Universitat Autònoma de Barcelona

INTEGRATION OF NOISE CHARACTERISATION AND ITS MITIGATION IN THE U-SPACE

Thesis submitted for the degree on Aeronautical Management

By

Muhammad Shargeel Shafique Sultana

Supervisor:

Jose Luis Muñoz Gamarra

Escuela de Ingeniería

Sabadell, June 2026

The undersigned, **Jose Luis Muñoz Gamarra**, supervisor of the Final Degree Thesis, professor at the School of Engineering of UAB,

CERTIFIES:

That the work corresponding to this report has been carried out under their supervision by

Muhammad Shargeel Shafique Sultana

And for the record, signs this document in Sabadell, June of 2026

MUNOZ GAMARRA JOSE LUIS - 75142324J

Firmado digitalmente por MUNOZ GAMARRA JOSE LUIS - 75142324J
Fecha: 2026.06.25 14:11:06 +02'00'

Signed: **Jose Luis Muñoz Gamarra**

Table of Contents

1. Introduction 9
2. U-space characterisation 11
   2.1. Definition to U-space 11
   2.2. Applicable regulatory framework 12
      2.2.1. U-Space regulations 12
      2.2.2. UAS regulations 13
   2.3. Typology of operations and UAS 13
   2.4. U-Space services 16
   2.5. U-Space implementation 18
   2.6. U-Space stakeholders 19
3. Noise characterisation & modelling 21
   3.1. Characterisation of sound source 21
   3.2. Physic-mathematical Model of Sound Propagation in Urban Environments 23
      3.2.1. Geometry of the direct acoustic path (source-receiver) 23
      3.2.2. Attenuation due to geometric divergence 24
      3.2.3. Attenuation due to atmospheric absorption 24
      3.2.4. Direct Sound Pressure Level 24
      3.2.5. Acoustic Reflections and the “Urban Canyon Effect” 25
      3.2.6. Final Sound Pressure Level 26
   3.3. Social impact review 27
   3.4. Noise simulation study 29
      3.4.1. Noise simulation results 30
4. Simulation framework 32
   4.1. Traffic simulation framework 36
5. Results 38
6. Conclusion 40
7. References 43
8. Appendix 46

List of Figures

FIGURE 1 – U-SPACE (ENAIRE, 2023) 11
FIGURE 2 – U-SPACE LEVEL (CORUS, 2019B) 19
FIGURE 3 – DJI PHANTOM 4 PRO PLUS (LUIS FERNANDO ANTOLÍN, 2017) 21
FIGURE 4 – EXAMPLE OF A NOISE SPECTROGRAM: TONAL, BPF, BROADBAND NOISE (MILJKOVIĆ, 2018) 22
FIGURE 5 – DISTANCE SOUND SOURCE-RECEIVER 23
FIGURE 6 – SOUND PRESSURE LEVEL AT DIFFERENT HEIGHT WITH THE SAME ENVIRONMENT 29
FIGURE 7 - LOGICAL STRUCTURE OF THE CODE 32
FIGURE 8 – TRAFFIC MAP 14:00H 36
FIGURE 9 – TRAFFIC MAP 6:00H 36
FIGURE 10 – INTEGRATION OF NOISE RESTRICTIONS, DCB, AND STRATEGIC DECONFLICTING WITHIN THE FLIGHT SERVICE AUTHORISATION FRAMEWORK 42

List of Tables

TABLE 1 – DRON OPERATION CATEGORIES (CORUS, 2019A) 14
TABLE 2 – GENERAL CHARACTERISATION OF DRONE CLASSES (CORUS, 2019A) 14
TABLE 3 – SUBCATEGORIES A1-A2-A3 (CORUS, 2019A) 15
TABLE 4 – MANNERS OF OPERATING A DRONE (CORUS, 2019B) 15
TABLE 5 – U-SPACE VOLUMES (CORUS, 2019A) 16
TABLE 6 – U-SPACE SERVICES (CORUS, 2019A) 18
TABLE 7 – U-SPACE PHASES (CORUS, 2019A) 18
TABLE 8 – U-SPACE STAKEHOLDERS (CORUS, 2019A) (EUROPEAN COMMISSION, 2021A) (GOBIERNO DE ESPAÑA - MINISTERIO DE TRANSPORTES Y MOVILIDAD SOSTENIBLE, 2024) 20
TABLE 9 - NOISE PERFORMANCE INDICATORS (GANIĆ ET AL., 2024) 28
TABLE 10 – NOISE SIMULATION STUDY 30
TABLE 11 – TRAFFIC FACTOR/COEFFICIENT 37
TABLE 12 – FLIGHT PLAN #516 38
TABLE 13 – FLIGHT PLAN #497 39

List of Equations

EQUATION 1 – DISTANCE SOUND SOURCE-RECEIVER 23
EQUATION 2 – ATTENUATION DUE TO GEOMETRIC DIVERGENCE 24
EQUATION 3 – ATMOSPHERIC ABSORPTION ATTENUATION 24
EQUATION 4 - DIRECT SPL 24
EQUATION 5 – SPL CONVERSION TO ENERGY 25
EQUATION 6 – TOTAL ENERGY 25
EQUATION 7 - FINAL SPL 26

List of Appendices

APPENDIX 1 – MODULE 1: GET_MAP_BUILDINGS 46
APPENDIX 2 – MODULE 2: ANALYSE_URBAN_ENVIRONMENT 47
APPENDIX 3 – MODULE 3: TRAFFIC_GENERATOR 48
APPENDIX 4 – MODULE 4: CALCULATE_URBAN_NOISE 49
APPENDIX 5 – MODULE 5: NOISE_ANALYZER 50
APPENDIX 6 – MODULE 6: SHOW_LAYOUT 51
APPENDIX 7 – MODULE 7: EXPORT_DATA (1/3) 52
APPENDIX 8 – MODULE 7: EXPORT_DATA (2/3) 53
APPENDIX 9 – MODULE 7: EXPORT_DATA (3/3) 54
APPENDIX 10 – MAIN EXECUTION 55

List of acronyms

UAS	Unmanned Aircraft Systems
ANS	Air Navigation Services
ANSP	Air Navigation Services Provider
ATM	Air Traffic Management
ATSP	Air Traffic Service Providers
BLOS	Beyond Visual Line of Sight
BPF	Blade Passing Frequency
CIS	Common Information Service
CISP	Common Information Service Provider
DAA	Detect and Avoid
DCB	Demand-Capacity Balancing
EASA	European Aviation Safety Agency
EVLOS	Extended Visual Line of Sight
MTOM	Maximum Take-Off Mass
SDSP	Supplementary Data Service Provider
SERA	Standardized European Rules of the Air
SPL	Sound Pressure Level
SSR	Secondary Surveillance Radar
UAM	Urban Air Mobility
UAV	Unmanned Aerial Vehicle
USSP	U-Space Service Providers
UTM	UAS Traffic Management
VLL	Very Low-Level
VLOS	Visual Line of Sight
WHO	World Health Organisation

Abstract

The large-scale deployment of Unmanned Aircraft Systems (UAS) in Urban Air Mobility (UAM) environments raises significant operational considerations relating to their noise emissions in cities. This research is based on the premise that noise impact is a key variable that can limit the capacity of U-space airspace, shaping route planning beyond the conventional boundaries of conflict avoidance and physical safety.

To test this, a simulation framework was developed in Python that integrates cartographic data from OpenStreetMap and a physical-mathematical propagation model to calculate the noise reaching ground level that affects the local population, as well as the noise indicators from the European MUSE project, within a dynamic 24-hour cycle. When this framework was applied to the Eixample district of Barcelona, the results show that continuous operations at low altitudes are not feasible, as the accumulated noise consistently exceeds the legal limit of 65 dB imposed by local regulations, making it necessary to move traffic to higher altitudes or to restrict night-time traffic flows.

As a methodological solution, a sequential architecture for U-space services is proposed, comprising a preliminary acoustic optimisation layer (responsible for defining the dynamic environmental capacity), the Demand-Capacity Balancing (DCB) service at network level, and the strategic deconfliction module at individual flight path level. This approach demonstrates that environmental and legal sustainability can be effectively coordinated with air traffic management to ensure the viability of unmanned aviation.

Resumen

El desarrollo a gran escala de los Sistemas de Aeronaves no Tripuladas (UAS) en entornos de Movilidad Aérea Urbana (UAM) conlleva importantes consideraciones operativas asociadas a su emisión sonora en las ciudades. Esta investigación parte de la premisa de que el impacto acústico constituye una variable fundamental que puede delimitar la capacidad del espacio aéreo U-space, modelando la planificación de rutas más allá de los márgenes convencionales de desconflictividad y seguridad física.

Para comprobarlo, se desarrolló un marco de simulación en Python que integra datos cartográficos tridimensionales de OpenStreetMap y un modelo físico-matemático de propagación para calcular los indicadores de ruido del proyecto europeo MUSE en un ciclo dinámico de 24 horas. Al aplicar este entorno en el distrito de l'Eixample de Barcelona, los resultados demuestran que las operaciones continuas a bajas altitudes (7,5 m y 10 m) son inviables, ya que la acumulación logarítmica del ruido excede sistemáticamente el límite legal de 65 dB, obligando a desplazar el tráfico a cotas superiores (12,5 m y 15 m) o a restringir los flujos nocturnos.

Como solución metodológica, se propone una arquitectura secuencial de servicios de U-space integrada por una capa preliminar de optimización acústica (encargada de acotar la capacidad ambiental dinámica), el servicio de Equilibrio de Demanda y Capacidad (DCB) a nivel de red, y el bloque de resolución de conflictos estratégicos a nivel de trayectoria individual. Este enfoque demuestra que la sostenibilidad ambiental y jurídica puede coordinarse eficazmente con la gestión del tráfico aéreo para garantizar la viabilidad de la aviación no tripulada.

Resum

El desplegament a gran escala dels sistemes aeris no tripulats (UAS) en entorns de mobilitat aèria urbana (UAM) planteja consideracions operatives significatives relacionades amb les seves emissions de soroll a les ciutats. Aquesta recerca es basa en la premissa que l'impacte sonor és una variable clau que pot limitar la capacitat de l'espai aeri U-space, configurant la planificació de rutes més enllà dels límits convencionals d'evitació de conflictes i de seguretat física.

Per provar-ho, es va desenvolupar un marc de simulació basat en Python que integra dades cartogràfiques tridimensionals d'OpenStreetMap i un model de propagació físic-matemàtic per calcular els indicadors de soroll del projecte europeu MUSE en un cicle dinàmic de 24 hores. Quan s'aplica aquest marc al districte de l'Eixample de Barcelona, els resultats mostren que les operacions contínues a altituds baixes són inviables, ja que l'acumulació logarítmica del soroll supera sistemàticament el límit legal de 65 dB, la qual cosa requereix la reubicació del trànsit a altituds més altes o la restricció dels fluxos de trànsit nocturns.

Com a solució metodològica, es proposa una arquitectura seqüencial per als serveis d'espai aeri no tripulat (U-space), que comprèn una capa preliminar d'optimització acústica (responsable de definir la capacitat ambiental dinàmica), el servei d'equilibri de demanda i capacitat (DCB) a nivell de xarxa i el mòdul de desconflicció estratègica a nivell de ruta de vol individual. Aquest enfocament demostra que la sostenibilitat mediambiental i legal es pot coordinar eficaçment amb la gestió del trànsit aeri per garantir la viabilitat de l'aviació no tripulada.

Keywords

UAS
U-space
Noise modelling & mitigation
Sound Pressure Level – SPL
Demand and Capacity Balance – DCB
Strategic Deconflicting
Drone traffic simulation

1. Introduction

The rapid evolution of unmanned aircraft systems (UAS) has profoundly transformed the landscape of contemporary aviation, driving a paradigm shift in the methodological and conceptual approach underpinning the operational management of very low-level (VLL) airspace. In response to the pressing need to accommodate this demand safely and in an orderly manner in Europe, the U-space regulatory framework has emerged. This is not conceived as a segregated volume of airspace reserved exclusively for drones, but rather as an ecosystem comprising digital services and highly automated procedures. Thanks to digitalisation and technological interconnection with traditional air traffic management (ATM) systems, this European framework aims to support large-scale commercial and civil operations, whilst safeguarding operational safety standards equivalent to those of conventional manned aviation. (Sesar, 2023) (CORUS, 2019a) (EASA, 2026)

However, unlike traditional aircraft, the acoustic footprint of drones is characterised by specific psychophysical properties, such as the presence of fixed pure tones, high frequencies and rapid dynamic fluctuations, which have a negative impact on human perception and require specific characterisation models to assess their actual influence on urban soundscapes (Study on the Societal Acceptance of Urban Air Mobility in Europe, 2021). Consequently, the technical and commercial viability of UAS flight paths cannot be determined solely on the basis of traditional criteria of efficiency and traffic optimisation; it is absolutely essential to incorporate socio-environmental impact and noise pollution as critical variables in operational design. (Torija et al., 2020) (Christian & Cabell, 2017)

Research Hypothesis & Objectives

In this context, this Bachelor's Thesis is based on the central hypothesis that the operational capacity and large-scale deployment of U-space airspace in metropolitan areas are not determined solely by the physical separation margins required to prevent collisions. On the contrary, it is posited that noise pollution acts as a structural constraint that limits route capacity and influences the management of traffic flows.

To verify this premise and assess the viability of Urban Air Mobility (UAM), this research seeks to provide direct answers to the following fundamental questions:

1. Does noise limit the use and maximum operational capacity of U-space airspace in urban environments?
2. Is the cumulative noise generated by the simultaneous and combined operation of multiple drones truly relevant when compared with the analysis of a single mission in isolation?
3. How should acoustic criteria and limits be integrated into the architecture of U-space's automated services (such as Demand Capacity Balancing or Strategic Deconfliction) to ensure sustainable integration?

