POLITECNICO DI TORINO

Department of Mechanical and Aerospace Engineering

Master of Science in

AEROSPACE ENGINEERING

MASTER DEGREE THESIS

Obstacle Detection and Avoidance Techniques for Unmanned Aerial Vehicles

Design and Simulation of a Reactive Collision Avoidance System

CANDIDATE:

ERIKA MAI

Student number: 255596

SUPERVISORS:

Prof. GIORGIO GUGLIERI

Ing. GIANLUCA RISTORTO

Academic Year 2019-2020

Abstract

Unmanned Aerial Vehicles (UAVs) represent a high potential for many fields of applications, from surveying to search and rescue missions, from delivery to photography. For all these applications, the interest in this field become higher for those that require automatic missions and autonomous flight operations. In order to consider autonomous flights, one of the most essential requirements is the capability of detect and avoid obstacles through reliable mechanisms. Moreover, it is becoming necessary to develop collision avoidance systems capable of handling the vehicle in the three-dimensional space.

In this thesis project, a three-dimensional collision avoidance algorithm has been used and optimized. Studying advantages and disadvantages of various methods, the most appropriate one has been chosen and an UAV equipped with suitable sensors has been designed to detect and avoid obstacles, providing a reactive collision avoidance system. The chosen algorithm is the 3DVFH+, that computes obstacle avoidance manoeuvres in a reactive manner. Hence, the algorithm has been optimised for the specific application and tested in a simulation environment. Parallel to the collision avoidance methods analysis, the specifications of the Guidance, Navigation and Control (GNC) system of the Unmanned Aerial System has been defined and developed as part of the BLUESLEMON project. Subsequently, the development of a simulation environment has allowed to simulate and test the system. Some assumptions have been made on different possible scenarios that the UAV could face in the real world and which could prove the functioning of the simulated collision avoidance system. A multirotor has been designed to perform this type of flight missions. It has to autonomously detect and avoid obstacles that can be or appear along the path previously planned.

Finally, this dissertation attempt to implement, simulate and optimise a collision avoidance method with the aim of validating an implementable and testable system in the real world. Simulations results reveal the functioning of the system, which is able to maintain the predetermined distances from obstacles, carrying out automatic missions. For each obstacle detected, the algorithm recalculates the optimal route to be carried out to accomplish the flight mission. However, future works are suggested in order to improve the accuracy and the sensibility of the method and to perform real world flight missions.

Index

List of Figures

INTRODUCTION

Unmanned Aerial Vehicles (UAVs) represent a high potential for many fields of applications, from surveying to search and rescue missions, from delivery to photography. For all these applications, the interest in this field become higher for those that require automatic missions and autonomous flight operations. In order to consider autonomous flights, one of the most essential requirements is the capability of detect obstacles and avoid them through reliable mechanisms. Nowadays, obstacle avoidance is an incredibly active field of research and development, considering that several aspects of this area are still unknown. For example, obstacle avoidance in three dimensions is less developed than two-dimensional methods, but represents a cardinal point for all types of systems that can move in three-dimensional space. Especially for multirotor UAVs, the industry is increasingly evolving towards the development of complex systems capable of autonomously carrying out different types of missions. For this reason, it is becoming necessary to develop collision avoidance systems that are capable of handling the vehicle in the three-dimensional space.

In the following thesis project, a three-dimensional collision avoidance algorithm will be used and optimized. It has been studied and chosen after an analysis about different collision avoidance methods. Subsequently, the development of a simulation environment has allowed to design and simulate a Guidance, Navigation and Control (GNC) system able to recognize and avoid obstacles. The simulations were carried out by implementing different scenarios, to test the behaviour of the algorithm and sensors applied in alpine environments.

In the subsections below, the background, the concept and the development motivations of this thesis will be described.

Background

In recent decades, automation and robotics are entering the industrial sphere and are changing the way to achieve specific goals. Robots are increasingly being used to perform repetitive and dangerous tasks in many applications, from industry, to home automation, to entertainment. In particular, Unmanned Aerial Systems (UAS) have a vast potential for many autonomous applications. These systems can be used in a variety of applications, such as film making, delivery, surveying, performing search and rescue missions and military's gathering intelligence. UAS are usually composed by a ground control system, one or more cameras, a GPS, software tools and the Unmanned Aerial Vehicle (UAV).

There are a great number of commercially available UAVs, designed for different types of consumers, from hobbyist to photographers and film makers, to professional industrial workers. For all type of consumers, one of the key aspects to consider before the use of this kind of systems is safety, first of all for people, such as pilots, workers or civilians and secondly for the vehicle. One of the major problems that an UAV usually faces during a flight operation is the possibility to collide with other elements in the three-dimensional space. This can cause damage to surrounding area structures, humans or the UAV itself.

To prevent this kind of damages, it is necessary to analyse the concept of obstacle. There are mainly two types of obstacles: fixed objects and targets in motion. As argued in (1), moving obstacles are a difficult challenge to threat, especially for autonomous systems which have to continuously detect the space around them and change the original trajectory planned every time they faced an obstacle. This approach is known as active collision avoidance and it is more difficult to be implemented than a simple collision avoidance method that considers only fixed obstacles and completely known environments.

In the Chapter 1 a list of collision avoidance methods will be analysed to better understand the problem and to assess different cases, implementations and researches that the actual state of the art offers.

Motivation

The following dissertation regards the study of obstacle detection methods and collision avoidance techniques, in particular for Unmanned Aerial Vehicles in three-dimensional environments.

The main problems analysed are the application of methods in the three-dimensional space, avoiding both static and moving obstacles, the outdoor application of processing algorithms and the optimisation of obstacle avoidance parameters. The scenario is an outdoor environment in which is required to grant semi-autonomous system for landslide monitoring. A multirotor has been designed to perform this type of flight missions. The UAV has to autonomously detect and avoid obstacles that can be or appear along the path previously planned.

This thesis will attempt to implement, simulate and optimise a collision avoidance method with the aim of validating an implementable and testable system in the real world.

Goals

The aim of this dissertation is to analyse an understanding of collision avoidance methodologies, in particular for UAVs in three-dimensional environments with six degrees of freedom.

Studying advantages and disadvantages of various methods, it will be chosen the most appropriate one and it will be designed an Unmanned Aerial Vehicle equipped with suitable sensors to detect and avoid obstacles providing a reactive collision avoidance algorithm. The algorithm will be optimised for the specific application environment studied and subsequently it will be tested in a simulation environment. The behaviour of the collision avoidance algorithm will be analysed and evaluated.

Finally, a comparison of the results will be argued, analysing strengths and drawbacks, how the algorithm can be extended and evaluating potential future work and research.

Dissertation Structure

This subsection describes how the dissertation is structured. There are 5 chapters and each of them includes the following themes:

- Chapter 1 - State of Art: Describes methods and researches made before the development of this thesis;
- Chapter 2 - Experimental Setup: introduces the background of the project and describes the UAV's components needed to achieve the project objectives;
- Chapter 3 - Testing Environment and Simulations: explain the Collision Avoidance approach and the implementation of the simulations;
- Chapter 4 - Simulations Results: shows results from different test environments and from different types of simulations;
- Chapter 5 - Conclusions and Future Work: analyses dissertation conclusions and discusses on what and how can be improved the work done.

CHAPTER 1: State of the Art

The study of existing methodologies and techniques used nowadays has made it possible to draw up an accurate analysis of the most suitable collision avoidance methods for this dissertation. Following an accurate analysis, it has been possible to choose the most suitable method. Similar projects already carried out have been also studied and they have been useful to decide the best approach to study and develop this project. In the following chapter the nowadays background and the most important works and methods about collision avoidance are analysed to better understand the method implemented in the dissertation.

1.1 Key concepts of collision avoidance methods

Every type of collision avoidance system has some factors in common with others. Their functionality is to ensure the localization of the obstacle, that can be a static or a moving target, and to avoid the collision reciprocally between the obstacle and the unmanned aerial vehicle. More specifically, the main problems that those types of system have to consider are:

- the modality of sensing the environment. There are different techniques that can provide a local sensing or a global mapping of the space around the vehicle;
- how to extract useful information from the environment specifically about obstacles, such as position, dimensions, speed, bearing angle;
- how to collect and analyse useful data to detect or decide that a collision is imminent;
- the modality of performing the collision avoidance, in particular the realisation of the manoeuvres and the timing of avoidance phase starts and ends.

Following the problems listed above, a collision avoidance technique can be studied divided into two main phases: ‘sensing and detection’, that include the resolution of the first three problems and ‘avoidance mechanisms and manoeuvre approach’ that provide the evaluation of the last main problem. Both of them are essential and in turn can be structured according to different approaches. Finally, each phase has its own features and design factors that can be divided into different sub-phases.

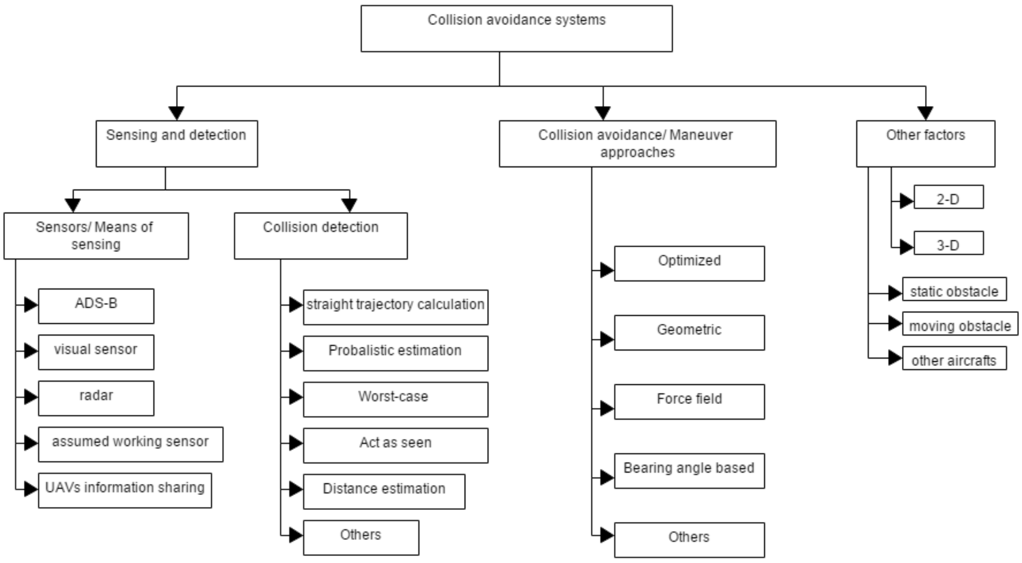


Figure 1: Collision avoidance systems phases and factors (2)

Other common design factors are important to define the background setting of the approach. The most relevant are the type of environment in which the vehicle has to move, such as indoor applications or the external world; directly connected to it, it is necessary to consider types of obstacles and their behaviour, sensing dimensions and especially the type of UAV used.

For this reason, before the description of obstacle avoidance methods analysed in this dissertation, it is necessary to introduce the type of UAV used for the project and to define the motion model necessary to approach the problem and simulate the system.

1.1.1 UAV kinematic model

The type of UAV used for simulations, tests and implementation of the collision avoidance system developed in this thesis is a multirotor, in specific a quadcopter. The system will be analysed in detail in Chapter 2. The individual components, reasons for choosing and comparisons made to design the entire system will be described in those part of the dissertation. However, the following paragraph introduces the concept of system modelling, which allows the development and application of methodologies based on the kinematic and dynamic behaviour of the vehicle.

