uc3m | Universidad Carlos III de Madrid

University Degree in Aerospace Engineering
Academic year 2023-2024

*Bachelor Thesis*

# Autonomous Navigation of Lightweight UAVs in GPS-denied Environments

Iago Senén Fernández García

Supervised by:
Javier García-Heras Carretero

September, 2024
Leganés (Madrid)

# Abstract

This work addresses the challenge of GPS-denied navigation for Unmanned Aerial Vehicles ($UAV$s), an emerging critical issue as reliance on GPS becomes increasingly impractical in various operational environments.

The study begins with an in-depth exploration of the GPS-denied navigation problem, highlighting fundamental challenges and existing solutions. A comprehensive review of current methodologies is provided, including visual odometry, inertial navigation or Simultaneous Localization and Mapping ($SLAM$), along with their applications and limitations.

Building on the State-of-the-Art analysis, this thesis identifies a notable gap in research concerning lightweight fixed-wing $UAV$s. To address this, a novel hardware and software architecture is proposed, designed to facilitate further investigation and comparison of alternative navigation systems. The proposed system architecture integrates multiple sensors and computational modules, with a focus on modularity and adaptability to diverse $UAV$ platforms.

The proposed solution includes detailed descriptions of the individual components, their interconnections, and the overall integration process, providing a framework for the implementation of alternative navigation algorithms. By offering a comprehensive blueprint for future research, this thesis aims to contribute in the field of GPS-denied navigation and improve the performance and reliability of $UAV$s in environments where GPS is compromised or unavailable.

**Keywords:** $UAV$, Autonomous navigation, GPS-denied environment, Artificial Intelligence, Computer vision.

# Dedication

I would like to dedicate this work to my family.

Throughout my life, I have always been encouraged and supported to pursue my dreams and passions. Being able to do so, first as a classical pianist and now as an aerospace engineer, has been possible largely thanks to both the financial and emotional support of my mother, who has always been there for me, regardless of the hurdles that I encountered.

> *" Far better is it to **dare mighty things**, to win glorious triumphs, even though checkered by failure ...than to rank with those poor spirits who neither enjoy nor suffer much, because they live in a gray twilight that knows not victory nor defeat. "*
>
> – Theodore Roosevelt, *Strenuous Life*, 1899

# Contents

# List of Figures

# List of Tables

# Acronyms

$ADC$ Air Data Computer.

$ADS$ Air Data System.

$ADS-B$ Automatic Dependent Surveillance-Broadcast.

$AESA$ Spanish Aviation Safety and Security Agency.

$AHRS$ Attitude and Heading Reference System.

$AI$ Artificial Intelligence.

$ANN$ Artificial Neural Network.

$ANS$ Alternative Navigation System.

$ATC$ Air Traffic Controller.

$ATTOL$ Autonomous Taxi, Take Off and Landing.

$CAD$ Computer Aided Design.

$CAM$ Computer Aided Manufacturing.

$CAN$ Controller Area Network.

$CFIT$ Controlled Flight Into Terrain.

$CIA$ Central Intelligence Agency.

$CIDA$ Consistent INS Drift Algorithm.

$CME$ Coronal Mass Ejections.

$CNN$ Convolutional Neural Network.

$COTS$ Commercial Off-The-Shelf.

$DEM$ Digital Elevation Map.

$DGPS$ Differential Global Positioning System.

$DME$ Distance Measuring Equipment.

*DOT* United States Department of Transportation.

*DSO* Direct Sparse Odometry.

*DTAM* Dense Tracking and Mapping.

*EASA* European Union Aviation Safety Agency.

*EGPWS* Enhanced Ground Proximity Warning Systems.

*EKF* Extended Kalman Filter.

*ESC* Electronic Speed Controller.

*ETSI* European Telecommunications Standards Institute.

*FAA* Federal Aviation Administration.

*FAST* Features from Accelerated Segment Test.

*FCAS* Future Combat Air System.

*FMS* Flight Management System.

*FOG* Fiber Optic Gyro.

*FOV* Field of View.

*FPGA* Field Programmable Gate Array.

*FPV* First-Person View.

*FREAK* Fast Retina Keypoint.

*GAO* United States Government Accountability Office.

*GBAS* Ground Based Augmentation Systems.

*GCS* Ground Control Station.

*GNSS* Global Navigation Satellite System.

*GPS* Global Positioning System.

*GSOC* Google Summer of Code.

*HITL* Hardware In The Loop.

*HOG* Histogram of Oriented Gradients.

*HTP* Horizontal Tail Plane.

*IATA* International Air Transport Association.

$ICBM$ Intercontinental Ballistic Missile.

$ILS$ Instrument Landing System.

$IMU$ Inertial Measurement Unit.

$INCOSE$ International Council on Systems Engineering.

$INS$ Inertial Navigation System.

$IRS$ Inertial Reference System.

$KF$ Kalman Filter.

$LEO$ Low Earth Orbit.

$LKF$ Lucas-Kanade Optical Flow.

$LRF$ Laser Range Finder.

$LSD-SLAM$ Large-Scale Direct SLAM.

$LiDAR$ Light Detection and Ranging.

$MAC$ Mid Air Collisions.

$MEMS$ Micro-Electro Mechanical Systems.

$MFD$ Multi-Function Display.

$ML$ Machine Learning.

$MLEAP$ Machine Learning Application Approval.

$MTOW$ Maximum Take-Off Weight.

$NASA$ National Aeronautics and Space Administration.

$NATO$ North Atlantic Treaty Organization.

$NAVAIDs$ Navigational Aids.

$ND$ Navigation Display.

$NDB$ Non-Directional Beacon.

$OF$ Optical Flow.

$ORB-SLAM$ Oriented FAST and Rotated BRIEF SLAM.

$PNT$ Positioning, Navigation, and Timing.

$PRF$ Pulse-Repetition Frequency.

$PTAM$ Parallel Tracking and Mapping.

$PWM$ Pulse Width Modulation.

$RANSAC$ Random Sample Consensus.

$RCNN$ Region-based Convolutional Neural Network.

$RE$ Runway Excursions.

$RFI$ Radio Frequency Interference.

$RMSE$ Root Mean Square Error.

$RNAV$ Area Navigation.

$ROS$ Robot Operating System.

$RSO$ Resident Space Object.

$RTK$ Real-Time Kinematics.

$RaDAR$ Radio Detection and Ranging.

$SAR$ Synthetic Aperture Radar.

$SES$ Single European Sky.

$SIFT$ Scale-Invariant Feature Transform.

$SITL$ Software In The Loop.

$SLAM$ Simultaneous Localization and Mapping.

$SNR$ Signal-to-Noise Ratio.

$SOTA$ State of the Art.

$STS$ Standard Training Scenarios.

$SURF$ Speeded-Up Robust Features.

$SoOP$ Signals of Opportunity.

$TAWS$ Terrain Avoidance Warning System.

$TDOA$ Time Difference of Arrival.

$TPS$ Theatre Positioning System.

$TRN$ Terrain Reference Navigation.

$UART$ Universal Asynchronous Receiver/Transmitter.

$UAS$ Unmanned Aircraft Systems.

$UAS-UTM$ Unmanned Aerial System Traffic Management.

$UAV$ Unmanned Aerial Vehicle.

$UBEC$ Universal Battery Elimination Circuit.

$UGV$ Unmanned Ground Vehicle.

$USDOD$ United States Department of Defense.

$UWB$ Ultra Wide Band.

$VIO$ Visual-Inertial Odometry.

$VLAD$ Vector of Locally Aggregated Descriptors.

$VLOS$ Visual Line of Sight.

$VNS$ Visual Navigation System.

$VO$ Visual Odometry.

$VOR$ VHF Omnidirectional Range.

$VTOL$ Vertical Take-off and Landing.

$VTP$ Vertical Tail Plane.

$XCS$ Xtool Creative Space.

$vSLAM$ Visual Simultaneous Localization and Mapping.

# 1 Introduction to this work

## 1.1 Motivation

The rapid expansion of the Unmanned Aircraft Systems ($UAS$) industry over the past decade has revolutionized both the military and civilian sectors, including transportation, logistics, aerial delivery, agriculture, environmental monitoring, emergency response, and infrastructure inspection.

The global market for transport and logistics supported by $UAS$ is projected to grow from USD 11 billion in 2022 to USD 29 billion by 2027 [1]. Additionally, $UAS$ production is expected to increase from 2 million units in 2021 to 6.5 million units by 2030. As highlighted by the Federal Aviation Administration ($FAA$) [2], $UAS$ operations are expanding at an unprecedented rate, driven by technological advances and the growing demand for autonomous aerial solutions. Similarly, the National Aeronautics and Space Administration ($NASA$)'s 2019 study on UAS demand and economic benefits [3] emphasizes that the commercial and civil drone market in the U.S. is positioned for significant growth, with more missions being conducted daily, leading to a corresponding increase in revenue generation. This growth assumes the continued development of supportive policies, regulations, and enabling technologies.

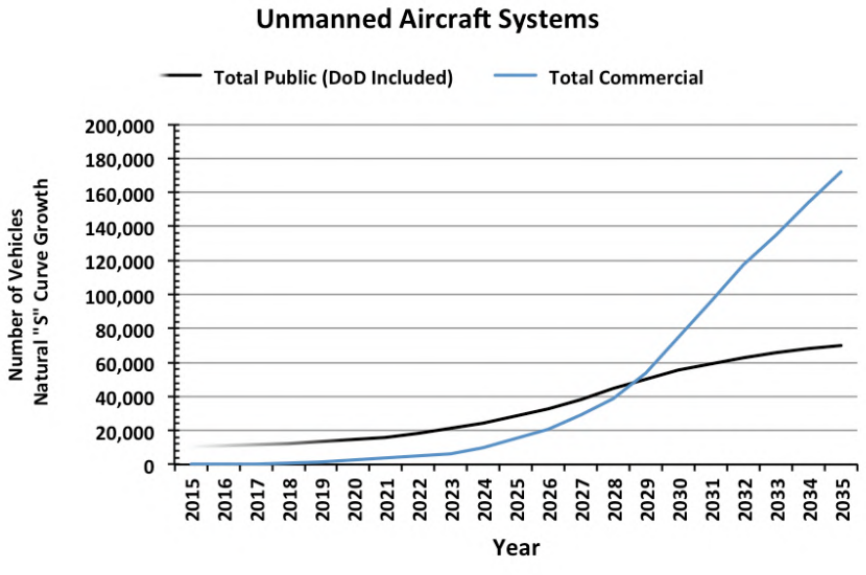



**Fig. 1.1:** Projected annual growth for of the $UAS$ market in the U.S. from 2015-2035, as extracted from the U.S. Department of Transportation [4].

As the use of $UAS$ proliferates, the challenge of managing this growing fleet becomes increasingly critical, particularly within the framework of Unmanned Aerial System Traffic Management ($UAS - UTM$). The vast majority of these drones are small [1], typically with an Maximum Take-Off Weight ($MTOW$) of less than 25 kg, and rely heavily on the Global Positioning System ($GPS$) for navigation [5]. However, the reliance on $GPS$ presents a significant vulnerability. $GPS$ signals are fragile and susceptible to interference, especially in environments like dense urban areas with tall buildings, indoor spaces, polar regions, or in scenarios involving electronic warfare, where jamming and spoofing attacks are common. [6, 7].

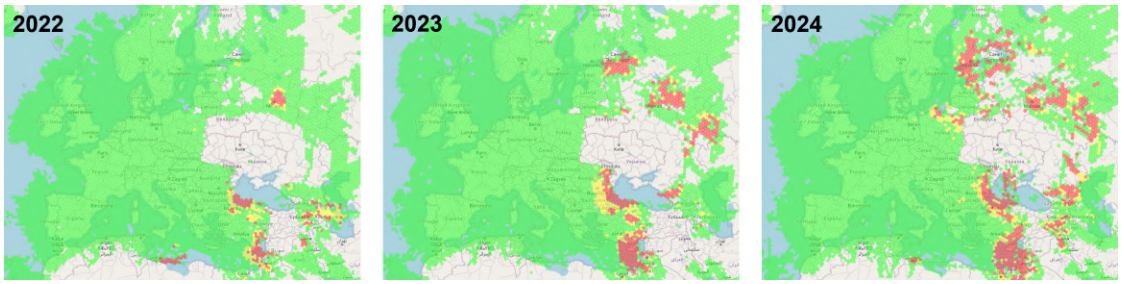



**Fig. 1.2:** Evolution of $GPS$-compromised zones in Europe from August 2022-2024, extracted from the ADS-B Exchange, after analyzing the integrity and accuracy of $GPS$ signals broadcast by the Automatic Dependent Surveillance-Broadcast ($ADS - B$) packages from civilian aircraft.

The potential consequences of $GPS$ signal disruption in such scenarios are alarming. Considering a future where thousands of drones operate simultaneously, the loss of $GPS$ could lead to widespread operational failures, as noted by $NASA$ [5], which has pointed out that drones without $GPS$ would drift, making it difficult to maintain control. Moreover, the $FAA$ has acknowledged the growing threat of $GPS$ jamming and spoofing, issuing safety alerts to operators with guidelines on handling such incidents [8]. The European Union Aviation Safety Agency ($EASA$) has also reported disruptions in aviation and logistics operations near the Baltic region due to $GPS$ interference linked to the ongoing conflict between Russia and Ukraine [6].

Given these challenges, there is an urgent need to develop **alternative navigation techniques** to ensure the safe and reliable operation of drones in $GPS$-denied environments. Various alternatives have been explored, as discussed by Miller *et al.* in a North Atlantic Treaty Organization ($NATO$) report outlining potential solutions when $GPS$ signals are compromised [9].

Simultaneously, recent advances in Artificial Intelligence ($AI$) and Machine Learning ($ML$), particularly in Deep Learning and Computer Vision, offer promising approaches for addressing this issue. Modern AI-optimized hardware has evolved to support complex $AI$ tasks, enabling the processing of data from multiple high-resolution sensors in real-time, which could be crucial in developing robust vision-based navigation systems for drones that can operate autonomously in the absence of $GPS$.

## 1.2 Statement of the problem

The primary problem this work addresses is developing effective navigation strategies for $UAV$s in scenarios where $GPS$ signals are unavailable or compromised. Although this situation affects all types of vehicles, it becomes critical for small $UAV$ that do not have a backup navigation solution in the case of $GPS$ failure. With increasing numbers of $UAV$s operating across different industries, the risk grows rapidly when considering thousands of these drones affected by this potential problem.

Existing approaches often involve complex sensor fusion techniques, sophisticated algorithms, or require heavy computational resources, which may not be suitable for lightweight $UAV$s with limited payload capacity and processing power.

At the same time, commercially available solutions are often military-grade or proprietary hardware, not accessible to university students or researchers. Therefore, the proposed solution to the problem should be understandable, replicable, and allow for future improvements and revisions.

As the problem is broad and complex, it needs to be narrowed down in order to limit its scope. This work will focus as follows:

- The type of operations that the aircraft will engage in will always be within Visual Line of Sight ($VLOS$) and below 120 meters in height.
- The aircraft involved will not exceed 25 kg of $MTOW$ nor exceed 3 meters of wingspan, thus falling under the "Open" category of $EASA$ regulations for $UAS$, as will be explained in section 2.3.
- Onboard selected hardware and other electrical components shall be restricted to those that are available as Commercial Off-The-Shelf ($COTS$).
- The considered $GPS$-denied environments will not include indoor, underground, or space scenarios. It will only apply to aerial navigation where the signal has been compromised due to jamming/spoofing or is not available due to physical obstructions (e.g., mountains, buildings).
- The weather will be characterized by unlimited visibility, no significant clouds, and temperature close to the ISA standard of 15ºC.
- This research will not address the issue by exploring new Global Navigation Satellite System ($GNSS$) signaling methods but will instead focus on alternative navigation techniques.
- While the V-model is employed as a requirements-based framework, this thesis will not include all the typical outputs (e.g., Concept of Operations document); instead, it adapts the model to guide the methodology presented in Section 5.

## 1.3 Objectives

The main goal of this thesis is to propose a suitable and functional platform for testing and evaluating alternative autonomous navigation solutions for $UAV$s in $GPS$-denied environments.

To accomplish this goal, the proposed solution must comply with the following requirements:

1. The design must be based on open-source $COTS$ components.
2. The solution must be reproducible, scalable, and upgradable.
3. The platform must be capable of operating in real-world flight conditions, demonstrating its ability to execute stable flight.
4. The solution must enable the possibility of testing autonomous flight.
5. It must consider the state of the art in current advances in the field of Alternative Navigation System ($ANS$).
6. It must take into account urban mobility market opportunities (e.g., long-distance delivery).
7. It must support a modular architecture, allowing for easy integration and testing of sensors and algorithms.
8. It must provide a way to compare the tested algorithms to a known ground truth.
9. It must be able to process data pipelines with heavy visual data.

## 1.4 Document Structure

This work is structured into 8 chapters:

1. **Introduction**: Provides the motivation behind this work, enunciates the problem statement, outlines the objectives to be achieved, and presents a brief comment of the methodology employed in the thesis.

2. **Regulatory Framework**: Discusses relevant institutions, applicable legislation, and operational categories related to the type of $UAV$s that this work focuses on.

3. **Theoretical Background**: Covers fundamental concepts that the reader might find useful before proceeding with Chapter 4. These include a brief description of the type of $UAV$s to which the thesis applies, an overview of the navigation techniques available for aerial vehicles, and the nature of the problem at hand: the $GPS$-denied environment and how autonomous navigation is achieved with an autopilot. It also introduces the topic of $AI$, particularly deep learning, and its potential applications in the aerospace industry.

4. **State of the Art**: Reviews the origins of the $GPS$-denied problem, current advancements in autonomous navigation for $UAV$s, and provides an exhaustive literature review of different approaches in alternative navigation systems.

5. **Methodology**: Outlines the steps taken to achieve the thesis objectives using the V-model, from identifying solution and system requirements to building a deployable solution. It includes design and manufacturing as well as verification and validation processes for the proposed architecture.

6. **Results**: Presents the onboard systems enclosure box, which contains all the hardware needed to test alternative navigation solutions. It also includes the results from the flight tests, demonstrating that the proposed platform is capable of flight.

7. **Socio-Economical Environment**: Analyzes the socio-economic impact, environmental impact, budget breakdown, and offers a word of caution for further research in this field, particularly considering the potential for controversial military uses.

8. **Conclusions**: Summarizes the work done and suggests directions for future research.

## 1.5 Methodology Approach

The methodology adopted in this work is based on the **V-model** as outlined by the International Council on Systems Engineering ($INCOSE$) standards for project development. This approach provides a rigorous framework for ensuring quality and consistency throughout the development process, facilitating clear validation and verification of initial requirements at each stage.

The methodology is divided into seven parts:

1. **Identification of Solution Requirements**: Before taking any actions, it is crucial to understand what can be achieved and the rationale behind it. To this end, the author has focused on deriving valuable insights from the reviewed literature in the State of the Art ($SOTA$) and identifying critical scenarios. Subsequently, and in conjunction with the thesis objectives and the applicable regulatory framework, a list of solution requirements is created. This list will later serve as a benchmark for validating the proposed solution.

2. **Identification of System Requirements**: Technical requirements are formulated to meet the previously established solution requirements. This includes a high-level overview of the proposed solution's components, the rationale for their selection, and their interconnections.

3. **Identification of Component Requirements**: Building on the high-level architecture of the solution, a more detailed approach is outlined for each component, considering their specific power and data transmission needs. This results in a detailed architecture of the solution.

4. **Manufacturing and Implementation**: The proposed solution is manufactured using available tools, while simultaneously integrating the necessary electrical components.

5. **Component Verification**: Each component's functionality is verified in standalone mode, with detailed information provided on the verification process.

6. **Integration Testing and Flight Testing**: The methodology for conducting flight tests and post-flight analysis is described. System integration is performed by checking communication between module pairs to ensure that data can travel freely and be used effectively.

7. **Deployment and Preparation for Future Upgrades**: Validation of the initial requirements is conducted to ensure that all solution requirements have been addressed. This step also includes preparations for potential future upgrades.

# 2 Regulatory Framework

## 2.1 Relevant Institutions

### 2.1.1 European Union Aviation Safety Agency (EASA)

The European Union Aviation Safety Agency ($EASA$) is the authority responsible for harmonizing aviation safety standards across all EU member states. $EASA$'s primary objective is to ensure a consistent and high level of safety in civil aviation operations throughout the European Union. This is achieved through the establishment and enforcement of common regulations applicable to all member states. In particular, for the standardization of $UAS$, $EASA$ has implemented Regulations **(EU) 2019/947** and **(EU) 2019/945**.

### 2.1.2 Spanish Aviation Safety and Security Agency (AESA)

The Spanish Aviation Safety and Security Agency ($AESA$) serves as the national regulatory body in Spain and is tasked with overseeing adherence to civil aviation standards within the country's aerospace sector. $AESA$ plays a critical role in promoting the development and application of aviation legislation, ensuring that the Spanish civil aviation system upholds the highest standards of safety, quality, and sustainability. In cases of non-compliance with aviation regulations within Spain, $AESA$ has the authority to enforce sanctions.

## 2.2 Applicable Legislation

### 2.2.1 Implementing Regulation (EU) 2019/947

The Implementing Regulation (EU) 2019/947 is an European Union regulation that establishes the operational rules and requirements for ($UAS$). It provides a legal framework for the use of $UAS$ across different categories of operations.

The regulation outlines the operational requirements and procedures for $UAS$ operators, including the need for operational authorizations and risk assessments when applicable. It sets standards for remote pilot competency, operational procedures, and safety management necessary to conduct $UAS$ flights safely and effectively.

Additionally, the Implementing Regulation (EU) 2019/947 integrates with the Delegated Regulation (EU) 2019/945 by defining operational requirements in relation to the $UAS$ classes established within. It details the specific operational limitations and conditions for each class of $UAS$, including requirements for the handling of $UAS$ in classes C0 through C4, and includes provisions for the safe integration of new $UAS$ classes introduced by the amendment in Delegated Regulation (EU) 2020/1058, classes C5 and C6.

This regulation also addresses the procedures for $UAS$ operators from third countries (non-$EASA$ member states) seeking to operate within the Single European Sky ($SES$) airspace, ensuring that their operations align with EU standards and safety regulations.

### 2.2.2 Delegated Regulation (EU) 2019/945

The Delegated Regulation (EU) 2019/945 is an European Union regulation that sets the rules and standards for $UAS$. It defines the types of $UAS$ that require certification in terms of design, production, and maintenance. The regulation also establishes guidelines for the commercialization of $UAS$ intended for use in the Open category, as well as for remote identification accessories (e.g. Drone Remote ID) .

It also outlines the requirements for the design and manufacture of $UAS$ intended for use under the conditions defined in the Implementing Regulation (EU) 2019/947.

### 2.2.3 Royal Decree UAS 517/2024

Its purpose is to establish the legal framework for those issues where the Implementing and Delegated Regulations from $EASA$ either grant member states (e.g., Spain) the authority to regulate or do not directly address these aspects.

### 2.2.4 Regulation (EU) 2024/1689: Artificial Intelligence Act

The Artificial Intelligence Act ($AI$ Act) of the European Union entered into force on the 1st of August 2024 [10] and aims to ensure $AI$ systems are safe, transparent, and ethical while fostering innovation and protecting fundamental rights [11]. It categorizes $AI$ systems by risk, imposing strict requirements on high-risk applications, such as those in aviation, which may impact public safety and fundamental rights. These requirements include robust risk management, transparency, human oversight, and data governance, ensuring $AI$ systems are reliable and secure.

The $AI$ Act introduces significant compliance obligations that could increase development costs and time. High-risk systems must meet strong standards to access the EU market, which may challenge innovation but ultimately aims to build trust and facilitate the broader adoption of $AI$ technologies within the EU.

## 2.3 Operational Categories

Regarding $UAS$, the Implementing Regulation (EU) 2019/947 defines three distinct categories [12]:

- **Open Category**: This is the least restrictive category and is designed for low-risk operations. It includes activities such as recreational flying and commercial operations that pose minimal risk to people and property. Operators must follow specific operational limitations (e.g., flying below 120 meters, maintaining $VLOS$). $UAS$ must be under 25 kg, and the pilot must ensure the drone does not fly over people or in restricted areas. No prior authorization is required, but registration and training as a remote pilot are compulsory for all operations, with the exception of drones weighing <250 g that do not fit a camera/sensor.

- **Specific Category**: This category covers medium-risk operations where a more detailed assessment is needed. It includes operations that might involve flying over people or in restricted areas but with mitigation procedures in place. Operators must conduct a risk assessment and obtain an operational authorization called Standard Training Scenarios ($STS$) from $AESA$. The requirements for $UAS$ and pilot qualifications can vary based on the specific risk assessment and operational procedures defined in the risk assessment.

- **Certified Category**: This category is designed for high-risk operations that are more complex and involve significant risk to people and property, such as those similar to manned aviation operations. It involves stringent requirements similar to those for manned aviation. $UAS$ must meet specific certification standards, and operators need to comply with strict safety regulations. It often includes requirements for advanced training and operational procedures, similar to those for commercial air transport.

### 2.3.1 The Open Category

This work will focus on civil $UAS$ that fall under $EASA$'s Open Category, although some of the work done may be applicable to other categories with the proper regulatory adjustments.

Within the Open Category, there are three subcategories that differentiate themselves based on the associated risk, aircraft weight, and operational limits:

1. **A1**: $UAS$ with an $MTOW$ of less than 250 g that can fly over people but not over assemblies of people.

2. **A2**: $UAS$ with an $MTOW$ of less than 4 kg that can fly close to people, but must maintain a horizontal distance of 30 meters (5 meters in low-speed configuration).

3. **A3**: $UAS$ with an $MTOW$ of less than 25 kg and a wingspan of 3 meters that can fly far from people, always maintaining a safe horizontal distance of 150 meters from nearby elements.

| Operation | | | Drone Operator / pilot | | | |
|---|---|---|---|---|---|---|
| C-Class | Max Take off mass | Subcategory | Operational restrictions | Drone Operator registration? | Remote pilot qualifications | Remote pilot minimum age |
| Privately build<br>legacy < 250g<br>C0 | <250g  | **A1**<br>Not over assemblies of people<br>(can also fly in subcategory A3) | Operational restrictions on the drone's use apply<br>(follow the QR code below) | **Yes**<br>No if toy or not fitted with camera/sensor  | Read user's manual | No minimum age<br>(certain conditions apply) |
| C1 | <900g  | | | Yes | Check out the QR code below for the necessary qualifications to fly these drones | 16 |
| C2 | <4kg  | **A2**<br>Fly close to people<br>(can also fly in subcategory A3) | | | | |
| C3<br>C4<br>Privately build<br>Legacy drones (art 20) | <25kg  | **A3**<br>Fly far from people | | | | |

**Fig. 2.1:** Applicable regulation chart for $UAS$ inside the Open Category, as extracted from $EASA$ [13].

Some additional rules that apply to all three subcategories are stated below:

- The height of 120 meters above ground level should not be exceeded, as the lower limit for general aviation is 150 meters. Therefore, there is only a 30-meter separation between manned aviation and $UAS$.
- The operator must always fly in $VLOS$, unless the aircraft is in "follow me" mode or the pilot is using First-Person View ($FPV$) goggles.
- The operator must be registered if the $UAS$ weighs more than 250 g or if the aircraft is equipped with a camera or sensor.
- The aircraft must have a remote identification ID [14], which comes by default in all C1-C6 categories with the exception of C4 and privately built aircraft.

# 3 Theoretical Background

This chapter will introduce several topics to establish the basic theoretical foundation for the subsequent chapters.

## 3.1 UAVs under the Open Category

Parallel to the type of classification of $UAS$ described in Figure 2.1, it is also possible to describe the types of aircraft that fall under the Open Category based on the type of platform they use.

The choice depends mainly on the mission the $UAS$ is designed for and the operational situation where it is going to be deployed. Overall, they can be divided into:

- **Rotary Wing**: This category includes any aircraft that does not have aerodynamic surfaces to generate lift and employs propellers instead. Multi-rotor configurations use several vertically oriented rotors and differential thrust to achieve both forward and lateral control. On the other hand, single-rotor configurations use the same control laws as in a helicopter platform, achieving control with cyclic and collective pitch. Aircraft in this category are able to take off and land vertically.

- **Fixed Wing**: This category includes any aircraft that uses aerodynamic surfaces to generate lift. They resemble traditional airplane configurations and need a runway to take off.

- **Hybrid Configurations**: This category combines the capabilities of multi-rotor and fixed-wing platforms. They are able to perform Vertical Take-off and Landing ($VTOL$) while benefiting from the range and endurance of a fixed-wing prototype.

The following table summarizes the qualitative strengths and weaknesses of each platform category, comparing them for seven relevant metrics.

| Metric | Fixed-Wing | Rotary-Wing | Hybrid |
|---|---|---|---|
| **Size** | Moderate | Good | Poor |
| **Datalink Range** | Good | Moderate | Good |
| **Ceiling Altitude** | Good | Moderate | Good |
| **Endurance** | Good | Poor | Moderate |
| **MTOW** | Good | Poor | Moderate |
| **Launch Capability** | Poor | Good | Moderate |
| **Recovery Capability** | Poor | Good | Moderate |

Table 3.1: Qualitative comparison of UAV configurations: Fixed-Wing, Rotary-Wing, and Hybrid for different key metrics [15].

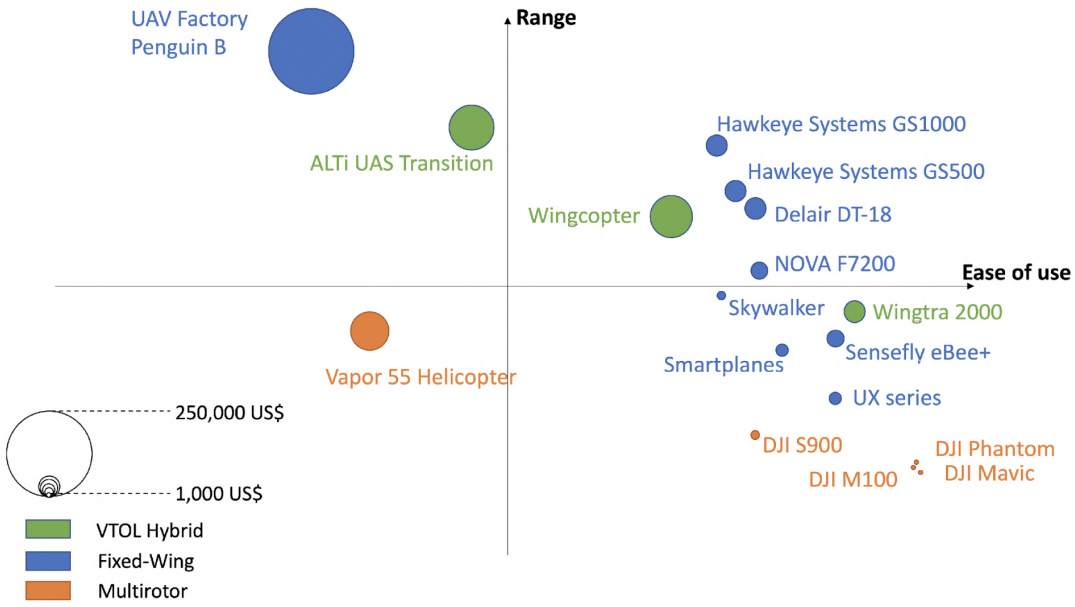



**Fig. 3.1:** *UAV* platform comparison with representative examples of commercially available systems [16].

## 3.2 Navigation Techniques

Due to the limited size of the $UAV$s described in Section 3.1, many of the radio-based Navigational Aids ($NAVAIDs$) present in general aviation, commercial aircraft, and almost any other aerial vehicle are unavailable for these small and lightweight $UAV$s. These include Non-Directional Beacon ($NDB$), VHF Omni-directional Range ($VOR$), Distance Measuring Equipment ($DME$), Instrument Landing System ($ILS$), and Area Navigation ($RNAV$).

Nevertheless, to obtain *positional awareness* and navigate effectively to fulfill any given mission, the following methods are most commonly employed in lightweight $UAS$.

### 3.2.1 GPS Navigation

$GPS$ technology is the principal method used to provide essential Positioning, Navigation, and Timing ($PNT$) data, allowing $UAV$s to determine their exact location, altitude, and speed, which is crucial for both autonomous flight and operator control.

The $GPS$ system was initially developed by the United States Department of Defense ($USDOD$) and became fully operational in 1995. The system is composed of a constellation of at least 24 satellites orbiting the Earth at an altitude of approximately 20,200 kilometers [17]. These satellites continuously transmit signals that can be received by $GPS$ devices on the ground, including those onboard $UAV$s. The $GPS$ device calculates its position by triangulating the signals from multiple satellites.

While the U.S. $GPS$ is the most widely used, several other satellite constellations exist, each developed by different countries:

- **GLONASS (Russia)**: Fully operational since 1996, with improvements over time, it provides global coverage similar to $GPS$.
- **Galileo (European Union)**: Introduced in 2016, it offers enhanced accuracy and is designed to be interoperable with other systems.
- **Beidou (China)**: Began regional service in 2000 and became fully operational globally in 2020, providing an alternative with additional features like short message communication.
- **QZSS (Japan) and NavIC (India)**: These are regional systems providing coverage over specific areas but are also used to complement global systems.

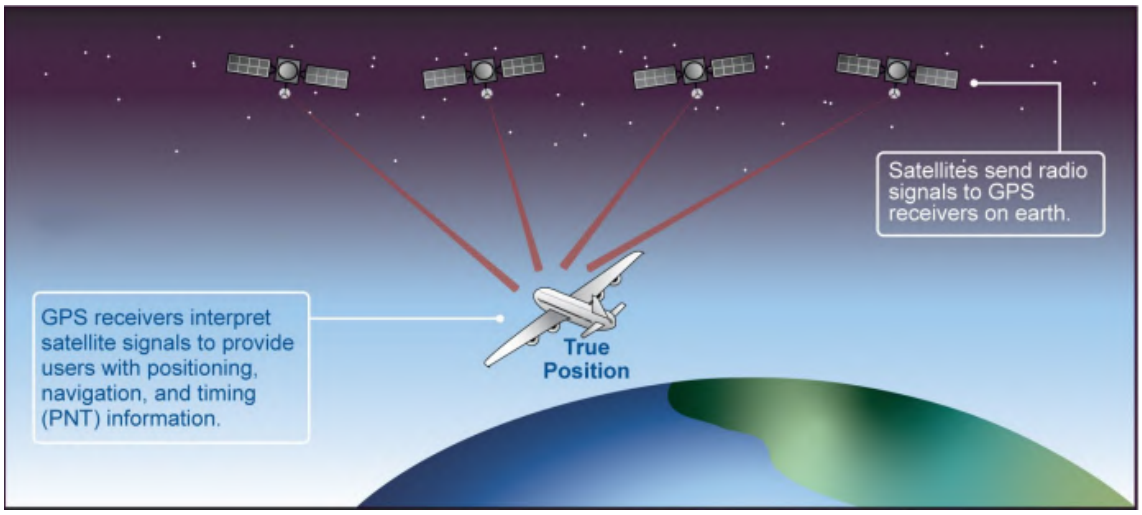



**Fig. 3.2:** Schematic illustration of the operational principle behind $GPS$ communication. [18]

One of the primary strengths of $GPS$ and its counterparts is their ability to offer near-global coverage, which allows $UAV$s to operate in almost any location on Earth. The widespread availability of $GPS$ signals provides $UAV$s with a reliable source for consistent navigation and enables the system to achieve positional accuracy within a few meters. This accuracy can be further improved with the use of Real-Time Kinematics ($RTK$) systems that incorporate corrections from a base station, reaching centimeter-level precision.

At the same time, $GPS$ signals are susceptible to interference from atmospheric conditions, topographical obstructions, and intentional disruptions such as jamming or spoofing, which can result in reduced accuracy or a complete loss of positioning data. Furthermore, $UAV$s that depend solely on $GPS$ for navigation are at risk of signal loss or degradation, posing a threat to mission safety and potentially leading to the loss of the aircraft.

While standard $GPS$ provides sufficient accuracy for many $UAV$s applications, more demanding tasks require greater precision. Real-Time Kinematic ($RTK$) technology is a type of Differential Global Positioning System ($DGPS$) that enhances $GPS$ accuracy by using a fixed base station to provide correction data to the $GPS$ receiver on the $UAV$. This process allows for centimeter-level accuracy, which is crucial for applications like surveying, mapping, and construction, where even small positional errors can lead to significant consequences.

$RTK$ systems work by measuring the phase of the $GPS$ signal's carrier wave, rather than just the signal timing used in standard $GPS$. By comparing this data between the base station and the $UAV$, the system can correct errors caused by atmospheric interference and other factors, resulting in highly precise positioning.

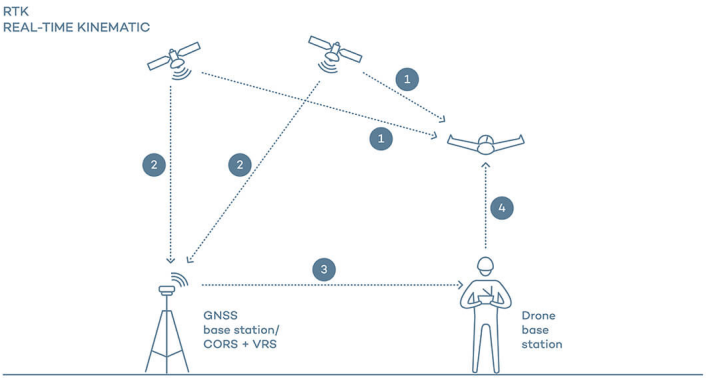



**Fig. 3.3:** GPS-RTK communication diagram.
1. Rover receives satellite signals. 2. Base station receives the same satellite signals. 3. Base station sends correction data to the drone operator. 4. Operator relays corrections to the rover for accurate positioning.

### 3.2.2 INS Navigation

Inertial Navigation System ($INS$) are a vital component in the navigation of any aircraft, particularly when used alongside $GPS$. $INS$ provides navigation data independently by using onboard sensors such as accelerometers, gyroscopes, and a magnetic compass to measure the $UAV$'s movement, enabling it to calculate position, velocity, and orientation without relying on external signals.

An $INS$ operates by tracking the $UAV$'s motion through continuous measurement of acceleration and rotational movement. Accelerometers detect linear acceleration along various axes, while gyroscopes measure the $UAV$'s rate of rotation. By integrating these measurements over time, the $INS$ can determine the $UAV$'s change in position and orientation from a set of initial conditions.

This method allows $UAV$s to navigate in environments where $GPS$ signals may be weak, obstructed, or unavailable. The ability of $INS$ to provide real-time navigation data without external input makes it an essential tool in a wide range of situations, as it shields the aircraft from external dependencies.

One of the primary challenges of relying solely on $INS$ is drift, which refers to the gradual accumulation of errors over time as the system integrates acceleration data to calculate position and velocity. Even small inaccuracies in Micro-Electro Mechanical Systems ($MEMS$) sensor measurements can lead to significant deviations in the calculated position as time progresses. This drift occurs because any small error in the measurement of acceleration or rotation is carried forward and compounded with each integration step.

Without correction, these errors can cause the $UAV$ to deviate significantly from its true path, making $INS$ alone insufficient for long-duration missions or precision-dependent tasks.

Several approaches are employed to address the drifting problem in $INS$:

- **Periodic $GPS$ Updates**: Regular $GPS$ updates help correct the drift by providing accurate position fixes that can reset the $INS$ calculations.
- **Kalman Filtering**: The use of Kalman Filters ($KF$s) helps to reduce noise and errors in the sensor data, optimizing the integration of $INS$ and $GPS$ information.
- **High-Quality Sensors**: Using higher-grade accelerometers and gyroscopes with lower noise and higher precision reduces the rate of drift, though this comes with higher costs and possibly increased weight.
- **Sensor Calibration**: Frequent calibration of the $INS$ sensors ensures that the baseline measurements are accurate, minimizing the initial errors that can lead to drift.

Although this method can yield powerful results, for situations where $GPS$ signals are unavailable for extended periods or where even small positional errors cannot be tolerated, additional systems like $ANS$ are often introduced to further enhance accuracy and reliability.

### 3.2.3 Alternative Navigation Systems (ANS)

$ANS$ allow aircraft and autonomous $UAV$s to operate when traditional navigation systems, such as $INS$ or $GNSS$, are degraded or unavailable. This concept involves using alternative methods, such as cameras, $LiDAR$, radar, radios, and star trackers, to support or replace conventional navigation systems [19].