In order to carry out this research, the main objectives of the study focus, first and foremost, on structuring and applying a detailed physical-mathematical model that accurately calculates sound propagation outdoors, taking into account the attenuation of noise from the drone to the receiver. Once these physical propagation rules have been validated on a controlled environment, the priority objective is to code and implement a large-scale simulation framework, developed in Python, which models the traffic of multiple simultaneous missions over a full 24-hour daily cycle.

2. U-space characterisation

2.1. Definition to U-space

U-space is the European framework designed that allows and studies the safe, efficient and secure integration of unmanned aircraft systems (UAS) in very low altitude airspace (VLL). It is understood as a set of services and operating procedures that support large-scale drone operations, especially in environments with increasing traffic density. The concept was created from the need to manage a growing number of civil and commercial activities with drones, while maintaining safety levels comparable to those of traditional aviation. For instance, to make drones flights orderly U-space implements UAS Traffic Management (UTM) in Europe, leveraging digital services (registries, notifications, etc.). (ENAIRE, 2023)

It is also a set of highly digitized and automated services and procedures that provides an enabling framework to support routine operations with drones and can ensure drone operation in all environments in a seamless manner addressing the support needs of all types of missions and can affect all drone users and categories. Therefore, U-space should not be considered as a defined volume of airspace, segregated and designated for the exclusive use of drones. (CORUS, 2019b, 2019a)



Figure 1 – U-Space (ENAIRE, 2023)

2.2. Applicable regulatory framework

The European regulations applied the U-space system are drawn up and adopted by the European Commission, with the technical support of the European Aviation Safety Agency (EASA) and the participation of the Member States of the European Union, which are then responsible for its implementation at national level.

2.2.1. U-Space regulations

The EU U-space regulatory framework consists of three Implementing Regulations (EU) 2021 – Regulations (EU) 2021/664, 2021/665 and 2021/666 – which entered into application in January 2023. (Gobierno de España - Ministerio de Transportes y Movilidad Sostenible, 2024)

- Regulation (EU) 2021/664 establishes the general regulatory framework for U-space, defining the concept of U-space (geographical areas designated for UAS operations) and the minimum digital services required to fly in it: electronic identification (e-ID), geolocation services, flight authorisation and traffic information. (EUROPEAN COMMISSION, 2021a)

- Regulation (EU) 2021/665 amends Regulation (EU) 2017/373 (common ATM/ANS rules) and imposes interoperability requirements between air navigation service providers (ANSPs/ATMs) and new U-space service providers. Specifically, it requires air traffic controllers to coordinate with USSPs and common information providers (CISPs), exchanging UAS flight plans and surveillance data in designated volumes of U-space. (EUROPEAN COMMISSION, 2021b)

- Regulation (EU) 2021/666 amends the SERA Regulation (EU 923/2012) to adapt the flight rules for manned aviation in U-space: it introduces new communication obligations (8.33 kHz radio frequency, flight plan submission for UAS), mandatory secondary transponders (SSR mode S–ELS) and electronic visibility/safety systems (e.g. ADS-B, FLARM or interoperable DAA) for manned aircraft operating in such spaces. (EUROPEAN COMMISSION, 2021c)

Together, this set of measures enables the formal designation of U-space volumes supported by automated services, ensures interoperability between UAS operators and the traditional ATM system, and updates the SERA to ensure that manned flights in these spaces comply with new identification and separation requirements.

2.2.2. UAS regulations

The EU UAS regulatory framework includes Implementing Regulation (EU) 2019/947 and Delegated Regulation (EU) 2019/945. (Gobierno de España - Ministerio de Transportes y Movilidad Sostenible, 2024)

- Commission Implementing Regulation (EU) 2019/947 establishes the rules and operating procedures applicable to the use of UAS, defining the categories of operations (open, specific, certified) and their limits as specified in Table 1. (EUROPEAN COMMISSION, 2019b)

- Delegated Regulation 2019/945 establishes the technical requirements for UAS (CE marking, technical classes C0–C4 for open category, labelling by class) and regulates third-country operators wishing to fly in the EU as it is in the Table 2. (EUROPEAN COMMISSION, 2019a)

2.3. Typology of operations and UAS

Operations involving unmanned aircraft systems (UAS) within the European framework are structured around a risk-based approach, which takes into account both the technical characteristics of the aircraft and the operational environment in which they are conducted. This conceptual approach enables the definition of proportionate requirements in terms of safety, operational limitations and the necessary level of regulatory oversight. (CORUS, 2019a) (CORUS, 2019b)

To put this risk-based regulatory framework into practice, European legislation (Implementing Regulation (EU) 2019/947) classifies UAS operations into three main categories, which determine the general conditions under which flights may be conducted: Open, Specific and Certified. This initial broad classification considers critical factors such as the type of operation, the environment affected and the level of potential risk to people and other aircraft. (CORUS, 2019a)

Table 1 below details the institutional characteristics, limitations and risk levels associated with each of these three categories:

OPEN	SPECIFIC	CERTIFIED
Low risk	Increased risk	High risk
Without involvement of aviation authority	Safety risk assessment	Comparable to manned aviation
Limitations (visual line of sight, maximum altitude, distance from airport and sensitive zones)	Approved by NAA possibly supported by Quality Entities unless approved operator with privilege	Type Certificate (TC), Certificate of airworthiness, noise certificate, approved organisations, licences
Flight over people is possible in sub-class A1 with drones in classes C0 and C1 only (less than 900g or 80J)	Operation authorisation with operations manual	C2 Link equipment and the remote pilot station could have separate TCs
No overflying crowds	Concept of accredited body	
CE marking	Airworthiness of drone and competence staff based on risk assessment	

Table 1 – Dron operation categories (CORUS, 2019a)

Once the overall risk map has been understood, it is necessary to examine the Open category in greater detail, as it is the most common and the one requiring the strictest technical subdivision. In accordance with (EUROPEAN COMMISSION, 2019a), unmanned aircraft systems operating in this category are classified into different classes (C0 to C4). This classification is based strictly on their intrinsic technical characteristics, such as maximum take-off mass (MTOM), maximum speed and integrated safety equipment. (Aniculaesei, 2026) (CORUS, 2019a)

Linking each aircraft to a specific class allows an indexed set of operational limitations to be associated with it, thereby facilitating risk management both on the ground and in flight. The technical parameters of these classes are described in detail in Table 2:

CLASS	MTOM	MAX SPEED	MAX HEIGHT	MAX NOISE	e-ID	GEO-AWARE	LIGHTS	SERIAL Nº
C0	250g	19 m/s	120m	-	No	No	No	No
C1	900g	19 m/s	120m	60db	Yes	Yes	Yes	Yes
C2	4kg	-	120m	60db	Yes	Yes	Yes	Yes
C3	25kg	-	120m	-	Yes	Yes	Yes	Yes
C4	25kg	-	-	-	-	-	-	-

Table 2 – General Characterisation of drone classes (CORUS, 2019a)

However, the technical specifications of the drone Table 2 are not sufficient on their own to guarantee safety; they must be considered in conjunction with the environment in which it flies. For this reason, operations within the Open category are further divided into operational subcategories: A1, A2 and A3. These subcategories regulate flight conditions based on proximity to people not involved in the operation and the population density of the surrounding area, thereby minimising the risk of impact on the ground. (CORUS, 2019a)

A1	Flights over people (but not over open-air assemblies of people) intended for hobby users flying UAs under 911g - class C1 or C0
A2	Flights close to people, but a safe distance from them for heavier UAS - class C2 - and require passing a recognised theory test
A3	Flights far from people – generally intended for model aircraft clubs - class C0 and C4

Table 3 – Subcategories A1-A2-A3 (CORUS, 2019a)

Beyond geographical, environmental or drone mass constraints, there is a critical human and operational factor: the way in which the operator interacts with the aircraft. Drone operations are also categorised according to the method used by the pilot to maintain situational awareness and visual control of the aircraft during the flight. (CORUS, 2019b)

This interaction gives rise to three fundamental modes of operation, which determine whether control is direct, assisted or fully remote, as set out in Table 4:

VLOS *– Visual Line of Sight*	The drone remains in the sight of the remote pilot at all times.
BLOS *– Beyond Visual Line of Sight*	The drone flies out of sight of the remote pilot or any assistant.
EVLOS *– Extended Visual Line of Sight*	The drone remains in the sight of the remote pilot or an assistant at all times.

Table 4 – Manners of operating a drone (CORUS, 2019b)

All the elements analysed previously (the risk of the operation, the types of drones and the ability to maintain visual contact) must coexist safely within the shared airspace. This is where the U-space ecosystem, specifically designed as part of the CORUS project, comes into its own. U-space introduces a suite of advanced digital services aimed at facilitating UAS traffic management, improving overall situational awareness and ensuring seamless coexistence with traditional manned aviation. (CORUS, 2019a)

To implement this efficiently, Very Low Level (VLL) airspace – where most drones operate – is divided into three distinct volumes according to air traffic control requirements: X, Y and Z (SESAR, 2020). This structuring of airspace allows U-Space services to be dynamically adapted to the required level of automation, traffic density and risk, linking directly to the drone's capabilities and its flight mode (VLOS or EVLOS). (CORUS, 2019a)

The requirements applicable to operators and aircraft in each of these volumes are set out in Table 5:

Type	MTOM
X	No conflict resolution service is offered. There are few basic requirements on the operator, the pilot or the drone. The pilot remains responsible for collision avoidance. VLOS and EVLOS flight are easily possible. Other flight modes in X require (significant) risk mitigation.
Y	Only pre-flight conflict resolution is offered. An approved operation plan. A pilot trained for Y operation. A remote piloting station connected to U-space. A drone and remote piloting station capable of position reporting when available.
Z	Pre-flight conflict resolution and in flight separation are offered. An approved operation plan. A pilot trained for Z operation and/or a compatible, connected automatic drone. A remote piloting station connected to U-space. A drone and remote piloting station capable of position reporting.

Table 5 – U-Space Volumes (CORUS, 2019a)

2.4. U-Space services

The U-space concept is based on the provision of a set of highly automated digital services that enable the safe and efficient management of UAS operations, particularly in high-traffic-density environments.

These services range from basic functions, such as registration or electronic identification, to advanced services for traffic management, operations planning, real-time monitoring and conflict resolution. Their main objective is to provide operators and authorities with comprehensive situational awareness, facilitate decision-making and ensure the safety of both drone operations and manned aviation.

Some of the most important and essential services provided according to CORUS are defined in the Table 6:

Registration	It allows you to register the drone, its owner, operator and pilot, as well as manage and view their details in accordance with the relevant permissions.
Registration assistance	Helps users during the registration process.
e-Identification	Enables remote identification of the drone and access to its information without the need for a physical inspection.
Geo-awareness	It provides up-to-date information on flight restrictions and permitted areas (including NOTAMs, legislation and restricted areas) so that operators know where they can fly before take-off.
Tracking and position reporting	It collects and combines location data to track the movement of drones in real time.
Drone Aeronautical Information Management (AIM)	It keeps airspace information (zones, temporary changes, etc.) up to date and provides it to other services.
Geo-fence provision (Dynamic)	It enables flight restrictions (geofences) to be updated and sent to drones in real time.
Operation plan preparation / optimisation	It helps the operator to prepare and optimise the flight plan.
Operation plan processing	It manages and validates flight plans, including authorisations and checks against airspace status.
Risk analysis assistance	It assesses operational risks by combining information on the environment, traffic and airspace.
Strategic Conflict Resolution	Detects potential conflicts before the flight and suggests solutions.
Emergency Management	Assists the pilot in emergency situations and communicates relevant information.
Monitoring	Monitors the flight in real time and generates alerts regarding deviations, conditions or risks.
Traffic Information	Alerts the operator to other nearby drones to prevent collisions.

Weather Information	Provides meteorological data relevant to the operation.
Procedural Interface with ATC	Enables flights to be coordinated with air traffic control in controlled airspace.
Dynamic Capacity Management	Balances drone traffic demand with airspace capacity during flight plan management.
Tactical Conflict Resolution	Detects conflicts in real time and instructs drones to avoid them (changes in speed, altitude or direction).
Collaborative Interface with ATC	Enables direct communication between the pilot and air traffic control in controlled areas, facilitating standardised instructions and authorisations.

Table 6 – U-Space Services (CORUS, 2019a)

2.5. U-Space implementation

The implementation of U-Space is taking place in stages of increasing automation, progressively incorporating more services and capabilities to optimise the use of airspace and make it safer. The Table 7 and Figure 2 outlines the phases of U-space.

U1	Basic services such as electronic drone registration, electronic identification and geofence management (geographically restricted areas).
U2	Initial operations management, including the submission and approval of flight plans, drone tracking and basic real-time airspace monitoring
U3	Advanced services, such as traffic capacity management and conflict detection in dense environments using collaborative detect-and-avoid (DAA) systems.
U4	Full integration with manned aviation, ensuring complete interoperability. This includes detect-and-avoid systems compatible with manned aircraft, standardised operating procedures and flight management protocols for very low-altitude flights (LFR), as well as strategies to ensure safety, real-time communication and public acceptance (noise, privacy and cybersecurity).

Table 7 – U-Space phases (CORUS, 2019a)

Each phase introduces new services whilst keeping safety as a priority, combining technological advances and regulations to facilitate the efficient operation of drones in increasingly complex environments.

