“Design and Implementation of an Integrated Navigation Device for Unmanned Aerial Vehicles”

Angelos Antonopoulos

Thesis Committee:
Assoc. Prof. Michail G. Lagoudakis
Prof. Apostolos Dollas
Assoc. Prof. Panagiotis Partsinevelos

Chania, 2021

Technical University of Crete

School of Electrical and Computer Engineering

Bachelor Thesis

"Design and Implementation of an Integrated Navigation Device for Unmanned Aerial Vehicles"

by

Angelos Antonopoulos

Thesis Committee:

Associate Professor Michail G. Lagoudakis
Professor Apostolos Dollas
Associate Professor Panagiotis Partsinevelos

Chania, 2021

Πολυτεχνείο Κρήτης

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Διπλωματική Εργασία

"Σχεδίαση και Υλοποίηση μιας Ολοκληρωμένης Συσκευής Πλοήγησης για Μη Επανδρωμένα Ιπτάμενα Οχήματα"

Άγγελος Αντωνόπουλος

Εξεταστική Επιτροπή:

Αναπληρωτής Καθηγητής Μιχαήλ Γ. Λαγουδάκης
Καθηγητής Απόστολος Δόλλας
Αναπληρωτής Καθηγητής Παναγιώτης Παρτσινέβελος

Χανιά, 2021

Abstract

Nowadays, Unmanned Aerial Vehicles (UAVs), commonly known as “Drones” have gained popularity in our everyday life. The functionality provided by those vehicles allows their usage in many applications. Such applications can be recreational, professional as well as applications which can have a higher impact in our lives. Aerial imaging, 3D mapping, precision agriculture as well as search and rescue missions are just some examples of how UAVs can influence our lives. Many of these scenarios require navigation. The navigation process can be executed by either the operator or an on-board flight command module autonomously. Both cases require continuous spatial awareness throughout the flight. Modern UAVs rely on Global Navigation Satellite Systems (GNSS) for their positioning and navigation. However, numerous scenarios require UAV operation in GNSS denied environments.

The aim of this study is the design and implementation of an integrated UAV navigation system which can be autonomously adjusted to the flight environment, providing spatial awareness to both the UAV and the operator.

The system designed and implemented in this study expands the functionality of UAVs as it allows navigation in different environments, while providing autonomous geolocation abilities. The system was designed as an auxiliary module of a UAV. A custom hexacopter was utilized as a host development and testing platform. Continuous spatial awareness is provided by a multi-tier localization architecture. GNSS estimations, inertial data acquired by an inertial measurement unit (IMU) as well as visual-depth perception data provided by an RGB-D sensor are combined to ensure continuous localization. Moreover, the system facilitates a geolocation module with the ability to autonomously track and estimate the position of objects in the flight area. The autonomous geolocation process is enabled by the use of a three-axis motorized gimbal system which houses a camera, integrated rotary encoders as well as distance measuring equipment. The implemented system was tested and evaluated in both simulated and actual flight conditions in multiple environments.

Περίληψη

Στην εποχή μας, τα συστήματα μη επανδρωμένων αεροσκαφών (ΣμηΕΑ), γνωστά και ως “Drones”, έχουν εισέλθει στην καθημερινότητα μας. Μέσω των δυνατοτήτων τους, μπορούν να χρησιμοποιηθούν για πολλαπλές εφαρμογές οι οποίες καλύπτουν ανάγκες ψυχαγωγίας και διασκέδασης, ανάγκες επαγγελματικής φύσεως, καθώς επίσης και για εφαρμογές που μπορούν να έχουν άμεση και σημαντική επίδραση στην ζωή μας. Αντιπροσωπευτικά παραδείγματα σημερινών εφαρμογών αποτελούν η εναέρια καταγραφή εικόνας, η χαρτογράφηση και τρισδιάστατη αποτύπωση περιοχών και εκτάσεων, η γεωργία ακριβείας καθώς επίσης αποστολές έρευνας και διάσωσης. Ένα μεγάλο σύνολο από τα σενάρια που εκτελούνται απαιτεί την πλοήγηση του οχήματος στον χώρο, είτε μέσω ενός χειριστή είτε μέσω ενός αυτόνομου συστήματος. Προκειμένου να μπορέσει να εκτελεστεί η διαδικασία της πλοήγησης, απαιτείται η διαρκής χωρική επίγνωση κατά την εκτέλεση της πτήσης. Τα σύγχρονα οχήματα βασίζονται κυρίως σε δορυφορικά συστήματα εντοπισμού θέσης για την πλοήγηση τους στον χώρο. Παρά την εύρυθμη λειτουργία αυτών των συστημάτων, τα μη επανδρωμένα οχήματα μπορεί να χρειαστεί να δράσουν σε περιβάλλοντα τα οποία δεν καλύπτονται επαρκώς από δορυφορικά συστήματα.

Σκοπός της συγκεκριμένης εργασίας είναι η σχεδιαση και η υλοποίηση ενός συστήματος πλοήγησης το οποίο θα μπορεί να προσαρμόζεται στις παραμέτρους του εκάστοτε περιβάλλοντος πτήσης. Το σύστημα θα πρέπει να παρέχει στο αεροσκάφος και στον χειριστή αυτού, χωρική πληροφορία σχετικά με το ίδιο το αεροσκάφος καθώς και αντικείμενων που βρίσκονται στην περιοχή της πτήσης.

Το σύστημα το οποίο σχεδιάστηκε και υλοποιήθηκε για τους σκοπούς της συγκεκριμένης εργασίας, επεκτείνει τις δυνατότητες των ΣμηΕΑ καθώς μέσω του αυτού μπορεί να πραγματοποιηθεί πλοήγηση σε πολλαπλά περιβάλλοντα, ενώ παράλληλα παρέχεται και η δυνατότητα αυτόνομου προσδιορισμού θέσης αντικειμένων του χώρου δράσης. Το σύστημα υλοποιήθηκε ως ένα βοηθητικό εξάρτημα για τα ΣμηΕΑ. Το αεροσκάφος το οποίο χρησιμοποιήθηκε ως πλατφόρμα ανάπτυξης και δοκιμής, πρόκειται για ένα πρωτότυπο εξακόπτερο όχημα. Το σύστημα αποκτά χωρική αντίληψη μέσω μίας πολυεπίπεδης αρχιτεκτονικής η οποία συνδυάζει δεδομένα από δορυφορικά συστήματα πλοήγησης (GNSS), αδρανειακές παρατηρήσεις που παρέχονται μέσω του αδρανειακού συστήματος (IMU) του αεροσκάφους, καθώς και οπτικά δεδομένα που περιέχουν τη συνιστώσα του βάθους (RGB-D). Επιπλέον, το σύστημα μπορεί να προσδιορίσει τη θέση αντικειμένων του χώρου πτήσης με συνεχόμενη αυτόνομη στόχευση αυτών. Η δυνατότητα αυτή επιτυγχάνεται με τη χρήση περιστρεφόμενου μηχανισμού στόχευσης ο οποίος φέρει κάμερα ορατού φάσματος, αισθητήρες μέτρησης γωνιών (Encoders) και απόστασης (Rangefinder). Η λειτουργία του συστήματος εξετάστηκε τόσο σε εργαστηριακό περιβάλλον με προσομοιωμένα ερεθίσματα, όσο και σε πραγματικές πτήσεις υπό διάφορες συνθήκες.

Acknowledgements

To begin with, I would like to thank the thesis committee, Associate Professor Michail G. Lagoudakis, Professor Apostolos Dollas and Associate Professor Panagiotis Partsinevelos for their valuable guidance.

I would like to thank the SenseLab research team, especially Panagiotis P., Sanantis K., Dimitris Ch., Stelios M., Stathis B., Theofilos C., Christos L., Ippokratis St., Zisis Ch., Tzanis F., Giorgos P. and Dimos M., my beloved friends and colleagues, not only for the valuable support and help I was given generously, but also for the wonderful moments we have shared and for the precious memories we have made.

Moreover, I would like to dedicate this thesis to my family, my heroes, for their love, endless support and encouragement.

Angelos Antonopoulos, October 2021

Table of Contents

Chapter 1: Introduction

1.1 Unmanned aerial vehicles

Unmanned Aerial Vehicles (UAVs), commonly known as “drones”, are aerial vehicles which do not carry a human operator, and can be utilized in a vast variety of applications. Usually, they are piloted by a remote operator or a ground control station. They are usually equipped with imaging devices which provide real-time optical feedback to the operator, as well as integrated navigation subsystems which enable fully autonomous missions.

Figure 1.1 (Left) Phantom 4 UAV source: DJI Technology Inc. (Right) MQ-1 Predator, source: US Air Force

UAVs are very popular in our everyday life since they are utilized in many different scenarios. They are applicable everywhere from aerial video recording, to military missions. Historically, UAVs have been predominantly used as life threatening devices. Today, they can be used as life saving devices, since they frequently find use in search and rescue missions.

Their broad application spectrum is constantly growing. Over the years, UAVs were only capable of collecting data. Today, through the use of modern embedded systems, UAVs can not only acquire data, but also process data on flight. Despite their small form factor, modern embedded systems used in UAVs have immense processing power. They can effortlessly handle the workload of both remotely piloted or autonomous flights, while continuously ensuring safe maneuvering.

1.2 Brief history of Unmanned Aerial Vehicles

Humankind had always wanted to conquer the skies. In the early stages of aviation, the goal was for humans to take flight. The goal was met on November 21, 1783, when Jean-Francois Pilâtre de Rozier and Francois Laurent, became the first humans to perform flight on a hot air balloon which was invented and created by Joseph Michel and Jacques Etienne Montgolfier. Almost a century later, on December 17, 1903, the brothers Orville and Wilbur Wright, performed the first manned flight on a powered aircraft they designed. [History of Flight Puzzle Piece].

Conquering the skies proved to be more than humans taking flight. The advantages that are provided by utilizing aerial vehicles, seem to be acknowledged even before the historic flight of the Wright brothers. Over the last few years Unmanned Aerial Vehicles have gained popularity. For decades, UAVs have been utilized especially for military applications. Today, we live in an era where we continuously try to unlock new features that UAVs can provide in order to help us in our everyday life. UAV technology has taken massive leaps over the years to become the technology we enjoy today.

Mid 1800s

Similarly to the first manned vehicle, the first historic unmanned aerial vehicle was also designed and constructed as a hot air balloon. UAV history begins in the summer of 1849 [https://www.historytoday.com/archive/bombs-over-venice], when Austria attacked Venice using unmanned balloons, filled with explosives. They were designed in such a way to take advantage of the wind flow in order to passively fly towards the target.

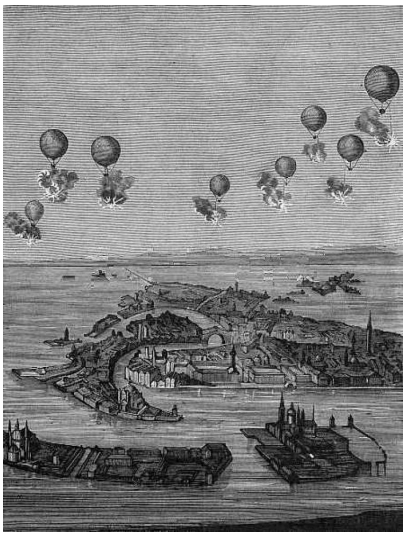


Figure 1.2 Source: History Today

Two attempts were made. The first, carried out on July 12, 1849, failed due to the wind blowing away from the target. On August 22, 1849, a second attempt was made. On that day around 200 balloons were launched, causing panic and minimal damages to the city of Venice [Remote Piloted Aerial Vehicles: An Anthology].

Early 1900s

In 1907, just a few years after the historic flight of Wright brothers, the first quadcopter was created by Jacques and Louis Bréguet brothers, with help of French physiologist Professor Charles Richet. Despite being a manned aircraft which required four operators, it is the inspiration behind the modern unmanned quadcopter and also the helicopter.

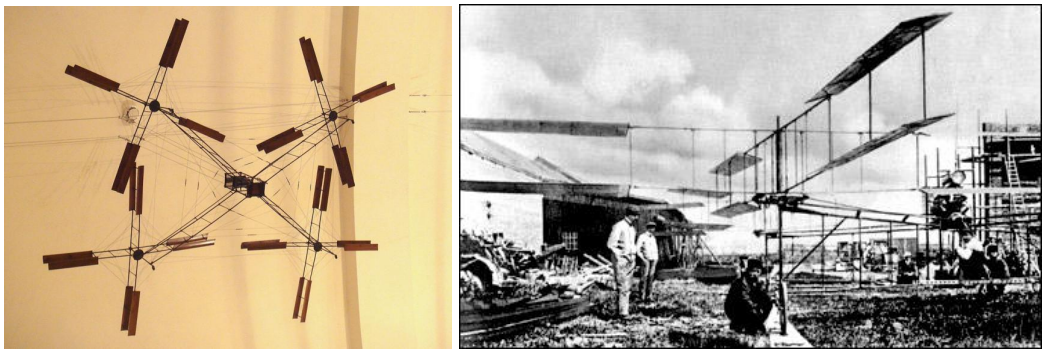


Figure 1.3 Sources: Bréguet Aviation (left), Aviastar (right)

The historic flight was rather unstable, however the aircraft was able to take off and hover 60 centimeters above the ground, proving that such an aircraft frame is capable of flight [Breguet-Richet Gyroplane No.1 helicopter - development history, photos, technical data].

In 1916, just a few years later, the first pilotless aircraft was developed. It was created by British engineer Archibald Low. It was remotely piloted with a radio system, and was used by the British military as an aerial target during World War I. It was commonly known as “Ruston Proctor Aerial Target” [https://interestingengineering.com/a-brief-history-of-drones-the-remote-controlled-unmanned-aerial-vehicles-uavs]. This aircraft became the forerunner of another pilotless aircraft built in 1917 by the United States for offensive purposes. It was no other than the “Kettering Bug” [https://www.nationalmuseum.af.mil/Visit/Museum-Exhibits/Fact-Sheets/Display/Article/198095/kettering-aerial-torpedo-bug/].

Figure 1.4 Source: National Museum of the United States Air Force

That aircraft was able to take off from a four-wheeled dolly that rolled down a portable track. After a predetermined length of time, the aircraft would enter the latest stage of the attack. An electrical circuit would shut off the engine. Then, the wings would be released, causing the aircraft to plunge to the ground. It was filled with 82 kg of explosives which would detonate upon impact.

Mid 1900s

Another breakthrough on UAV applications was made in 1962, when Ryan Aeronautical, a US company known for producing the successful unmanned aerial target aircraft series called “Ryan Firebee”, created the first reconnaissance Unmanned Aerial Vehicle (UAV). The aircraft created was the Ryan Model 147, which was utilized in the Vietnam War [Ryan AQM-34N > National Museum of the United States Air Force™ > Display].

Figure 1.5 (Left) KDA Firebee under the wing of a JD-1 Invader, source: 2-58 KDA-1 KD-4152 on wing of JD-1 4x5 color trans_3 (Right) Ryan AQM-34N (Ryan Model 147) in the Cold War Gallery at the National Museum of the U.S. Air Force, source: U.S. Air Force

This model offers a very low radar cross section given its small size. Moreover, it is a jet powered aircraft. It is very clear that it made a very compelling case for the time period used, as a spy, reconnaissance aircraft.

Despite the fact that UAVs were used for military operations, some remotely piloted recreational aircrafts became available in the 1960s. Transistor technology enabled miniaturised radio circuits to be used in custom

vehicles, created by enthusiasts and hobbyists. Radio Controlled (RC) aircraft clubs also emerged during this decade, signaling the creation of a new industry [A Brief History of Drones: The Remote Controlled Unmanned Aerial Vehicles (UAVs)].

Late 1900s - 2000s

UAV technology was flourishing. Despite being expensive and unreliable to some extent, a whole new field of possibilities had emerged. It was not before the 1980, that the US Government set a goal to build an inexpensive UAV capable of fleet missions [A Brief History of Drones: The Remote Controlled Unmanned Aerial Vehicles (UAVs)]. U.S. and Israel joint forces and were able to develop RQ2 Pioneer, a medium-sized reconnaissance aircraft in 1986.

Figure 1.6 (Left) RQ-2A, reconnaissance UAV, source: Smithsonian

Development of high performance assault UAVs was inevitable. The ability to perform offensive actions at a large scale without the risk of casualties, provides an essential strategic advantage. In 1994, General Atomics Aeronautical Systems was awarded a contract to execute the medium-altitude endurance Predator programme. The first flight was conducted in 1994 and Predator entered production in August 1997.

Today (2021)

UAV technology has come a long way since the day UAVs were pilotless balloons. They have become a part of our everyday life. Their vast application spectrum enables UAVs to possess many forms, everywhere from radio controlled toys, premium video recording tools, all the way to high performance search and rescue modules and military aircrafts.

Figure 1.7 (Left) Auxdron Search and Rescue drone developed by General Drones S.L. source: General Drones S.L. (Right) DJI M600 equipped with RED camera module, source: DJI Technology Inc.

The constant development of both software and hardware will always accompany the development of UAVs. Modern materials and embedded systems will continuously increase flight performance as well as real-time and on board application execution, either in remotely piloted or fully autonomous operations.

1.3 Historic navigation methods

The word ‘navigate’ is derived from the Latin words ‘navis’, meaning ship, and ‘agere’, meaning to move or direct [https://www.vocabulary.com/dictionary/navigation]. But what is actually navigation? Navigation is both a science and an art as indicated by history. Here are some definitions of navigation.

‘The process or activity of accurately ascertaining one's position and planning and following a route’
-Lexico, Oxford English Dictionary
[https://www.lexico.com/definition/navigation]

‘Science of directing a craft by determining its position, course, and distance traveled. Navigation is concerned with finding the way to the desired destination, avoiding collisions, conserving fuel, and meeting schedules’
-Britannica, Encyclopedia
[https://www.britannica.com/technology/navigation-technology]

Over the years, humanity had always wanted to travel. However, in order to be able to travel or even expand the knowledge of previously unknown lands, one must first be able to navigate. Humanity has taken giant leaps towards more efficient as well as more accurate navigation methods. Modern navigation methods support millimeter-level positioning accuracy, and involve artificial orbiting satellites and precision timekeeping with atomic clocks. The technology behind such methods became available just a few decades back. There are many different methods that have been utilized throughout the years over different time frames.

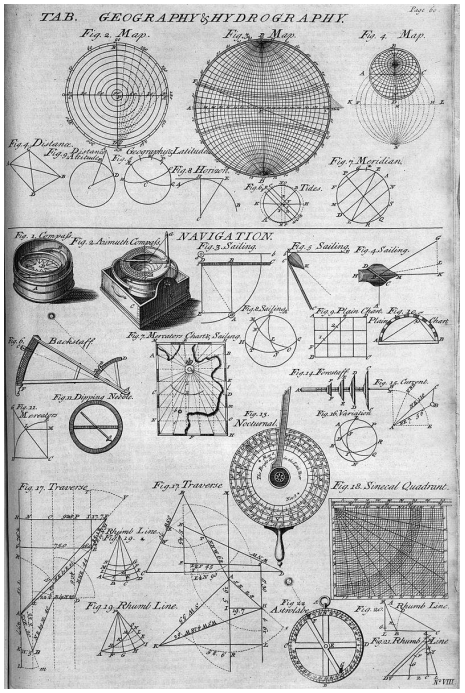


Figure 1.8 Navigation techniques and measurements, source: Cyclopædia

Celestial navigation

Celestial navigation is also known as astronavigation. This type of navigation was well used in ancient times when it was realized that by measuring relative angles between the horizon line and known celestial objects, one could calculate his position on earth along with time. Many tools were invented in order for such calculations to

be performed both precisely and rapidly. Such tools are the astrolabe (220 - 150 B.C) and the octant (1730 A.C.).

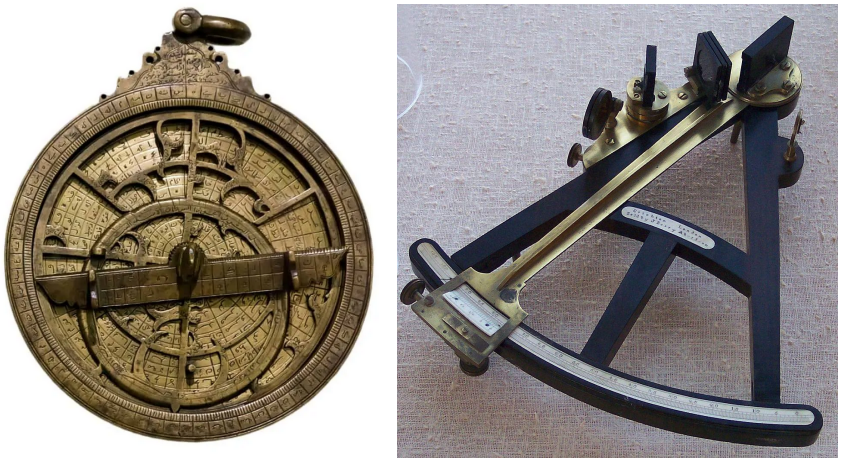


Figure 1.9 (Left) Astrolabe, source: Shutterstock (Right) Octant, source: Wikipedia

Land navigation

This form of navigation refers to the fusion of every available piece of information regarding known position of land features on map, the use of compass as well as distance and angle measurements both on land and map. In order to perform this navigation method, one should choose the appropriate type of map with respect to the environment of interest.

Dead reckoning

Most navigation methods require no prior knowledge. However, sometimes due to lack of navigation equipment or even the inhospitality of the environment, it is impossible to perform absolute positioning. On the contrary, by retaining data regarding last known position, speed and course, dead reckoning can be performed. It is a method which allows for position estimation by utilizing prior knowledge in a cumulative fashion. Yet, such a method is highly susceptible to cumulative errors, and should be performed only in scenarios where absolute positioning methods cannot be used.

Electronic navigation

Nowadays, we can appreciate the fact that we possess the technology needed for precise navigation. Electronic navigation refers to the use of available electronic systems on land, sea and even space, which allow for navigational tasks to be performed. Such systems are satellites, radars and radio aids.

The use of Global Navigation Satellite Systems (GNSS) enabled precise and absolute positioning. Today, many nations have developed independent GNSS. The most dominant are the Global Positioning System (GPS) developed by the United States of America and GLONASS developed by Russia. Cost effective receiver-ends for such systems can pinpoint position with a global position error of a few centimeters. However, some GNSS approaches like Real-Time Kinematics can further lower the error to millimeter scale.

In radio navigation, the execution of navigational tasks relies on signal processing conducted on electromagnetic waves (usually radio waves) produced by one or more transmitters and acquired by one or more receivers. Processing such signals aims to determine relative angles, distance and speed among transmitters and receivers, either in a two dimensional plane or even a three dimensional space.

Radar (*Radio Detection and Ranging*) systems are widely utilized for vessel and aircraft navigation. Radar systems provide relative angle, distance and speed measurements between the sensor and nearby objects or obstacles. In active radar systems, the radar sensor acts as both transmitter and receiver of electromagnetic signals. Frequently, radar sensors are implemented in array configurations.

1.4 Problems addressed

Many UAV designs have been implemented over the years. Some designs suffer from structural aerodynamic instabilities, where other designs are able to self stabilize effortlessly. UAVs currently commercially available, are developed in a way where the installed embedded systems allow for ease of use. They also provide many failsafe mechanisms in order to ensure safe flights, even with amature operators. The key factor which is responsible for the ease of use during flight, is the electronic flight controller's ability to continuously estimate and correct both precisely and rapidly, the desired aircraft's attitude for the requested maneuver. In such an operational scenario, the electronic flight controller receives numerous measurements from many sensors, regarding current flight parameters. Usually, installed sensors provide real-time position, heading, acceleration vectors, angular velocity vectors and height above ground data. After processing all available data, the flight controller is capable of controlling the aircraft's attitude between safe operating margins.

In GNSS available environments, safe flights can be performed since global positioning data are taken into account by the flight controller. However, many flights are performed in GNSS denied environments. When GNSS signals are weak, spoofed or even not available, the flight controller struggles to ensure a safe flight, since desired attitude estimations are missing critical input data. Moreover, the operator will have a hard time navigating the aircraft visually.

Prior knowledge of GNSS availability in an area is in many cases the decisive factor between performing a mission manually with an active operator or autonomously. Despite the immense processing power available on board many UAVs, the lack of localization data is a prohibitive factor for autonomous missions. In some cases, GNSS receivers may become unavailable to provide reliable data during flight, either because of a malfunction or poor signal reception.

Some applications require positioning data not only for navigation, but also for other application specific reasoning. Mapping applications require continuous knowledge of position to ensure adequate coverage, accuracy and precision. In search and rescue missions, being able to find someone in danger is of no use, if this person's position cannot be pinpointed.

This study emphasizes on designing and implementing a navigation module, which is capable of providing localization and positioning data even in GNSS denied environments, for unmanned aircraft systems. This module will be equipped with a multi-tier localization subsystem as well as with another subsystem which will enable the geolocation of nearby objects.

1.5 Structure

In chapter one, introductory material was provided regarding the early stages of UAV development and utilization through different eras. Furthermore, a brief introduction to navigation was also provided, with respect to various navigation approaches.

In chapter two, theoretical background will be presented regarding this study. Knowledge provided in chapter two, will cover the topics of modern navigational procedures and methodologies that are currently in use among UAVs. Moreover, introductory material will be presented regarding embedded systems, sensors and filtering

algorithms, the Robotic Operating System (ROS) which is the framework selected for this study as well as related work.

Chapter three will present the problem statement accompanied by the proposed system's architecture, what this study specifically tackles and what are the key aspects to each problem.

Chapter four is dedicated to the actual design and implementation of the proposed system of this study. Validation of the system will be provided in chapter five, presenting results produced after both indoors and field testing.

At last, conclusions will be drawn in chapter six, regarding both the advantages and the limitations of the system.

Chapter 2: Theoretical Background

2.1 Global Navigation Satellite Systems (GNSS)

Global navigation satellite system (GNSS) is a general term describing any artificial satellite constellation that provides positioning, navigation, and timing (PNT) services on a global or regional basis [Other Global Navigation Satellite Systems (GNSS)]. GNSS today are the state-of-the-art systems regarding positioning and navigation since they provide coverage almost everywhere on our planet. The development of GNSS receivers allows for integration anywhere from smartphones to military aircrafts. Depending on the application, their receivers vary in terms of measurement accuracy and precision, from factor and pricing.

The first form of GNSS was developed in 1973 by the Department of Defense, of the United States of America. The project was given the name Global Positioning System (GPS). The system's inception began in the Sputnik era, when scientists discovered the ability to track the satellite with shifts in its radio signal, known as the "Doppler Effect". That became the foundational idea for modern GPS. The first GPS satellite reached orbit in 1978 and the constellation of 24 satellites required for global coverage became operational by 1993. Today, GPS is well established and is able to pinpoint a three dimensional position to meter-level accuracy and time to the 10-nanosecond level, worldwide. Moreover, continuous development has managed to further extend the number of operational satellites to a constellation of 30 satellites.

GPS comprises three distinct segments: the space segment, control segment and user segment [What is GPS?]. Every GPS satellite is equipped with redundant atomic clocks and tracked by a ground control network. Atomic clocks provide extremely high accuracy timekeeping. Thes can provide time with an accuracy of 100 billionths of a second [https://www.gps.gov/applications/timing/]. Each satellite transmits its position and time at regular intervals and those signals are intercepted by GPS receivers. The receiver is able to determine its position by calculating how long it took for the signals to reach it. Such a calculation is commonly known as multilateration [Multilateration - an overview].