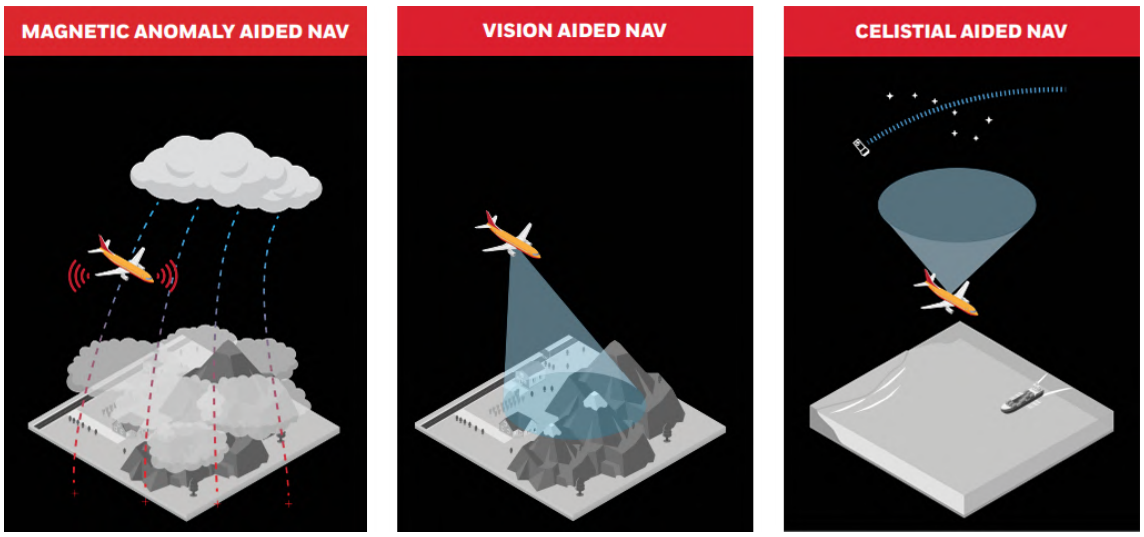



**Fig. 3.4:** Alternative Navigation methods, extracted from Honeywell Aerospace [19].

From all these options, Vision-Aided Navigation (Visual Navigation System ($VNS$)) is particularly attractive for $UAS$. $VNS$ refers to the use of visual data obtained from cameras to assist in the autonomous navigation and operation of an $UAS$. These systems enable $UAV$s to understand and interpret their surroundings, make decisions, and navigate through environments without human intervention. They are often combined with Inertial Measurement Units to create a Visual-Inertial Navigation System.

Some key scenarios where $VNS$ is used include:

- **Object Detection and Avoidance**: By employing cameras to map the surroundings in real time, the aircraft can recognize and avoid objects that may otherwise result in collisions.
- **Landing and Takeoff Assistance**: Precision landings are typically assisted by an onboard camera.
- **Visual Odometry**: This process estimates an aircraft's position and orientation in space by analyzing the changes between images from a downward-facing camera. It helps the $UAS$ track its position changes over time and reduces drift compared to using $INS$ alone.
- **SLAM**: This method allows an $UAV$ to simultaneously construct a map of its environment while tracking its location within that environment. Visual $SLAM$ uses camera images for this purpose and offers significant advantages when navigating in environments without pre-existing maps [20].

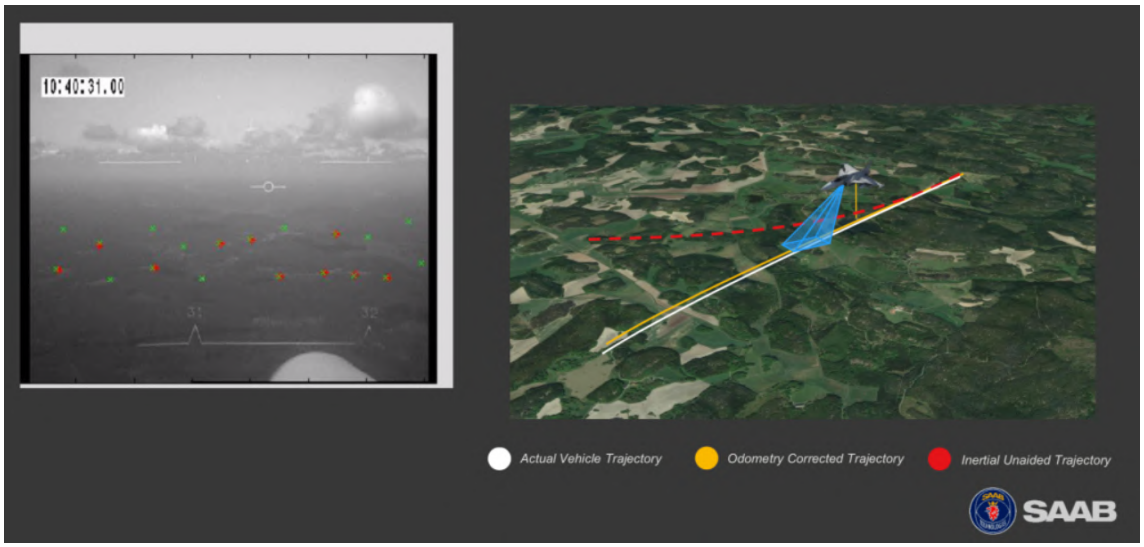



**Fig. 3.5:** Example from SAAB AB, comparing the trajectory estimation of an fighter based solely on INS versus odometry-assisted methods based on feature-tracking [21].

## 3.3 The GPS-denied environment

As anticipated in Section 1.1, scenarios may arise when an aircraft encounters unreliable, weak, or completely unavailable $GPS$ coverage.

The primary concern in these $GPS$-denied environments is the unreliability of $PNT$ data, which can result from various factors. These include **jamming**, where intentional interference disrupts signal reception; **spoofing**, where false signals mislead navigation systems; **constellation failure**, which occurs when satellite networks experience operational issues; and **topographical signal obstruction**, where physical barriers such as mountains or buildings block the signal, or in cases of interior and underwater navigation. In such conditions, navigation devices may deliver inaccurate location information or cease functioning altogether. This may cause problems and inconveniences for any type of aerial vehicle, but its repercussions are even more severe for $UAS$, as this is often their sole means of achieving *positional awareness*.

### 3.3.1 Jamming

$GPS$ signals are very weak by the time they reach Earth, especially compared to radio signals used by terrestrial devices. Jamming devices, or "jammers", confuse receivers by emitting radio signals at the same frequency as the $GPS$ but at much higher power. This form of interference hinders the ability of the $GPS$ to determine its correct position, as the receiver becomes incapable of detecting the satellite signals.

Commercial jammers and counter-$UAV$ systems operate by targeting aircraft with powerful electromagnetic energy at frequencies commonly used by $UAV$s, such as 900 MHz, 2.4 GHz, or 5.8 GHz for communications, and the L1 and L2 bands for $GNSS$. The intense signals emitted by these jammers reduce the Signal-to-Noise Ratio ($SNR$) to such a low level that the receiver is unable to distinguish the real signal from the noise [22].

This interference can occur intentionally or unintentionally when a $GPS$ receiver, which provides false information, is located near a high-power transmitter of some kind.

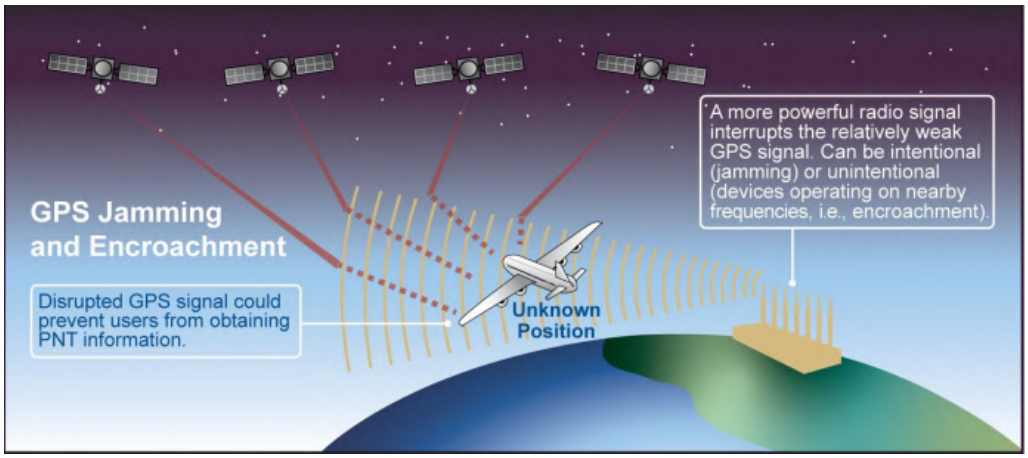



**Fig. 3.6:** Diagram of *GPS*-signal jamming. A high-powered signal blocks the communication layer between the aircraft and various *GNSS* constellations, resulting in an inability to achieve *PNT* data [18].

Jamming affects all forms of aerial transport. Commercial airliners are equipped with multiple *NAVAIDs*, and while they may encounter these issues near active conflict areas, the crew is well-prepared for such scenarios. They are trained to rely on backup instruments to maintain safe flight operations [23].


**Fig. 3.7:** Navigation Display (*ND*) and Multi-Function Display (*MFD*) of a Scandinavian Airline's A350 flight from Copenhagen to Bangkok during a period of jamming near the Ukrainian border. The *GNSS* antennas receive no information, displaying "NAV PRIMARY LOST," as shown in 3.8. Pilots also noted that the internal clock had to be manually reset [23].

| | Potential effect on systems/functions/ operations | Main Cockpit effects | | | | | |
|---|---|---|---|---|---|---|---|
| | | A300/A310 | A220 | A320 | A330/A340 | A380 | A350 |
| Navigation | Downgraded aircraft position computation | NAV GPS x FAULT GPS PRIMARY LOST on ND and CDU | GNSS NOT AVAIL on EICAS | NAV GPS x FAULT GPS PRIMARY LOST on ND | | | NAV GNSS x FAULT NAV PRIMARY LOST on ND |
| | Impact on **RNP operations** (RNP4, RNP 2, RNP 1) | | UNABLE RNP on EICAS | | | | |
| | Loss of **RNAV (GNSS)** capability | | APPROACH NOT AVAIL on EICAS | | | | |
| | Loss of **RNP AR** capability | N/A | | GPS PRIMARY LOST on ND | | N/A | NAV RNP AR CAPABILITY DOWNGRADED NAV RNP AR CAPABILITY LOST |
| | Loss of the **GLS** function (No GLS approach) | N/A | N/A | NAV GLS x FAULT | | | |
| | Loss of the **SLS** function (No LPV approach) | N/A | LPV NOT AVAIL on EICAS | N/A | N/A | N/A | NAV SLS x FAULT |
| | Limited **FLS** function (only available for VOR or NDB approaches) | N/A | N/A | FLS LIMITED TO F-APP+RAW on ECAM STATUS F-APP+RAW on FMA | | | |
| | Loss of **OANS/ANF** function | N/A | SMS FAIL on EICAS | N/A | ARPT NAV POS LOST on ND | | |
| Surveillance | Loss of the **Predictive TAWS** functions | TERR MODE "FAULT" light | TAWS MAP FAIL on EICAS | NAV GPWS TERR DET FAULT | | SURV TERR SYS x FAULT | |
| | Loss of **ADS-B out** reporting | No alerting system | ADS-B OUT FAIL on EICAS | NAV ADS-B RPTG x FAULT | | ADS-B POS RPTG LOST Memo | |
| | Loss of the **ADS-B IN (ATSAW)** function | N/A | N/A | NAV ADS-B TRAF FAULT or NAV ADS-B FAULT | | N/A | SURV ADS-B TRAFFIC x FAULT |
| | Loss of **ROW/ROPS** | N/A | N/A | SURV ROW/ROP LOST | | | |

**Fig. 3.8:** Potential effects of the loss of the $GNSS$ signal on Airbus systems/functions and their associated cockpit indications [24].

According to Airbus [24]: "In the event of a loss of $GPS$ signal, the Flight Management System ($FMS$) transitions from a mixed $GPS$/Inertial Reference System ($IRS$) position to an $IRS$-$DME/DME$ position, or $IRS$-$VOR/DME$, or purely $IRS$, in that order of priority." This option is not available for $UAS$, as $GPS$ is typically their primary and often only means of navigation.

### 3.3.2 Spoofing

Spoofing involves the use of an electronic device to transmit deceptive signals that mimic legitimate ones. In the context of radar, spoofing requires the transmitter to operate at the same frequency and Pulse-Repetition Frequency ($PRF$). The radar's main pulse triggers the spoofing transmitter, which, after a delay, sends a false echo [25]. Similarly, $GPS$ spoofing involves transmitting counterfeit GPS signals to mislead an aircraft about its actual location, potentially leading to navigation errors or unsafe operational conditions.

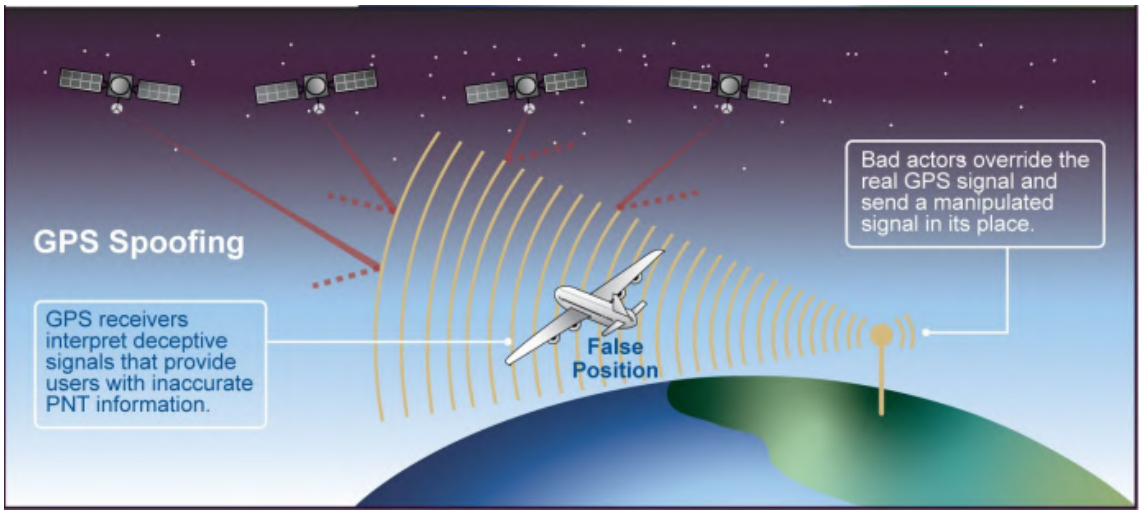



**Fig. 3.9:** Diagram illustrating *GPS* spoofing. Malicious entities transmit false information to confuse a target's navigation systems, making it believe it is in a different location [18].

A specific type of spoofing, known as "meaconing", involves intercepting and rebroadcasting navigation signals. In this process, intercepted signals are retransmitted at the same frequency but with higher power than the original, aiming to confuse and mislead navigation systems.

In military environments, particularly during electronic warfare, *GPS* spoofing is used as a tactic to disrupt or mislead an adversary's navigation and targeting systems. This form of electronic deception is intended to create confusion, misdirect operations, or undermine military strategies. By interfering with an aircraft's *GPS* signals, adversaries can manipulate flight paths, alter mission parameters, or create vulnerabilities for further attacks.

One method of executing a *GNSS* spoofing attack, known as the "carry-off" attack, involves initially transmitting signals that are synchronized with the legitimate signals detected by the target receiver. The power of the fake signals is then gradually increased, causing the receiver to lose track of the genuine signals.

The consequences of *GPS* spoofing can be severe. If an aircraft is deceived into believing it is in a different location, it could inadvertently enter restricted airspace, collide with other aircraft, or fail to reach its intended destination. In critical situations, such as combat operations or search and rescue missions, precise navigation is crucial, and any disruption can have serious consequences [26].

To counteract the threat of *GPS* spoofing, military and aviation organizations use various countermeasures and technologies. These may include alternative navigation systems, robust signal authentication protocols, and advanced detection techniques to identify and mitigate spoofing attempts.

### 3.3.3 Topographical Signal Obstruction

Topographical signal obstruction occurs when natural or artificial features block or attenuate $GPS$ signals, significantly impacting navigation accuracy. Natural obstructions like mountains, hills or forests can prevent $GPS$ signals from reaching the receiver, while man-made structures such as tall buildings, bridges, or tunnels can create "urban canyons" that either completely block signals or cause multipath errors. This results in either a total loss of $GPS$ data or reduced positioning accuracy due to signal degradation.

### 3.3.4 Constellation Failure

Constellation failure refers to situations where the $GPS$ satellite network is compromised due to a lack of operational satellites. This can occur because of technical malfunctions, aging satellites, or orbital issues. Satellite outages, orbital decay, or disruptions from space weather can impact the overall performance of the $GPS$ system. To address constellation failures, users can rely on other constellations like Glonass, Galileo, or Beidou, and leverage Ground Based Augmentation Systems ($GBAS$) to provide additional navigation support.

### 3.3.5 Space Weather

Several space weather anomalies can affect $GPS$ systems, including solar flares, geomagnetic storms, and Coronal Mass Ejectionss ($CME$s). Solar flares can emit high levels of X-rays and ultraviolet radiation that increase the density of the ionosphere, leading to signal delays. Geomagnetic storms, caused by disturbances in Earth's magnetic field, can create irregularities in the ionosphere that disrupt $GPS$ signal paths. $CME$s, which release massive bursts of solar wind and magnetic fields, can further worsen these effects, leading to significant signal degradation.

Ionospheric scintillation, on the other hand, refers to the rapid fluctuations in radio waves caused by irregularities in the ionosphere. When these fluctuations are intense, they can disrupt the ability of a $GPS$ receiver to maintain a stable connection with the satellite signals, potentially leading to a complete loss of signal lock. As a result, the $GPS$ receiver may be unable to determine an accurate position [27,24].

## 3.4 Autopilots

To achieve autonomous navigation, *UAS* require an autopilot system. This is accomplished through a flight controller onboard the aircraft, paired with a Ground Control Station (*GCS*) on the ground. Both commercial and open-source options are available for enabling autonomous flight.

Commercial solutions typically have a higher price point and focus on certification standards such as DO-254 and DO-178C [28]. These systems often maintain high-quality hardware components and a higher reliability due to their emphasis on flight safety. However, they usually operate with military-grade hardware, meaning their software and *GCS* modules are not publicly accessible.

Some notable examples of commercial autopilot systems include UAV Navigation, Embention, and Advanced Navigation.


**Fig. 3.10:** Examples of commercial autopilots for *UAS*, from left to right: Advanced Navigation's Boreas A70, Embention's 1X, and UAV Navigation's VECTOR.

In contrast, open-source options offer greater flexibility and customization, often at a lower cost. These systems are adaptable to various applications and experimental setups, making them appealing for research and development. Open-source autopilot systems benefit from a community-driven approach, which can lead to rapid innovation and the development of new features.


**Fig. 3.11:** Examples of open-source autopilots for *UAS*, from left to right: Orange Cube, Pixhawk 6C, and CUAV V5.

Several hardware and software options are available for open-source autopilots. Currently, Ardupilot and PX4 are the most widely used software solutions, while Paparazzi was well-regarded in academia over the past decade.

Regarding hardware, there is a diverse range of choices. Examples from the Pixhawk family include the Orange Cube, CUAV V5, or the Pixhawk 6X/C.

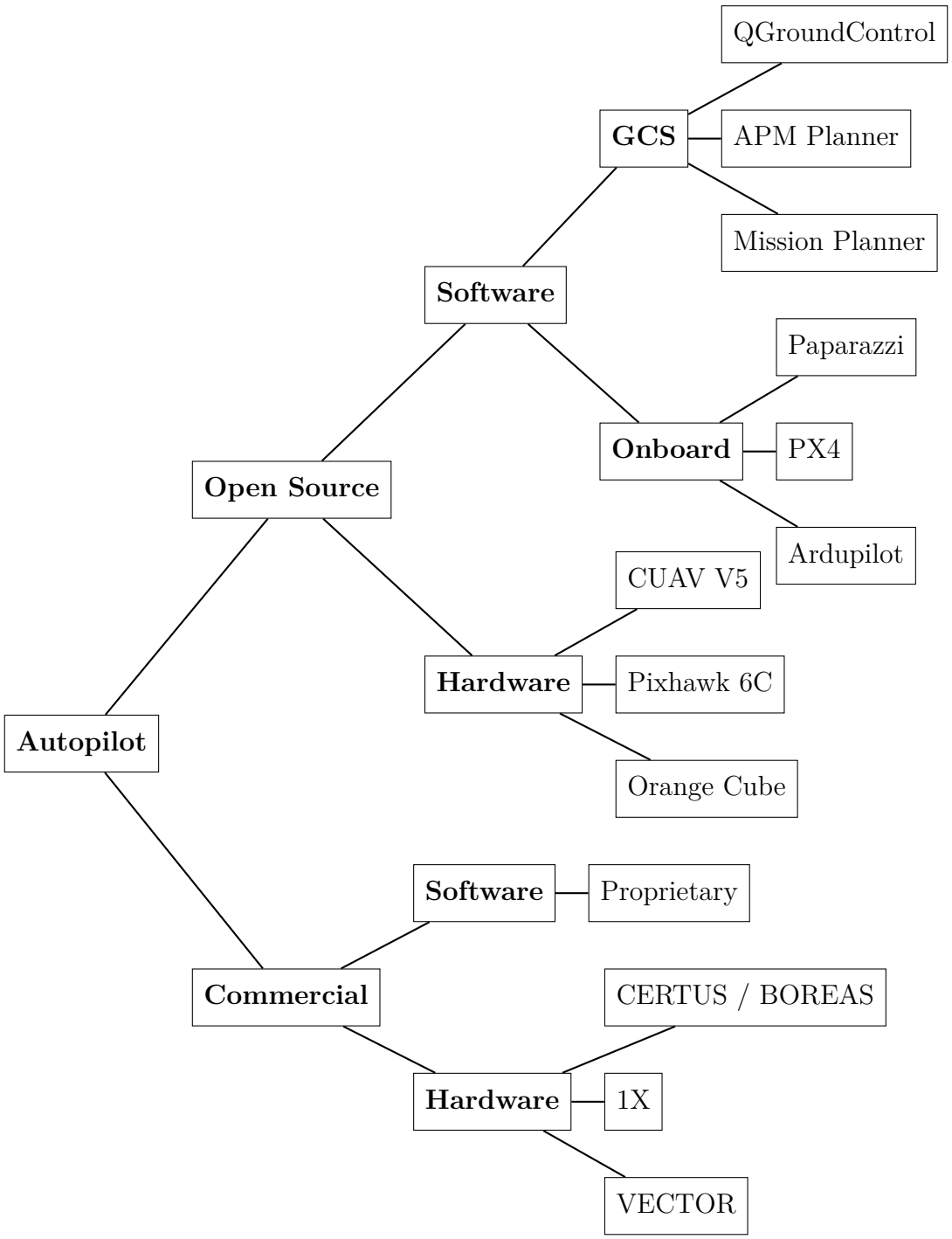



**Fig. 3.12:** Autopilot Systems Classification Chart, with relevant examples.

## 3.5 Artificial Intelligence in the Aerospace Industry

$AI$ and $ML$ are rapidly transforming industries by enabling machines to mimic human intelligence and learn from data. Significant advancements in these technologies over the last few years have been driven by increased computing power and vast amounts of data. $AI$ and $ML$ are now integral to tasks such as image recognition, natural language processing, and predictive analytics.

In the aerospace sector, and specifically in the $UAS$ industry, $AI$ and $ML$ are becoming key drivers of innovation. They have the potential to enhance flight safety, optimize operations, and enable autonomous navigation for UAVs. As $AI$ and $ML$ continue to evolve, they are set to revolutionize aerospace, making air travel and drone operations more efficient [29, 30, 31].

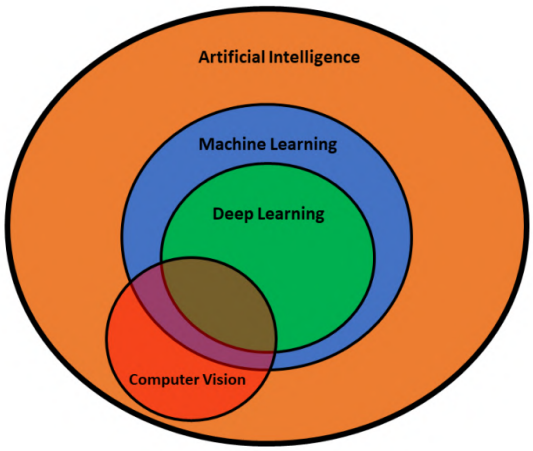



**Fig. 3.13:** Venn diagram of Artificial Intelligence, Machine Learning, Deep Learning, and Computer Vision [32].

There are numerous potential use cases for $AI$ in the aerospace industry. According to Airbus [33], some of these include:

- Knowledge Extraction
- Computer Vision
- Anomaly Detection
- Conversational Assistance
- Decision Making: Optimizing solutions for very complex constrained problems
- Autonomous Flight: Enabling the next generation of aerial vehicles
  - Autonomous Air-to-Air Refueling
  - Enhanced Situational Awareness and Advanced Analytics for Better Decision Making in Aviation (AirSense)

– Autonomous Taxi, Take Off and Landing ($ATTOL$), using computer vision

– DeckFinder, using computer vision for $GPS$-shaded environmental conditions

– $UAS-UTM$ for scalable, certifiable autonomy systems that power self-piloted aircraft

Traditional aviation software is certified for determinism under guidelines such as DO-178C (for avionics software) and DO-254 (for avionics hardware). However, $AI$ can cause the same inputs to produce different outcomes as the software evolves and "learns" over time. This evolving nature of $AI$ presents a challenge for maintaining certification determinism while ensuring safety.

To address this challenge, $EASA$ is developing the Machine Learning Application Approval ($MLEAP$) as part of its ongoing efforts to ensure the safe integration of $AI$ and $ML$ in aviation, in accordance with legislation from the European Union [10]. $MLEAP$ is a research project under the $EASA$-$AI$ Roadmap, aimed at establishing concrete guidelines for the certification and approval of machine learning applications in aviation, and finding suitable ways to integrate $AI$ and $ML$ into the current aerospace regulatory framework [34, 30].

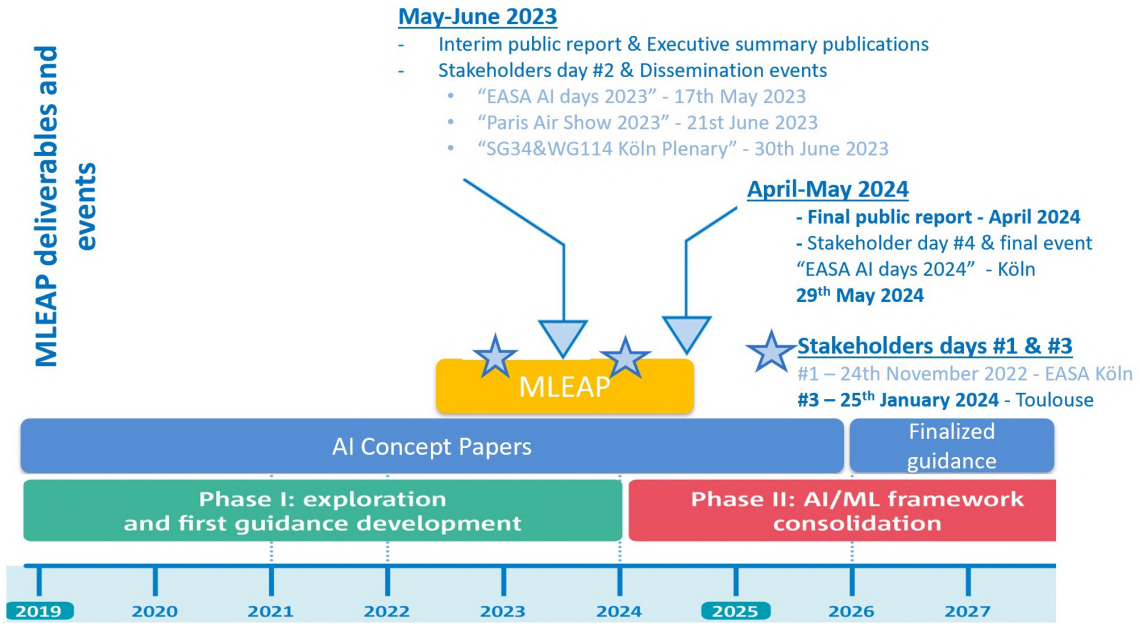



**Fig. 3.14:** $EASA$'s $MLEAP$ roadmap for $AI$ adoption in aviation [34].

It is also important to mention the development of the W-shaped Learning Assurance Framework, which is an essential component of the AI trustworthiness concept. This framework adapts development assurance principles to learning-based approaches.

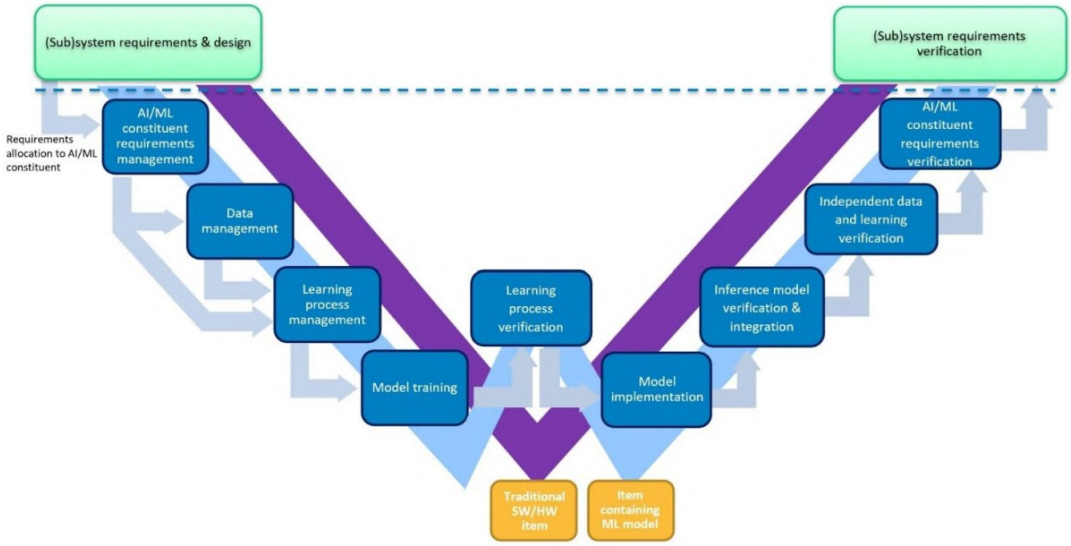



**Fig. 3.15:** Global view of the W-shaped learning assurance process overlapping the non-AI component V-cycle and safety assessment process. The dashed line separates the system level (upper part) and the AI level (lower part) [34].

### 3.5.1 Deep Learning

Deep learning is a subset of machine learning that focuses on algorithms inspired by the structure and function of the brain, known as Artificial Neural Networks ($ANNs$). These networks consist of layers of interconnected nodes that process and learn from large amounts of data. Deep learning models can automatically learn complex patterns and representations from raw input data, making them particularly powerful for tasks involving images, speech, and text [35].

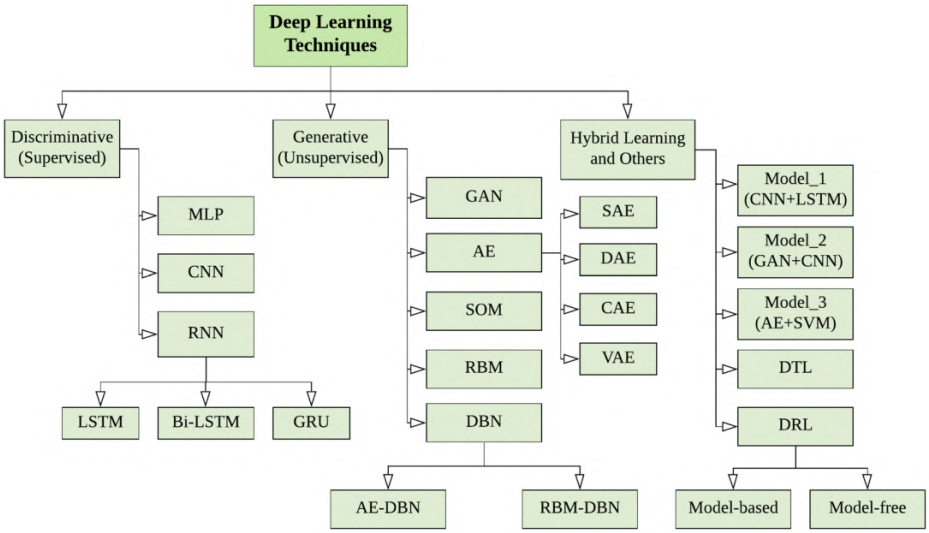



**Fig. 3.16:** Deep Learning Techniques Classification [36].

Deep learning becomes crucial when $UAV$s must interpret vast amounts of visual data from onboard cameras to navigate complex environments autonomously. Traditional algorithms struggle with the variability and complexity of real-world visual scenes, but deep learning models can handle this complexity by learning directly from data, leading to more robust and adaptable navigation systems.

Convolutional Neural Networks ($CNN$s) are a specialized type of deep learning model particularly well-suited for processing grid-like data, such as images. They automatically learn spatial relationships of features from input images, making them effective for visual tasks like image classification, object detection, and segmentation.

In $UAV$s and autonomous visual navigation, $CNN$s are the backbone of computer vision and visual perception systems. They enable $UAV$s to analyze visual data from onboard cameras and extract meaningful information, such as recognizing obstacles, identifying landing zones, or tracking moving objects. $CNN$s allow $UAV$s to operate autonomously in dynamic and unstructured environments, where real-time visual processing is critical for decision-making and navigation.

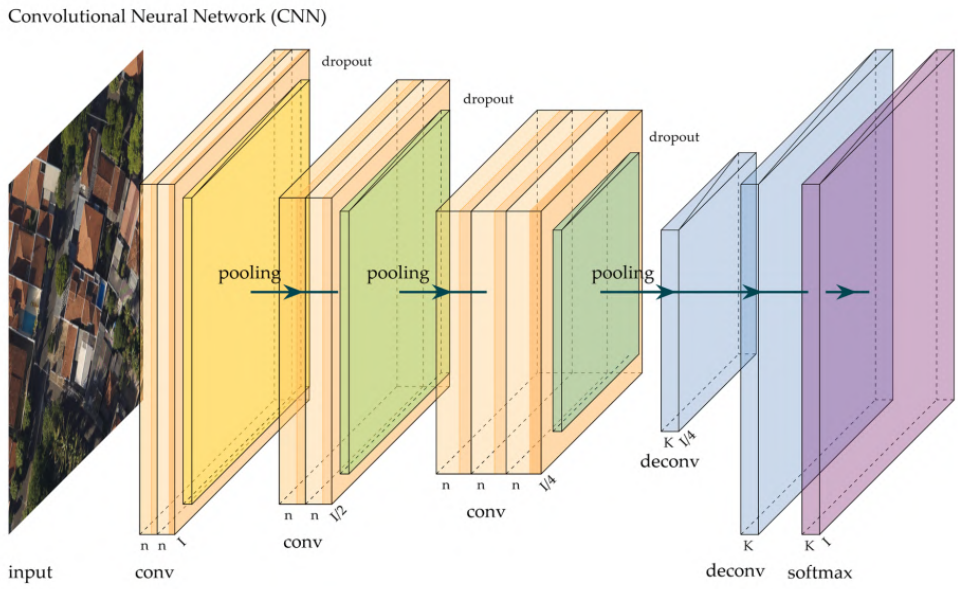



**Fig. 3.17:** Overall architecture of a CNN that includes several convolutional layers [37].

# 4 State of the Art

## 4.1 Introduction to GPS-denied Navigation

### 4.1.1 Historical Context

The first GPS satellite was launched in 1978, but it was not until 1995 that the initial constellation was declared fully operational. The technology became available for civilian use shortly thereafter, with its release occurring in 2000 [24].

Following the introduction to GPS-denied environments in Section 3.3, the issue has evolved over the past few decades from a primarily military concern, typically encountered only on the battlefield, to a significant problem with the potential to disrupt entire operations and industries in the civilian sector. A study from the United Kingdom estimated that the economic impact of $GNSS$ loss over a five-day period could reach €6 billion, affecting all industries, with the greatest impact on the transportation sector [38].

The first known attempt to disrupt $GPS$ communications dates back to 2003, during the Gulf War, when Iraqi armed forces attempted to throw US and UK precision-guided weapons off target [39].

Although these efforts were mainly unsuccessful at the time, the idea of exploiting the vulnerabilities of $GPS$ continued to grow. In December 2011, Iran successfully hijacked and captured a Central Intelligence Agency ($CIA$)'s RQ-170 unmanned aircraft [40]. The methods used for this cyber attack included both jamming and spoofing.

In the East, North Korea was also found responsible for $GPS$ jamming near the border city of Kaesong [41], affecting the traffic of at least 533 South Korean aircraft, which reported GPS failure.

On the civilian aviation front, three major incidents occurred from 2019 to 2023. The first involved a passenger aircraft that flew off-course during a GPS jamming event and nearly crashed into a mountain in June 2019 [42]. The other two events affected Denver and Dallas airports, which were impacted by $GPS$ jamming for 33 and 24 hours, respectively [43, 44]. These events involved runway closures and traffic rerouting. At Denver Airport, the $GBAS$ was unavailable within 50 nautical miles, and Air Traffic Controller ($ATC$) had to issue a notice to airmen to inform them of the situation. Investigations followed to find a viable solution to locate the jammer and restore normal airport operations as soon as possible.

After the major escalation event of the Russo-Ukrainian War on February 24, 2022, which included a full invasion of Ukraine, the situation in Europe worsened and prompted International Air Transport Association ($IATA$) to issue a safety assessment in September 2023 [45, 46]. This assessment highlighted that growing threats in the region might pose a significant safety risk. In the report, Airbus disclosed that the number of Radio Frequency Interference ($RFI$) events increased from 10,843 in 2021 to 49,605 within a year. $IATA$ emphasized that $GPS$ is critical for many aircraft subsystems, including $FMS$, Terrain Avoidance Warning System ($TAWS$), and Enhanced Ground Proximity Warning Systems ($EGPWS$). Should these problems continue to escalate, possible consequences included Mid Air Collisions ($MAC$), Controlled Flight Into Terrain ($CFIT$), and Runway Excursions ($RE$). The report recommended that receivers become multi-frequency and multi-constellation to minimize risk.

At the same time, in the context of the Russo-Ukrainian War, Ukraine began modifying open-source flight control software for their $UAV$s to include GPS-jamming detection algorithms and fallback options using visual-based navigation systems [47].

The conflict in the Baltic region has accelerated innovation in the $UAV$ industry, with a notable example in July 2024 of $AI$-enabled drones across the front line being targeted with increasing signal jamming from the Russian front [48]. This development has enabled $UAV$s to operate in larger groups, making the concept of intelligent swarms a reality.

In parallel, the civil world is experiencing shared effects, with companies like Finnair suspending flights to countries in the Baltic region, such as Estonia, due to constant $GPS$ jamming [49, 50].

Tensions also continue to grow in the Middle East since the surprise attack by Hamas on Israel on October 7, 2023. A significant event in this active conflict zone occurred in September 2023 over the UM688 Airway, a major Europe-Middle East route near Iraq and close to the Iranian border. More than 20 reports were issued due to spoofing [51], claiming GPS was unreliable. Some airplanes, such as an Embraer 650 flying to Dubai, almost entered Iranian airspace without proper clearance, while another airplane required $ATC$ vectors all the way to its destination in Doha. The $FAA$ issued a warning memo about "safety flight risk to civil aviation operation", and pilots reported that the $IRS$ became unreliable due to false position data injected by the spoofing attack, rendering all navigational aids useless as the $IRS$ relies on $GPS$ updates. There were also reports of $VOR/DME$ failures and UTC clock issues. Aircraft like Embraer, Gulfstream, and Challenger are not prepared for such events and experienced $GPS$ position shifts of over 60 nautical miles.

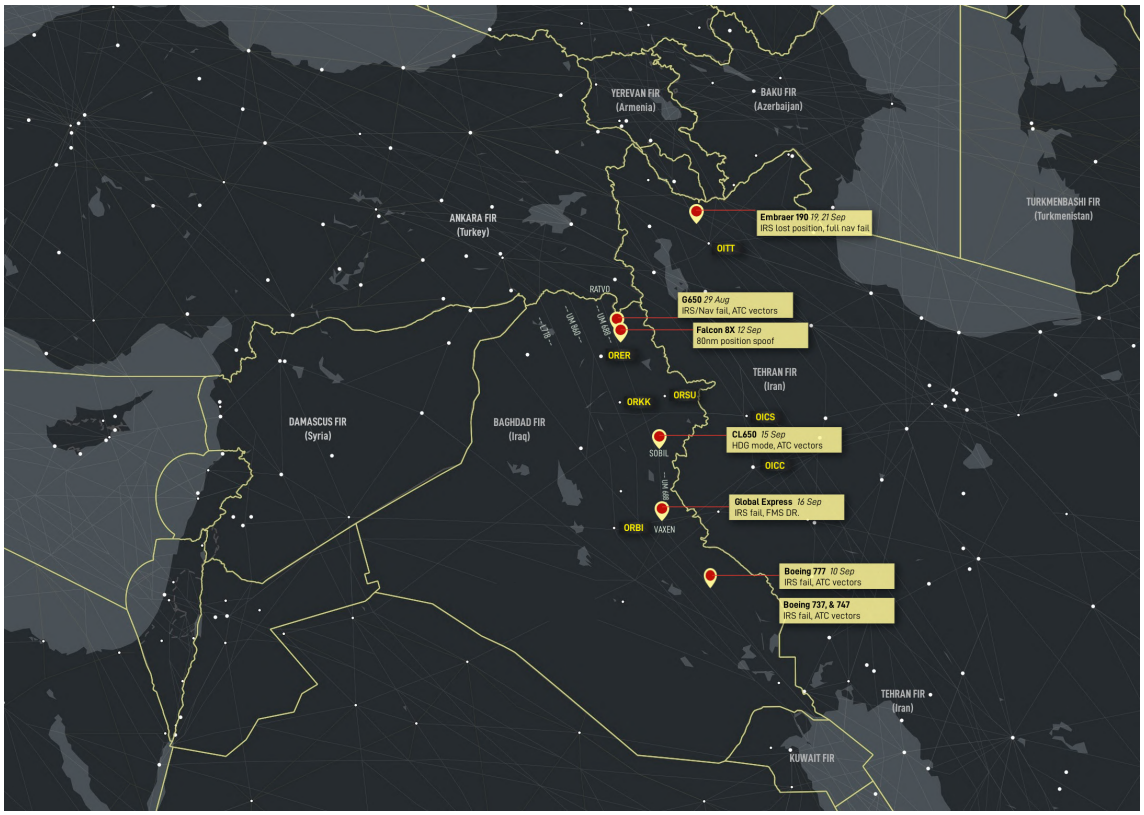



**Fig. 4.1:** Spoofing map in the Middle East during September 2023 [51].

Although these situations are tense and can lead to troubling outcomes, the situation worsens when imagining a future where thousands of drones are active in an urban area or near it, and suddenly they experience attacks on their sole navigational system, which is *GPS*-based. The Community-based ArduPilot Development Team is aware of this issue and has been working on solving various problems related to GPS-denied navigation through different Google Summer of Code (*GSOC*) events.