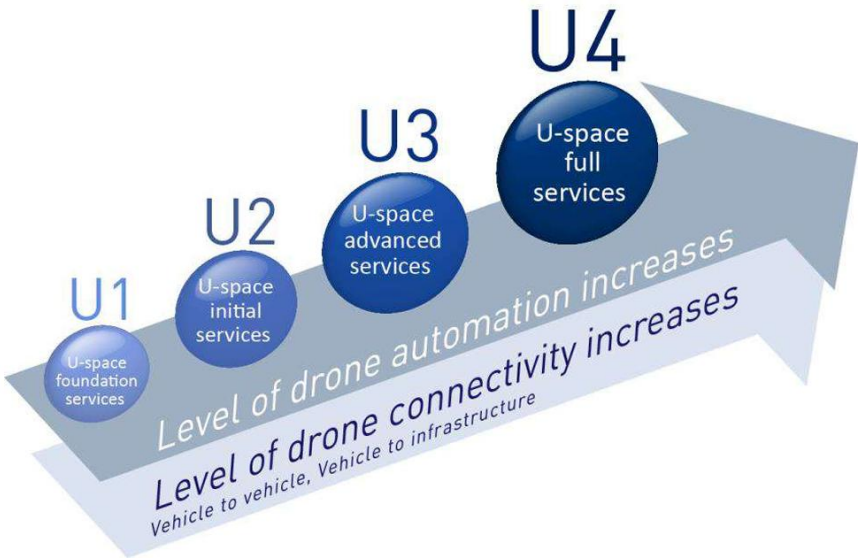


Figure 2 – U-Space Level (CORUS, 2019b)

2.6. U-Space stakeholders

The proper functioning and effective management of U-space involves and requires the participation of multiple stakeholders who belong to the aviation sector or have a direct impact on it.

Cooperation between stakeholders is key to ensuring the safe, efficient, and scalable integration of UAS operations into U-space, as well as ensuring interoperability between manned and unmanned aviation. Each of these stakeholders has a specific role in the defined U-space ecosystem, as shown in the Table 8.

STAKEHOLDER	ROLE
CISP – Common Information Service Providers	CIS providers will disseminate the common information necessary to exchange static and dynamic operational data, serving as the sole and reliable source of all common information in the U-space airspaces under their responsibility. There will be a single CIS provider for each designated U-space airspace or a single provider for each national airspace.
USSP – U-Space Service Providers	U-space services will be provided by certified USSPs during all phases of operation. They will actively coordinate with other ATSPs and USSPs, either directly or through the CIS provider in the case of CIS information, to exchange information and maintain proper operation. There may be more than one per U-space airspace.
ATSP – Air Traffic Service Providers	They are responsible for providing air traffic management services for manned aircraft and assist in the safe integration of UAS traffic with conventional air traffic, especially in controlled airspace.
CNS Providers – Communication, Navigation, Surveillance	They provide the technological infrastructure that enables the communications, navigation, and surveillance necessary for monitoring and controlling air operations.
SDSP – Supplementary Data Service Providers	They offer additional data services that complement U-space services, such as advanced weather information, traffic analysis, or operational planning support services.
Airport operators and vertiports	They manage airport infrastructure and coordinate the integration of drone operations in the vicinity of airports or future urban air mobility infrastructure.
UAS manufacturers	They design and develop unmanned aircraft systems, incorporating the technological capabilities necessary to operate safely within the U-space environment and complying with European regulatory requirements.
Operators, pilots and UAS owners	They are responsible for the execution of drone operations and compliance with applicable regulations, including flight planning and the use of services provided by USSPs.
Security forces and (public) agencies	They ensure regulations are followed, supervise operations, and can intervene in emergency situations or cases of misuse of airspace.
Aviation users and general public	They include other airspace users and society in general, whose social acceptance is essential for the development and expansion of drone operations.

Table 8 – U-space stakeholders (CORUS, 2019a) (EUROPEAN COMMISSION, 2021a) (Gobierno de España - Ministerio de Transportes y Movilidad Sostenible, 2024)

3. Noise characterisation & modelling

The integration of Unmanned Aerial Vehicles (UAVs or drones) into urban environments presents significant challenges, with noise pollution being one of the main factors limiting the social acceptance of the U-space framework. Unlike noise from traditional aviation, the noise generated by drones has specific psychoacoustic characteristics (pure tones, high frequency and rapid fluctuations). Therefore, to assess the feasibility of flight paths and their impact on the population, it is essential to develop an acoustic characterisation model that simulates the propagation of sound from the source (the drone) to ground-based receivers.(Study on the Societal Acceptance of Urban Air Mobility in Europe, 2021)

3.1. Characterisation of sound source

The starting point for any acoustic impact model is the precise characterisation of the aircraft's sound signature at the source of emission. For the purposes of this project, the reference unmanned aerial vehicle (UAV) is a commercial and professional-grade multirotor that has been extensively documented in field acoustic studies: the DJI Phantom 4 Pro Plus.

The technical and kinematic characteristics of this model, which directly determine its sound emission profile and operational impact, are as follows (Miguel-Ayala et al., 2025) (DJI, 2017) (WetalkUAV, 2017):

- Weight: 1388 g.
- Maximum flight time: 30 minutes.
- Wingspan (excluding propellers): 350 mm.
- Maximum horizontal speed: 16 m/s in P mode – Positioning.
- Sound level: 80 dB.

Flight time will change depending on different flight modes and patterns, weather conditions and altitudes.



Figure 3 – DJI Phantom 4 Pro Plus (Luis Fernando Antolín, 2017)

As this is a quadcopter in the under-2-kg weight category (operating under the Open or Specific categories of European regulations), the noise it generates does not come from combustion engines, but is predominantly of an aerodynamic and electromechanical nature. Physically, the emission spectrum is divided into three main components (Insausti et al., 2020):

- Tonal (Rotational) Noise: This is the most dominant acoustic component and the most annoying to the human ear. It is caused by the periodic aerodynamic forces (lift and drag) acting on the propeller blades as they rotate and cut through the air. This noise concentrates most of its energy at the frequency known as the Blade Passing Frequency (BPF) and its respective higher harmonics.
- Broadband noise: This is caused by random aerodynamic phenomena, specifically by the interaction of turbulent boundary layers on the surface of the blades and the vortices that form at the trailing edge of the propellers during flight.
- Engine Noise (Ego-Noise): Caused by mechanical vibrations from structural components and electromagnetic noise resulting from the commutation of the aircraft's brushless electric motors.

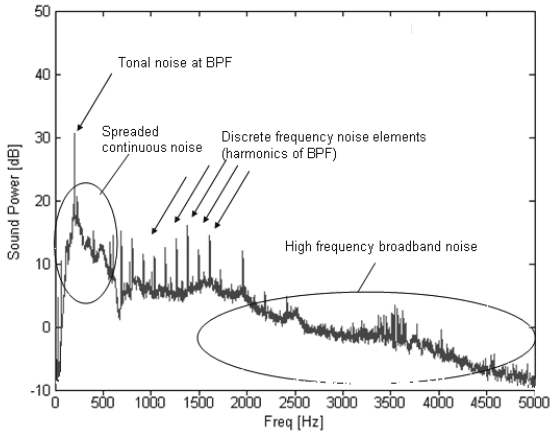


Figure 4 – example of a noise spectrogram: tonal, BPF, Broadband Noise (Miljković, 2018)

For the mathematical modelling and spatial simulation conducted in this study, it is necessary to simplify this complex frequency spectrum into a manageable scalar value. To this end, we follow the official guidelines of the European Aviation Safety Agency (EASA) regarding noise measurement in unmanned aircraft systems (UAS) with a take-off mass of less than 600 kilograms — a regulatory framework within which our 1,375 kg UAV fits perfectly. (Miguel-Ayala et al., 2025)

Using this standardised assessment methodology, the acoustic signature of the DJI Phantom 4 Pro Plus is quantified using a single acoustic reference value: the sound pressure level at one metre ($L_{P,1m}$). This value is input into the geometric algorithm as a spherical (omnidirectional) emission constant, assuming that it represents the average acoustic power of the stabilised drone under cruise and hovering flight conditions.

3.2. Physic-mathematical Model of Sound Propagation in Urban Environments

Once the sound pressure level at the source has been determined ($L_{P,1m}$), It is necessary to assess how this acoustic energy travels through the atmosphere and how it interacts with the city's topography before reaching the human receptor. To this end, a physical-mathematical model of outdoor attenuation has been implemented, which projects noise onto the road network whilst considering the phenomena of divergence, absorption, and urban reflection (Scozzaro et al., 2019). The use of these physics-based models is now crucial for accurately predicting the noise footprint of aircraft in complex urban environments (Merino-Martinez et al., 2024).

3.2.1. Geometry of the direct acoustic path (source-receiver)

The model assumes that the drone is flying at a relative altitude h (above ground level) and is located at a horizontal distance r from the measurement point on the street. The direct line-of-sight distance (d) travelled by the acoustic wavefront from the transmitter to the receiver is calculated using spatial trigonometry as justified in the Equation 1 and represented in the Figure 5:

$$d = \sqrt{h^2 + r^2}$$

d: Direct distance or line-of-sight distance (m).
h: UAV flight altitude (m).
r: Horizontal distance projected onto the ground surface (m).

Equation 1 – Distance Sound Source-Receiver

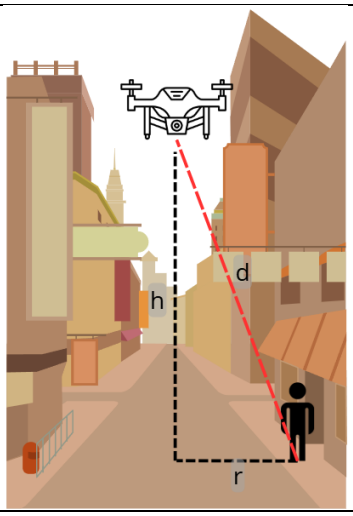


Figure 5 – Distance Sound Source-Receiver

3.2.2. Attenuation due to geometric divergence

The sound emitted by the drone's rotor (considered a point source in free space) propagates in the form of spherical waves. As the wavefront moves away, the initial energy is distributed over the surface of an ever-larger sphere. According to the inverse-square law, sound intensity decreases in proportion to the square of the distance. In the logarithmic decibel scale, and in accordance with the international standard for outdoor sound attenuation this (A_{div}) is modelled by Equation 2 (International Organization for Standardization, 1996):

$$A_{div} = 20 * \log_{10}(d)$$

Equation 2 – Attenuation due to geometric divergence

3.2.3. Attenuation due to atmospheric absorption

As it travels through the air, the sound wave experiences additional dissipative loss due to molecular thermal friction and the relaxation phenomena of oxygen and nitrogen in the atmosphere. This absorption is particularly significant for the high frequencies generated by drone rotors (Scozzaro et al., 2019) (Merino-Martinez et al., 2024). The model incorporates this energy dissipation using a linear attenuation coefficient proportional to the distance travelled in the Equation 3 (International Organization for Standardization, 1996):

$$A_{atm} = \alpha * d$$

Where the coefficient α has been empirically set at 0.005 dB/m for standard weather conditions in urban environments. (20°C – 70% humidity)

Equation 3 – Atmospheric Absorption Attenuation

3.2.4. Direct Sound Pressure Level

Subtracting the physical losses described from the initial emission level gives the sound pressure level that would reach the ground directly, assuming a free-field scenario (with no obstacles) as can be seen in the Equation 4:

$$SPL_{direct} = L_{p,1m} - A_{div} - A_{atm}$$

Equation 4 - Direct SPL

3.2.5. Acoustic Reflections and the "Urban Canyon Effect"

Unlike open spaces, the urban environment features massive architectural obstacles. Streets flanked by buildings physically behave like waveguides or "street canyons". When the wavefront strikes a front wall, a considerable proportion of the energy rebounds back into the street, generating a reverberant field that significantly increases the perceived noise level at ground level. (Richard H. Lyon, 1974)

Recent studies confirm that the presence of these reflections and the urban landscape significantly alter actual propagation in the environments where Urban Air Mobility (UAM) will operate (Merino-Martinez et al., 2024). To simulate this accumulation phenomenon mathematically, it is essential to work within the linear domain of acoustic energy (E):

$$E_{direct} = 10^{(\frac{SPL_{direct}}{10})}$$

Equation 5 – SPL conversion to energy

The number of reflections (n) that the wave undergoes before dissipating depends on the geometric characteristics of the road section (building height and road width). Each successive bounce (i) attenuates a portion of the energy depending on the façade's reflection coefficient (C_{ref}). For this model, C_{ref} has been set at 0.8, representative of standard rigid building materials (concrete, asphalt), which implies that 80% of the incident acoustic energy is reflected at each impact. (Scozzaro et al., 2019)

The total energy (E_{total}) The energy accumulated in the receiver is the sum of the direct energy and the series of multiple reflections. (Richard H. Lyon, 1974)

$$E_{total} = E_{directa} + \sum_{i=1}^{n} (E_{directa} * (C_{ref})^{i})$$

Equation 6 – Total Energy

Justification of the number of rebounds (n)

In acoustic simulations of road cannons, the model can be configured to set, by definition, an upper limit on the number of countable reflections (usually n=2 or n=3) (Kotzen & English, 1999). This simplification is justified by the principle of acoustic energy dissipation and the properties of the logarithmic sum of decibels.

As mentioned earlier, with each impact against a rigid façade, the sound wave loses 20% of its acoustic energy through absorption; added to this is the natural attenuation caused by geometric divergence, due to the greater distance travelled by the reflected sound ray. When the sound contributions at the receiver are summed logarithmically, the energy contributed after three or more reflections is numerically negligible compared to the dominant magnitude of the direct sound field and the early reflections. (David A & Colin H, 2009) (Scozzaro et al., 2019)

A geometric boundary condition is integrated into the algorithm to account for this elastic behavior: if the drone's operational flight altitude exceeds the height of the adjacent building on the street being analyzed, the system automatically disables the generation of acoustic reflections for that section. This criterion assumes that acoustic energy that passes beyond the cornice plane "escapes" freely into the upper atmosphere or undergoes diffraction phenomena over the rooftops, thereby ceasing to contribute to the reverberant field confined within the street section. (Richard H. Lyon, 1974) (Mehra et al., 2013)

This geometric limitation is a standard in urban noise modeling and is consistent with the European CNOSSOS-EU methodological framework (established in European Commission Directive 2015/996) (Kephalopoulos et al., 2012). This standard, which is mandatory to produce noise maps in the European Union, establishes that accounting for multiple specular reflections rarely results in significant variations in the overall sound pressure level, thereby validating the limitation to the first reflections to optimise the model's performance without compromising technical rigour.