Multirotor UAVs are platforms based on a set of rotors that permits the motion in a three-dimensional space. Their design can be great different, but a common configuration is to place rotors symmetrically with respect to the horizontal plane. Each rotor generates a thrust along the normal direction to the engine and also a torque. In Figure 2 a multirotor UAV with four rotors and the standard coordinate system is represented. It is also indicated the relation between axis and pitch, roll and yaw rotations.

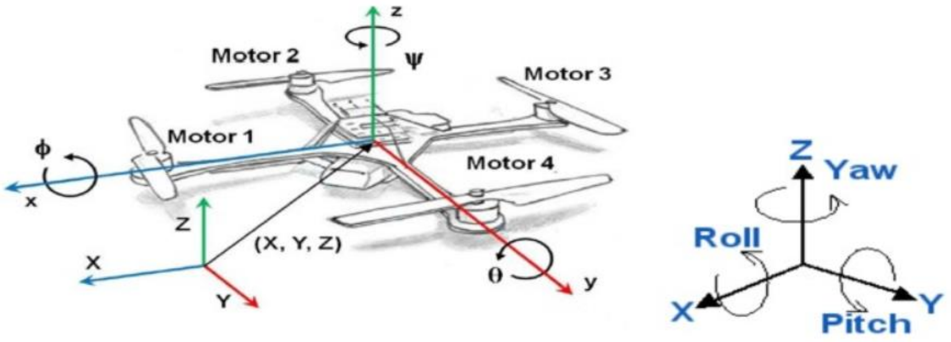


Figure 2: Multirotor UAV: coordinate system (3)

The two pair of contralateral rotors rotate in opposite directions leading the control of the torques. To create a motion model of a multicopter is necessary to consider a variety of factors such as rotor speeds, mass of the entire UAV, drag, air pressure, inertia and the complexity of the system is sometimes reduced by approximations.

First of all, it is necessary to define the major frames of references. Considering $\{A\}$ as a right-handed inertial frame of reference, represented by $\{\vec{a}_1, \vec{a}_2, \vec{a}_3\}$ that are vectors corresponding to $\{\vec{x}, \vec{y}, \vec{z}\}$

coordinate axis. Considering B as the body frame of reference with $\{\vec{b}_1, b_2, \vec{b}_3\}$ as unit vectors. To find the orientation of the body frame, a rotation matrix R_B is introduced:

$\overrightarrow{b_1} = R_B\vec{x};\;\; \overrightarrow{b_2} = R_B\vec{y};\;\; \overrightarrow{b_3} = R_B\vec{z}.$

It is defined by roll, pitch and yaw angles (respectively ϕ, θ, ψ) (4):

$$R_B = \begin{pmatrix} cos\psi\cos\theta - sin\phi sin\psi \sin\theta & -cos\phi sin\psi & cos\psi sin\theta + cos\theta sin\phi sin\psi \\ cos\theta sin\psi + cos\psi sin\phi sin\theta & cos\phi cos\psi & sin\psi sin\theta - cos\psi cos\theta sin\phi \\ -cos\phi sin\theta & sin\phi & cos\phi cos\theta \end{pmatrix}$$

It is also useful to introduce the body-plane fixed frame $\{$C$\}$ because it describes the heading of the platform related to the horizontal plan of $\{$A$\}$; the unit vectors are:

$\overrightarrow{c_1} = R_C\vec{x};\;\; \overrightarrow{c_2} = R_C\vec{y};\; \overrightarrow{c_3} = R_C\vec{z}.$

The rotation matrix R_C depends exclusively on the yaw of the vehicle:

$$R_C = \begin{pmatrix} cos\psi & -sin\psi & 0 \\ sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Following the definition of the frames of reference, is possible to describe the kinematic model chosen to simulate the quadcopter (4) (5):

$$\dot{\zeta} = v$$

$$m\dot{\mathrm{v}} = mg\overrightarrow{a_3} + R_B F$$

$$\dot{R}_B = R_B\omega_x$$

$$I\dot{\omega} = -\omega \times I\omega + \tau$$

ζ is the position of the vehicle $\zeta = [x, y, z]^T$; $v = [\dot{x}, \dot{y}, \dot{z}]^T$ is the velocity, $\omega = \left[\dot{\phi}, \dot{\theta}, \dot{\psi}\right]^T$ is the angular velocity of the multirotor and ω_x is the skew symmetric matrix of ω; I is the inertia matrix and $\tau = [\,\tau_1,\, \tau_2,\, \tau_3]^T$ represent the torque generated by the rotors. Representing the state space matrix of a multicopter (5) (6) (7):

$$X = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} & \phi & \theta & \psi & \dot{\phi} & \dot{\theta} & \dot{\psi} \end{bmatrix}^T$$

in which x, y, z describe the position, $\dot{x}, \dot{y}, \dot{z}$ are linear velocities, ϕ, θ, ψ

are respectively roll, pitch and yaw and $\dot{\phi}, \dot{\theta}, \dot{\psi}$ are angular velocities. The state space representation allows to describe a changing state after a given control input U, as $\dot{X} = f(X, U)$.

An important consequence of the definition of the motion model is that the linear acceleration in the horizontal plane is a function of roll, pitch and yaw angle. Therefore, a variation of pitch changes the acceleration along c_1 and a change in roll leads to the variation of acceleration along c_2. A variation in overall rotor thrust results in an altered vertical acceleration along z axis. According to this, the kinematic dynamics of the multicopter can be expressed by the overall thrust of the rotors and the torque (4). Moreover, is defined the force vector as

$$F = T_{\Sigma}\vec{z} + \Delta$$

in which Δ is used to model aerodynamical phenomena when the vehicle is not in hovering, for example rotor flapping or drag. Defining a motion model for a specific type of multirotor and implementing a system able to translate control commands into rotor speeds is a complicate task and it is considered beyond the aim of this dissertation. Therefore, in the next paragraphs it will be assumed the use of a defined control system allowing for a phenomenological approximation and considering the motion model as a dynamic point model. The main purpose leading to the choice of the model is the ability to evaluate the collision avoidance method and not the accuracy of the kinematic model. The motion model evaluated and used is analysed in (8). It defines the model of the linear velocity of the vehicle as:

$$\dot{\zeta} = v = [\dot{x} \quad \dot{y} \quad \dot{z}]^T$$

and in the same way of defining the state space matrix described before, it has been modelized also the state space matrix of the control input $\dot{X} = f(X, U)$ to define a basic control solution. This simple introduction on how to define a motion model is important to better understand the functioning of the main collision avoidance methodologies analysed in the next paragraphs.

1.2 Obstacle Avoidance Methodologies

In recent years many collision avoidance techniques have been studied and developed. In the following paragraph, only the most significant methods for this study will be reported. For each of them will be described their functioning, peculiarities, strengths and disadvantages. Many techniques are born to be applied in two dimensional environments, but can be extended to three dimensional worlds. In general, collision avoidance systems are based on two main functioning: sensing and detection phase and collision avoidance mechanisms. Based on their characteristics, each phase is divided into different functioning categories.

1.2.1 Geometrical methods

All methods based on a geometric approach are defined as geometrical methods. The base of a geometric approach is the simulation of motion and of trajectories accomplished by the vehicle and at the same time by the obstacles detected. Therefore, they are considered active methods, because it is possible to detect both static and dynamic obstacles. To calculate the trajectories, it is necessary to know parameters of both UAV and obstacles, such as position, heading and velocity.

As demonstrates in (9), to determine collisions it has been calculated trajectories and estimated distances. In specific, the authors calculated the subtraction of own UAV and intruder vehicle’s movement vectors in a bidimensional environment, to determine the shortest distance between them. Subsequently, the UAV’s trajectory is modified by the shortest distance vector calculated, in order to not to exceed the minimum distance established from the obstacle. Depending on how close the obstacle is, so the smaller the minimum vector will be. In this situation the change in trajectory will be the larger.

As can be seen in Figure 3, the proximity to an obstacle causes the alteration of the predetermined trajectory. Comparing case (a) and case (b), it is evident as at relative shorter distances, the quadcopter modifies its mission path more (case b). Those cases are represented considering fixed obstacles, but the same approach is valid also for moving targets.

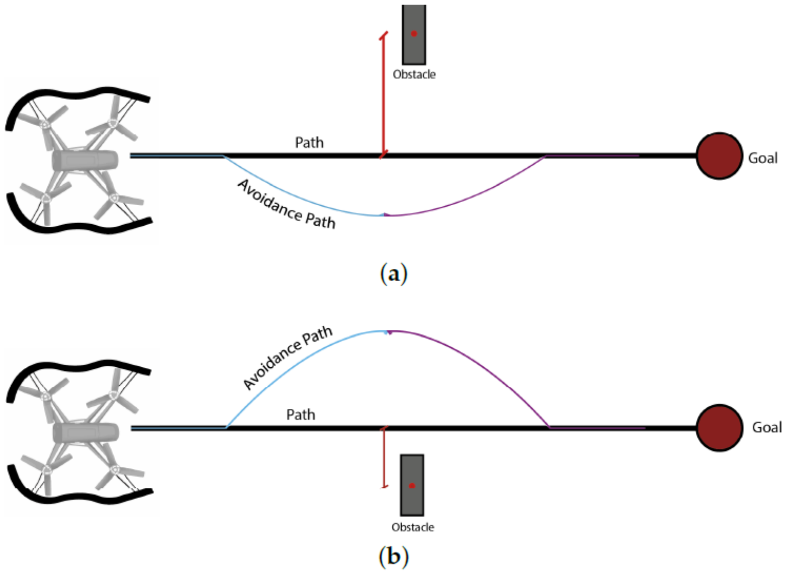


Figure 3: Alteration of trajectories depending on obstacle's distance (10)

An example of active collision avoidance using geometrical approach is studied in (11), considering the worst case of obstacle detection in a bidimensional environment. It is considered an intruder aircraft at its maximum turn rate as the moving obstacle and its future possible trajectory in a short time is calculated. After the definition of a threat region, if the position of the UAV will be inside it in the same range of time, the collision threat is detected and a new manoeuvre is generated. After the threat region is gone, the vehicle path returns to the original one.

This kind of approach is simple and effective, but it has some weaknesses. Analysing moving targets, it is evident that the UAV has to cooperate with the target to stabilize the corrections of its trajectory, so the system has to know some specific parameters of the obstacle's motion. Moreover, the method requires precise calculations, because the trajectories alterations are sensitive to noises in input data. Finally, geometrical approach applications are only implemented using bidimensional worlds, because the application in a three-dimensional environment would greatly complicate the calculations and would affect their accuracy.

1.2.2 Optimized trajectory method

The optimized trajectory method has some common factors with the geometrical approach, especially in the geometrical way of calculating trajectories. The main feature that differentiates it from the previous one is the ability to calculate an optimized trajectory, so as to avoid all the obstacles present along the route, distancing itself as little as possible from the original trajectory. The main types of obstacles optimal for this method are static targets. To collect enough information to best optimize the path, the vehicle has to obtain information by the obstacles such as their velocity, dimensions and accurate position. Those parameters can be detected by different type of sensors. Therefore, the problem is more theoretical than practical, because of the sensor functioning and not more the calculation approach.

On the other hand, the process of optimization requires time and processing power. These two practical problems are usually a limit for all types of UAV, because of the limited processing power and moreover the limited time that a vehicle has to act before colliding to a target. But in general, the results of an optimized trajectory are considered greatly interesting. Therefore, this approach has a great value especially for studying and research.

There are some examples in literature of the optimized trajectory method. An interesting study for this dissertation is reported in (12). The authors developed a model in a three-dimensional space of an UAV using time parameters. The model can predict the global path of the vehicle and consequently the future commands required in a short period. It is based on a cost function, calculated using parameters such as current position and assumed future coordinates; minimizing this function, it is possible to evaluate the best set of commands that constitute the optimal trajectory. Those sets of commands are processed using geometrical trajectory calculation methods and subsequently they are compared; if the best set results in a path difficult to flight, the system evaluate the constraints and chose another set of commands similar to the first choice. Therefore, the cost function is calculated several times during the process of path optimization.