In 2019, they addressed the "Integration of Ardupilot & Visual-Inertial Odometry (*VIO*) Tracking Camera for *GPS*-less Localization and Navigation," followed by the 2022 edition [52], which included "Robot Operating System (*ROS*) Integration for Non-*GPS* Navigation and Off-board Path Planning," and the 2023 edition: "*GPS*-denied Autonomous Exploration with *ROS* 2" [53].

The latest edition, which began in 2024 and is set to conclude in November 2024, focuses on "High Altitude Non-*GPS* Position Estimation." This edition addresses the problem using a vision-based approach, employing a companion computer and a downward-facing camera. This demonstrates the ongoing research-driven need to solve navigational challenges through alternative navigation systems [54].

Lastly, *GPS*-denied navigation is not only an Earth-related problem. NASA's autonomous rover vehicles, such as Curiosity [55], Perseverance [56], Spirit [57], and Opportunity [58], have dealt with this issue daily, relying on alternative means of navigation.

Special interest in the field of *UAS* is *Ingenuity*, NASA's autonomous helicopter on Mars that shares the planet with the Perseverance rover. On April 19, 2021, this lightweight rotorcraft achieved the first-ever autonomous and controlled extraterrestrial flight [59].

While orbiters provided high-altitude aerial imagery with limited resolution and rovers allowed for rich, detailed images but are slowed by terrain and *VLOS* constraints, the introduction of an *UAS* able to work as a scouting platform and provide high-quality images proved to be a valuable innovation [60].

The goal of *Ingenuity* was to demonstrate that fully autonomous flight in a non-GPS environment, such as Mars, was indeed possible. The 1.8 kg vehicle performed 72 flights over the span of 3 years, far exceeding the 5 flights originally planned during the design phase to prove its airworthiness [61].

A visual inertial system was proposed to address the challenge of the lack of navigational aids, such as *GPS* or a strong magnetic field [62]. The proposed system ran on *COTS* hardware, combining a *MEMS* Inertial Measurement Unit (*IMU*) with a *LRF*, an inclinometer, and a nadir-oriented camera [63, 64].

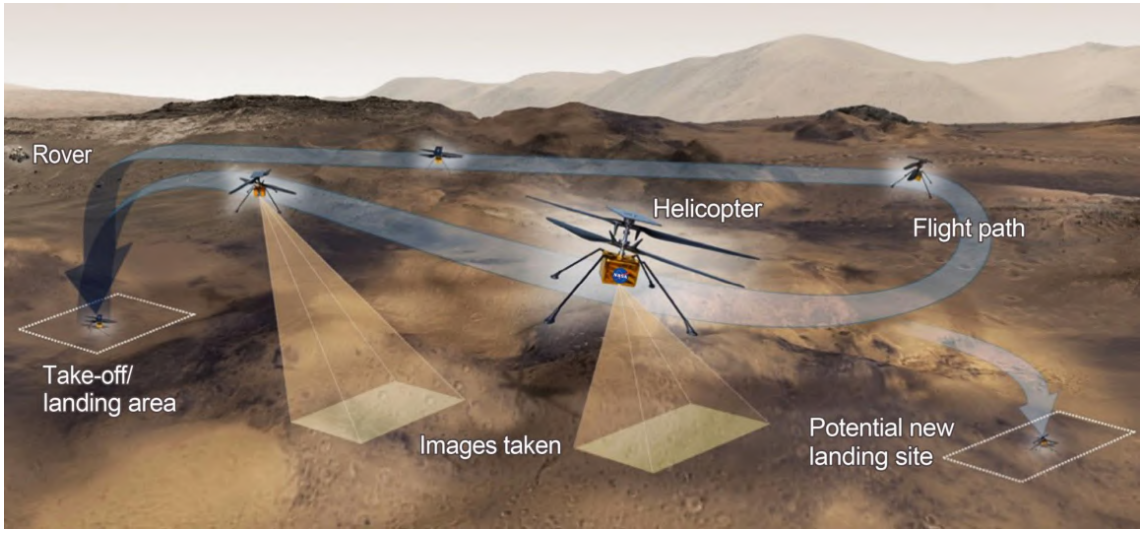



**Fig. 4.2:** Design Flight Pattern for NASA's autonomous helicopter *Ingenuity* [61].

### 4.1.2 GPS-denied Navigation Timeline

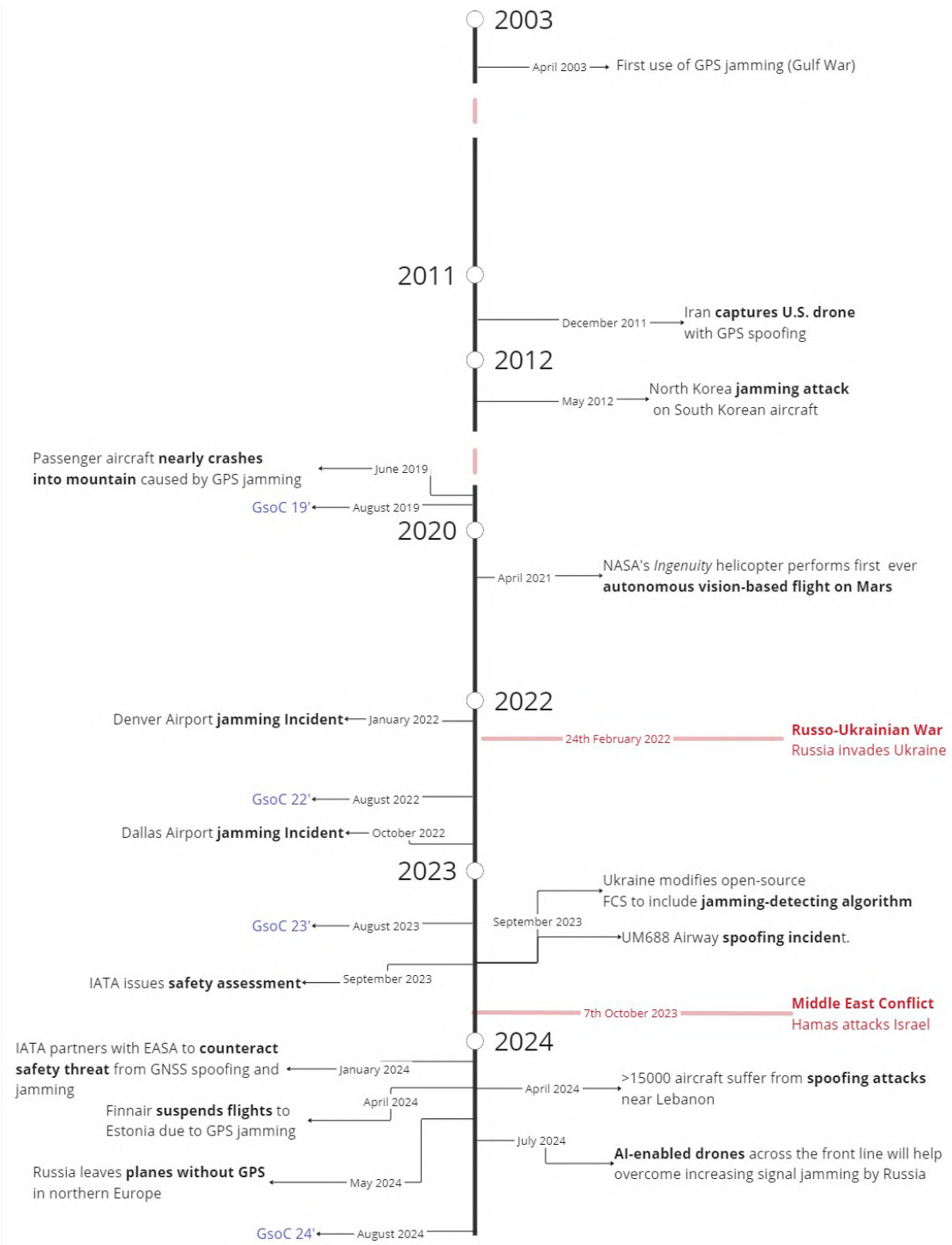



**Fig. 4.3:** GPS-denied navigation timeline, highlighting major events from 2003 to September 2024.

## 4.2 Alternative Navigation Techniques

Several alternative navigation approaches for solving the GPS-denied problem are presented throughout this chapter. Figure 4.4 displays a general map of these methods from the perspective of the technology employed.

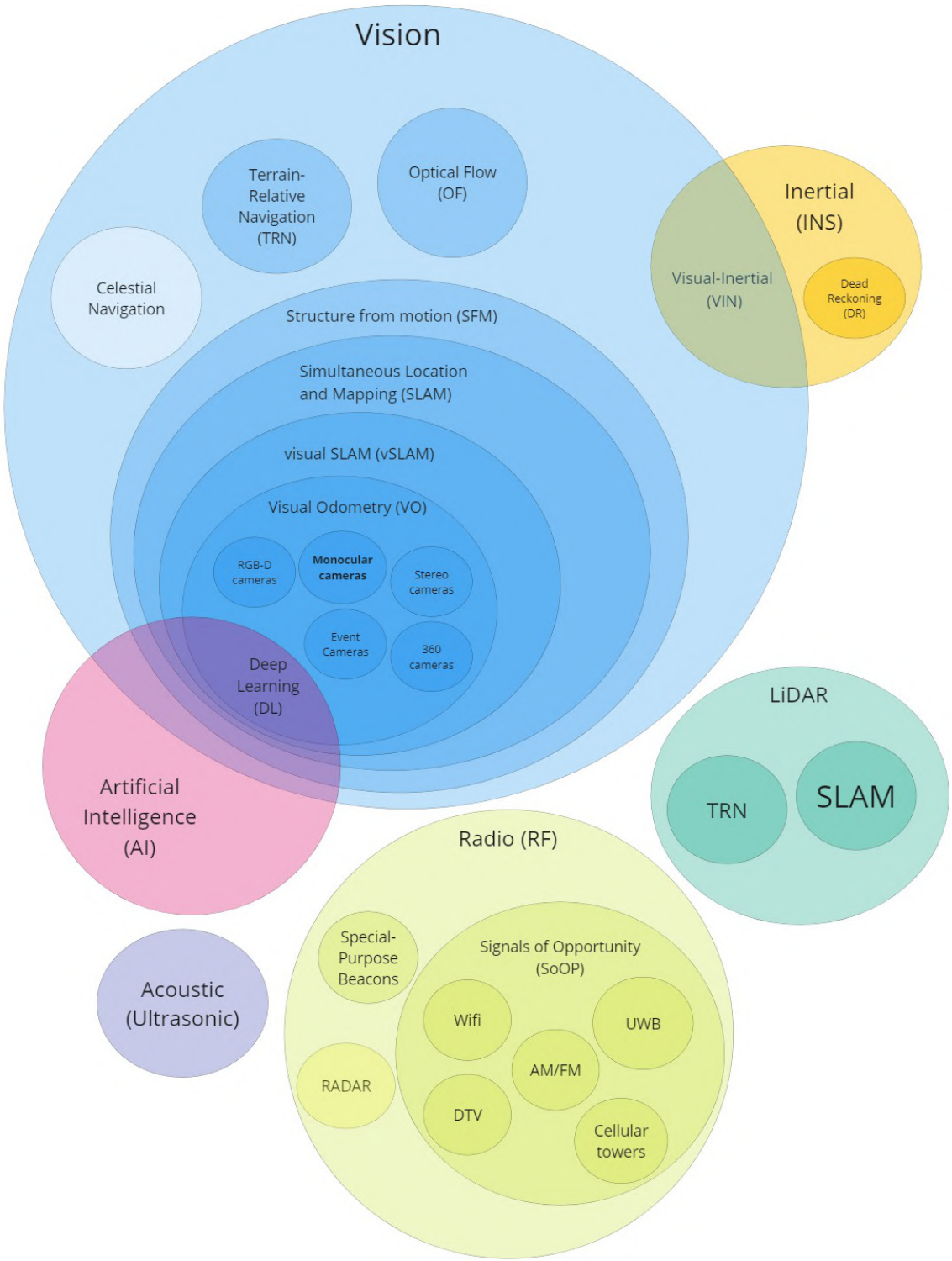



**Fig. 4.4:** Technology classification map of the Alternative Navigation Systems (ANS) for the state estimation problem.

### 4.2.1 Radio-based Navigation

Radio-based navigation encompasses various techniques that use radio signals to determine an aircraft's position and course. This includes traditional methods familiar to aircraft, such as $VOR$ and $DME$, which, as explained in Section 3.2, are incompatible with the type of $UAV$s this work focuses on. It also includes more advanced techniques [65, 66], such as leveraging **Signals of Opportunity** ($SoOP$) or deploying **Special Purpose Beacons**. However, the latter option cannot be deployed in denied areas and is therefore not considered further in this work.

$SoOP$ are non-navigation signals present in the environment that were not originally intended for positioning. Nevertheless, they are ubiquitous and freely available, which makes them attractive for assisting in the estimation of an aircraft's position [67]. Examples of $SoOP$ include:

- Digital TV signals
- Wi-Fi [68, 69]
- Cellular Towers
- Ultra Wide Band ($UWB$) [70]
- Radio broadcast AM/FM

By directly measuring amplitude and phase, it is possible to triangulate a position using different receiving signals from multiple known transmitters, calculating the Time Difference of Arrival ($TDOA$) or the signal strength difference.

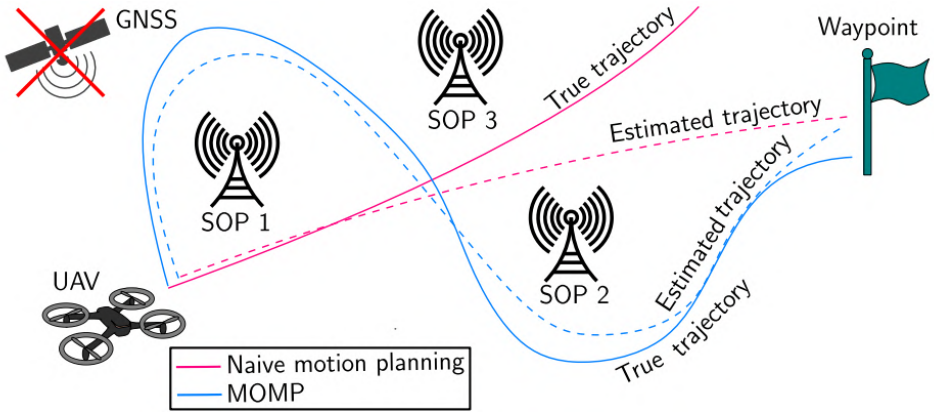



**Fig. 4.5:** Multi-objective motion planning (MOMP) strategy using $SoOP$ [71].

According to Duckworth and Baranoski [65], these $SoOP$ can be combined with an $IMU$ to create a hybrid approach using both direct-measurement and inertial/dead reckoning methods. This combination achieves high robustness at minimal cost. Data fusion is suggested to be performed using a Extended Kalman Filter ($EKF$), an algorithm that estimates the state of a nonlinear system by iteratively predicting

and correcting using noisy measurements. This makes it useful for accurately fusing $IMU$ data to estimate position, velocity, and orientation in real-time.

Yang *et al.* [72] highlighted some disadvantages and critical issues concerning $SoOP$, revealing why this approach has not been explored in depth by the research community:

- Lack of independent $SoOP$.
- Clock errors and synchronization.
- Compromised or unstable signal integrity.

At the same time, the necessary onboard equipment required to acquire and process a range of different signals is usually heavy and may not meet the payload requirements and limits for lightweight UAVs.

Other approaches were presented by Ma *et al.* [73], who used a military backup communication system for $GPS$ called Theatre Positioning System ($TPS$). This system employed a low-frequency radio signal to achieve better penetration through obstacles and reach indoor environments, although it suffered from unpredictable signal propagation of the $TPS$ groundwave signal.

Hardy *et al.* [74], on the other hand, suggested a swarm-based approach using two $UAV$s to determine the relative position from $IMU$ data and peer-to-peer ranging radio data without *a priori* knowledge of the aircraft pose. A relative pose measurement was then calculated using features extracted from a downward-facing camera on both $UAV$s, and this data was merged with $IMU$ data using an $EKF$. Monte Carlo simulations were performed for this approach, but no experimental testing was conducted.

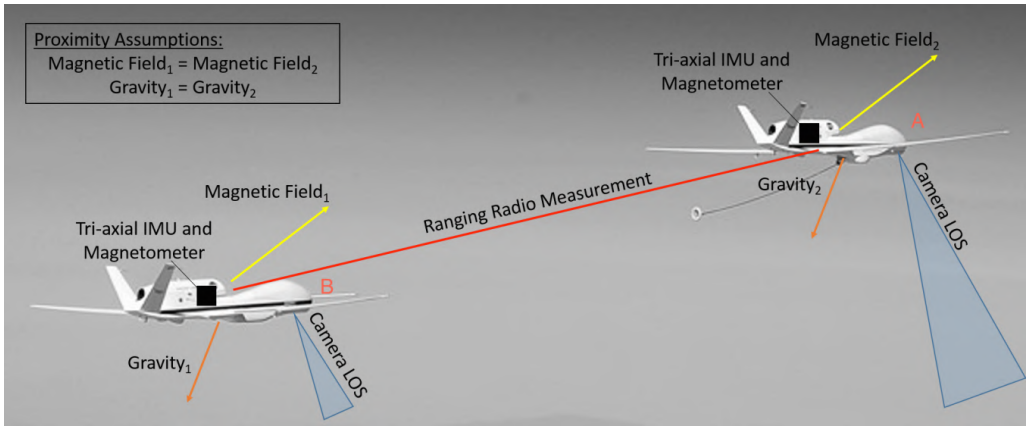



**Fig. 4.6:** Swarm-style cooperative pose estimation approach based on ranging radio data [74].

#### 4.2.1.1 Radar-based Navigation

Both Radio and Radio Detection and Ranging ($RaDAR$) technologies employ electromagnetic waves, but they serve distinct purposes and operate differently. While radio navigation employs known radio signals from $SoOP$ to determine a $UAV$'s position through triangulation, trilateration, or signal strength analysis, $RaDAR$ systems detect objects and measure distances by emitting radio waves and analyzing the time it takes for the reflected waves to return to the radar receiver.

Radar odometry has been explored for both Unmanned Ground Vehicle ($UGV$) [75] and $UAV$s [76, 77, 78, 79]. Quist and Beard [78] highlighted that $RaDAR$ offers advantages over vision approaches in terms of range resolution and is unaffected by environmental factors or visibility issues. Its resolution remains independent of range and performs well at high altitudes. Although radar has been used for motion estimation from moving platforms, its size, weight, power requirements, and cost have limited its application. However, advancements in technology over the past decade, including in Synthetic Aperture Radar ($SAR$), have mitigated these issues. This miniaturization makes radar a viable sensor for small UAVs, providing precise range measurements in all conditions. Despite progress and positive results, functional radar odometry systems are still lacking, indicating a need for further research.

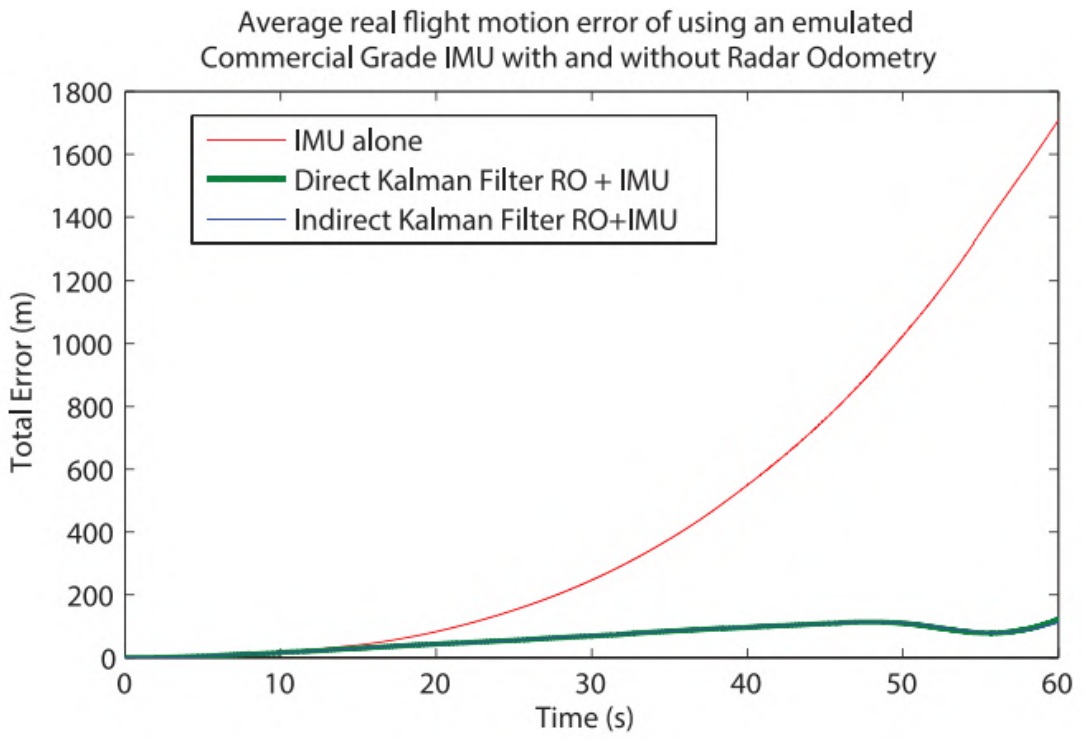



**Fig. 4.7:** Flight results of $SAR$ mounted on a Cessna aircraft compared to a commercial-grade $IMU$. The average drift error is reduced from 43% using only an $IMU$ to 6% using the radar odometry technique presented by Quist and Beard [78].

White *et al.* [79] extended this approach by incorporating a $CNN$ to learn and predict initial errors in $SAR$ image data, successfully reducing noise in the algorithm. However, the research cautions that the model may be overfitting, meaning it performs well on training data but may fail to generalize to new, unseen data, potentially capturing noise or irrelevant patterns.

### 4.2.2 Acoustic Navigation

Acoustic navigation, inspired by natural echolocation, utilizes ultrasonic sensors to measure distances based on the time delay, frequency shift, and amplitude variations of acoustic signals. This method operates independently of electromagnetic interference, making it immune to jamming [80].

Cabral *et al.* [81] proposed a solution that combines ultrasonic sensors for measuring distances between emitter and receiver modules mounted on quadrotors. This data is integrated with $IMU$ data using an $EKF$. The ultrasonic sensors measure distances, which are then fused with accelerometer, gyroscope, and magnetometer data from the $IMU$. This approach performs well in indoor environments, achieving centimeter-level accuracy at a 1 Hz frequency rate [82], [81], [83]. However, its scalability to outdoor environments is limited due to several factors:

- Attenuation and degradation of sound waves over longer distances.
- Multi-path interference, where sound waves reflect off surfaces, causing the $UAV$ to receive multiple, slightly delayed versions of the same signal.
- Limited effective range, typically a few meters.

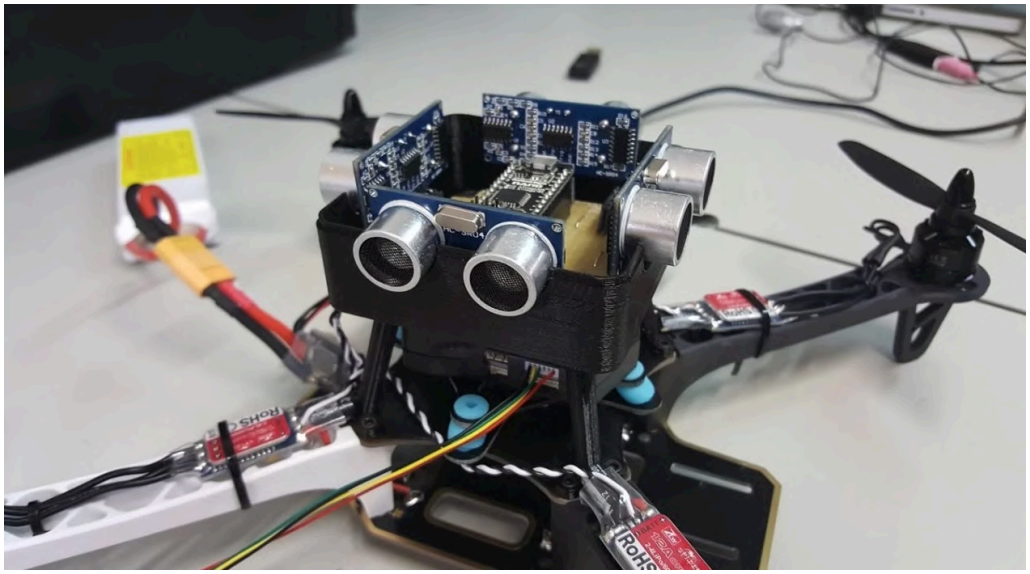


**Fig. 4.8:** Quadrotor drone equipped with four HC-SR04 ultrasonic sensors for obstacle avoidance. Although acoustic navigation is not ideal for outdoor environments, it is extensively used for obstacle avoidance due to its high frequency and accurate readings at close distances.

### 4.2.3 Laser-based Navigation

#### 4.2.3.1 *LiDAR* Approach

*LiDAR*-based navigation systems employ laser pulses to measure distances to objects in the UAV's surroundings, generating detailed 3D maps of the environment. By emitting laser beams in different directions, a 3D point cloud map of the surroundings can be created.

Braga *et al.* [84] proposed an efficient method to estimate a *UAV*'s position while flying over water, using a nadir-pointed camera to capture images. This method, limited by a maximum depth of 60 meters, relies on geo-referenced *LiDAR* images of the expected trajectory. The onboard system compares these stored images with incoming *LiDAR* data from the flight after performing transformations and edge detection.

Another study by Milijas *et al.* [85] explored various *LiDAR SLAM* algorithms, including LOAM (*LiDAR* Odometry and Mapping), Google's Cartographer, and HDL Graph Slam. LOAM uses *LiDAR* data to generate a map by extracting feature points from point clouds and tracking their relative motion to estimate odometry. However, LOAM does not support loop closures, limiting its ability to correct pose estimates based on previously mapped areas. In contrast, Cartographer is a graph-based *SLAM* method that integrates *IMU* data with *LiDAR*. The *IMU* provides an initial estimate, which the *LiDAR* refines by comparing the scan with the existing map. Cartographer's approach of periodically rearranging small submaps helps reduce odometry errors, resulting in smaller accumulated drift over time, particularly in the vertical direction, which is beneficial for higher altitude flights.

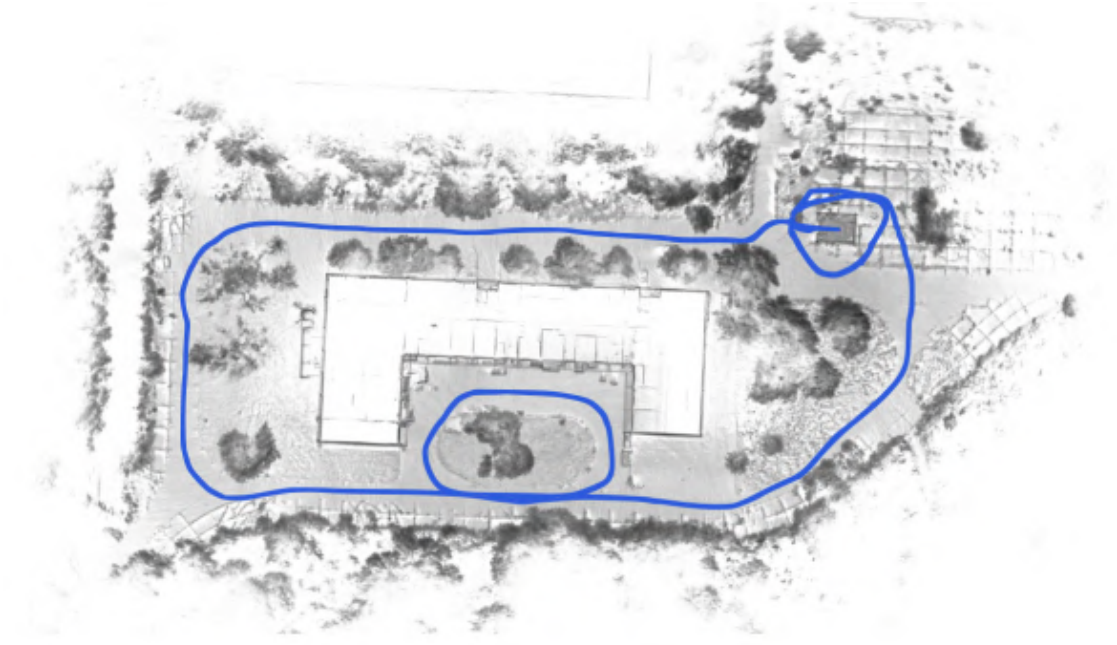


**Fig. 4.9:** Top view of the map generated by the Cartographer *LiDAR SLAM* algorithm. The blue line indicates the *UAV* trajectory [85].

Additionally, advancements by companies such as QinetiQ have led to new methods for communications in satellite-denied environments. For instance, the SDML (Software Defined Multifunction $LiDAR$) facilitates laser-based communications in $VLOS$ between a portable ground station and an airborne relay [86].

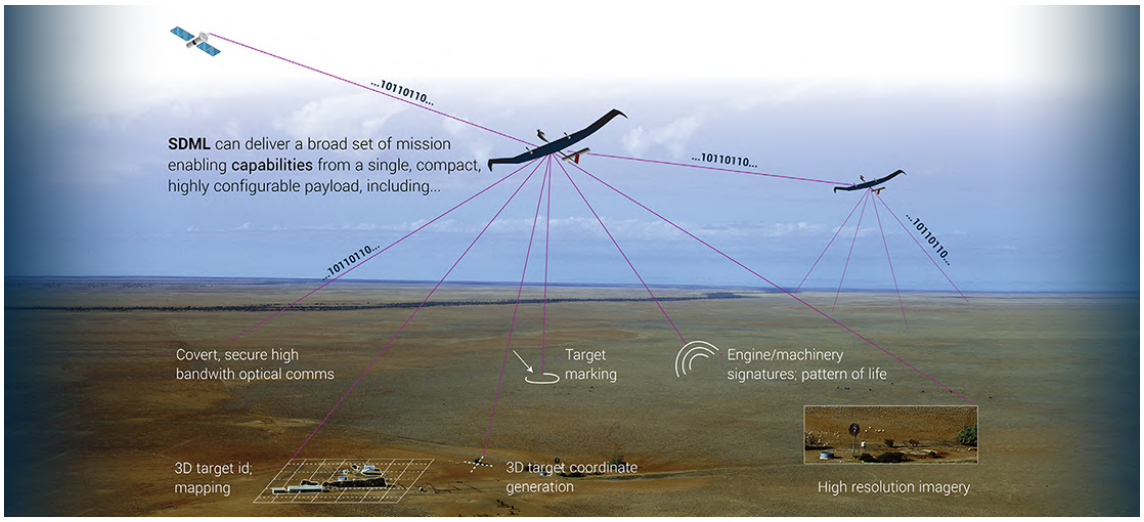



**Fig. 4.10:** Laser-based communication using the SDML system with an airborne relay [86].

#### 4.2.3.2 $LRF$ Approach

Laser Range Finders differ from $LiDAR$ in that $LiDAR$ studies the amount of optical power at the receiver end while $LRF$s focus on the exact time the optical signal bounces back. $LRF$s are well-suited solutions for long-range, low-illumination, and low-texture scenarios [87], where they significantly outperform visual methods.

Qin *et al.* [87] experimented with a rotating 2D $LRF$ onboard an aircraft to map a 3D environment. The solution was also coupled with a stereo camera, and the final results showed that the proposed system was robust, accurate, and fast.

Another representative example is the work by Carroll and Canciani [88] using a Terrain Reference Navigation ($TRN$) approach with a gimbal-controlled steerable $LRF$ coupled with an $INS$ to retrieve the maximum amount of navigational information. $TRN$ compares measurements of an aircraft's height above the local terrain with a Digital Elevation Map ($DEM$) to produce an estimate of its position. The method underperforms in complicated weather conditions, but otherwise, it proved to be a viable source of $PNT$ information for low-flying $UAV$s. However, a potential drawback is the weight of the required onboard systems.

### 4.2.4 Inertial Navigation

An inertial navigation system refers to a stand-alone device that uses a computer processor and motion sensors from an $IMU$, typically accelerometers, gyroscopes, and magnetometers, to calculate the position, velocity, and orientation of a given aircraft. The process of calculating the current position by using a previously computed one and integrating it over time is called *dead reckoning*.

This technique does not rely on external references and is, therefore, resistant to jamming or spoofing, making it particularly attractive for $GPS$-denied environments. However, because it relies on continuous updates from sensors, small errors in measurement can accumulate over time, leading to drift in the estimated position.

An $IMU$ consists of three types of sensors [89], which are typically $MEMS$-based for low-cost units:

- **Accelerometers**, which measure gravitational forces within a fixed coordinate system. When the aircraft is in motion, inertial forces are added to this measurement, resulting in a noisy signal that needs to be filtered.
- **Gyroscopes**, which measure angular velocity. $MEMS$ gyros use a small radially-vibrating structure to calculate the Coriolis force.
- **Magnetometers**, which may or may not be part of the $IMU$, act as a compass to determine the direction of Earth's magnetic field in two dimensions.

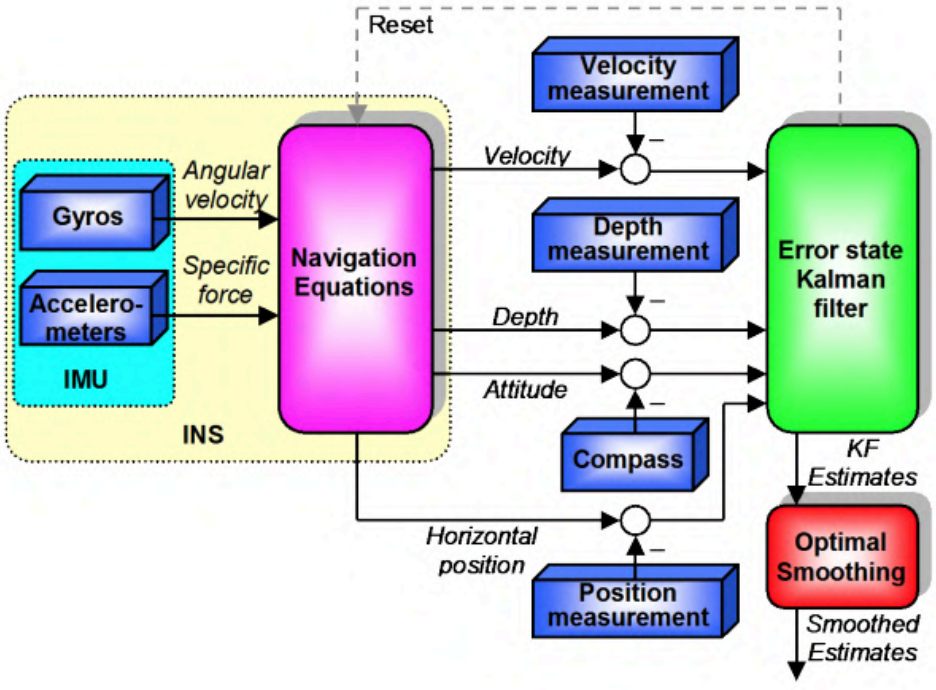



**Fig. 4.11:** High-level Inertial Navigation System Architecture [90].

By specifying initial values for velocity, position, and orientation, it is possible to integrate the sensed acceleration to retrieve the velocity, followed by a second integration to obtain the position. Integration must be performed in the correct direction; hence, orientation data is needed. This is achieved by integrating the sensed angular velocity [90].

As these time-dependent calculations are performed at each time step, errors in the gyroscopes and accelerometers accumulate, resulting in unlimited drift [91]. This drift can be reduced with the implementation of a $KF$ [92,93].

Solutions to this problem often involve combining $GPS$ data with $IMU$ data to perform periodic corrections and counteract the drift [94]. However, in GPS-denied situations, other methods must be employed to perform these periodic corrections.

### 4.2.5 Vision-based Navigation

Vision-based navigation comprises all methods that involve the use of cameras to analyze the environment and provide an $UAV$ with sufficient $PNT$ information.

Several approaches exist based on the type of vision sensors used, the algorithms for processing visual data, and the integration of additional data sources. These methods can range from simple feature detection and tracking techniques to more complex deep learning-based systems. For example, traditional methods might use techniques like optical flow or feature matching to estimate motion and location, while modern approaches might employ $CNN$s to recognize and classify objects or infer the structure of the environment. Additionally, some systems integrate visual information with other sensor data, such as $IMU$s, to enhance accuracy and reliability in different environments.

Vision sensors are becoming increasingly attractive for autonomous navigation thanks to the rapid development of computer vision. They are capable of providing substantial information about a vehicle's surroundings, have anti-interference capabilities, making them resistant to jamming from external sources, and are also passive sensors, which helps prevent the sensing system from being detected [95].

The types of cameras typically employed in vision-based navigation can be classified into four categories:

- **Monocular**: Captures 2D images and relies on additional algorithms for depth estimation. Although limited in depth perception, monocular cameras are widely used due to their simplicity and cost-effectiveness. Examples include the IMX477, GoPro series, or any other conventional camera that provides RGB information from a single lens.

- **Stereo**: Uses two synchronized cameras to provide depth information through disparity calculations. These cameras offer better depth perception than monocular cameras, but because the distance between the two cameras is small and fixed, the baseline is constrained by this [96], making the accuracy

of depth estimation poorer when the distance to the scene is much larger than the stereo baseline. Examples include the ZED 2 series or Intel RealSense Depth Cameras.

- **RGB-D**: Combines RGB and depth information from an IR sensor in a single image, offering direct depth data and simplifying navigation tasks in structured environments. These cameras are mostly used indoors as the IR sensor has difficulties in outdoor environments. Microsoft's Kinect V2 or Asus's Xtion are some of the most representative examples.

- **360º**: Captures panoramic views of the surroundings, providing excellent full situational awareness, but may struggle with depth information and increased data complexity. GoPro and DJI are among the manufacturers that provide 360º solutions.


**Fig. 4.12:** Typical visual sensors used for visual navigation. From left to right: Monocular IMX477, Stereo ZED 2, RGB-D Kinect V2, and 360º XPhase Pro X2.

One of the challenges associated with working with cameras and image-based pipelines in $UAV$s is the hardware constraints related to power, storage, and computational complexity due to the limitations of these vehicles.

#### 4.2.5.1 vSLAM

Simultaneous Localization and Mapping, or $SLAM$, is a framework that solves two problems at the same time: it estimates the motion of a system while building a map of the surrounding environment.

The goal of $SLAM$ is to obtain a global, consistent estimate of the $UAV$'s path. This involves maintaining a map of the environment to determine when the aircraft has returned to a previously visited area[1].

Visual Simultaneous Localization and Mapping ($vSLAM$) is a specific type of $SLAM$ that relies exclusively on images from cameras to solve the pose estimation problem. The already challenging problem of $SLAM$ becomes even more difficult with $vSLAM$ compared to using other sensors like RGB-D or Lidar, due to the ill-posed problem of reconstructing a 3D structure from 2D images. This approach

[1]This is called **loop closure**. When detected, this information is used to reduce the accumulated drift.

is often employed in systems where size, weight, and computational power are constrained [97]. According to a research review of the last decade by Gyagenda *et al.* [98], SLAM-based solutions for autonomous navigation comprised 76% of all the presented papers.