3.2.6. Final Sound Pressure Level

Finally, the combined acoustic energy is converted to the standardised logarithmic scale to obtain the final Sound Pressure Level (SPL_{final}), which represents the projected noise at ground level in the urban environment:

$$SPL_{final} = 10 * \log_{10}(E_{total})$$

Equation 7 - Final SPL

3.3. Social impact review

The physical model described in 3.2.6 Final Sound Pressure Level- enables the calculation of the projected final Sound Pressure Level (SPL_{final}) on urban infrastructure. However, for this data to be useful in optimising flight paths, it is necessary to transition from a purely acoustic metric to a socio-environmental impact metric.

Recent research in the field of soundscapes demonstrates that the introduction of drone operations in urban environments generates a perceptual impact that goes beyond acoustic intensity. Through field assessment techniques (soundwalking), it has been shown that drone noise events drastically alter the balance of pre-existing sounds, negatively affecting the psychological dimensions of comfort and calm for the public. (Green & Torija Martinez, 2024)

To quantify and mitigate this perceptual distortion, the algorithm implemented is based on the Environmental and Social Performance Framework developed as part of the European MUSE project for U-space. (Krstić Simić et al., 2024)

To realistically assess public nuisance, the system sets a critical disturbance threshold at 55.0 dB. In accordance with the World Health Organisation's (WHO) guidelines on environmental noise, sound levels below this threshold in urban environments during the day are not considered harmful or disruptive to human activities. (World Health Organization, 2018)

When the SPL_{final} on a section of track exceeds 55 dB, the model processes demographic and track infrastructure data to calculate the following performance indicators shown in the Table 9. (Ganić et al., 2024)

Noise Performance Indicator	Unit	Concept
NO_1	Person	It measures the total number of people in an area who are exposed to a continuous equivalent noise level above a specific threshold (measured in dB) over a given period of time. Essentially, it is a count of how many people are exposed to noise levels exceeding the set limit.
NO_2	Person	It is very similar to NO-1 but is standardised to assess exposure over a full 24-hour day. It counts the number of people in an area who are exposed to a noise level exceeding the set threshold throughout the entire daily cycle.
NO_3	Person	The amount of people exposed to a sound exposure level higher than a certain threshold in dB, for a single drone operation, for a time period fixed by the drone trajectory, within an area.
NO_4	N.person	This indicator takes frequency into account. It multiplies the number of events (N) that exceed a certain noise level by the number of people exposed to those events over a fixed period of time in a given area. It is useful for measuring the impact of repetitive noise (for example, the passage of several drones or aircraft).
NO_5	D.person	This indicator focuses on time. It multiplies the duration (D) during which noise levels exceed the set threshold by the number of people exposed over a given period. It assesses the impact of prolonged noise.

Table 9 - Noise Performance Indicators (Ganić et al., 2024)

3.4. Noise simulation study

To introduce the practical application of the Physic-mathematical Model of Sound Propagation in Urban Environments, Figure 6 graphically illustrates the baseline evolution of the noise level reaching the ground receiver across four different flight altitudes under identical environmental and geometric conditions. This chart enables a direct primary analysis of the correlation between the drone's operational altitude and the resulting sound pressure level at the pedestrian level. As visually demonstrated, the dataset reveals a clear inverse relationship: the higher the drone flies, the lower the sound pressure level that reaches the ground receiver.

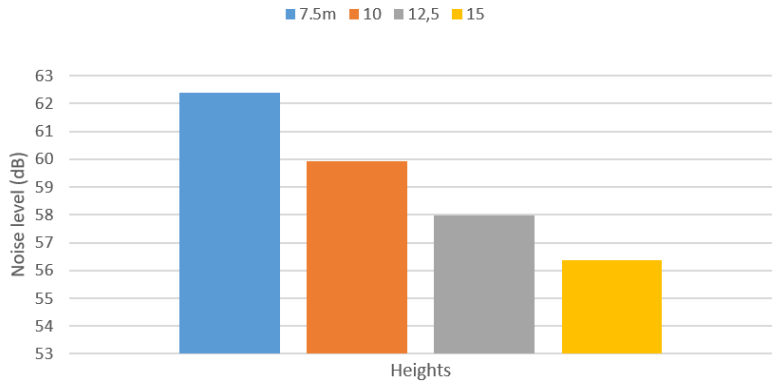


Figure 6 – Sound Pressure Level at different height with the same environment

After applying the Physic-mathematical Model of Sound Propagation in Urban Environments described above -specifically, attenuation due to geometric divergence, atmospheric absorption and the urban canyon effect caused by sound reflections off buildings- the resulting impacts were analysed in a controlled environment comprising a single street 10 metres wide, flanked by building façades 21 metres high and with 3 acoustic reflections at every altitutde. For this initial road-scale simulation, four distinct flight altitudes were defined (7.5 / 10 / 12.5 / 15 metres, with a 2.5 metre interval between each level) in order to evaluate all possible operational combinations, ranging from the presence of a single drone at the lowest level to the simultaneous operation of four drones at each altitude. The acoustic results obtained for the combination of these different altitudes and drone densities are presented in detail in the following table:

Level - 4 (15.0m)	Level - 3 (12.5m)	Level - 2 (10.0m)	Level - 1 (7.5m)	Noise in the receiver (dB)
0	0	0	1	64,94
0	0	1	0	62,46
0	0	1	1	66,88
0	1	0	0	60,52
0	1	0	1	66,28
0	1	1	0	64,61
0	1	1	1	67,79
1	0	0	0	58,94
1	0	0	1	65,91
1	0	1	0	64,06
1	0	1	1	67,53
1	1	0	0	62,81
1	1	0	1	67,01
1	1	1	0	65,65
1	1	1	1	68,32

Table 10 – Noise simulation study

3.4.1. Noise simulation results

As analysed in the Noise simulation study and as shown in the results in Table 10, when applying the Physic-mathematical Model of Sound Propagation in Urban Environments and using the 65 dB limit established by the (Ajuntament de Barcelona, 2020) as a reference since this city serves as the testing scenario, it is concluded that Level 1, which is situated at a height of 7.5 m, is not viable under any circumstances, as it exceeds this tolerance threshold both on its own and in combination with the other flight levels.

Excluding the first level, Level 2 at 10.0 m is acceptable only if it operates in isolation; when combined with any other flight operation at another level, it again exceeds the permitted limit, so their combined use must be ruled out. Consequently, only Levels 3 (12.5 m) and 4 (15.0 m) comply with local regulations, either individually or simultaneously, provided that there is no flight plan for the lower altitudes.

Furthermore, given that the drone's noise emission remains constant at 80 dB both day and night, the perceived noise impact on the receiver does not vary. Consequently, the simulation demonstrates that operations in the current U-space are not feasible during night-time hours, as none of the configurations evaluated manage to comply with the regulatory restrictions for this period. To allow night-time flights, the solution would require raising the operating altitude to ensure that noise levels on the ground do not exceed 55 dB (Ajuntament de Barcelona, 2020) or the 40 dB recommended by the (World Health Organization, 2018).

Once the behaviour of noise at the individual street level has been understood and validated, both through numerical values and their graphical representation on the ground, the next logical step in this research is to scale up the analysis to a much more complex, dynamic and realistic scenario: an entire city. To manage the enormous density of variables, simultaneous trajectories and large-scale urban structures that this involves, a specific large-scale simulation framework has been designed and developed. This working environment will enable the previously validated physical rules to be projected to measure the overall acoustic impact of a UAS transport network across an entire metropolitan area, as described in the Simulation framework.

4. Simulation framework

To implement this framework on a large scale, a Python script has been developed, specifically designed to model and analyse UAS traffic, quantitatively evaluate the impact of the noise generated on the urban population, and enable the study of the operational consequences of this traffic density within the U-Space environment. This software has several specialised modules, strategically programmed to ensure the robustness of the simulation and guarantee the traceability of the data for subsequent statistical and visual analysis.

To clearly illustrate the internal architecture of this tool, Figure 7 presents a flowchart describing the logical structure of the code developed. This diagram provides a sequential and detailed overview of the algorithm's entire life cycle: the process begins with system configuration and the loading of input parameters (such as the drone's speed, the number of flight plans to be generated-ops/h or the coordinates of the environment under study); it progresses through the critical stages of traffic generation and noise propagation calculation; and concludes with the consolidation of the results and the export of graphical and numerical deliverables.

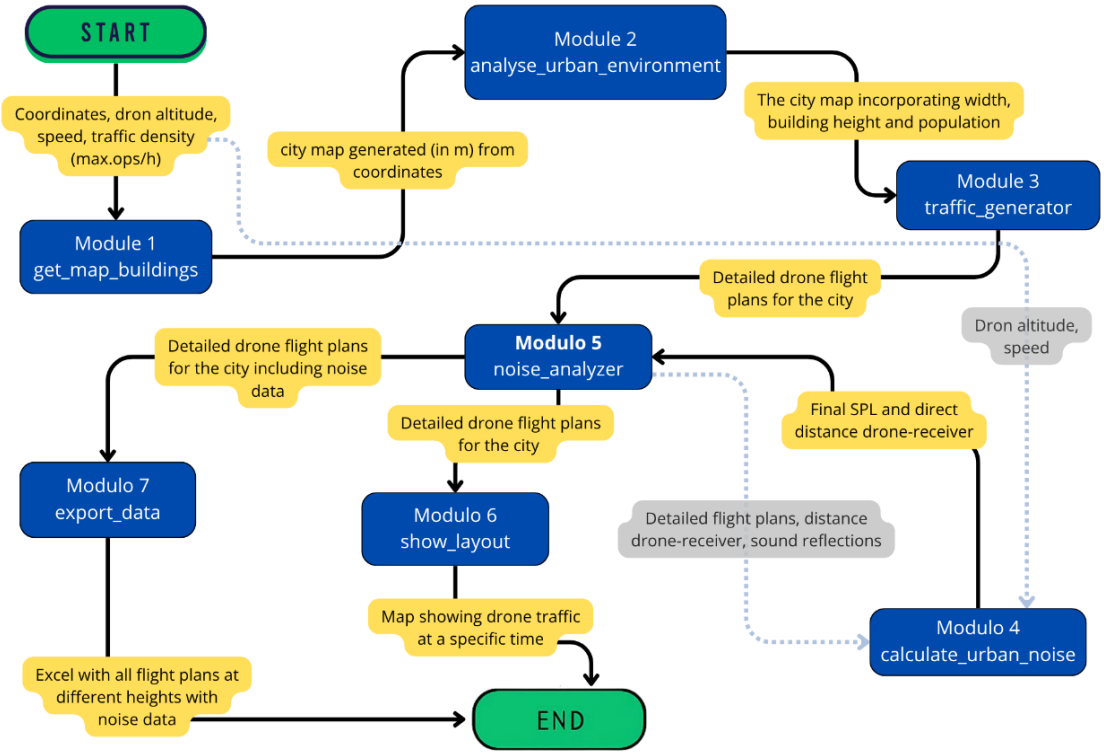


Figure 7 - Logical structure of the code

Having outlined the sequential structure and overall life cycle of the algorithm in Figure 4, it is essential to break down the functional blocks that make it up in detail. To ensure the proper development and implementation of this simulation framework in Python, a set of specialized libraries has been used to optimize the management of geographic data, the analysis of complex networks, and numerical computation.

Specifically, the execution environment includes the following libraries:

- OSMNX for downloading and 3D modelling of the urban environment based on OpenStreetMap.
- NETWORKX for structural, topological, and routing management of the road graph.
- PANDAS for manipulating, filtering, and packaging data tables and flight plans.
- MATPLOTLIB.PYPLOT for the visual and graphical rendering of traffic maps.
- In addition, the native MATH and RANDOM modules are used to solve the physical equations of sound propagation and to model the stochastic injection of hourly demand, respectively.

Therefore, in order to understand the inner workings of the framework developed in Python, each module is analysed individually below, outlining its operational purpose, its input parameters and its return variables or outputs.

MODULE 1: get_map_buildings

- **Function**: Acts as the system's data collector. It is responsible for connecting to OpenStreetMap to download the road network and buildings for a real-world area, converting those spherical coordinates into metres so that all subsequent distance calculations are physically accurate. It is the starting point that constructs the three-dimensional virtual environment by estimating the heights of each structure.
- **Input**: Geographical coordinates (latitude and longitude) as the central point of the area. In addition to the radius in metres that defines the circle from the given coordinate.
- **Output**: Road network graph projected in metres. A table containing the geometry of the buildings in the area is also generated.

For a detailed visual breakdown and the complete source code implementation of this module, see Appendix 1.

MODULE 2: analyse_urban_environment

- **Function:** Acts as the environment characteriser. It is responsible for examining each street to determine its physical width and analysing the adjacent buildings within a 20-metre radius. It is the component that translates the concrete infrastructure into demographic data, estimating the base population living in each section that will be exposed to air traffic.
- **Input:** Street graph in metres. Layer of buildings with their calculated heights.
- **Output:** The same input graph, but integrated with three new attributes for each street (edge): street width, maximum building height and population.

For a detailed visual breakdown and the complete source code implementation of this module, see Appendix 2.

MODULE 3: traffic generator

- **Function:** This module is responsible for simulating the city's behaviour over the course of a day, injecting aircraft flows stochastically according to hourly demand curves (simulating peak and off-peak hours). It is the engine that generates the movement and calculates the exact trajectories in space and time.
- **Input:** Enriched urban graph. Maximum number of flight operations to be generated per hour. Constant drone speed in metres per second (m/s).
- **Output:** Updated graph where each street contains a list of data for every drone that crossed it (hour, minute, second, altitude). List of detailed flight plans for all missions of the day (global ID, departure time, origin/destination nodes, list of trajectory nodes and breakdown of segments with crossing times).

For a detailed visual breakdown and the complete source code implementation of this module, see Appendix 3.

MODULE 4: calculate_urban_noise

- **Function:** Acts as the scientific core of the software. It is responsible for applying the laws of wave physics to calculate sound attenuation from the drone in the air to a receiver on the ground. It is the mathematical engine that evaluates three-dimensional distance, air absorption and the effect of urban reverberation in narrow streets where noise bounces off building facades.
- **Input:** Sound pressure level emitted by the drone measured at 1 metre of distance (in dB). Flight altitude of the drone in metres. Horizontal distance from the vertical line of the drone to the receiver. Number of acoustic reflections off walls.
- **Output:** Final sound pressure level received on the ground (in dB). The straight-line distance (m) calculated between the drone in the air and the receiver on the ground.