In GNSS at least four satellites are required in order to estimate a three dimensional position. One satellite provides an initial timestamp and the rest can provide time differences in relation to the initial timestamp. The calculation of an unknown position based on exactly three known ranges and initial position measurements is called trilateration. However, the accuracy obtained by taking into account only three measurements is not adequate for the majority of modern applications. This is why more measurements need to be taken into account. Naturally, more measurements translate into more satellites.

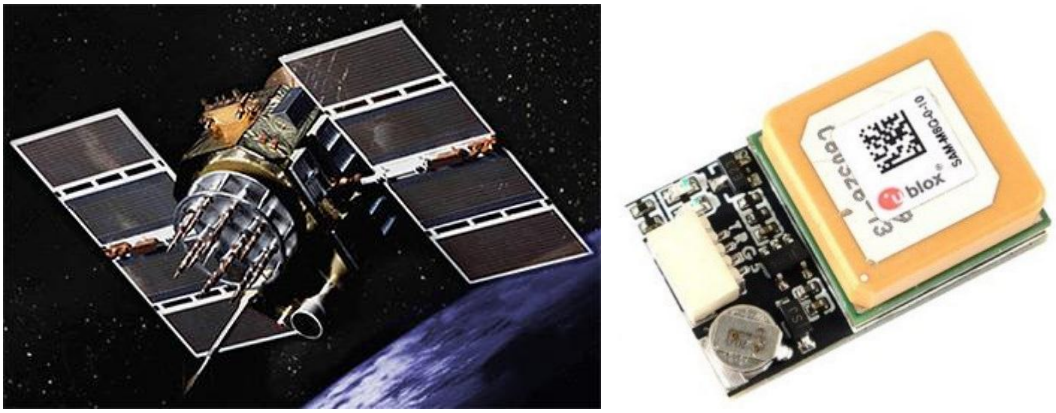


Figure 2.1 (Left) Navstar-2A GPS Satellite, source: US Air Force (Right) GNSS Receiver, source: Ublox

Many nations have developed their own GNSS service to ensure positioning and navigation redundancy. GLONASS (Globalnaya Navigazione Sputnikovaya Sistema, or Global Navigation Satellite System) is a global

GNSS owned and operated by the Russian Federation. BeiDou, or BDS, is a regional GNSS owned and operated by the People's Republic of China and Galileo is a global GNSS owned and operated by the European Union [Other Global Navigation Satellite Systems (GNSS)].

GNSS positioning accuracy depends on many factors. Satellite geometry, signal blockage, receiver design or even atmospheric conditions can affect the overall performance of the system. [GPS Accuracy] Usually the accuracy supplied by the use of GNSS is typically within 5 meters when direct line of sight exists, between the satellites and the receiver. Accuracy worsens in urban environments, near buildings, bridges, and trees since the signals might get blocked or even reflected before getting received, resulting in wrong assumptions regarding the actual distance between the satellites and the receiver.

Sometimes positioning accuracy is a key ingredient in many applications. When higher accuracy is required, high-end dual-frequency receivers and augmentation systems can be used. These can enable real-time positioning within a few centimeters and in some cases at the millimeter level. The most popular technique which enables centimeter or even millimeter accuracy is Real-Time Kinematics (RTK).

Real-Time Kinematics (RTK) [Real Time Kinematics - Navipedia] is a differential GNSS positioning methodology which provides high accuracy performance. RTK requires the existence of a stationary receiver commonly called “Base”, whose position is known and fixed, yet is constantly monitored, and one or more nearby non-stationary receivers called “Rovers”, whose position needs to be calculated. According to the European Space Agency (ESA), the technique is based on the following principles:

- In close range of a clean-sky location, the main errors in the GNSS signal processing are constant. This is why they cancel out when they undergo differential processing. This also applies to the error in the satellite clock bias, the satellite orbital error, the ionospheric delay and the tropospheric delay.
- The noise of carrier measurements is much smaller than the one of the pseudo-code measurements. However, the processing of carrier measurements is subject to the so-called carrier phase ambiguity, an unknown integer number of times the carrier wave length, that needs to be fixed in order to rebuild full range measurements from carrier ones.
- The phase ambiguity can be fixed for dual-frequency differential measurements for two close receivers.

The base station will continuously transmit in real-time its known location, along with the code and carrier measurements regarding L1 and L2 frequencies for every available satellite which is in line of sight with the base, and its signal can be received. By receiving this stream of data, the rover is able to correct phase ambiguities and can calculate its position relative to the base. Furthermore, by adding the global position of the base, the rover can determine its global position as well.

However, this technique has some limitations. To begin with, this methodology can perform adequately in a radius of no more than 20 km from the base. Moreover, a real-time and constant communication channel needs to be kept online throughout the mission of the rover.

2.2 Embedded Systems

Embedded Systems are everywhere. Despite being unseen, they are inside the cars we drive, the planes we travel with, our computers, even inside our coffee makers and washing machines. Embedded systems are designed to accomplish everything from minor tasks to extremely important ones. Modern semiconductor manufacturing methods allow systems to be integrated in the form of microprocessor chips. Embedded systems can be miniaturized enough to be used in nearly every possible application. They can be produced to provide high performance characteristics especially in mission critical scenarios, like controlling an aircraft’s engine. Many definitions have been given regarding embedded systems.

“Embedded systems are information processing systems embedded into a larger product”

“Dortmund” Definition, Peter Marwedel

“Embedded software is software integrated with physical processes. The technical problem is managing time and concurrency in computational systems.”

“Berkeley” Definition, Edward A. Lee

The first modern real-time embedded system was created in the 1960s by Dr. Charles Stark Draper at the Massachusetts Institute of Technology. It was the Apollo Guidance Computer, designed to automatically collect data and provide mission-critical calculations for the Apollo Command Module and Lunar Module [What is an Embedded System? Definition and FAQs].

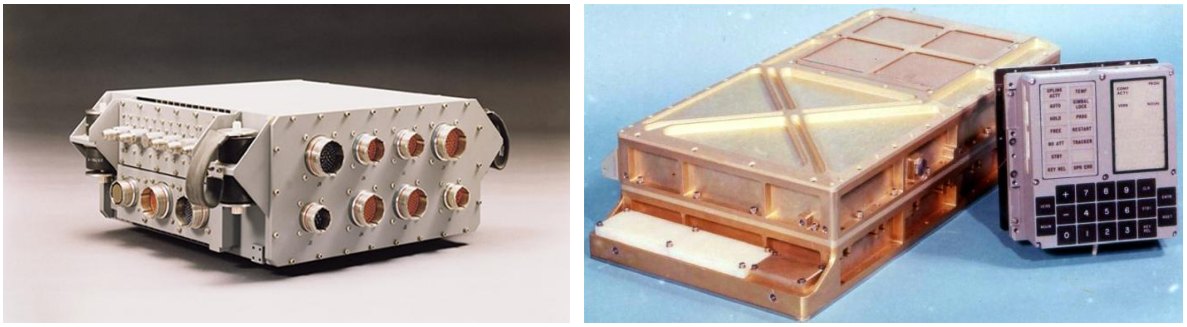


Figure 2.2 (Left) Full Authority Digital Engine Control 2 (FADEC-2) unit which is used to control turbofan engines designed for Airbus and Boeing aircrafts, source: FADEC International, LLC (Right) Apollo Guidance module, was used as the main computer on the Apollo missions, source: Wikipedia

Embedded systems are marvels of modern engineering. They are designed to perform some specific task, hence they provide the optimal balance between processing capability, connectivity, form factor and power for a given application. They sense the outside world with the use of sensors connected to their interfaces. Data provided to them can be processed in real-time. Moreover, in many applications, they interact with the outside world through the use of actuators.

When designing an embedded system, many considerations need to be made. Such considerations can be for both the actual operation of the system as well as the environment in which the embedded system will interact. Risk is also a factor which plays a highly important role in the overall design of such systems. In some cases, human intervention might be impossible and in some others, a single failure of the system might have catastrophic consequences. This is why reliability of embedded systems needs to be evaluated.

Extensive testing needs to be performed when designing embedded systems for large scale and safety-critical applications. To begin with, performance and fault tolerance should be evaluated. The system should be able to execute specific tasks in an acceptable or even specific amount of time, while consuming an adequate amount of energy. When interfacing with peripheral devices, communication related errors might occur. Hence the stability of the system needs to be evaluated even in such cases. Testing should also validate whether the system can operate in the required environmental parameters ranges as is temperature, relative humidity, pressure, acceleration or deceleration, magnetic flux or even exposure to electromagnetic radiation. Nowadays, modern tools allow embedded devices to undergo validation checks even in the design process as well as providing extensive testability of the complete design.

We live in an era where we have successfully implemented embedded systems, which are able to operate at extreme conditions, to be used inside devices sent to space or even other planets, millions of miles away from humanity.

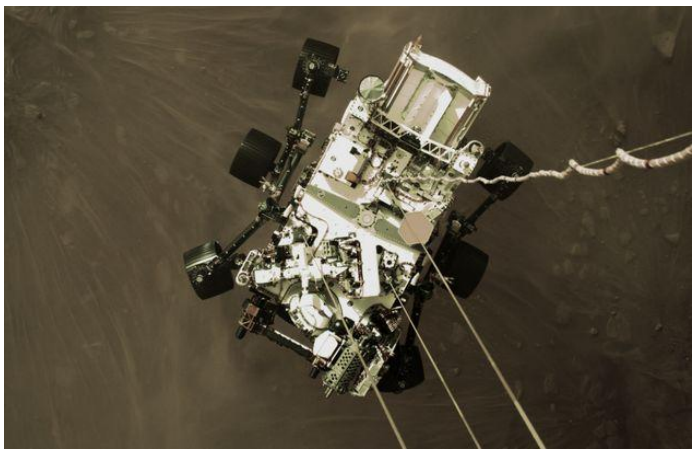


Figure 2.3 Perseverance rover moments before touchdown on the Martian surface, as captured by the descent stage module, source: NASA-JPL

2.3 Inertial Measurement Units (IMUs)

Inertial Measurement Units (IMUs) are sensors which enable the measurement of a body's acceleration, angular velocity and in some cases orientation. Modern IMUs allow three-dimensional measurements. In order for such measurements to be performed IMUs are equipped with distinct components, where each component supplies crucial data. Three axis accelerometers supply acceleration data. Angular velocity data are captured by the use of a three axis gyroscope. When the system is equipped with a three axis magnetometer, orientation data relative to the magnetic north pole can be obtained. When all three components are available, the inertial measurement unit can be defined as an attitude and heading reference system (AHRS).

Today, IMUs can be integrated inside many devices. They play an essential role in aviation since they supply vital information regarding the attitude of the aircraft in real-time and also allow operations during night, or even when Instrument Flight Rules (IFR) apply. The smartphone industry also makes great use of IMUs as every smartphone is equipped with at least one unit. Mobile applications utilize IMU data, either for visual purposes like orienting the phone's screen or operational functionality like navigation.

The three distinct parts of a modern IMU are:

- Accelerometer (3 axis)
- Gyroscope (3 axis)
- Magnetometer (3 axis)

The accelerometer is able to measure the linear acceleration along each one of the three principal axes. Usually the model of XYZ axes is used, where X corresponds to the longitudinal axis of the IMU, Y to the lateral and Z to the vertical. In aviation terms, rotation about the longitudinal axis X is called "Roll", about the lateral axis Y is called "Pitch" and about the vertical axis Z is called "Yaw".

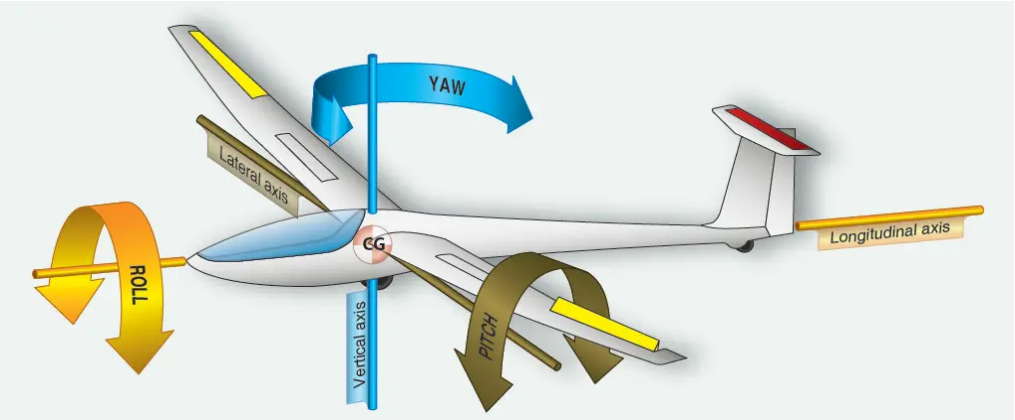


Figure 2.4 The principal axes, source: Flight Literacy

The gyroscope can provide readings regarding the angular velocity as they can sense changes in the rotation about an axis. Gyroscopes can be divided into three different categories:

- Mechanical Gyroscope
- Fibre-Optic Gyroscope (FOG) and Ring Laser Gyroscope (RLG)
- Microelectromechanical systems (MEMS)

Mechanical gyroscopes operate by having a spinning mass rotating about its axis. Its axis is attached to two freely rotating gimbals and then to the gyroscope frame. The mass tends to remain parallel to itself and to oppose any attempt to change its orientation, hence the spinning mass will remain spinning in its initial orientation assuming no drifts occur. The whole gyroscope is able to freely rotate around the spinning mass. By measuring the relative angles among the frame and the gimbals, the gyroscope can produce angular measurements.

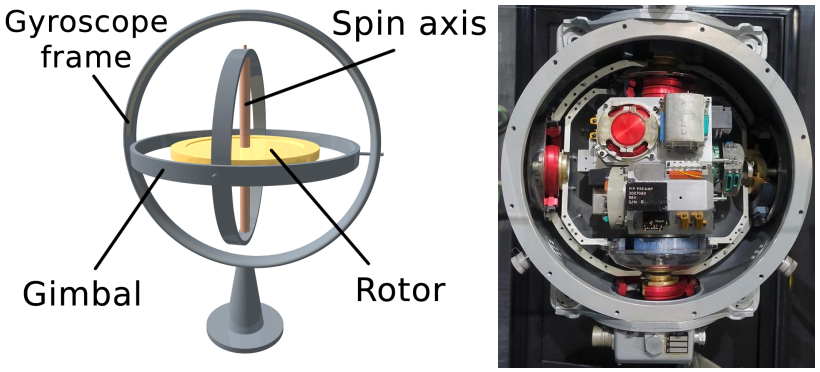


Figure 2.5 (Left) Basic mechanical gyroscope structure, source: Wikipedia (Right) Apollo missions gyroscope model, source: NASA

Fibre-Optic (FOG) and Ring Laser Gyroscopes (RLG) are systems which take advantage of the Sagnac effect. This effect is also known as Sanac interference. In FOG systems, two laser beams are applied to both ends of a fiber-optic cable loop. The beam which is traveling against the rotation results in a shorter path delay when compared to the other beam, producing a differential phase shift. By utilizing interferometry methods, phase shift can be translated into angular velocity. An RLG operates in a similar fashion, with the difference that the fiber optic setup is replaced by a ring laser and mirrors.

Mechanical as well as FOGs or RLGs systems are expensive and cannot come in a form factor adequate for integration inside small UAVs. The technology which launched the usage of gyroscopes and IMUs in general was microelectromechanical systems commercially known as MEMS. MEMS technology allows micrometer

scale mechanical structures to be etched in silicon. These structures contain an etched mass as well and are allowed to move or even rotate. Their shape allows those tiny structures to emulate the operation of a variable capacitor. Depending on the device's operation, while performing translation or rotation, the mass inside these structures forces the gap between these structures to change, resulting in change of capacitance. By electronically measuring the capacitance, the accelerometer can estimate the linear acceleration, whereas the gyroscope can estimate angular velocity. Today, the majority of MEMS sensors integrate a three-axis accelerometer with a three-axis gyroscope.

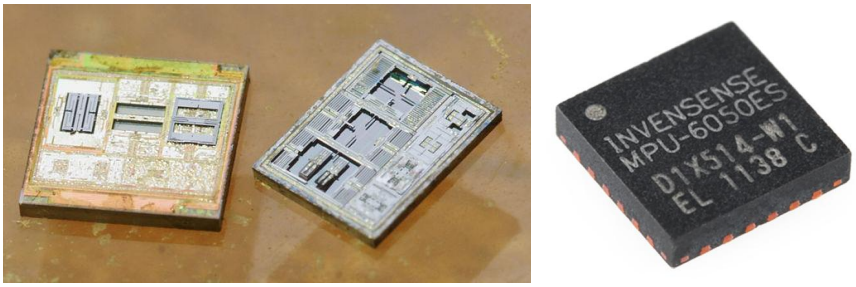


Figure 2.6 (Left) MPU6050 Integrated 3-axis accelerometer and 3-axis gyroscope MEMS sensors exposed circuit, source: zeptobars.ru (Right) MPU6050 IC, source: Sparkfun

The magnetometer is a modern instrument which can measure a magnetic field and sometimes the direction of the field as well. They found practical application in compasses and they complete an IMU as they supply the system with an absolute reading, regarding a three dimensional angle between the IMU and the magnetic north pole.

2.4 Sensor filtering and fusion

When a system like UAV interacts with the outside world, it uses sensors. Different sensors provide readings for a variety of purposes. They can be used to calculate current global position, altitude, attitude, climb or descent rate, indicated airspeed, or even outside air temperature. However, every sensor is susceptible to different types of noise. Sometimes, the noise can provide inaccurate readings and when those readings are vital for conducting a UAV flight, they can force the system towards instability, which might result in crash.

Different types of noises require different noise elimination strategies. When performing a flight, many calculations need to be conducted in order to obtain correct readings which will result in a safe flight. One great example is the pipeline used to estimate an aircraft’s true airspeed (TAS), which is the speed of an aircraft relative to the air it's flying through. True airspeed sensors are simply non-existent. However, aircrafts have installed static ports which provide static air pressure measurements, and pitot tubes which provide total air pressure measurements. By subtracting static pressure from the total pressure, dynamic pressure can be obtained, hence indicated airspeed (IAS) can be calculated. By correcting for both instrument and positional errors, calibrated airspeed (CAS) is determined. At last, by applying corrections for altitude and air temperature, TAS is finally computed. This pipeline requires many different measurements, each of which cumulatively inserts some error.

When trying to perform filtering on sensor measurements, the initial step is usually to check if the readings are logical. Depending on the sensor, there are various ways to determine whether the readings should be accepted or not. At first, the supplied measurement should be within the sensor’s operating limits. A measurement outside the operational limits of the sensor is frequently the result of a faulty sensor. Moreover, another way to further filter measurements by applying logic, is to perform solid estimations and check how the sensed data “follow” the estimated. In the previous example, IAS required sensing both total and static air pressure. It is known that in straight and level flight, static air pressure cannot exceed total air pressure. If the system is “sure” that it is performing straight and level flight, yet the static pressure measured is higher than the total, it is certain that an error has occurred.

In many scenarios logic can play an important role when dealing with accepting data. However, in modern applications which manipulate extensive amounts of data, logic cannot be utilized easily. In data analysis, identifying rare items, events or even suspicious observations is called outlier detection. Today, artificial intelligence and machine learning provide promising tools towards high performance outlier detection.

Frequently, having multiple sensors which simultaneously measure the same attribute can increase the fault tolerance as well as both the overall precision and accuracy of the system. Precision and accuracy refer to different metrics regarding sensory systems. Accuracy describes how close the measured values are to a specific value, whereas precision describes how close the measurements are to each other.

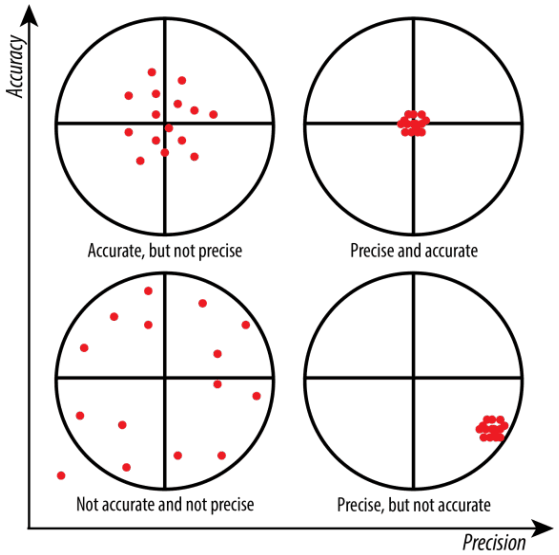


Figure 2.7 Accuracy vs precision, source: St. Olaf College

Mean and median filters

Mean and median filters are easy to implement filters and can be used for simple applications. In a mean filtering technique, an estimation output is given as the average of a defined window of either previous or simultaneous measurements. In a similar fashion, a median filter provides an output which is the median value of those stored values. Mean and median filters are mainly applicable in image processing applications, as they are easy to implement pre-processing filters and usually are applied as N-by-N kernels in pixel regions. Furthermore, mean filtering can further aid smoothing images, whereas median filter can eliminate “salt and pepper noise”.

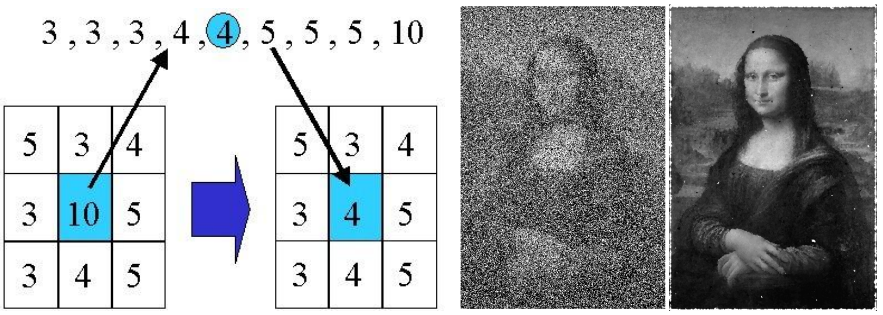


Figure 2.8 (Left) Median filter kernel operation which swaps the central value with the median, source: Researchgate, Vu Hoang Minh, Umeå University (Right) Salt and pepper noise elimination before and after, source: OpenCV-Examples by bveyseloglu, Github

Kalman filter (KF) and Extended Kalman filter (EKF)

The Kalman filter (KF) is a highly popular filtering method in statistics and control theory. This method was named after Rudolf E. Kálmán who was the forerunner of this technology. It is also known as linear quadratic estimation (LQE). Kalman filtering is an algorithm that takes into account not only measurements containing statistical noise but also their captured timestamp. This algorithm produces estimates of unknown variables by assuming a predefined model of a system, which results in being more accurate than single measurements, by continuously estimating a joint probability distribution over the variables for each time period.

The algorithm is split into two stages. Initially, during the first stage which is called “prediction step”, the filter obtains estimations of the current state variables, while also calculating uncertainties. Later, when new measurements become available, the estimates are updated. This stage is called the “update step”. The estimates are updated by using a weighted average among the new measurements and the previous estimates, where more weight is supplied to estimates with higher certainty. This algorithm operates in a recursive fashion, and requires only the present input measurements and the previously calculated state along with an uncertainty matrix.

The basic operation of the filter can be shown bellow:

Prediction step:

$$\hat{\mathbf{x}}_{t|t-1} = \mathbf{F}_t \hat{\mathbf{x}}_{t-1|t-1} + \mathbf{B}_t \mathbf{u}_t$$
$$\mathbf{P}_{t|t-1} = \mathbf{F}_t \mathbf{P}_{t-1|t-1} \mathbf{F}_t^T + \mathbf{Q}_t$$

Update step:

$$\hat{\mathbf{x}}_{t|t} = \hat{\mathbf{x}}_{t|t-1} + \mathbf{K}_t \left(\mathbf{y}_t - \mathbf{H}_t \hat{\mathbf{x}}_{t|t-1} \right)$$
$$\mathbf{K}_t = \mathbf{P}_{t|t-1} \mathbf{H}_t^T \left(\mathbf{H}_t \mathbf{P}_{t|t-1} \mathbf{H}_t^T + \mathbf{R}_t \right)^{-1}$$
$$\mathbf{P}_{t|t} = (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t) \mathbf{P}_{t|t-1}$$

Where,

$\hat{\mathbf{x}}$: Estimated state.

$\mathbf{F}$: State transition matrix (i.e., transition between states).

$\mathbf{u}$: Control variables.

$\mathbf{B}$: Control matrix (i.e., mapping control to state variables).

$\mathbf{P}$: State variance matrix (i.e., error of estimation).

$\mathbf{Q}$: Process variance matrix (i.e., error due to process).

$\mathbf{y}$: Measurement variables.

$\mathbf{H}$: Measurement matrix (i.e., mapping measurements onto state).

$\mathbf{K}$: Kalman gain.

$\mathbf{R}$: Measurement variance matrix (i.e., error from measurements).

This method can also enhance data while it is able to perform sensory data fusion. Many sensors can provide concurrent measurements about an attribute in a system. Given the model of the system the Kalman filter will be able to estimate the uncertainty regarding each sensor over time and will provide an output by applying a weighted average on the estimated data, while applying more weight to data estimated by less noisy sensors.

The KF assumes that the noise embedded in measurements is Gaussian in nature and the system in which the filter is applied is linear. However, many systems are characterized by non-linearities. In such cases extended Kalman filters (EKF) can be utilized, as they provide linearization about an estimate of the current mean and covariance. By taking advantage of both system modeling and time, these filters are currently utilized in many applications such as object tracking systems, guidance and navigation modules, stabilizing depth measurements in computer vision applications or even economics.

Basic KFs or even EKFs are highly applicable in many use cases, especially in aircraft systems and UAV modules where data fusion is mandatory. Flight controllers require continuous as well as high frequency monitoring of many flight related attributes. Such data are provided by many distinct sensors which in some cases are identical modules to ensure redundancy in case a malfunction occurs or a sensor becomes excessively noisy. Moreover, since the model of the system is known, fusing data does not apply only on identical sensors. In modern aircrafts multiple identical pitot tubes are used when measuring the airspeed. These devices are usually equipped with heating elements which prevent icing. If a heating element malfunctions, the tube can get either partially or fully blocked by ice and will definitely produce noisy measurements. By taking into account all the available measurements from every pitot tube sensor, the Kalman filter will provide the flight controller with a fused output regarding total air pressure in which the noisy sensor would be nearly rejected. Another important application of such filters in UAVs is the estimation of the position as well as the velocity of the module in three dimensions. In that scenario, GNSS data are fused with IMU data and the required output is produced.

2.5 Simultaneous Localization and Mapping (SLAM)

Multiple robotic and UAV applications require the estimation of their module's position. Position can describe either a global or a local attribute. By the use of a GNSS sensor, a system is able to estimate its position with relation to the earth whereas by using a lidar (Light Detection and Ranging) sensor, a position relative to nearby obstacles can be estimated. The general term which is used when describing the function of a system when positioning itself in an environment of interest is called localization. Frequently, in such applications, unknown environments require exploration. In order to explore the environment, the system should be able to keep track of what is learned regarding the area while exploring. In other words, to perform dynamic mapping of the area with the ability of expansion during the exploration. In robotic and UAV applications, the problem of performing mapping while concurrently providing localization data, is known as simultaneous localization and mapping (SLAM).