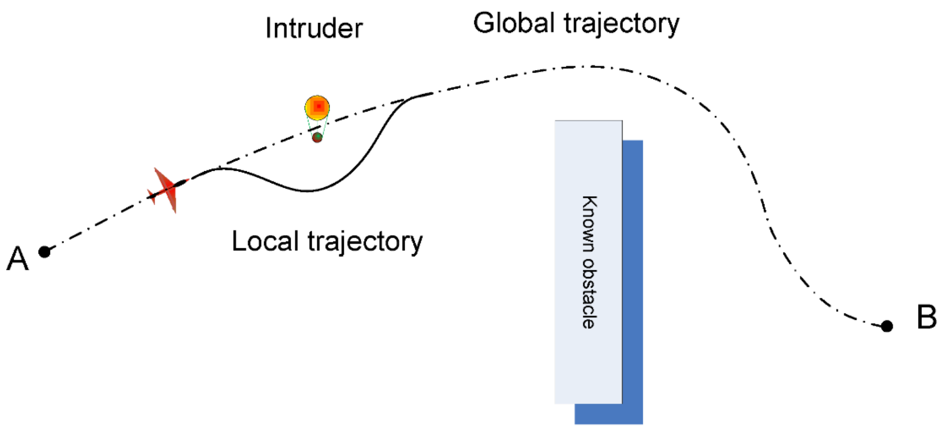


Figure 4: Global path and local trajectory optimization (13)

In Figure 4 a scheme of local trajectory optimization is represented; it is greatly explained in (13), in which the concepts of collision avoidance systems and trajectory optimizations are applied for UAV flying in civil aerospace. The avoidance manoeuvres proposed in the paper are parameterized using a geometric approach. The vehicle used is a fixed wing aircraft and the main purpose of the study is the optimization of an obstacle detection algorithm that works in real-time for local trajectory planning.

Finally, another interesting example can be found in (14), where a bi-dimensional map is used to process obstacles position, size and shape. The results of the evaluation are disposed in a weighted graph and then a collision free path is determined maintaining a good closeness with the original trajectory.

This method, while being an improvement of the geometric approach, maintains some of its disadvantages and increases the complexity of its practical application.

1.2.3 Potential field methods

This kind of methods has been used first for robotics and then it became very widespread for UAVs applications. They are suitable to great introduce the concept of reactive collision avoidance, especially for the simplicity of assumption that have to be made. In general, they express navigation problems as physics concepts; for example, waypoints are treated as attractive forces and obstacles as repulsive ones. Other types of forces are essentially arbitrary, such as potential energy; it is considered higher in points closer to obstacles and lower in points near waypoints. Therefore, using simple electrostatic equations is possible to define safe trajectories; the one that has the lowest flux density become the new path for the UAV.

There is a common standard way to calculate forces at a defined position identified by the x variable (15):

$$F_{(x)} = F_{att(x)} + F_{rep(x)}$$

$$F_{att(x)} = -k_{att}(x - x_{des})$$

$$F_{rep(x)} = \begin{cases} k_{rep}\left(\frac{1}{\rho_{(x)}} - \frac{1}{\rho_0}\right)\frac{1}{{\rho_{(x)}}^2}\frac{x - x_{obs}}{\rho_{(x)}} & if\ \rho_{(x)} \leq \rho_0 \\ 0 & otherwise \end{cases}$$

In which $\rho_{(x)}$ represent the distance to the obstacle, ρ_0 represent the distance of influence and k_{att} and k_{rep} define the overall strength of the forces. This approach is used especially for distributed and local collision avoidance in uncrowded environments, where state information is known for all vehicle and obstacles (16). However, it is not complicate to adapt the method to work reactively. For example, the Virtual Force Field algorithm (VFF), first implemented by using a mobile differential drive robot with sonar to detect obstacles (17), works by defining a histogram grid of values that describes the belief of an obstacle that is occupying a point in space. Analysing both histogram and sonar data it is possible to estimate the relative position of obstacles. The distance of a target point is considered proportional to an attractive force, while repelling forces for each cell in the grid are defined by weighing the force by the inverse cell distance and certainty value. Calculating the sum of all forces and setting the steering rate of the differential drive motion model proportional to the angle of the resulting force vector and to the robot heading, it can be seen better performances compared to previous methods.

However, there are some problems with potential fields, in particular in practical systems, for example saddle points and local minima that may occur when generating a dynamic potential field. This can cause problems such as aircraft loss of control or collision threat. Another difficulty that can occur in a practical application is that the dynamic limitations of the aircraft have to be considered. If this does not happen, the vehicle will not be able to fly the generated path. Moreover, considering the high importance of the availability of state information for this type of methods, any deficiency in this information may generate wrong field formation. It can cause aggressive control commands that may affect aircraft performances (18).

Finally, as argued in (19), there are other criticism of using a mathematical model to describe how the vehicle dynamics is affected by changes; for example, the method collapses all forces into one singular resulting force. It causes the loss of information about obstacles location and consequently, even if it would be physically possible to traverse some difficult passages, it couldn’t be possible in the real space. In the same way, oscillations can occur as a result of moving near obstacles or through narrow corridors. To resolve some of this difficulties, other types of methods have been studied and developed by different authors; one of the most important is the Vector Field Histogram.

1.2.4 Vector Field Histogram (VFH)

The Vector Field Histogram method is a reactive real-time obstacle avoidance method. It was first introduced by Borestein and Korem (20) using a mobile robot. They explain how the robot can detect unknown obstacles and avoid them while it is moving and steering around the targets. The VFH method makes use of a two-dimensional cartesian histogram grid considering it as the world model. The model is frequently updated by data received by sensors integrated on the system thanks to a two-stage data reduction process. This process manages to compute desired control commands first by reducing a constant size subset of the 2D histogram grid around the position of the robot in a one-dimensional polar histogram and then selecting the most suitable sector of the polar histogram, defining a new direction of motion. The method can be divided in three main steps:

1. Creation of a bidimensional cartesian histogram grid representing the world around the vehicle with obstacles;
2. Selection of an active window around the robot position of the 2D histogram grid and turn into a 1D polar histogram;

3. Calculation of the steering angle and velocity commands from the 1D polar histogram resulting from an optimisation process.

To better explain the three phases, it will be subsequently reported the main concepts regarding each previous point, using some images taken by (21).

The creation of the 2D cartesian histogram is made considering all coordinates taken by each range sensor measurement and putting them into a grid map. This process is independent from the type of sensor used for obstacle detection, such as laser range, ultrasound or camera sensors. As can be seen in Figure 5, for each range reading, the cell lying on the central axis and corresponding on a fixed distance *d* is incremented. It causes the incrementation of the Certainty Value of that cell. This part of the first phase is presented and precisely explained in (22). The Certainty Value of the cells continuously updates during the vehicle motion.

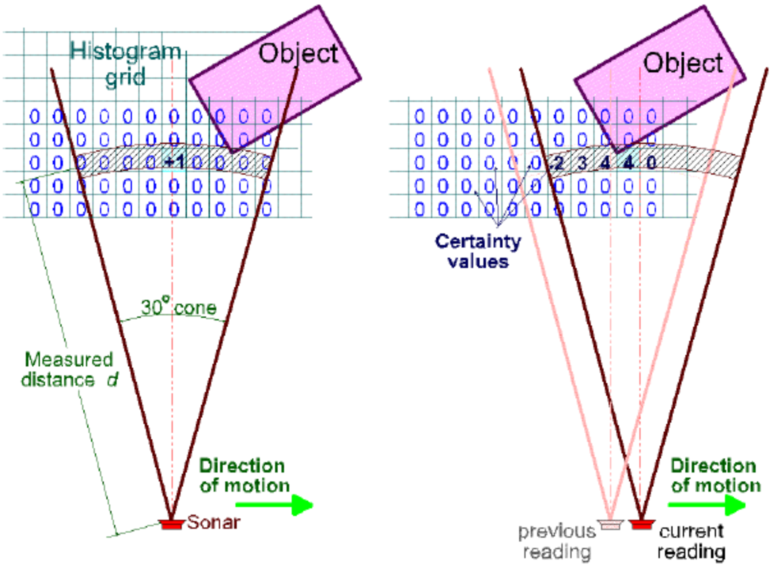


Figure 5: Creation of the bidimensional cartesian histogram grid (21)

The second phase allow to translate the bidimensional grid map in a one-dimensional structure. To better threat information about an obstacle rather than process the whole grid map, the active window concept is introduced. To restrict the 2D grid map, it is considered a constant dimensions area centred on the vehicle position; consequently, it moves with the vehicle and it represent a local area around it.

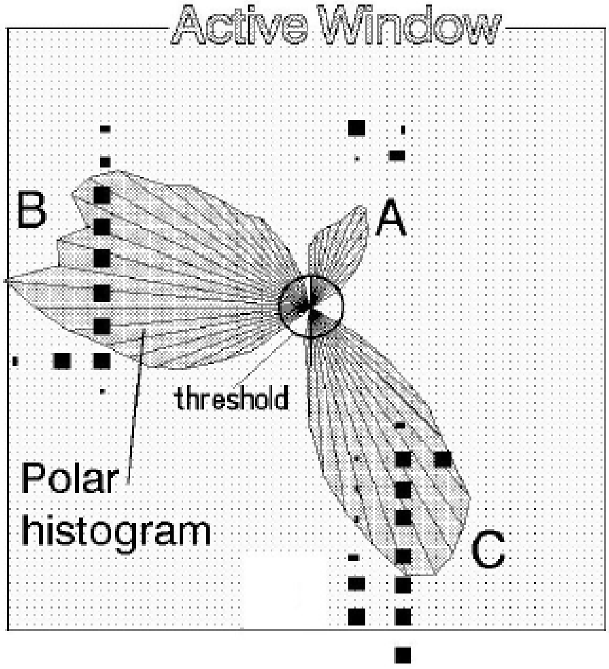


Figure 6: Active window and polar histograms (21)

The new grid is mapped in a one-dimensional structure called polar histogram. In Figure 6 it can be seen the projection of the one-dimensional polar histogram in the active window expressed in polar form. It represents a situation with three obstacles around the vehicle; the histogram is overlapped with the referred obstacles.

Finally, during the third phase, the required steering direction for the vehicle is evaluated. It is calculated by a given sector of the one-dimensional histogram in which the vehicle velocity is adapted according to the obstacle polar density. To choose the best sector of the active window to pass through it is necessary to analyse the polar histogram. Usually, it is composed by peaks, that are sectors with high polar density, and valleys that represent sectors with low polar density. To evaluate the best path to pass around the obstacle, is necessary to consider a variety of factors, such as if the valley is large enough to permit the motion of the vehicle. If consecutive sectors are all defined as candidate valleys and considering other factors, like the alignment of the vehicle to the target, the difference between the current and the desired direction and the difference between the previously selected direction and the new one, the new path is established.

The VFH method overcomes some of potential field methods limitations. For example, the influence of bad sensor information is minimized, the absence of attractive or repulsive forces eliminates the problem of being trapped in local minima. But it has also some weaknesses; for this reason, some extensions of the method have been studied.

The VFH+ method is introduced in (23); the main characteristic of the extension of the method is the consideration of vehicle's dimensions. Indeed, due to not considering the dimensions of the system, the obstacle cells were considered enlarged by the radius of the vehicle. In addition, it has been developed and used a binary polar histogram instead of the polar density histogram, because of its tendency of rapidly changing. The binary polar histogram is a representation of the polar density histogram but with only two values, 0 or 1, depending on a known threshold; if the value of the polar density histogram is over the threshold the result will be 1, otherwise the result will be 0. Moreover, the motion model of the vehicle has been considered. It allowed to remove possible path candidates that didn't respect parameters such as the vehicle minimum steering angle. These two factors lead to the definition of a new histogram, called as a masked histogram, in which valleys are constituted by all cells with a value of 0 and represent possible trajectories. This approach permits to avoid ambiguous steering commands.