For a detailed visual breakdown and the complete source code implementation of this module, see Appendix 4.

MODULE 5: noise_analyzer

- **Function**: It acts as the environmental integrator. It is responsible for cross-referencing flight records with physical equations to assess the cumulative noise footprint over a 24-hour period across the entire city. It is the engine that translates decibels into actual human impact, calculating regulatory urban noise nuisance indices and determining how many residents are affected by noise depending on the type of street.
- **Input:** Graph showing flight records in metres. Base drone noise at 1 metre. Drone speed (m/s).
- **Output:** Returns the modified graph with all noise metrics calculated and integrated into the edges, but projected back onto traditional geographical coordinates (Latitude/Longitude) so that it can be plotted correctly on maps.

For a detailed visual breakdown and the complete source code implementation of this module, see Appendix 5.

MODULE 6: show_layout

- **Function:** It acts as the simulator's graphical interface. It is responsible for converting numerical data into a map that is easily understandable at a glance for a specific time of day. It is the component that renders the city's status at specific points in time, using a traffic light colour code to show congestion and the points of origin or destination of traffic.
- **Input:** Map in Lat/Long coordinates. List of all flight plans for the day. The exact time of day (0 to 23) for which you wish to view the traffic status.
- **Output:** Returns no variables; displays the map on screen showing the traffic generated at a specific time.

For a detailed visual breakdown and the complete source code implementation of this module, see Appendix 6.

MODULE 7: export_data

- **Function:** Acts as the output bridge and advanced optimisation module. It is responsible for simulating a radar system that adjusts the altitudes of the drones to avoid mid-air collisions, and for mathematically processing all binary permutations of vertical occupancy on critical streets. This module packages all the analytics into a multi-tab Excel file ready to feed external optimisation models such as CPLEX.
- **Input:** Graph in Lat/Long coordinates with noise data. List of flight plans for the day. Drone base noise at 1m. Drone speed (m/s). Name of the Excel file to be created.
- **Output:** Does not return variables in Python; returns an .xlsx file with the flights at each altitude.

For a detailed visual breakdown and the complete source code implementation of this module, see Module 7 code implementation.

4.1. Traffic simulation framework

For the traffic simulation framework, the city of Barcelona was selected as the definitive test scenario. This choice is particularly relevant due to the city's characteristic grid-like layout, especially in the Eixample district, which provides a structured and realistic environment for analysing the spatial distribution of UAS traffic and its associated noise impact.

The simulation was parameterised using the geographical coordinates (41.388907, 2.156873) as the central point of the study area, with a radius of 1 km to represent the surrounding urban environment. This configuration allows the framework to project UAS trajectories over a real city map and to evaluate how different levels of drone activity interact with the urban layout.

Within this scenario, the framework generates UAS traffic flows according to a defined operational demand. These flows make it possible to analyse not only the acoustic impact of drone operations, but also the spatial concentration of traffic over the urban grid. In particular, the model allows different traffic intensities and time periods to be compared, providing a basis for evaluating how peak and off-peak demand conditions affect the distribution of operations.

Furthermore, beyond a purely acoustic analysis of sound pressure levels, the programme enables the spatial visualisation of the distribution and density of aircraft traffic over the urban grid. In this scenario, the tool generates a map showing drone traffic at a specific time, indicating the maximum number of operations. By way of example, below is a comparison between a scenario with a limit of 150 operations per hour at 14:00 (considered a peak time due to demand) and the system's behaviour under the same theoretical limit at 06:00:

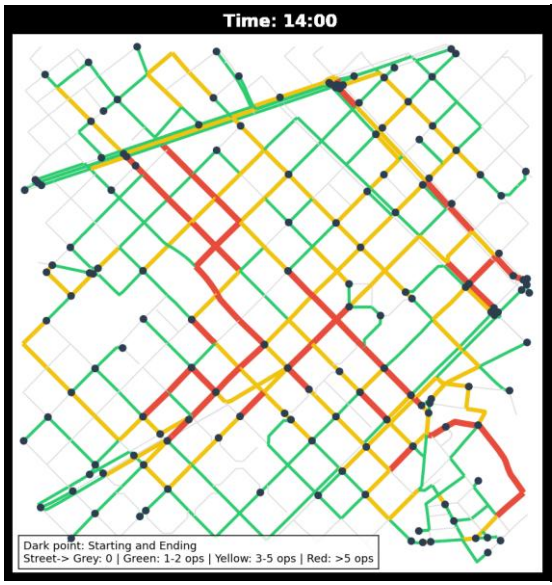


Figure 8 – Traffic Map 14:00h

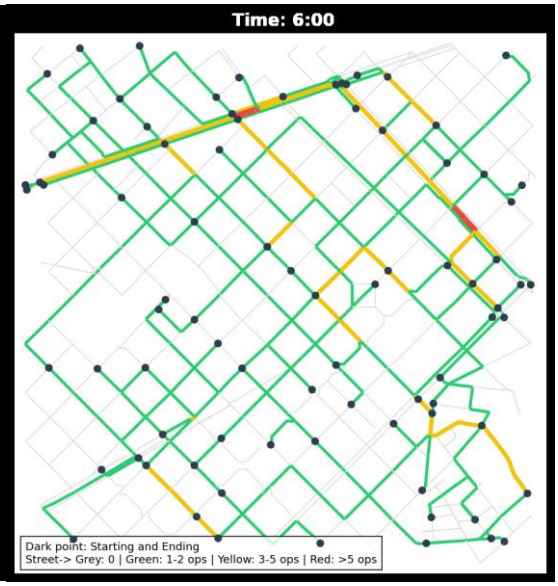


Figure 9 – Traffic Map 6:00h

Edge colours
Green: 1-2 drone ops. | Yellow: 3-5 drone ops. | Red: >5 drone ops

The visual comparison reveals a significant change between the two time slots analysed. In Figure 8, corresponding to 14:00 (peak time), the urban airspace shows a high density of flight plans, accurately reflecting the behaviour and congestion of road traffic in the real world with so many edges/streets in red. In contrast, in Figure 9, at 06:00, the volume of operations is seen to decrease very significantly and greener than the previous one.

This marked variation is due to the implementation in the code of a dynamic "traffic factor or coefficient' for each hour, which has been specifically modelled to bring the simulation closer to the activity patterns of a real-world scenario. Thus, whilst the maximum of 150 operations per hour is only reached during peak demand periods, at 06:00 only 30 per cent of this theoretical capacity (150 × 0.3) is utilised.

The traffic coefficient assigned for the different times of day is set out in the Table 11:

Time (h)	Traffic factor/coefficient
0:00 – 06:00	0,3
07:00 – 11:00 & 13:00 – 16:00	1
17:00 – 20:00	0,8
21:00 – 23:00	0,6

Table 11 – Traffic factor/coefficient

5. Results

Driven by the framework implemented in Python, this phase utilises the maximum number of operations per hour as its primary control parameter. Based on this variable and incorporating traffic flow fluctuations between peak and off-peak periods (previously modelled through traffic coefficients), the system automatically generates comprehensive flight plans across a full 24-hour daily cycle. In this scenario, in comparison to the Noise simulation results, flight plans are generated separately for each of the operational altitudes (7.5, 10, 12.5 and 15 metres). That is to say, the identical set of flight plans is generated while varying the initial altitude; however, in the event of a conflict between two drones, the flight level of one of the flight plans is dynamically reassigned.

Through this process, the resulting daily flight plans capture comprehensive metrics essential for air traffic management and environmental impact analysis; specifically: the flight plan number, the origin and destination nodes, the street or arc identifier (ID), the exact time of passage through that section, the drone's instantaneous altitude, the noise level actually reaching the ground receiver and noise metrics defined in Social impact review. An example of this structured record is shown in Table 12 when the maximum number of operations per hour is 150:

Flight Plan	Starting node	Destination node	Edge/Street ID	Time	Altitude (m)	Noise in the receiver (dB)	NO1 (person)	NO2 (person)	NO3 (person)	NO4 (N.person)	NO5 (D.person)
516	30238017	30246860	1409614868	9:54:00	7,5	67,09	460	0	460		3710,54
516	30246860	30246820	1352761260	9:54:08	7,5	62,38	0	0	0	0	0
516	30246820	30246778	1352761260	9:54:16	7,5	62,38	0	0	0	0	0
516	30246778	30237587	449991621	9:54:24	**10**	59,91	0	0	0	0	0
516	30237587	30246777	449991621	9:54:33	**10**	59,91	0	0	0	0	0
516	30246777	1624948210	1408598861	9:54:41	**10**	59,91	0	0	0	0	0
516	1624948210	30246768	1408598861	9:54:45	**10**	59,91	0	0	0	0	0
516	30246768	30237610	4744944	9:54:49	**10**	64,61	0	0	0	0	0
516	30237610	30313113	4744944	9:54:58	**10**	64,61	0	0	0	0	0
516	30313113	4470657985	237960948	9:55:05	**10**	59,91	0	0	0	0	0
516	4470657985	30237612	1408598864	9:55:07	**10**	64,61	0	0	0	0	0
516	30237612	30237613	1408598865	9:55:14	**10**	64,61	0	0	0	0	0
516	30237613	30237614	1408598866	9:55:23	7,5	67,09	170	0	170	340	1159,26
516	30237614	30237615	1408598867	9:55:31	7,5	67,09	170	0	170	510	1738,89
516	30237615	30237635	237914520	9:55:40	7,5	64,94	0	0	0	0	0
516	30237635	30237637	1409649743	9:55:48	7,5	64,94	0	0	0	0	0
516	30237637	1362899971	536376965	9:55:50	7,5	64,94	0	0	0	0	0
516	1362899971	1311619488	237659844	9:55:51	7,5	62,38	0	0	0	0	0
516	1311619488	1311619445	544617969	9:56:10	7,5	62,38	0	0	0	0	0

Table 12 – Flight Plan #516

Looking at Table 12, the drone's altitude varies along its flight path. As mentioned, when describing the system's operational rules, the Simulation framework algorithm features a dynamic conflict-resolution mechanism: when it detects the proximity of another aircraft, it automatically readjusts the flight level by increasing the altitude by 2.5 metres (7.5m -> 10m) to ensure a safe spatial separation and faithfully replicate air traffic management in a real-world operational environment. This temporary increase in flight altitude simultaneously results in a slight reduction in the noise impact perceived by the receiver on the ground during that specific section of the flight. Once the flight paths no longer intersect and the risk of conflict has passed, the drones descend to return to their original cruising altitude.

This behaviour is clear when analysing the interaction between actual flight paths within the simulation: – Flight Plan #516 comes into conflict with – Flight Plan #497, specifically from waypoint 449991621 onwards, which immediately triggers this safety vertical climb. To illustrate this transition and the data overlap, the following Table 13 shows a breakdown of – Flight Plan #497:

Flight Plan	Strating node	Destination node	Edge/Street ID	Time	Altitude (m)	Noise in the receiver (dB)	NO1 (person)	NO2 (person)	NO3 (person)	NO4 (N.person)	NO5 (D.person)
497	30238021	30238020	730337227	09:54:00	7,5	62,38	0	0	0	0	0
497	30238020	4840145821	730337227	09:54:08	7,5	67,09	886	0	886	2604	7397,92
497	**4840145821**	**30246778**	**449991621**	**09:54:16**	**7,5**	**67,09**	**788**	0	**788**	**1806**	**8941,53**
497	**30246778**	**30237587**	**449991621**	**09:54:24**	**7,5**	**62,38**	0	0	0	0	0
497	**30237587**	**30246777**	**449991621**	**09:54:33**	**7,5**	**62,38**	0	0	0	0	0
497	30246777	1624948210	1408598861	09:54:41	7,5	62,38	0	0	0	0	0
497	1624948210	30246768	1408598861	09:54:45	7,5	62,38	0	0	0	0	0
497	30246768	30237610	4744944	09:54:49	7,5	67,09	386	0	386	1230	1525,78
497	30237610	30313113	4744944	09:54:58	7,5	67,09	627	0	627	2357	5212,08
497	30313113	4470657985	237960948	09:55:05	7,5	62,38	0	0	0	0	0
497	4470657985	30237612	1408598864	09:55:07	7,5	67,09	450	0	450	1584	2648,61
497	30237612	30237613	1408598865	09:55:14	7,5	67,09	508	0	508	1933	4818,60
497	30237613	30237633	4744966	09:55:23	7,5	64,94	0	0	0	0	0
497	30237633	1311773901	4744966	09:55:32	7,5	64,94	0	0	0	0	0
497	1311773901	30237770	4744966	09:55:34	7,5	64,94	0	0	0	0	0
497	30237770	364811897	237659855	09:55:40	7,5	64,94	0	0	0	0	0
497	364811897	1311619500	536376972	09:55:42	7,5	67,09	515	0	515	2154	4908,85
497	1311619500	1311619340	397481042	09:55:43	7,5	62,38	0	0	0	0	0

Table 13 – Flight Plan #497

This dynamic behaviour and the data collected over the 24-hour daily cycle empirically reaffirm and validate the conclusions originally drawn in the Noise simulation results. Despite the temporary 2.5 metre elevations designed to mitigate the risk of collision, which provide localised relief from sound pressure, the large-scale simulation within the Simulation Framework Result confirms that there are numerous urban sections where the baseline noise generated at Level 1 (7.5 m) systematically exceeds the maximum permitted limit of 65 dB set by (Ajuntament de Barcelona, 2020). This demonstrates that continuous operation at realistic densities at the lowest altitude is unsafe for the city's acoustic health, forcing the system to rely constantly and saturate the upper levels of the U-Space airspace.

6. Conclusion

The generalized introduction of Unmanned Aircraft Systems (UAS) in Urban Air Mobility (UAM) environments should ideally include an operational framework that contemplates environmental restrictions alongside traditional physical risks of collision.