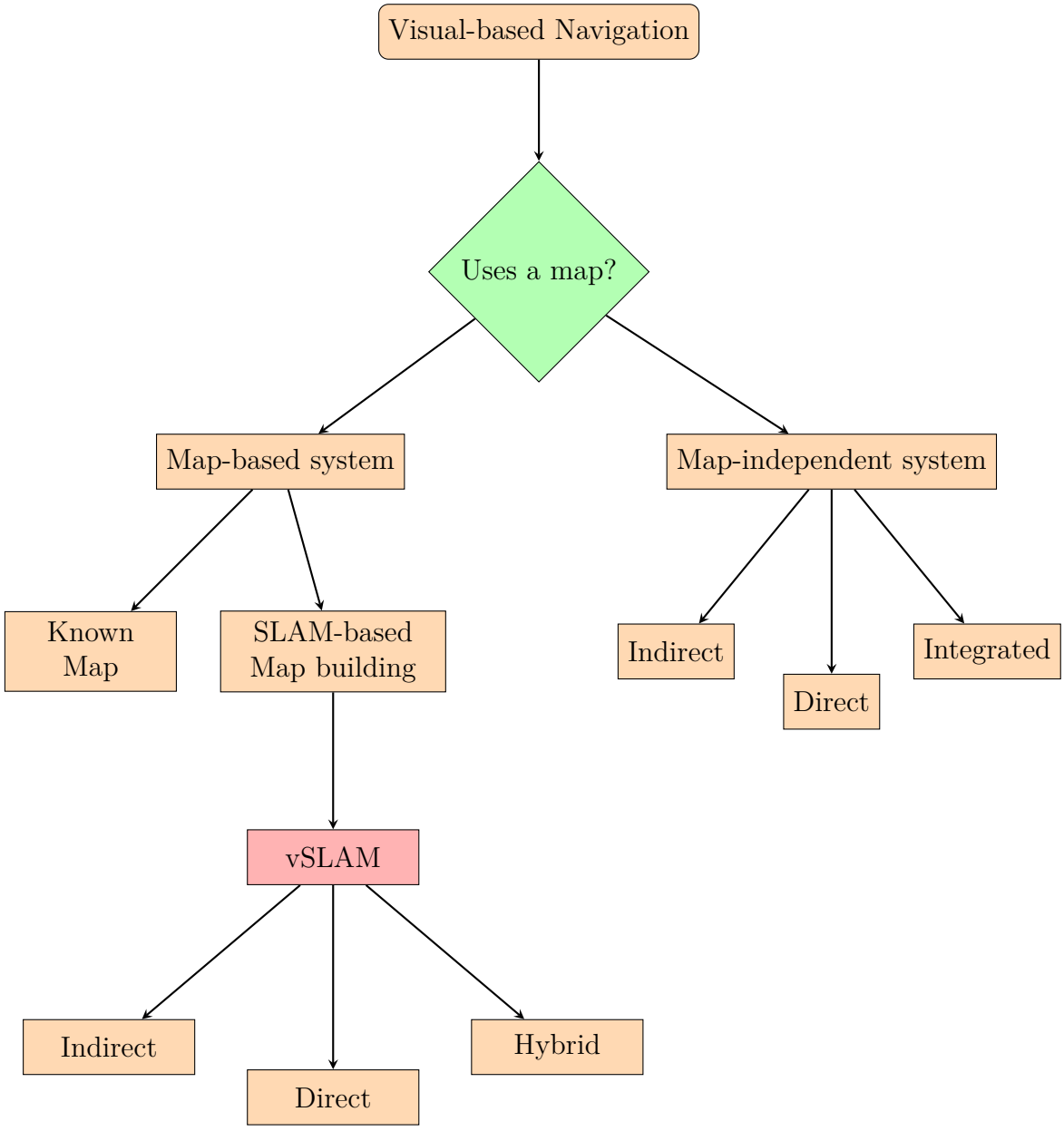



**Fig. 4.13:** Classification of Visual Navigation approaches [99]

The map-based system allows the UAV to navigate by incorporating detour behavior and movement planning, using a predefined map and the surrounding environment. These maps can vary in detail, ranging from a comprehensive 3D model of the environment to a simple diagram showing the connections between different elements. When a 3D map is available, the spatial distribution is already known through an octree map or an occupancy grid map [100].

When this information is not known, a map-building approach like $vSLAM$ is required. According to Lu *et al.* [95], this approach can be divided into three possible categories:

- **Indirect methods**: These methods first detect and extract features (such as points, lines, or edges) in the images, and then use these features as inputs for motion estimation and localization. The issue with this approach is that it is error-prone in texture-less environments, where features are difficult to find. Representative algorithms from this class include Parallel Tracking and Mapping ($PTAM$) [101] and Oriented FAST and Rotated BRIEF SLAM ($ORB - SLAM$) [102]. The latter presents improvements over traditional $SLAM$ by using binary descriptors that are less computationally expensive than the usual Scale-Invariant Feature Transform ($SIFT$)[2] or Speeded-Up Robust Features ($SURF$), thus achieving real-time performance in both indoor and outdoor environments.

- **Direct methods**: These methods use raw pixel information to accomplish localization and mapping. Since they do not rely on features, they perform well in texture-less environments and usually provide a more robust solution to photometric and geometric distortions in the image. Direct methods find denser correspondences but at the expense of increased computational cost. Representative algorithms from this class include Dense Tracking and Mapping ($DTAM$) [103] and Large-Scale Direct SLAM ($LSD - SLAM$) [104].

- **Hybrid methods**: These methods combine both of the above approaches. They first initialize feature correspondences using indirect methods and then refine camera poses based on direct methods, which are faster and more accurate.

The work by Al-Kaff [105] on vision-based navigation systems using a monocular camera employed a combination of $SIFT$ and a Fast Retina Keypoint ($FREAK$) descriptor, successfully reducing computational time while maintaining the performance of feature point matching. The proposed algorithm calculates the world-to-frame and frame-to-frame homographies[3] and then decomposes them to obtain the translation and rotation matrices of the quadcopter.

[2]The SIFT algorithm detects and describes local features in images, making them invariant to scale, rotation, and illumination changes, enabling robust image matching and object recognition.

[3]An homography is a transformation that maps points from one plane to another in projective space.

#### 4.2.5.2 Visual Odometry

Visual Odometry ($VO$) refers to the process of estimating the motion of the $UAV$ by analyzing changes in sequential images captured by the onboard camera, from one frame to the next. It tracks the position and orientation of the system relative to its starting point [106, 105].

$VO$ focuses on key points in the frame and infers the direction in which the vehicle has moved. It differs from the $vSLAM$ approach in that it **does not build a map** and is concerned only with the local consistency of the trajectory. As a result, $VO$ offers a trade-off between real-time performance and precision, being faster because it does not need to keep track of past orientations of the $UAV$.

It typically uses a monocular or stereo camera, although the monocular approach with a camera pointing downward is the most widely used method according to a study by Belmonte *et al.* [97].

For successful Visual Odometry navigation, the environment must exhibit certain characteristics:

- Sufficient illumination to ensure the image presents enough texture for feature extraction.
- Adequate overlap between consecutive frames of at least 50%.

The foundational model of $VO$ was established by Moravec [107], who, in his thesis, defined the first motion-estimation pipeline using a sliding camera that took pictures from different positions to find edges and navigate through the scene. This work laid the groundwork for both monocular and stereo $VO$.

Stereo and monocular approaches differ in that, with monocular $VO$, both the relative motion and the 3D structure data must be calculated from the 2D image data. Unlike stereo $VO$, which provides absolute scale measurements, monocular $VO$ initially assumes a distance of one between the first two images. This distance is further corrected when a third image arrives, based either on the 3D structure information or the trifocal tensor, which establishes projective geometric relationships between the three images [106].

$VO$ methods follow the same classification described for $vSLAM$:

- Indirect feature-based methods, which comprise the majority of $VO$ implementations.
- Direct (global) appearance-based methods.
- Hybrid methods.

The main issue with $VO$ is the accumulation of errors over time, which causes the vehicle to drift from its desired trajectory. An approach to mitigate this problem,

proposed by Fraundorfer *et al.* [108], is called "sliding window bundle adjustment." This method involves local optimization of the last camera poses.

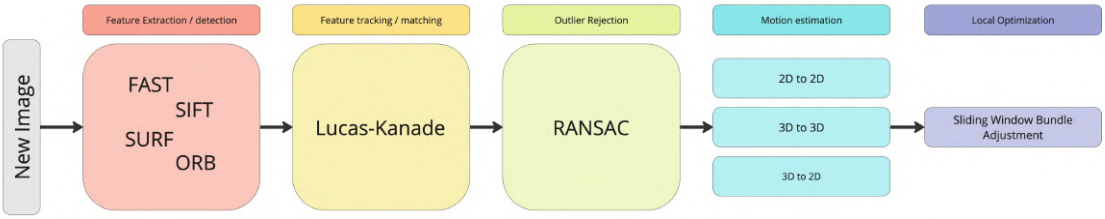



**Fig. 4.14:** Diagram of a Visual Odometry pipeline, with relevant algorithms for each step.

#### 4.2.5.3 Visual-Inertial Navigation

Visual-Inertial Odometry, or $VIO$, introduces a powerful augmentation to enhance navigation accuracy and robustness. $VIO$ integrates visual information from cameras with inertial data from $IMU$s, creating a more comprehensive and precise navigation system. The typical approach to achieve this integration is through an $EKF$ [109, 20].

Ning [110] argued that visual-only systems require many "perfect conditions" to achieve good results. These conditions include static scenes, sufficient illumination, enough scene overlap, and Lambertian textured surfaces.[4] In this context, $IMU$s, which are not limited by these conditions, add significant value to the overall solution.

Examples in the literature have tested these methods and compared them to $VO$-only approaches. Sharifi *et al.* [109] employed a ZED stereo camera coupled with an NVIDIA Jetson Tegra board and an $IMU$, comparing the results against $VO$-only methods (*libviso2* and *fovis* libraries for $VO$). Their findings showed that the drift error decreased from 13.9% to 1.3% when compared to the ground truth, revealing the powerful capabilities of data fusion.

Another approach by Leishman *et al.* [111] combined $vSLAM$ and an $IMU$ using an M$EKF$, a variation of the $EKF$ designed for systems with multiplicative noise, further reducing drift error. The results were verified through flight tests. Similarly, Zhao *et al.* [112] found that the position error was significantly reduced when compared to inertial data alone.

[4] Lambertian surfaces are idealized surfaces that are neither transparent nor reflective.

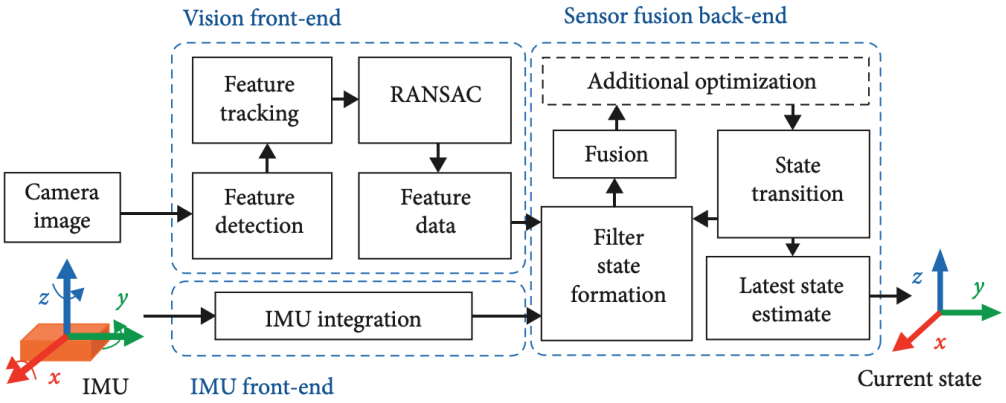



**Fig. 4.15:** Diagram of the Visual-Inertial Odometry pipeline [113]

Most of the research on $VIO$ has been conducted by $NASA$ in relation to Mars rover vehicles. In the field of $UAV$s, the most notable example is the successful deployment of the *Ingenuity* helicopter in April 2021.

This rotorcraft was equipped with multiple sensors to perform data fusion: $IMU$, $LRF$, a nadir-oriented (vertically downward) greyscale navcam, and an inclinometer. To compensate for error growth and drift, the vehicle combined data with MAVeN, a long-baseline visual odometry algorithm. This algorithm identified features tracked in subsequent images and calculated the difference between the $IMU$ prediction and the location determined by the feature tracker to correct the estimated state. MAVeN was implemented as an $EKF$.

The visual-inertial pipeline started with feature detection and tracking, performed using the Features from Accelerated Segment Test ($FAST$) algorithm to select corner-like features with sufficient contrast between the center pixel and its surroundings [114]. A Kanade-Lucas-Tomasi tracker was then used to track these features between images [115, 116]. To identify poor measurements, a Random Sample Consensus ($RANSAC$) algorithm was employed to prevent them from corrupting new estimates [117]. This method assumed that the base image features and search image features were related by an homography, implying that all features lay on a single plane, with the ground considered flat [5]. This assumption led to the neglect of misleading matches or features that violated the ground-plane assumption. Although this assumption was not entirely accurate, the flights on Mars, not exceeding 90 seconds, limited the drift. By assuming a single homography relates both images, the state only needed to be augmented with six scalar elements (three for position and three for attitude), drastically reducing the required computational power compared to traditional $SLAM$ algorithms, which would increase the Kalman Filter with three states for each of the $n$ features observed [62,64].

[5]Note that this simple assumption of a flat ground ultimately caused the helicopter's destruction. The rotorcraft tipped during landing, leading to a blade striking the ground and breaking.

#### 4.2.5.4 Optical Flow

Optical Flow ($OF$) sensors are devices used to measure the relative motion of an object with respect to its surroundings using a grayscale camera.

The typical configuration consists of an image sensor coupled with a processor running an optical flow algorithm, and sometimes a rangefinder to measure distances. By analyzing and tracking the edges of objects, it is possible to represent a distribution of apparent velocities based on changes in brightness [118].

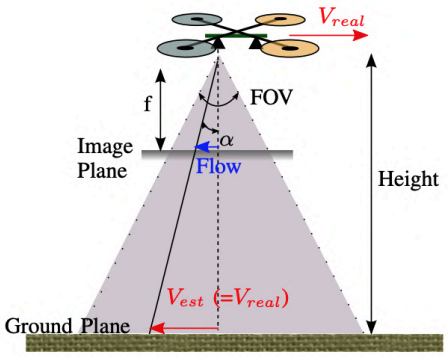



**Fig. 4.16:** $OF$ velocity estimation [119].


**Fig. 4.17:** HereFlow Optical Flow sensor.

Determining optical flow from image sequences involves computer vision algorithms that first extract features from the image and then track them. Several options are available, such as the $SIFT$ algorithm, which detects key points or features in the image and generates a *descriptor*, a vector describing the local image region around a key point, which is later used to match key points across different images. Another approach involves using a feature detector like $FAST$ or the Shi-Tomasi corner detector, followed by a local feature tracker like Lucas-Kanade Optical Flow ($LKF$). These methods are typically computationally expensive and require powerful computers to perform in real time [119].

Hu *et al.* [120] and Huang *et al.* [121] proposed solutions that couple $OF$ with $GPS/INS$ systems [122] and integrate everything using an $EKF$ [120, 121], rather than relying exclusively on $OF$ as done by Gageik *et al.* [123], which resulted in a higher-than-expected position error. This can be observed in Figure 4.18.

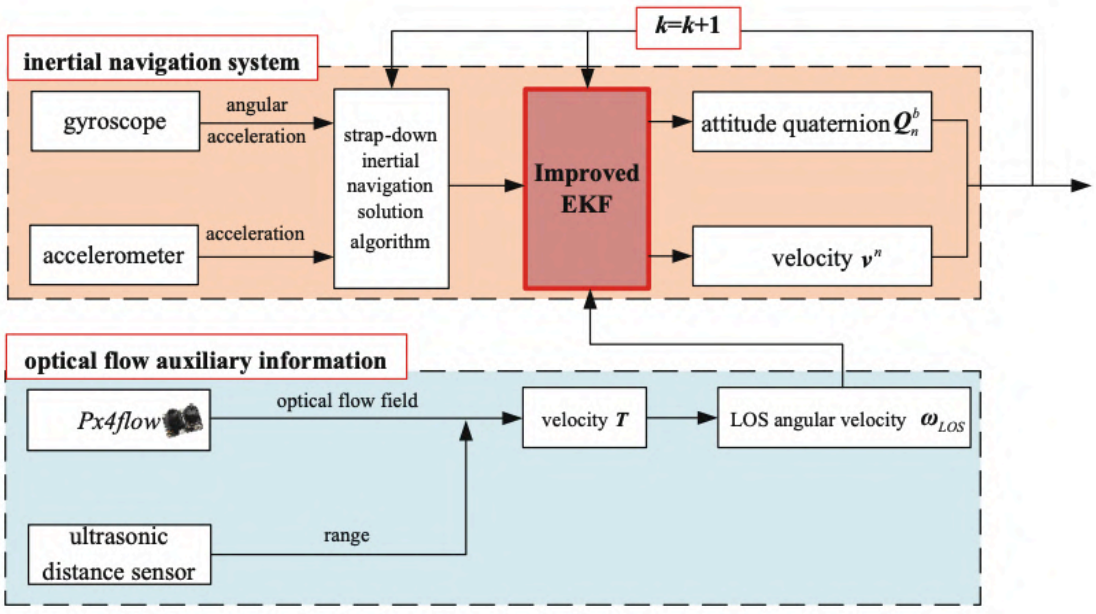



**Fig. 4.18:** Optical Flow assisted navigation system [120].

To validate these findings, Chao *et al.* [124] performed experimental testing of a $SIFT$-based $OF$ system using a GoPro camera combined with an Optic-Logic RS400 $LRF$. Data processing was performed on the ground, showing desirable matching between the vision data and the $IMU$ data. Researchers expect that further work will include a GPU-powered computer onboard to enable real-time testing.

Shan *et al.* [125] explored the possibility of a geo-referenced localization framework based on Google Maps data to achieve drift-free navigation using $OF$ sensors. Arguing that $SLAM$ is not useful if the scene is not revisited more than once, the study uses Histogram of Oriented Gradients ($HOG$) features[6] to compare to Google Maps, achieving excellent results after offline testing, with Root Mean Square Error ($RMSE$) of the same order of magnitude as the ground truth provided by $GPS$. The proposed method, called HOP, written in C++ using OpenCV, combines $HOG$, $OF$, and a particle filter to conduct a coarse-to-fine search to find the $UAV$. Drawbacks of this method include the effects of weather, changes between Google Maps data and reality, and variations in scale, orientation, and illumination of the scene.

[6]HOG features are used for object detection by dividing an image into cells, calculating the gradient orientation in each cell, normalizing the results, and forming a descriptor that captures the local shape of objects.

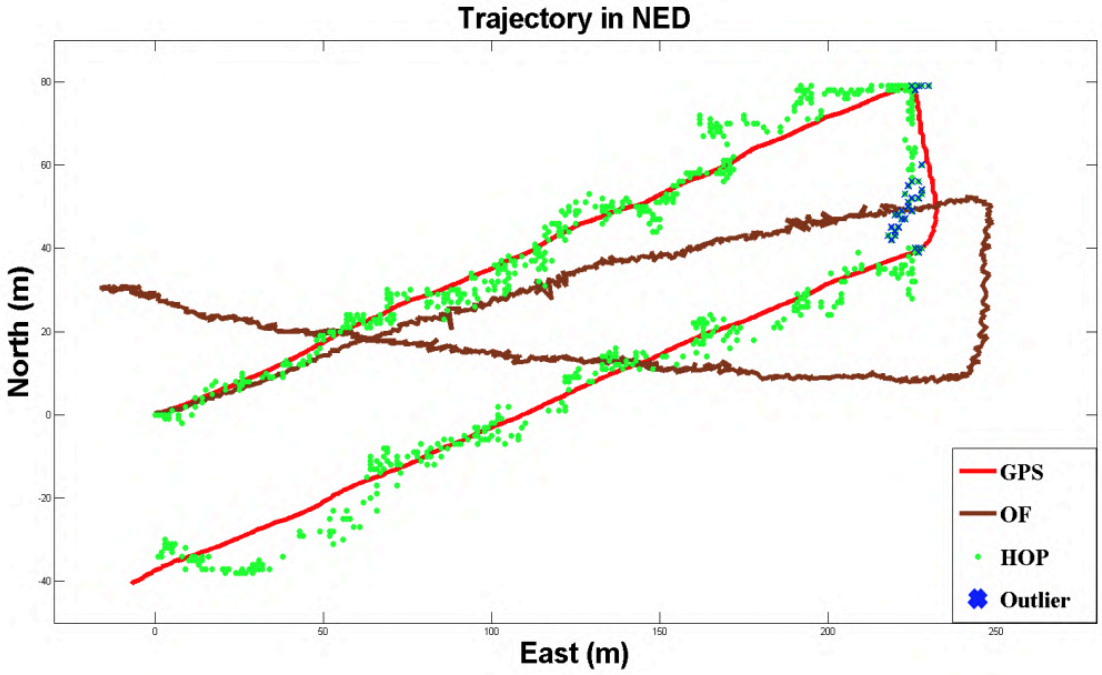



**Fig. 4.19:** Trajectory analysis comparing *GPS* (red), dead-reckoning *OF* (brown), and the HOP algorithm (green dots), showcasing the drift reduction achieved by Shan *et al.* [125].

#### 4.2.5.5 Celestial Navigation

Celestial Navigation is based on the highly predictable nature of celestial objects, which can be used as references for navigation both at night and during the day [126]. Historically, it has been employed in military aircraft systems like the SR-71, RC-135, and B2, as well as in Intercontinental Ballistic Missile (*ICBM*) guidance systems [127, 128].

**Fig. 4.20:** SR-71's Astro-inertial Navigation System from 1962, before *GPS* was available [128].

Celestial-based sensors can calculate absolute position and provide highly accurate attitude information regardless of $GPS$ accessibility. One example is the Sky Position and Azimuth Sensing System (Skypass), a ground station location system [129].

Honeywell's celestial navigation solution combines $IMU$ data with star tracker image data through an $EKF$ to provide a blended navigation solution. The star tracker monitors two types of objects: stars and Resident Space Objects ($RSO$s), which are primarily Low Earth Orbit ($LEO$) satellites. The observations of the stars are compared with their known positions in space, allowing for accurate estimations of the vehicle's attitude. Simultaneously, the observation of $RSO$s relative to background star formations provides accurate information about the aircraft's position, after comparing it to ephemeris data stored in the $UAV$'s onboard computers [130]. Although this technology has been successfully tested on both fixed-wing and rotorcraft vehicles [131], it is limited by weather conditions, and is heavy and expensive. Further research by the academic community is needed to improve this technology as a reliable potential solution.

#### 4.2.5.6 Deep Learning and AI-supported Navigation

Parallel to traditional map-oriented methods for solving the state estimation problem, other approaches have started to emerge in recent years. The computer vision domain underwent a revolution thanks to breakthroughs achieved in Deep Learning, significantly increasing the performance of image processing algorithms [132].

Some of these implementations have led to advancements in path planning and obstacle-avoidance algorithms. One of the main paradigms in this area is DroNet, a $CNN$ that allows $UAV$s to navigate while avoiding obstacles [133]. Although trained on city streets from the perspective of urban vehicles, DroNet's navigation policy generalizes well. A study by Amini Rad *et al.* [134] using DroNet running on a GPU-powered companion computer (Jetson Nano) revealed that the drone achieved real-time performance at 47 fps, with potential for doubling that frame rate at the cost of increased power consumption. Palossi *et al.* [135] conducted similar experiments, this time with nano drones achieving 18 fps on a 64 mW computer. In another related study, Kyrkou *et al.* [136] focused on image detection using similar deep learning techniques.

In the field of state estimation, the most notable applications are related to Visual Odometry (VO). Messikommer *et al.* [137] proposed a method where a neural network trained within a Reinforcement Learning (RL) framework guides decisions within the $VIO$ pipeline, such as triggering keyframes and tuning the grid size parameter used to detect features within these frames. This approach eliminates the common need for manual tuning of hyperparameters in $VO$ implementations.

Wang *et al.* [138] highlighted that while $VO$ solutions have demonstrated effectiveness in various scenarios, each implementation typically requires specific fine-tuning to perform well in those environments. They proposed DeepVO, an end-to-end framework for monocular $VO$ using Region-based Convolutional Neural Network ($RCNN$). Instead of following the traditional approach (as explained in Figure 4.14), DeepVO directly performs pose estimation using the $RCNN$. This is a supervised learning approach that aims to train a deep neural network model on labeled datasets to construct a function that maps consecutive images to motion transformations, rather than exploiting the geometric structures of images as in conventional VO algorithms [139].

Testing was conducted against the well-known KITTI dataset for mobile robotics and autonomous driving, revealing competitive performance of DeepVO compared to existing methods.

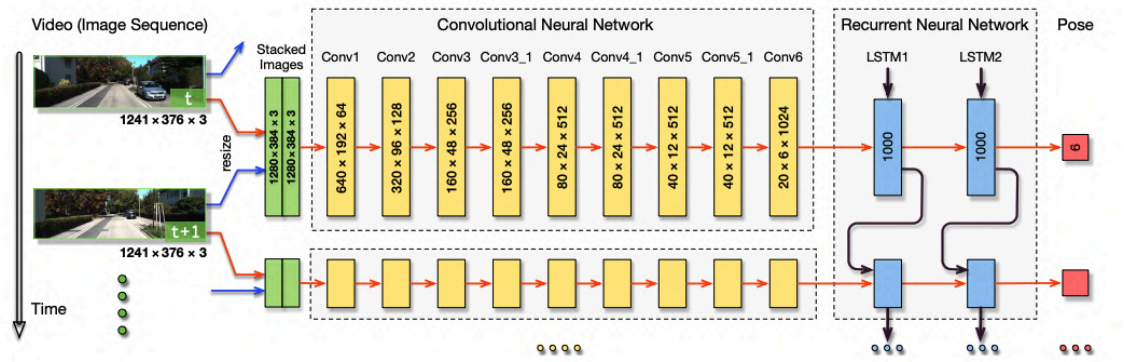



**Fig. 4.21:** DeepVO Architecture for Visual Odometry, inferring the pose directly from the $RCNN$ [138].

Similar works using $CNN$s include SalientDSO, proposed by Liang *et al.* [140]. This algorithm builds on the principles of Direct Sparse Odometry ($DSO$), a type of visual odometry that estimates a camera's motion by directly minimizing photometric error across sparse points in an image, rather than relying on feature extraction and matching. Both SalientDSO and DeepVO were analyzed by Cartolano and Colombini [141] against different datasets to assess their performance, although they yielded poor results, apparently due to insufficient image data.

Deep learning implementations in $VIO$ represent a cutting-edge area of research that has only recently gained traction. However, integrating deep learning with $VIO$ is challenging due to the complexity of combining visual and inertial data streams. Current research has yet to fully address critical issues such as real-time processing, generalization of models, and robustness across a variety of scenarios.

These implementations are often limited to specific datasets, highlighting the need for more comprehensive research. Advancing this field will require the development of novel architectures, more extensive datasets, and rigorous testing in real-world scenarios.

## 4.3 Emerging Trends and Future Directions

### 4.3.1 Current Companies Invested in UAV Navigation

Several companies are currently at the forefront of developing $GPS$-denied navigation solutions for $UAV$s, focusing on technologies that integrate advanced sensors, computer vision, and artificial intelligence, while also providing certified hardware and software that meet regulatory standards.

- **Advanced Navigation** offers powerful $INS$ solutions with $MEMS$-based and Fiber Optic Gyro ($FOG$)-based $IMU$s that run at high frequencies (1000 Hz) to provide fast, accurate, and robust navigation.
- **UAV Navigation** provides both $INS$ and $VNS$ solutions. Their $INS$, such as the POLAR AD-AHRS, achieves drift rates of less than 35 meters per minute. The $VNS$ solution, called VNS01, uses a belly-mounted camera and performs visual odometry plus pattern recognition to reduce error in $GPS$-denied environments. The hardware consumes as little as 5W at 12VDC and can achieve zero drift if it has previously flown over the covered area.
- **Embention** offers autopilot solutions and is currently developing visual-based navigation products for $GPS$-denied environments.
- **Inertial Labs** provides $INS$+$GPS$ solutions as well as Attitude and Heading Reference System ($AHRS$). Their $IMU$s utilize $MEMS$ or $FOG$ technologies, along with Air Data Computer ($ADC$) systems, for operation in $GPS$-denied environments.
- **Honeywell** is investing in complementary navigation technologies, including Vision-aided navigation, Celestial-aided navigation, and magnetic anomaly-aided navigation solutions.
- **BAE Systems** has developed solutions for navigation via Signals of Opportunity ($SoOP$).
- **AeroVironment** recently unveiled the visual-based navigation system PUMA, which uses visual odometry along with a series of sensors ($LRF$, EO/IR sensor) and an onboard computer to process the data.
- **Safran** plans to enter the market by acquiring the company Orolia, which offers products like Talen-X and Skydel.

### 4.3.2 Next-generation Open Source Solutions

At the same time, commercial open-source solutions that combine AI-ready onboard computers with flight controllers are also starting to emerge. Examples include products from Holybro, Airvolute or DroneAI.

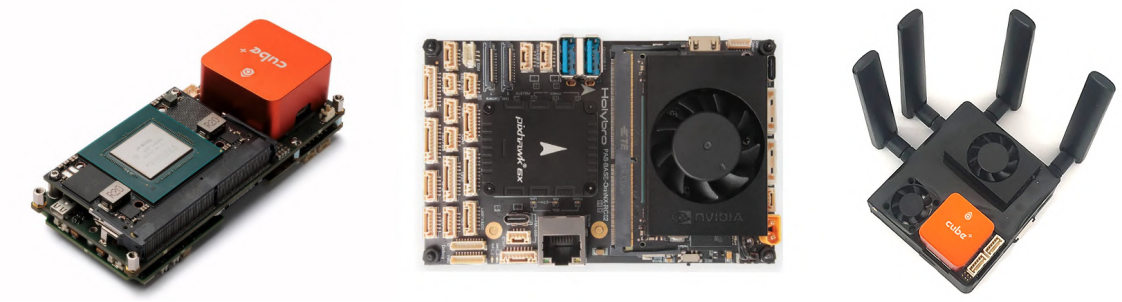


**Fig. 4.22:** Next generation of open-source flight controllers with Jetson boards for embedded AI applications. From left to right: Holybro's Pixhawk Jetson Baseboard, Airvolute's Drone Core 2 model, and DroneAI's DroneHawk AI5G.

### 4.3.3 Event Cameras

**Event cameras** are revolutionizing the computer vision landscape by offering a novel approach to capturing dynamic scenes with unprecedented precision.

Unlike traditional frame-based cameras, which capture images at fixed intervals, event cameras are frame-free and detect changes in the scene at a much finer resolution by recording pixel-level changes asynchronously. This allows for low-bandwidth, low-computational resource requirements, and real-time performance. Event cameras outperform conventional cameras in high-speed scenarios and low-light conditions. The work by Scaramuzza and Fraundorfer [106] and Mahlknecht *et al.* [142] with the Robotics and Perception Group at ETH Zurich is among the most promising research in this field in Europe, with patents already filed for event camera implementations in obstacle avoidance and navigation [143].

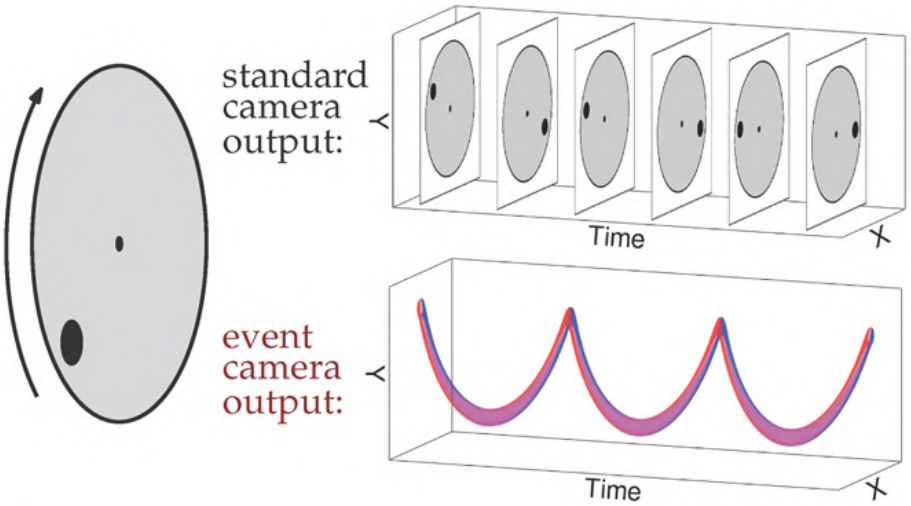



**Fig. 4.23:** Comparison between a frame-free event camera and a conventional frame-based camera for capturing a black dot on a rotating disk.

### 4.3.4 Quantum Navigation

A new and emerging field of research is the use of quantum sensor technology in conjunction with inertial navigation to enhance the accuracy of measurements compared to traditional dead-reckoning methods [144]. The first inertial navigation system of this type is currently under development by *NASA* for upcoming Moon and Mars missions.

### 4.3.5 Collaborative Navigation

Aligned with the specifications for sixth-generation fighter aircraft, the Future Combat Air System (*FCAS*) program is expected to support swarm technology and collaborative navigation.

As Neuhauser [145] explained in a *NATO* technical report, using relative sensor measurements to assist and synchronize the absolute navigation of swarm members can be achieved through inter-swarm relative measurements.

This new paradigm for operating in *GPS*-denied environments is based on the Consistent INS Drift Algorithm (*CIDA*), which is considered the core of collaborative navigation. *CIDA* integrates both relative and absolute navigation information to accurately reconstruct swarm geometry and minimize overall state estimation errors by exchanging state estimates and sensor data among swarm members. This approach ensures that if at least one swarm member has access to an absolute drift-free navigation system, the entire swarm benefits from it.

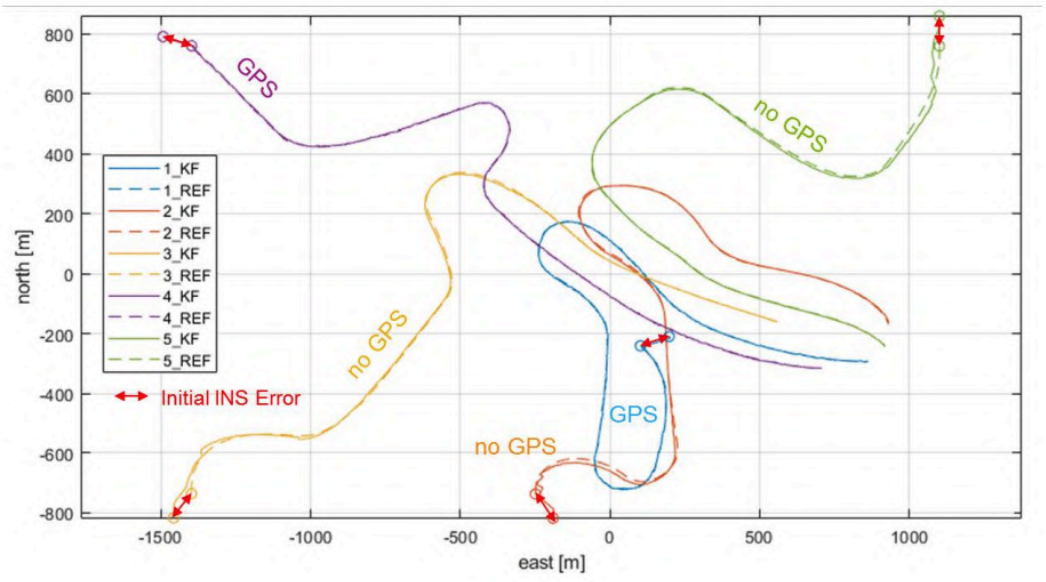



**Fig. 4.24:** Swarm Navigation Simulation. Despite an initial *INS* position error (double arrow red line), the aircraft forming the swarm reduce this error after joining a fixed formation at 50 seconds through a collaborative approach [145].

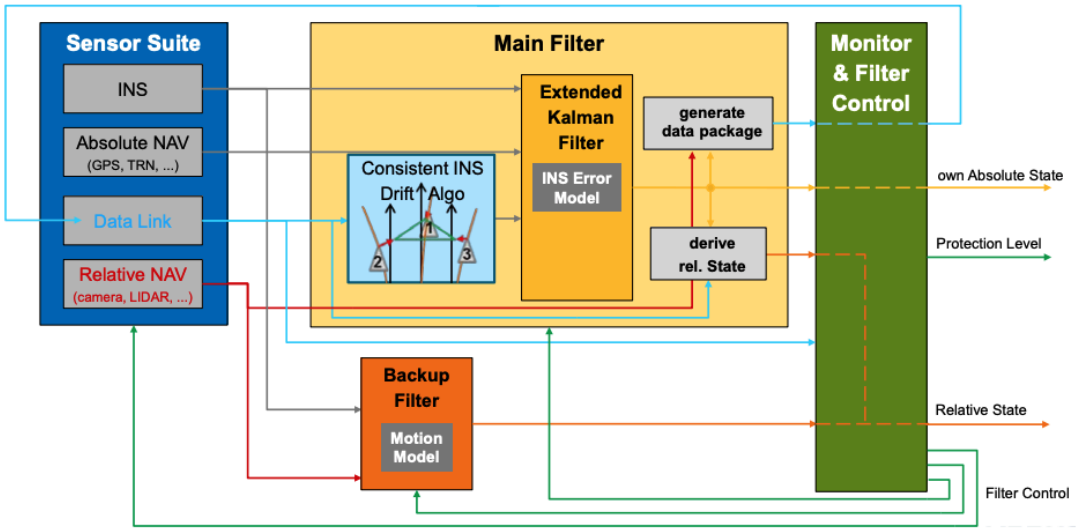



**Fig. 4.25:** Proposed collaborative navigation architecture for the $FCAS$ swarm members [145].

Figure 4.25 illustrates the proposed collaborative navigation architecture for the $FCAS$ swarm members. Data fusion is expected to merge all sensory information using an $EKF$, coupled with the $CIDA$ algorithm.

A similar concept has been proposed by Polizzi *et al.* [146] for decentralized collaborative state estimation among drones, specifically targeted at the next Mars mission aerial robots. In this case, multi-agent localization allows each drone to fly independently while exchanging data with the others when possible, thereby refining its state estimation. This approach is based on the use of Vector of Locally Aggregated Descriptors ($VLAD$) descriptors, a technique that summarizes local image features into a single, fixed-length vector for efficient image retrieval and recognition. Results indicated a 46% improvement in trajectory estimation compared to $VIO$-only approaches while significantly reducing communication data flow between agents.

Previous research in this field included collaborative approaches but was inefficient in terms of communication [147, 148, 149, 150, 151].

# 5 Methodology

This chapter covers the methodology followed to achieve the objectives proposed at the beginning of this thesis. As mentioned in Section 1.5, the approach selected for this work is based on the **V-model methodology** [152], a well-established framework in the aerospace industry that is effective for projects requiring rigorous validation and verification processes.

In the initial phase of the project, a thorough analysis of the state of the art was conducted, as detailed in Section 4. This analysis was essential in identifying current trends, technologies, and practices employed by researchers working on alternative navigation systems. By understanding the latest advancements and challenges in the field, informed conclusions were drawn to guide the direction of the proposed solution. Additionally, the constraints defined by the regulatory framework in Section 2 were carefully considered, ensuring that the proposed solution would be not only innovative but also compliant with industry standards and legal requirements.

Figure 5.1 illustrates the various stages of the V-model as applied in this thesis. This will be covered in the following sections in this chapter.

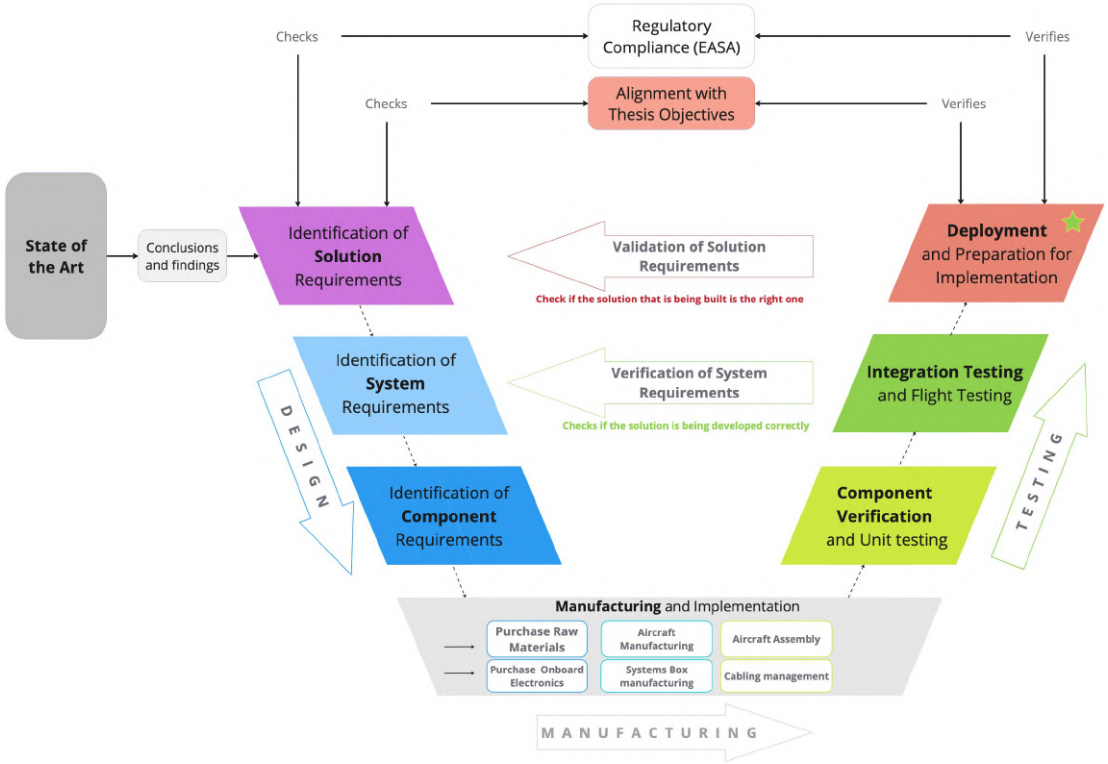



**Fig. 5.1:** Methodology diagram based on the V-model development life cycle.

### 5.0.1 Conclusions from the State of the Art

The selected solution should result from an identified research opportunity that has the potential for further study and investigation. After carefully reviewing the literature presented in the $SOTA$, the following conclusions were drawn:

- Most of the reviewed literature involves testing with $VTOL$ platforms (e.g., quadcopters), rather than fixed-wing or hybrid $UAV$s. From a market perspective, $UAV$s with aerodynamic surfaces are more efficient for long-distance flights, as they offer better range and endurance. They can also carry more payload. Therefore, a research opportunity is identified: the proposed solution should focus on fixed-wing or hybrid aircraft capable of long-range flight.