Today, many SLAM algorithms have been implemented and they support a variety of different sensor readings as input. Lidar sensors sense their environment through the use of laser pulses to perceive their surroundings either as a 2D plane or even a 3D space. They measure distances by emitting pulses to distinct directions around the sensor, and measure the time taken for the pulses to return after bouncing in nearby obstacles. Then by taking into account the speed of light, each round-trip duration can be converted into distance. Today, lidar readings alone can be fed to SLAM algorithms and provide reliable localization and mapping data. Hector SLAM [3] is a remarkable implementation of SLAM algorithm, operating on lidar data.

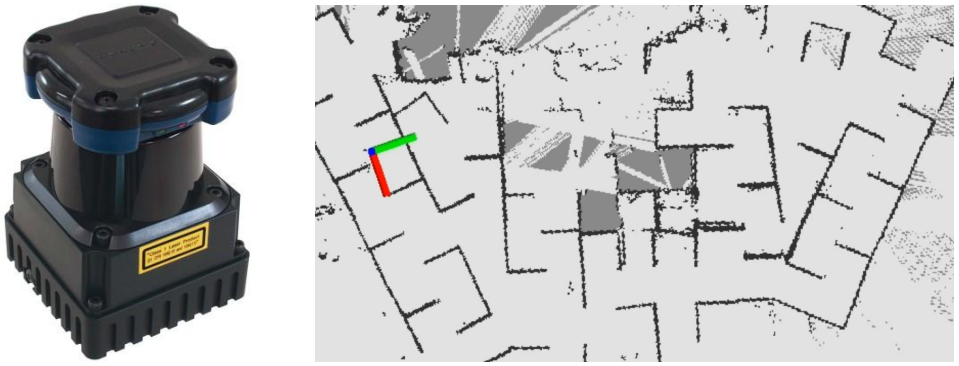


Figure 2.9 (Left) Hokuyo lidar sensor, source: Hokuyo Automatic Co. Ltd (Right) Hector SLAM output, source: Team Hector, TU Darmstadt

Modern SLAM algorithms support optical, or even depth data as input. Such methods are called Visual SLAM (V-SLAM) approaches. Optical data are essentially obtained by the use of either a monocular or a stereo camera sensor. In some cases, three dimensional data acquired by depth cameras or RGB-D sensors can also be injected into a SLAM pipeline. Each sensory data has a different processing pipeline as they refer to different spatial attributes. In this study, ORB-SLAM2 [4] implementation is utilized as a component of the localization submodule.

2.6 Related work

Navigation is a very attractive research field since it is a very helpful tool in our everyday life. Researchers aim to increase the availability of navigational services even in GNSS-challenged environments, as well as to develop methods which enhance precision. Already proposed methods ([5] , [6]) indicate that GNSS measurements can be enhanced when fused with optical and inertial data. Optical data can be converted into attitude and velocity data in real-time. In the proposed studies, such data are fused with position and acceleration measurements, which were collected by GNSS and IMU sensors accordingly. Through the use of Kalman filtering algorithms, global position, attitude, acceleration and velocity data can be provided in a filtered form. Moreover, adapting a Kalman filter can further enhance the overall update rate of the system's inertial state estimation.

Selective Integration of GNSS, Vision Sensor, and INS Using Weighted DOP Under GNSS-Challenged Environments [5] , is a study which provides adequate evidence that integrating the aforementioned data in a selective fashion, can enhance both precision and availability of navigation. As presented in figure, the architecture of the system enables pre-processing of the data, as they need to be evaluated before getting further processing inside the filter.

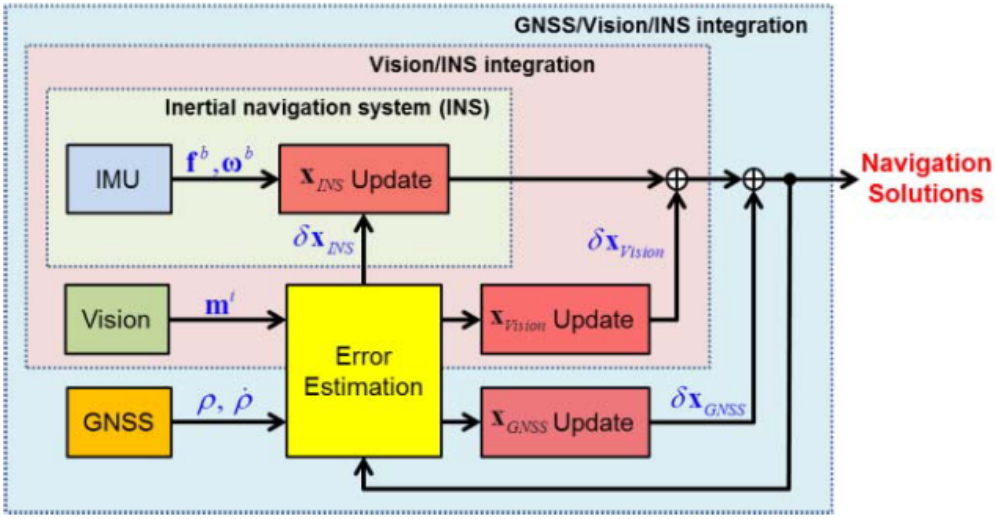


Figure 2.10 Top-Level design of the proposed system.

This study proposes WDOP (weighted dilution of precision), as a quality performance index that indicates the quantity of navigational errors based on the geometrical deployment of all measurements. Selective integration of data is enabled through the use of thresholds calculated for both the GNSS as well as the vision data, after calculating the WDOP. A block diagram of the selected integration can be found below, in figure.

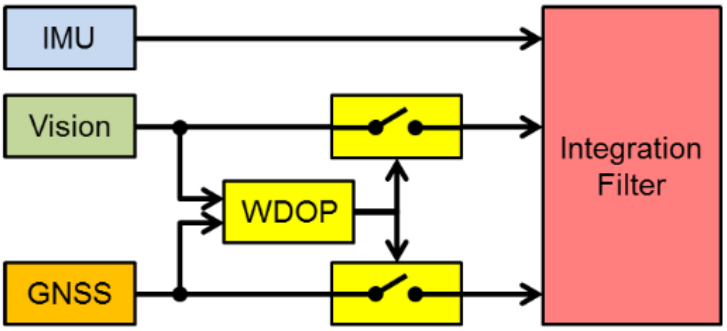


Figure 2.11 Block diagram of the selective integration system

Each component updates the overall state as soon as a measurement becomes available. It is worth mentioning that general purpose GNSS receivers have an update rate of around 1-10 Hz, whereas a simple IMU can provide data with a frequency of about 50 - 100 Hz. The feasibility of the proposed method was verified through simulations and an experimental testing. The results obtained by this study can be found on figures and .

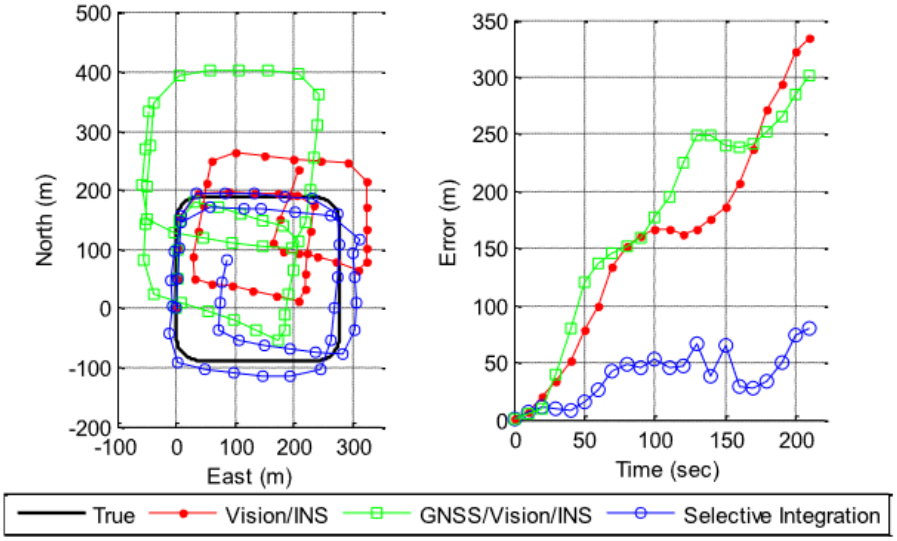


Figure 2.12 Results obtained after performing a simulation

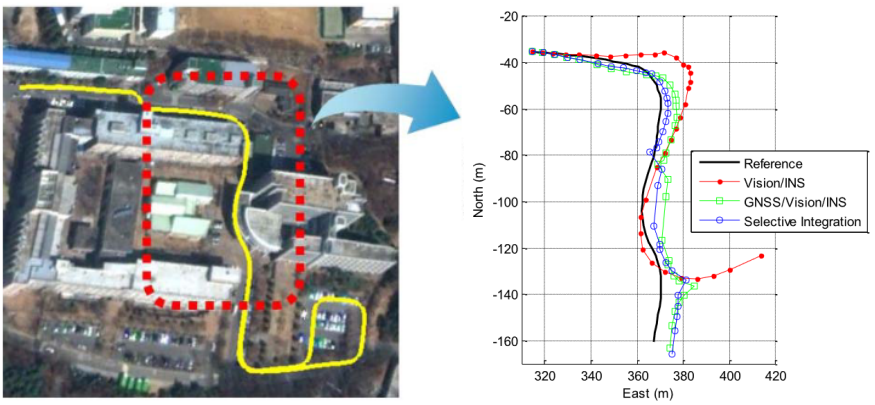


Figure 2.13 Results obtained after experimental test

Another study approaches cognitive autonomy on small drones [Enabling Cognitive Autonomy on Small Drones by Efficient On-Board Embedded Computing: An ORB-SLAM2 Case Study] . This case study focuses on the efficient utilization of the ORB-SLAM2 [4] simultaneous localization and mapping algorithm, on low power embedded systems. Study conducted on an ARM based Odroid-XU4 board indicates the ability of integration of low power embedded computers to on drones, as well as an optimal operating point. To enable cognitive autonomy, the researchers modeled and implemented ORB-SLAM2 as a Kahn Process Network (KPN) which exploits pipeline parallelism. Data-level parallelism is exploited. The pipeline of the algorithm is shown below on figure.

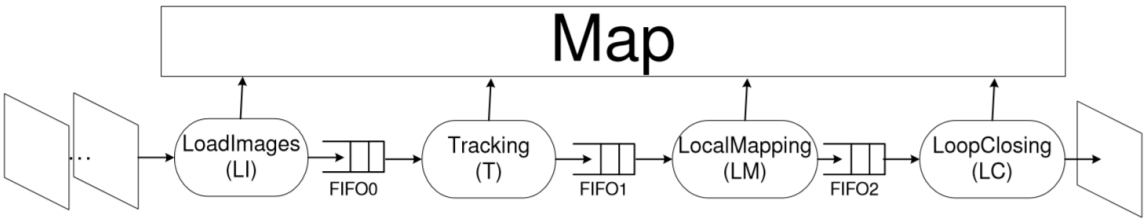


Figure 2.14 KPN model of ORB-SLAM2

According to the researchers, the system was able to perform in real-time at rates up to 18.21 fps. A performance vs power consumption trade-off (figure) presented the ability of the algorithm to execute with adequate performance with a limited power budget of 9 to 12 W on the on-board embedded module. It is worth mentioning that data obtained for this research correspond to 84 distinct system configurations.

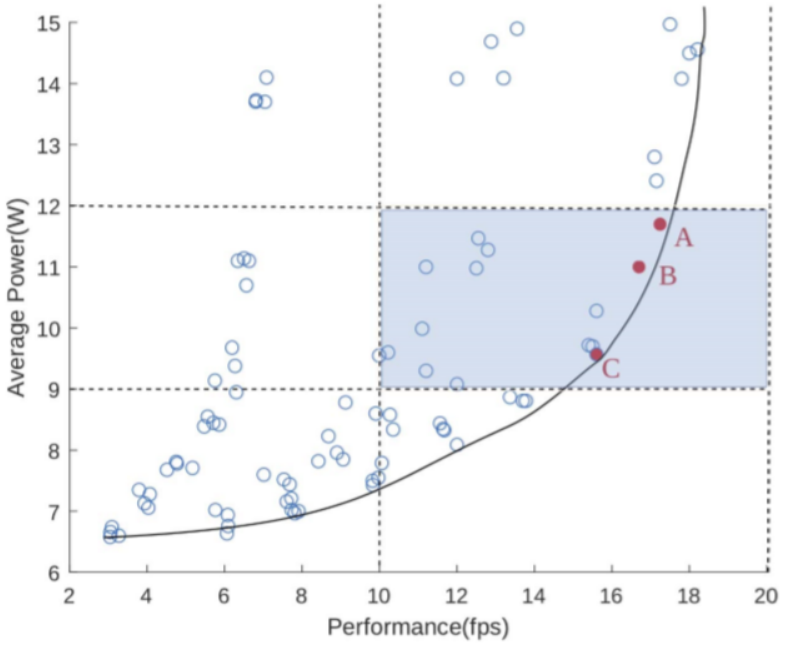


Figure 2.15 Performance-Power Consumption Trade-off

Chapter 3: System Architecture

3.1 Problem Statement

Navigation can be a demanding task. A navigation system needs to overcome many obstacles in order to operate adequately, especially in UAV applications. The first obstacle that needs to be tackled is localization, since robust navigation requires robust localization. Different environments require different localization techniques, hence operating a UAV in different environments requires interchangeable operation on localization submodules.

GNSS provides an easy to integrate localization solution. Despite GNSS's ability to provide global coverage and in some cases high precision positioning, it is susceptible to noise and can operate appropriately only outdoors, under specific conditions. Indoor localization cannot utilize GNSS since the satellite signal cannot pass through concrete. Another problem that GNSS receivers suffer from is when multipath receptions occur. Multipath describes the situation where received GNSS signals do not follow a direct path as they often get reflected. Moreover, UAVs usually become targets of GNSS spoofing attacks, where fake GNSS signals get generated and transmitted towards UAVs, leading to major errors on the three dimensional position fix.

Inertial measurement units (IMUs) supply vital data for a flight, since they can estimate the aircraft's attitude in real-time. In UAVs, IMUs are usually part of the attitude and heading reference system (AHRS). They provide data at high update rates and can withstand extreme levels of acceleration. However, positioning techniques which rely on IMU data tend to drift over time as acceleration errors accumulate. In a similar fashion, rotational estimations also drift.

In many applications, being able to provide real-time and precise geolocation of nearby objects can be vitally important. However, in order for a UAV system to be able to pinpoint such positions, being able to localize itself is required. Having the ability to perform geolocation tasks, a UAV can be used in a variety of different scenarios such as land survey, mapping as well as search and rescue missions.

This study tackles the challenge of designing and implementing a device which is capable of navigating a UAV in different environments by providing a multi-tier localization submodule which relies on data fusion among GNSS, inertial and visual-depth data. This submodule should be capable of providing continuous localization estimations, even when swapping operating environments. In addition, this module aims to increase the navigation and geolocation capabilities of UAVs. Thus a geolocation submodule will also undergo design and implementation. The goal of this submodule is to provide autonomous tracking and precise geolocation of nearby objects of interest.

3.2 Localization module

UAVs are vehicles which operate in a vast variety of environments. Each and every environment of operation has its own attributes. Some of them allow the use of GNSS, whereas some others don't. Some have strong magnetic fields and some others do not allow sunlight to enter. This is why UAVs need to adapt to each environment. Adapting in such environments begins by ensuring adequate localization in every possible scenario.

The proposed architecture of the localization submodule should be able to provide reliable positioning data for the UAV even when operating in GNSS denied environments. In addition, the submodule should allow stable execution of localization tasks, even when the environment changes during flight. This goal can be achieved with concurrent execution of localization tasks, performed on distinct data input streams.

Input data streams of the localization module

- **GNSS** data will be the primary source of positioning estimations. Either single GNSS receiver or differential GNSS receiver data will be used when applicable, as the fundamental source of positioning. Utilization of GNSS data comes with significant advantages. To begin with, global coverage is provided with the use of GNSS. In addition, GNSS measurements refer to absolute positioning data which is crucial for the proposed system since they do not suffer from over-time cumulative errors. Moreover they refer to a widely used standardized referencing system. However, GNSS signals are susceptible to noise and can only be used only outdoors under specific environmental conditions as stated earlier.

- **IMU / AHRS** data will be the primary source of attitude related data. IMUs can supply stable data regarding the current attitude as well as linear acceleration and angular velocity estimations in three dimensions. Attitude data provided by an IMU / AHRS refer to absolute measurements which are not susceptible to cumulative errors. Moreover, inertial data provided by an IMU can be vital when performing dead reckoning (1.3), if required. At last, the IMU needed for this study should be able to perform sufficiently in different temperature and humidity ranges given the UAV can climb and descend during the flight into the atmosphere.

- **Visual - depth** data are data acquired with an RGB-D camera (also known as visual - depth sensor), which contain depth information referenced with an optical image regarding the field of view of the sensor (see Figure 3.2.). The idea behind the choice of such sensor type is simple. When the UAV cannot utilize the GNSS in some environments, the system should be able to have spatial awareness by taking advantage of the optical features found in the flight area. By having optical and three-dimensional perception of the surroundings, the UAV should be capable of performing localization. The RGB-D sensor for this purpose must be able to perform the calculations needed for the three dimensional perception in its own asic. Such calculations can be performed on the on-board computer of the system. However, they are highly expensive in terms of power usage and processing requirements, hence the need for a complete visual-depth sensor. Furthermore the sensor should be capable of operating even in low light conditions.

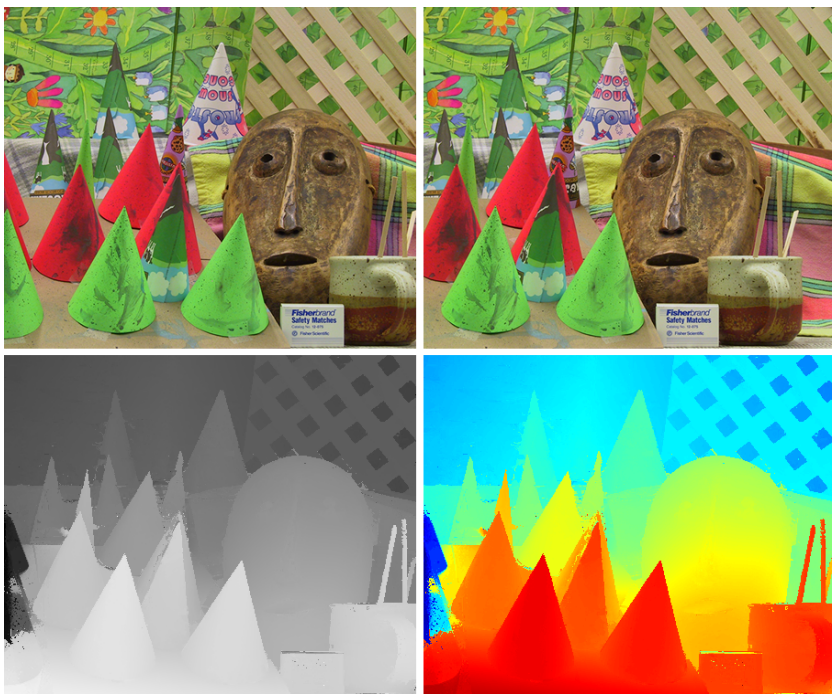


Figure 3.1: (Upper) Left and Right stereo pair image input, (Lower) Depth perception of the scene in grayscale (left) and remapped (right) using colormap "Jet". In the resulting image red indicates depth values closer to the sensor, whereas blue indicates more distant values.
[source: ethan-li-coding: https://github.com/ethan-li-coding/PatchMatchStereo]

The strategy is to enable the system to process multiple input streams concurrently, which provide different measurements and indications regarding position, as well as the movement of the UAV as shown in the table.

System	Position		Attitude	
	Local	Global	Local	Global
GNSS	✓	✓		
IMU / AHRS			✓	✓
Visual-Depth	✓		✓	

Table 3.1: Position and attitude data availability on the proposed systems

The available streams will be fused in different combinations to provide independent localization estimations. A process will be responsible to constantly monitor the input streams and assess the quality of the estimations. In a later stage, filtering and fusion techniques should be applied on these data in order to provide a single output state vector as shown in the block diagram below (Diagram).

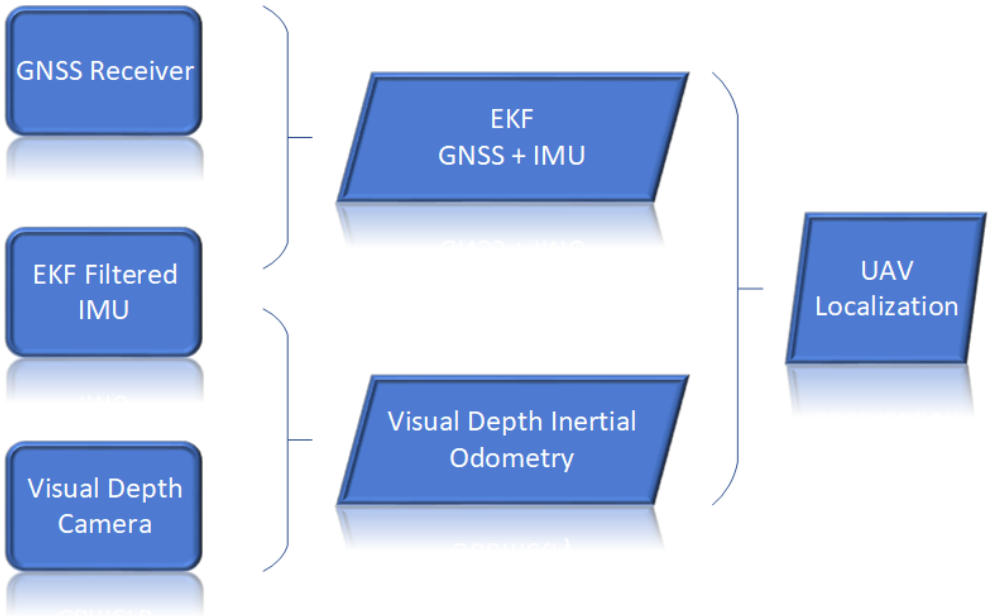


Diagram 3.1: Block Diagram of the Localization process

3.3 Geolocation module

It is well established that UAVs find good use in many applications. In some use cases, UAVs are utilized in order to estimate the position of an object or an individual. The process of estimating such positions at a geographic global level is called geolocation. Geolocation tasks differ in terms of accuracy. Some tasks require approximations within a few decameters, whereas others require centimeter level accuracy. Being able to

provide centimeter level accuracy, a geolocation enabled UAV, can be used in survey and mapping applications as well as search and rescue missions.

When high accuracy geolocation is required, a lot of considerations need to be taken into account while designing a geolocation module. Generally, the higher the accuracy needed, the more sophisticated the equipment should be. In this study, the geolocation module will be implemented as a three axis gimbal system, accompanied with an optical spectrum camera as well as a rangefinder. The key aspects of the operation of the geolocation module can be found below:

- **The module must be able to autonomously detect the object of interest.**
- **The module should continuously "follow" the geometric center of the object of interest**
- **The module must be able to determine the relative angles and distance between the object of interest and the UAV.**
- **The module must supply the user with geolocation estimations in real-time.**

For the geolocation module of this study, the proposed architecture is based on providing high accuracy readings regarding the relative angles and distance between the geometric center of the target object and the UAV's frame center. The measurements will be initially referenced with relation to the UAV frame. If the UAV is referenced with global attributes, then the estimations can be referenced in a global grid as well. In this study, the geometry will follow a tree hierarchy among the physical components. The localization submodule will provide the transform between a global origin and the UAV and the geolocation submodule will calculate the transform among the UAV and the objects of interest.

Geolocation module components

- **Gimbal System:** The gimbal system will be configured as a three axes motorized platform that will house all the required sensors. Each axis should be able to be configured individually. The platform should allow the sensors to be easily aligned with its main axes of rotation. The system should be as light as possible. Moreover, the sensors should be placed in a way they minimize the torque needed for quick corrections. Torque will be provided to the system by the gimbal motors. The motors should use their power to apply abrupt corrections. If the completed gimbal system is able to self balance in idle conditions, the motors will apply the least effort during the flight, resulting in longer lifespan for the motors, less power demand and longer flight time.

- **IMU:** Through an IMU fixed on the platform, roll and pitch angles (Chapter 2.3) will be measured and will be fed into the gimbal controller. The IMU sensor will be the primary source of error estimation in the closed control loop operating the gimbal system. The IMU needed for this purpose requires at least a 3-axis accelerometer and a 3-axis gyroscope. There is no need for a 3-axis magnetometer in this control loop, because the gimbal system should be capable of correcting its yaw axis relative to the frame of the aircraft. This angle can be supplied to the closed control loop through the use of an axial rotary encoder. For this study the required accuracy of the IMU angle for each axis must be 0.1°.

- **Rotary Encoders:** Each motorized axis will be continuously monitored with an absolute rotary encoder. Every rotary encoder will be able to measure with adequate accuracy each relative angle among the axes. The encoders must be able to communicate with both the gimbal controller as well as the processing unit in real-time. Moreover, allowing processing of rotary encoders' readings in the control loop, allows more precise motor control since the algorithm can always know the actual angle of each three phase motor. The required accuracy of each rotary encoder for this study is 0.05°.

- **Gimbal Controller:** The gimbal controller will be responsible for applying corrections to the platform. When the system is in active tracking mode, the controller will apply the corrections given by the processing unit. However, if the system is not in active tracking mode, the controller should stabilize the platform at a given setpoint angle relative to the horizon and relative to the aircraft's longitudinal axis. The controller will continuously execute a Proportional, Integral and Derivative (PID) feedback loop at a high update rate. After calculating the correction of each axis, it will drive the circuitry that controls each individual motor. This module should interface directly with the platform's IMU as well as the rotary encoders. Furthermore, the controller must be capable of external configuration in real-time to support the active tracking feature.

- **Camera:** The camera sensor will be fixed to the gimbal system and will provide an optical video stream to the processing unit. The camera should be aligned with the axes of the gimbal system. The output feed of this sensor will be processed in order to track the object of interest. The result of the processing will be the relative angles between the object and the camera's frame. In order to achieve high precision in the resulting data, the camera should be calibrated in order to eliminate distortion.

- **Rangefinder:** A rangefinder sensor will also be fixed to the gimbal's platform. The sensor must also be finely aligned with the platform as well as the camera, since it will be responsible for measuring the distance to the object of interest. For this study, the rangefinder should be capable of measuring distances of at least 30m with an accuracy of less than 1 cm. It is highly important that the rangefinder sensor provides a single point / one directional measurement. At last, the rangefinder unit must support external interfacing for real-time control and data processing.

- **Processing Unit:** The processing unit will be the core component of the geolocation submodule as it will be able execute the active tracking and the geolocation processes in real-time. It will be responsible to continuously interface with and control each individual component while processing the available data streams. For this study, this processing unit will also act as the processing unit of the localization submodule.