This final thesis has addressed this issue by developing and implementing a simulation framework in Python, thereby rigorously validating the hypotheses presented in the introduction:

Does noise limit the use and maximum operational capacity of U-space airspace in urban environments?

According to the data collected, it has been categorically verified that noise critically limits the use and capacity of U-space airspace. The noise restrictions imposed by local ordinances act as an invisible physical barrier; operating without prior environmental planning would quickly exceed the limits tolerated by the population, demonstrating that noise criteria influence the geometric design and availability of VLL routes in the same way as traditional operational risks.

Is the cumulative noise generated by the simultaneous and combined operation of multiple drones truly relevant when compared with the analysis of a single mission in isolation?

It has been empirically proven that the cumulative noise level is an extremely important and critical factor in urban fleet management. Due to the strictly logarithmic nature of the decibel scale, the simultaneous presence or continuous succession of multiple drones drastically amplifies the overall sound energy at ground level.

This significance was demonstrated in the Noise simulation study, which evaluated the combination of different altitudes and simultaneous drone operations in a controlled environment (a 10-meter-wide street with 21-meter-high building facades). The study demonstrated that Level 1 (7.5 m) is completely unfeasible in any combined scenario, as it systematically exceeds the limit of 65 dB set by local regulations. The massive intersection of flight paths within the general framework confirms that, when combining simultaneous operations, even Level 2 (10 m) is compromised, confirming that the cumulative noise necessitates restricting traffic or mandatorily shifting routes to higher altitudes (12.5 m and 15 m) to avoid violating Barcelona's local noise protection laws.

How should acoustic criteria and limits be integrated into the architecture of U-space's automated services (such as Demand Capacity Balancing or Strategic Deconfliction) to ensure sustainable integration?

To address the issue of accumulated noise overload detected in flight plans, this thesis offers an innovative methodological solution by defining a sequential and hierarchical chain of three work blocks within the U-space automated services architecture (SESAR, 2020):

1. Acoustic Optimization Layer: This serves as the ecosystem's first preventive and binding filter. It applies the physical laws of the model to mathematically define the "dynamic environmental capacity" of each street or sector, setting the maximum allowable decibel level, number of simultaneous aircraft before ordinary traffic services are permitted or minimum drone flight altitude.
2. Demand and Capacity Balancing (DCB): Once the acoustic filter has determined the ecological capacity of the environment, the DCB service intervenes at the aggregate traffic scale. Its function is to balance and regulate the total demand from operators; if the hourly accumulation of missions in a corridor threatens to exceed noise limits, the DCB intervenes by proactively restricting or rescheduling incoming flows.
3. Strategic Conflict Resolution: Finally, with traffic volume already balanced and validated in environmental terms by the DCB, this service operates at the individual mission level just before take-off. It is responsible for resolving specific geometric overlaps between individual aircraft to prevent mid-air collisions, safely applying the vertical separation algorithm (such as the 2.5-meter safety margin modelled in our code).

To visualize how these three layers are structurally integrated into the verification process of a flight plan, Figure 10 outlines the operational workflow:

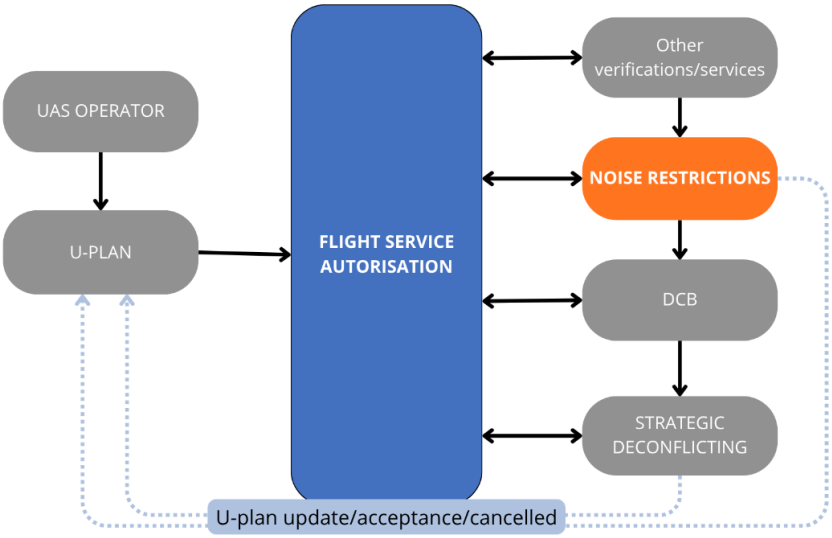


Figure 10 – Integration of noise restrictions, DCB, and strategic deconflicting within the Flight Service Authorisation framework

As illustrated in the Figure 10, the Flight Service Authorisation interface coordinates these gates hierarchically. Acoustic Optimization Layer (highlighted in orange) serves as the primary gateway of the mentioned; if a U-Plan passes this environmental filter, it progresses to DCB and then Strategic Deconflicting.

In conclusion, this sequential approach would provide a solid guarantee that future operations in U-space not only meet the mathematical safety requirements for preventing physical collisions but are also rigorously sustainable from an environmental and legal perspective, ensuring the success, legal compliance with municipal regulations, and long-term social acceptance of unmanned aviation in our cities. (Study on the Societal Acceptance of Urban Air Mobility in Europe, 2021) (SESAR, 2020)

7. References

Ajuntament de Barcelona. (2020). *On es localitza la població més exposada al soroll?* www.bcnregional.com

Aniculaesei, A. (2026). Regulatory Frameworks and Development Standards for Civilian Unmanned Aircraft Systems: From Regulatory Safety Intent to Development Lifecycles. *Drones*, *10*(4). https://doi.org/10.3390/drones10040271

Christian, A., & Cabell, R. (2017). *Initial Investigation into the Psychoacoustic Properties of Small Unmanned Aerial System Noise*. https://ntrs.nasa.gov/api/citations/20170005870/downloads/20170005870.pdf

CORUS. (2019a). *U-Space Concept of Operations Enhanced Overview_Vol1*. https://doi.org/10.10.10

CORUS. (2019b). *U-Space Concept of Operations_Vol2*.

David A, B., & Colin H, H. (2009). *Engineering Noise Control*. www.sponpress.com

DJI. (2017). *Phantom 4 Pro/Pro Plus User Manual_ES*. https://dl.djicdn.com/downloads/phantom_4_pro/20170719/Phantom_4_Pro_Pro_Plus_User_Manual_ES.pdf

EASA. (2026, January 21). *Más de 150.000 operadores de UAS registrados en AESA en 2025*. https://www.seguridadaerea.gob.es/es/noticias/mas-de-150000-operadores-de-uas-registrados-en-aesa-en-2025

ENAIRE. (2023). *U-Space y el rol de ENAIRE*. https://www.enaire.es/servicios/drones/todo_lo_necesario_para_volar_tu_dron/uspace_y_el_rol_de_enaire

EUROPEAN COMMISSION. (2019a). *COMMISSION DELEGATED REGULATION (EU) 2019/ 945 - of 12 March 2019 - on unmanned aircraft systems and on third-country operators of unmanned aircraft systems*. http://data.europa.eu/eli/reg_impl/2019/945/oj

EUROPEAN COMMISSION. (2019b). *COMMISSION IMPLEMENTING REGULATION (EU) 2019/ 947 - of 24 May 2019 - on the rules and procedures for the operation of unmanned aircraft*. http://data.europa.eu/eli/reg_impl/2019/947/oj

EUROPEAN COMMISSION. (2021a). *COMMISSION IMPLEMENTING REGULATION (EU) 2021/664 of 22 April 2021 on a regulatory framework for the U-space (Text with EEA relevance)*. http://data.europa.eu/eli/reg_impl/2021/664/oj

EUROPEAN COMMISSION. (2021b). *COMMISSION IMPLEMENTING REGULATION (EU) 2021/665 - of 22 April 2021 amending Implementing Regulation (EU) 2017/373 as regards requirements for providers of air traffic management/air*

navigation services and other air traffic management network functions in the U-space airspace designated in controlled airspace. http://data.europa.eu/eli/reg_impl/2021/665/oj

EUROPEAN COMMISSION. (2021c). *COMMISSION IMPLEMENTING REGULATION (EU) 2021/666 of 22 April 2021 amending Regulation (EU) No 923/2012 as regards requirements for manned aviation operating in U-space airspac*. http://data.europa.eu/eli/reg_impl/2021/666/oj

Ganić, E., Simić, T. K., Baena, M., Mocholí, D., Alonso, M., LeGriffon, I., Ruaud, E., & Barrado, C. (2024). *D3.1 U-space Environmental and Social Performance Framework*.

Gobierno de España - Ministerio de Transportes y Movilidad Sostenible. (2024). *Divulgacion del concepto u-space*. https://www.transportes.gob.es/recursos_mfom/paginabasica/recursos/divulgacion_del_concepto_u-space_v1_vfinal.pdf

Green, M. C., & Torija Martinez, A. J. (2024). *Soundwalking in Salford: A Soundscape Approach to Drone Noise Assessment Session: Community Impact, Engagement and Perception*.

Insausti, X., Hogstad, B. O., & Patzold, M. (2020). Modelling and Simulation of Ego-Noise of Unmanned Aerial Vehicles. *IEEE Vehicular Technology Conference*, *2020-May*. https://doi.org/https://doi.org/10.1109/VTC2020-Spring48590.2020.9128572

International Organization for Standardization. (1996). *CD 4.48 International Standard ISO 9613-2 – Acoustics – Attenuation of sound during propagation outdoors – Part 2 General method of calculation*. https://www.warrington.gov.uk/sites/default/files/2023-04/CD%204.48%20%20%20%20International%20Standard%20ISO%209613-2%20%E2%80%93%20Acoustics%20%E2%80%93%20Attenuation%20of%20sound%20during%20propagation%20outdoors%20%E2%80%93%20Part%202%20General%20method%20of%20calculation.pdf

Kephalopoulos, Stylianos., Paviotti, Marco., & Anfosso-Lédée, Fabienne. (2012). *Common noise assessment methods in Europe (CNOSSOS-EU) : to be used by the EU Member States for strategic noise mapping following adoption as specified in the Environmental Noise Directive 2002/49/EC*. OPEU.

Kotzen, Benz., & English, Colin. (1999). *Environmental noise barriers : a guide to their acoustic and visual design*. https://ndl.ethernet.edu.et/bitstream/123456789/45445/1/202.pdf

Krstić Simić, T., Ganić, E., Mirković, B., Baena, M., LeGriffon, I., & Barrado, C. (2024). U-Space Social and Environmental Performance Indicators. *Drones*, *8*(10). https://doi.org/10.3390/drones8100580

Luis Fernando Antolín. (2017, October 9). *DJI Phantom 4 Pro [Photo]*. Grupo TYC GIS Formación. https://www.cursosteledeteccion.com/dji-phantom-4-pro/

Mehra, R., Raghuvanshi, N., Antani, L., Chandak, A., Curtis, S., & Manocha, D. (2013). Wave-based sound propagation in large open scenes using an equivalent source formulation. *ACM Transactions on Graphics*, *32*(2). https://doi.org/10.1145/2451236.2451245

Merino-Martinez, R., Snellen, M., Altena, A., Amiri-Simkooei, A., Cappagli, C., Morin, A., Quaroni, L. N., Yunus, F., & Yupa-Villanueva, R. (2024, September). Research on drone and urban air mobility noise: Measurement, modelling, and human perception Session. *QuietDrones 2024*.

Miguel-Ayala, P., Estévez-Mauriz, L., Romero-Romero, C., & Torija, A. J. (2025). *ON-FIELD MEASUREMENTS FOLLOWING THE EASA GUIDELINES ON NOISE MEASUREMENTS OF UNMANNED AIRCRAFT SYSTEMS LIGHTER THAT 600 KG*. https://doi.org/10.61782/fa.2025.0656

Miljković, D. (2018, May). *Methods for Attenuation of Unmanned Aerial Vehicle Noise* . ResearchGate. https://www.researchgate.net/figure/Sources-of-propeller-aerodynamic-noise-according-to-23_fig3_325426655/actions#reference

Richard H. Lyon. (1974). Role of multiple reflections and reverberation in urban noise propagation. *The Journal of the Acoustical Society of America*, 493–503. https://doi.org/https://doi.org/10.1121/1.1914527

Scozzaro, G., Delahaye, D., & Vela, A. E. (2019). *Noise Abatement Trajectories for a UAV Delivery Fleet*. https://www.sesarju.eu/sites/default/files/documents/sid/2019/papers/SIDs_2019_paper_53%20v2.pdf

SESAR. (2020). *SUPPORTING SAFE AND SECURE DRONE OPERATIONS IN EUROPE A report of the consolidated SESAR U-space research and innovation results*. https://doi.org/10.2829/935445

Sesar. (2023). *U-SPACE CONCEPT OF OPERATIONS (CONOPS)*. https://www.sesarju.eu/sites/default/files/documents/reports/U-space%20CONOPS%204th%20edition.pdf

Study on the societal acceptance of Urban Air Mobility in Europe. (2021). https://www.easa.europa.eu/sites/default/files/dfu/uam-full-report.pdf

Torija, A. J., Chaitanya, P., & Li, Z. (2020). *Psychoacoustic analysis of contra-rotating propeller noise for Unmanned Aerial Vehicles*.