The VFH* method is presented in (24); the additional extension regarding the two previous approaches consist in taking into account not only the current state of directions but also future configurations of the vehicle. The current directions that leads to the lowest-cost path is selected for the resulting final command. This approach permits to avoid local minima conditions.

In conclusion, the main disadvantages of the VFH method are two: the first is that it doesn't consider the motion model of the vehicle and its dimensions; this implies some difficulties in implementing the method in the reality, because of the possibility of results physically correct but impossible to be followed by the vehicle. The second disadvantage is that the method born as a bi-dimensional approach to the problem of collision avoidance. Indeed, in all papers, the vehicles implied are robots or terrain vehicles. It is quite complicated to extend this method to a three-dimensional world.

1.2.5 Nearness Diagram (ND)

Maintaining some aspects in common with the VFH method, the Nearness Diagram approach has been presented in (25) and then better analysed and expanded in (26). The world around the vehicle is represented by a diagram divided in sectors as the previous approach. The main differences are the avoidance strategy and the measured metrics. During each time step of measurements, data are assumed with accurate directional distances. Subsequently, for each sector, a nearness metric is calculated and the diagram is generated divided by gaps. They are considered discontinuities that delineate free walking areas, known as valleys in the previous method. In Figure 7 it is represented an example of free walking area between obstacles.

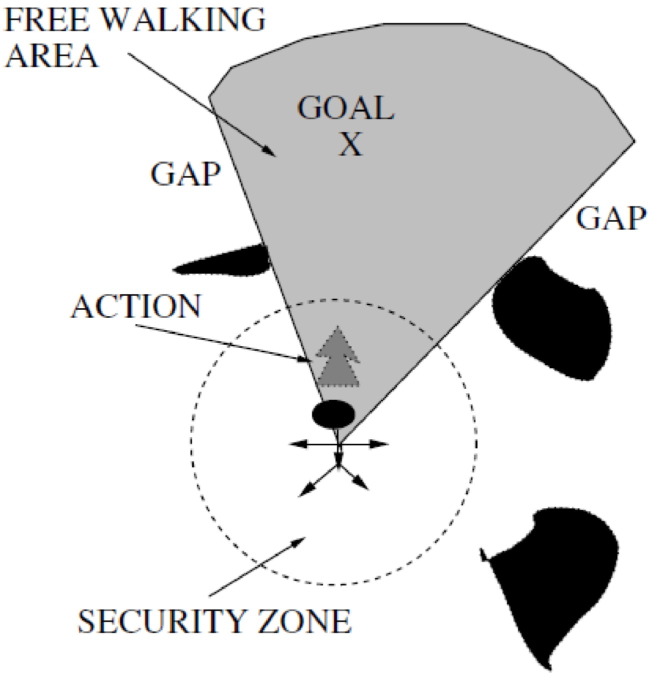


Figure 7: Example of a ND method action result (26)

The avoidance approach is defined by five main factors, in particular:

- if the vehicle is too close to the obstacle on one or both sizes of the free walking area;
- if the zone of the free walking area is wide or narrow;
- if the following waypoint of the path is positioned in a free walking area or not.

This method has been developed specifically for motion models designed in a two-dimensional environment and to obtain desired measurements it is necessary to adopt accurate omnidirectional range finders, for example a 2D laser scanner. For this reason, the disadvantage of the ND method is that it has not yet been studied how the method can be applied in a three-dimensional world.

1.2.6 Dynamic Window Approaches

Some different obstacle avoidance approaches consider the motion model of the vehicle directly, introducing the reactive collision avoidance concept. Two of the most used approaches are the Curvature-velocity method (27) and the Steer Angle Field method (28). Their main feature is to calculate the avoidance approach by using differential drive trajectories. As argued in (29) and in (30), the dynamic window approach allow to calculate steering commands that manage the vehicle motion around an obstacle while considering the ability of the vehicle to affect its velocity. The velocity is defined by the linear and the angular velocities (v, w).

The first phase of the method is defined by the creation of a set of velocities not resulting in collisions, using the following expression:

$$V_a = \left\{(v,w) | v \leq \sqrt{2 \cdot dist(v,w) \cdot \dot{v}_b} \land w \leq \sqrt{2 \cdot dist(v,w) \cdot \dot{w}_b}\right\}$$

In which $\dot{v}_b$, $\dot{w}_b$ represent the brake accelerations and $dist(v, w)$ indicates the smallest distance to an obstacle that intersect the trajectory generated by selecting the velocity (v, w). Subsequently, a set of reachable velocities is calculated as follows:

$$V_d = \{(v,w) | v \in [v_a - \dot{v}t, v_a + \dot{v}t] \land w \in [w_a - \dot{w}t, w_a + \dot{w}t]\}$$

In which (v_a, w_a) is the actual velocity. The set of velocity that are not in collision is calculate as follow:

$$V_r = V_s \cap V_a \cap V_d$$

Where V_s is the set of all possible velocities. Defined a direction of travel, a heuristic cost function is used in order to find the desired velocity:

$$G(v,w) = \propto heading(v,w) \cdot \beta dist(v,w) \cdot \gamma velocity(v,w)$$

In which $heading(v, w)$ indicates the alignment between the vehicle heading and the direction of motions, and the $velocity(v, w)$ represent v projection.

This principle allows to implement reactive collision avoidance on a drive motion model vehicle. The resulting trajectories are approximated as circular arcs. The performances of this approach are greatly comparable with previous methods.

1.3 Related Work in literature

Much work has already been done on the study and analysis of collision avoidance methods. Some authors presented nice surveys of collision avoidance approaches (31), arguing comparisons and describing their key characteristics. Others summarized recent collision avoidance techniques, creating their own simulations and comparing between them results (14). More specifically for this dissertation, in recent literature it can be found also some works regarding obstacle avoidance for UAVs, in particular for multicopters. Most of the studies found in literature deal with the subject in outdoor environments, where there are wider environments for various type of manoeuvres. The following paragraphs describe some approaches to the methods described so far and will report on UAV collision avoidance applications available in the literature.

1.3.1 Potential fields approach applications

Potential field methods are greatly diffused in collision avoidance applications on multicopter UAVs. For example, in (32) it has been implemented an obstacle avoidance system based on two wide-angle stereo cameras and a laser scanner. The information acquired by the sensors allow to create a discrete three-dimensional occupancy grid in which the UAV is discretized into cells. Each cell is subject to two types of force. Waypoints generated by the calculation of the ideal trajectory are a source of attractive forces, while obstacles close to the UAV are a source of repulsive forces. The latter shall be calculated by a weighted average of the forces generated by each neighbouring obstacle. The resulting will be the force that will go to oppose the forces attracting to waypoints in order to generate a new path plan.

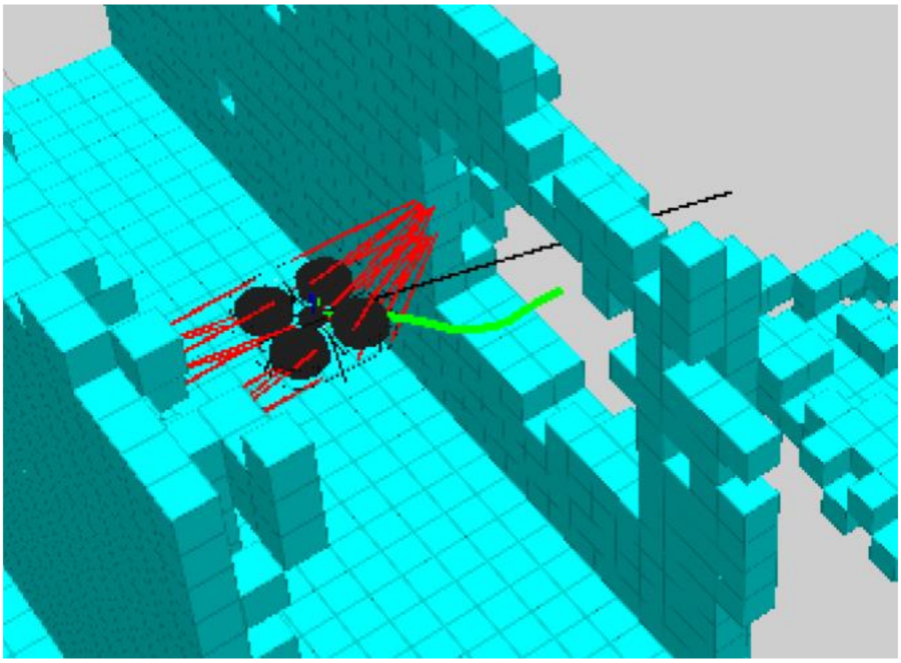


Figure 8: UAV simulation model with forces and path planning (32)

The authors developed a prediction estimate of the future trajectory, visible in Figure 8 as the green line; when the vehicle is near an obstacle, the velocity of motion is automatically lowered. The red lines represent repulsive forces generated by obstacles to the model cells. They also implemented a learn motion model based on considering the flight dynamics of the UAV as a time-discrete linear dynamic system in order to precisely predict the trajectory. It is finally been optimized using motion capture data.

Another experimental approach is described in (33), where a quadcopter has been equipped with a payload of low-cost sensors, in particular ultrasound and infrared sensors. Combining distance data from these two types of sensor it is possible to better estimates distances, also using information from optical flow and IMU data. The perimeter of the UAV is divided by sectors, in particular as front, back, left and right. Sensors are distributed along the sectors and resulting data from the evaluation of all sensors are given to a state machine that determines eventual corrections needed in pitch and roll commands to the vehicle. The avoidance phase is characterised only by horizontal translations. The system has been tested in different environment to evaluate the quality performances of the method. The results are acceptable, but it turns out that areas full of high obstacles, like corridors, cause the formation of an oscillatory motion which generates difficulties in the vehicle control system. The authors evaluated first the motion model of the UAV and then the collision avoidance implementation using the Gazebo simulator, that is a really diffused tool used to simulate and evaluate systems in motion.

Finally, considering the implementation described in (34) as interesting for this thesis, especially because of the use of another type of sensor, it is reported as final example of potential field methods implementations. The system tested is a helicopter equipped with a 29 kg payload; the obstacle detection sensor is a Laser Radar, known as LIDAR with which the UAV can avoid obstacle at the velocity of 10 m/s. The system has been tested in an outdoor environment in which obstacles are identified by spherical coordinate system after an estimation of the probability of being in a precise position. Thanks to a variant of the potential field approach, the collision avoidance algorithm has been developed to manage three-dimensional detection. The main disadvantage of this application is the limited field of view of the LIDAR; that induces a reduction in degrees of freedom of the vehicle.

1.3.2 Optical flow methods

Optical flow techniques are very common in the UAVs applications because optical flow provides a series of very useful and functional information both regarding the analysis of the surrounding environment and the vehicle motion control system. The main concept is that relative differences in images, identified by velocity vector of moving pixels, generated by a moving camera, result in a series of rotations and translations of the camera. From this, geometric estimation of obstacles can be made. Another important fact is that higher pixel velocity is directly proportional to the closeness to obstacles. In (35), (36), (37) some techniques to estimate these distances are presented and discussions about optical flow applications for UAVs are argued.