- Visual-Inertial Odometry ($VIO$) solutions seem to be the most suitable approach for outdoor flights. Visual sensors are lightweight and affordable, and sensor fusion techniques allow for the combination of different mechanisms that can lead to better stability and accuracy in navigation, enhancing overall flight performance in various environmental conditions.

- Deep Learning for $VIO$ with $CNN$s shows promising results. The proposed solution should include the handling of data pipelines with high visual content.

- Monocular cameras are better suited for outdoor flight than stereo cameras because they are lighter, simpler, and avoid the complications associated with the stereo baseline issue. While a larger baseline improves depth accuracy for distant objects, in stereo cameras, this distance is fixed, which worsens depth accuracy as objects move farther from the camera.

- Modularity of the solution is fundamental, as components are upgraded on a yearly basis.

- The solution must rely primarily on $COTS$ components.

- Redundancy must be considered in the case of component failure.

- Having more than one $IMU$ in a $UAV$ improves reliability and accuracy by averaging out noise and errors from individual sensors.

## 5.1 Solution Requirements

Based on a careful analysis of the conclusions from the *SOTA* and the objectives outlined in Section 1.3, the following requirements for the proposed solution can be established. These requirements are designed to ensure clear and precise specifications with no ambiguity.

1. The system must use *COTS* components and open-source software for all applicable parts of the design unless a specific requirement justifies the use of proprietary components.

2. The system shall include a modular testing framework that allows future researchers to easily integrate and evaluate various visual and visual-inertial navigation techniques.

3. The system must provide a feature that enables users to compare *PNT* data against a known ground truth reference. This feature should allow for quantitative and accurate evaluation of the drone's pose estimation.

4. The system must process data in real-time and relay it to a ground user for monitoring.

5. The system must have a mechanism for logging the data it generates for later analysis.

6. The system should demonstrate stable flight capability, with or without the involved electronics, ensuring that the center of mass remains unchanged across different scenarios.

7. Redundancy in power management is required to ensure system reliability in the event of a main battery failure.

8. The system should be able to incorporate fail-safe mechanisms and emergency protocols to handle unexpected failures or critical situations safely.

9. The system must be scalable, allowing for future upgrades and the integration of additional sensors or components without significant redesign.

10. The system must comply with relevant industry standards and regulations established by the regulatory framework described in Section 2.

11. The system must support standard communication protocols for data transmission and integration with other systems or components.

## 5.2 System Requirements

The following requirements describe the specifications that the system must comply with:

1. The aerial platform should be a **fixed-wing or hybrid aircraft** to enable testing of alternative navigation solutions in high-speed environments.

2. The aerial platform shall comply with $EASA$ regulations for the **Open Category**, with maximum limits set at 25 kg of $MTOW$ and 3 meters of wingspan. This compliance facilitates faster testing by eliminating the need for complex authorizations. Flight tests can be conducted after obtaining the A1/A3 certificate for drone piloting.

3. The aircraft must be equipped to test visual-inertial odometry-based algorithms and should therefore include the following **necessary hardware**:

    a) A small, lightweight monocular camera.

    b) An open-source flight controller capable of performing autonomous missions. It must also include redundant power inputs and sufficient I/O ports to connect peripherals.

    c) An onboard computer capable of handling data-intensive workflows, particularly for visual imagery. The device should be lightweight and have low power consumption.

    d) Multiple $IMU$s for redundancy, including accelerometers, gyroscopes, and magnetometers.

    e) Multiple batteries for redundancy.

    f) A communication module for establishing bidirectional data transfer with a ground control station.

    g) A GPS module with $DGPS$ ($RTK$) capability for testing solutions against centimeter-level accuracy ground truth.

    h) A differential pressure sensor to measure free-stream velocity and support the construction of an Air Data System ($ADS$).

    i) Servo motors for actuating the aircraft's control surfaces.

    j) An Electronic Speed Controller ($ESC$) to interface with the electric motor.

4. The system should also accommodate integration with $LiDAR$, $LRF$ or other sensors for further testing.

Overall, a high-level architecture of the system is displayed in Figure 5.2.

### 5.2.1 High-Level Architecture

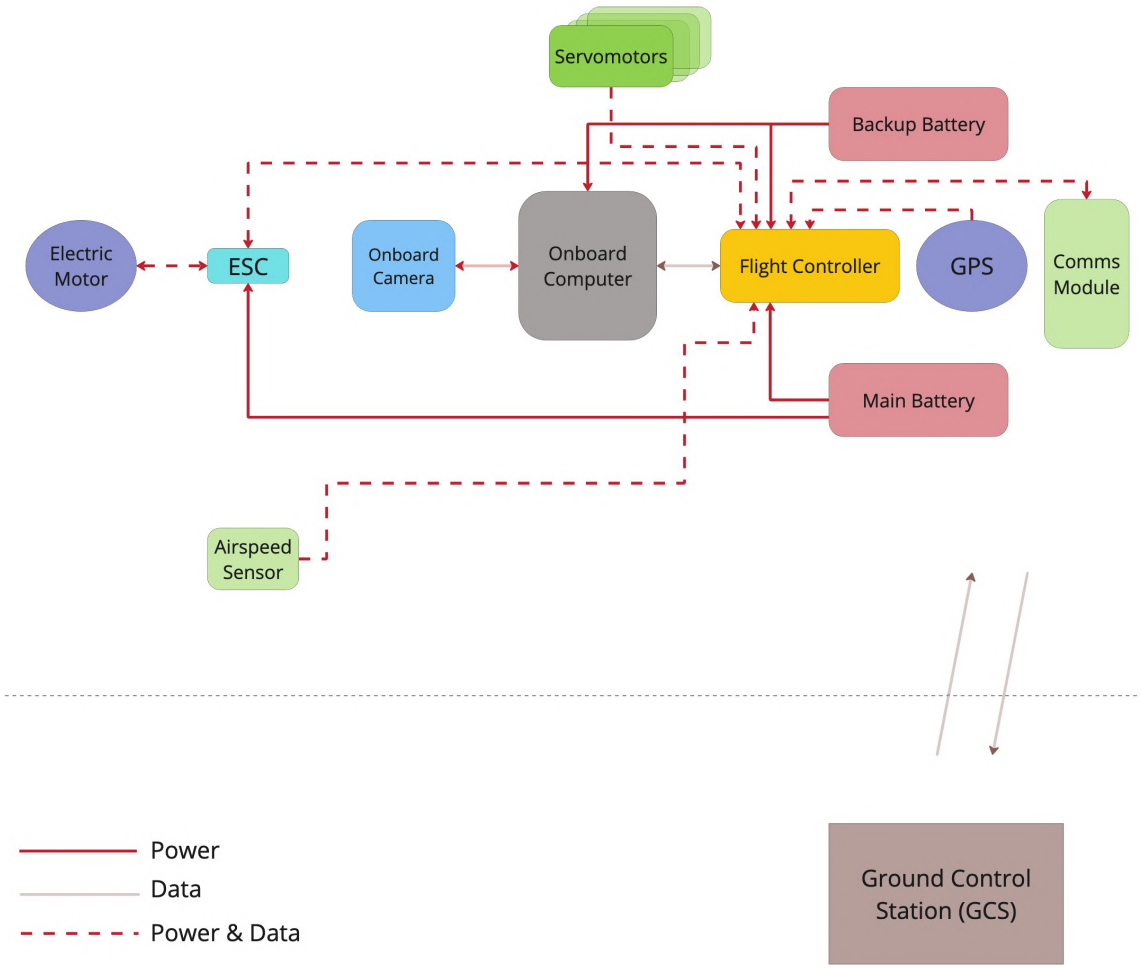



**Fig. 5.2:** High-level architecture of the proposed solution.

## 5.3 Component Requirements

The following requirements are set for each individual component to achieve the proposed system requirements.

### 5.3.1 Onboard Systems Selection

The selection of hardware and software is constrained and guided by the requirements described in Section 5.1. Therefore, a careful analysis of available commercial hardware and software is presented to justify the selection.

#### 5.3.1.1 Flight Controller

The flight controller is a critical component of the autopilot system. It is responsible for autonomously managing and stabilizing the flight of a $UAV$ by processing sensor data and executing control commands to maintain stable flight, execute missions, and ensure safe operation.

The flight controller requires both **hardware** and **software** components. The selected hardware/software solution must meet the following requirements:

1. **Autonomous Flight Operations**
    - Must support autonomous flight operations and mission planning, including waypoint navigation and automated flight path execution.
2. **Sensor Integration**
    - Should include multiple vibration-isolated and temperature-controlled $IMU$s.
    - Must integrate with a variety of sensors (dynamic pressure sensors, $GPS$, $LiDAR$) and allow connection to a peripheral onboard computer.
3. **Data Handling**
    - Should be capable of real-time data processing with minimal latency to ensure timely response to changing flight conditions and commands.
    - Needs to provide various communication interfaces, including serial ports (I2C), CAN bus, and telemetry.
4. **Safety Features**
    - Must include safety features such as failsafe mechanisms, parachute, emergency landing protocols, and redundancy systems.

5. **Compatibility and Customization**

   - Should allow for customization and configuration of flight parameters and control settings to adapt to different types of $UAV$s and mission requirements.

   - Should be compatible with commonly used flight stack frameworks and software, allowing for easy integration into existing systems.

6. **Power Management**

   - Must have robust power management capabilities and dual battery input.

Common available hardware solutions can be categorized as follows [153]:

| Category | Platform | I/O Interfaces |
|---|---|---|
| **FPGA-based** | Phoenix Pro | High-speed GPIO, UART, SPI, I2C, CAN |
| | OcPoc | GPIO, UART, SPI, I2C, CAN |
| **ARM-based** | Pixhawk (PX4) | PWM, UART, SPI, I2C, CAN, ADC |
| | Pixhawk 2 | PWM, UART, SPI, I2C, CAN, ADC |
| | Paparazzi | PWM, UART, SPI, I2C |
| | CC3D & Atom | PWM, UART, I2C |
| **ATM-based** | APM (Ardupilot Mega) | PWM, UART, I2C, SPI |
| | FlyMaple | PWM, UART, I2C |
| **Raspberry Pi-based** | Erle-Brain 3 | GPIO, UART, SPI, I2C, HDMI, USB |

Table 5.1: Comparison of Autopilot Hardware Platforms.

Firmware options can be categorized as follows:

| Usage Extent | Platform | Key Features |
|---|---|---|
| **Widely Used** | PX4 | BSD license, fully featured autopilot |
| | ArduPilot | GPL license, fully featured autopilot |
| | Betaflight | Focused on FPV racing |
| **Moderately Used** | INAV | Forked from Betaflight, added waypoint support |
| | Cleanflight | Merged with Betaflight |
| **Less Common** | Paparazzi | Research-focused, hardware-dependent |
| | LibrePilot | Not as feature-rich as ArduPilot |
| **Others** | Dronin | Focused on FPV racing |
| | Multiwii | Original autopilot for quadcopters, not active |

Table 5.2: Summary of Open-Source UAV Software Platforms by Usage Extent.

To comply with the requirements, the **Pixhawk 2 / ArduPilot** combination was selected.

Pixhawk 2 was chosen over an Field Programmable Gate Array ($FPGA$)-based solution due to its quicker time-to-market, enabling faster prototype development and reducing the time and effort required for integration and testing. Raspberry Pi-based solutions were discarded despite their abundant I/O interfaces due to constraints with the associated hardware. ArduPilot was selected due to its alignment with the requirements and its GPL license, which enforces strong copyleft requirements and ensures that all derivative works remain open source.

Several options are available in this category:

- Cube Orange ADS-B
- Pixhawk 6X/C
- CUAV X7
- ARK V6

**Fig. 5.3:** Pixhawk-based autopilot solutions compatible with ArduPilot.

The **Cube Orange ADS-B** was selected due to its lower price compared to the CUAV X7, while maintaining comparable specifications to the Pixhawk 6X (including triple redundant IMU and dual power input) and featuring an ADS-B antenna receiver, which is useful for monitoring nearby aircraft. The ARK V6 model was discarded due to insufficient connectivity with peripherals.

#### 5.3.1.2 Onboard Computer

The onboard computer is responsible for managing all operations related to the data being collected. It must handle data-heavy workflows, particularly those involving image processing.

The following requirements are specified for an ideal onboard computer:

1. **Processing Power**
    - CPU: Quad-core ARM or x86 processor.
    - GPU or dedicated module for image processing.
    - Minimum of 4 GB RAM and 32 GB storage.
2. **Connectivity**
    - MIPI-CSI interfaces for high-resolution camera integration.
    - Wi-Fi connectivity, or support for adding a Wi-Fi module*.
    - USB 3.0, HDMI, Ethernet.
3. **Power Requirements**
    - Operates below 11.1 V (3S LiPo battery).
    - Optimized for low power consumption.
4. **Image Processing**
    - Compatible with TensorFlow, PyTorch, OpenCV.
    - Handles high data throughput from cameras.
5. **Form Factor and Durability**
    - Small and compact, suitable for space-constrained environments.
6. **Future Upgrades**
    - Supports future hardware upgrades.
    - Allows integration of additional sensors.

Table 5.3 presents a technical comparison of four possible onboard computers: Raspberry Pi 5, Arduino Mega 2560, and NVIDIA's Jetson modules.

| Feature | Raspberry Pi 5 | Arduino Mega 2560 | Jetson Nano 4GB | Jetson Orin Nano 4GB |
|---|---|---|---|---|
| **CPU Cores** | 4 | 1 | 4 | 8 |
| **CPU Architecture** | ARM Cortex-A76 | 8-bit AVR | ARM Cortex-A57 | ARM Cortex-A78 |
| **CPU Clock Speed** | 1.8 GHz | 16 MHz | 1.43 GHz | 1.5 GHz |
| **GPU** | Broadcom VideoCore VII | None | 128-core Maxwell GPU | 512-core Ampere GPU |
| **Memory (RAM)** | 8 GB LPDDR5 | 256 KB SRAM | 4 GB LPDDR4 | 4 GB LPDDR5 |
| **Storage** | MicroSD, SSD/HDD | MicroSD | MicroSD | NVMe SSD |
| **Ethernet** | Yes | No | Yes | Yes |
| **Wi-Fi** | Yes | No | Yes* | Yes* |
| **Bluetooth** | Yes | No | No | No |
| **Voltage** | 5V | 7-12V | 5V | 7-10V |
| **Connector Type** | USB-C | Barrel jack | Barrel jack | Barrel jack |
| **MIPI-CSI** | 2 | No | 4 | 4 |
| **Price** | 60-80 € | 40-50 € | 100-130 € | 200-300 € |

Table 5.3: Technical comparison of onboard computers.

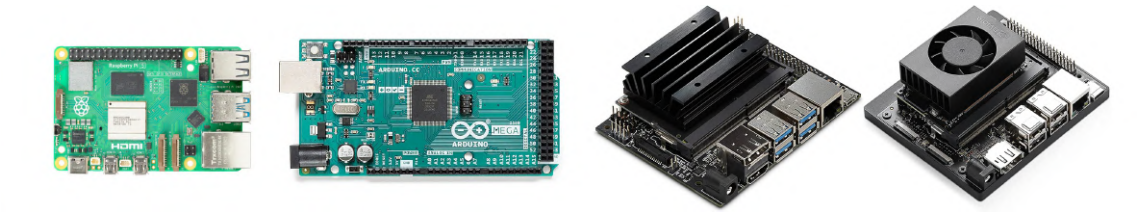


**Fig. 5.4:** Onboard computers. From left to right: Raspberry Pi 5, Arduino Mega 2560, NVIDIA Jetson Nano, and NVIDIA Jetson Orin Nano.

The **Jetson Nano** was chosen for its GPU-accelerated capabilities and cost-effectiveness compared to other NVIDIA boards. It excels in handling machine learning tasks, particularly in computer vision, while consuming low battery power.

#### 5.3.1.3 GPS Module

Although the proposed solution is focused on GPS-denied environments, including a GPS module is crucial for comparing the performance of the proposed algorithms against a known ground truth during testing. Therefore, the GPS module must meet the following requirements:

1. **RTK Support**
   - The GPS module should support *RTK* technology, enabling high-precision positioning with centimeter-level accuracy.
2. **Multiple Constellation Support**
   - The GPS module must be capable of receiving signals from multiple constellations (GPS, GLONASS, Beidou, and Galileo).
3. **Interface Compatibility**
   - The GPS module should be compatible with the Pixhawk autopilot through common interfaces like CAN bus, UART, or I2C.
4. **Integrated Compass**
   - The GPS module should include an integrated magnetometer (compass) to provide orientation data.

Analyzing the available products compatible with the selected autopilot, CubePilot's Here GPS modules are compared in Table 5.4.

| Feature | Here 2 (M8N) | Here 3 (M8P) | Here 4 (NEO-F9P) |
|---|---|---|---|
| **Constellation Coverage** | 4 | 4 | 4 |
| **RTK Support** | No | Yes | Yes |
| **Connection Protocol** | I2C/CAN | CAN | CAN |
| **Frequency** | 10 Hz | 8 Hz | 20 Hz |
| **Integrated Compass** | Yes | Yes | Yes |
| **Built-in IMU** | Yes | Yes | Yes |
| **DroneID** | No | No | Yes |
| **Price** | 70-120 € | 130-180 € | 200-250 € |

Table 5.4: Comparison of GPS Modules Compatible with Orange Cube ADS-B.

The **Here 3 GPS** is selected due to its *RTK* compatibility, which is critical for establishing a ground truth in future testing.

#### 5.3.1.4 Onboard camera

The onboard camera is a crucial component within the visual-odometry framework of the drone. It significantly impacts operational efficacy, especially at high speeds. The camera must capture high-resolution visual data, which is essential for precise motion and positional estimates. To ensure the camera meets performance standards, the following requirements are specified:

1. **Sensor Type**
    - The camera must be equipped with either a CCD or CMOS sensor to ensure high image quality and performance.
2. **Frame Rate and Resolution**
    - The camera should be capable of outputting video at a minimum of 100 frames per second (fps) at a resolution of 1080p to support high-speed image capture.
3. **Connectivity**
    - The camera must feature a MIPI-CSI interface for high-speed data transfer and seamless connectivity with the onboard computer.
4. **UVC Compliance**
    - The camera should comply with USB Video Class (UVC) standards to ensure compatibility with a wide range of systems and simplify integration if other interconnection methods prove unsuitable.
5. **Compatibility**
    - The camera must be compatible with the NVIDIA Jetson Nano platform, ensuring smooth operation and integration with the processing unit.
6. **Resolution**
    - The camera must have a minimum resolution of 5 megapixels (MP) to provide high image detail and clarity.

To address size and weight restrictions and ensure compatibility with the onboard computer (Jetson Nano), various camera modules have been compared in terms of their specifications:

| Feature | Arducam IMX219 | Arducam IMX477 | GoPro Hero 7 | eCam50 CUNX |
|---|---|---|---|---|
| **Sensor Size** | 1/4” | 1/2.3” | 1/2.3” | 1/2.5” |
| **Sensor Type** | CMOS | CMOS | CMOS | CMOS |
| **Megapixels** | 8 MP | 12.3 MP | 12 MP | 5 MP |
| **Interface Type** | MIPI-CSI | MIPI-CSI | USB | MIPI-CSI |
| **UVC Compliant** | No | Yes* | No | No |
| **Max Output Frame Rate** | 180 fps (720p) | 240 fps (1080p) | 240 fps (720) | 100 fps (720p) |
| **Max Video Resolution** | 1080p (60 fps) | 4K (60 fps) | 4K (60 fps) | 1944p (28 fps) |
| **Power Requirements** | 1.2V (Digital) | 1.05 V (Digital) | 5 V | 3.3 V |
| **Shutter Type** | Rolling | Rolling | Rolling | Rolling |
| **Field of View** | 62° | 77° | 170° | 60° |
| **Cost** | 45-50 € | 60-80 € | 350-400 € | 90-100 € |

Table 5.5: Comparison of Camera Specifications for Various Models.


**Fig. 5.5:** Camera module comparison. From left to right: Arducam IMX219, Arducam IMX477, GoPro Hero 7, and eCam50 CUNX.

After careful analysis, the **Arducam IMX477** with a 6mm lens was selected due to its higher output frame rate and MIPI-CSI high-speed interface for efficient data transfer. Additionally, this camera has the potential to be UVC compliant if acquired as the Arducam B0280 model, which ensures compatibility across different operating systems without requiring additional drivers.

#### 5.3.1.5 Communication Module

Communication with a Ground Control Station ($GCS$) is required for performing autonomous missions with $UAV$s. This communication layer must comply with a minimum set of requirements to ensure that the $UAV$ can be monitored in real time and that the performance of the alternative navigation system can be tracked as the mission progresses.

1. **Real-Time Monitoring**
    - The system must enable continuous real-time monitoring, allowing the $GCS$ to receive and display the UAV's telemetry data, including position, altitude, speed, and overall system health status.
2. **Bidirectional Communication**
    - The system must support bidirectional communication between the $GCS$ and the $UAV$, allowing for the transmission of control commands and the reception of data.
3. **Mission Capabilities**
    - The system must allow for designing missions and sending remote updates to mission parameters and flight plans from the $GCS$ during $UAV$ operations, ensuring that changes can be implemented with minimal delay.
4. **Fail-Safe Mechanisms**
    - Automatic fail-safe mechanisms must be in place to initiate emergency procedures, such as return-to-home or controlled landing, in the event of communication loss or critical system failures during testing.
5. **Communication Range**
    - The communication link between the $UAV$ and the $GCS$ must be reliable within a range determined by $VLOS$, though it is not limited by it.
6. **Regulatory Compliance**
    - The system must comply with European aviation regulatory standards and the European Telecommunications Standards Institute ($ETSI$), which strictly establishes the frequency band to be used in the European Union: 868 MHz ISM.

Several choices are available for open-source *GCS* compatible with the Ardupilot firmware.

| Feature | Mission Planner | QGroundControl | APM Planner 2.0 | MAVProxy |
|---|---|---|---|---|
| **Supported Platforms** | Windows | All | All | All |
| **Primary User Base** | ArduPilot | PX4 | ArduPilot | ArduPilot |
| **Interface** | GUI | GUI | GUI | Command-line |
| **Ease of Use** | Moderate | User-friendly | Moderate | Complex |

Table 5.6: Comparison of Ground Control Station Software

Based on the similarity in available features between the given options, **Mission Planner** is chosen for its built-in GUI and rich parameter tuning capabilities. Nevertheless, any of the other options would have satisfied the required criteria.

On the other hand, to achieve bilateral information flow from the *UAV* to the *GCS* and vice versa, several options are available.

| Feature | Cubepilot Herelink | RFD868x | CUAV Air Link |
|---|---|---|---|
| **Max Range** | 12-20 km | 40 km | Unlimited |
| **Data Transmission** | 2.4 GHz HD video, telemetry | 868 MHz, telemetry | HD video, telemetry |
| **Protocol** | RF and Wi-Fi | RF | 4G LTE |
| **Installation Complexity** | Low | Medium | Low |
| **Price** | 1,200 € | 300 € | 900 € |

Table 5.7: Comparison of Communication Modules


**Fig. 5.6:** Comparison of Communication Modules. From left to right: CubePilot Herelink, RFD868x, and CUAV 4G Air Link unit

Based on the reviewed options, the **RFD868x** is selected due to its lower price compared to the alternatives and its sufficient data-transmission features that meet the established requirements. The open-source radio control protocol ELRS is chosen for its low latency and long-range performance, compared to other options like FrSky or Crossfire.

#### 5.3.1.6 Power Distribution

Proper power distribution is critical for a $UAV$. The following requirements must be met for the proposed power architecture:

1. **Power Source**
    - The $UAV$ must use electrical power generated from LiPo (Lithium Polymer) batteries.
2. **Power Monitoring**
    - The system must include a power sensor to monitor the performance of each battery cell in real time, with the data transmitted to the $GCS$ for continuous performance assessment.
3. **Connector Type**
    - The power connectors used in the $UAV$ must be XT60 or XT90 to ensure secure and reliable electrical connections.
4. **Battery Configuration**
    - The $UAV$ batteries must not exceed the 4S 14.8V configuration.
5. **Power Redundancy**
    - The system should count with redundant power in the case that the main battery fails. Therefore both batteries should share the same nominal voltage.
6. **Voltage Regulation**
    - Each component of the $UAV$ must receive the appropriate power according to its specifications. If a component requires a lower voltage than what the battery provides, an Universal Battery Elimination Circuit ($UBEC$) must be installed to step down the voltage and provide the correct power level to that component.

#### 5.3.1.7 Other Peripherals

The remaining components, which vary depending on the choice of aerial platform, are described here along with their minimum requirements.

1. **Servomotors**

    - Servomotors must use either Pulse Width Modulation ($PWM$) or Controller Area Network ($CAN$) protocols for communication.

    - All servomotors must be equipped with metallic gears to ensure durability and longevity in operation.

2. **Differential Pressure Sensor**

    - The Differential Pressure Sensor must use an I2C, $CAN$ or Universal Asynchronous Receiver/Transmitter ($UART$) interface and must not exceed a power draw of 5.5V DC.

    - The sensor must be fully compatible with the selected flight controller.

    - It must provide two ports to measure both dynamic and static pressure.

3. **Electronic Speed Controller** ($ESC$)

    - The $ESC$ must be compatible with the selected battery configuration, ensuring it can handle the amperage draw without overheating or failure.

    - It must be equipped with an XT60 connector to ensure a reliable connection to the power source.

4. **Motor**

    - The motor must be a brushless electric motor.

    - The motor must be connected to the flight controller via an $ESC$ to enable precise control of the motor's performance.

### 5.3.2 Platform Selection

When selecting the $UAV$ platform, the payload capacity and internal volume are crucial constraints. The following requirements are established to guide the selection process for a fixed-wing aircraft platform:

1. **Manufacturability**
    - The $UAV$ platform must be easily manufacturable, with all necessary Computer Aided Design ($CAD$) files available for editing and customization as needed.
    - The platform must be cost-effective to manufacture, minimizing overall production costs.
    - It must not require components that are difficult to source, ensuring that all materials and parts are readily available.
2. **Available Volume**
    - The platform must provide a minimum internal volume of 2500 $cm^3$ to accommodate necessary components and payloads.
3. **Payload Capacity**
    - The platform must support a minimum payload capacity of 1 kg.
4. **Camera Integration**
    - The design must allow for the integration of different monocular cameras, ensuring flexibility in mission-specific sensor requirements.
5. **Component Allocation**
    - The platform must allow for the secure allocation of all components, either directly within the structure or using appropriate enclosures, to protect the electronics and ensure stable operation.
6. **Modularity**
    - The platform must be modular, allowing for easy assembly and disassembly, enabling customization and upgrades.
    - The modular design must allow the platform to be compacted and fit into a box for easy storage and transport.

Several platforms are available to the public that may meet the proposed requirements.

| Specification | Roadrunner | Stallion | Lark | Titan Cobra |
|---|---|---|---|---|
| **Aircraft Type** | Fixed Wing | Fixed Wing | Fixed Wing | Hybrid VTOL |
| **Wingspan** | 2080 mm | 1340 mm | 1290 mm | 2021 mm |
| **Length** | 1431 mm | 990 mm | 865 mm | 1200 mm |
| **AGW** | 2.3 kg | 1.5 kg | 1.1 kg | 3-4 kg |
| **MTOW** | 5.4 kg | 3 kg | 3 kg | 6 kg |
| **Payload Capacity** | 8250 $cm^3$ | ? | ? | ? |
| **Materials** | Wood + PLA | LW-PLA + PETG | LW-PLA + PETG | LW-PLA + PC |
| **Cost** | Free[7] | 35 € | 30 € | 50 € |
| **Filetype** | CATPart/STEP & DXF | STEP | STEP | STEP |
| **Launch** | Runway | Handheld | Handheld | VTOL |
| **Motor Config.** | Single motor (puller) | Twin motor | Single motor (pusher) | 5 motors |
| **Cruise Speed** | 15 m/s | 18 m/s | 14-16 m/s | 12.5-18 m/s |
| **Modular Design?** | Yes | No | Yes | Yes |
| **Camera Opening?** | Yes | No | No | No |
| **Build Cost** | 250-300 € | 665-1170 € | 600-900 € | 875-1600 € |

Table 5.8: Specification breakdown for different UAV platforms

**Fig. 5.7:** Comparison of UAV platforms. From left to right: Roadrunner, Stallion, Lark, and Titan Cobra models

The **Roadrunner** platform is selected as it meets the volume and weight constraints, has the lowest estimated cost, includes a camera opening, and allows for modifications to the original files if needed.

[7]Files available on request.

### 5.3.3 Onboard Components Compatibility

To ensure that the chosen components can be properly connected, a set of requirements must be established for power and data transmission processes.

#### 5.3.3.1 Power Requirements

Each device requires specific voltage and current settings to operate correctly. The following requirements have been identified to ensure the proper functioning of the components:

1. A backup battery should be installed for redundancy. Its capacity can be equal to or lower than that of the main battery, but the voltage must be the same to ensure compatibility with the modules already being powered.

2. The Orange Cube flight controller operates within a voltage range of 4.1V to 5.7V. A power-down module or UBEC must be placed between the battery and the flight controller.

3. The Orange Cube flight controller should power the servomotors, which require 4.8V.

4. The Orange Cube flight controller should power the differential airspeed sensor, which requires 4-6V.

5. The Orange Cube flight controller should power the radio modem, which requires 5V.

6. The Orange Cube flight controller should power the GPS, which requires 5V.

7. The Jetson Nano onboard computer operates at 5V and 4A. A power-down module or similar device must be placed between the battery and the computer to ensure proper functioning.

8. The Jetson Nano onboard computer should provide 1.05-2.8V to the IMX477 camera through the MIPI-CSI connection, or 5V through the USB port if UVC compatibility is the only option for data transfer between the two components.

9. The *ESC* must draw power for the electric motor directly from the battery, with the exception of a power-down module that acts solely as a bypass.

10. If fans are installed, they will require 5V. This power should be derived from specific power-down modules, UBECs, or the onboard computer itself.

#### 5.3.3.2 Interface Requirements

The following requirements are established for data transfer, considering the specifications of each component. Refer to Figure 5.9 for a more detailed view.

1. The radio modem should be connected via a 6-pin JST-GH to the Orange Cube's TELEM1 UART port for communication with the ground control.

2. The differential pressure sensor should be connected to the Orange Cube's I2C2 port via a 4-pin I2C connector.

3. The Here 3 GPS should be connected to the Orange Cube's CAN1/CAN2 port via its built-in droneCAN cable.

4. The *ESC* and servomotors should be connected using a 3-pin Futaba "J" connector to the Orange Cube's servo header and mapped through the *GCS*.

5. The onboard computer should interface with the Orange Cube flight controller via a USB-to-UART-type HMI cable.

6. The IMX477 camera should communicate with the Jetson Nano through a 22-to-15 pin MIPI-CSI connector.[8]

### 5.3.4 Detailed Architecture

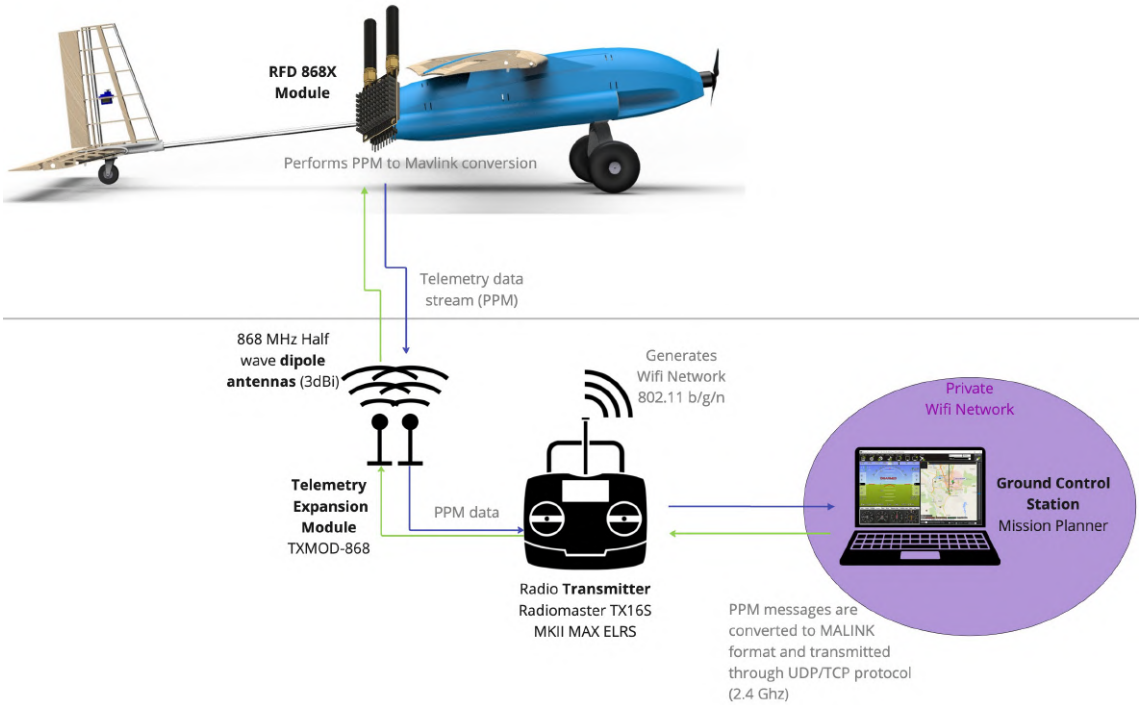



**Fig. 5.8:** Ground Control Station to UAV Communication Architecture.

[8]If a MIPI-CSI connector is not available, data transmission shall be done through USB after installing an Arduino-based MIPI-to-USB adapter on the camera side.

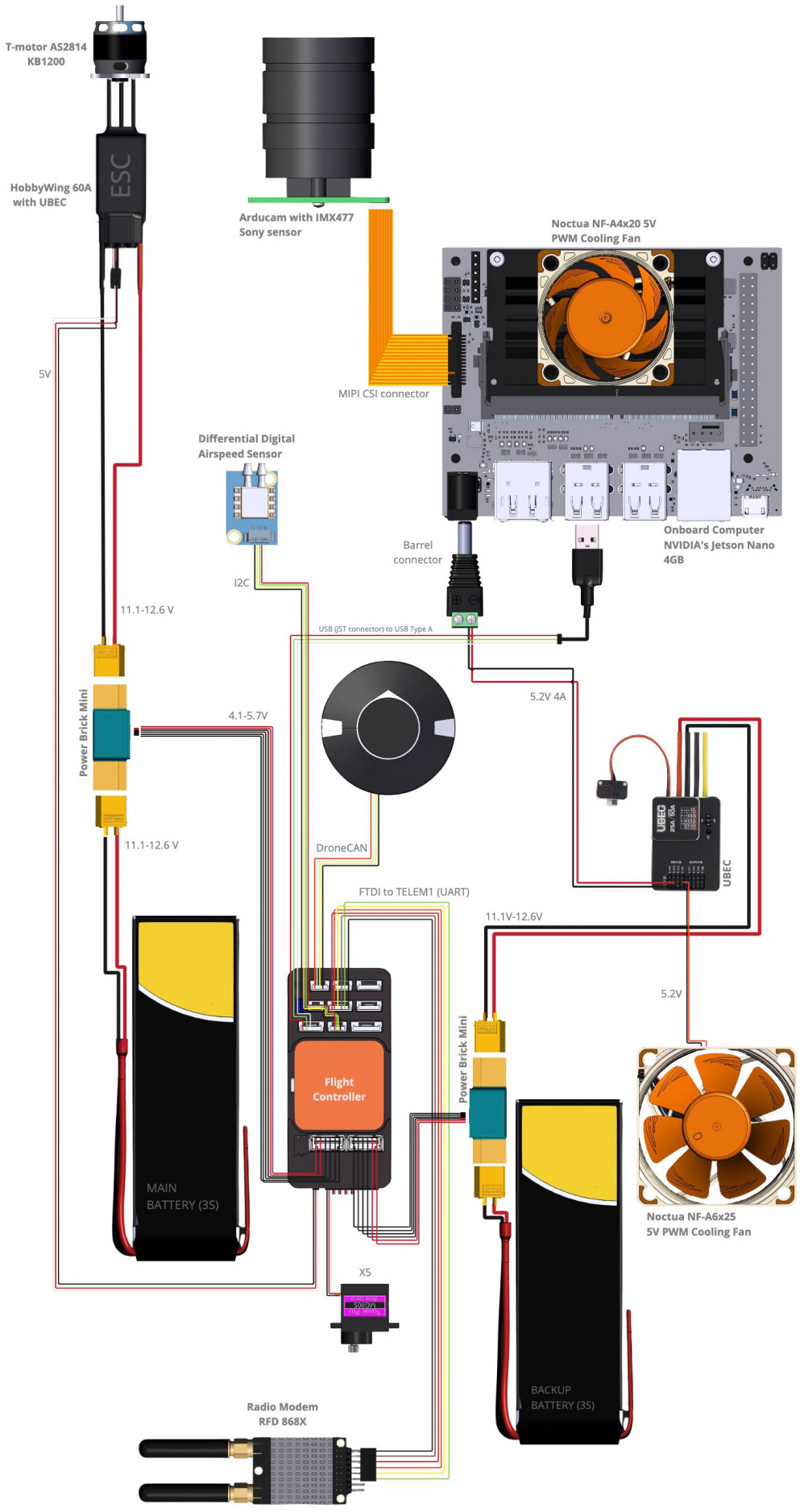



**Fig. 5.9:** Detailed breakdown of the pin-out for each component of the *UAV*'s onboard systems.

## 5.4 Manufacturing and Implementation

This chapter details the processes and tools used to fabricate the different components of the project, as well as the methodologies applied during the design, assembly, and integration stages. A modular design approach was adopted to ensure flexibility and ease of access, allowing for the independent modification or replacement of individual components without the need to redesign the entire system.

To achieve the desired precision and efficiency in manufacturing, a range of specialized software tools were employed throughout the design and fabrication stages. These tools were selected based on their ability to meet the specific requirements of the project, including *CAD*, laser cutting, 3D printing, and material optimization. The tools used are listed below:

- **Autodesk Fusion 360**: Used for *CAD* design, this software enabled detailed modeling and refinement of the enclosure that houses all the electronic devices, ensuring that volume limitations are respected and that the integration of components takes into account the appropriate tolerances.
- **Xtool Creative Space**: This software was employed for the laser cutting of the aircraft's structure, providing precise control over the cutting process to produce high-quality parts from plywood and balsa wood.
- **Bambu Lab Studio**: This slicer software for 3D printing was used to prepare 3D models by converting them into layers and generating the necessary G-code for the 3D printer.

This chapter will also describe the step-by-step procedures followed during the assembly of the components, as well as any challenges encountered and the solutions implemented to address them.

### 5.4.1 Aircraft Manufacturing

The Roadrunner is a 2.1 m wingspan lightweight $UAV$, primarily made of plywood, balsa wood, and 3D-printed parts [154].

The manufacturing of this aircraft began with importing the .dxf files into the laser cutting Computer Aided Manufacturing ($CAM$) software Xtool Creative Space ($XCS$), which was used to cut the ribs and spars that later formed the internal structure of the $UAV$'s wing, fuselage, vertical stabilizer, and horizontal stabilizer.