Initially the unit should acquire and undistort the camera feed. Next, a detection algorithm will detect the object of interest and feed the gimbal control loop with angular corrections. Then, the gimbal controller will receive the corrections and will apply the desired angular setpoints to continuously track the target object. The goal of the control loop will be to align the geometric center of the target object with the optical center of the camera. After ensuring that the object is sufficiently aligned, the processing unit will try to get a distance measurement through the rangefinder while simultaneously acquiring the angular measurements supplied by the rotary encoders. Data collected will then be processed through geometric calculations and filtering in order to produce an estimation of the required position. The block diagram of the proposed geolocation module is shown below (Diagram).

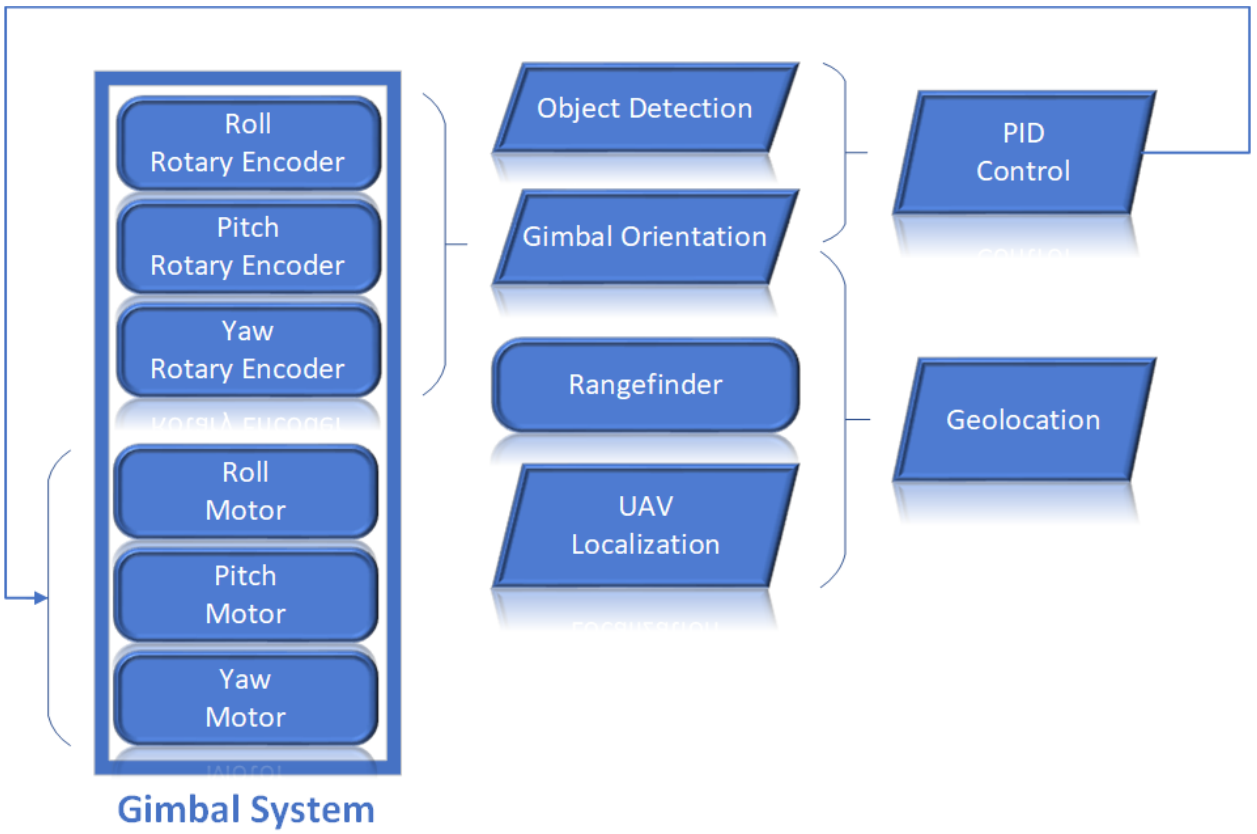


Diagram 3.2: Block Diagram of the Geolocation process

3.4 Human Machine Interface (HMI)

Human Machine Interface (HMI) is a user interface that connects a person to a machine. For the purposes of this study, the HMI is the medium between the UAV and its operator. It includes every component and visual interface that interacts with the operator and controls either the aircraft or even distinct modules. In addition, HMI is also responsible for feeding the operator with data regarding the overall status of both the aircraft and the flight.

Controlling a UAV can be challenging sometimes. When performing a manual flight, the operator should constantly apply input to the aircraft's remote control system while concurrently monitoring the current state of the aircraft. When the aircraft is not in the vicinity of the operator, a manual flight can only take place if the operator can have access to either a video stream or flight data of the aircraft. To ensure a safe flight, the operator must maintain situational awareness and should be constantly available to apply corrections.

Nowadays, the most common way of applying control to a commercial UAV is through the use of a handheld radio control (RC) system. The handheld transmitter (Figure) usually contains two sticks which provide the primary control of the aircraft as they govern the aircraft's desired translation and rotation. Auxiliary switches and potentiometers are also included in the handheld unit. In addition, modern RC systems support bi-directional communication between the handheld controller and the UAV unit. Such a feature allows telemetry data to be sent back to the operator through the already established channel. Other more industrial options provide a total ground station computer with attached controls (Figure). They are usually developed with a form factor of a heavy-duty suitcase hence they are completely portable.

Figure 3.2: (Left) FrSky Taranis X9D handheld radio control unit [source: FrSky], (Right) UAV portable ground station [source: Unmannedrc]

Such solutions allow portability and ease-of-use which are key aspects of this study. They can both provide sufficient control as well as indicate vital information about the aircraft and the flight. The option of the handheld unit provides higher portability among those two options whereas the ground station supports higher functionality and allows for greater situational awareness as more data and instruments can be monitored.

In aviation, there are some key instruments that the operator needs to monitor continuously throughout the flight [https://books.google.gr/books?hl=en&lr=&id=uxNUBAAAQBAJ&oi=fnd&pg=PP1&dq=aircraft+key+instruments&ots=xOHAZbGO-S&sig=A49ud6u4uuCOC3ehLet48pMNi-g&redir_esc=y#v=onepage&q=aircraft%20key%20instruments&f=false]. These instruments are:

- **The airspeed indicator:** An instrument which indicates the airspeed of the aircraft.
- **The attitude indicator (artificial horizon):** An instrument which indicates the orientation of the aircraft relative to Earth's horizon.
- **The altimeter:** An instrument which indicates altitude.
- **The turn coordinator:** An instrument which displays turning rate and roll information, as well as turning quality.
- **The heading indicator:** An instrument which shows the current heading of the aircraft.
- **The vertical speed indicator:** An instrument which shows the climb or descent rate of the aircraft.

Usually, in general aviation aircrafts, these instruments are fixed on a panel as a group. However, modern aircrafts provide the indications of these instruments as part of an electronic flight display commonly known as “glass cockpit”. Both options will provide the pilot with at least the aforementioned key data. A direct visual comparison is provided in the picture below (Fig).

Figure 3.3: Round instruments (Left) vs Glass cockpit (Right) [source: Mid-Continent Instruments and Avionics]

The panel option is user friendly and allows multiple individuals to immediately understand the instruments as they are universal across aircraft types. Moreover, gauges provide quick updates and mitigate the need for focusing on exact numbers. The glass cockpit option allows more information to be shown to the pilot while providing more precise readings. Moreover, the glass cockpit can alert the pilot for warnings and errors [https://www.mcico.com/resources/flight-instruments/glass-cockpit-vs-original-cockpit].

All the aforementioned designs have advantages and limitations. They are the result of many years of work, experimentation and continuous development, hence they provide a great source for design and architecture considerations. Creating the HMI architecture of this study is a challenging task as the interface should satisfy many requirements. Those requirements revolve around the manipulation of the aircraft and its components, as well as allowing the operator to have sufficient situational awareness. The HMI architecture of this study addresses the following concerns.

- The remote control unit must be portable and lightweight. One of the intended uses of the system is search and rescue missions. In such a use case, the system must be deployed as soon as possible, hence additional setup would increase deployment time.

- The remote control unit must support standard aircraft manipulation while allowing direct and real-time control of on-board systems, such as the geolocation module.

- Controlling the system must allow single operator usage.

- Controlling the on-board systems should require minimum input from the operator, as the system is intended for single operator usage.

- The operator must have a live view from the aircraft, to allow fights beyond visual line of sight (VLOS) between the operator and the UAV.

- The operator must be capable of monitoring the vital parameters of the aircraft and the flight. Having such knowledge available has a great effect in increasing situational awareness, resulting in safer flights.

The whole system will be controlled by a single operator. The operator will control the system through a radio control unit. On a digital display, the operator will be able to watch a video feed broadcasted by the aircraft. The video stream will be composed of two layers. The first layer will contain live view from the UAV, whereas the second layer will provide aircraft, flight and mission related data. Despite the fact that the system will only be controlled via a single operator, the video stream can be broadcasted to many viewers. This feature allows even more scenarios to be executed with this integrated system as it enables interoperability. The block diagram of the proposed architecture can be found below (Diagram).

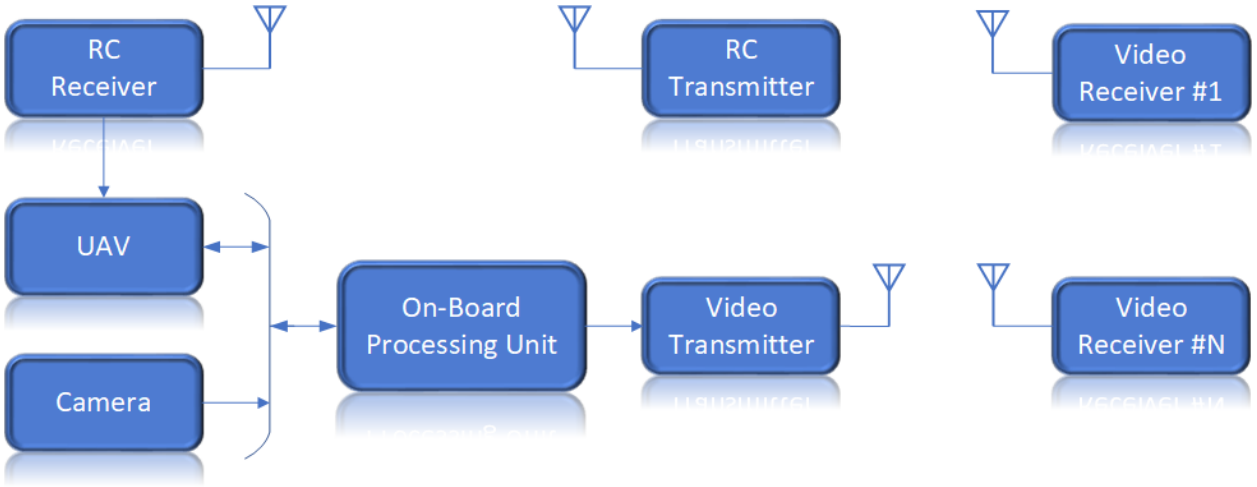


Diagram 3.3: Block diagram of HMI communication

Chapter 4: System design and implementation

4.1 Host aerial platform

In the previous chapter (Chapter 3), the proposed architecture of each module of the system was presented. Functional as well as non-functional requirements of the system were also stated. This chapter will cover the design and implementation phase, as it will present the transformation of the already stated requirements into a functional prototype. The prototype of the integrated navigation system of this study, will be attached to a custom UAV which will act as a host platform. Coupling the navigation system with the host UAV is also part of this study.

4.1.1 Selection process

Designing and implementing an integrated navigation system with geolocation ability is a demanding task. It requires knowledge of modern navigation systems, sensors, geometry, control theory, both software and hardware design, as well as basic aircraft control (see Chapter 2). The first step of designing such a system is to find a target host aerial platform.

Commercial UAVs usually fall into two categories. These categories are “Rotorcraft” and “Fixed Wing” UAVs (Figure). Apart from the shape, the main difference between those two classes is the component that supplies the required lift as well as the thrust for the flight. In rotorcrafts, lift and thrust are supplied by spinning rotors [https://www.faa.gov/regulations_policies/handbooks_manuals/aviation/helicopter_flying_handbook/media/hfh_ch02.pdf]. In fixed wing aircrafts, lift is primarily supplied by the wings and thrust is produced by the engines [https://www.nasa.gov/audience/foreducators/k-4/features/F_Four_Forces_of_Flight.html]. Sometimes, hybrids are created as custom solutions but they are not so popular commercially.

Figure 4.1: (Left) Multi-rotor UAV [source: HSE-UAV], (Right) Fixed-Wing aircraft [source: Quantum-Systems]

In order for the host platform to be chosen, the requirements of this study need to be taken into account. This study requires a host platform capable of carrying the navigation system without dramatically affecting its overall flight performance while also allowing some room for experimentation. The platform required must offer flight stability, precise control as well as the ability to maintain its position. In addition, fault tolerance is also another key factor towards safety when performing a flight.

Since evaluation of every possible commercial UAV platform is not a realistic scenario, possible candidates were studied from each UAV category. Initial preselection was performed in order to exclude platforms which are not intended for commercial use or have not proven air-worthiness. The searching process resulted in three configurations (Figure) which reflect the key features that each category provides.

A B C

Figure 4.2: Possible UAV configurations, (Left) Option A - Quadcopter [source: DJI Technology Inc], (Middle) Option B - Hexacopter [source: DJI Technology Inc], (Right) Option C - Fixed-wing light aircraft [source: Hobbyking]

The overall evaluation process of each candidate was performed in cooperation with UAV researchers and operators of the SenseLab research team, located in Technical University of Crete [http://www.senselab.tuc.gr]. During this process, each parameter was evaluated. Some attributes were evaluated based on actual experience, since appropriate metrics could not be found. The results of the evaluation process can be found in the table below (Table).

Attributes	Option A	Option B	Option C
Category	Rotorcraft	Rotorcraft	Fixed Wing
Type	Multicopter	Multicopter	Airplane
Number of Rotors	4	6	-
Stability	4/5	5/5	4/5
Precise Control	4/5	5/5	3/5
Position Hold	YES	YES	NO
Ease of use	5/5	5/5	3/5
Fault Tolerance	NO	YES	YES
Max Payload Weight	1.5 Kg	3 Kg	1 Kg
Expandability	4/5	5/5	3/5
Price (0 Low - 1 High)	0.7	1.0	0.8

Table 4.1: Evaluation results of the selected configurations

The final selection was made by taking into account the data provided by the evaluation process as well as the already established requirements of this study. By checking the resulting data, option C can be excluded as it does not satisfy the position hold functionality which is required. Despite its low price point compared to the other remaining option, option A can be excluded as it cannot provide sufficient fault tolerance. Moreover, Option B satisfies all the requirements while also providing greater expandability. Having some room for further experimentation can be really helpful sometimes when working with a prototype, as design changes may occur during development.

4.1.2 Overview

The target deployment platform selected for this study is the hexacopter. For this study, the hexacopter host platform will be a custom made prototype (Figure). It satisfies all the aforementioned requirements and has proven air-worthiness. The integrated system of this study will act as an auxiliary aircraft component, hence it will be designed as an external device which can be attached easily to the host aircraft. Developing this system as a device for the selected hexacopter can be very helpful in terms of industrial design as this aircraft represents the average hexacopter.

Figure 4.3: The selected hexacopter UAV which will act as the host platform, designed and developed at SenseLab, Technical University of Crete.

This hexacopter features a carbon fiber frame, rendering the aircraft sturdy and lightweight. It is propelled by six carbon fiber propellers, mounted on high power brushless DC (BLDC) motors. The whole aircraft is powered by two Lithium-Polymer (LiPo) batteries. The core of the system is the flight controller unit (FCU) which is responsible for converting the desired action of the aircraft into motor thrust control signals. Such signals are processed by the electronic speed controllers (ESCs) which ultimately drive the motors. The position of the aircraft can be determined via the GNSS unit which is installed at the top of the aircraft. The module which houses the externally mounted GNSS receiver also features an electronic compass. Communication between the aircraft and the operator can be achieved through the use of a radio control (RC) link. Moreover, communicating with ground stations and other systems is also enabled via data-links. The complete top-level design of this aircraft can be found below (Diagram).

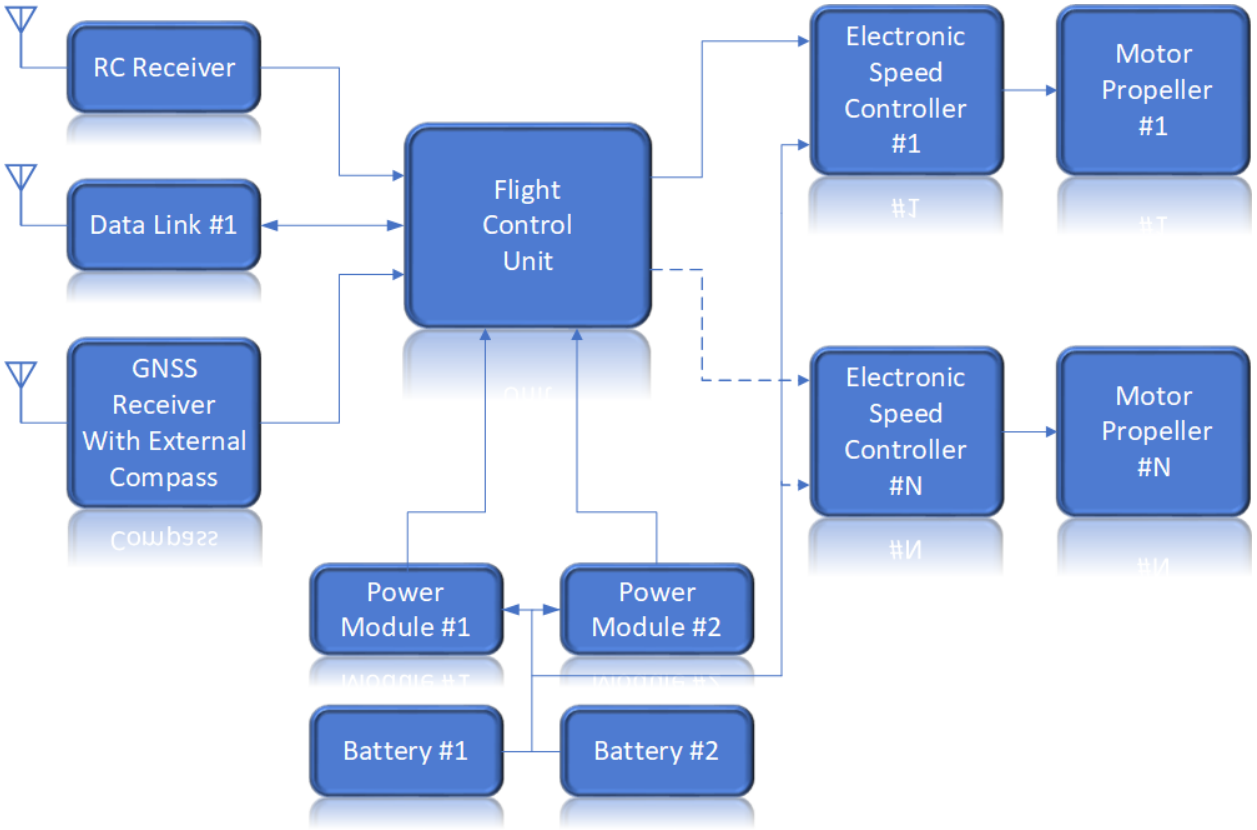


Diagram 4.1: Block diagram of the selected host hexacopter.

4.1.3 Hardware

Having a top-level understanding of the host system as shown in the previous section (section) is necessary for the design phase. However, developing an integrated navigation system requires detailed knowledge on the aircraft's components and interfaces.

Aircraft Frame

The frame of the aircraft is a Tarot 960 carbon fiber frame (Figure). Carbon fiber is a material composed of a long chain of carbon atoms. The fibers are extremely strong and light. By bonding fibers together, composite components are created. Due to the nature of the fibers, the strength to weight ratio of a carbon fiber component is much higher than steel [https://www.materialsciencejournal.org/vol14no1/carbon-fibres-production-properties-and-potential-use/]. The frame features six arms with CNC processed aluminum motor mounts as well as two legs which provide sufficient support for the whole unit. The carbon fiber plates which house the mechanical linkages of the frame offer many mounting positions for both mandatory and auxiliary equipment.

Figure 4.4: Tarot 960 carbon frame [source: RC Copter]

Parameter	Value
Weight	1050 g
Tube diameter	25 mm
Rack diameter	1000 mm
Centre cover size	210 × 210 × 2.0 mm
Motor mounting pitch	16 mm/19 mm/25 mm/27 mm
Wheelbase	960 mm
Max gross weight	15 Kg

Table 4.2: Tarot 960 frame specifications

Propulsion

The aircraft features a propulsion combination of six 17", 5.5" (Length, Pitch) carbon fiber propellers coupled with six T-Motor MT3520 400KV three-phase brushless DC motors (Figure). The propellers are split into two groups. There are three clockwise (CW) as they are three counter-clockwise (CCW) rotating propellers (Figure). This design eliminates gyroscopic effects when there is equal thrust distribution. According to the specifications of the motor (Table) [https://www.himodel.com/electric/T-Motor_MT3520_400KV_Outrunner_Brushless_Motor_for_Multi-copter_4-8S.html] , this combination has a max theoretical total thrust of 21600g when the motor is powered at 22.2 Volts.

Figure 4.5: (Left) T-Motor MT3520 with a quick release propeller mount. (Right) A pair of CW and CCW carbon fiber propellers.

Current (A)	Power (W)	Thrust (g)	RPM	Efficiency (g/W)
1	22,2	340	2000	15,315
5	111	1170	3600	10,541
10	222	1830	4450	8,243
16	355,2	2440	5100	6,869
20	444	2780	5410	6,261
26	577,2	3220	5790	5,579
32	710,4	3600	6060	5,068

Table 4.3: Performance chart of a T-Motor MT3520 400KV motor coupled with a 17" x 5,5" carbon fiber propeller while is powered at 22.2 volts

Controlling a three-phase brushless dc motor is a demanding task, as it involves high amounts of power and continuous monitoring of the electro-magnetic fields induced inside the motor. The units responsible for the precise control of the motors are the electronic speed controllers (ESCs) of the aircraft. These units are controlled by the flight control unit (FCU) and drive the motors by continuously adjusting the switching interval of each motor phase. As the motors are synchronous, the units need to meticulously measure the counter-electromotive force among the phases of the motor in order to estimate the orientation of the rotor, and adjust the switching timing. This aircraft features six T-Motor Air ESCs which support continuous 40A current handling at 22.2 Volts.

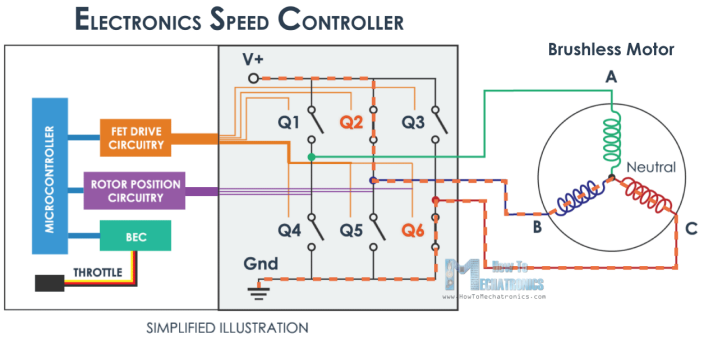


Figure 4.6: (Left) T-Motor Air 40A ESC unit. (Right) Block diagram of an ESC [source: https://howtomechatronics.com/how-it-works/how-brushless-motor-and-esc-work/]

Energy storage and flight performance

The hexacopter requires high amounts of energy for its operation as data of the previous section suggest. The selected energy storage modules for this UAV are two Gens-Tattu 12000mAh, 6-cell Lithium Polymer (LiPo) batteries (Figure). These modules pack a total of 532.8 Wh and can safely handle a peak current of 360 amperes.

Parameter	Value
Cell capacity	12000mAh
Configuration	6S1P / 22.2V / 6 cells
Discharge Rate	15C
Net Weight	1470g (±20g)
Dimensions	184mm x 71mm x 61mm

Table 4.4: Tattu 12000mAh LiPo Battery specifications

GNSS Module

The host UAV supports both manual and autonomous flights. One of the key elements that play a major role in autonomous missions is the GNSS module. The module installed in this aircraft is a HEX here2 system. This system features a Ublox M8N GPS & GLONASS enabled receiver along with an IMU module. The global position accuracy supplied by this system is 2,5 m, which is a satisfactory figure for an average flight. This module is externally mounted at the top of the aircraft's frame, making it less susceptible to electromagnetic interference produced by the motors. This allows for more accurate heading readings on the integrated IMU.

Parameter	Value
GNSS receiver	Ublox-M8N
IMU	9DOF + Barometer
Position Accuracy	2,5 m
Protocols	NMEA, UBX
Navigation Rate	up to 10Hz

Table 4.5: Here2 GNSS module specifications

Flight Control Unit (FCU)

The brain behind the operation of the aircraft is its flight controller unit (FCU). This module is responsible for the complete control of the UAV. It receives control input from the operator via either the radio control (RC) unit

or a data link, acquires measurements by interfacing with the sensors of the aircraft, and calculates the thrust setting for each motor to continuously satisfy the desired flight parameters. The flight controller of this hexacopter is a HEX/ProfiCNC Pixhawk 2 Back Cube (Figure).

This is a sophisticated module as it supports many UAV configurations and dynamic flight profiles. Its primary processor is a 32-bit ARM Cortex M4 core with floating-point unit (FPU). The board is also equipped with a secondary 32-bit failsafe co-processor. Given the redundancy required for a safe flight, this module features three independent sets of temperature-controlled IMUs, two of which are vibration-isolated mechanically and two independent barometers [https://ardupilot.org/copter/docs/common-thecube-overview.html].

Figure 4.7: HEX/ProfiCNC Pixhawk 2 Black Cube

This flight control unit supports a big range of interfacing abilities. Communication between the FCU and the sensors of the aircraft can be ensured by the available interfaces. At a hardware level, this board features UART, I2C, SPI and CAN bus. Motor control signals are supplied to the ECSs via PWM outputs of the FCU. Interfacing with the flight controller is critical for the purposes of this study, as the navigation system requires input from the sensors of the aircraft. In addition, the system might need to take over control of the aircraft which can be achieved by issuing control commands to the flight control unit. The interface which enables the communication between the navigation system and the FCU is UART. Pixhawk 2 Cube is a highly compatible module for this study, as it features five Serial ports, two of which support flow control.

4.1.4 Firmware and functionality

The host UAV of this study is hexacopter aircraft with proven airworthiness. Via the installed flight control unit (FCU), the aircraft is well suited to perform both manual flights and autonomous missions. Despite those well designed sophisticated components, the heart of the FCU is the firmware. The firmware is responsible for processing data provided by sensors and external devices such as radio control units and ground stations, while managing the aircraft's behavior, components and actuators. Moreover, a very crucial element of the firmware is its ability to maintain the aircraft within safe operation limits.