The principle of steering away from either the right or the left side by changing the yaw of the vehicle, depending on the side that has the highest optical flow, is quite diffused in different applications. The concept is that a high optical flow corresponds to a nearby obstacle. Describing one method as an example, in (38) has been tested on a simulated UAV the optical flow method, using an autopilot software in a two-dimensional environment. The simulated vehicle resulted able to navigate around the two-dimensional space, but some collisions there have been counted. Subsequently, authors expanded the first implementation introducing a way of limiting the forward translation speed. It has been possible by calculating a time to contact approximation and introducing a method of compensating optical flow introduced by rotation (39). These implementations allow to avoid collisions in the same testing environments.

This type of applications evidences the main disadvantage of optical flow techniques. In order to get precise values from the sensors in a fixed direction, the vehicle has to move as perpendicular as possible to this direction. This causes limitations in motion in many ways depending on how many sensors are mounted on the vehicle and how they are fixed.

1.3.3 SLAM, ORCA and other methods

Some other methods have been developed for different applications of the collision avoidance field. For example, the FastSLAM method is presented in (40) for safe teleoperation applications. The method is based on Simultaneous Localization And Mapping (SLAM) concept (41) and it is implemented by using sonars as unique sensors. The approach is to approximate the layout of the space around the UAV by processing data received from the sensors. This layout is then turned into a two-dimensional map in which the vehicle model is divided into cells. The avoidance is made by restricting the velocity based on the collisions time to occupied cells. Tests proved that the vehicle is able to avoid obstacles.

The Optical Reciprocal Collision Avoidance (ORCA) approach is considered an innovative methodology greatly diffused especially in swarms' applications (42) (43) (44). It is based on algorithms working on reciprocal velocity between the obstacle and the UAV. The result of the processing data is a set of velocities that are collision-free and a set of optimal paths for vehicles in motion. This method is a great example of reactive collision avoidance approach, because each system in motion change its best path depending on its position related to other target locations.

Safety-ball and Mass Point Models methods are presented and compared in (45). Both of them consider the vehicle in motion towards the destination while the obstacles around it are avoided by creating a set of intermediate points fixed near the obstacles. For the first method, points are fixed on a radius considered safe around the UAV and all future waypoints are fixed to this minimum distance from the vehicle. Instead the Mass Point Model assumes that obstacles are mass points and it evaluates the risk of collision with these points. Subsequently a sphere is put around the obstacle and the vehicle avoid the collision with entire circle. After a comparison between the two approaches, the authors argued that the Safety-ball method has higher performances.

The obstacle avoidance problem is also been reduced to a control problem (46). Assuming that the UAV is in motion towards a waypoint, if an obstacle is detected, an intermediate time-optimal safe point is fixed using the optimal control approach. The algorithm has been tested in a simulation environment in which the vehicle faced obstacle individually in order to reach a new waypoint along the optimal path.

CHAPTER 2: Experimental Setup

In the following chapter, the specifications of the Guidance, Navigation and Control (GNC) system of the Unmanned Aerial System will be defined and developed as part of the BLUESLEMON project. In particular, hardware and software components of the avionic system will be analysed, including the software for mission planning, the ground segment with its human-machine interface and the system for communication between the on-board technologies and the UAV.

Starting from the mission profile and the respective operational requirements, it was necessary to evaluate different design solutions of both technical and operational natures, in order to identify the best option to ensure the correct and complete execution of the activities planned.

The BLUESLEMON project is a research project aimed at developing an Unmanned Aerial System to monitor landslides through automatic missions. Some different technologies are employed, for example the use of Bluetooth Beacons fixed to the ground for sending useful data to the UAV and for the monitoring of possible displacement of the portion of land on which the Beacon is fixed. Appendix 1 describes the project in detail, summarises its objectives and defines its main features. Moreover, it introduces a brief presentation of MAVTech srl (47), the company with which it has been possible to realize this thesis.

Therefore, the project has also established the requirements that the vehicle will have to comply. Starting from these guidelines, an in-depth analysis of the components that will compose the GNC system has been carried out, following general considerations on the nature of the aircraft and its peculiarities.

2.1 Flight Segment

The choice of the type of UAV used to carry out the requested missions was made by analysing the different types of vehicles and their main characteristics. Focusing on structural differences, two main types of aircraft has been analysed: fixed-wing and multi-rotor UAVs. Fixed-wing vehicles have some aerodynamic characteristics that allow them to have some benefits over other types of UAVs. The most important of them is the possibility to cover long distances, optimizing consumption and maintaining high flight performances.

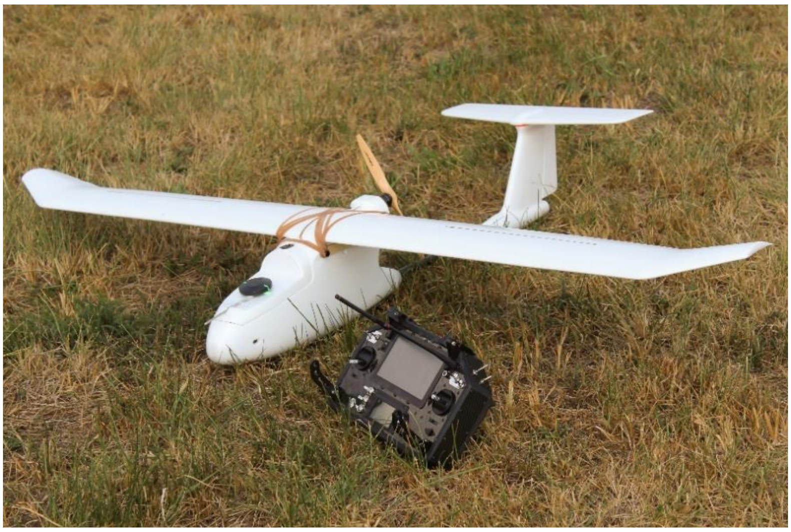


Figure 9: Fixed-wing UAV: Agri 1900 developed by MAVTech (47)

They can also reach higher speeds in relation to consumption and glide for a long time maintaining low gradients of descent. These features allow much more endurance, that makes this type of system especially suitable for long missions. Take-off and landing mode shall be only horizontal for fixed-wing vehicles. Finally, they cannot achieve hovering like multi-rotor systems.

On the other hand, multi-rotor UAVs have some different features: they are not usually used for long range missions because of their high consumption in relation to the distance accomplished, causing less endurance than the previous type. The main features that make this system the most suitable for application in BLUESLEMON project and in this thesis are the hovering capability and especially the vertical take-off and landing mode. This makes it possible to carry out fully automatic and autonomous missions.

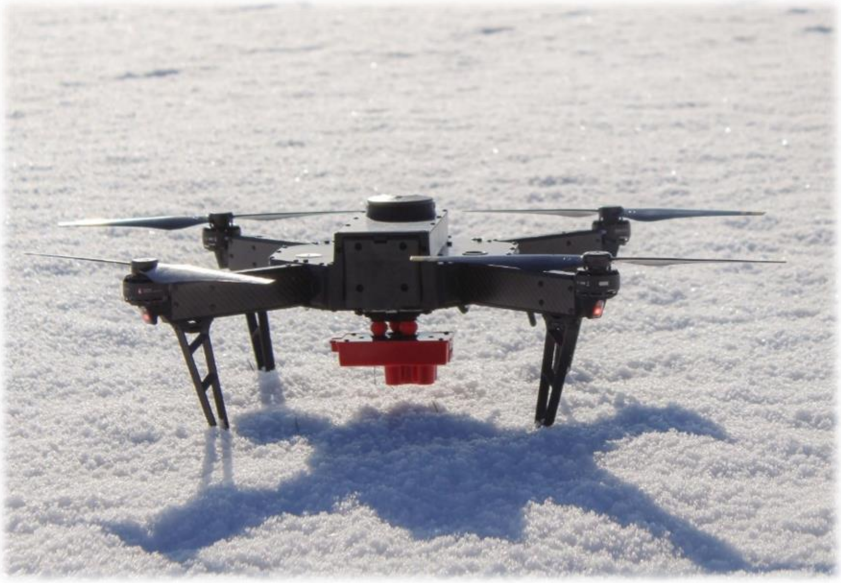


Figure 10: Multi-rotor UAV: Q4E drone developed by MAVTech (47)

Moreover, a multi-copter is much manoeuvrable and is able to work in various type of environment, for example in a space with different types of obstacles. Specifically, it has been chosen to use a quadcopter as evaluated as the most suitable for this type of missions. It is light and manoeuvrable, it does not need to load heavy payloads and defines an excellent compromise between consumption, weight, costs and performances.

In the following paragraphs only the characteristics of the GNC system of the quadcopter designed for the project will be described, as of greatest interest for the activities related to the following dissertation.

2.1.1 Hardware Components

Initially, it has been necessary to define the hardware components used to carry out all the activities envisaged by the project, starting with the standard actions for controlling attitude and stability of the vehicle. The main functionalities of each component are described and where significant the salient points of the comparative analysis between the different technologies on the market, capable of performing a certain function, are reported.

2.1.1.1 Autopilot

For flying over areas in automatic mode, each aerial platform needs an autopilot able to manage and pilot the UAV autonomously during the mission and to command any payloads on board. In the specific case of the BLUESLEMON project, automatic missions will be programmed to fly over landslide fronts in order to detect the positioning of beacons and their relative movement.

The Pixhawk 4 (48) autopilot was developed in collaboration between Holybro and PX4. It is equipped with more computing power and a double RAM than the previous versions; it has additional ports for better integration and expansion, new sensors and an integrated vibration damping mechanism. For these reasons, despite the absence of previous experience of the company regarding its application, it was chosen to proceed with the use of this autopilot.

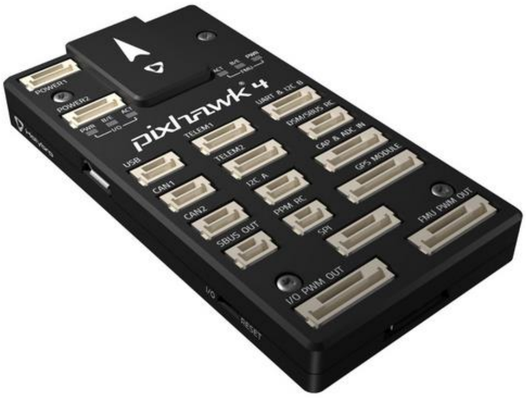


Figure 11: Holybro Pixhawk 4 autopilot (48)

2.1.1.2 Power Management Board

To ensure the complete integration of the autopilot it is necessary to analyse and board the power modules. Generally, the main necessary modules are the BEC and the power module. The Battery Eliminating Circuit (BEC) is a voltage regulator designed to provide a constant voltage of 5 V. This electronic component allows the power of the radio control receiver (RC) and all other utilities that must be powered at 5V, without the need to load an additional battery. Another important function of the BEC is to ensure the power supply to the utilities, preventing the engine from absorbing an amount of energy that makes it impossible to power the RC and payloads. This avoids the interruption of the signal and the consequent loss of control of the UAV. The autopilot is powered by an analog power module that provides a stable power supply and supports measurement of battery voltage and power consumption.

The two components described above allow the complete integration of autopilots similar to the chosen one and constitute the configuration used in previous experiences. For the integration of the Pixhawk 4, Holybro, has developed an integrated board that allows both the power of the autopilot, both the connection and the power of the ESC, also sending information to the autopilot on parameters such as battery voltage and the current supplied to the flight controller and engines. This is called Power Management Board and it is showed in Figure 12.