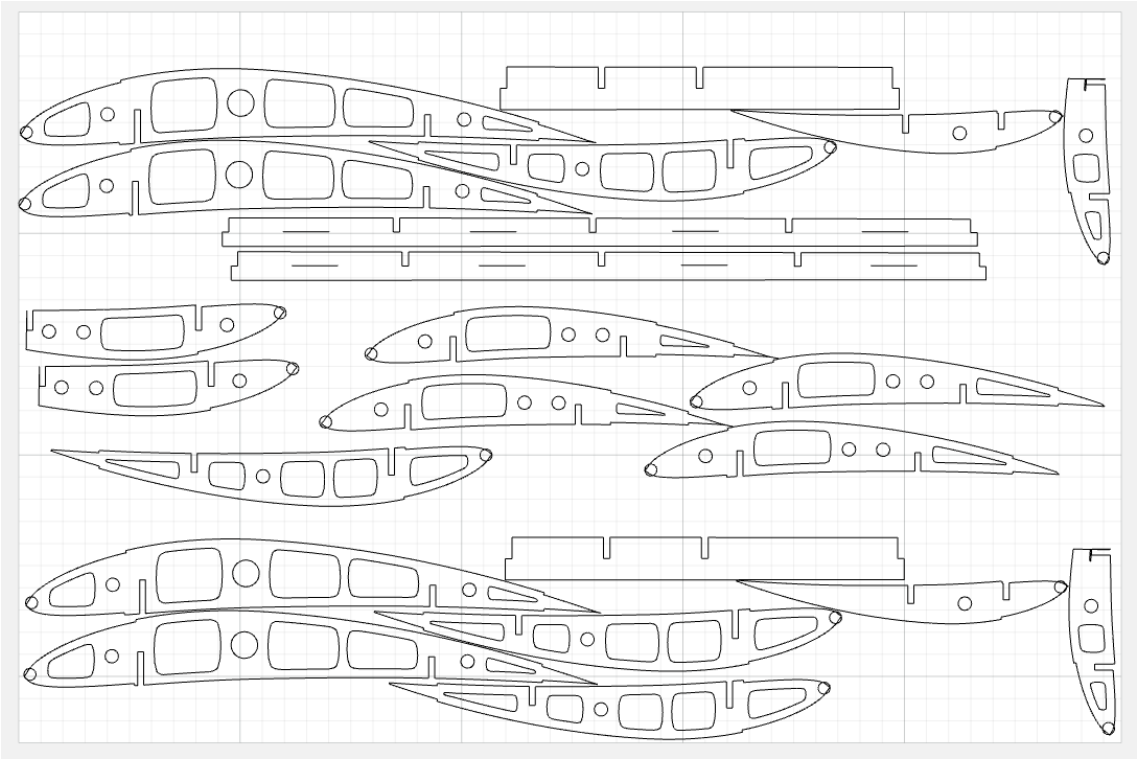
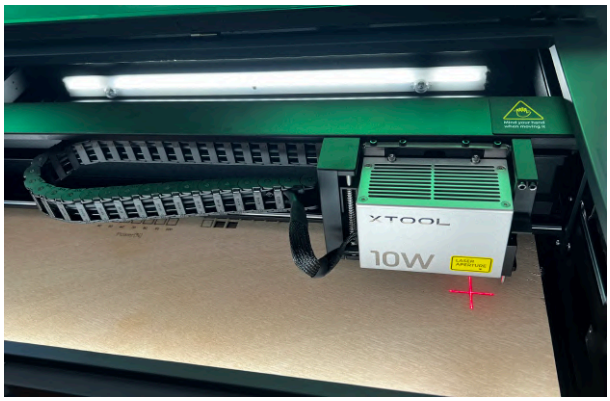


**Fig. 5.10:** Ready-to-cut board inside the Xtool CAM software.

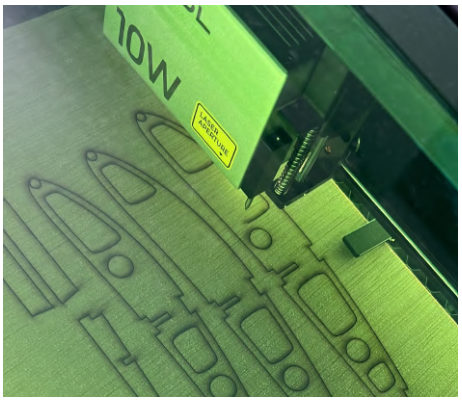
The following settings were then adopted on the 10W Xtool S1 laser cutter, depending on whether aerodynamic surfaces or the fuselage was being cut.

| Component | Material | Power | Feedrate | Passes |
|---|---|---|---|---|
| **Fuselage** | 3 mm Plywood | 100% | 7 mm/s | 1 |
| **Aerodynamic Surfaces** | 3 mm Balsa wood | 100% | 12 mm/s | 1 |

Table 5.9: Laser cutting settings for different UAV components based on material type.



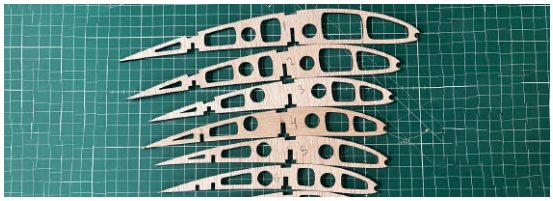

Wood sheet placement prior to cutting



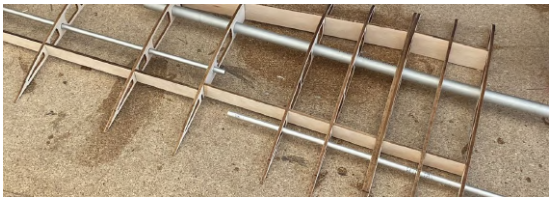

Cutting the wing's internal structure

**Fig. 5.11:** Laser cutting with the Xtool S1.

Once all the pieces were cut, they were laid on a table for assembly, which was performed using liquid adhesive, such as cyanoacrylate. Aluminum spars were introduced during this process to ensure the alignment of the pieces.



Balsa wood ribs prior to assembly



Alignment of ribs with aluminum spar

**Fig. 5.12:** Assembly of the wing's internal rib structure.

The balsa wood sheets were then glued to the leading and trailing edges to provide a surface for the vinyl skin.

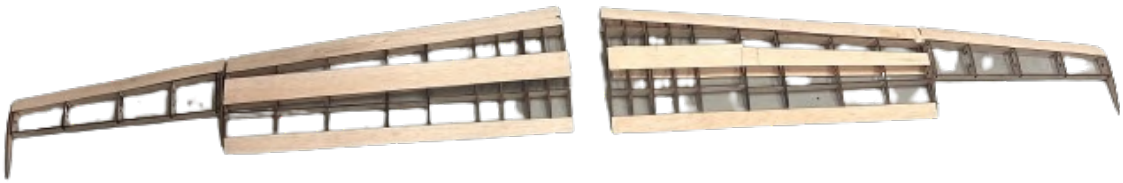


**Fig. 5.13:** Internal wing structure after assembly.

The Oracover vinyl skin was then applied using a hot iron and tensed slowly until the wing acquired the desired aerodynamic shape.

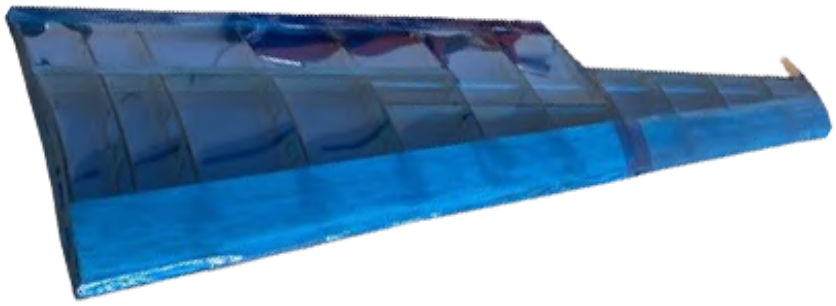


**Fig. 5.14:** Left wing covered with Oracover film.

The same procedure was applied to the vertical stabilizer, horizontal stabilizer, and control surfaces. The assembly of the internal structure of the fuselage was also performed in a similar manner.

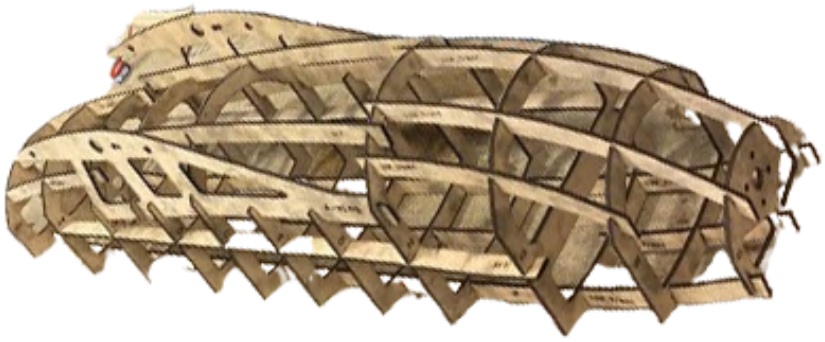
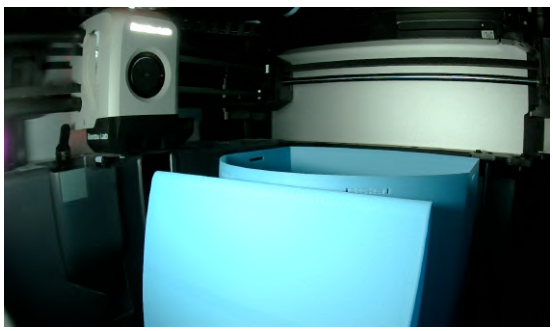
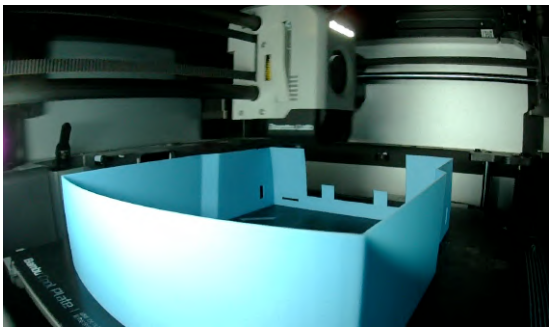


**Fig. 5.15:** Internal plywood structure of the assembled fuselage.

In parallel, the fuselage fairings were 3D printed.



Fuselage fairings during printing



Completed fuselage fairings

**Fig. 5.16:** 3D printing of the fuselage fairings.

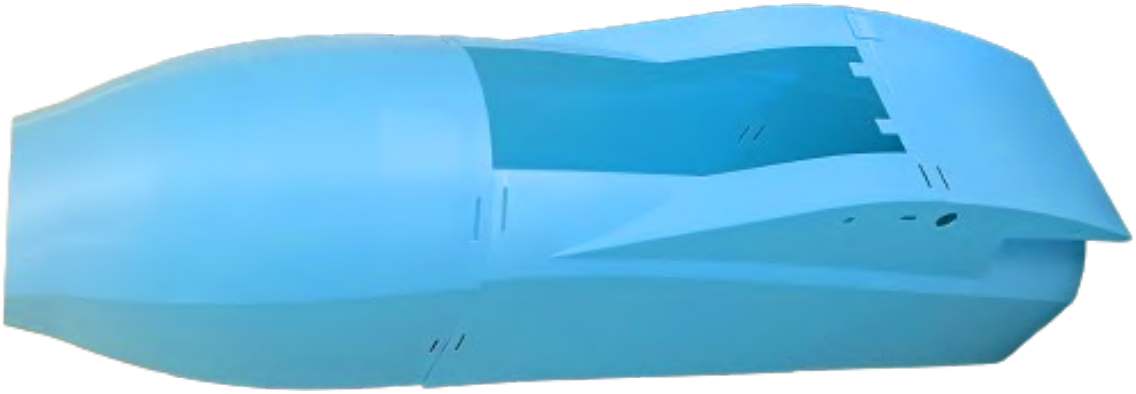


**Fig. 5.17:** Complete fuselage fairing

The remaining 3D-printed pieces included the tail integration piece, which connected the vertical stabilizer with the horizontal stabilizer, the alignment holes for the integration of the 12 mm aluminum tube that connected the tail with the fuselage, and the landing gear structure. The landing gear was also 3D printed but made from polycarbonate (PC) instead of PLA.

A series of holes were then drilled using a Bosch PB40 vertical press drill on the 12 mm tail aluminum spar to securely integrate the horizontal spar of the Horizontal Tail Plane ($HTP$) with the Vertical Tail Plane ($VTP$).

The final aircraft was then ready for the installation of its basic electronics, including servomotors, the electric motor, and $ESC$.

**Fig. 5.18:** Final assembled aircraft.

### 5.4.2 Onboard System Implementation

Once the electronic components were identified and their connections established, as illustrated in Figure 5.9, the acquisition process was initiated. This process involved the procurement of the specified hardware, with the exception of certain components that were provided on loan.[9]

Immediately after receiving a component, its dimensions were verified using a caliper. The Grabcad library was then consulted to check if that specific component was already modeled in *CAD*. This was true for almost every component except for the batteries and the 25A *UBEC*, which were modeled before starting the integration.

Following this, the process of creating a custom enclosure began. The purpose of this enclosure was to fit all the components inside to minimize the occupied volume and provide a solution that could be interchangeable between different aircraft for testing. To ensure easy implementation and reproducibility, additive manufacturing was selected as the manufacturing process for the enclosure.

Initial volume constraints were determined from the Roadrunner master *CAD* and later verified using the manufactured fuselage. These constraints were found to be: 11 x 10 x 31 cm (width x height x depth). The proposed enclosure was then designed, taking into account the size of each component and placing them in a compact arrangement while ensuring adequate airflow through them. Tolerances were also accounted for by enlarging each component's projection by 0.2 mm to ensure a tight fit during assembly.

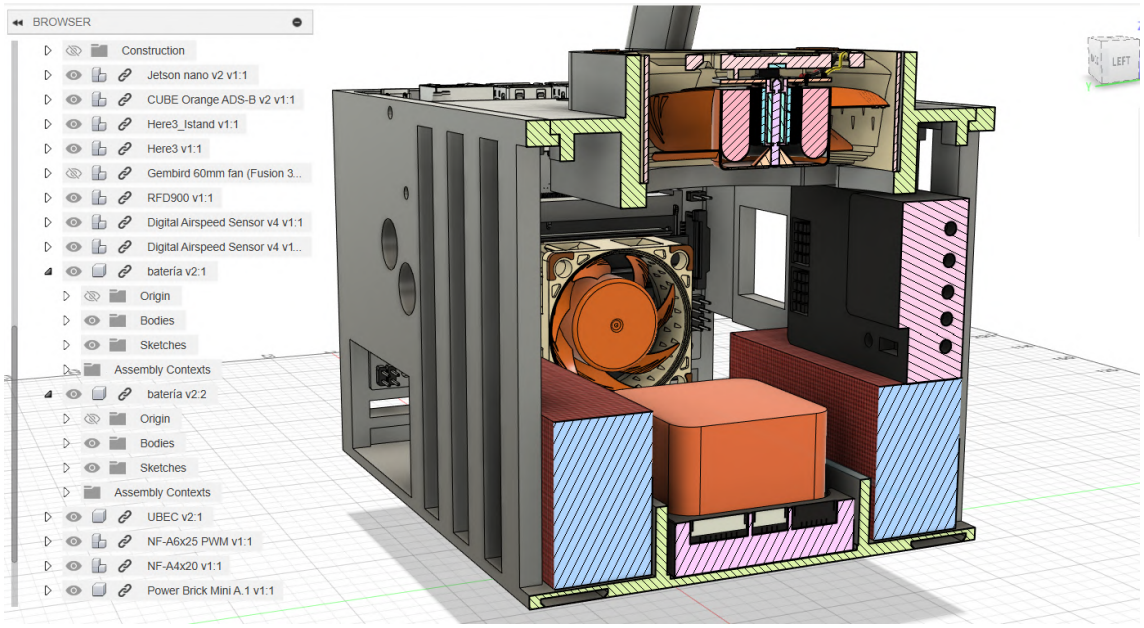


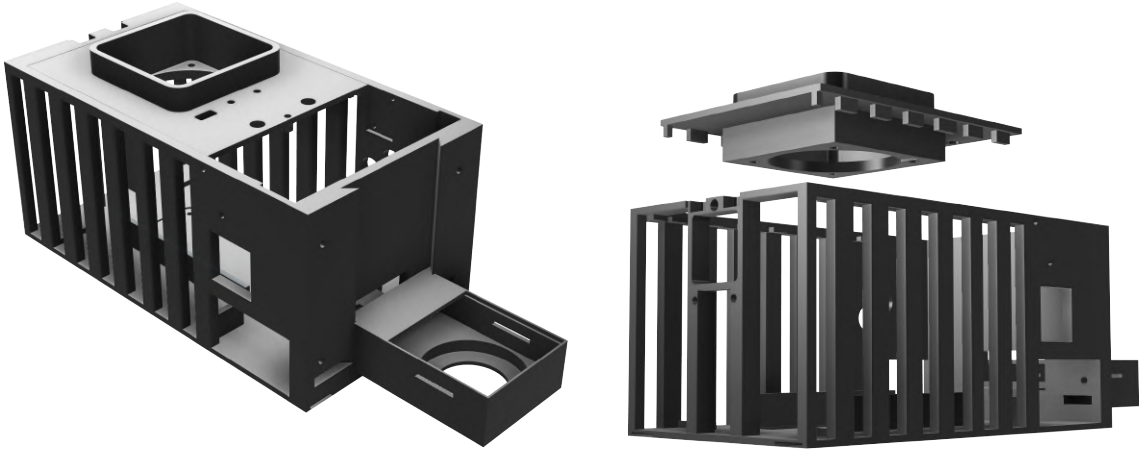

**Fig. 5.19:** Cross-section view of the enclosure in Autodesk Fusion.

[9]Section 7.3 provides a budget breakdown where these loaned components are identified.

Multiple iterations were made to accommodate the components in different configurations until the final distribution was selected. Decisions were based on weight balance and accessibility.

- The Jetson Nano was placed vertically so that the connection hub would be accessible from the top for easier access.
- The Orange Cube flight controller was placed in the middle to ensure that it remained aligned with the longitudinal direction of the aircraft.
- Batteries were positioned symmetrically with respect to the flight controller and secured by partition walls to prevent movement during flight.
- Differential Pressure sensor sockets were placed on both sides of the enclosure to allow the Pitot tube to be installed on either semi-wing.
- Dovetail rails were incorporated at the bottom of the enclosure for connection with the fuselage spars of the Roadrunner test aircraft. This design allows the enclosure to be used with other connectors if the connection breaks.
- The design featuring a removable and replaceable lid was selected to enhance accessibility to the internal components.

Figure 5.20 displays the final design of the enclosure.



Front isometric view Detached lid

**Fig. 5.20:** 3D render of the onboard systems enclosure box.

In keeping with the modular design approach and considering that the camera opening was positioned closer to the aircraft's nose than to the center of gravity, the camera's enclosure was separated from the main box and designed with a quick-swap dovetail joint. This allowed the camera to be exchanged without modifying the rest of the enclosure, as long as the dovetail connection was maintained.

Isometric view Back view

**Fig. 5.21:** Different views of the camera enclosure component.

After integrating all the components into the box, the final design was created.

**Fig. 5.22:** Front isometric view of the final design of the enclosure, featuring all the components.

**Fig. 5.23:** Isometric view of the final design of the enclosure with detached lid, featuring all the components.

After generating the model, it was imported into the available .step file of the aircraft to verify that the enclosure fit well inside the cargo bay.

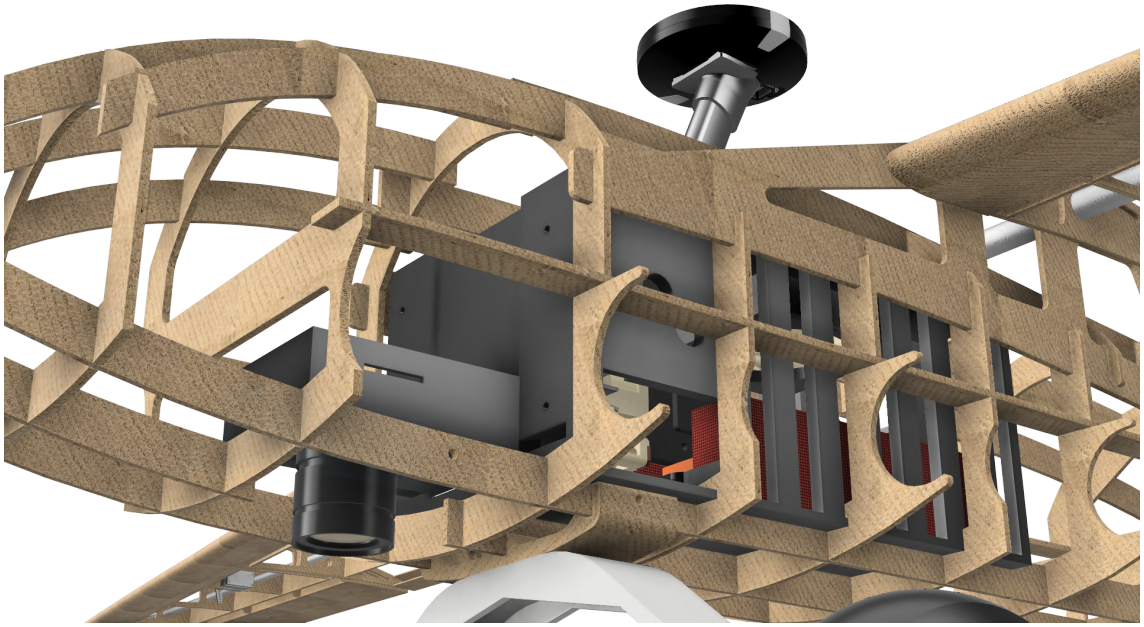


**Fig. 5.24:** Oblique side view of the system's enclosure inside the master CAD model. The image shows that the enclosure fits well inside the cargo bay of the aircraft and that the opening for the camera aligns with the camera module.

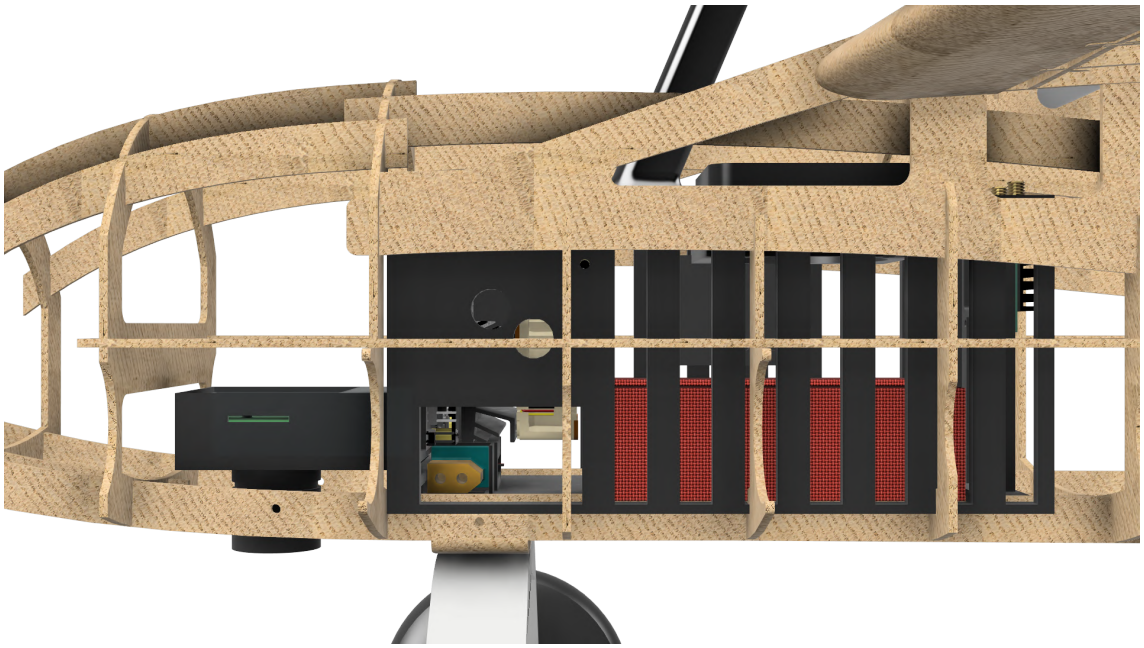
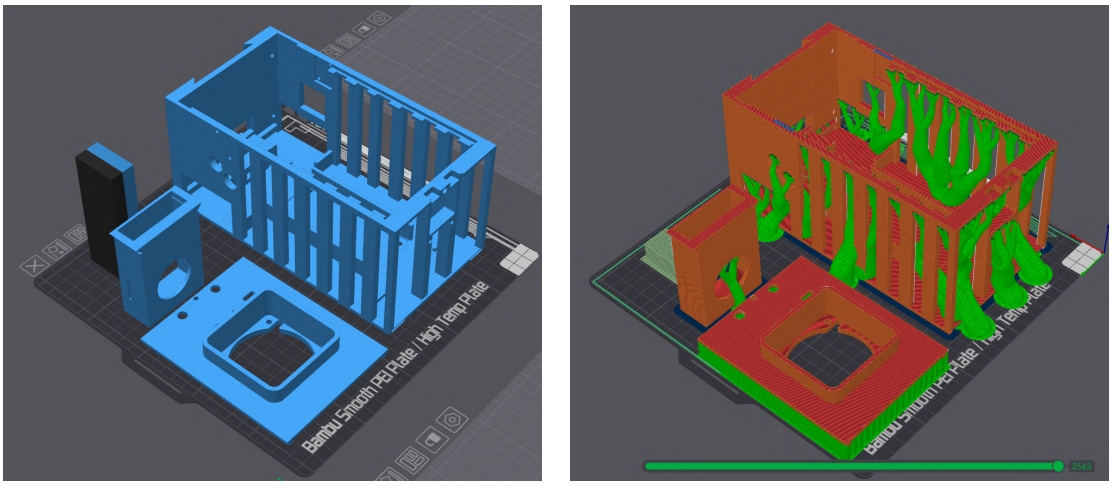


**Fig. 5.25:** Lateral view of the system's enclosure inside the master CAD model. Openings were designed in accordance with the position of the fuselage's ribs for quicker access.

Once the designed enclosure was validated against the master *CAD* model, it was exported from Fusion as a .3mf manufacturing format and imported into the slicing software Bambu Studio to generate the 3D printing G-code.



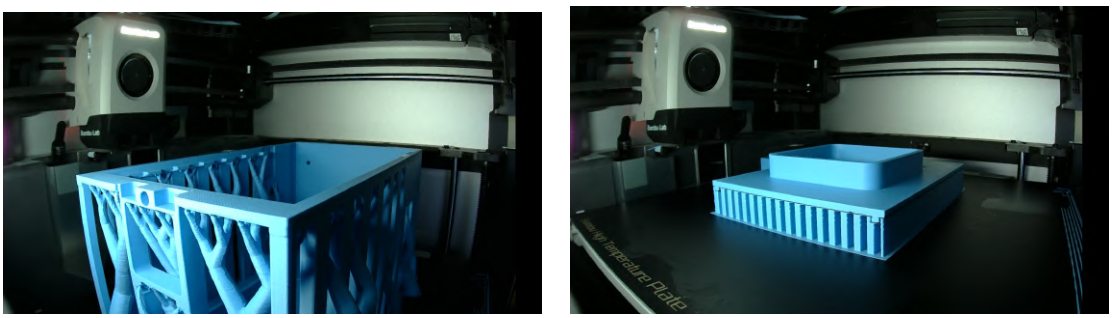

Imported file Sliced file

**Fig. 5.26:** 3D printing of the main enclosure in the slicing software Bambu Studio.

As the enclosure featured many overhangs, tree supports were added to prevent the collapse of the structure and to minimize material waste. The prototype was printed using PLA at a nozzle temperature of 220ºC and a bed temperature of 55ºC, with the following relevant parameters that differed from the standard PLA printing profile provided by the software:

- 1 wall to minimize weight.
- Outer brim active to prevent warping and separation of the piece from the bed.
- 10% infill density.

After generating the G-code, it was sent to the Bambu Lab X1C for manufacturing, which took 10 hours. A total of 278.83 g of material was used, with 190.67 g belonging to the final piece and the remainder being waste material from the supports.



Enclosure Lid

**Fig. 5.27:** 3D printing manufacturing process with the Bambu Lab X1 Carbon.

Once the 3D printing process was completed and the parts were cleaned of support material, threaded inserts were installed at the locations designated for bolts. This setup facilitates the straightforward attachment and detachment of components from the enclosure. This approach supports the objectives of modularity and ease of implementation, enabling efficient integration and maintenance.

**Fig. 5.28:** Front view of the 3D printed enclosure after support removal.

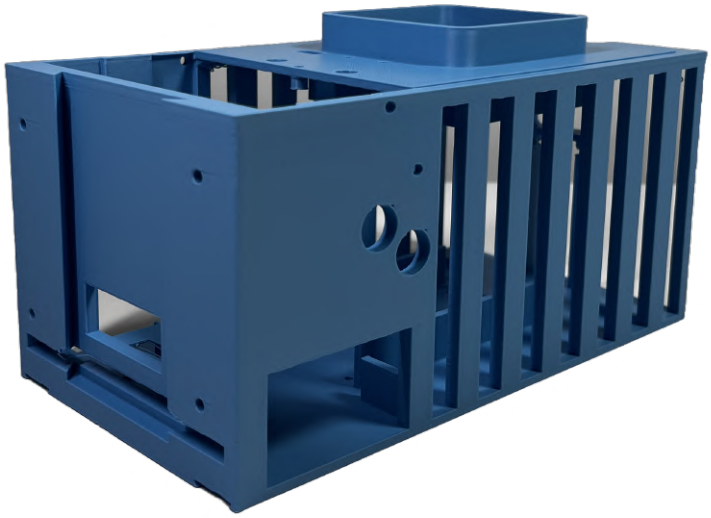


**Fig. 5.29:** Isometric view of the 3D printed enclosure with lid installed.

## 5.5 Component Verification

A set of verification procedures was established to ensure that the different components met their technical specifications. These procedures focused on testing the components in standalone mode. Instructions for verifying each component's functionality are provided in this section.

### 5.5.1 Onboard Computer

Before powering up the Jetson Nano, as illustrated in Figure 5.9, a *Jetson Nano Developer Kit SD Card Image* needed to be written to the microSD card, which served as the main flash drive of the onboard computer. In this case, a 128 GB microSD card was used. The process involved the following steps [155]:

- Formatting the card using an SD Memory Card Formatter to exFAT format.
- Flashing the downloaded ISO image with Etcher.
- Inserting the flashed microSD card into the back of the Jetson Nano.
- Connecting an HDMI cable to an external screen to oversee the installation. [10]

After completing these steps, the Jetson Nano could be booted up. Correct installation and setup were verified if the screen displayed as shown in Figure 5.30.

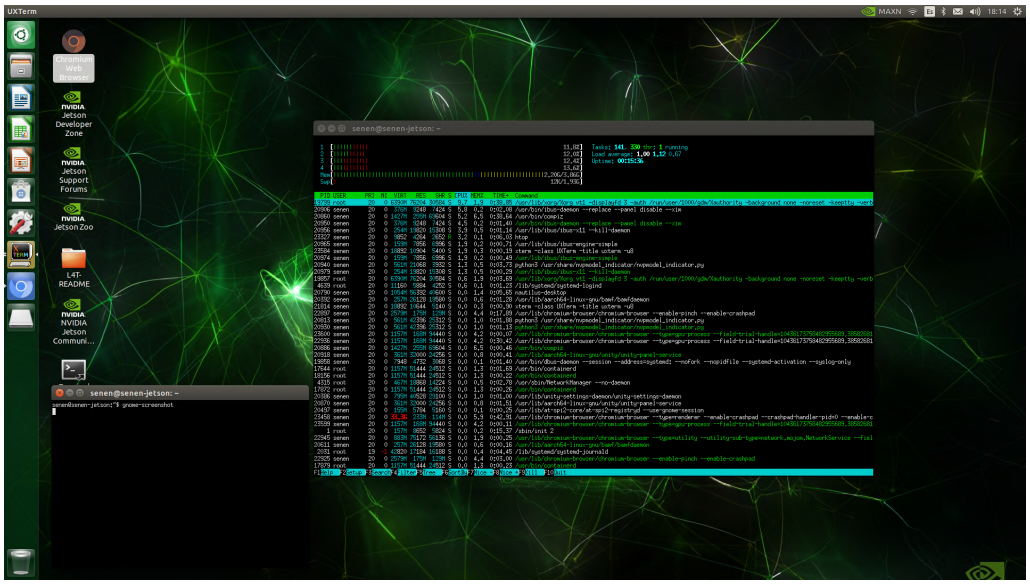


**Fig. 5.30:** Jetson Nano setup complete, with a display of core metrics, memory usage, and running processes.

[10] Note that a "headless" installation was also possible, which involved performing the installation through a serial COM port (USB) without an external screen. This option reduced memory usage as there was no active GUI. However, for this project, a "headed" setup was chosen for better visualization.

### 5.5.2 Camera

Once the camera was connected via the MIPI-CSI port to the onboard computer using the 22-to-15 flex ribbon cable (with the pins facing towards the inside of the computer), it was necessary to install a specific driver. [11] To install the driver, the following commands were executed from the console:

```
$ cd ~
$ wget https://github.com/ArduCAM/MIPI_Camera/releases/download/v0.0.3 \
/install_full.sh
$ chmod +x install_full.sh
$ ./install_full.sh -m imx477
```

After rebooting, the successful installation of the driver was verified by executing the following command:

```
$ dmesg | grep -E "arducam|imx477|imx219"
```

The cable connection was checked by running:

```
$ ls /dev/video*
```

The command was expected to return "/dev/video0" to indicate an active video connection.

To display the image from the camera, the following script was used, which employed GStreamer, an open-source multimedia framework for managing image acquisition pipelines:

```
$ gst-launch-1.0 nvarguscamerasrc sensor_id=0 ! \
'video/x-raw(memory:NVMM),width=1920, height=1080, framerate=30/1' ! \
nvvidconv flip-method=0 ! 'video/x-raw,width=960, height=540' ! \
nvvidconv ! nvegltransform ! nveglglessink -e
```

The command is broken down as follows:

- **gst-launch-1.0**: GStreamer command-line tool used to build and run multimedia pipelines.
- **nvarguscamerasrc sensor_id=0**:

[11] This was not required if connected through the USB (B type) port provided by the UVC-compliant enclosure. However, since the MIPI-CSI port offered higher data transfer speeds, it was recommended.

  - **nvarguscamerasrc**: GStreamer plugin for capturing video from a camera using the Argus camera API (specific to NVIDIA Jetson platforms).
  - **sensor_id=0**: Selects the first camera (sensor 0) if multiple cameras are connected.

- **!**: Chains different elements in the GStreamer pipeline, connecting the output of one element to the input of the next.
- **'video/x-raw(memory:NVMM),width=1920, height=1080, framerate=30/1'**:
  - **video/x-raw**: Specifies raw video data.
  - **memory:NVMM**: Indicates the use of NVIDIA's memory management (specific to Jetson).
  - **width=1920, height=1080**: Sets the video resolution to Full HD.
  - **framerate=30/1**: Sets the frame rate to 30 frames per second.
- **nvvidconv flip-method=0**:
  - **nvvidconv**: GStreamer plugin for video conversion (resizing, cropping, flipping, etc.).
  - **flip-method=0**: Specifies no flipping of the video.
- **'video/x-raw,width=960, height=540'**: Further specifies the video format, resizing the video to 960x540 pixels.
- **nvvidconv**: Another instance of the video conversion plugin, likely for additional processing.
- **nvegltransform**: Plugin for performing OpenGL transformations on the video stream, preparing it for display.
- **nveglglessink**: The sink element that renders the video to the screen using OpenGL (EGL).
- **-e**: Sends an EOS (End of Stream) signal to the pipeline upon termination, allowing for graceful cleanup.

If everything was set up correctly, the image was displayed on the screen, verifying the connection with the camera.


**Fig. 5.31:** IMX477 Camera setup with video feed running through GStreamer.

Additionally, to verify the performance of the Jetson Nano, the following feature tracking algorithm (Harris Corner Detector) was run using a demo video feed:

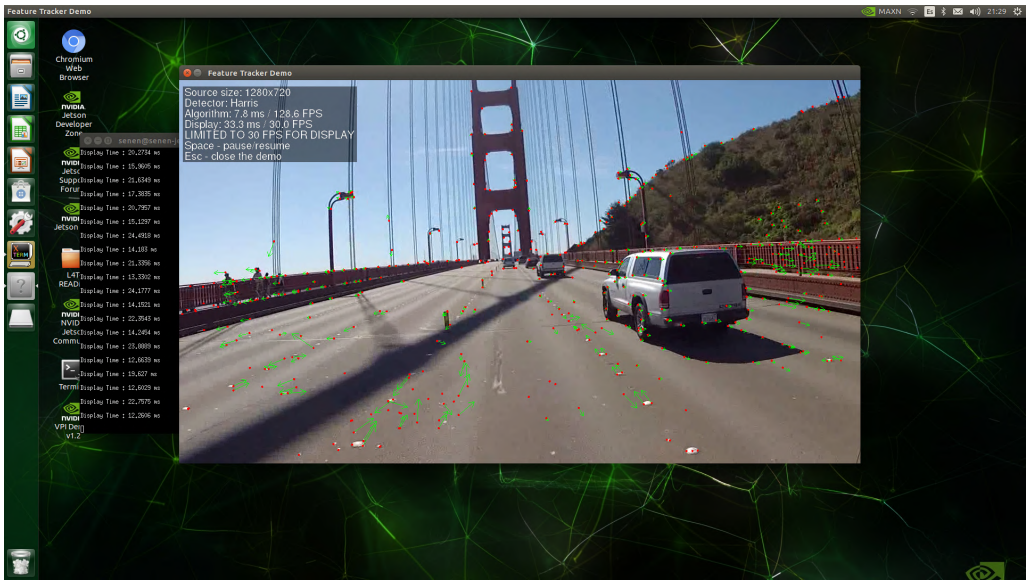
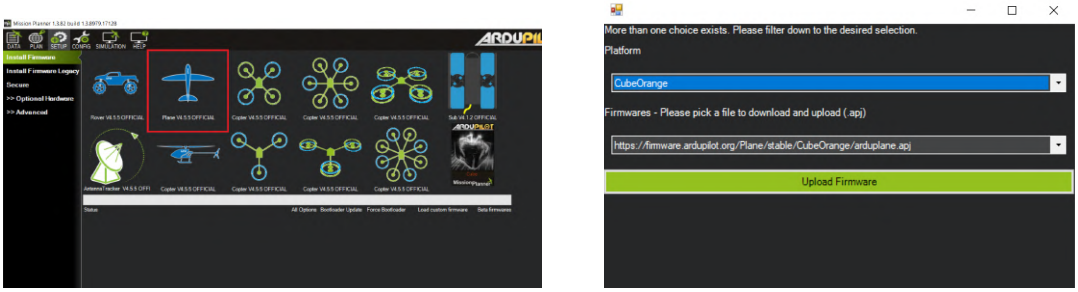



**Fig. 5.32:** Jetson Nano running a feature tracking algorithm (Harris Corner Detector) for performance verification with the video feed. Arrows describe the relative velocities of the tracked points.

### 5.5.3 Flight Controller

The Orange Cube was connected directly to a Windows computer via USB. Using the *GCS* software Mission Planner, the flight controller was flashed with the appropriate firmware for the type of aircraft being used. For fixed-wing aircraft, the correct setup involved the *Arduplane* firmware v4.5.5.




Firmware selection Firmware upload

**Fig. 5.33:** Arduplane firmware installation procedure.

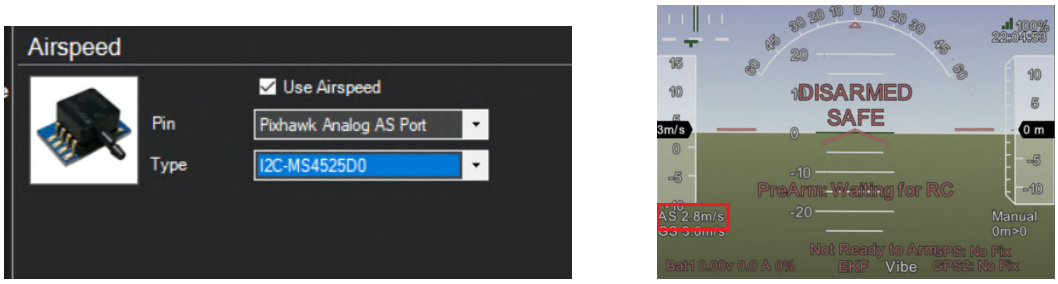
After the firmware was installed, the *GCS* had the possiblity to the flight controller via multiple communication ports. COM6 (Mavlink) was selected in the top right corner to verify that the information was being received correctly.

If the GPS was connected through the CAN1 port, further adjustments were required to enable the port. This was accomplished through the configuration panel in the full parameter list:

- CAN_P1_DRIVER was set to 1 to enable the bus.
- GPS_TYPE was set to 9 to configure the communication protocol to droneCAN.


**Fig. 5.34:** Mission Planner primary display with *IMU*, *GPS*, and airspeed sensor readings.

Simultaneously, by connecting the airspeed sensor to the I2C2 port, the readings could be automatically observed in the main control panel.




Airspeed sensor connection Airspeed sensor reading

**Fig. 5.35:** Differential Pressure sensor via I2C2 port.

### 5.5.4 Radio Controller

The radio link was verified by displaying on the radio controller (TX16S) the telemetry received from the radio modem RFD868x, using the TELEM1 port on the Orange Cube.

### 5.5.5 PWM-based Components

The remaining components, such as the Motor, ESC, and servomotors, were directly tested with a PWM signal.


**Fig. 5.36:** Verification of PWM-based components with a simple servo tester.

## 5.6 Integration and Flight Testing

### 5.6.1 System Integration

This chapter will focus on how bidirectional communication between pairs of devices was verified to ensure that data from the camera and flight controller can be transferred to a Python environment within the onboard computer for real-time analysis.

#### 5.6.1.1 Camera - Onboard Computer

Although the relationship between the camera and the onboard computer was previously verified using Gstreamer in section 5.5, since the working environment for further development was going to be Python, establishing access to the camera module from Python libraries such as OpenCV was necessary.