Currently, the firmware operating the FCU is an ArduCopter 3.6.12 Hexa X [https://firmware.ardupilot.org/Copter/stable-3.6.12] version of the Ardupilot open-source project [https://ardupilot.org/]. This firmware features various flight modes which ensure stable operation. The supported flight modes allow for dynamic levels of maneuvering assistance for manual flights as well as completely autonomous missions. The flight modes essentially affect the operation of the control algorithm. A Proportional Integral and Derivative (PID) feedback loop shapes the core of the control algorithm. This process takes into account the current state of the aircraft as well as control input, and adjusts the behaviour of the aircraft accordingly.

Utilizing the ArduCopter firmware comes with some important advantages. As mentioned in the previous section (Section), the FCU facilitates independent Inertial Measurement Units (IMUs). This firmware option supports data filtering and fusion through the use of Extended Kalman Filters (EKFs) (see Chapter..). At any given moment, many instances of an EKF might execute, however only the one that reports the most consistent

data is selected. In terms of redundancy, this is a great design as it can easily detect anomalies during the flight and provide reliable measurements even if an IMU fails completely.

Another critical advantage that comes with using the ArduCopter firmware is the integration of MAVLink [https://mavlink.io/en/] protocol. The name of the protocol comes from the phrase “Micro Aerial Vehicle” and provides a very efficient way of communication with UAVs. This protocol supports data exchange between UAVs and ground stations as well as direct communication among UAVs. Moreover, various components can communicate with the FCU of the UAV via a data link using MAVLink. The protocol follows a publish - subscribe and point-to-point architecture for publishing data streams, while configuration messages are point-to-point with retransmission.

ArduCopter has 23 built-in flight modes to support different levels of flight assistance. For this study, the selected flight modes are “Stabilize”, “Loiter” and “Guided”. These flight modes were selected in this exact sequence with the idea of gradually increasing the automation level of the flight. More detailed descriptions of each flight mode can be found below.

Stabilize

“Stabilize” [https://ardupilot.org/copter/docs/stabilize-mode.html#stabilize-mode] is a manual flight mode which assists the operator to maintain the aircraft level, related to the horizon. In this flight mode, the operator can adjust both the attitude of the aircraft and the overall thrust of the aircraft. Inside the control loop, the flight control unit will read the desired attitude requested by the operator as well as the current attitude and thrust setting, and will adjust the thrust percentage provided by each motor/propeller to match the target attitude and thrust. Despite its simplicity, this flight mode is considered challenging especially for new operators, as it requires continuous control even to maintain position both horizontally and vertically. This flight mode can be utilized as a failsafe, in case more advanced modes fail.

Loiter

“Loiter” [https://ardupilot.org/copter/docs/loiter-mode.html] is another manual flight mode which provides position holding assistance. In this flight mode, the operator can directly adjust the desired velocity of the aircraft both horizontally and vertically. It requires minimum input from the operator as the attitude of the UAV is managed by the FCU. The control loop of this flight mode needs to consider the requested velocity vector, the current position as well as the current inertial state and adjust its behavior accordingly. Loiter provides ease of use as idle operator input results in maintaining current position.

Guided

“Guided” [https://ardupilot.org/copter/docs/ac2_guidedmode.html] is a flight mode that allows controlling the UAV by feeding the FCU with target locations via a data link. This flight mode offers just the right level of automation for the requirements of this study. The navigation system is able to take over control of the aircraft when this flight mode is selected. In this case, the FCU has to manage all the flight parameters. Despite the ease of use this system provides, it requires many more mechanisms to function correctly compared to the flight modes mentioned previously, increasing the risk of a fault.

4.2 On-board processing unit

In order to support the functionality that this study requires, an on-board computer is facilitated inside the navigation module. The on-board processing unit is the core component of the system. It enables the functionality of both the localization submodule as well as the geolocation submodule. Moreover, navigation

utilities are also included in this module. The on-board unit communicates in real-time with the flight control unit (FCU) of the aircraft, exchanging vital data for both localization and navigation functionality.

4.2.1 Hardware

The selected on-board unit of this study is a Raspberry Pi 4 board (Figure). According to the datasheet of the module [https://datasheets.raspberrypi.org/rpi4/raspberry-pi-4-datasheet.pdf], his board operates on a Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC clocked initially at 2.2 GHz. This model features 8 GB of RAM, two USB 3.0 ports, two USB 2.0 ports as well as sufficient hardware level interfacing. More specifically, it facilitates UART interfaces, I2C and SPI buses, as well as GPIO pins.

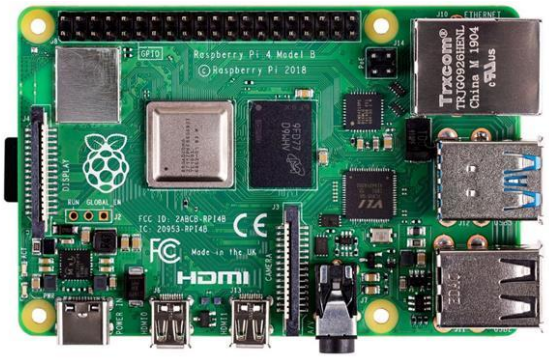


Figure 4.8: Raspberry Pi 4, Model B

The selection of the board was made after taking into account many parameters and constraints. The software of the navigation system is designed as a package for the Robotic Operating System (ROS) (Chapter 2) environment. ROS (ROS 1) can execute on top of many modern operating systems, however the greatest compatibility comes with Linux distributions. As a sequence, the board needed to support Linux distributions. Regarding interfacing options requirements during the design phase, the system required at least one USB 3.0 port to communicate with the majority of commercially available RGB-D sensors, an SPI or I2C bus for interfacing with the rotary encoders of the geolocation module, and at least three UART ports. One UART port was needed to enable the communication between the system and the FCU, one for interfacing with the gimbal controller and another one to control the laser rangefinder of the geolocation module. Moreover, the board also needed to have both adequate processing power and be lightweight with a small form factor.

This board offers satisfactory performance while having a small form factor. The CPU of this module was initially clocked at 1.5 GHz, however during the implementation process, the clock speed was raised to 2.2 GHz. Further discussion regarding the overall performance will be provided in the following chapter (Chapter 5). Raspberry Pi 4 satisfies the aforementioned requirements while allowing even more interfacing options without sacrificing the small form factor.

Parameter	Value
CPU	BCM2711B0 Quad-Core A72 (ARMv8) 64-bit @1,5 GHZ
GPU	Broadcom VideoCore VI
RAM	8GB LPDDR4-3200 SDRAM
Networking	2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Gigabit Ethernet
GPIO interfaces	28x user GPIO supporting various interface options: Up to 6x UART, Up to 6x I2C, Up to 5x SPI, 1x SDIO interface, 1x DPI (Parallel RGB Display), 1x PCM, Up to 2x PWM channels, Up to 3x GPCLK outputs
Weight	46 g
Dimensions	88 mm × 58 mm × 19.5 mm

Table 4.6: Raspberry Pi 4 Model B - 8GB specifications
[https://datasheets.raspberrypi.org/rpi4/raspberry-pi-4-datasheet.pdf]

4.2.2 Operating System and Software

Selecting an operating system for the on-board processing unit was primarily based on ROS 1 compatibility, as the system is built on top of this environment. Another important factor is the architecture of the CPU. The featured Cortex-A72 (ARM v8) is a 64-bit processor and the installed RAM is 8GB, hence the operating system should support arm64 architectures.

The selected operating system is a custom lightweight version of Ubuntu 18 for arm64 architectures. This operating system was selected as it satisfies all the requirements while being lightweight. It provides sufficient compatibility with ROS 1 modules and the additional software components used in this study. Moreover, it allows user defined configuration of key system parameters as is clock frequency and active interfaces when booting the kernel.

On top of the operating system of the navigation module lie many software components that play crucial roles in the functionality of the system. Implementing a navigation system with localization and geolocation abilities, requires the creation of various processes, each one responsible for a different fragment of the initial problem. The combined operation of all those processes, shapes the system. This behaviour can be achieved through the use of some very important pieces of software.

ROS (Robotic Operating System)

ROS (Robotic Operating System) [https://www.ros.org/], is a software component that enables design and implementation of robotic applications. It is not an actual operating system, yet it provides functionality that operating systems support. This functionality ranges from hardware abstraction and low-level device control, to inter-process message passing and package management.

ROS is built to provide a modular approach for robotic applications. Developing individual functional modules can be very helpful especially when designing high complexity applications. This approach allows for straightforward development, as each module has a very specific task. An application is constructed after combining the developed modules in a top level design. Being able to compose applications through the use of modular design, can further aid future expansion of the application. New functionality can be integrated effortlessly in the design in a modular fashion.

Figure 4.9: PR2 Robot running ROS, designed by Willow Garage, source: IEEE

Development of ROS began in 2007 by a robotics company named Willow Garage. ROS proved to be a sustainable ecosystem right after its inception. Today, the ROS community consists of tens of thousands of users worldwide as it is the framework of choice in various applications ranging from hobby projects to large industrial automation systems.

Various applications are enabled to be structured inside a ROS environment through the use of primary ROS tools. ROS modular functional components are called “nodes”. Nodes communicate with each other via “topics”. A node can publish data in the form of topics by the usage of “publishers” and other nodes can listen to such topics through “subscribers”. Moreover ROS “services” and “actions” enable functionality execution on demand.

Nodes

Nodes are individual functional blocks, developed to perform a specific task. ROS allows nodes to be written either in C++ or Python. When designing ROS environments, the functionality of the top level application should be divided into individual nodes. Each node can both publish and receive data while performing processes. For example, imagine an application where a UAV performs autonomous following of a car. A node should be responsible for detecting and tracking the car whereas another node should correct the position and rotation of the UAV relative to the car, by taking into account the data supplied by the previous node.

Topics

In ROS, topics form the data channels among nodes. Topics can hold custom messages which need to be exchanged within the ROS environment, in order for an operation to be performed. Custom messages can embed a various range of data such as integers and floating point numbers to images and point cloud data. Topics transfer data messages broadcasted by some node’s “publisher” to other nodes where they are obtained by “subscribers”. Moreover, they are usually synchronous, as they are broadcasted in relatively constant intervals, yet they can be asynchronous as well.

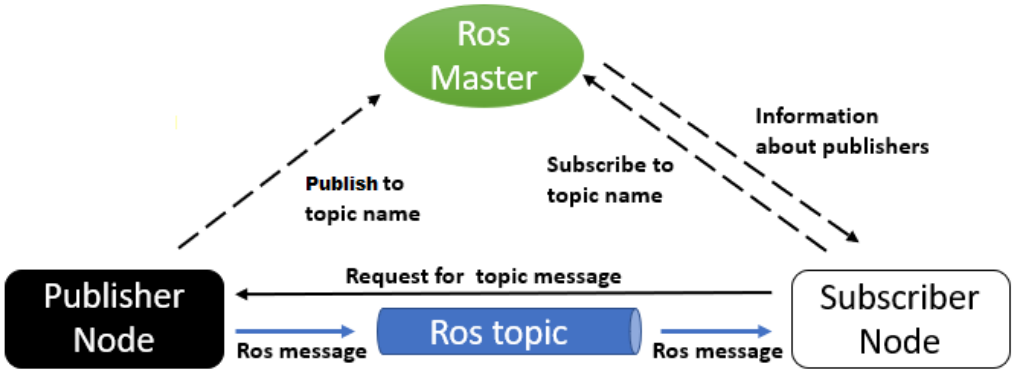


Figure 4.10: ROS Communication, source: trojrobert, Github

Services and Actions

Frequently, ROS nodes provide on demand functionality. This is achieved with the utilization of services and actions, available at runtime. When requested by the user or a ROS node, a service can be executed. The service can take as input data supplied through the request and can provide output data through a response after its execution has completed. Transferring data to and from a service is not mandatory. Services are usually brief in execution and behave similar to functions in a program, like performing a mathematical operation or applying a filter to an image. However, when the demanded functionality is a time consuming process, actions are used. Actions work similar to services. They apply in cases where the execution time of their assigned task is unknown, like moving a robot to a specific location.

Today, the ROS framework is undergoing improvements as the ROS-2 middleware is currently developing. The next generation of this system aims to take greater advantage of later versions of both C++ and Python programming languages in its API, while also improving overall performance. In ROS 2, it will even be possible to implement real-time nodes, when using a proper Real-time operating system (RTOS).

OpenCV

OpenCV [https://opencv.org] (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCVwas built with the aim to provide a common infrastructure for applications towards computer vision. The library is equipped with a vast variety of image processing tools and utilities as more than 2500 optimized algorithms are featured. This machine vision toolbox includes both classic as well as machine learning algorithms. Through the use of OpenCV, an application is well suited to perform demanding image processing tasks. Such tasks are object detection and identification, feature extraction, camera tracking as well as three-dimensional perception.

OpenCV was originally developed and initially released by Intel Corporation, in June 2000, and was later supported by Willow Garage (The forerunner of ROS). Throughout the years OpenCV became more efficient as more high performance features were added. Acceleration with the use of GPU played a highly important role in the performance improvement of the library.

Today OpenCV is included as an integral part of many modern image processing applications. It provides interfaces for C/C++, Python, Java and MATLAB. Moreover, it is supported by the majority of the operating systems of both stationary and mobile platforms.

ORB-SLAM2

ORB-SLAM2 [4] is a SLAM (see Chapter) algorithm implementation which allows monocular, stereo and RGB-D data to be utilized as input. Monocular data can only provide two-dimensional perception in scale, whereas stereo and RGB-D data can directly enable three-dimensional spatial awareness.

The principle on which this V-SLAM operates is the selection, identification and interpretation of distinct visual features as three-dimensional space registered points. The visual features are both extracted and matched among the supplied image streams, through the use of ORB [8] algorithm. As a system moves through unknown space, it supplies the algorithm with visual data which results in expanding the margins of the previously known map, while continuously estimating the trajectory traveled. Henche, each time new data arrive, the algorithm will output the latest estimated position and orientation of the device relative to a defined coordinate system, commonly known as “pose”.

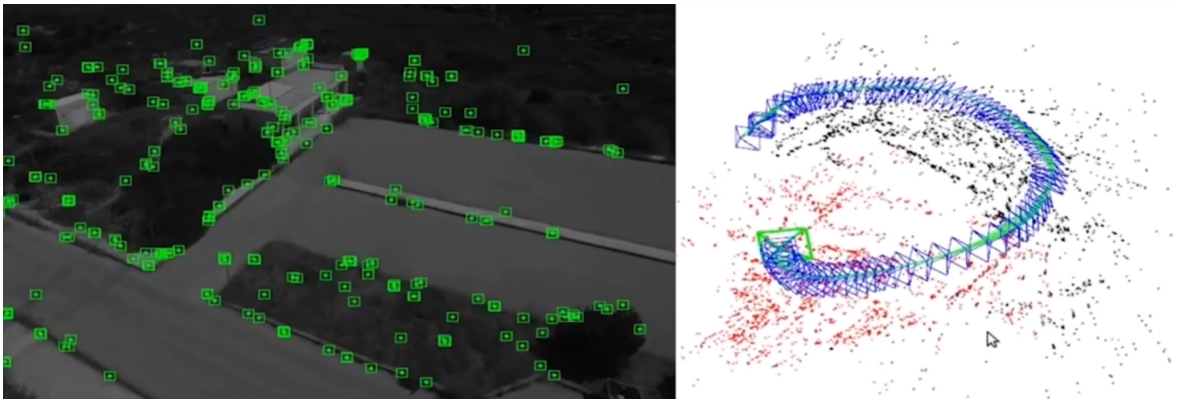


Figure 4.11: ORB-SLAM2 execution on monocular data onboard a UAV source: SenseLab, Technical University of Crete

ORB-SLAM2 pipeline begins by acquiring visual and depth data. Stereo image or RGB-D streams are fed into the preprocessing routine (Figure) where feature extraction via the ORB algorithm takes place. The features that are extracted by this process are called keypoints. The algorithm natively operates on Stereo and monocular keypoints, hence, in case the input is an RGB-D instead of stereo, features are extracted from the RGB image and each depth registered feature will transform its depth value into a virtual right coordinate. This process results in two sets of keypoints.

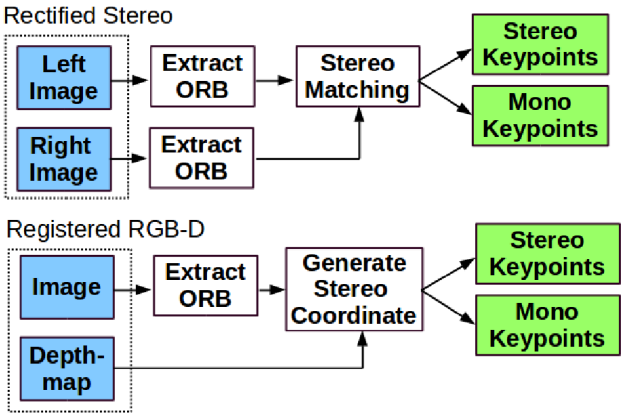


Figure 4.12: ORB-SLAM2 Pre-processing

Monocular keypoints correspond to all those ORB features for which a stereo match could either not be found or have invalid depth data. These points do not provide scale information as stereo keypoints allow, yet they assist in estimating rotation and translation.

When the operation of the algorithm is initialized, a keyframe is created with the first available frame which sets the pose of the tracking to the origin. In addition, an initial map is created from all the available stereo keypoints. While the camera moves through space, new keyframes are created and inserted into a bundle which contains keyframes in a spanning tree topology.

ORB-SLAM2 performs motion-only bundle adjustment to optimize the camera pose in the “Tracking” thread. As new keyframes are inserted into the local mapping bundle, local bundle adjustment takes place in order to optimize a local window of keyframes and points in the “Local Mapping” thread. Concurrently, another thread called “Loop Closing” performs loop detection in the current local mapping. When a loop is detected, a full bundle adjustment is performed to provide an overall correction in the mapping process and update the map. In case the loop is detected during the optimization process, the process is aborted and the loop closing takes place. When the full bundle adjustment completes, the subset of keyframes and points need to be merged and updated with the non-updated keyframes and points that were inserted while the optimization was running. This process occurs by propagating the correction previously calculated during the pose optimization process, of updated keyframes to non-updated keyframes through the spanning tree.

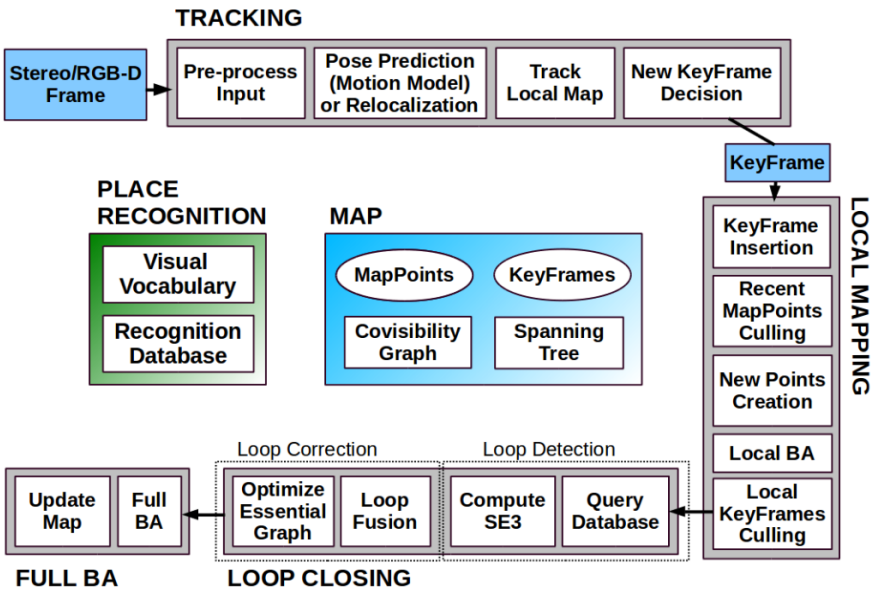


Figure 4.13: ORB-SLAM2 threads

4.3 Localization Module

It is already well established that in order to support navigation features in multiple environments, the system requires spatial awareness throughout the flight. This requirement is satisfied in this study through the implemented localization module of the system. This system is responsible to process the streams that provide both inertial and spatial awareness and provide a single output vector containing the estimated localization data.

The goal of this module is achieved via the use of both hardware and software components, designed and implemented for the purposes of this study. The following sections describe both the design and the implementation of those vital components.

4.3.1 Hardware

The architecture of this module mandates that this system will be used as an auxiliary module on UAVs. Modern UAVs, as the hexacopter utilized in this study, facilitate many sensors in order to support both inertial and spatial awareness. Most of these sensors provide high quality measurements while ensuring stable operation of the system. Hence, this module should be capable of utilizing these sensors when possible, as including additional sensors which support similar functionality is neither cost effective nor efficient in many cases.

Concurrent usage of distinct inertial and spatial awareness components is the key idea behind the operation of the localization module. The architecture of this module requires the integration of Global Navigation Satellite Systems (GNSS), inertial as well as visual-depth data.

Global Navigation Satellite Systems (GNSS)

The primary source of the localization module is a GNSS receiver. During the initial phase of the implementation, the system was capable of acquiring positioning data from the UAV. The stream of data was published by the flight control unit (FCU) of the aircraft after receiving it from the GNSS module. The installed GNSS module is a Ublox M8N receiver. This module features a position estimation rate of up to 10 Hz, yet the 2.5m of position accuracy is not a satisfactory figure for the purposes of this study. It is worth mentioning that the system is able to operate with this module, however this study tackles higher levels of precision. In order to achieve this goal, another GNSS module needed to be selected and tested. That module was Ublox F9P [https://www.u-blox.com/en/product/zed-f9p-module].

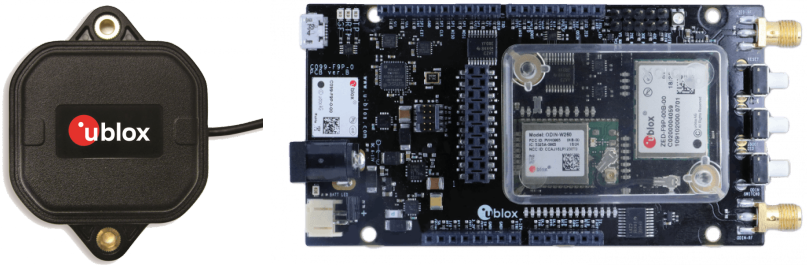


Figure 4.14: (Left) Ublox ANN-MB dual-band GNSS antenna (Right) Ublox C099-F9P application board

Ublox F9P is an emerging professional and cost effective GNSS solution. As indicated by the datasheet of this product [https://www.u-blox.com/sites/default/files/ZED-F9P-02B_DataSheet_UBX-21023276.pdf], this global positioning solution features centimeter-level positioning through the use of Multi-band Real-Time Kinematicks (RTK). This module supports concurrent reception of GPS, GLONASS, Galileo and BeiDou while offering position estimation rates of up to 25 Hz, In terms of performance (Table), this module offers more than satisfactory precision by taking advantage of RTK processing while providing sufficient update rate.

	GPS GLO GAL BDS	GPS GLO GAL	GPS GAL	GPS GLO	GPS BDS	GPS
RTK Nav rate	8 Hz	10 Hz	15 Hz	15 Hz	15 Hz	20 Hz
RAW Nav rate	20 Hz	20 Hz	25 Hz	25 Hz	25 Hz	25 Hz
Convergence Time	< 10s	< 10s	< 10s	< 10s	< 10s	< 30s
Horizontal position accuracy (RTK)	0.01 m + 1 ppm CEP	0.01 m + 1 ppm CEP	0.01 m + 1 ppm CEP	0.01 m + 1 ppm CEP	0.01 m + 1 ppm CEP	0.01 m + 1 ppm CEP
Vertical position accuracy (RTK)	0.01 m + 1 ppm R50	0.01 m + 1 ppm R50	0.01 m + 1 ppm R50	0.01 m + 1 ppm R50	0.01 m + 1 ppm R50	0.01 m + 1 ppm R50

Table 4.7: Ublox F9P performance

After experimenting with this system, the parameters provided by the manufacturer were validated. The system was able to perform as described. As a design note, the Ublox ANN-MB dual-band antenna should be mounted on top of a goundplane disk in order to mitigate unwanted multipath errors due to signal interference. Multipath errors can occur frequently, especially when operating a UAV close to the ground, near buildings as well as near trees as their leaves can reflect the signals coming from the satellites. Moreover, the ground plane should be as light as possible and should not be composed of ferromagnetic materials as it would have a negative impact on the magnetometers of the UAV. With this design note, the antenna was mounted on top of the UAV with a custom 3D printed base, and a 0.5 mm thick aluminum disk which acts as a ground plane by significantly attenuating signals coming from unwanted directions (Figure).

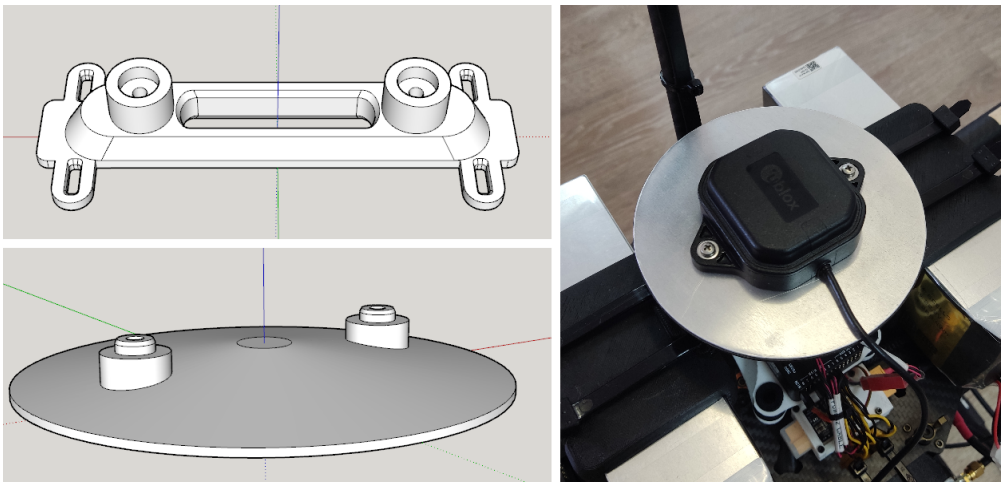


Figure 4.15: Mounting the GNSS antenna

Inertial Measuring Unit (IMU)

Considering the specifications and capabilities of the sensors featured in the UAV (see Chapter 4.1.3), the inertial measurements provided by the Flight Control Unit (FCU) of the aircraft are high quality data. It is a fact that these readings are provided by fusing the output of temperature-controlled IMUs through hardware implemented Extended Kalman Filters (EKFs).

The primary IMUs installed in the Pixhawk 2 Cube FCU are MPU9250, ICM20948 and ICM20648. The EKFs featured in the ArduCopter firmware operate on the measurements provided by these MEMS cores. Cube FCU provides near optimal operating conditions as it keeps two IMU cores isolated under constant temperature in a temperature-controlled environment. In addition, they are vibration-isolated mechanically. This architecture reflects the level of sophistication required, when designing and implementing vital UAV equipment.

After successfully interfacing with the FCU, the system was capable of acquiring both raw as well as filtered IMU readings. The FCU was able to successfully send to the system inertial readings at a rate of 25 Hz. The quality of the readings was acceptable as the Roll and Pitch angles reported accurate measurements with a variance of less than 0.1° while the Yaw axis presented variance of about 1.1°. The tests were performed with the aircraft stationary.