WetalkUAV. (2017, February 18). *DJI Drones Sound Test - How noisy is your drone? [Video]*. YouTube. https://www.youtube.com/watch?v=uprXhH6-FNI&t=137s

World Health Organization. (2018). *NOISE GUIDELINES for the European Region*. World Health Organization. Regional Office for Europe. https://iris.who.int/server/api/core/bitstreams/f53c45ba-11d3-4502-a424-c1cf49f5a053/content

8. Appendix

Module 1 code implementation

```python
# ==========================================================================
# MÓDULO 1: DESCARGA Y CREACIÓN DEL GRAFO (MAPA BASE Y EDIFICIOS)
# ==========================================================================
def get_map_buildings(punto_central, radio):
    print(f"1. Descargando mapa base y edificios en radio de {radio}m...")
    # Descarga la red vial para vehículos en el radio configurado
    G = ox.graph_from_point(punto_central, dist=radio, network_type="drive")

    try:
        # Descarga geometrías de edificios con la etiqueta 'building'
        edificios = ox.features_from_point(punto_central, dist=radio, tags={'building': True})
    except Exception:
        print(" -> No se encontraron edificios en esta zona.")
        edificios = None

    print("2. Proyectando coordenadas a metros...")
    # Es pasar de grados (lat/lon) a metros para poder calcular distancias reales
    G_proj = ox.project_graph(G)
    edificios_proj = None

    # Si hay edificios, estimamos sus alturas en metros si no vienen especificadas
    if edificios is not None and not edificios.empty:
        alturas = []
        for _, row in edificios.iterrows():
            altura = 0
            # Si el edificio ya incluye su altura en metros
            if 'height' in row and not pd.isna(row['height']):
                try: altura = float(row['height'].replace('m', '').strip())
                except: pass
            # Si incluye el número de pisos, estimamos 3 metros por cada planta
            elif 'building:levels' in row and not pd.isna(row['building:levels']):
                try: altura = float(row['building:levels']) * 3.0
                except: pass
            alturas.append(altura)

        # Guardamos la altura calculada y proyectamos los edificios al mismo sistema métrico del mapa
        edificios['altura_calculada'] = alturas
        edificios_proj = edificios.to_crs(G_proj.graph['crs'])

    return G_proj, edificios_proj
```

Appendix 1 – Module 1: get_map_buildings

Module 2 code implementation

```python
# ==========================================================================
# MÓDULO 2: ESTUDIO DEL ENTORNO FÍSICO (CALLES Y ESTIMACIÓN DE POBLACIÓN)
# ==========================================================================
def analyse_urban_environment(G_proj, edificios_proj):
    print("3. Analizando entorno urbano (calles y edificios)...")

    # Recorremos cada calle (arista/edge) del mapa
    for u, v, key, data in G_proj.edges(keys=True, data=True):
        # 1. Determinación de la anchura de la calle
        anchura = data.get('width', None)
        if isinstance(anchura, list): anchura = anchura[0]
        try:
            data['anchura_calle'] = float(anchura)
        except:
            # Si no hay ancho registrado, lo estimamos multiplicando el número de carriles por 3 metros + acera de 3m (1.5m por cada lad
            lanes = data.get('lanes', 1)
            if isinstance(lanes, list): lanes = lanes[0]
            try: data['anchura_calle'] = (float(lanes) * 3.0) + 3
            except: data['anchura_calle'] = 5.0 # Valor por defecto si no hay datos

        altura_max = 0.0
        poblacion_base = 0

        # 2. Estimación de habitantes en el área de influencia de la calle
        if edificios_proj is not None and not edificios_proj.empty and 'geometry' in data:
            # Creamos un pasillo/colchón de 20 metros alrededor del eje de la calle
            zona_influencia = data['geometry'].buffer(20)
            # Buscamos qué edificios intersecan con este pasillo de 20 metros
            edificios_cercanos = edificios_proj[edificios_proj.geometry.intersects(zona_influencia)]

            if not edificios_cercanos.empty:
                # Registramos el edificio más alto de esa calle para el posterior cálculo de rebotes acústicos
                altura_max = edificios_cercanos['altura_calculada'].max()

                # Estimación de la población de cada edificio
                for _, edificio in edificios_cercanos.iterrows():
                    area_m2 = edificio.geometry.area
                    altura = edificio['altura_calculada']
                    if altura <= 0: altura = 3.0 # Mínimo un piso

                    plantas = max(1, int(altura / 3.0))
                    # Fórmula urbana estándar: (Área / 80m² por piso) * número de pisos * 2.5 personas por vivienda
                    # García-Palomares, J. C., Gutiérrez, J., & Romanillos, G. (2012).
                    #Estimación de la población a partir de datos catastrales y de teledetección: aplicaciones en el análisis urbano.
                    #Anales de Geografía de la Universidad Complutense.
                    poblacion_edificio = (area_m2 / 80.0) * plantas * 2.5
                    poblacion_base += poblacion_edificio

        # Guardamos estos datos directamente dentro de las propiedades de la calle
        data['altura_max_edificios'] = round(altura_max, 2)
        data['poblacion_base'] = int(poblacion_base)

    return G_proj
```

Appendix 2 – Module 2: analyse_urban_environment

Module 3 code implementation

```python
# ==============================================================================
# MÓDULO 3: GENERADOR DE TRÁFICO (TRAFFIC GENERATOR)
# ==============================================================================
def traffic_generator(G_proj, traffic_density, velocidad):
    print("4. Generando rutas por franjas horarias...")

    nodos_lista = list(G_proj.nodes())
    planes_de_vuelo = []

    # Preparamos una lista vacía en cada calle para registrar los vuelos exactos que pasen por ella
    for u, v, key, data in G_proj.edges(keys=True, data=True):
        data['registro_vuelos_exactos'] = []

    id_vuelo_global = 1

    # Simulamos hora por hora (de 0 a 23)
    for hora in range(24):
        # Aplicamos un factor de demanda comercial según la hora del día
        if 0 <= hora <= 6: factor = 0.3       # Madrugada: Actividad muy baja
        elif 7 <= hora <= 11 and 13 <= hora <= 16: factor = 1.0 # Horas pico laborales
        elif 17 <= hora <= 20: factor = 0.8   # Tarde/Salida trabajo
        else: factor = 0.6                    # Noche / Transición

        # Número de vuelos a generar en esta hora en concreto
        operaciones_esta_hora = int(traffic_density * factor)

        for _ in range(operaciones_esta_hora):
            intentos = 0
            while intentos < 10: # Límite de intentos para evitar bucles si no hay caminos conectados
                try:
                    origen = random.choice(nodos_lista)
                    destino = random.choice(nodos_lista)

                    if origen != destino:
                        # Encuentra la ruta más corta en metros usando algoritmo Dijkstra (shortest_path)
                        ruta = nx.shortest_path(G_proj, origen, destino, weight='length')
                        minuto_salida = random.randint(0, 59)

                        # Convertimos el tiempo de salida a segundos absolutos del día
                        tiempo_actual_seg = (hora * 3600) + (minuto_salida * 60)
                        tramos_detallados = []

                        # Registramos el paso del dron tramo por tramo de la ruta
                        for i in range(len(ruta) - 1):
                            u = ruta[i]
                            v = ruta[i+1]
                            data_arista = G_proj[u][v][0]

                            distancia = data_arista.get('length', 10.0)
                            # Tiempo en segundos que tarda el dron en recorrer este tramo (t = d / v)
                            tiempo_cruce = distancia / velocidad

                            # Calculamos la estampa de tiempo exacta del cruce
                            h_actual = int((tiempo_actual_seg // 3600) % 24)
                            m_actual = int((tiempo_actual_seg % 3600) // 60)
                            s_actual = int(tiempo_actual_seg % 60)

                            # Registramos el vuelo en la calle (asumimos altura inicial de 15 metros)
                            data_arista['registro_vuelos_exactos'].append((h_actual, m_actual, s_actual, 15.0))

                            tramos_detallados.append({
                                'u': u, 'v': v,
                                'hora': h_actual, 'minuto': m_actual, 'segundo': s_actual,
                                'tiempo_cruce_seg': tiempo_cruce
                            })
                            # El tiempo avanza para el siguiente tramo del mismo viaje
                            tiempo_actual_seg += tiempo_cruce

                        # Guardamos el plan de vuelo completo
                        planes_de_vuelo.append({
                            'id_global': id_vuelo_global,
                            'hora_salida': hora,
                            'origen': origen,
                            'destino': destino,
                            'trayectoria': ruta,
                            'tramos': tramos_detallados
                        })
                        id_vuelo_global += 1
                        break
                except nx.NetworkXNoPath:
                    intentos += 1 # Si los nodos no están conectados de forma vial, reintenta

    print(f" -> Total de misiones generadas en el día: {len(planes_de_vuelo)} vuelos.")
    return G_proj, planes_de_vuelo
```

Appendix 3 – Module 3: traffic_generator

Module 4 code implementation

```python
# ==========================================================================
# MÓDULO 4: ACÚSTICO (MODELO MATEMÁTICO DE PROPAGACIÓN FÍSICA)
# ==========================================================================
def calculate_urban_noise(lp_1m, altura, distancia_horizontal, num_rebotes):
    # Distancia en línea recta (hipotenusa) desde el dron volando hasta el receptor en el suelo (centro de la calle hasta el receptor en "acera")
    distancia_directa = math.sqrt(altura**2 + distancia_horizontal**2)
    if distancia_directa == 0:
        return lp_1m, 0

    # 1. Atenuación por divergencia geométrica esférica: 20 * log10(distancia)
    atenuacion_distancia = 20 * math.log10(distancia_directa)
    # 2. Atenuación atmosférica: disipación en el aire (aprox. 0.005 dB por metro)
    atenuacion_aire = 0.005 * distancia_directa

    # Nivel de presión sonora directo (SPL) que llega al suelo
    spl_directo = lp_1m - atenuacion_distancia - atenuacion_aire

    # Pasamos de la escala logarítmica (dB) a la escala energética lineal para poder sumar los ecos
    energia_directa = 10 ** (spl_directo / 10)
    energia_total = energia_directa
    coeficiente_reflexion = 0.8 # Las fachadas de hormigón/cristal reflejan el 80% de la energía sonora

    # Sumamos la energía de las ondas que rebotan en las fachadas de la calle estrecha
    for i in range(1, num_rebotes + 1):
        energia_rebote = energia_directa * (coeficiente_reflexion ** i)
        energia_total += energia_rebote

    # Volvemos a transformar la energía total sumada a la escala de decibelios (dB)
    spl_final = 10 * math.log10(energia_total)
    return spl_final, distancia_directa
```

Appendix 4 – Module 4: calculate_urban_noise

Module 5 code implementation

```python
# ==========================================================================
# MÓDULO 5: ANALIZADOR DE IMPACTO ACÚSTICO (NOISE ANALYZER)
# ==========================================================================
def noise_analyzer(G_proj, ruido_dron_db, velocidad):
    print("5. Analizando propagación de ruido al nivel del suelo...")
    umbral_evento_db = 55.0  # Nivel a partir del cual el ruido de un evento

    for u, v, key, data in G_proj.edges(keys=True, data=True):
        distancia_calle = data.get('length', 10.0)
        anchura = data.get('anchura_calle', 5.0)
        altura_edificios = data.get('altura_max_edificios', 0.0)
        tiempo_cruce_seg = distancia_calle / velocidad
        data['tiempo_cruce_seg'] = tiempo_cruce_seg

        # El receptor se asume en la acera (a la mitad del ancho de la calle)
        dist_horizontal = anchura / 2

        # Modificamos la población expuesta según el tipo de calle (en zonas industriales la gente no vive ahí)
        personas_calle = data.get('poblacion_base', 0)
        tipo_calle = data.get('highway', '')
        if isinstance(tipo_calle, list): tipo_calle = tipo_calle[0]

        if tipo_calle in ['residential', 'living_street', 'pedestrian']: m = 1.0 # 100% expuesta (viviendas)
        elif tipo_calle in ['commercial', 'retail']: m = 0.5                     # 50% expuesta (comercios)
        elif tipo_calle in ['industrial', 'motorway']: m = 0.0                   # 0% expuesta
        else: m = 0.8

        poblacion_expuesta = int(personas_calle * m)
        data['poblacion_expuesta'] = poblacion_expuesta

        vuelos_exactos = data.get('registro_vuelos_exactos', [])
        total_vuelos_24h = len(vuelos_exactos)

        energia_total_24h = 0.0
        ruido_maximo_suelo = 0.0

        # Calculamos el impacto energético de cada vuelo individual sobre esta calle
        for (h, minuto, segundo, alt_real) in vuelos_exactos:

            if alt_real > altura_edificios:
                # Si el dron supera la altura del edificio, forzamos mínimo 1 rebote
                rebotes = 0
            else:
                # Si vuela por debajo del edificio (dentro del cañón), aplicamos
                rebotes = 3 if (altura_edificios > 5 and anchura < 10) else (1 if altura_edificios > 5 else 0)

            ruido_vuelo_suelo, _ = calculate_urban_noise(ruido_dron_db, alt_real, dist_horizontal, rebotes)
            if ruido_vuelo_suelo > ruido_maximo_suelo:
                ruido_maximo_suelo = ruido_vuelo_suelo

            # La energía acústica total depende de los decibelios y de cuánto tiempo dure el dron cruzando la calle
            energia_cruce = (10 ** (ruido_vuelo_suelo / 10.0)) * tiempo_cruce_seg
            energia_total_24h += energia_cruce
        # Calculamos el indicador L_Aeq_24h (Promedio de energía continuo equivalente en 24 horas)
        if total_vuelos_24h > 0 and energia_total_24h > 0:
            L_Aeq_24h = 10 * math.log10(energia_total_24h / 86400.0) # 86400 son los segundos de un día
            L_DEN = L_Aeq_24h
        else:
            L_Aeq_24h = 0.0
            L_DEN = 0.0

        # Indicadores de impacto normativos:
        # NO_1: Personas que sufren picos de ruido mayores a 55 dB
        data['NO_1'] = poblacion_expuesta if ruido_maximo_suelo > umbral_evento_db else 0
        # NO_2: Personas expuestas a un promedio diario integrado mayor a 40 dB
        data['NO_2'] = poblacion_expuesta if L_DEN > 40.0 else 0

        # Historial horario para optimizaciones detalladas
        historial_ruido = {}
        for hora in range(24):
            ops_hora = sum(1 for (h, m, s, alt) in vuelos_exactos if h == hora)
            historial_ruido[hora] = {
                'operaciones': ops_hora,
                'NO_4': round(ops_hora * data['NO_1'], 2),                  # Impacto por cantidad de eventos
                'NO_5': round(tiempo_cruce_seg * ops_hora * data['NO_1'], 2), # Impacto por tiempo total expuesto
                'tiempo_cruce_seg': round(tiempo_cruce_seg, 2)
            }
        data['historial_ruido_24h'] = historial_ruido

    # Devolvemos el grafo listo convirtiendo las coordenadas de vuelta a Geográficas para pintar mapas
    return ox.project_graph(G_proj, to_latlong=True)
```