First the python library OpenCV was downloaded using the usual approach from the terminal:

```
$ sudo apt-get install python3-opencv
```

Then a python file was created, which initialized a Gstreamer pipeline for image acquisition from OpenCV and display the images through a videoCapture object.

```
import cv2

def open_camera():
    #GStreamer pipeline for MIPI-CSI camera
    gst_str = (
        "nvarguscamerasrc ! "
        "video/x-raw(memory:NVMM), width=(int)1280, height=(int)720, \
        format=(string)NV12, framerate=(fraction)30/1 ! "
        "nvvidconv flip-method=2 ! "
        "video/x-raw, width=(int)1280, height=(int)720,  \
        format=(string)BGRx ! "
        "videoconvert ! "
        "video/x-raw, format=(string)BGR ! appsink"
    )

    cap = cv2.VideoCapture(gst_str, cv2.CAP_GSTREAMER)

    if not cap.isOpened():
        print("Error: Cant open the camera.")
        return
```

```
    while True:
        ret, frame = cap.read()
        if not ret:
            print("Error: Cant read the frame.")
            break

        cv2.imshow("Camera", frame)

        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    cap.release()
    cv2.destroyAllWindows()

open_camera()
```

After that, proper communication was verified by running the file.

```
$ sudo python3 cv2_connection_camera.py
```

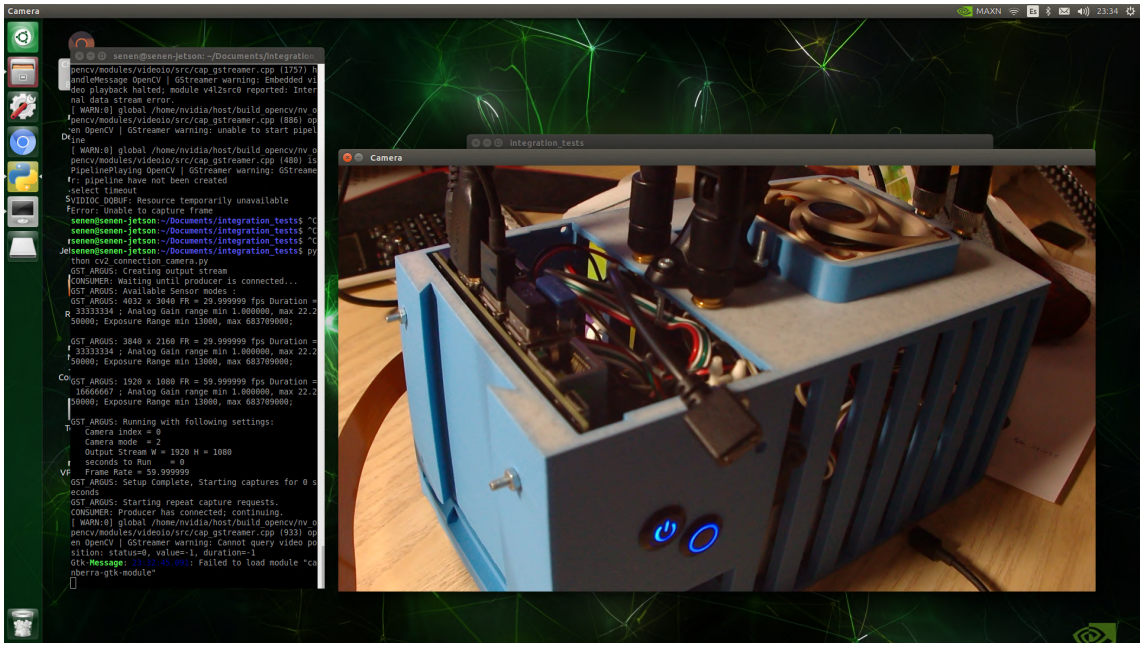



**Fig. 5.37:** Established communication between camera and Jetson through Python.

#### 5.6.1.2 Onboard Computer - Flight Controller

There are several ways to establish data transfer between the flight controller and the onboard computer. The most commonly used methods are:

- MAVROS (MAVLink + ROS): a *ROS* package that allows controlling $UAV$s via the MAVLink protocol.
- Pymavlink: a Python implementation of the MAVLink protocol.
- Dronekit (forked from Pymavlink): an easy-to-implement Python library that can be used to connect to, monitor, and control a $UAV$.

In this case, **Dronekit** was used to verify that the sensor data from the $UAV$ could be accessed through Python. Peripheral sensors were also verified through this method, such as the differential pressure sensor, which provides readings of the aerodynamic velocity.

First, the necessary dependencies were installed:

```
$ pip3 install dronekit pymavlink pyserial
```

Then, the following script was created to connect the Orange Cube to the Jetson.

```
from dronekit import connect, VehicleMode
from pymavlink import mavutil
import logging
import time

# Suppress logging for the 'dronekit' logger
logging.getLogger('dronekit').setLevel(logging.CRITICAL)

#Specify connection string to
connection_string = '/dev/serial/by-id/usb-Hex_ProfiCNC_CubeOrange_  \
2C004E000C51313132383631-if00'
baud_rate = 115200

print(f'Connecting to vehicle on: {connection_string}')
vehicle = connect(connection_string, baud=baud_rate, wait_ready=True)

# Function to display sensor data
def display_sensor_data(vehicle):
    print("\n========== VEHICLE STATE ==========")
    print(f"GPS Fix Status: {vehicle.gps_0.fix_type} \
    (0=no fix, 1=3D fix, etc.)")
    print(f"Number of Satellites: {vehicle.gps_0.satellites_visible}")
    print(f"Latitude: {vehicle.location.global_frame.lat:.6f}, \
```

```
    Longitude: {vehicle.location.global_frame.lon:.6f}")
    print(f"Altitude (Relative): \
    {vehicle.location.global_relative_frame.alt:.2f} meters")
    print(f"Altitude (Absolute): \
    {vehicle.location.global_frame.alt:.2f} meters")
    print(f"IMU Pitch: {vehicle.attitude.pitch:.6f} rad")
    print(f"IMU Yaw: {vehicle.attitude.yaw:.6f} rad")
    print(f"IMU Roll: {vehicle.attitude.roll:.6f} rad")
    print(f"Velocity: {vehicle.velocity} \
    (m/s in NED frame [North, East, Down])")
    print(f"Airspeed: {vehicle.airspeed:.2f} m/s")
    print(f"Groundspeed: {vehicle.groundspeed:.2f} m/s")
    print(f"Heading: {vehicle.heading} degrees")
    print("===================================\n")

# Main loop
try:
    while True:
        display_sensor_data(vehicle)
        time.sleep(1)  # 1 second
except KeyboardInterrupt:
    print("Exiting...")

# Close vehicle object
vehicle.close()
print("Vehicle disconnected")
```

Finally the python file was run with admin-privileges (sudo). This was necessary because the file was accessing specific ports of the computer.

```
$ sudo python3 orange_cube_to_jetson.py
```

The command line then displayed the readings of the $IMU$, airspeed sensor and GPS with the frequency established in the script, verifying that the communication layer between these two devices was unlocked, together with all the sensors connected to the flight controller, as it can be observed in Figure 5.38.

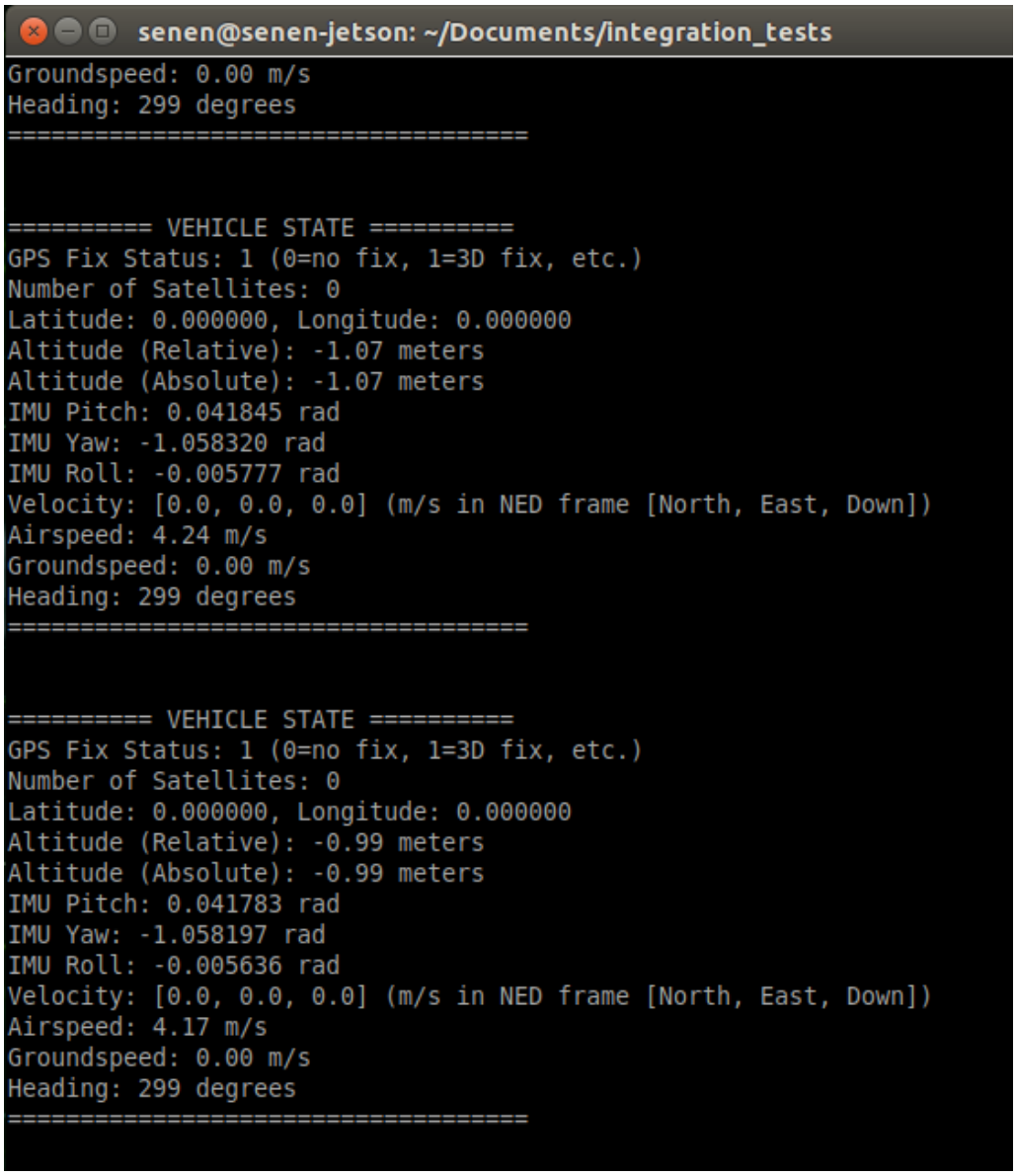



**Fig. 5.38:** Established communication between Orange Cube and Jetson through Python, displaying the sensorial data through the console.

### 5.6.2 Flight Testing

A series of flight tests were conducted to assess the aircraft's capability to maintain stable flight. These tests took place at "Club Petirrojo" in Valdemorillo (Madrid) and were carried out by a certified operator on the ground, always within $VLOS$.

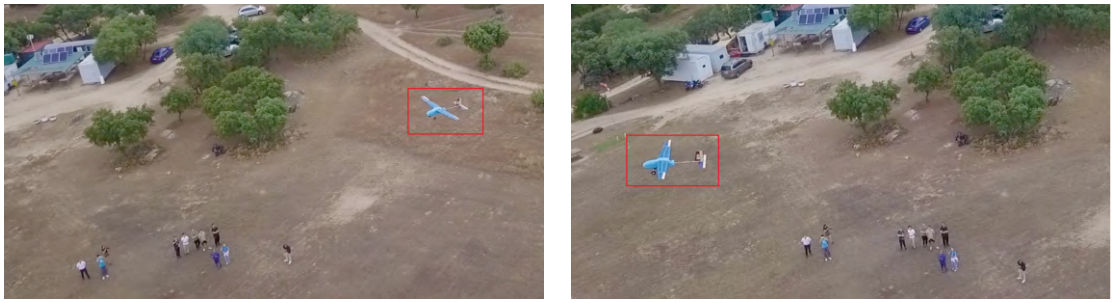


**Fig. 5.39:** Test drone captured at two distinct time points during a flight test at Valdemorillo airfield.

**Fig. 5.40:** Aerial view from onboard GoPro during the test flight.

At least on four occasions, the test flights were instrumented with a simple data acquisition system to collect information from multiple sensors. The approach involved incorporating a smartphone onboard, wrapped in a foam case, and running an application that registered sensor data from accelerometers, gyroscopes, barometers, and GPS, among others. This setup aimed to analyze the $UAV$'s performance after the flight, with the foam enclosure designed to absorb impacts in case of a crash, thereby protecting the electronics.

The registered data was organized into four datasets, which are publicly available through Github:

1. 2024-06-16 | Single flight that experienced an operational failure.
2. 2024-06-25 | Single flight that experienced an operational failure.
3. 2024-06-29 | Single flight that experienced an operational failure.
4. 2024-07-07 | Four consecutive successful flights.

#### 5.6.2.1 Data Acquisition

Two different data acquisition applications were used. Flights 1, 2, and 4 were recorded with SensorLog v5.3, which followed the "comma-separated-value" (csv) standard with a header containing 82 different variables. Flight Nº3 was recorded with a simpler application, the Engineering Physics Toolbox Sensor Suite. All datasets were obtained using these applications running on an iPhone 13 Pro Max, onboard the aircraft and positioned at the center of gravity.

#### 5.6.2.2 Post-Flight Analysis

Once the data became available, it was first cleaned and then processed to extract valuable metrics. **Python** was used for data processing. The workflow included the following steps:

- A conda environment was created to encapsulate all dependencies within the same virtual environment, using a .yaml file. This setup allowed external users to quickly recreate the findings and review the data.
- The header of the csv file was analyzed to select relevant metrics from the data collection.
- To estimate the approximate start time of the flight, the altitude versus time was plotted to visualize when the flight occurred in relation to when the application started recording data. The application often began recording several minutes before the actual flight took place.
- Since the flights typically lasted no longer than 5 minutes, the dataset was split around the identified "highest altitude reached" $\pm 3$ minutes.
- The datasets were saved for further use, ensuring that all information was relevant for subsequent analysis.
- Relevant metrics were plotted, and data was transformed to facilitate the retrieval of key flight insights.

## 5.7 Deployment and Preparation for Future Upgrades

Validation against the original solution requirements, as outlined in section 5.1, is performed by comparing these requirements with the proposed architecture and flight test results.

1. The system uses *COTS* components and open-source software (Pixhawk, Mission Planner, Python, Ardupilot).

2. The system integrates a modular framework allowing for different camera modules or GPU-powered units. The system's enclosure can be adapted for use in another *UAV* if the volume requirements are met.

3. The system enables the evaluation of any navigation method against a ground truth with RTK-GPS, allowing for comparison of different navigation implementations.

4. The system processes data in real time and relays it back to the *GCS* through the RFD 868X radio modem. Information is displayed on the screen with Mission Planner through a WiFi network created by the TXMOD component at the radio transmitter.

5. The system features three different data logging methods: using the micro SD card integrated inside the flight controller, through the onboard computer that receives telemetry via the USB port, or with the *GCS* software on the ground.

6. The system demonstrated its capability for stable flight through a series of flight tests.

7. The system includes two battery modules connected to the flight controller. If one battery is compromised, the system continues operating as both batteries have the same nominal voltage and can be interchangeably used for system operations or propulsion.

8. The system is equipped with fail-safe mechanisms implemented through the configuration environment at the ground control station. More complex control laws and emergency protocols can also be deployed through the onboard computer.

9. The system is designed to accommodate future upgrades, both to the onboard systems and the aerial platform, as the files can be edited by anyone.

10. The system complies with applicable legislation, falling under *EASA*'s Open Category, with an *MTOW* of less than 25 kg, a wingspan of less than 3 meters, and always flown within *VLOS*.

11. The system uses CAN, UART, USB, and I2C for its cabled communications, and 868 MHz and 2.4 GHz for its radio and WiFi communications, respectively, all of which are industry standards.

Further work would require that the proposed solution be deployed and tested by external users or researchers. This critical phase would ensure that the system meets the established performance criteria and fulfills its intended purpose through real-world testing.

Throughout the development phase, a systematic progression from requirement definition to system design, implementation, and verification was followed, with each stage meticulously executed to align the system with the initial specifications. The final stage of this process would typically involve deploying the complete system onto a specific $UAV$ platform and conducting comprehensive tests to assess its performance in a realistic operational environment.

However, due to the need for selecting and developing an alternative navigation method before testing in a production environment, the decision was made to postpone full-scale deployment. Since this thesis focused on providing a framework that allows future researchers to choose and implement the navigation approach they wish to test, the final validation process was not performed. This left the choice of navigation strategies open, enabling future work to explore and validate the system using different approaches in a controlled testing environment.

To facilitate ongoing development and validation, all relevant files, datasets, and code were made publicly available in the author's GitHub repository. This repository serves as a resource for future researchers, allowing them to implement the proposed solution, conduct further testing, and provide feedback. By maintaining these resources openly, the V-model's iterative feedback loop can be supported, helping to advance open-source solutions in the $UAV$ space and enabling continuous improvement of the proposed solution over time.

# 6 Results

## 6.1 Integration with aircraft

Once all the components have been verified to be working properly, the final assembly of the enclosure box can be performed. After the threaded inserts are set in place, the components are secured in their respective positions using M2.5 screws. Two buttons are added on the left side to independently control the startup of the onboard computer.

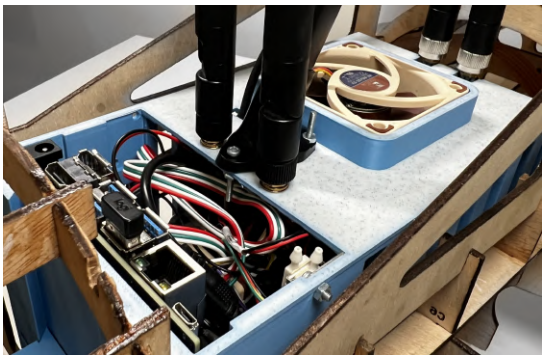
The result is shown in Figure 6.1. The enclosure fits all the components and ensures that nothing moves during flight. The final mass of the enclosure, including all the electronics, is **1.2 kg**.

**Fig. 6.1:** Final enclosure box with all components installed and running through the LiPo batteries.

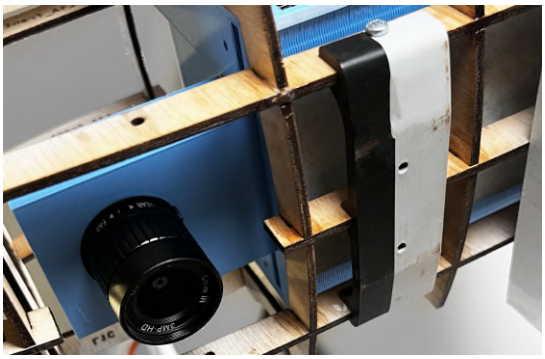
The integration of the electronics enclosure box into the drone's fuselage clearly confirms the design's effectiveness. The box fits well inside the payload bay and does not create any moment along the pitch axis, as it is positioned at the aircraft's center of mass.

**Fig. 6.2:** Final enclosure inside the aircraft's fuselage.

Detailed views from Figure 6.3 show easy access to all the I/O panels of the onboard computer as well as the pressure sensor ports, which facilitate development and integration with other sensors, as well as testing in Software In The Loop ($SITL$)/Hardware In The Loop ($HITL$) scenarios. In the image on the right, the camera lens can be observed resting on the camera opening that the Roadrunner features.



Enclosure view



Downward-facing camera view

**Fig. 6.3:** Detailed views of the enclosure inside the fuselage.

## 6.2 Flight Test Results

As anticipated in Section 5.6, during the flight tests, the aircraft was instrumented to collect data. The goal of this action was to demonstrate that the aircraft had the capacity for stable flight, allowing further testing in the future with the onboard systems box presented in Section 6.1.

A series of plots are presented here to illustrate the results of the final flight testing campaign. This includes the four successful flights conducted on July 7, 2024, at the Valdemorillo airfield. Other flights have been excluded due to issues encountered during testing. From the dataset containing 82 variables, the following ones were selected for analysis:

| Variable | Units |
|---|---|
| **locationTimestamp_since1970** | s |
| **locationLatitude** | WGS84 |
| **locationLongitude** | WGS84 |
| **locationAltitude** | m |
| **locationSpeed** | m/s |
| **locationSpeedAccuracy** | m/s |
| **altimeterPressure** | kPa |

Table 6.1: Data Types for Selected Dataset Variables

Location-based variables are inferred from the GPS receiver using triangulation. These variables were selected instead of integrating the position with accelerometer and gyroscopic data to obtain a "ground truth" position of the aircraft.

### 6.2.1 Height vs Time

Both GPS-derived and altimeter-derived height readings are plotted against time, with their maximum values identified in Figure 6.4. The altitude data have been normalized by its first value of the dataset, with the aircraft on the runway, to obtain the height relative to the ground and not exceed the 120 m ceiling established by regulations, considering the field as a flat plane. The altimeter-based reading was obtained using the usual barometric expression relating pressure and altitude accounting for the values at sea level.

Notice how the altimeter readings are always lower (and delayed) compared to the GPS data. The discrepancy arises because the altimeter measures altitude based on atmospheric pressure, which can be influenced by weather conditions and altitude changes, while GPS provides a direct measurement of position, often leading to more immediate and accurate altitude readings. Additionally, the altimeter's lag could be due to its slower response to atmospheric pressure changes compared to real-time GPS data.

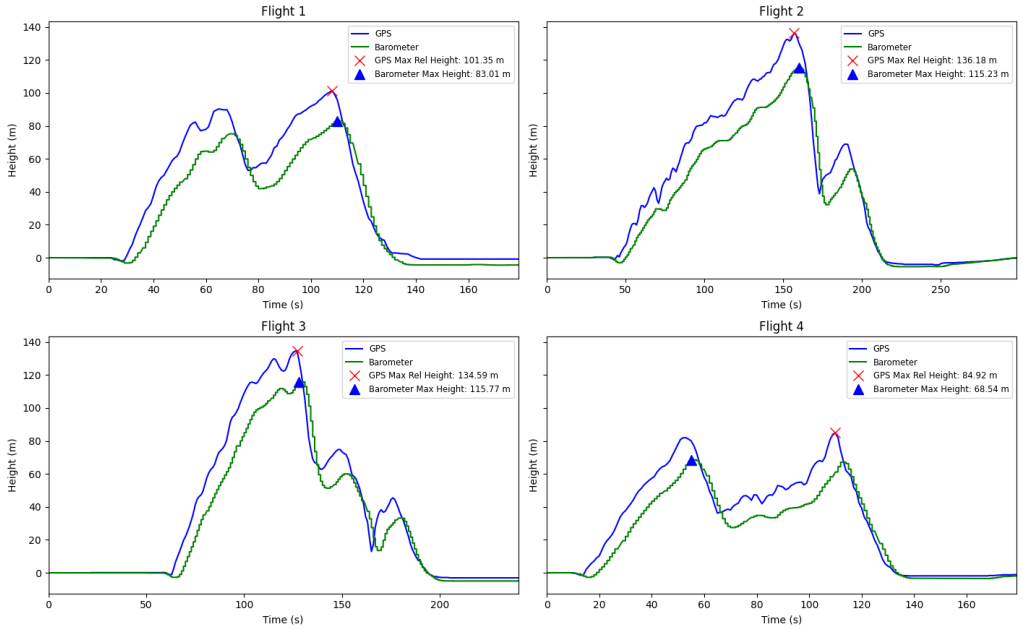



**Fig. 6.4:** Height vs Time plots for flights on July 7$^{\text{th}}$, 2024. Both GPS-derived and altimeter-derived heights are displayed, with their maximum values identified.

### 6.2.2 Velocity vs Time

The GPS-derived speed is represented in Figure 6.5 and calculated using the Haversine formula to obtain the distance traveled from latitude and longitude data, then divided by the time between measurements. In this case, the speed accuracy has been added to the plot as shaded bars to provide a sense of the possible errors involved in the measurements.

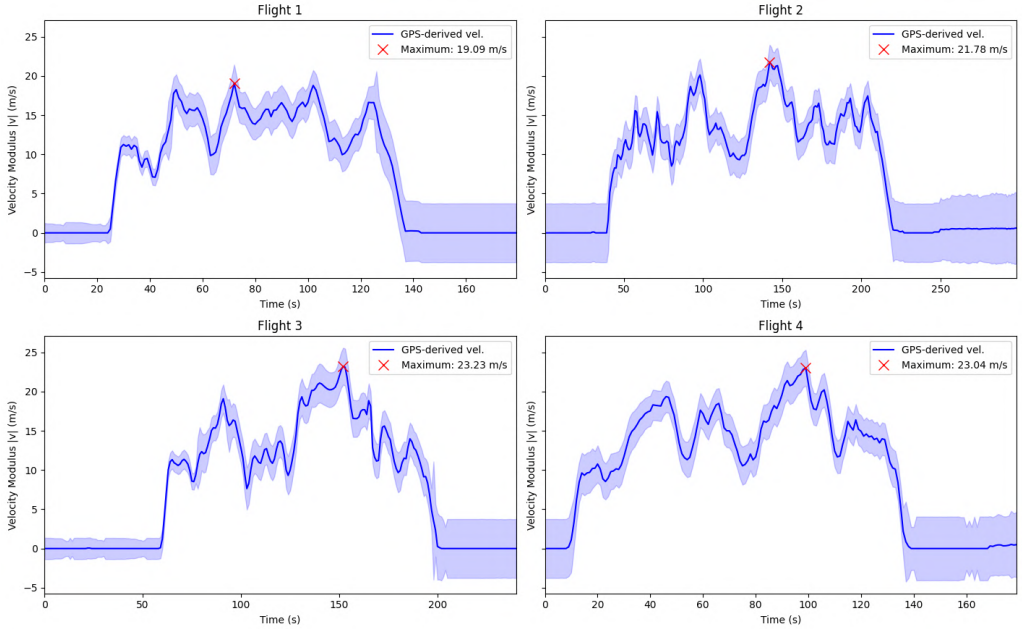



**Fig. 6.5:** Velocity vs Time plots for flights on July 7$^{\text{th}}$, 2024. Maximum readings are noted by a red cross, while shaded bars indicate accuracy errors.

### 6.2.3 4-D Trajectory

The 4D trajectory is constructed from latitude and longitude data in WGS84 format. This format approximates the shape of the Earth as an ellipsoid and uses latitude and longitude coordinates to locate positions on the surface. After projecting onto a plane, the plots in Figure 6.6 can be displayed.

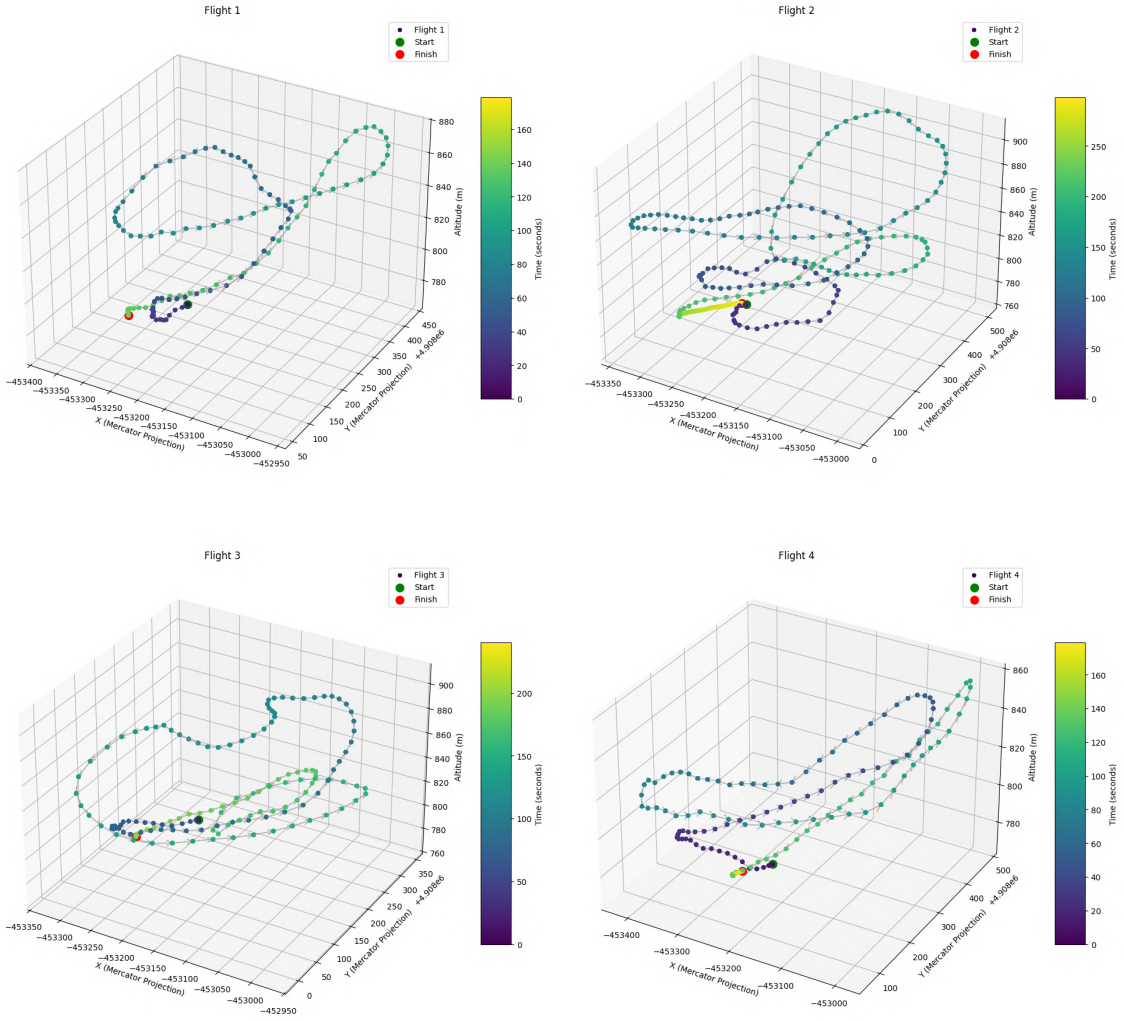



**Fig. 6.6:** 4-D trajectory plots for flights on July 7$^{\text{th}}$, 2024. A trajectory progress bar allows tracking the 3D position as a function of time for better representation.

The graphs presented in this section validate the aircraft's capability for flight and demonstrate that the initial data processing system provided reliable insights. These findings confirm the aircraft's performance. For subsequent analysis, the data acquisition system outlined in Section 6.1 will be used for further research.

# 7 Socio-Economical Environment

## 7.1 Socio-Economical Impact

The development of autonomous navigation systems for $UAV$s in $GPS$-denied environments holds significant socio-economic implications across several critical sectors. These impacts can be categorized into five primary fields: **transportation, infrastructure inspection, security and defense, disaster response, and space exploration.**

### 7.1.1 Transportation

The transportation sector, particularly in the fields of package delivery and medical equipment logistics, can benefit greatly from advancements in autonomous $UAV$ navigation. In regions where $GPS$ signals are compromised or completely unavailable, $UAV$s equipped with robust autonomous navigation systems can continue to operate effectively, ensuring timely delivery of critical items. Currently, the main and only way an $UAV$ can position itself with accuracy is through $GPS$, which becomes a critical dependency for an aerial system if not paired with a backup navigation system. The capability of having an alternative that is not reliant on external inputs is particularly important in remote areas where conventional transportation infrastructure is lacking or in densely populated urban environments. $UAV$s can contribute to decreased traffic congestion, lower emissions, and enhanced delivery efficiency. Moreover, the potential to deliver critical medical supplies, such as blood, vaccines, and other life-saving medications, to hard-to-reach locations could revolutionize healthcare delivery, improving logistic capabilities and saving lives. This is the business model of companies like Zipline [156].

### 7.1.2 Infrastructure Inspection

Autonomous $UAV$s are increasingly employed in the inspection and maintenance of infrastructure, including bridges, electricity lines, and telecommunications towers. Such is the case of the Spanish startup FuveX [157]. These structures can be present anywhere, even in regions where the $GPS$ signal is unavailable. However, $UAV$s equipped with alternative autonomous navigation systems can operate reliably in these conditions, providing both high-resolution imagery and data critical for assessing the condition of infrastructure. This capability reduces the need for manual

inspections, which are often time-consuming, expensive, and hazardous. As a result, infrastructure maintenance becomes more efficient and cost-effective, extending the lifespan of critical infrastructure.

### 7.1.3 Security and Defense

In security and defense, the ability to navigate autonomously in $GPS$-denied environments is a major challenge in electronic warfare. In military operations, particularly in conflict zones where $GPS$ jamming and spoofing are constant problems, $UAV$s must rely on alternative navigation methods to complete their missions. Autonomous $UAV$s can be deployed for a variety of tasks, including surveillance, reconnaissance, target acquisition, and even direct combat roles. By ensuring reliable operation in $GPS$-degraded environments, $UAV$s enhance the operational capabilities of military forces, providing critical intelligence and reducing the risk to human operators.

### 7.1.4 Disaster Response

Autonomous $UAV$s capable of operating in $GPS$-denied environments can play a vital role in search and rescue operations, particularly in areas where disasters have disrupted $GPS$ infrastructure or in locations where $GPS$ signals are naturally weak due to the topography. The $FAA$ has invested in research centers to ensure that the technology advances at a rapid pace [158]. These $UAV$s can quickly and efficiently cover large areas, identifying survivors, assessing damage, and delivering emergency supplies. The ability to navigate autonomously in challenging conditions enhances the effectiveness of disaster response teams, potentially saving lives and accelerating recovery efforts. Moreover, the use of $UAV$s reduces the need for first responders to enter hazardous areas, minimizing the risk to life and enabling a more coordinated and safer response.

### 7.1.5 Space Exploration

$GPS$-denied navigation is also of critical importance for $UAV$s aiming to explore other planets, such as Mars or Titan, one of Saturn's moons. As space agencies like $NASA$ continue to push the boundaries of space exploration, tasks such as conducting aerial surveys, geological assessments, and atmospheric studies cannot rely on $GPS$. In these cases, $UAV$s must rely on autonomous navigation techniques to traverse terrain, map surface features, scout landing areas, or gather scientific data. This technology could significantly enhance our understanding of other planetary environments, aiding in the selection of landing sites for future missions and supporting the search for extraterrestrial life. The ability to operate safely in these extreme and uncharted environments is thus a critical research opportunity for the future of these missions [159, 160].

## 7.2 Environmental Impact

The rapid development and deployment of $UAV$s for commercial applications have significant environmental implications. As drone technology continues to advance, it is crucial to consider both its potential benefits and drawbacks on the environment. This section explores how this impact is translated to nature, wildlife, and ecosystems, focusing on issues such as noise pollution, energy consumption, and potential hazards, as well as the positive contributions drones can make to environmental conservation and management.

### 7.2.1 Impact on Nature and Wildlife

Propeller-powered $UAV$s can have a noticeable impact on wildlife and natural habitats. The noise generated by drone propellers can be disruptive to animals, mostly birds, which may be sensitive to such disturbances. This noise pollution can interfere with birds' natural behaviors, such as feeding, mating, and migration patterns. In extreme cases, the presence of drones may lead to stress or displacement, potentially affecting survival and reproductive success. Additionally, drones operating in sensitive or protected areas may inadvertently disturb wildlife or disrupt their habitats. It is essential to conduct environmental impact assessments and develop guidelines to minimize disturbances and ensure that large scale drone operations respect nature and does not create a negative ever lasting effect.

### 7.2.2 Energy Consumption and Battery Hazards

Drones are typically powered by lithium-polymer (LiPo) batteries, which can be used as part of a sustainable solution but come with their own environmental concerns. Improper handling, storage, or disposal of LiPo batteries can lead to hazardous situations, including fires and explosions. These incidents not only pose risks to human safety but also contribute to environmental pollution if not handled correctly. The manufacturing, disposal, and recycling of these batteries have implications for resource use and waste management.

### 7.2.3 Positive Environmental Contributions

$UAV$s also offer several positive contributions to environmental management and conservation efforts. Examples of this include reforestation projects by planting trees and monitoring the growth of newly planted areas. They provide valuable assistance in agricultural practices by monitoring crop health, optimizing resource use, and reducing the need for chemical inputs. Additionally, they can also play a role in wildlife monitoring and habitat mapping, helping researchers track animal populations and assess ecosystem health. All these applications can contribute to more effective conservation strategies and promote sustainable land management.

Overall While $UAV$s present certain risks, they also offer valuable tools for society. This tradeoff of benefits and risks requires ongoing innovation and responsible practices to ensure that drone technology contributes positively to environmental sustainability.

## 7.3 Budget Breakdown

This chapter provides a detailed breakdown of the resources used to develop this thesis. The budget is divided into three categories: **manufacturing tools**, **consumables**[12], and **onboard electronics**.

Starting with machine costs, Table 7.1 lists the manufacturing tools purchased for this work. This table does not include servicing, maintenance, or spare parts for these tools.

| **Item** | **Model** | **Total (€)** |
|---|---|---|
| 3D Printer | Bambu Lab X1C+AMS | 1,808.14 |
| Laser Cutter | Xtool S1 10W | 1,179.00 |
| Press-Drill | Bosch PBD40 710W | 287.00 |
| Soldering Station | Zawaer 100W | 25.16 |
| Sub-total | | 3,299.30 |

Table 7.1: Budget breakdown for manufacturing and fabrication tools.

Next, Table 7.2 shows the amount of material required to manufacture one prototype, along with its associated cost. The actual cost may be higher, as numerous manufacturing trials were conducted prior to the final prototype to experiment with machine settings and other parameters.

| **Item** | **Qty.** | **Unit (€)** | **Total (€)** |
|---|---|---|---|
| Plywood | 1.08 $m^2$ | 11.10 | 12.00 |
| PLA Filament | 0.8 kg | 23.99 | 19.19 |
| Polycarbonate Filament | 0.2 kg | 42.99 | 8.60 |
| Oracover Vinyl | 1.24 $m^2$ | 15.00 | 18.60 |
| Cyanoacrylate | 0.01 kg | 1,198 | 11.98 |
| Sub-total | | | 70.37 |

Table 7.2: Budget breakdown for consumable materials.

[12]Unit prices are presented per kg or per $m^2$, depending on the quantity.

Finally, Table 7.3 summarizes all the electronic components used in the development of this thesis, along with their associated costs.[13]

| Item | Model | Qty. | Unit (€) | Total (€) |
|---|---|---|---|---|
| Camera | Arducam IMX477* | 1 | 133.83 | 133.83 |
| Computer | NVIDIA Jetson Nano 4GB | 1 | 140.00 | 140.00 |
| Flash Storage | Sandisk 128GB U3 A2 | 1 | 17.46 | 17.46 |
| Flight Controller | Orange Cube ADS-B* | 1 | 485.00 | 485.00 |
| Speed Sensor | PX4 Airspeed Sensor | 2 | 36.32 | 72.64 |
| GPS Module | HEX Here 3* | 1 | 235.95 | 235.95 |
| Electric Motor | T-motor AS2814KV1200 | 1 | 31.00 | 31.00 |
| ESC | HobbyWing SkyWalker 60A | 1 | 27.99 | 27.99 |
| LiPo Battery | Tattu 3S 2300 mAh | 2 | 23.99 | 47.98 |
| Wifi Receiver | Waveshare AC8265 | 1 | 24.99 | 24.99 |
| Servo Motor | TowerPro MG90S | 5 | 3.24 | 16.20 |
| RC Controller | TX16S (ExpressLRS)* | 1 | 249.99 | 249.99 |
| Signal Splitter | Futaba Y splitter | 1 | 2.48 | 2.48 |
| Radio Modem | RFD 868X TXMOD* | 1 | 423.00 | 423.00 |
| Cooling Fan | Noctua Fan A6X25 | 1 | 16.90 | 16.90 |
| Cooling Fan | Noctua Fan A4X20 | 1 | 15.90 | 15.90 |
| Cables | Pixhawk Cable Bundle | 1 | 22.00 | 22.00 |
| Cables | RFD 868X to Pixhawk | 1 | 5.99 | 5.99 |
| Cables | MIPI-CSI 22-to-15 pin | 1 | 3.63 | 3.63 |
| Stand | Here 3 iStand | 1 | 19.00 | 19.00 |
| UBEC | HobbyWing 25A HV UBEC* | 1 | 57.99 | 57.99 |
| Power Module | Power Brick Mini* | 1 | 32.00 | 32.00 |
| Servo Tester | ToolKitRC ST8 | 1 | 63.90 | 63.90 |
| Sub-total | | | | 2,145.64 |

Table 7.3: Budget breakdown for electronic components.

The total cost for this thesis amounts to **5515.31 €**. This total does not include the author's invested time, nor the fuel charges or fees for traveling to and from the airfield to conduct flight tests.

Licenses for software such as Autodesk Fusion or MATLAB were also not included in the budget, as student versions were used, which did not incur any direct costs for the development of this work. However, it is worth mentioning that these programs are not free outside of an academic environment.

[13] All presented data refers to real invoices except where marked with an asterisk (*), indicating that the material was lent to the author by an external entity.

## 7.4 Word of Caution

This thesis seeks to push further the development of alternative navigation systems for $UAV$s operating in $GPS$-denied environments. It does so by offering an open source solution that enables experimental testing of $VIO$-based algorithms in fixed-wing or hybrid configurations.

While the motivation behind this work is to advance $UAV$ navigation for applications such as search and rescue, disaster response, or package delivery, there is an ethical responsibility to acknowledge the potential for unintended uses of this technology.

It is important to recognize that technological innovations, while often intended for constructive purposes, can be adapted for applications that conflict with their original intent. In particular, the use of $GPS$-independent navigation systems in military contexts, including their potential deployment in autonomous or semi-autonomous weapons systems, raises significant ethical concerns. The prospect of these technologies being utilized in conflict zones, particularly in weaponized drones or other offensive capacities, is deeply troubling and contrary to the principles of this work.

This cautionary note serves as a reminder that **scientific research and technological development must always be guided by ethical considerations**. The potential for misuse of this technology should prompt reflection among students, researchers, and other interested parties. It is crucial that efforts are made to ensure that advancements in this field are directed toward peaceful and humanitarian applications, rather than contributing to the escalation of conflict and violence.