Those remarks lead to the conclusion that additional IMU is not required as the IMUs featured on the FCU satisfy the requirements of this study. Moreover, by utilizing the integrated IMU, the navigation system is allowed to have exactly the same inertial frame of reference as the FCU resulting in less required floating-point calculations.

Visual-Depth Sensor

The localization system of this study tackles the need for accurate positioning even in GNSS denied environments. Despite the absence of GNSS signals, many environments provide distinct optical features. These features can be interpreted as space referencing points. By estimating their translation and rotation relative to the UAV, the system can perform localization tasks.

Stereo cameras are adequate sources of input. Stereo cameras are modules that facilitate two identical camera modules with a known baseline distance between them. By finding common features on both camera frames, given the parameters of the cameras as well as the baseline distance, geometric transformations can estimate the 3D translation and rotation of the feature relative to the stereo camera module. In a way, they imitate the way our depth perception works.

To clarify the way depth perception occurs with stereo modules, a visual example is supplied in figure .
Two identical cameras are placed at a known distance (baseline T) while having the same orientation. A point in space (P) is visible by those two cameras. Let O_l be the optical center of the left camera and O_r the optical center of the right camera, while Z is the unknown depth of the point P relative to the stereo module and f is the focal length of the sensors. At last, X_l is the distance of the projected point P_l in the left camera frame relative to the center of the frame, and similarly X_r is the distance of the projected point P_r in the right camera frame relative to the center of the frame.

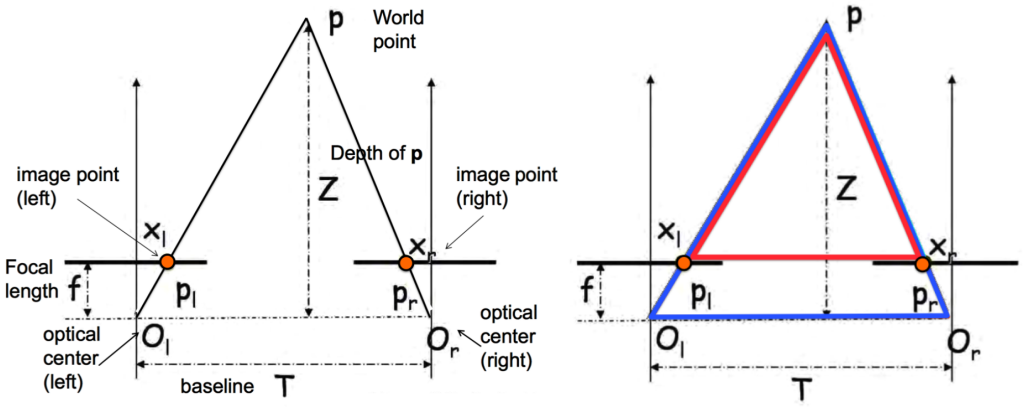


Figure 4.16: Stereo depth perception [source: Penn State University]

By examining the blue and red triangle of the figure, it can be determined that these are similar triangles. Given the nature of these triangles, geometry dictates that :

$$\frac{T}{Z} = \frac{T + Xr - Xl}{Z - f} \Leftrightarrow Z = \frac{f \cdot T}{Xl - Xr}$$

The depth extraction pipeline process (Figure) initially requires synchronized feed from both the left and right camera. By identifying and matching feature points between the acquired image pair and finding the relative distances in the camera space of reference, results in the disparity map. By taking into account the parameters of the camera, depth for each individual point can be estimated. This process requires knowing both the extrinsics as well as the intrinsics parameters of each camera, as different cameras feature different lenses and different sensors. In order to acquire the camera parameters, a camera calibration process [https://www.mathworks.com/help/vision/ug/camera-calibration.html] needs to take place. Despite the fact that stereo cameras provide the information required for depth estimation, the depth extraction process pipeline can be very expensive in terms of processing power required for real-time performance.

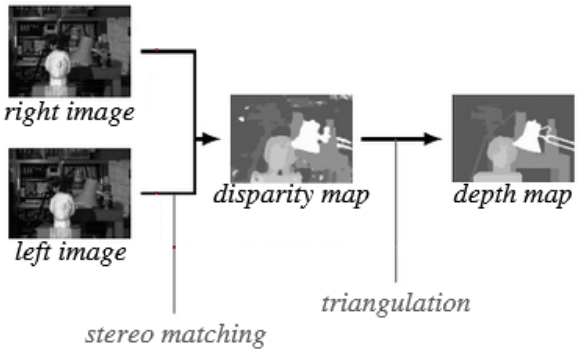


Figure 4.17: Depth estimation pipeline

RGB-D cameras based on stereo modules, feature depth perception with an added value. They can execute the complete depth estimation pipeline inside their integrated circuitry. The depth camera selected for the purposes of this study is an Intel Realsense D435 sensor [https://www.intelrealsense.com/depth-camera-d435/]. This sensor suits the needs of this study as it provides a Field of View (FOV) of 87° × 58°, while providing a maximum depth estimation rate of 90 Hz and supports a maximum resolution of 1280 X 720 pixels as mentioned in the datasheet [https://www.intelrealsense.com/wp-content/uploads/2020/06/Intel-RealSense-D400-Series-Datasheet-June-2020.pdf].

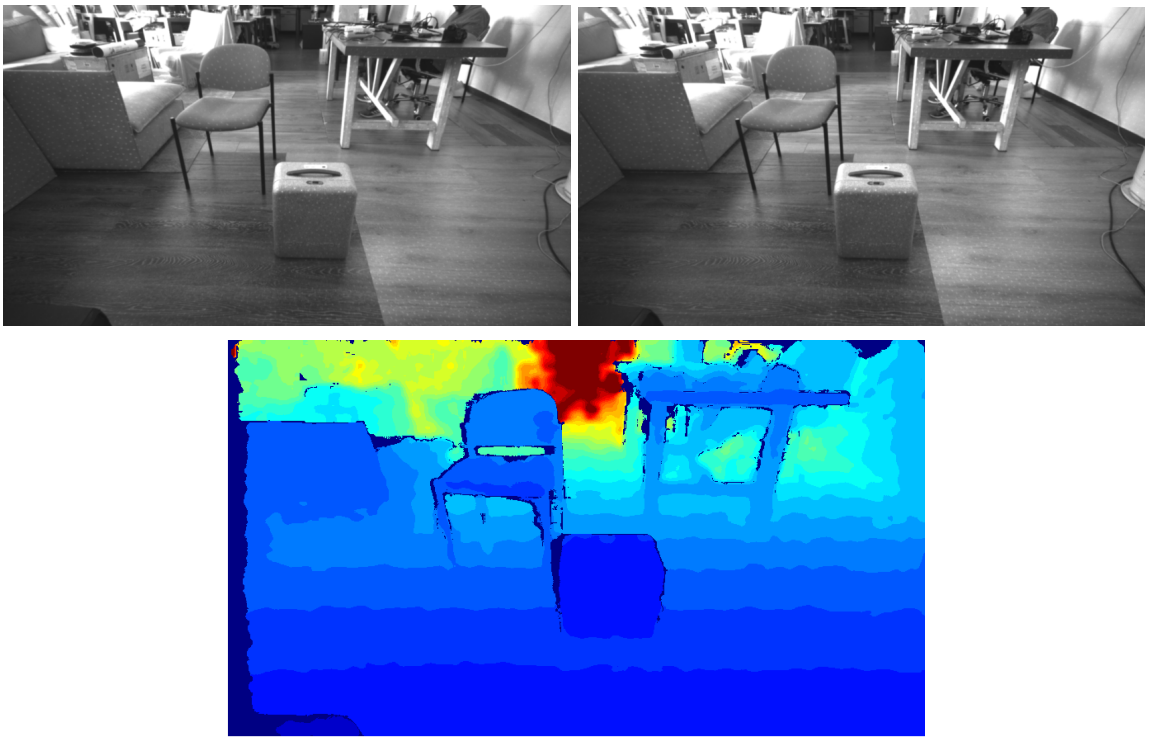


Figure 4.18: Stereo Left and Right images (Upper image) taken by an Intel d435 depth camera module. By processing the stereo feed, this module produces a depth image (Lower image). The colormap on the picture above indicates distance from the sensor with a color map ranging from blue to red. "Near" appears blue whereas "far" appears red.

Parameter	Value
Type	RGB-D
Depth Estimation	Stereoscopic with infrared laser projector
Depth Resolution	1280 x 720, 16 bits
Depth Accuracy	<2% at 2 m
Max Range	10 m
Interfacing	USB 3.1 via USB-C
Power (max)	3,5 W
Weight	72 g
Dimensions	90 mm × 25 mm × 25 mm

Table 4.8: Intel RealSense D435 key parameters

Facilitating the sensors required for the localization submodule of the navigation system is a crucial and challenging task, as the sensors need to be mounted in an efficient way in terms of weight balancing, as well as to allow minor modifications for experimentation. In order to satisfy these requirements, a custom mount was designed and 3D printed (Figure). This mount is an integral part of the complete navigation module which enables the integration of both the on-board processing unit of the system (see section), as well as the RGB-D camera.

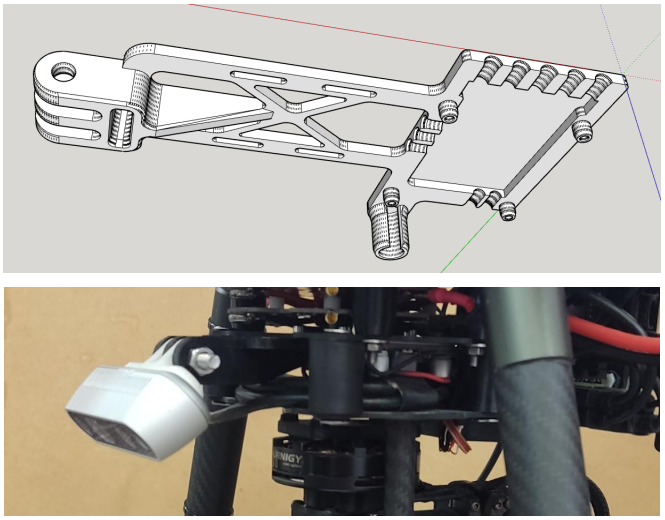


Figure 4.19: Mounting mechanism for the on-board processing unit and the RGB-D sensor.

4.3.2 Localization Approach

The localization module is responsible for processing the data provided by the available data streams, while providing a single output stream, conveying the localization data to the system. The localization module of this system is built as a set of distributed ROS nodes. In this section, the localization_core node is presented. The vdio_slam node is presented in the following section (section 4.3.3).

Localization Core

Inside the ROS environment, available data streams convey vital information regarding position and orientation estimations of the UAV. The role of the localization_core node is to monitor the quality of these estimations for each available system and decide which one will be used for localization. As the architecture of the localization module mandates, the system must support estimations provided primarily by GNSS while allowing other nodes

to provide localization estimations as secondary sources. Such a node is the vdio_slam which provides localization based on visual, depth and inertial measurements.

The operation of this node is highly important as the localization_core needs to continuously monitor the available streams in order to be always up-to-date, regarding both the localization availability and quality. Moreover, it is the responsibility of this node to feed the flight control unit with localization data.

As the node initializes, the origin (0,0,0) (X, Y, Z) of the local three-dimensional space is referenced with global coordinates of a fixed point described in the form of WGS84 (latitude, longitude, altitude). This process can be done either automatically or manually. The system is able to accept as origin the first valid satellite fix which also satisfies the quality parameters set by the user. In addition, the origin point can be supplied to the system as a user-specified parameter.

The navigation system of this study complies with the rules of REP 103 [https://www.ros.org/reps/rep-0103.html] and REP 105 [https://www.ros.org/reps/rep-0105.html], hence the ENU convention is used for referencing the local three-dimensional space. As indicated in figure , the local X, Y, Z axes, align with East, North, Up accordingly. In such a case, X increases while travelling eastwards, Y increases while travelling northwards and Z increases while moving upwards.

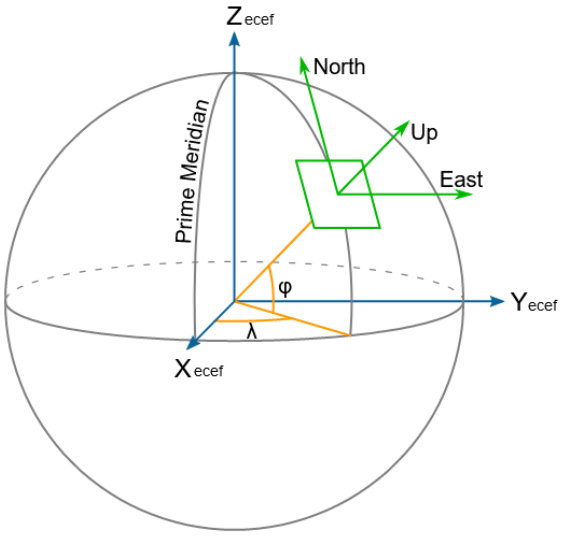


Figure 4.20: ENU referencing system

The localization_core operates as a multiplexer controlled by a state machine. By monitoring the available localization streams, it selects as the systemwide localization source, the system with the highest priority which also reports healthy status. The system with the highest priority is the GNSS. Next follows the SYS_0 input source, leaving SYS_1 as the input source with the lowest priority. SYS_0 and SYS_1 are sources that can be supplied by other nodes such as the vdio_slam, which was designed and implemented for the purposes of this study. SYS_1 is an input source for further interfacing with other localization nodes. In [9], a documentation can be found, regarding interfacing with the localization module.

As the node monitors the available streams, it can be in one of the following states:

- GNSS HEALTHY: GNSS provides reliable data
- GNSS WARNING: GNSS provides unreliable data
- SYS_0 HEALTHY: SYS_0 provides reliable data
- SYS_0 WARNING: SYS_0 provides unreliable data
- SYS_1 HEALTHY: SYS_1 provides reliable data

- SYS_1 WARNING: SYS_1 provides unreliable data

The localization_core broadcasts continuously in a systemwide fashion, it's status containing the latest valid localization data. The idea is that other nodes will be able to receive such data and update their internal knowledge regarding the localization state. Those nodes will continuously report their latest positioning estimations and status to the localization_core and in case a warning arises on one subsystem, another one can take over.

The system can be configured to operate based on GNSS measurements when certain criteria are satisfied. The criteria selected for this study are the available satellite count and the pDOP values, supplied by the GNSS receiver. The configurable parameters are:

- **Minimum Satellites Count**: The minimum number of satellites which are considered acceptable for positioning regarding a specific flight. Available satellite counts below this value trigger the GNSS_WARNING state. The available input range is [5, 20].

- **Minimum Satellite count drop rate**: This value indicates the satellite count drop percentage with respect to time, which triggers the GNSS_WARNING state. The value assumes 1 s as a time constant. The available input range is [1, 99].

- **Maximum pDOP**: pDOP (position Dilution of Precision) is a dimensionless value supplied from the GNSS receiver which indicates the quality of the position estimations. Values closer to 1 are considered high quality whereas values near 100 are considered extremely low quality. When a higher pDOP is reported relative to this value, the GNSS_WARNING state is triggered. The available input range is [2, 99]

- **Minimum pDOP increase rate**: This value indicates the pDOP increase percentage with respect to time, which triggers the GNSS_WARNING state. The value assumes 1 s as a time constant. The available input range is [1, 99].

- The user can select which parameters to set. If the user does not specify any parameter, the system will operate continuously with the GNSS.

<u>Policies which apply on SYS_0 and SYS_1 input:</u>

- Each input is responsible for reporting its status to the localization_core, via the defined **TakeoverPose.msg** ros message [9].

- If each system fails to update the localization_core in a defined time window, it will automatically trigger a SYS_N_WARNING.

The available states of the localization_core can be found below (Figure). Moreover, the state switching algorithm is also illustrated in the following figures. For the purposes of this study, the SYS_0 will correspond to the output of the vdio_slam and the SYS_1 will be left unconnected.

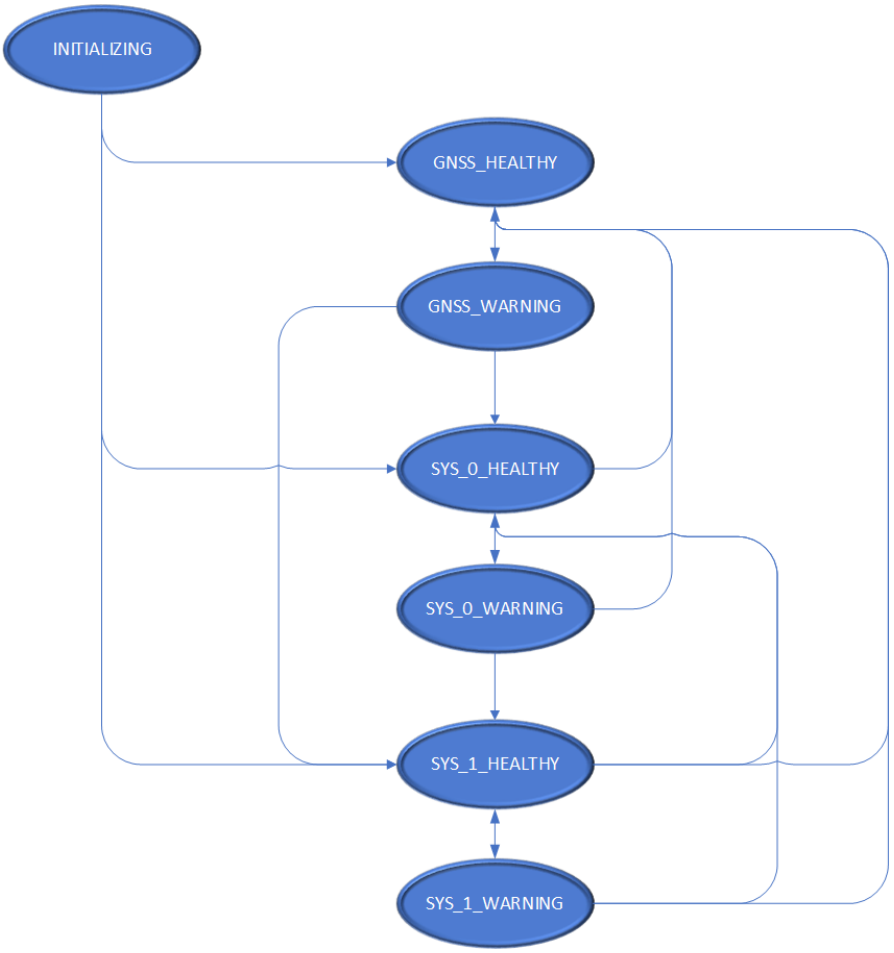


Diagram 4.2: The possible localization states

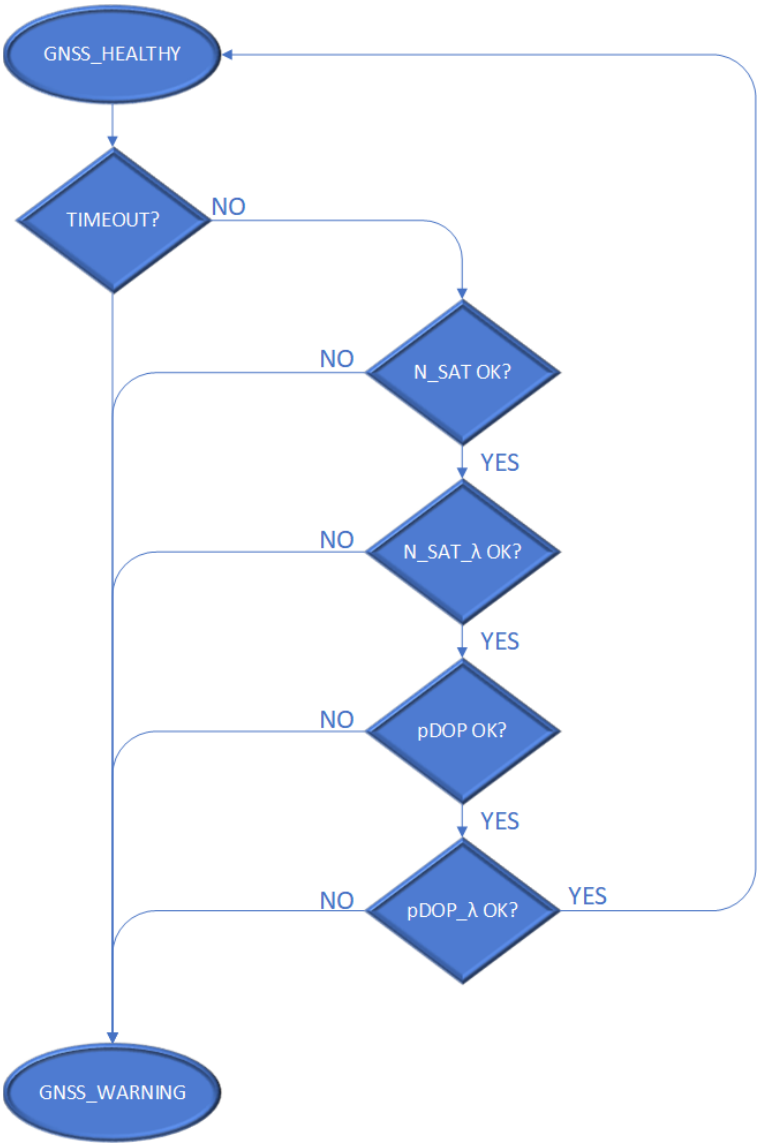


Diagram 4.3: GNSS health evaluation

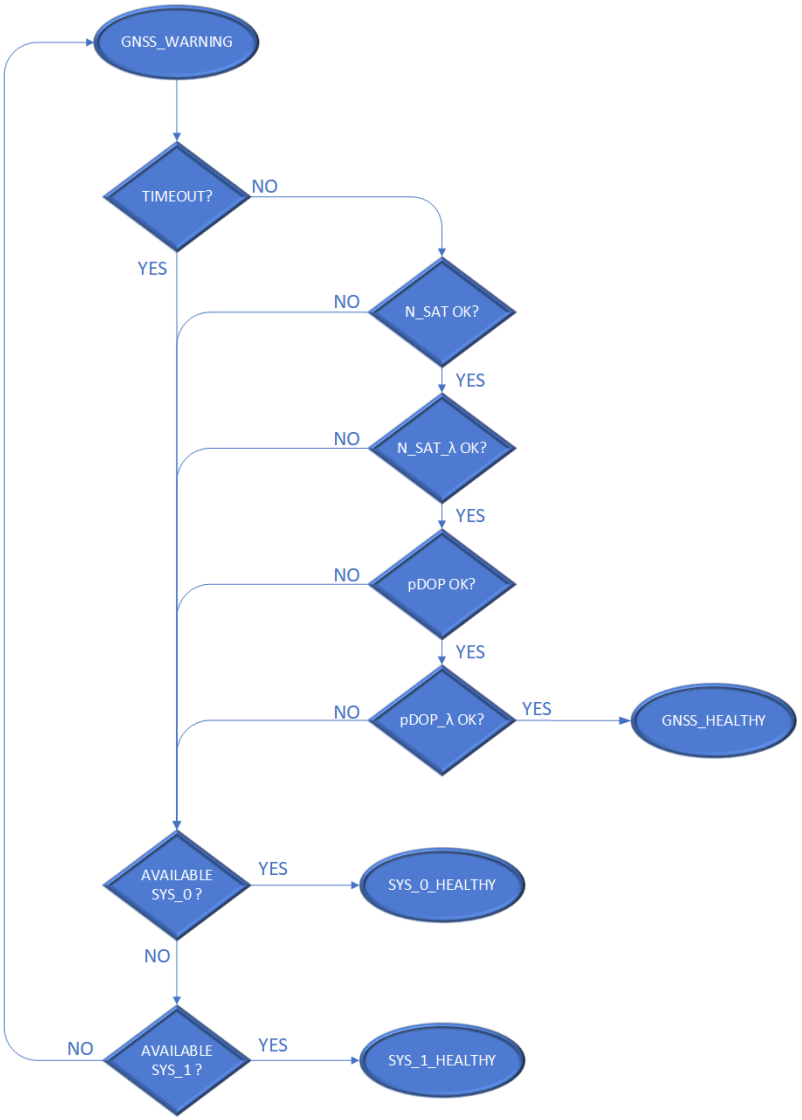


Diagram 4.4: State switching in GNSS_WARNING

This node houses two callback queues, handled by two individual threads. In the primary queue are assigned the callbacks that are related to the selected localization source, and the rest are assigned to a secondary queue for monitoring purposes. After each time the localization system changes, so change the assigned callbacks on those two threads. This design results in the minimum amount of dropped messages.

4.3.3 Depth Inertial SLAM Approach

Operating a UAV in GNSS hard environments can be challenging. In order to tackle the issue of low quality GNSS measurements, a slam ROS node was created. This node is called vdio_slam as it uses Visual, Depth and Inertial data to provide odometry readings. Vdio_slam is primarily based on the ORB_SLAM2 [ORB] algorithm.

By utilizing this node, the UAV will be able to gain spatial awareness by moving through space and acquiring visual features as spatial points of reference (Figure). The input streams of this node are the visual and depth frames of the RGB-D sensor as well as the IMU data stream of the UAV.

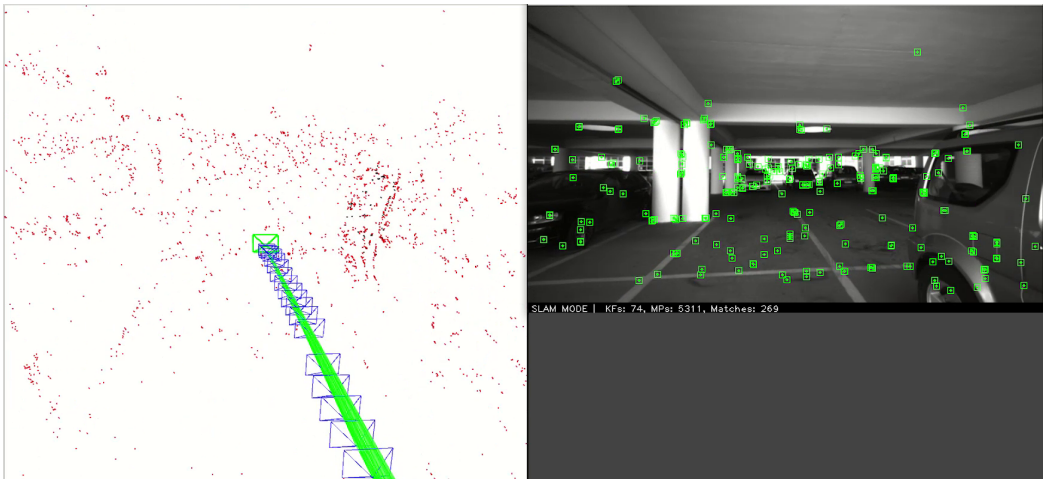


Figure 4.21: Spatial awareness gained by ORB_SLAM2 on RGB-D sensor

The visual and the depth streams are fed directly to the ORB_SLAM2 pipeline. The execution of the algorithm is expensive in terms of performance. After performing tests with the highest resolution settings, the on-board processing unit was able to perform pose estimation on the surrounding area at a rate of 11 Hz given the input streams were set to 15 Hz. Such performance can be sufficient for positioning use cases. However, many applications require higher rates. Increasing the localization rate can result in higher overall stability of the system.