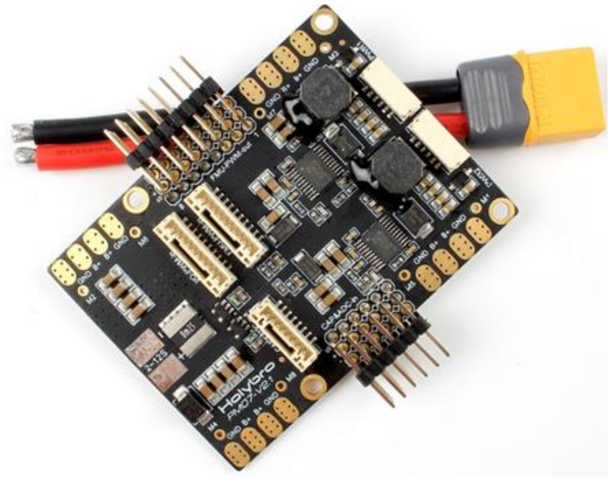


Figure 12: Power Management Board for Pixhawk 4 (49)

2.1.1.3 GPS RTK

The choice of the GPS tracking device is very important for maintaining adequate positioning performances during flight missions. Following a preliminary analysis, in which the conventional Here2 and the RTK Here+ models (50) were compared, the hypothesis of using a conventional GPS was discarded due to errors in positioning too much high (the Here2 model analysed has a positioning accuracy of 2.5 m). Considering the need to detect Beacon position changes for distances of the order of one centimetre, it was necessary to consider GPS that supported RTK technology. In this paragraph the main features of the model of GPS RTK chosen will be described, following an accurate analysis between different models on the market. For more details, the complete analysis is given in Appendix B.

The Drotek Sirius RTK GNSS rover (F9P) and base system (51) was developed by Drotek Electronics, which provides the complete kit of both rover module and RTK base (Figure 13). The Sirius rover module, based on ZED-F9P U-blox technology, offers a multi-band GNSS for high performance and reliability in various industrial applications. The F9P module provides a positioning accuracy of about 1 cm, a convergence time of less than 10 seconds and a navigation update speed of up to 20 Hz. Following an in-depth research and a comparison with the performance of the RTK Here+ GPS, previously tested by the company, it was decided to use this model for the project, as it reflects the best compromise between performance required, market price and technological offer.

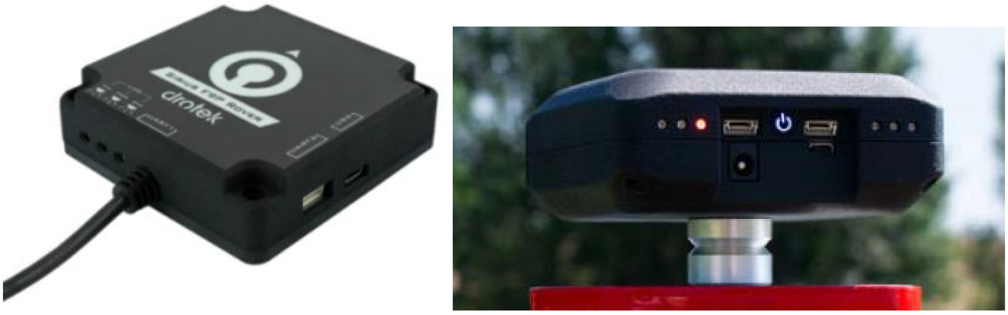


Figure 13: Drotek RTK GPS Sirius model, rover (left) and base (right) (51)

2.1.1.3 Telemetry

A bidirectional telemetry ensures communication between APR and computers, manages flight parameters during mission and monitors the status of the APR during flight. The telemetry kit works at frequencies of 433MHz and has a maximum transmission power of 100 mW. The choice is mainly due to regulatory limits of transmission, as the Italian legislation requires the use of telemetry devices with power up to a maximum of 100 mW. There are devices on the market that reach up to 500 mW of power, allowing to maintain communication up to a radius of two kilometres away. However, keeping the emission powers in accordance, an UAV can fly at a maximum of one kilometre distance from the pilot, in addition to the fact that for ENAC (Ente Nazionale per l'Aviazione Civile, the Italian Aviation Authority) a UAV can fly at a maximum distance from the pilot of 500 meters (VLOS, Visual Line of Sight).

Considering these problems and using a radio control-receiver system technologically advanced, it was possible to avoid the use of a telemetry kit, thanks to the use of the model Herelink radio controller described in the next paragraphs. This choice has brought benefits in terms of space and possible interference, as it was possible to remove the telemetry module on board. The telemetry kit has been replaced by a single receiver that communicates only with the pilot's remote controller, but provide also the telemetry feedback.

Thus, the entire system communicates at frequencies of 2.4 Ghz and the performances are much higher considering the distance of communication, which is of the order of tens kilometers.

2.1.1.4 Radio controller receiver

In order to be able to control the UAV remotely, it is necessary, during the design phases, to provide the integration onboard of a compatible receiver associated with the radio command. The model defined by the Hex developers as air unit, specific for the ground unit (Herelink radio controller, chosen and described in the next paragraphs), allows the UAV to communicate with the ground also from high distance ranges, up to twenty kilometres away.

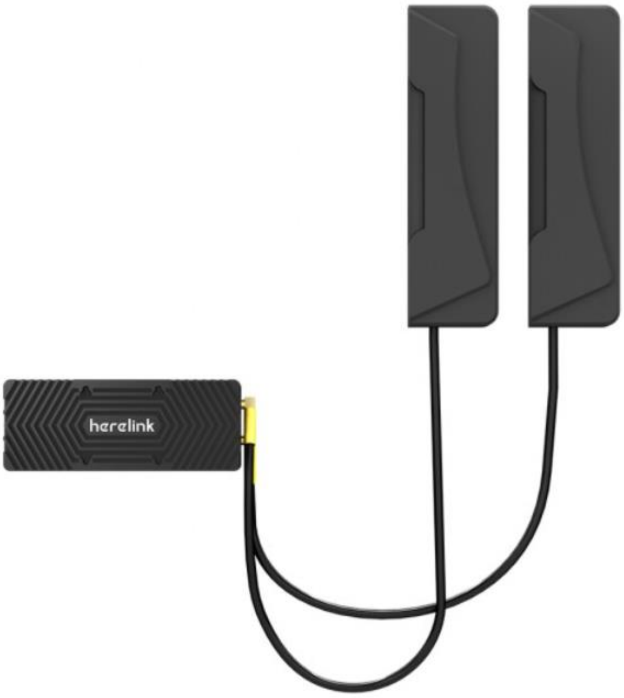


Figure 14: Herelink reciver – air unit (52)

2.1.1.5 Altimeter

The altimeter is an essential component for the Guidance, Navigation and Control system. It allows to detect the flight altitude of the UAV during the mission. For the project, the Lightware laser altimeter model SF11/C has been chosen. Thanks to its characteristics of lightness and ease of integration with the autopilot, it is considered ideal to be integrated in small size vehicles. Moreover, it is specifically designed for multi-rotors and fixed wing aircrafts.

Figure 15: Laser altimeter Lightware SF11/C model (53)

The model has two main hardware interfaces: a Micro-USB port and a multi-pin port through which it is possible to feed and connect the sensor to the autopilot, via digital (serial or I2C) or analog outputs (12-bit). In this case it has been chosen to establish communication via I2C connection. The connection scheme is represented in Figure 16.

The sensor is sold fully calibrated and once connected and powered, it sends directly to the autopilot the altitude data detected, taking into account an offset due to the difference in altitude between the positioning of the sensor and the base of the UAV landing gear. It will be possible to view the data by configuring the parameters of the type "rangefinder" and visualizing the data indicating the status of the aircraft, in particular in this case the value indicated as "sonar range", on the management software of the Ground Control Station.

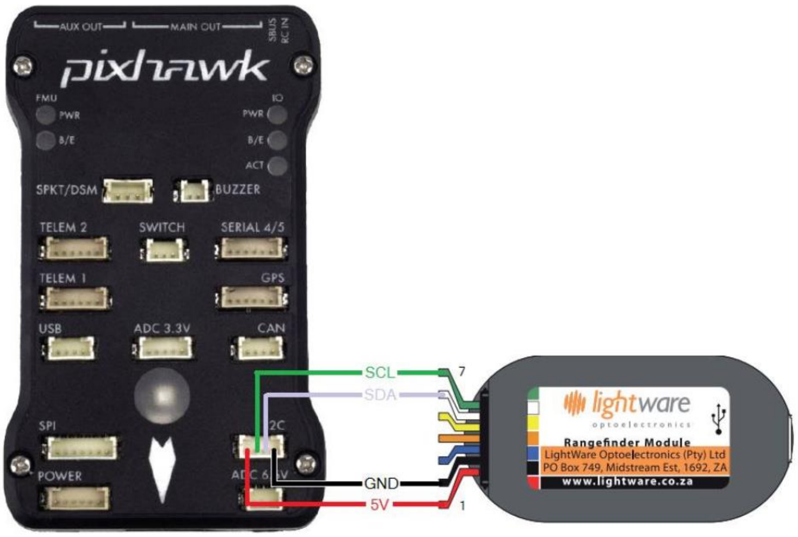


Figure 16: I2C interface connections between altimeter and autopilot

2.1.1.6 Navigation camera

The navigation camera is a very useful instrument, if not indispensable in potentially critical flight conditions. For example, during non-automatic mission phases or in case the automatic mission should be interrupted at high distances from the pilot, having an additional view given by the navigation camera is advantageous for the pilot. Moreover, if it is necessary to fly very close to a specific target, the navigation camera is very useful to correct the flight trajectory, ensuring high levels of precision. The navigation with external camera must always be done keeping the drone in visual line of sight; in fact, according to ENAC regulation, it is not possible to pilot the RPAS (Remotely Piloted Aircraft System) in First Person View (FPV). The camera model chosen for the BLUESLEMON project is the Firefly Split, from the Hawkeye developer.

Figure 17: Hawkeye navigation camera Firefly Split model (54)

In addition to high video performance, the main feature for which this model has been chosen is the presence of an HDMI output compatible with the Herelink radio control receiver. In this way it is possible to send video images directly to the remote-control screen, through the receiving module, without the need of any additional video transmission modules on board the aircraft. This avoids problems of encumbrance and interference between antennas and receivers, taking advantage of the module and the high performance of the Herelink system.

Figure 18 shows a section of the CAD model of the quadcopter developed for the project. To integrate the camera, a case has been designed to rotate the camera in different angles to be able to fix it in an appropriate position for different types of mission.