The scientific community and the aerospace industry are encouraged (and have the responsibility) to take proactive steps to prevent the misuse of these technologies, emphasizing their application in areas that provide value to the human race and address global challenges.

In this way the author hopes and expects that this work will be used in ways that promote safety, security, and the well-being of society as a whole.

# 8 Conclusions

This work presented a comprehensive framework to enable future researchers to test alternative navigation techniques on lightweight fixed-wing $UAV$s operating in $GPS$-denied scenarios. The framework includes a detailed software and hardware systems architecture, which extends down to the pinout of the components, as well as the choice of an aerial vehicle to serve as platform for further testing.

Chapter 4 provided a thorough review of the navigation problem in $GPS$-compromised situations. It began with an in-depth historical background, emphasizing key events and their ramifications across different industries. This historical context was followed by a meticulous examination of alternative navigation methods, with a particular focus on their application to $UAV$s.

By analyzing and building on the current state of the art, both an aerial platform and an integrated onboard systems module were proposed. This approach ensured that the proposed solution includes all the necessary tools to compare different implementations against a ground truth or each other. Special emphasis was placed on visual-inertial methodologies that leverage high-resolution imagery from camera feeds, which requires specialized hardware for processing these data-intensive pipelines.

The objectives outlined in Section 1.3 were considered at every stage of the methodology described in Chapter 5, which was based on $INCOSE$'s V-model—a widely adopted framework in aerospace and systems engineering. This approach ensured that a set of requirements was established from the outset and reviewed at each step of the development process as the project progressed. Following the design phase of the V-model, the manufacturing of both the aircraft and the system enclosure box was carried out. Consequently, this methodology facilitated precise monitoring of project progress and adherence to the established specifications throughout the development lifecycle.

Finally, the results presented in Chapter 6 demonstrated the successful execution of flight tests, confirming the suitability of the chosen aerial platform and verifying its capability for stable flight. Additionally, the proposed hardware and software for the alternative navigation module were both validated and integrated into the aircraft's fuselage via a custom-made enclosure, making the system fully operational and ready for further testing.

## 8.1 Future Work

The results presented in this work highlight an exciting research opportunity in navigation systems, enabling the implementation and experimental testing of various algorithms based on visual-inertial odometry. Some ideas for future work are listed below:

- **Algorithm Comparison**: Conduct flight tests comparing various algorithms to determine which one most effectively reduces position error for fixed-wing aircraft in outdoor environments.
- **Camera Gimbal Implementation**: Integrate a servo-actuated camera gimbal to control the pitch of the main camera based on the aircraft's angle of attack, ensuring nadir-oriented positioning of the image at all times.
- **Effects of Lens Field of View (FOV)**: Analyze how the lens Field of View ($FOV$) affects position error estimation for different algorithm implementations.
- **Event Cameras**: Study how event cameras can benefit navigation and obstacle avoidance due to their high-speed tracking capabilities.
- **RTK GPS Integration**: Integrate RTK GPS for improved position readings and to establish a centimeter-level accurate ground truth benchmark for comparison.
- **Parallel Processing**: Explore parallel processing techniques and consider using CUDA acceleration and multithreading.
- **Graphics Card Upgrade**: Upgrade to a more powerful NVIDIA graphics card, such as the Jetson Orin or Xavier, to boost the processing capabilities of $VIO$ algorithms.
- $LiDAR$ **Integration**: Incorporate $LiDAR$ technology to enhance the accuracy of vertical estimates above ground.
- **Miniaturization of Electronic Components**: Investigate the miniaturization of electronic components to reduce the overall weight and size of the system while maintaining functionality.
- **Cable Management**: Develop strategies for efficient cable management to enhance maintenance ease.

# Bibliography


[1] ICAO, *Emerging and cross-cutting aviation issues — increased use of unmanned aircraft systems (uas)*, https://www.icao.int/annual-report-2021/Pages/emerging-and-cross-cutting-aviation-issues-increased-use-of-unmanned-aircraft-systems-uas.aspx, Accessed on 08/08/2024, 2021.

[2] FAA, *Emerging aviation entrants: Unmanned aircraft systems and advanced air mobility*, https://www.faa.gov/data_research/aviation/aerospace_forecasts/unmanned_aircraft_systems.pdf, Accessed on 08/08/2024, 2024.

[3] J. Linkel, *Unmanned aircraft systems demand and economic benefit forecast study*, Presentation at NASA Armstrong Flight Research Center, Accessed on 08/08/2024, 2019. [Online]. Available: https://ntrs.nasa.gov/api/citations/20190007020/downloads/20190007020.pdf.

[4] *Unmanned aircraft system (uas) service demand 2015-2035: Literature review and projections of future usage*, https://irp.fas.org/program/collect/service.pdf, version 0.1, Accessed on 09/08/2024, Sep. 2013.

[5] NASA, *Autonomous drone navigation system ends reliance on gps*, https://spinoff.nasa.gov/spinoff2020/ps_5.html, Accessed on 08/08/2024, 2020.

[6] EASA, *Safety information bulletin: Global navigation satellite system outage and alterations leading to communication / navigation / surveillance degradation*, https://ad.easa.europa.eu/blob/EASA_SIB_2022_02_R3.pdf/SIB_2022-02R3_1, Accessed on 08/08/2024, Feb. 2022.

[7] M. Garcia, J. Dolan, and G. Sirigu, *Aireon white paper: Gps interference*, https://aireon.com/wp-content/uploads/2024/05/Aireon-White-Paper_GPS-Interference_May2024.pdf, Accessed on 08/08/2024, May 2024.

[8] FAA, *Safo 24002: Recognizing and mitigating global positioning system (gps) / global navigation satellite system (gnss) disruptions*, https://www.faa.gov/other_visit/aviation_industry/airline_operators/airline_safety/safo/all_safos/SAFO24002.pdf, Accessed on 08/08/2024, Jan. 2024.

[9] M. M. Miller *et al.*, *Navigation in gps denied environments: Feature-aided inertial systems*, https://www.sto.nato.int/publications/STO%20Educational%20Notes/RTO-EN-SET-116-2011/EN-SET-116(2011)-07.pdf, Accessed on 08/08/2024, 2011.

[10] European Commision Directorate-General for Communication, "Ai act enters into force," *European Commission News*, Aug. 2024, Accessed on 08/12/2024. [Online]. Available: https://commission.europa.eu/news/ai-act-enters-force-2024-08-01_en.

[11] European Union, *Regulation (eu) 2024/1689 of the european parliament, artificial intelligence act*, https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=OJ:L_202401689, Accessed on 08/12/2024, Jun. 2024.

[12] EASA, *Commission implementing regulation (eu) 2019/947*, https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32019R0947, Accessed on 10/08/2024, 2019.

[13] EASA, *Open category - low risk - civil drones*, https://www.easa.europa.eu/en/domains/drones-air-mobility/operating-drone/open-category-low-risk-civil-drones, Accessed on 10/08/2024, 2024.

[14] EASA, *Commission delegated regulation (eu) 2019/945*, https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32019R0945, Accessed on 10/08/2024, 2019.

[15] P. S. Ramesh and J. M. L. Jeyan, "Comparative analysis of fixed-wing, rotary-wing and hybrid mini unmanned aircraft systems (uas) from the applications perspective," *INCAS Bulletin*, vol. 14, no. 1, p. 13, 2022. [Online]. Available: https://bulletin.incas.ro/files/ramesh__muruga-lal-jeyan__vol_14_iss_1.pdf.

[16] *Uav mapping guidelines*, https://uav-guidelines.openaerialmap.org/pages/05-choosing-the-right-uav/, Accessed on 11/08/2024, 2019.

[17] *Gps system: Space segment*, https://www.gps.gov/systems/gps/space/, Accessed on 12/08/2024, 2022.

[18] U. S. G. A. Office, "Gps disruptions: United States Department of Transportation (*DOT*) Could improve efforts to identify interference incidents and strengthen resilience," United States Government Accountability Office (*GAO*), Dec. 2022, Accessed on 12/08/2024. [Online]. Available: https://www.gao.gov/assets/gao-23-105335.pdf.

[19] Honeywell Aerospace, *Honeywell alternative navigation*, https://aerospace.honeywell.com/content/dam/aerobt/en/documents/landing-pages/brochures/Aero-Honeywell-Alt-Navigation-Brochure.pdf, Accessed on 08/12/2024, 2023.

[20] L. Zhuang, X. Zhong, L. Xu, C. Tian, and W. Yu, "Visual slam for unmanned aerial vehicles: Localization and perception," *Sensors*, vol. 24, no. 10, 2024. DOI: 10.3390/s24102980.

[21] S. Joshi, *No signal: Combat aviation in the time of gps denial*, https://stratpost.com/no-signal-combat-aviation-in-the-time-of-gps-denial/, Accessed on 08/12/2024, Feb. 2022.

[22] Alpha Unmanned Systems SL, *Challenges & solutions: Navigation in gnss-denied environments for small unmanned helicopters*, https://www.unmannedsystemstechnology.com/wp-content/uploads/2023/06/Challenges-Solutions-Navigation-in-GNSS-Denied-Environments-for-Small-Unmanned-Helicopters-.pdf, Accessed on 08/12/2024, Jun. 2023.

[23] C. Lomas, *Gps jamming: The benign, the bad, and the scary*, https://www.flightradar24.com/blog/types-of-gps-jamming/, Accessed on 08/12/2024, Jun. 2024.

[24] J. Frard, L. Martin Sacristan, D. Rambach, F. Tranchet, and T. Warns, *Gnss interference*, https://safetyfirst.airbus.com/gnss-interference/, Accessed on 08/12/2024, Sep. 2019.

[25] N. A. W. C. W. Division, "The electronic warfare and radar systems engineering handbook," Naval Air Warfare Center Weapons Division, NAWCWD TP 8347, Sep. 2013, Accessed on 12/08/2024. [Online]. Available: https://apps.dtic.mil/sti/tr/pdf/ADA617071.pdf.

[26] Republic of San Marino Civil Aviation Authority, *Gnss signal disruption*, https://www.smar.aero/wp-content/uploads/2024/02/SN-02-2024-GPS-SPOOFING.pdf, Safety Notice No. 02/2024 Issue 01, Accessed on 08/12/2024, Feb. 2024.

[27] NOAA - NWS, *Ionospheric scintillation*, https://www.swpc.noaa.gov/phenomena/ionospheric-scintillation, Accessed on 08/12/2024, 2024.

[28] UAV Navigation, *Open source vs uav navigation solutions*, https://www.uavnavigation.com/sites/default/files/docs/2021-05/UAV_Navigation_Open_Source.pdf, Accessed on 08/12/2024, May 2021.

[29] S. Sai, A. Garg, K. Jhawar, V. Chamola, and B. Sikdar, "A comprehensive survey on artificial intelligence for unmanned aerial vehicles," *IEEE Open Journal of Vehicular Technology*, vol. PP, pp. 1–26, Jan. 2023. DOI: 10.1109/OJVT.2023.3316181.

[30] EASA, *Easa artificial intelligence roadmap 2.0*, https://www.easa.europa.eu/en/downloads/137919/en, Accessed on 08/12/2024, May 2023.

[31] C. Li, "Artificial intelligence technology in uav equipment," Oct. 2021, pp. 299–302. DOI: 10.1109/ICISFall51598.2021.9627359. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9627359.

[32] L. Mohimont, F. Alin, M. Rondeau, N. Vaillant-Gaveau, and L. A. Steffenel, "Computer vision and deep learning for precision viticulture," *Agronomy*, vol. 12, Oct. 2022. DOI: 10.3390/agronomy12102463.

[33] Airbus, *Artificial intelligence*, https://www.airbus.com/en/innovation/digital-transformation/artificial-intelligence, Accessed on 08/12/2024, 2024.

[34] MLEAP Consortium, "Easa research – machine learning application approval (mleap) final report," EASA, May 2024. [Online]. Available: https://www.easa.europa.eu/en/downloads/139926/en.

[35] W. Xu, "Efficient distributed image recognition algorithm of deep learning framework tensorflow," *Journal of Physics: Conference Series*, vol. 2066, p. 012 070, Nov. 2021. DOI: 10.1088/1742-6596/2066/1/012070.

[36] I. H. Sarker, "Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions," *SN Computer Science*, vol. 2, no. 6, p. 420, 2021. DOI: 10.1007/s42979-021-00815-1. [Online]. Available: https://doi.org/10.1007/s42979-021-00815-1.

[37] L. P. Osco *et al.*, "A review on deep learning in uav remote sensing," *International Journal of Applied Earth Observation and Geoinformation*, vol. 102, p. 102 456, 2021. DOI: https://doi.org/10.1016/j.jag.2021.102456.

[38] "Economic impact to the uk of a disruption to gnss," London Economics, Apr. 2017, Showcase Report, Accessed on 08/12/2024. [Online]. Available: https://londoneconomics.co.uk/wp-content/uploads/2017/10/LEIUK-Economic-impact-to-UK-of-a-disruption-to-GNSS-SHOWCASE-PUBLISH-S2C190517.pdf.

[39] Flight International, *Conflict sees first use of gps jamming*, https://www.flightglobal.com/conflict-sees-first-use-of-gps-jamming/47630.article, Accessed on 08/12/2024, Apr. 2003.

[40] S. Ragan, *Reports say u.s. drone was hijacked by iran through gps spoofing*, https://www.securityweek.com/reports-say-us-drone-was-hijacked-iran-through-gps-spoofing/, Accessed on 08/12/2024, Dec. 2011.

[41] S. Hyon-hee, *Gps jamming highlights n.k. cyber war threat*, http://www.koreaherald.com/view.php?ud=20120508001326, Accessed on 08/12/2024, May 2012.

[42] NASA, *Asrs callback issue 473*, https://asrs.arc.nasa.gov/docs/cb/cb_473.pdf, Accessed on 08/12/2024, Jun. 2019.

[43] D. Goward, *What happened to gps in denver?* https://www.gpsworld.com/what-happened-to-gps-in-denver/, Accessed on 08/12/2024, Sep. 2022.

[44] Z. Liu, J. Blanch, S. Lo, and T. Walter, "Investigation of gps interference events with refinement on the localization algorithm," in *Proceedings of the 2023 International Technical Meeting of The Institute of Navigation*, Long Beach, California, Jan. 2023, pp. 327–338. DOI: 10.33012/2023.18627.

[45] IATA, *Global navigation satellite system: Gnss-radio frequency interference safety risk assessment*, https://www.iata.org/contentassets/c8e90fe690ce4047a8edfa97f4824890/iata_safety_risk_assessment_gnss_interference.pdf, version 3, Accessed on 08/12/2024, Sep. 2023.

[46] IATA, *Easa partners with iata to counter safety threat from gnss spoofing & jamming*, https://www.iata.org/en/pressroom/2024-releases/2024-01-26-01/, Press Release No: 4, Accessed on 08/12/2024, Jan. 2024.

[47] M. Jacobsen, *Ukraine's drone strikes are a window into the future of warfare*, https://www.atlanticcouncil.org/blogs/new-atlanticist/ukraines-drone-strikes-are-a-window-into-the-future-of-warfare/, Accessed on 08/12/2024, Sep. 2023.

[48] M. Hunder, *Ukraine rushes to create ai-enabled war drones*, https://www.reuters.com/technology/artificial-intelligence/ukraine-rushes-create-ai-enabled-war-drones-2024-07-18/, Accessed on 08/12/2024, Jul. 2024.

[49] *Finnish carrier suspends estonia flights after gps interference prevents 2 landings*, https://apnews.com/article/finnair-finland-estonia-gps-interference-flights-c347fe58902b553a936a3efc42e6cc2f, Accessed on 08/12/2024, Apr. 2024.

[50] C. Torralba, *Russia leaves thousands of planes without gps in northern europe*, https://english.elpais.com/international/2024-05-03/russia-leaves-thousands-of-planes-without-gps-in-northern-europe.html, Accessed on 08/12/2024, May 2024.

[51] M. Zee, *Flights misled over position, navigation failure follows*, https://ops.group/blog/gps-spoof-attacks-irs/, Accessed on 08/12/2024, Sep. 2023.

[52] ArduPilot, *Update ros integration for non-gps navigation and off-board path-planning*, https://discuss.ardupilot.org/t/gsoc-2022-update-ros-integration-for-non-gps-navigation-and-off-board-path-planning/86948, Accessed on 08/12/2024, 2022.

[53] ArduPilot, *Gps-denied autonomous exploration with ros 2*, https://discuss.ardupilot.org/t/gsoc-2023-gps-denied-autonomous-exploration-with-ros-2/101121, Accessed on 08/12/2024, 2023.

[54] ArduPilot, *High altitude non-gps position estimation*, https://ardupilot.org/dev/docs/gsoc-ideas-list.html, Blog entry, Accessed on 08/12/2024, Aug. 2024.

[55] R. Cardoso *et al.*, "A review of verification and validation for space autonomous systems," *Current Robotics Reports*, vol. 2, Sep. 2021. DOI: 10.1007/s43154-021-00058-1.

[56] V. Sun *et al.*, "Overview and results from the mars 2020 perseverance rover's first science campaign on the jezero crater floor," *Journal of Geophysical Research: Planets*, vol. 128, Jun. 2023. DOI: 10.1029/2022JE007613.

[57] T. Estlin *et al.*, "Enabling autonomous rover science through dynamic planning and scheduling," Apr. 2005, pp. 385–396, ISBN: 0-7803-8870-4. DOI: 10.1109/AERO.2005.1559331.

[58] Arvidson *et al.*, "Opportunity mars rover mission: Overview and selected results from purgatory ripple to traverses to endeavour crater," *Journal of Geophysical Research*, vol. 116, Feb. 2011. DOI: 10.1029/2010JE003746.

[59] J. Maki *et al.*, "Ingenuity mars helicopter cameras: Description and results," in *Tenth International Conference on Mars 2024 (LPI Contrib. No. 3007)*, Accessed on 08/16/2024, Jet Propulsion Laboratory/California Institute of Technology, Pasadena, CA, 2024. [Online]. Available: https://www.hou.usra.edu/meetings/tenthmars2024/pdf/3479.pdf.

[60] H. Grip *et al.*, "Guidance and control for a mars helicopter," Jan. 2018. DOI: 10.2514/6.2018-1849.

[61] J. Delaune, *Vision-based navigation for mars helicopters: Ingenuity & mars science helicopter*, Presentation at ETH-UZH, Accessed on 08/16/2024, Jet Propulsion Laboratory (NASA), Dec. 2021.

[62] H. F. Grip *et al.*, "Flight control system for nasa's mars helicopter," *Jet Propulsion Laboratory, California Institute of Technology*, 2019, Accessed on 08/16/2024.

[63] B. Balaram *et al.*, "Mars helicopter technology demonstrator," Jan. 2018. DOI: 10.2514/6.2018-0023.

[64] D. Bayard *et al.*, "Vision-based navigation for the nasa mars helicopter," Jan. 2019. DOI: 10.2514/6.2019-1411.

[65] G. Duckworth and E. Baranoski, "Navigation in gnss-denied environments: Signals of opportunity and beacons," in *Military Capabilities Enabled by Advances in Navigation Sensors*, ser. Meeting Proceedings RTO-MP-SET-104, Paper 3, Available from: http://www.rto.nato.int. Accessed on 08/16/2024, RTO, 2007, pp. 3-1–3-14.

[66] J. Zhou, L. He, and H. Luo, "Real-time positioning method for uavs in complex structural health monitoring scenarios," *Drones*, vol. 7, no. 3, 2023. DOI: 10.3390/drones7030212.

[67] DARPA, *Adaptable navigation systems (ans) (archived)*, https://www.darpa.mil/program/adaptable-navigation-systems, Accessed on 08/16/2024.

[68] Z. Li and Y. Zhang, "Constrained eskf for uav positioning in indoor corridor environment based on imu and wifi," *Sensors*, vol. 22, p. 391, Jan. 2022. DOI: 10.3390/s22010391.

[69] W. Nie, Z.-C. Han, M. Zhou, L.-B. Xie, and Q. Jiang, "Uav detection and identification based on wifi signal and rf fingerprint," *IEEE Sensors Journal*, vol. 21, no. 12, pp. 13 540–13 550, 2021. DOI: 10.1109/JSEN.2021.3068444.

[70] Z. Li, D. Yin, X. Xiaojia, D. Tang, C. Zhang, and S. Zhang, "Research on relative positioning system of uavs swarm based on distributed uwb," Nov. 2020, pp. 5561–5566. DOI: 10.1109/CAC51589.2020.9327419.

[71] Y. Yang, J. Khalife, J. J. Morales, and Z. M. Kassas, "UAV waypoint opportunistic navigation in GNSS-denied environments," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 58, no. 1, pp. 663–678, 2022. DOI: 10.1109/TAES.2021.3103140.

[72] C. Yang, T. Nguyen, D. Qiu, J. Casper, and M. Quigley, *Positioning with mixed signals of opportunity*, Jan. 2011.

[73] X. Ma, S. Djouadi, S. Sahyoun, P. Crilly, and S. Smith, "Navigation in gps-denied environments," Aug. 2011, pp. 1–7.

[74] J. Hardy *et al.*, "Unmanned aerial vehicle relative navigation in gps denied environments," in *2016 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, IEEE, 2016, pp. 344–352.

[75] N. J. Abu-Alrub and N. A. Rawashdeh, *Radar odometry for autonomous ground vehicles: A survey of methods and datasets*, 2023. [Online]. Available: https://arxiv.org/abs/2307.07861.

[76] E. B. Quist and R. W. Beard, "Radar odometry on small unmanned aircraft," Brigham Young University, Provo, UT, USA, Tech. Rep., 2013, Accessed on: August 16, 2024. [Online]. Available: http://www.et.byu.edu/~beard/papers/preprints/QuistBeard13.pdf.

[77] M. Mostafa, S. Zahran, A. Moussa, N. El-Sheimy, and A. Sesay, "Radar and visual odometry integrated system aided navigation for uavs in gnss denied environment," *Sensors*, vol. 18, no. 9, 2018. DOI: 10.3390/s18092776.

[78] E. B. Quist and R. W. Beard, "Radar odometry on fixed-wing small unmanned aircraft," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, no. 1, pp. 396–410, 2016. DOI: 10.1109/TAES.2015.140186.

[79] T. White, J. Wheeler, C. Lindstrom, R. Christensen, and K. Moon, *Gps-denied navigation using sar images and neural networks*, Oct. 2020.

[80] R. Kapoor, A. Gardi, and R. Sabatini, "Acoustic positioning and navigation system for gnss denied/challenged environments," Apr. 2020, pp. 1280–1285. DOI: 10.1109/PLANS46316.2020.9110156.

[81] K. M. Cabral, S. R. Barros dos Santos, C. L. Nascimento, and S. N. Givigi, "Alos: Acoustic localization system applied to indoor navigation of uavs," in *2019 International Conference on Communications, Signal Processing, and their Applications (ICCSPA)*, 2019, pp. 1–6. DOI: 10.1109/ICCSPA.2019.8713689.

[82] D. G. Davies, R. C. Bolam, Y. Vagapov, and P. Excell, "Ultrasonic sensor for uav flight navigation," in *2018 25th International Workshop on Electric Drives: Optimization in Control of Electric Drives (IWED)*, 2018, pp. 1–7. DOI: 10.1109/IWED.2018.8321389.

[83] L. Tsay *et al.*, "An acoustic based local positioning system for dynamic uav in gps-denied environments," *Applied Engineering in Agriculture*, vol. 39, pp. 315–323, Jan. 2023. DOI: 10.13031/aea.15397.

[84] J. Braga, H. Campos Velho, and E. Shiguemori, "Estimation of uav position using lidar images for autonomous navigation over the ocean," Dec. 2015, pp. 811–816. DOI: 10.1109/ICSensT.2015.7438508.

[85] R. Milijas, L. Markovic, A. Ivanovic, F. Petric, and S. Bogdan, "A comparison of lidar-based slam systems for control of unmanned aerial vehicles," in *2021 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2021, pp. 1148–1154. DOI: 10.1109/ICUAS51884.2021.9476802.

[86] C. Spruit and C. Cockroft, *Laser based communications: Reducing reliance on satellites*, QinetiQ Ltd, Principal engineer (Electronics) with contributions by Dr. Colin Cockroft, Feb. 2022.

[87] H. Qin *et al.*, "A stereo and rotating laser framework for uav navigation in gps denied environment," in *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, 2016, pp. 6061–6066. DOI: 10.1109/IECON.2016.7793246.

[88] J. D. Carroll and A. J. Canciani, "Terrain-referenced navigation using a steerable-laser measurement sensor," *NAVIGATION: Journal of the Institute of Navigation*, vol. 68, no. 1, pp. 115–134, 2021. DOI: 10.1002/navi.406.

[89] UAV Navigation, *What is an imu?* Accessed: August 16, 2024, 2024. [Online]. Available: https://www.uavnavigation.com/products/ahrs-imu/what-is-an-imu.

[90] K. Gade, *Introduction to inertial navigation and kalman filtering*, Tutorial for IAIN World Congress, Stockholm, Sweden, Oct. 2009, 2009.

[91] E. Petritoli, F. Leccese, and M. Leccisi, "Inertial navigation systems for uav: Uncertainty and error measurements," in *2019 IEEE 5th International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, 2019, pp. 1–5. DOI: 10.1109/MetroAeroSpace.2019.8869618.

[92] S. Seliquini, "Implementation of an extended kalman filter using inertial sensor data for uavs during gps denied applications," Master of Science (MS) Thesis, Old Dominion University, 2022. DOI: 10.25777/f3f4-b307.

[93] L. Lasmadi, A. I. Cahyadi, S. Herdjunanto, and R. Hidayat, "Inertial navigation for quadrotor using kalman filter with drift compensation," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 7, no. 5, pp. 2596–2604, 2017. DOI: 10.11591/ijece.v7i5.pp2596-2604.

[94] G. Wang *et al.*, "A gnss/ins integrated navigation algorithm based on kalman filter," *IFAC-PapersOnLine*, vol. 51, no. 17, pp. 232–237, 2018, 6th IFAC Conference on Bio-Robotics BIOROBOTICS 2018. DOI: https://doi.org/10.1016/j.ifacol.2018.08.151.

[95] Y. Lu, X. Zhucun, G.-S. Xia, and L. Zhang, "A survey on vision-based uav navigation," *Geo-spatial Information Science*, pp. 1–12, Jan. 2018. DOI: 10.1080/10095020.2017.1420509.

[96] A. H. A. Al Kaff, J. M. Armingol Moreno, and A. D. L. Escalera Hueso, "A vision-based navigation system for unmanned aerial vehicles (uavs)," *Integrated Computer-Aided Engineering*, vol. 26, no. 3, pp. 297–310, Apr. 2019. DOI: 10.3233/ICA-190601.

[97] L. Belmonte, R. Morales, and A. Fernández-Caballero, "Computer vision in autonomous unmanned aerial vehicles—a systematic mapping study," *Applied Sciences*, vol. 9, p. 3196, Aug. 2019. DOI: 10.3390/app9153196.

[98] N. Gyagenda, J. Hatilima, H. Roth, and V. Zhmud, "A review of gnss-independent uav navigation techniques," *Robotics and Autonomous Systems*, vol. 152, p. 104 069, Feb. 2022. DOI: 10.1016/j.robot.2022.104069.

[99] Y. Chang, Y. Cheng, U. Manzoor, and J. Murray, "A review of uav autonomous navigation in gps-denied environments," *Robotics and Autonomous Systems*, vol. 170, p. 104 533, Sep. 2023. DOI: 10.1016/j.robot.2023.104533.

[100] M. Y. Arafat, M. Alam, and S. Moh, "Vision-based navigation techniques for unmanned aerial vehicles: Review and challenges," *Drones*, vol. 7, p. 89, Jan. 2023. DOI: 10.3390/drones7020089.

[101] A. Harmat, M. Trentini, and I. Sharf, "Multi-camera tracking and mapping for unmanned aerial vehicles in unstructured environments," *Journal of Intelligent Robotic Systems*, vol. 78, May 2014. DOI: 10.1007/s10846-014-0085-y.

[102] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "Orb-slam: A versatile and accurate monocular slam system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015. DOI: 10.1109/TRO.2015.2463671.

[103] R. Newcombe, S. Lovegrove, and A. Davison, "Dtam: Dense tracking and mapping in real-time," Nov. 2011, pp. 2320–2327. DOI: 10.1109/ICCV.2011.6126513.

[104] J. Engel, T. Schoeps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," vol. 8690, Sep. 2014, pp. 1–16, ISBN: 978-3-319-10604-5. DOI: 10.1007/978-3-319-10605-2_54.

[105] A. H. Al-Kaff, "Vision-based navigation system for unmanned aerial vehicles," PhD dissertation, Universidad Carlos III de Madrid, Leganés, Spain, Oct. 2017. DOI: 10.3233/ICA-190601.

[106] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *IEEE Robotics and Automation Magazine*, vol. 18, pp. 80–92, 2011. DOI: 10.1109/MRA.2011.943233.

[107] H. P. Moravec, "Obstacle avoidance and navigation in the real world by a seeing robot rover," National Aeronautics and Space Administration, Jet Propulsion Laboratory, Ph.D. dissertation, Robotics Institute, Carnegie-Mellon University, Sep. 1980. [Online]. Available: https://www.ri.cmu.edu/pub_files/pub4/moravec_hans_1980_1/moravec_hans_1980_1.pdf.

[108] F. Fraundorfer, D. Scaramuzza, and M. Pollefeys, "A constricted bundle adjustment parameterization for relative scale estimation in visual odometry," in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 1899–1904. DOI: 10.1109/ROBOT.2010.5509733.

[109] M. Sharifi, X. Chen, and C. Pretty, "Experimental study on using visual odometry for navigation in outdoor gps-denied environments," Aug. 2016, pp. 1–5. DOI: 10.1109/MESA.2016.7587182.

[110] Y. Ning, *A comprehensive introduction of visual-inertial navigation*, 2023. [Online]. Available: https://arxiv.org/abs/2307.11758.

[111] R. Leishman, T. McLain, and R. Beard, "Relative navigation approach for vision-based aerial gps-denied navigation," May 2013, pp. 343–352, ISBN: 978-1-4799-0815-8. DOI: 10.1109/ICUAS.2013.6564707.

[112] S. Zhao, F. Lin, K. Peng, B. Chen, and T. Lee, "Homography-based vision-aided inertial navigation of uavs in unknown environments," Aug. 2012, ISBN: 978-1-60086-938-9. DOI: 10.2514/6.2012-5033.

[113] T. Nguyen, G. Mann, A. Vardy, and R. Gosine, "Ckf-based visual inertial odometry for long-term trajectory operations," *Journal of Robotics*, vol. 2020, pp. 1–14, Jun. 2020. DOI: 10.1155/2020/7362952.

[114] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," vol. 3951, Jul. 2006, ISBN: 978-3-540-33832-1. DOI: 10.1007/11744023_34.

[115] J. Shi and C. Tomasi, "Good features to track," *Proceedings / CVPR, IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 600, Mar. 2000. DOI: 10.1109/CVPR.1994.323794.

[116] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision (ijcai)," vol. 81, Apr. 1981.

[117] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," vol. 24, no. 6, pp. 381–395, Jun. 1981. DOI: 10.1145/358669.358692.

[118] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, no. 1, pp. 185–203, 1981. DOI: https://doi.org/10.1016/0004-3702(81)90024-2.

[119] K. McGuire, G. de Croon, C. de Wagter, B. Remes, K. Tuyls, and H. Kappen, "Local histogram matching for efficient optical flow computation applied to velocity estimation on pocket drones," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2016. DOI: 10.1109/icra.2016.7487496. [Online]. Available: http://dx.doi.org/10.1109/ICRA.2016.7487496.

[120] K. Hu, X. Wu, H. Xu, J. Li, and J. Li, "An optical-flow-assisted navigation method for uavs using sage-husa ekf," in *Automatic Control and Emerging Technologies*, H. El Fadil and W. Zhang, Eds., Springer Nature Singapore, 2024, pp. 14–28, ISBN: 978-981-97-0126-1.

[121] L. Huang, J. M. Song, P. H. Chen, and G. H. Cai, "Visual navigation for uav using optical flow estimation," in *Proceedings of the 33rd Chinese Control Conference*, 2014, pp. 816–821. DOI: 10.1109/ChiCC.2014.6896732.

[122] W. Ding *et al.*, "Adding optical flow into the gps/ins integration for uav navigation," Jan. 2009.

[123] N. Gageik, M. Strohmeier, and S. Montenegro, "An autonomous uav with an optical flow sensor for positioning and navigation," *International Journal of Advanced Robotic Systems*, vol. 10, p. 1, Jan. 2013. DOI: 10.5772/56813.

[124] H. Chao, Y. Gu, J. Gross, G. Guo, M. L. Fravolini, and M. R. Napolitano, "A comparative study of optical flow and traditional sensors in uav navigation," in *2013 American Control Conference*, 2013, pp. 3858–3863. DOI: 10.1109/ACC.2013.6580428.

[125] M. Shan, F. Wang, F. Lin, Z. Gao, Y. Tang, and B. Chen, "Google map aided visual navigation for uavs in gps-denied environment," Dec. 2015, pp. 114–119. DOI: 10.1109/ROBIO.2015.7418753.

[126] I. Liubarets, "Celestial navigation as the emergency gnss backup: Enhancing navigational reliability," *Scientific Journal of Gdynia Maritime University*, vol. 129, pp. 7–21, Mar. 2024. DOI: 10.26408/129.01.

[127] G. H. Kaplan, *Current directions in navigation technology*, Symposium on the History and Future of Celestial Navigation, Mystic Seaport, CT, Nov. 2017. [Online]. Available: http://fer3.com/mystic2017/Kaplan-Current-Directions.pdf.

[128] T. Rogoway, *Sr-71's "r2-d2" could be the key to winning future fights in gps denied environments*, Apr. 2019. [Online]. Available: https://www.twz.com/17207/sr-71s-r2-d2-could-be-the-key-to-winning-future-fights-in-gps-denied-environments.

[129] L. Eshleman, *Trust the sky to guide you home*, Accessed: August 16, 2024, Jun. 2023. [Online]. Available: https://insidegnss.com/trust-the-sky-to-guide-you-home/.

[130] M. S. Kim, "Celestial aided inertial navigation by tracking high altitude vehicles," Master's Thesis, Air Force Institute of Technology, Air University, Wright-Patterson Air Force Base, Ohio, 2017. [Online]. Available: https://apps.dtic.mil/sti/tr/pdf/AD1054679.pdf.

[131] A. Rai, *Honeywell successfully demonstrates alternative navigation capabilities in gps-denied environments*, Accessed: August 16, 2024, Apr. 2022. [Online]. Available: https://aerospace.honeywell.com/us/en/about-us/press-release/2022/04/honeywell-demonstrates-alternative-navigation-capabilities.

[132] A. Palmas and P. Andronico, *Deep learning computer vision algorithms for real-time uavs on-board camera image processing*, 2022. [Online]. Available: https://arxiv.org/abs/2211.01037.

[133] A. Loquercio, A. I. Maqueda, C. R. del-Blanco, and D. Scaramuzza, "Dronet: Learning to fly by driving," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1088–1095, 2018. DOI: 10.1109/LRA.2018.2795643.

[134] P. Amini Rad, D. Hofmann, S. Mendez, and D. Goehringer, "Optimized deep learning object recognition for drones using embedded gpu," Sep. 2021, pp. 1–7. DOI: 10.1109/ETFA45728.2021.9613590.

[135] D. Palossi, A. Loquercio, F. Conti, E. Flamand, D. Scaramuzza, and L. Benini, "A 64mw dnn-based visual navigation engine for autonomous nano-drones," *IEEE Internet of Things Journal*, 2019. DOI: 10.1109/JIOT.2019.2917066.

[136] C. Kyrkou, G. Plastiras, T. Theocharides, S. I. Venieris, and C.-S. Bouganis, "Dronet: Efficient convolutional neural network detector for real-time uav applications," in *2018 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2018, pp. 967–972. DOI: 10.23919/DATE.2018.8342149.

[137] N. Messikommer, G. Cioffi, M. Gehrig, and D. Scaramuzza, "Reinforcement learning meets visual odometry," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. [Online]. Available: https://rpg.ifi.uzh.ch/docs/ECCV24_Messikommer.pdf.

[138] S. Wang, R. Clark, H. Wen, and N. Trigoni, "Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, May 2017. DOI: 10.1109/icra.2017.7989236.

[139] C. Chen, B. Wang, C. X. Lu, N. Trigoni, and A. Markham, *Deep learning for visual localization and mapping: A survey*, 2023. [Online]. Available: https://arxiv.org/abs/2308.14039.

[140] H.-J. Liang, N. J. Sanket, C. Fermüller, and Y. Aloimonos, *Salientdso: Bringing attention to direct sparse odometry*, 2018. [Online]. Available: https://arxiv.org/abs/1803.00127.

[141] L. Cartolano and E. Colombini, "Deep learning for visual odometry," Relatório Técnico - IC-PFG-20-16, Projeto Final de Graduação, M.S. thesis, Universidade Estadual de Campinas, Instituto de Computação, Aug. 2020. [Online]. Available: https://www.ic.unicamp.br/~reltech/PFG/2020/PFG-20-16.pdf.

[142] F. Mahlknecht *et al.*, *Exploring event camera-based odometry for planetary robots*, 2022. [Online]. Available: https://arxiv.org/abs/2204.05880.

[143] "Visual-inertial odometry with an event camera," Accessed on: August 16, 2024, Jul. 12, 2023. [Online]. Available: https://patents.google.com/patent/EP3679549B1/en.

[144] A. Navigation, *The future of inertial navigation is classical-quantum sensor fusion*, Jun. 2024. [Online]. Available: https://www.advancednavigation.com/tech-articles/the-future-of-inertial-navigation-is-classical-quantum-sensor-fusion.

[145] T. Neuhauser, "Collaborative navigation – using relative sensor measurements to aid and synchronize the absolute navigation of swarm members," in *STO Meeting Proceedings*, ser. STO-MP-SET-275, NATO Science and Technology Organization, 2023, pp. 1–12. [Online]. Available: https://www.sto.nato.int/publications/STO%20Meeting%20Proceedings/STO-MP-SET-275/MP-SET-275-12.pdf.

[146] V. Polizzi, R. Hewitt, J. Hidalgo-Carrió, J. Delaune, and D. Scaramuzza, "Data-efficient collaborative decentralized thermal-inertial odometry," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10 681–10 688, 2022. DOI: 10.1109/LRA.2022.3194675.

[147] P. Schmuck and M. Chli, "Multi-uav collaborative monocular slam," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3863–3870. DOI: 10.1109/ICRA.2017.7989445.

[148] P. Schmuck, T. Ziegler, M. Karrer, J. Perraudin, and M. Chli, *Covins: Visual-inertial slam for centralized collaboration*, 2021. [Online]. Available: https://arxiv.org/abs/2108.05756.

[149] T. Cieslewski, S. Choudhary, and D. Scaramuzza, *Data-efficient decentralized visual slam*, 2017. [Online]. Available: https://arxiv.org/abs/1710.05772.

[150] P. Zhu, Y. Yang, W. Ren, and G. Huang, "Cooperative visual-inertial odometry," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 13 135–13 141. DOI: 10.1109/ICRA48506.2021.9561674.

[151] P. Zhu, P. Geneva, W. Ren, and G. Huang, *Distributed visual-inertial cooperative localization*, 2021. [Online]. Available: https://arxiv.org/abs/2103.12770.

[152] D. D. Walden, *INCOSE Systems Engineering Handbook*, 5th. Wiley-INCOSE, 2023, pp. 102, 194.

[153] E. S. M. Ebeid, M. Skriver, and J. Jin, "A survey on open-source flight control platforms of unmanned aerial vehicle," Aug. 2017, pp. 396–402. DOI: 10.1109/DSD.2017.30.

[154] I. S. Fernández García *et al.*, *Technical report xtrachallenge 2024 saeta t1*, Accessed: 2024-08-27, Jul. 2024. [Online]. Available: https://xtra2upv.com/informes/xc24/new/11_NEW-min.pdf.

[155] NVIDIA Dev, *Get Started With Jetson Nano Developer Kit*, Accessed: 2024-08-27, 2024. [Online]. Available: https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit#write.

[156] E. Ackerman and M. Koziol, *In the air with zipline's medical delivery drones*, Accessed: 2024-08-27, Apr. 2019. [Online]. Available: https://spectrum.ieee.org/in-the-air-with-ziplines-medical-delivery-drones.

[157] AESA, *Esp-oa-00059/000 operational authorization for fuvex civil s.l.* Feb. 22, 2024. [Online]. Available: https://www.seguridadaerea.gob.es/sites/default/files/Registro%20autorizaciones_EASA.pdf.

[158] FAA, *Faa awards 2.7m in drone research to support disaster preparedness, emergency response*, Aug. 18, 2022. [Online]. Available: https://www.faa.gov/newsroom/faa-awards-27m-drone-research-support-disaster-preparedness-emergency-response.

[159] L. Hall, "Mars aerial and ground global intelligent explorer (maggie)," Jan. 4, 2024. [Online]. Available: https://www.nasa.gov/general/mars-aerial-and-ground-global-intelligent-explorer/.

[160] B. Keeter, "NASA's Dragonfly Rotorcraft Mission to Saturn's Moon Titan Confirmed," Apr. 16, 2024. [Online]. Available: https://science.nasa.gov/missions/dragonfly/nasas-dragonfly-rotorcraft-mission-to-saturns-moon-titan-confirmed/.