In order to increase the pose estimation rate, the idea was to take advantage of the IMU data. The flight controller can easily provide IMU readings at 25 Hz. Hence, another thread was created to handle the IMU readings. The idea is simple. Integrating the readings of acceleration in time, results in velocity. Integrating velocity results in translation. The angular velocity supplied by the IMU was also taken into account. As the ORB_SLAM2 runs on the first thread, the other provides estimations based on the IMU. By taking advantage of the IMU, the output rate of the node reached a stable 32 Hz pose estimation output.

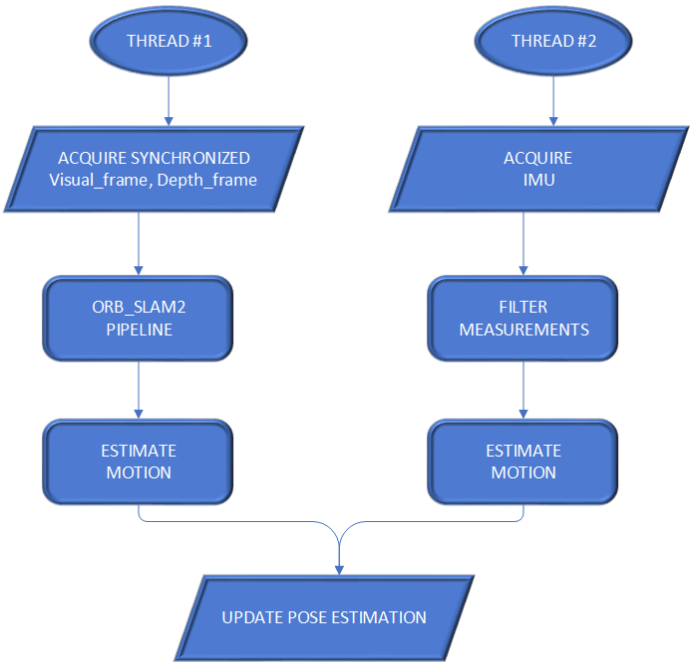


Diagram 4.5: Pose estimation on vdio_slam

The vdio_slam_node interfaces with the localization_core to exchange vital data throughout the flight. As the localization_core reports GNSS_HEALTHY or SYS_1_HEALTHY state, the vdio_slam resets the pose estimation and sets the origin of the tracking, on the latest reported pose provided by the localization_core. In case the localization_core reports a WARNING state and the vdio_slam is able to perform pose estimation, a HEALTHY output will be forwarded.

4.4 Geolocation module

UAVs are usually utilized in scenarios where providing the position of objects in space is important. Examples of such use cases include 3D mapping, topographic surveys, situational awareness during crises as well as search and rescue missions. By taking into account the need for UAV utilization in such use case scenarios, this study provides a geolocation solution.

The localization module (Sect) implemented in this system enables the positioning as well as the navigation of the UAV. By combining the localization module with the geolocation module presented in this section, the navigation system designed in this study can provide precise positioning information not only for the UAV, but also for other objects in the flight area.

As the UAV flies through space, it comes across objects which require geolocation as a mission objective. In order to geolocate the object of interest, the UAV needs to initially estimate the position of the objects relative to the aircraft. In order to perform a successful geolocation during flight, the module must be able to track the objects autonomously and perform relative distance and angle measurements with the objects. By applying geometric calculations, the position of the objects is initially found relative to the aircraft. After associating those measurements with the UAV localization data, the objects can be referenced with a fixed frame of reference.

4.4.1 Hardware

Geolocation is a process which requires adequate precision and accuracy. Hence, this study tackles such needs by facilitating suitable hardware. In order to detect the object of interest, the on-board computer unit should initially process a video feed coming from the geolocation module. As maneuvering the UAV can be challenging on its own, the geolocation module must be capable of executing the complete geolocation process autonomously.

As the architecture of the module dictates, the sensory components of this module must be installed on a motorized gimbal system which allows three degrees of freedom, composing the geolocation apparatus. The apparatus must be reliably controlled via software in order to successfully detect, track and estimate the position of objects of interest. The hardware components which enable the required functionality are described below.

Gimbal System

The gimbal system (Figure) designed and implemented for the purposes of this study, acts as the frame of the geolocation apparatus. In order to facilitate all the sensors required for data acquisition, the gimbal system was fabricated with carbon fiber tubes and links which provide structural integrity and low weight. The gimbal system was designed to facilitate the sensors required for this study while allowing spare space for future add-on sensors. The platform features three brushless DC (BLDC) motors, responsible for rotating the frame around roll, pitch and yaw axes.

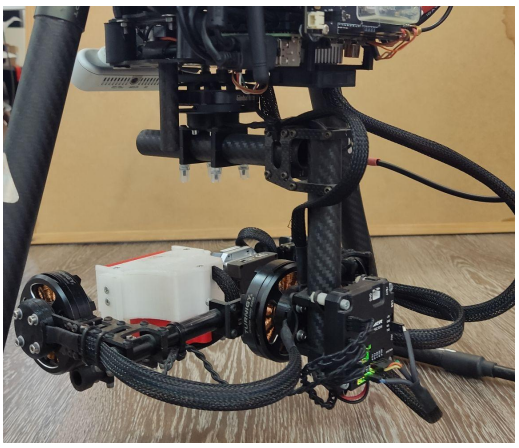


Figure 4.22: The gimbal frame of the geolocation apparatus

The implemented gimbal system has the ability to provide stabilizing functionality, when tracking is not engaged. In order to ensure precise stabilization, a BaseCam SimpleBGC 32-bit [https://www.basecamelectronics.com/simplebgc32bit/] (Figure) gimbal controller was utilized. This controller features an ARM Cortex M4 clocked at 72 MHz, capable of controlling and stabilizing the motorized gimbal frame. In addition, the controller board is capable of driving the motors as it features integrated Electronic Speed Controller (ESC) FETs. The controller executes a Proportional Integral Derivative (PID) algorithm to continuously apply corrections to the platform. Originally, the PID algorithm was receiving feedback solely by an IMU sensor mounted on the platform. However, after just a few minutes of operation, the system would start to drift dramatically around the yaw axis. This problem occured as the IMU installed on the platform, did not feature an integrated magnetometer in order to have an absolut reading regarding the heading of the module. To resolve this issue, magnetic rotary encoders were added axially on each motor. The output of each encoder was fed into the PID algorithm and the drifting was eliminated.

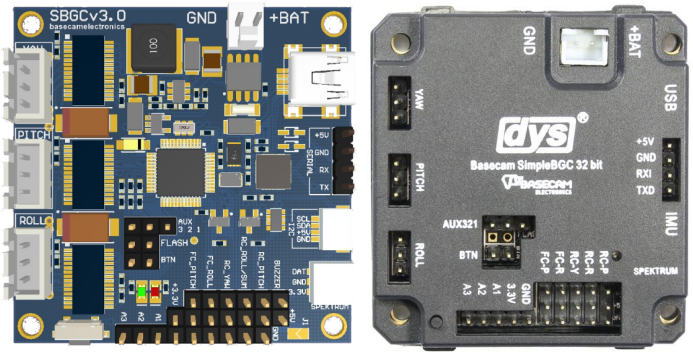


Figure 4.23: Basecam SimpleBGC 32-bit controller

This controller features USB 2.0 and UART for interfacing with other systems as control sources. For the purposes of this study, the on-board processing unit communicates with the gimbal controller via UART, in order to feed angular set-points to the control algorithm. The controller can handle both absolut angle and angular speed setpoints, for each axis of rotation. Moreover, when the object tracking mode is not engaged in the processing unit, the gimbal system is configured to follow the yaw rotation of the UAV during flight. A block diagram of the gimbal system can be found below.

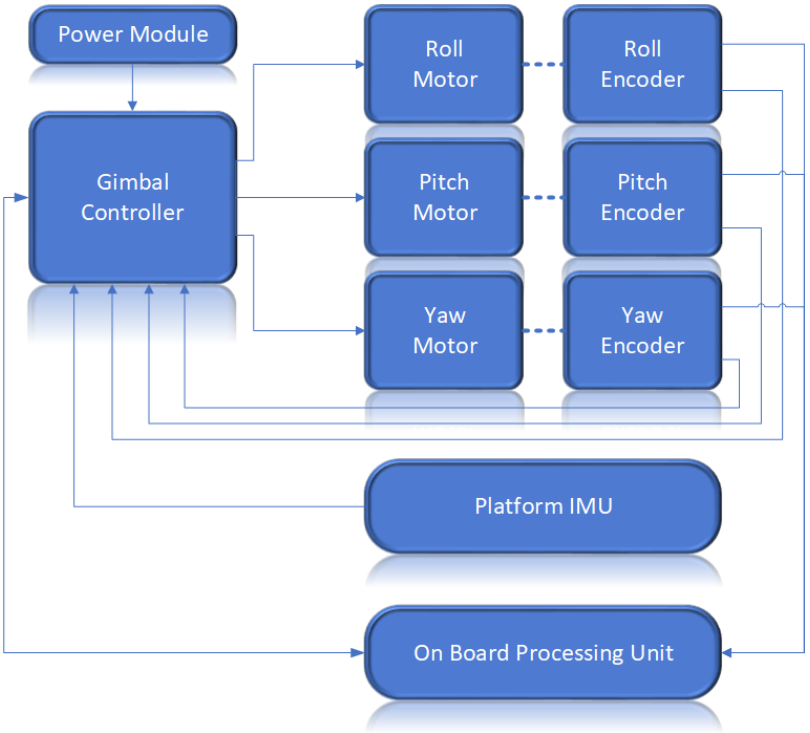


Diagram 4.6: Gimbal Block Diagram

Encoders

Both the accuracy and precision of the geolocation module are important. These attributes depend on both the hardware and software used when estimating positions. Feeding the geolocation algorithm with sufficiently accurate and precise readings is really helpful. Hence, in order to supply the processing unit with adequate measurements regarding the relative angles between the UAV and the orientation of the geolocation apparatus, magnetic rotary encoders were integrated into the system.

During the design phase, it was clear that the use of encoders could solve two problems simultaneously as the gimbal drifting issue was not resolved, yet a reliable angle measuring mechanism was not present. The integration of encoders proved to be an excellent design choice as they are now utilized for both feeding the processing unit and the gimbal controller with reliable angular data.

The selection of the encoders was made by taking into account the specifications of the gimbal controller, the precision and accuracy of each candidate module as well as their ability of supporting at least two concurrent interfacing options. The module which satisfied the requirements while providing a low price point as well as a small form factor, was the AMS AS5048A [https://ams.com/as5048a] rotary position sensor.

As the datasheet of the AMS AS5048A module suggests [https://ams.com/documents/20143/36005/AS5048_DS000298_4-00.pdf/910aef1f-6cd3-cbda-9d09-41f152104832], the sensor features contactless absolute angle measurement, by sensing a 2-pole magnet which rotates either above or below IC (Figure). The module provides a resolution of 14-bit (16384 positions/revolution), resulting in a precision of around 0,022°. This figure is acceptable, as the established precision constraint is 0,05° per axis. Additionally, this module provides concurrent SPI and PWM interfacing. As a design note, the SPI provides full resolution 14-bit data, whereas the PWM supports 12-bit. By taking that piece of information into account, the SPI will be utilized for interfacing with the on-board processing unit and the PWM will be fed to the gimbal controller.

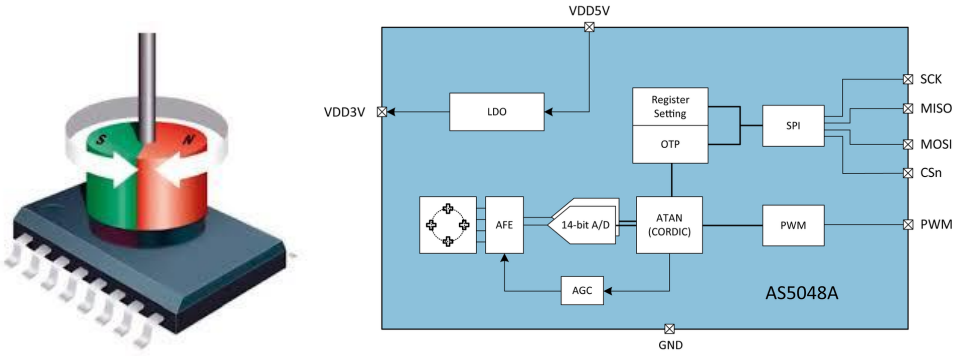


Figure 4.24: Operation of the AS5048A

Integrating the rotary sensors to the gimbal frame was a challenging task as. A magnet needed to be attached on the axis of each motor and be centered, as the sensor provides optimal linearity when the center of the magnet aligns perfectly with the center of the sensor [datasheet]. The center of the sensor for this type of magnet is considered a circle area with a diameter of 0,25mm (Figure). Moreover, the magnet should be placed at a distance between 0.5mm to 2.5mm from the package surface. The magnet selected for the application was a 6 mm diameter by 3 mm height cylindrical neodymium magnet, capable of providing sufficient magnetic flux of between ±30mT and ±70m, on the die surface.

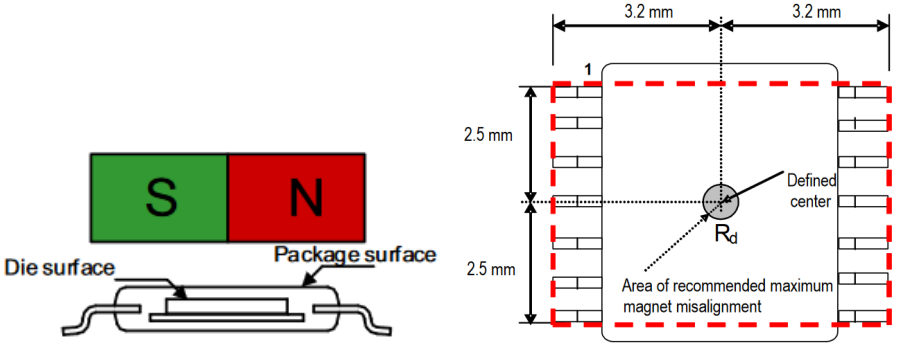


Figure 4.25: Magnet and sensor alignment

In order to integrate the ICs of the sensors, custom PCBs (Figure) were designed and fabricated in order to fit the needs of both the sensor and the gimbal motors. Furthermore, detailed custom mounts were also designed and 3D printed, to successfully align each motor with each sensor.

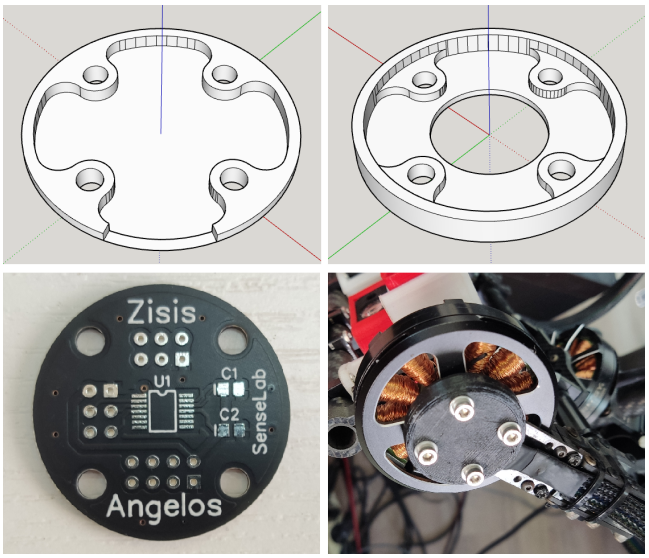


Figure 4.26: Fabricated components, designed to properly install the AS5048A sensors on the motors

Camera and Ranging

The geolocation module must be able to perform autonomous detection and tracking of objects of interest. For the purposes of this study, the module will rely on visual characteristics in order to perform object detection. To achieve this goal, the use of a camera is required. Moreover, the module must be capable of measuring the relative distance between the geolocation apparatus and the object of interest. To satisfy that requirement, the integration of a single point distance sensor is appropriate.

As the camera sensor, a Logitech C270 module was selected. The selection of the camera was primarily based on the featured resolution, field-of-view (FOV) and maximum frame rate. The Logitech C270 camera features resolution 1280 x 720 pixels, diagonal FOV of 55° at a maximum frame rate of 30 frames per second (fps). The camera offers satisfactory characteristics while being a cost effective option.

Regarding the rangefinder, the architecture of the geolocation module mandates that the sensor must be capable of measuring distances of at least 30 m with an accuracy of less than 1 cm. The sensor module selected to tackle the single-point ranging is an OEM laser rangefinder. According to the documentation provided for this sensor [http://bit.ly/LRFDocs], the sensor satisfies the criteria, as it supports measuring distances of 40 m, while offering an accuracy of 3.5 mm at that distance. Additionally, this module can be accessed directly by the on-board processing unit via UART. More information about the laser rangefinder can be found in the following table.

Parameter	Value
Maximum range	40 m
Resolution	0.01 mm
Accuracy	Less than 10m: 2 mm More than 10 m: 2 + 0.05 * (D - 10) mm
Laser Type	620-690 nm, Class II, < 1 mW
Spot Diameter	6 mm at 10 m
Weight	60 g
Dimensions	48 mm x 37 mm x 18 mm
Interfacing	UART

Table 4.9: Laser rangefinder key parameters

As a design note, it can be really helpful for the geolocation process to align the optical center of the camera, with an infinitely far approximated point along the measuring direction of the distance sensor, while keeping the camera as close to the distance sensor as possible. This process will allow the tracking algorithm to continuously maintain the object of interest in the center of the camera frame, while simultaneously acquiring distance measurements.

By taking into account the aforementioned design note, the camera and the distance sensor are attached to a custom made structure (Figure) which allows fine tuning of each individual module. This mounting structure was designed and 3D printed, according to the technical drawings of both sensors.

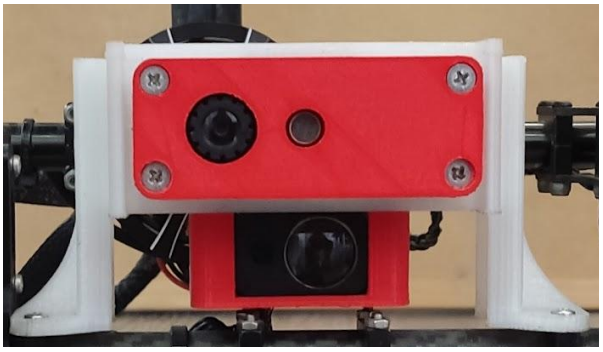


Figure 4.27: Camera and rangefinger

4.4.2 Geolocation Approach

In order to provide an automated geolocation operation, the manipulation of the geolocation apparatus was assigned to the on-board processing unit. A ROS node is responsible for detecting, tracking and estimating the positions of the objects of interest. The node responsible for that operation is the geolocation_node.

The camera mounted on the geolocation apparatus is responsible for feeding the geolocation_node with visual data about the focused section of the flight area. For this study, the objects of interest are represented by Aruco tags [https://www.uco.es/investiga/grupos/ava/node/26] (Figure).

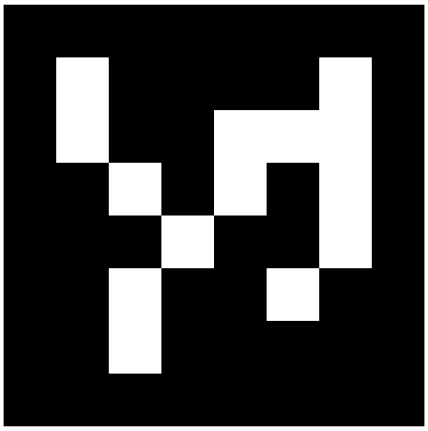


Figure 4.28: Aruco Tag

Such tags provide an effective way of developing both robotic and UAV applications with tracking requirements, as they can be identified by their pattern. In later stages of system development, other object detection approaches can be used.

The automated geolocation process begins after the operator selects the “tracking” mode on the remote control. At any point during the flight the operator is able to make the camera follow the orientation of the UAV if needed. Moreover, the pitch axis can be set easily on the remote control. To provide an easy to use UI for the gimbal operation as well as the geolocation process, the available modes are encoded on a three position switch:

- **Follow Yaw**: By pushing the mode switch fully forward, the operator can set the mechanical apparatus which also facilitates the optical camera required for flight monitoring, to follow the heading of the aircraft. It provides an easy to both understand and remember way of instantaneously setting the camera facing forward. Pushing forward, makes the camera go forward.

- **Pitch Control**: A convenient way of controlling the vertical angle of the camera is the pitch control mode. By setting the mode switch in the middle position, the operator can adjust the vertical angle of the camera via a rotating knob.

- **Tracking**: When the operator moves the switch fully backwards, the tracking mode gets engaged and the control of the gimbal is given to the geolocation_node.

The object identification and tracking algorithm receives the identity of the tag by the operator and tries to locate it inside the area of the frame. After finding the center of the appropriate tag with respect to the geolocation apparatus targeting center, a PID control algorithm takes control of the apparatus. Initially the relative angles of the mechanism and the tag are calculated and fed into the PID algorithm. As the object remains visible, it will be centered by the closed loop shortly afterwards as shown in Figure.

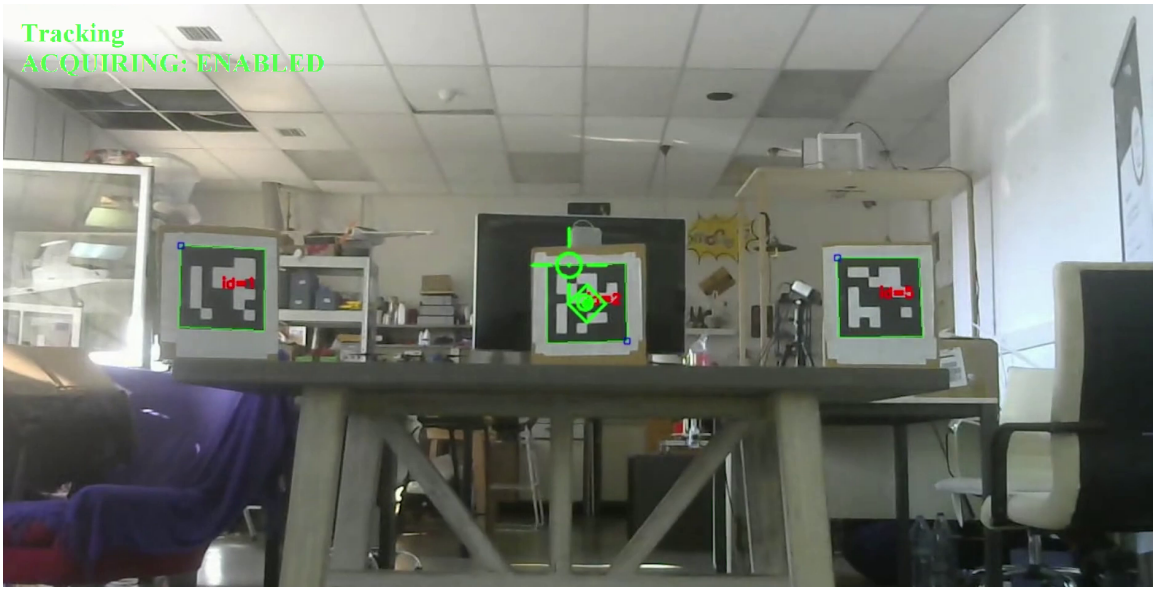


Figure 4.29: Tracking Aruco Tags with the geolocation module

As the center center of the tag aligns with the targeting center of the tracking mechanism and stabilizes, the module is ready to perform geolocation. If the operator has enabled the measurement acquisition, repeated laser pulses from the rangefinder hit the tag and get reflected back to the module. By knowing the speed of light and after precisely measuring the duration of this round trip (also called time-of-flight), the rangefinder provides the estimated distance measurement. Simultaneously, the relative angles are also measured with the use of the installed magnetic encoders. By applying trigonometric calculations based on the readings, a position estimation for the tag is calculated. The operation of the complete geolocation process can be shown below.

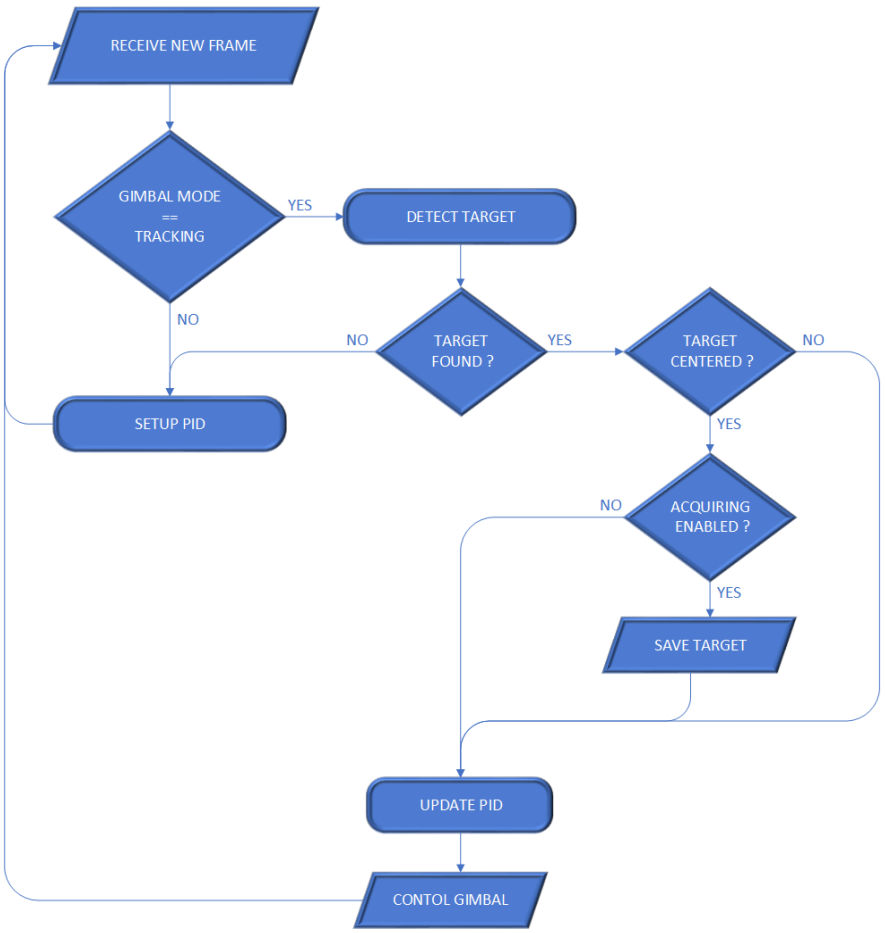


Diagram 4.7: Geolocation Process

4.5 Overview

The overall design of the system features many sensors and mechanisms which are utilized for both the localization and the geolocation purposes of the system. Each module was designed to be integrated into a ROS enabled platform. ROS provides a very convenient way to design individual components and integrate them in various configurations in a top level architecture. The architecture of this system can be found in [9].

Designing and developing with the use of ROS proved to be a convenient strategy as the structure of the system was designed in a bottom-up fashion. This resulted in straightforward developing and testing of the components of the system both individually and as a whole.