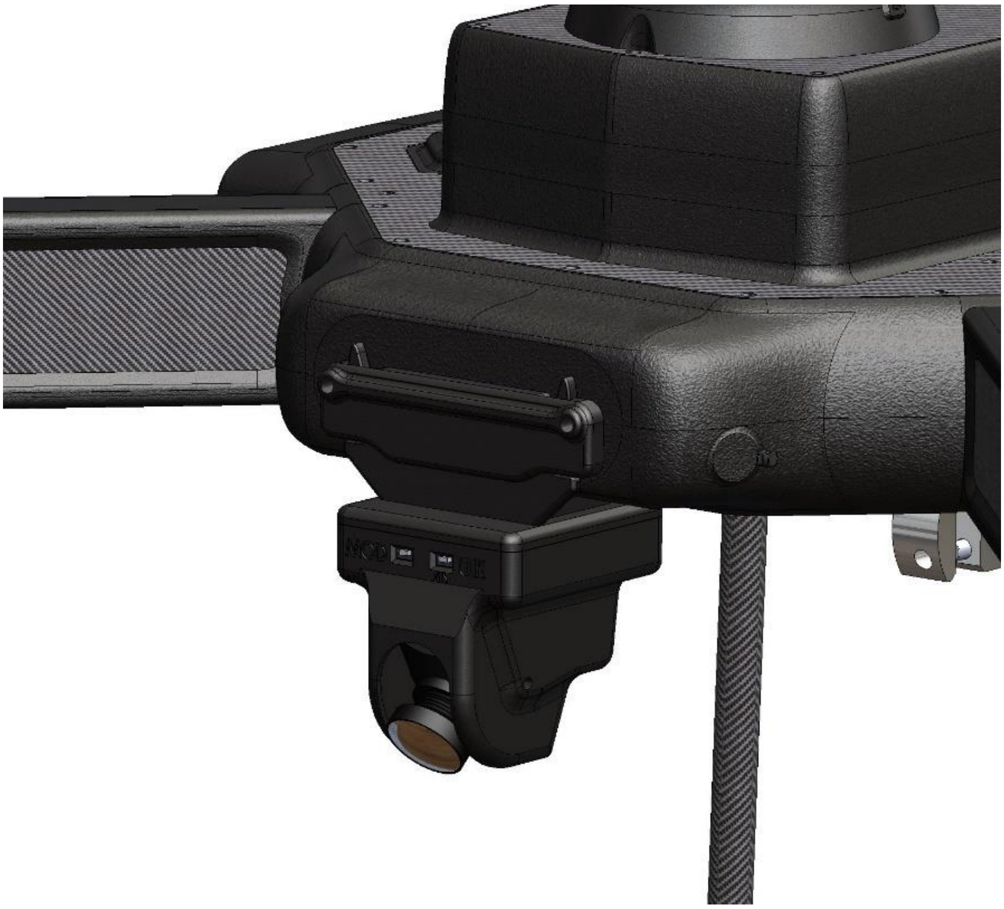


Figure 18: Navigation camera - CAD integration system

2.1.1.7 Obstacle Detection sensors evaluation

An accurate comparison analysis has been carried out to select the suitable sensor for the obstacle detection. The analysis has examined different technologies. The following considerations combine the salient steps and technologies descriptions of greatest interest for this thesis.

Following the study of different obstacle detection and collision avoidance methodologies, it has been possible to select the types of sensors that allow to work with these methods. Using an iterative decisional analysis, sensors that allow to apply as many methodologies as possible with adequate performances have been studied and selected. This type of systems can be generally defined as distance sensors; they are useful for many applications, such as altitude measurements, UAV terrain following, environmental mapping and collision avoidance. Two types of sensors best approach the application of the methods studied, offering the best performance and ensuring high levels of precision and processing speed of data collected. They are LIDAR and camera sensors.

LIDAR

A Light Detection and Ranging sensor, even told Laser Imaging Detection and Ranging, is a technology based on a precise measurement of the time delay between a transmission of a pulsed optical laser light signal and its reception. The sensor exploits two light signals: the reference signal fed from the transmitter and the received signal reflected from the obstacle. Using a signal processing method, it calculates the time delay between the two signals and define the distance between the sensor and the object detected. This permits to analyse objects that are far or close from the UAV and also allows to produce space mapping. LIDAR sensors usually work at different wavelengths depending on the specific application. For terrestrial mapping it is usually used at near-infrared wavelengths. Other similar technologies are RADAR and SONAR, less used in UAV applications for obstacle detection.

Basic LIDAR models are often applied for vertical obstacle detection, for example in the case of altimeters, such as the one previously described for the designed system, or for horizontal obstacle detection, for simple applications or experimental tests. They are static, have a limited range and do not allow high performance applying a collision avoidance method in three-dimensional space. For this reason, dynamic lidars have been developed, defined as rotating lidar. Thanks to the rotation, the field of view is complete, 360 degrees. For the project, two models of rotating LIDAR have been analysed: Slamtec RPLidar A2 and Slamtec RPLidar A3.

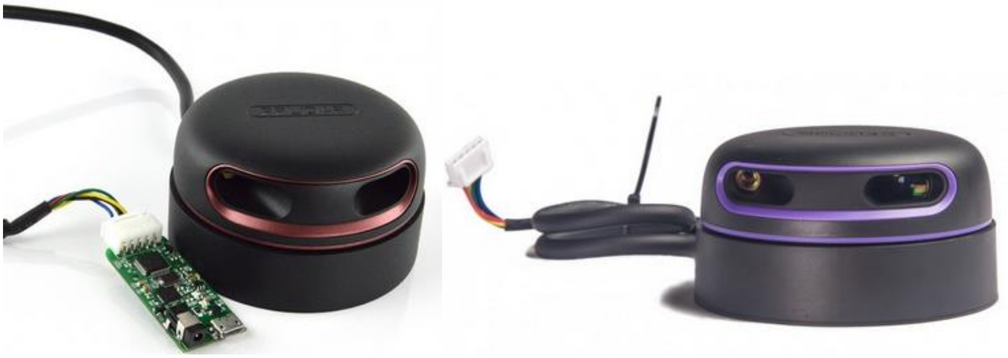


Figure 19: Slamtec RPLidar A2 and A3 models (55)

They are a low-cost type of LIDAR sensor suitable for indoor robotics applications, especially for SLAM and space mapping. Their range distance varies according to different models. For this reason, the main features of the two models have been analysed, in particular the action range, the frequency of signals sampling and the cost. RPLidar A2 has a distance range of 6-18 metres and a sampling frequency of 4000-8000 sampling/second. Instead, RPLidar A3 can work in a distance range of 10-25 metres and manage a frequency of 10000-16000 sampling/second, but has a double prize compared to the other model. It is also necessary to consider the greater computational power required to process a greater amount of data. Both sensors are quite compact, with equal size: 76 x 41 mm (ø x h) and weight of 190 g.

Figure 20 illustrates the top view of a typical output obtained from a rotating LIDAR. In particular, the rays of light are reflected at the moment they encounter an obstacle and remain undisturbed in unobstructed space. The results of a lidar analysis are usually showed on a two-dimensional plane and must be combined with the flight data of the vehicle to compose a three-dimensional space representation.

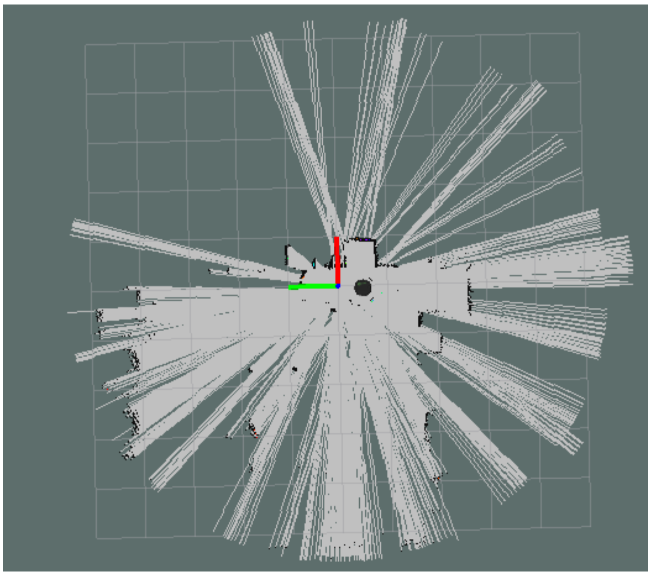


Figure 20: LIDAR sensor output example (56)

Camera

Whereas LIDAR data processing technology usually gives two-dimensional outputs, an even more suitable technology for three-dimensional obstacle detection is analysed: the use of optical, stereoscopic and depth cameras. Cameras are very common on UAVs for many reasons and on many types of applications. They can be very compact, allow to observe the surrounding space and sometimes to store images, for spatial mapping and digital space reconstruction. The sensors considered for the project are based on the principle of stereoscopy to calculate the distances at which the surrounding objects are located. For the stereoscopy principle, all cameras of this type have two objectives. Many of these sensors are also defined depth cameras as they are able to generate a depth map (57) in real-time during the internal images processing phase. The first sensor evaluated is the ZED Stereo Camera developed by Stereolabs.

Figure 21: Stereolabs ZED Stereo Camera model (58)

It is a 3D camera for depth detection and motion tracking; it captures a large-scale 3D map of the environment and understands how the UAV moves in space. It is compact and lightweight (dimensions: 175 x 30 x 33 mm, weight: 159 g), it has a field of view of 110 degrees and a distance range of 20 metres. The ZED camera analyses space in three dimensions. Using binocular vision and high-resolution sensors, it is able to establish the distance of objects around the vehicle from 0.5 to 20 m at 100 FPS, in both indoor and outdoor applications. Figure 22 shows an example of real-time outputs that a stereo camera generates and processes.

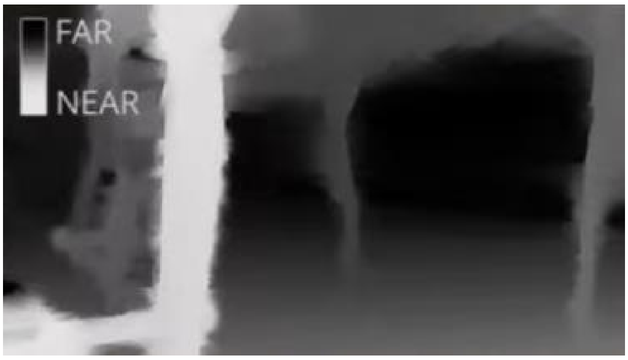


Figure 22: Stereo camera outputs result (58)

The camera is also able to track its position and orientation in 3D space at an update rate up to 100 Hz, with millimetre precision. The camera exploits stereo visual odometry, without the need for markers or external sensors. It is also possible to capture a 3D map of the space in real-time. The resulting mesh can be used to avoid obstacles in real-time, for visual effects or AR (Augmented Reality) on a global scale.

The other important sensor studied is the Intel RealSense Depth Camera model D435i. It is more compact than the previous one (dimensions: 90 x 25 x 25 mm) and cheaper. Moreover, it has some additional features very useful for high precision analysis, for example it has an integrated IMU (Inertial Measurement Unit). The IMU allows to refine depth awareness where the camera moves. It allows to do SLAM and tracking and allow better point-cloud alignment, improving environmental awareness for robotics and drones. The D435i camera also has an infrared projector and an RGB sensor integrated.

Figure 23: Intel RealSense Depth Camera D435i model (59)

Figure 24 illustrates both the Depth Histogram and the Infra-Red stream in real-time during a mission flight. The sensor has lower performance in terms of field of view and maximum range than the ZED camera: it has a field of view of 92 degrees and a distance range of 10 metres. Its field of view is suitable for robotics and augmented reality applications and in general it is considered ideal for fast moving applications.

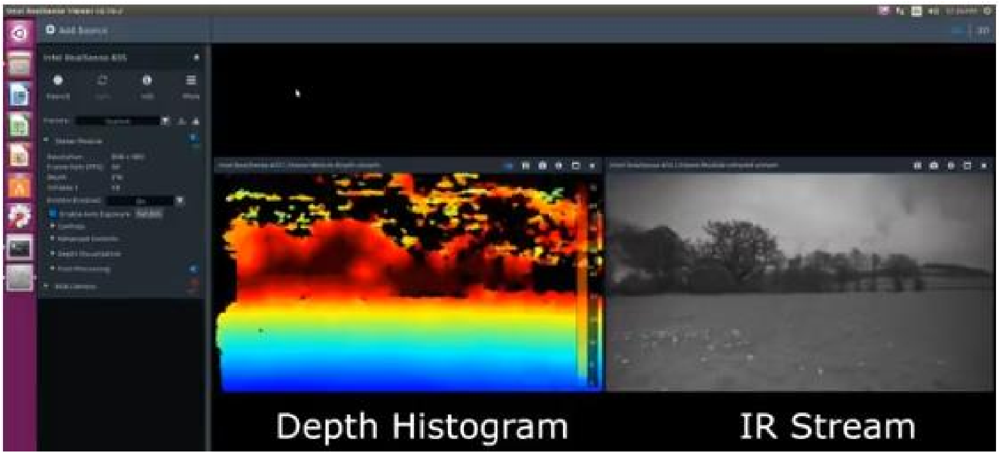


Figure 24: Depth camera outputs result (60)

At the end of the analysis of sensors suitable for obstacle detection, only one of them has been selected. The most acceptable sensor has been chosen following an iterative analysis between the main features of the sensors and the search for collision avoidance algorithms that supported them.