Appendix 5 – Module 5: noise_analyzer

Module 6 code implementation

```python
# ==========================================================================
# MÓDULO 6: VISUALIZACIÓN GRÁFICA DEL TRÁFICO
# ==========================================================================
def show_layout(G, planes_generados, hora_a_mostrar=10):
    print(f"\nPreparando el mapa de tráfico para la franja de las {hora_a_mostrar}:00...")

    # Identificamos los nodos que sirven como despegue o aterrizaje en la hora seleccionada
    nodos_origen_destino_globales = set()
    for plan in planes_generados:
        if plan['hora_salida'] == hora_a_mostrar:
            nodos_origen_destino_globales.add(plan['origen'])
            nodos_origen_destino_globales.add(plan['destino'])

    # Configuración visual de los nodos del mapa
    colores_nodos = []
    tamanos_nodos = []
    for nodo in G.nodes():
        if nodo in nodos_origen_destino_globales:
            colores_nodos.append('#2C3E50') # Color oscuro para bases de drones
            tamanos_nodos.append(40)
        else:
            colores_nodos.append('none')    # Nodos de paso invisibles para no saturar la vista
            tamanos_nodos.append(0)

    planes_hora = [p for p in planes_generados if p['hora_salida'] == hora_a_mostrar]
    colores_trafico = []
    grosores_trafico = []

    # Calculamos cuántas operaciones cruzan cada calle a la hora elegida para asignarle un color
    for u, v, key, data in G.edges(keys=True, data=True):
        ops_calle = 0
        for plan in planes_hora:
            for tramo in plan['tramos']:
                if (tramo['u'] == u and tramo['v'] == v) or (tramo['u'] == v and tramo['v'] == u):
                    ops_calle += 1

        # Semáforo de tráfico aéreo urbano
        if ops_calle == 0:
            colores_trafico.append('#E0E0E0')   # Gris: Sin tráfico
            grosores_trafico.append(1.0)
        elif ops_calle <= 2:
            colores_trafico.append('#2ECC71')   # Verde: Tráfico bajo
            grosores_trafico.append(2.5)
        elif ops_calle <= 5:
            colores_trafico.append('#F1C40F')   # Amarillo: Tráfico medio
            grosores_trafico.append(3.5)
        else:
            colores_trafico.append('#E74C3C')   # Rojo: Tráfico denso / Saturación
            grosores_trafico.append(5.0)

    # Dibujamos el grafo con la librería OSMnx
    fig, ax = ox.plot_graph(G, show=False, close=False,
                            node_size=tamanos_nodos, node_color=colores_nodos, node_zorder=5, bgcolor='white',
                            edge_color=colores_trafico, edge_linewidth=grosores_trafico)

    ax.set_title(f" Time: {hora_a_mostrar}:00", fontsize=14, fontweight='bold', color='white')
    ax.text(0.02, 0.02, 'Dark point: Starting and Ending \nStreet-> Grey: 0 | Green: 1-2 ops | Yellow: 3-5 ops | Red: >5 ops',
            transform=ax.transAxes, fontsize=9, color='black', bbox=dict(facecolor='white', alpha=0.8, edgecolor='black'))
    plt.show()
```

Appendix 6 – Module 6: show_layout

Module 7 code implementation (1/3)

```python
# ==========================================================================
# MÓDULO 7: EXPORTACIÓN DE RESULTADOS DE SIMULACIÓN A EXCEL
# ==========================================================================
def export_data(G, planes_generados, ruido_base_dron_dB, velocidad, nombre_archivo="datos_operaciones_cplex_OO.xlsx", hora_objetivo=14):
    print("\n7. Evaluando colisiones y Exportando datos a Excel...")

    alturas_dron = [7.5, 10.0, 12.5, 15.0] # Las 4 capas de vuelo disponibles en el espacio aéreo
    datos_por_altura = {}

    # --- HOJAS 1 A 4: SIMULACIÓN DE ESCENARIOS EN ALTURAS FIJAS ---
    for h_base in alturas_dron:
        filas_altura = []
        ocupacion_radar = {} # Diccionario para detectar si dos drones chocan en espacio y tiempo

        for plan in planes_generados:
            id_vuelo = plan['id_global']
            for tramo in plan['tramos']:
                u, v = tramo['u'], tramo['v']
                h_hora, m, s = tramo['hora'], tramo['minuto'], tramo['segundo']

                # Definimos una celda espacio-temporal única en el radar
                clave_radar = (u, v, h_hora, m, s)
                if clave_radar not in ocupacion_radar:
                    ocupacion_radar[clave_radar] = []

                alt_real = h_base
                # MECANISMO ANTI-COLISIÓN DE EMERGENCIA:
                while alt_real in ocupacion_radar[clave_radar]:
                    alt_real += 2.5

                # Registramos que esa altura ya queda reservada en el radar
                ocupacion_radar[clave_radar].append(alt_real)

                data_arista = G[u][v][0]
                aresta_id = data_arista.get('osmid', f"{u}_{v}")
                if isinstance(aresta_id, list): aresta_id = aresta_id[0]
                anchura = data_arista.get('anchura_calle', 5.0)
                altura_ed = data_arista.get('altura_max_edificios', 0.0)
                if alt_real > altura_ed:
                    rebotes = 0
                else:
                    rebotes = 3 if (altura_ed > 5 and anchura < 10) else (1 if altura_ed > 5 else 0)

                # Volvemos a calcular el ruido real que llega al suelo considerando el ajuste de altura anti-colisión
                ruido_suelo, _ = calculate_urban_noise(ruido_base_dron_dB, alt_real, 1.0, rebotes)

                # --- NUEVA INCLUSIÓN: VARIABLES DE IMPACTO ACÚSTICO NO ---
                poblacion = data_arista.get('poblacion_base', 0)
                t_cruce = data_arista.get('length', 10.0) / velocidad

                no1_val = poblacion if ruido_suelo > 65.0 else 0

                # Cálculo de la penalización de la Normativa Europea según la hora real del tramo (L_DEN)
                energia_hora = (10 ** (ruido_suelo / 10.0)) * t_cruce
                term_day = 0.0
                term_evening = 0.0
                term_night = 0.0
```

Appendix 7 – Module 7: export_data (1/3)

Module 7 code implementation (2/3)

```
            if 6 <= h_hora < 18:
                L_D = 10 * math.log10(energia_hora / (12 * 3600)) if energia_hora > 0 else 0.0
                term_day = (12 / 24) * (10 ** (L_D / 10.0))
            elif 18 <= h_hora < 22:
                L_E = 10 * math.log10(energia_hora / (4 * 3600)) if energia_hora > 0 else 0.0
                term_evening = (4 / 24) * (10 ** ((L_E + 5) / 10.0))
            else:
                L_N = 10 * math.log10(energia_hora / (8 * 3600)) if energia_hora > 0 else 0.0
                term_night = (8 / 24) * (10 ** ((L_N + 10) / 10.0))

            suma_lineal = term_day + term_evening + term_night
            lden_aprox = 10 * math.log10(suma_lineal) if suma_lineal > 0 else 0.0
            no2_val = poblacion if lden_aprox > 40.0 else 0

            sel_a_max = ruido_suelo + 10 * math.log10(t_cruce) if t_cruce > 0 else 0
            no3_val = poblacion if sel_a_max > 65.0 else 0

            ops_hora = 1 # Cada tramo representa un único evento individual
            no4_val = ops_hora * no1_val
            no5_val = t_cruce * ops_hora * no1_val

            filas_altura.append({
                'Flight_Plan': id_vuelo,
                'Nodo_Origen': u,
                'Nodo_Destino': v,
                'ID_Aresta': aresta_id,
                'Hora': f"{h_hora:02d}:{m:02d}:{s:02d}",
                'Altitud Real (m)': alt_real,
                'Ruido en Receptor (dB)': round(ruido_suelo, 2),
                'NO1 (person)': no1_val,
                'NO2 (person)': no2_val,
                'NO3 (person)': no3_val,
                'NO4 (N.person)': round(no4_val, 2),
                'NO5 (D.person)': round(no5_val, 2)
            })

        datos_por_altura[f"Escenario_{h_base}m"] = pd.DataFrame(filas_altura)

    # --- HOJA 5: COMBINACIONES EN ESTRICTO ORDEN BINARIO (SÓLO RUIDOS Y NIVELES) ---
    todas_las_calles = list(G.edges(keys=True, data=True))
    todas_las_calles.sort(key=lambda x: str(x[3].get('osmid', '')))

    calles_seleccionadas = []
    osmids_vistos = set()

    for calle in todas_las_calles:
        u, v, key, data = calle
        aresta_id = data.get('osmid', f"{u}_{v}")
        if isinstance(aresta_id, list): aresta_id = aresta_id[0]
        aresta_id = str(aresta_id)

        if aresta_id not in osmids_vistos:
            osmids_vistos.add(aresta_id)
            calles_seleccionadas.append(calle)

        if len(calles_seleccionadas) == 2:
            break

    filas_combinaciones = []
    hora = hora_objetivo
```

Appendix 8 – Module 7: export_data (2/3)

Module 7 code implementation (3/3)

```
    for index_calle, (u, v, key, data) in enumerate(calles_seleccionadas):
        aresta_id = data.get('osmid', f"{u}_{v}")
        if isinstance(aresta_id, list): aresta_id = aresta_id[0]

        anchura = data.get('anchura_calle', 5.0)
        altura_ed = data.get('altura_max_edificios', 0.0)

        if index_calle > 0:
            filas_combinaciones.append({})

        for i in range(1, 16):
            nivel_1 = 1 if (i & 1) else 0
            nivel_2 = 1 if (i & 2) else 0
            nivel_3 = 1 if (i & 4) else 0
            nivel_4 = 1 if (i & 8) else 0

            combo = []
            if nivel_1: combo.append(7.5)
            if nivel_2: combo.append(10.0)
            if nivel_3: combo.append(12.5)
            if nivel_4: combo.append(15.0)

            ops_hora = len(combo)

            ruidos_individuales = []
            for h in combo:
                if h > altura_ed:
                    rebotes_dinamicos = 0
                else:
                    rebotes_dinamicos = 3 if (altura_ed > 5 and anchura < 10) else (1 if altura_ed > 5 else 0)

                ruido_ind, _ = calculate_urban_noise(ruido_base_dron_dB, h, 1.0, rebotes_dinamicos)
                ruidos_individuales.append(ruido_ind)

            energia_sumada = sum(10 ** (r / 10.0) for r in ruidos_individuales)
            ruido_suelo_combinado = 10 * math.log10(energia_sumada) if energia_sumada > 0 else 0

            # Estructura limpia sin métricas NO en la matriz combinatoria
            filas_combinaciones.append({
                'Hora': f"{hora:02d}:00",
                'ID_Aresta': aresta_id,
                'Ops/simultaneas': ops_hora,
                'Nivel_4 (15.0m)': nivel_4,
                'Nivel_3 (12.5m)': nivel_3,
                'Nivel_2 (10.0m)': nivel_2,
                'Nivel_1 (7.5m)': nivel_1,
                'Ruido en receptor (dB)': round(ruido_suelo_combinado, 2)
            })

    df_combinaciones = pd.DataFrame(filas_combinaciones)

    # --- GUARDAR EXCEL FINAL ---
    with pd.ExcelWriter(nombre_archivo, engine='openpyxl') as writer:
        for nombre_hoja, df_hoja in datos_por_altura.items():
            df_hoja.to_excel(writer, sheet_name=nombre_hoja, index=False)
        df_combinaciones.to_excel(writer, sheet_name='Combinaciones', index=False, na_rep='')

    print(f"¡Hecho! Archivo '{nombre_archivo}' generado correctamente.")
```

Appendix 9 – Module 7: export_data (3/3)

Main code implementation

```
# ==========================================================================
# ZONA DE EJECUCIÓN (PRINCIPAL)
# ==========================================================================
    # CONFIGURACIÓN DE PARÁMETROS DE LA SIMULACIÓN (Ejemplo centrado en Barcelona)
coordenada = (41.388907, 2.156873)
radio_metros = 1000            # Radio de acción del mapa urbano
traffic_density = 150          # Densidad base de drones diarios a simular
ruido_base_dron_dB = 80.0      # Ruido del dron a 1 metro de distancia (Lp)
velocidad_dron_ms = 16.0       # Velocidad constante del dron (aprox. 57 km/h)

# Paso 1: Descargar mapas viales y analizar la geometría de las calles
grafo_proyectado, edificios_proyectados = get_map_buildings(coordenada, radio_metros)
grafo_proyectado = analyse_urban_environment(grafo_proyectado, edificios_proyectados)

# Paso 2: Ejecutar el algoritmo de enrutamiento y planes de vuelos diarios
grafo_proyectado, planes_generados = traffic_generator(grafo_proyectado, traffic_density, velocidad_dron_ms)

# Paso 3: Evaluar los niveles de ruido acumulados e indicadores NO en toda la cuadrícula
grafo_proyectado = noise_analyzer(grafo_proyectado, ruido_base_dron_dB, velocidad_dron_ms)

# Paso 4: Exportar los informes formateados y listos para los solvers matemáticos en Excel
export_data(grafo_proyectado, planes_generados, ruido_base_dron_dB, velocidad_dron_ms, "datos_operaciones_cplex_OO.xlsx", hora_objetivo=14

# Paso 5: Renderizar en pantalla el mapa de congestión a la hora deseada
if planes_generados:
    show_layout(grafo_proyectado, planes_generados, hora_a_mostrar=6)
```

Appendix 10 – Main execution