The resulting system features localization based on two modules, the RTK enabled GNSS receiver as well as the VDIO-SLAM module for usage in scenarios where GNSS cannot be trusted. In addition, another localization system can be added without changing the source code of the localization_core node as an additional localization source input is implemented. After selecting an appropriate localization source, the node broadcasts the data to the whole ROS environment as well as the UAV for navigation purposes. Such data also reach the geolocation module which is also a ROS component, to contribute to the overall geolocation process.

The module responsible for enabling the communication between the ROS workspace and the aircraft is the MavROS [http://wiki.ros.org/mavros] package. MavROS facilitates the MavLink protocol in order to successfully convey data between the UAV and the system. By utilizing this module, every node is capable of

receiving vital information about the aircraft. Moreover, this package also enables parsing of the localization data produced by the navigation system of this study, to the aircraft.

In order to facilitate the equipment required for both the localization and geolocation purposes, many mechanical components required design and fabrication. The majority of those parts were extruded with PLA by a Fused Filament Fabrication (FFF) 3D printer. The finished prototype is depicted below.

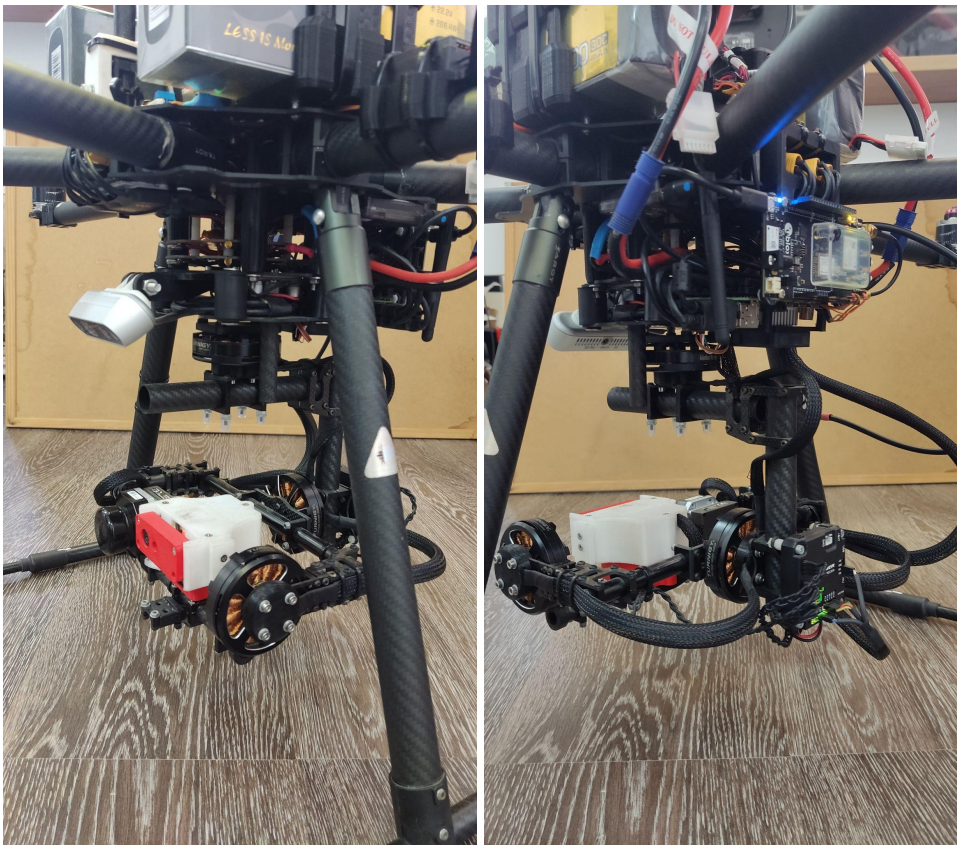


Figure 4.30: The implemented prototype mounted on the UAV

Chapter 5: System testing and evaluation

After the design and implementation phase, the system reached the prototype form and was able to be put to the test. Each module was carefully mounted on the UAV and each connection was double checked. Initially static tests took place in a fully controlled environment. Later, after evaluating the performance of the system on ground, field tests were performed.

5.1 Static testing

Early in the process of designing and developing the system, simulated testing was performed with UAV simulation software. This process was initially performed in order to evaluate the communication information between the on-board processing unit and mavlink enabled flight controllers. The software used is Ardupilot's SITL Simulator (Figure) [https://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html].

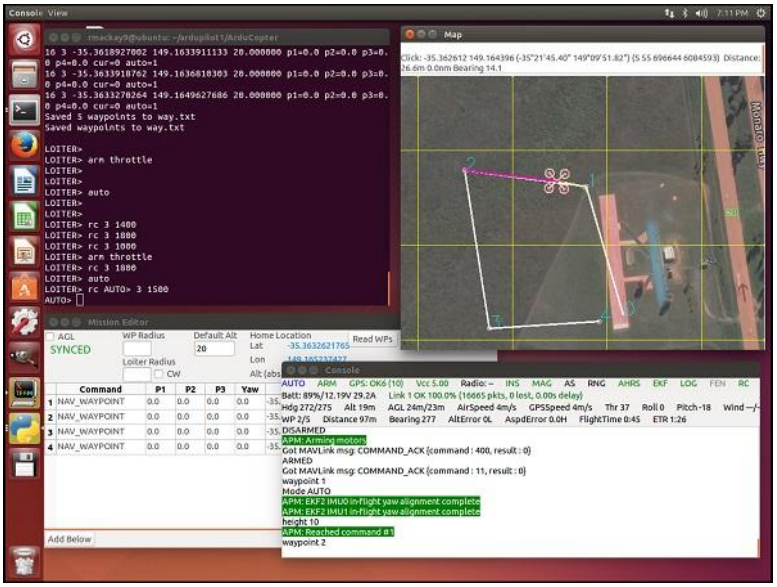


Figure 5.1: Ardupilot SITL Simulator Environment

This piece of software proved to be very helpful as it allowed the integration of the exact firmware version installed in the actual flight controlled unit of the hexacopter. Aside from the communication and interfacing aspects, the simulation software allows the execution of a physics engine. The engine was utilized later in the development as the navigation module was posting navigation data to the simulated flight controller. In that stage it proved a challenging task to provide such data and receive valid results. The localization module based on the vdio_slam was feeding the information, however the simulator could not receive external IMU data available on the navigation system, hence it would always come to the concussion of system warning due to EKF variances.

As the system is built on top of ROS, it is able to take advantage of multiple available developing utilities. A very helpful tool proved to be the rosbag utility [http://wiki.ros.org/rosbag]. As stated in a previous chapter, ROS utilizes topics as forms of message interface. Rosbag is able to record on demand available topics and allow playback later. When designing the localization module, it was convenient to record many topics related to the slam, and verify later that the algorithm can support those visual features as well as to check for scaling, translational and rotational errors.

During the static testing phase, many experiments were conducted in order to find near optimal PID performance for the gimbal. In such tests, step by step PID tuning was performed. After finding parameters which provided both stability and precision, the geometry tree of the system needed to be configured. Each module should be described with extreme precision. That was a detailed process that required millimeter level accuracy.

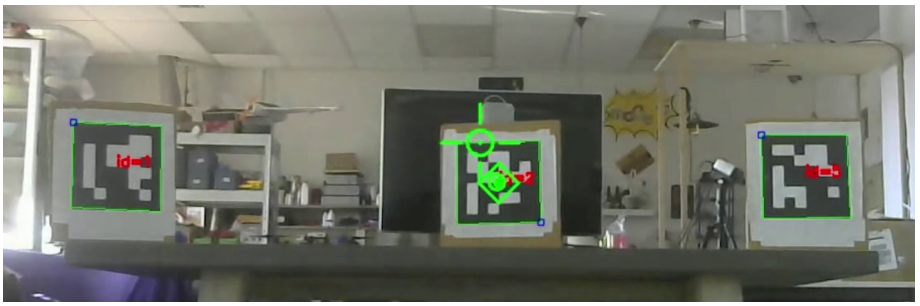


Figure 5.2: Multiple target setup for PID tuning

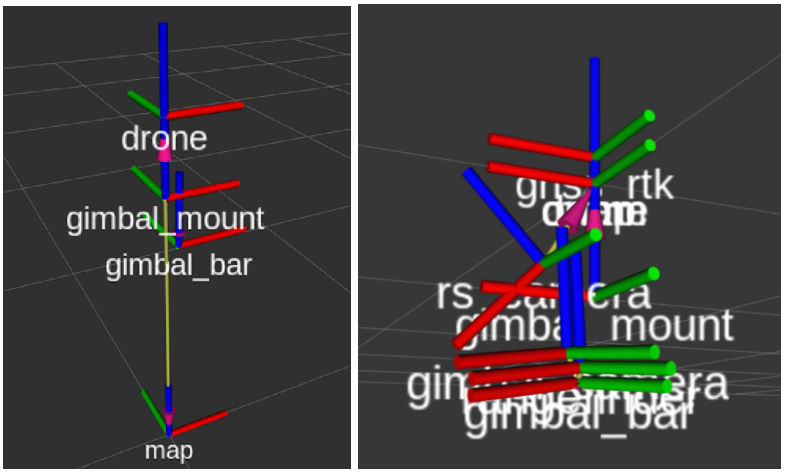


Figure 5.3: Geometric representation of the system

The geolocation module was initially tested indoors. As depicted on the Figure: , after the successful PID tuning, the accuracy of the module was measured at a close range. By individually measuring the local coordinates of all three Aruco tags, the module was able to provide a 3D error of about 2 cm at 6 m from the targets.

5.2 Field testing

Before the prototype was completed, field testing was conducted in order to evaluate the GNSS equipment. Initially the GNSS receiver was a Ublox M8N. This type of receiver cannot be configured with RTK technology, hence it cannot provide high accuracy and precision readings. In order to evaluate the receiver in terms of precision and accuracy, Ublox U-Center [https://www.u-blox.com/en/product/u-center] software was utilized.

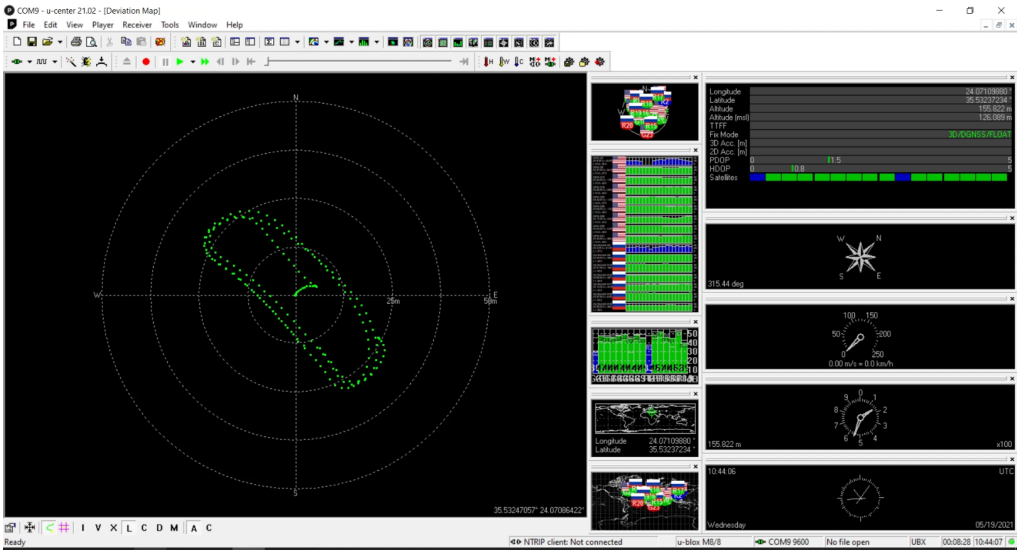


Figure 5.4: GNSS Testing

As executed the receiver could provide an estimated position with an accuracy of around 3 m. EKF integration with an IMU was also tested, however, given the low refresh rate of the sensor, adequate filtering could not be conducted and the EKF would drift after only a few seconds.

The need for higher precision resulted in integrating the Ublox - F9P which managed to successfully provide centimeter accurate measurements. For this purpose, an RTK base station was installed at a distance of 30 m from the receiver, in a position with known coordinates and was able to send the required RTCM-3 messages to the receiver.

Before conducting flight tests with the navigation module installed, the hexacopter was configured with a simulated mass in the place of the navigation system. After successfully performing the initial flights, the PID parameters of the UAV were also configured. The tool which enabled the configuration of such parameters is the Mission Planner software (Figure) [https://ardupilot.org/planner/]. This piece of software provides a user-friendly user interface for both monitoring the status of the aircraft as well as reviewing flight logs and parameters.

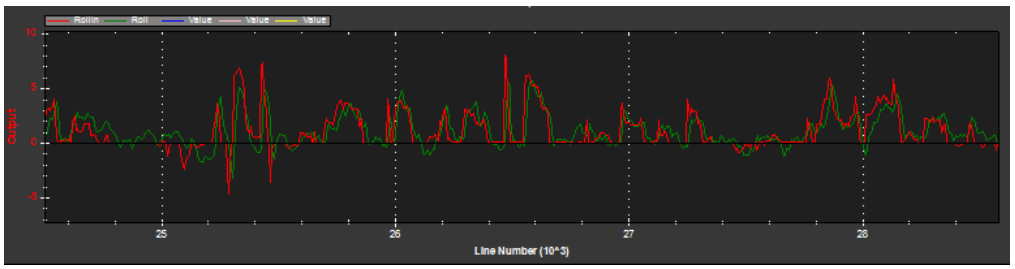


Figure 5.5: Mission Planner ground station software

After successfully flying the UAV (Figure) with improved precision and stability provided by the PID tuning, the navigation module was added to the aircraft. All flights confirmed that the UAV could fly stably with the navigation module mounted.

Figure 5.6: Moments from the flights tests with the navigation module mounted

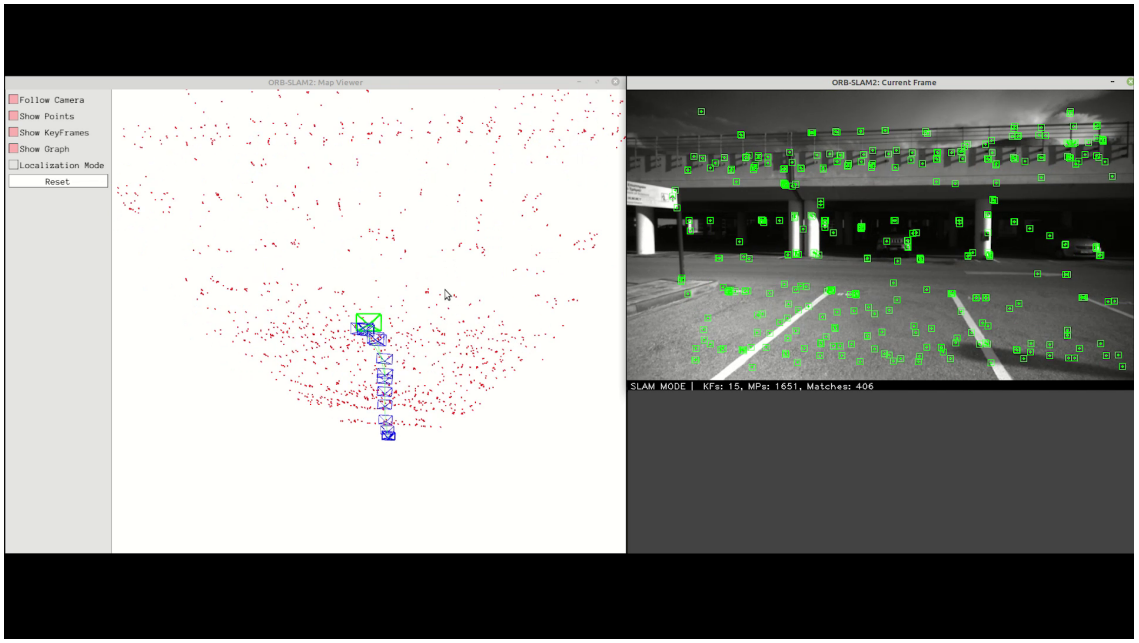
More than twenty flights were conducted in order to verify the stability of the UAV with the navigation module installed. After no sign of stability issues, the geolocation module was tested during flight (Figure).

Figure 5.7: Testing the geolocation module in flight

Tracking the target was more difficult during flight. However, despite the 30% reduction in available measurements when compared to the static tests, the system was able to successfully provide a 3D estimation error of 15 cm from a single position and 8 cm when performing multilateration from 4 different positions on a 8 m radius from the target. The 15 cm provided during that test, can be justified by the error added from the compass of the UAV as it reaches 2°.

During the field tests, flights were performed to evaluate the health of the measurements provided by the navigation module to the UAV. That task proved to be challenging as it required continuous flying initially in manual flight modes to validate the stability of the measurements. During that phase, the UAV was able to receive adequate positioning data without indications of instabilities.

Field tests were also conducted to assess the navigation system in scenarios with very low GNSS coverage. This process required setting up an experiment where the UAV would initially have sufficient GNSS coverage and be very low shortly afterwards. This experiment (Figure) indicated that in short ranges the vdio_slam can be used successfully.



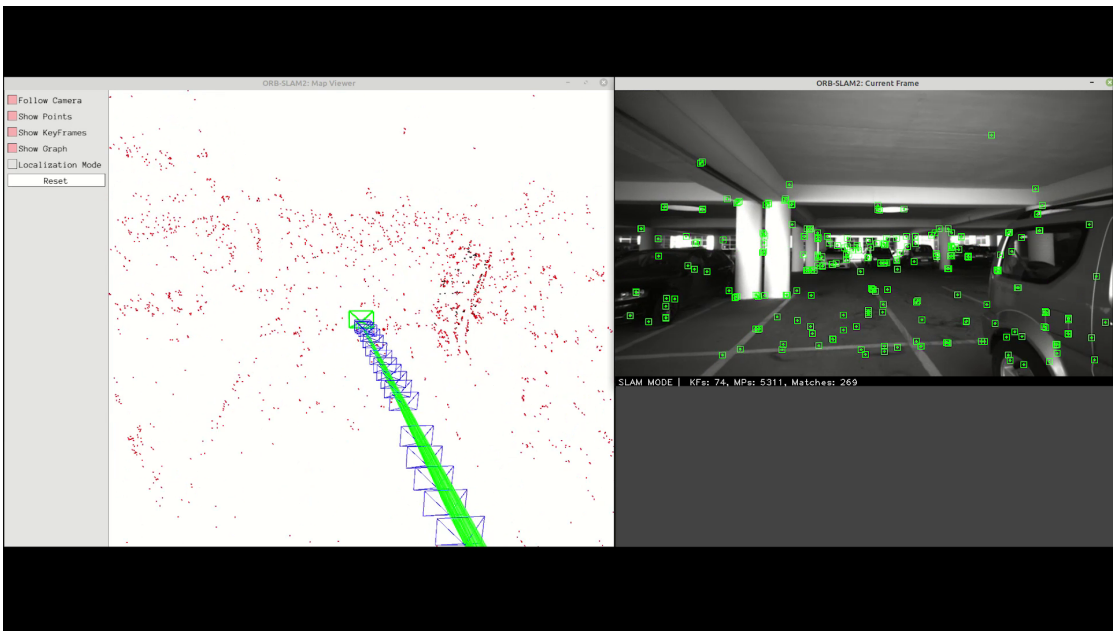
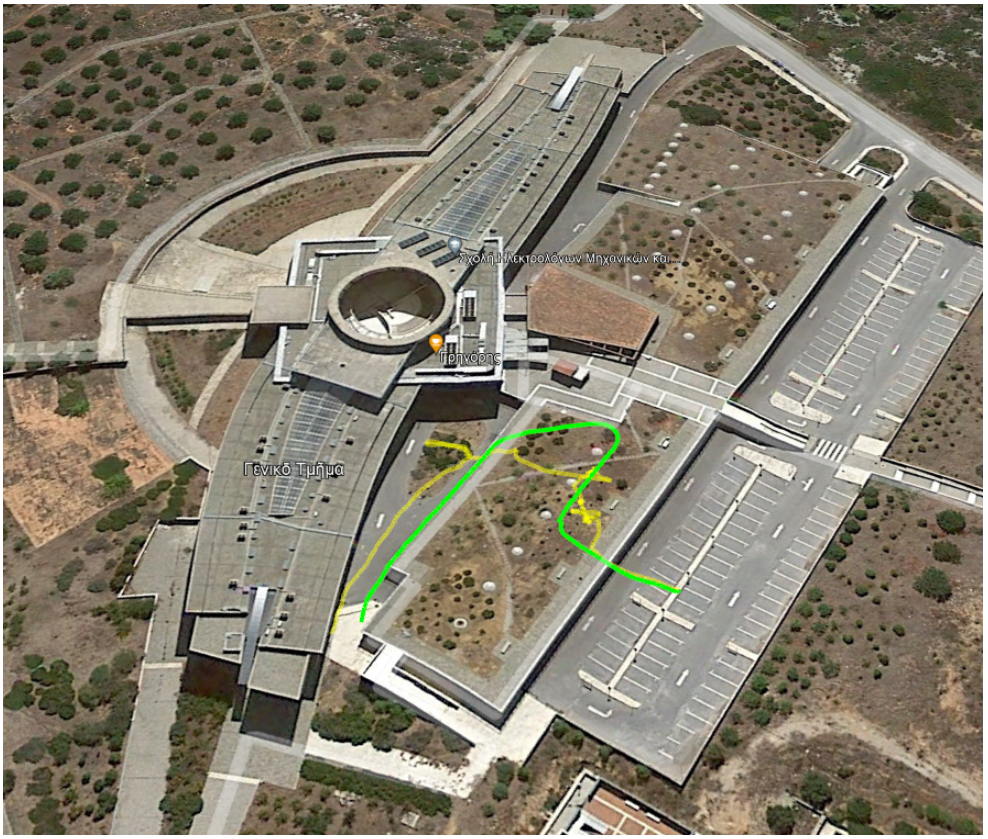


Figure 5.8: Entering a GNSS hard area

In figure, the GNSS captured ground track as well as the vdio_slam ground track are indicated. The comparison of the heigh can be found in the following figure.



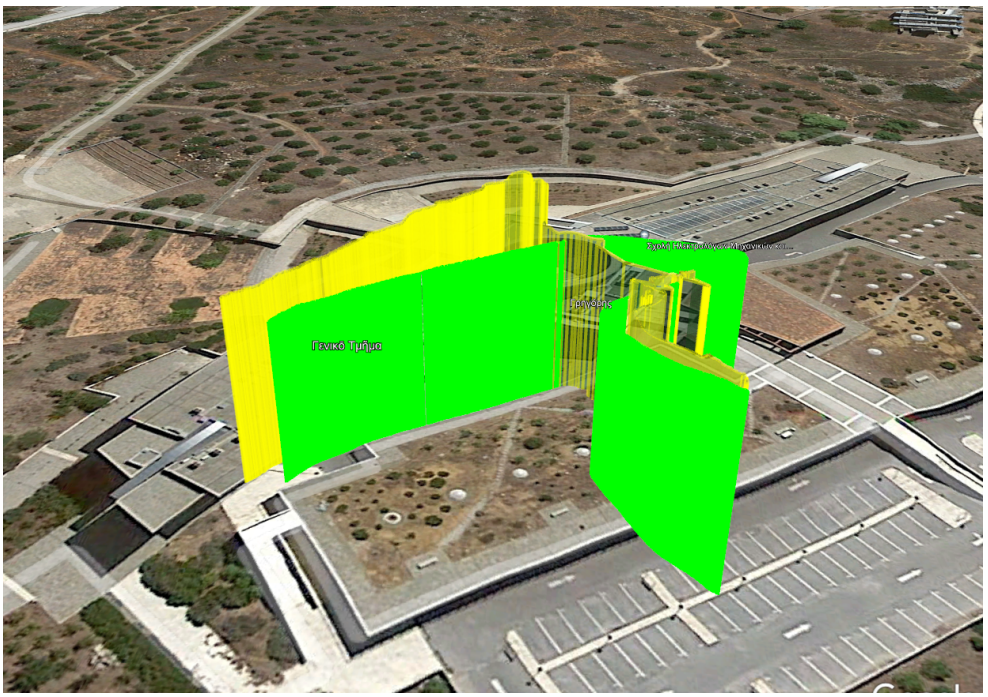


Figure 5.9: (Upper) The yellow line indicates the ground track of the GNSS whereas the green line indicates the ground track provided by the vdio_slam (Lower) The trajectory acquired by the GNSS is illustrated with yellow, whereas green corresponds to the trajectory supplied by the vdio_slam.

Chapter 6: Summary

6.1 Conclusion

The designed and implemented navigation system is able to tackle both localization and geolocation tasks. The system was structured as an auxiliary component of a modern UAV in order to facilitate both the needs of localization and geolocation abilities, in order to provide sufficient navigation utilities.

Regarding the localization aspect of the system, it provides a simple way of integrating separate localization architectures with the use of custom ROS messages as explained in the documentation of the package [9]. The system utilizes a GNSS receiver as primary localization source, and is capable of continuously providing auxiliary mechanisms in case the GNSS cannot provide positioning input. A key element in the design of the localization_core ROS node, which is a vital component of the system, is the prioritization.

With the use of the RTK - GNSS module, missions which require sub-meter level of accuracy can be achieved as the integrated module accepts RTCM-3 correction messages by either a nearby RTK bases or the internet.

The use of the RGB-D (visual-depth) sensor proved to be a suitable choice as it enabled high performance execution of three dimensional perception processes inside the integrated hardware, allowing processing resources available for other demanding tasks of the system.

The geolocation module is capable of continuously tracking the objects of interest and estimating both their local and global position with adequate accuracy. Moreover, as the geolocation module is enabled to save multiple metadata regarding each estimation, post processing is also available via the use of this module. Currently, multilateration is used as a post processing method.

6.2 Future work

Despite the proven abilities of this system, there is always room for improvements as well as added functionality. The system was designed in this configuration in order to allow expandability while maintaining low weight. Improvements and functionality can be added in both the localization and the geolocation submodule as well as the system in general.

Localization

- As a filtering method, Extended Kalman Filtering (EKF) should be integrated which can take as input both GNSS readings and IMU data.

- Regarding the mechanism that estimates the quality of each localization module, modern learning algorithms can be tested as they might provide a more efficient way of both detecting the need to switch and also select the localization module.

Object detection

- Given another on-board processing unit with artificial intelligence capabilities, neural networks should be integrated as an object detection approach.

- In order to tackle search and rescue missions, the system should be upgraded with a thermal imaging camera, as well as object detection capabilities on the thermal-infrared spectrum.

References

[1] ROS: The Robot Operating System

[2] OpenCV: Machine Vision Library

[3] HectorSlam ROS Mapping Package

[4] Raul Mur-Artal, Juan D. Tardos: ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras

[5] Dae Hee Won; Eunsung Lee; Moonbeom Heo: Selective Integration of GNSS, Vision Sensor, and INS Using Weighted DOP Under GNSS-Challenged Environments

[6] D. A. Mercado; G. Flores; P. Castillo: GPS/INS/optic flow data fusion for position and Velocity estimation

[7] Erqian Tang; Sobhan Niknam; Todor Stefanov: Enabling Cognitive Autonomy on Small Drones by Efficient On-Board Embedded Computing: An ORB-SLAM2 Case Study

[8] Ethan Rublee; Vincent Rabaud; Kurt Konolige: ORB: An efficient alternative to SIFT or SURF

[9] The sense_nav ROS package of this study