navigation error	Inappro. guidance	Boundary violation	
		F	T
F	F	1	0
F	T	0,99	0,01
T	F	0,95	0,05
T	T	0,99	0,01

Table E.21: CPT of node "Remain well clear error" in Fig. 6.6.

Traffic density	Inappro. guidance	Remain well clear error	
		F	T
Low	F	1	0
Low	T	9,9950e−1	5,0000e−4
Medium	F	1	0
Medium	T	9,9900e−1	1,0000e−3
High	F	1	0
High	T	9,9850e−1	1,5000e−3

Table E.22: CPT of node "Separation error" in Fig. 6.6.

Traffic density	Boundary violation	RWC error	Separation error	
			F	T
Low	F	F	1	0
Low	F	T	0	1
Low	T	F	9,9950e−1	5,0000e−4
Low	T	T	0	1
Medium	F	F	1	0
Medium	F	T	0	1
Medium	T	F	9,9900e−1	1,0000e−3
Medium	T	T	0	1
High	F	F	1	0
High	F	T	0	1
High	T	F	9,9850e−1	1,5000e−3
High	T	T	0	1

Table E.23: CPT of node "NMAC" in Fig. 6.6.

Separation error	NMAC	
	F	T
F	1	0
T	0	1

Table E.24: CPT of node "SAA ineffective" in Fig. 6.6.

Pilot ineffective	SAA error	
	F	T
F	9,9005e−1	9,9502e−3
T	0	1