The Realsense D435i camera represents the best compromise, in term of overall dimensions, weight, operative range and cost. The field of view is sufficiently wide to ensure optimum performance. In addition, automatic flight missions will be programmed taking UAV always with the front facing the direction of flight, therefore it is not necessary to have a 360 degrees field of view coverage. Thus, the camera will be integrated in the front of the vehicle.

2.1.1.8 Companion Computer

A Companion Computer (CC) is an electronic board with considerable computing capabilities, equipped with I/O interfaces to communicate with the flight control units of drones. CCs applications can also be very complex. Before the advent of CCs, in order to create new features in drones, it was necessary to modify part of the autopilot code, which was very complicated, given the considerable complexity of programming. With the use of CCs, developers can create advanced drone features with on-board artificial intelligence, provide connectivity to the cloud across the network, integrate custom payloads and realize or connect custom web interfaces. In recent years, hardware and software libraries have been developed to exploit the capabilities of CCs.

Considering the implementation and management of the chosen obstacle detection sensor, it is necessary to integrate a Companion Computer able to manage the amount of data that the camera produces in output. In fact, the autopilot doesn't have enough computational power to do it autonomously. For the project it was first made an accurate analysis of the main CCs with appropriate characteristics and finally the Nvidia Jetson NANO model (Figure 25) has been selected.

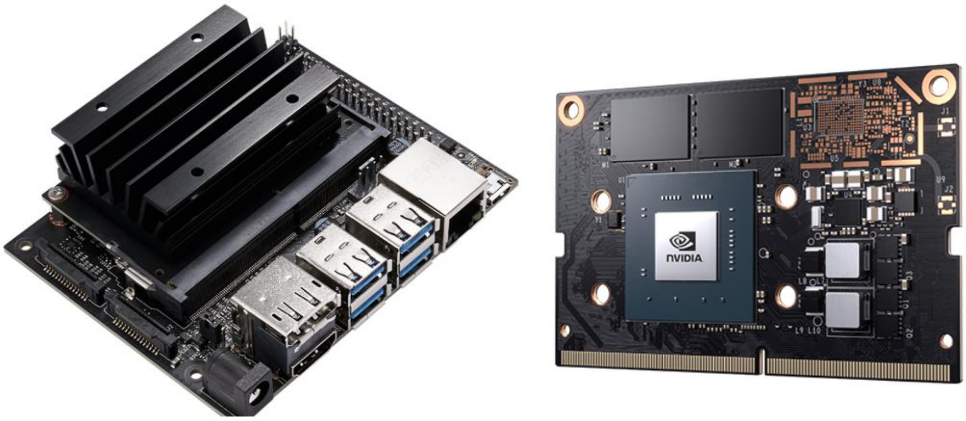


Figure 25: Nvidia Jetson NANO Companion Computer (61)

It has a GPU Maxwell 128 CUDA core, a 4 GB RAM 64 bit and 25,6 GB/s, a WiFi-Bluethoot module integrable and a built-in cooling fan module. It enables the development of new small, low-energy, economic AI systems. This powerful CC opens new possibilities for integrated IoT applications, including auxiliary UAV functions such as payloads management and sensors processing power augmentation.

2.1.2 Software Components

Once the hardware components of the GNC system have been defined, it is necessary to evaluate the software alternatives to support the usage components. The software chosen for the management of the autopilot will be described below.

During the evaluation of the software that has to manage the Pixhawk 4 autopilot, it was chosen to install the firmware PX4, for reasons related to the specificity of development, the ease of use of the human-machine interface and for the ability to manage additional components in interaction with the autopilot, such as sensors for collision avoidance. PX4 is an open source professional autopilot management software, developed specifically for Pixhawk 4 hardware, capable of handling almost all types of vehicles, from racing drones to cargo UAVs, up to terrestrial vehicles and submarines. It is able to meet the flight requirements needed for applications such as aerial photography and complex missions performed in automatic mode; the latter can be programmed by setting virtual waypoints along the route, during the software definition phase of the mission. The software is also supported by many online libraries that allow to improve and expand the management and support capabilities of additional features and devices. The firmware can be installed on different compatible hardware platforms.

The firmware is configured accordingly to the UAV type. A firmware configured for a quadcopter is different to a firmware configured for a fixed-wing UAV. The quadrotor developed in the BLUESLEMON project will be equipped with brushless engines, which are operated by the flight controller via Electronic Speed Controllers (ESC). ESCs convert an incoming signal from the flight controller into an adequate level of RPM. The PX4 firmware supports ESCs that take a PWM input, ESCs that use some different standards and UART protocol. The connection of the engines with the Pixhawk 4 autopilot is carried out through the Power Management Board, unlike the direct connection that would be on the Carrier Board of previous autopilot models.

The software is able to support the management of various types of vehicles, in particular considering them divided into the following categories: Multicopter, Plane, VTOL (Vertical Take-Off and Landing), Underwater robot and Rover. As for the category of interest for the project, there are many types of multicopter configurations, both according to the number of rotors present, and according to the configuration assigned. For the following project, the "X" quadrotor configuration has been chosen (Figure 26).

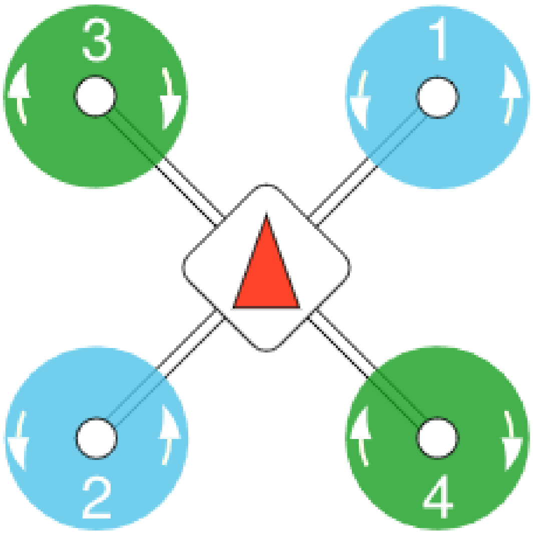


Figure 26: Quadrotor "X" configuration (62)

The figure above shows the placement of the rotors with respect to the central frame of the UAV body and the rotation direction of the four rotors. The rotors which must rotate counter-clockwise are identified in blue, while the rotors which must rotate clockwise are identified in green. For the four-rotor X configuration, the following standard is considered for the output channels of the remote control that allow the control of UAV by sending commands:

- AUX1: pass throughout RC AUX1 channel;
- AUX2: pass throughout RC AUX2 channel;
- AUX3: pass throughout RC AUX3 channel;
- AUX4: pass throughout RC FLAPS channel.

The following figures report some CAD images of the UAV designed for the experimental setup. In Figure 27 a construction scheme of the entire designed vehicle is represented.

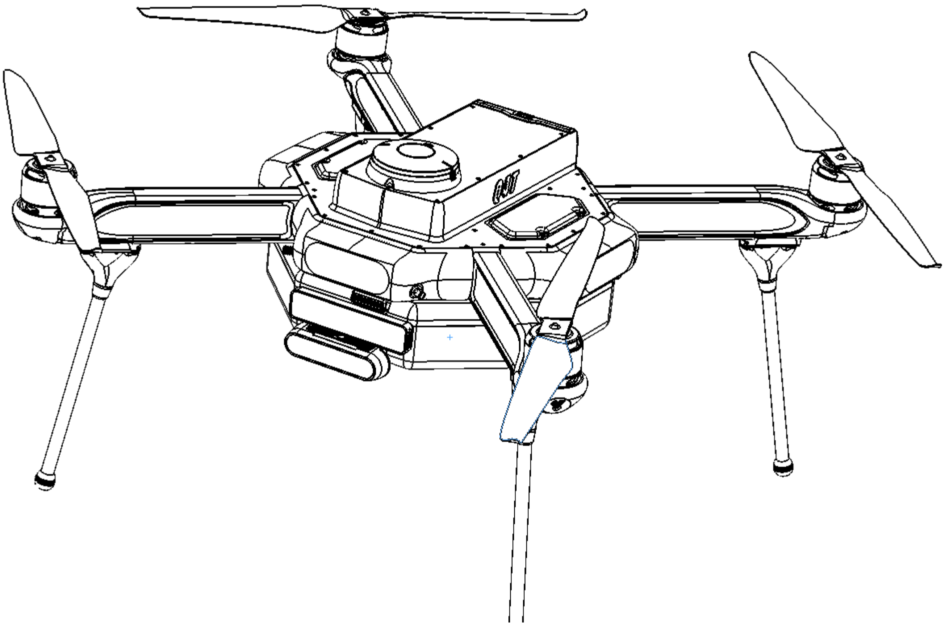


Figure 27: Construction scheme of the UAV

In Figure 28 it is possible to observe the CAD model realized, with perspective view from the bottom, in particular with an additional case in transparency. It has been designed as a modular element for integration of collision avoidance sensors (the Realsense camera is visible, located on the front of the vehicle, out of the case), the Companion Computer and some other components of the GNC system.

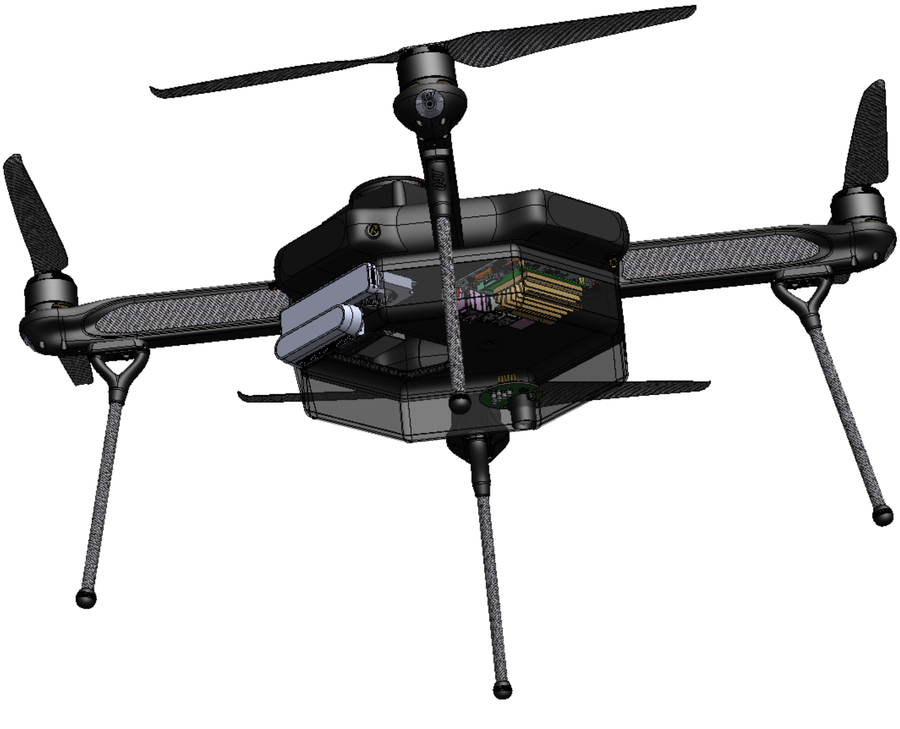


Figure 28: CAD model of the UAV designed

Finally, in Figure 29, a section of the structure that highlights only the central body of the aircraft and the modular case showed in transparency is represented. The Companion Computer inside the case, an aeration fan and the camera outside the case are clearly visible.

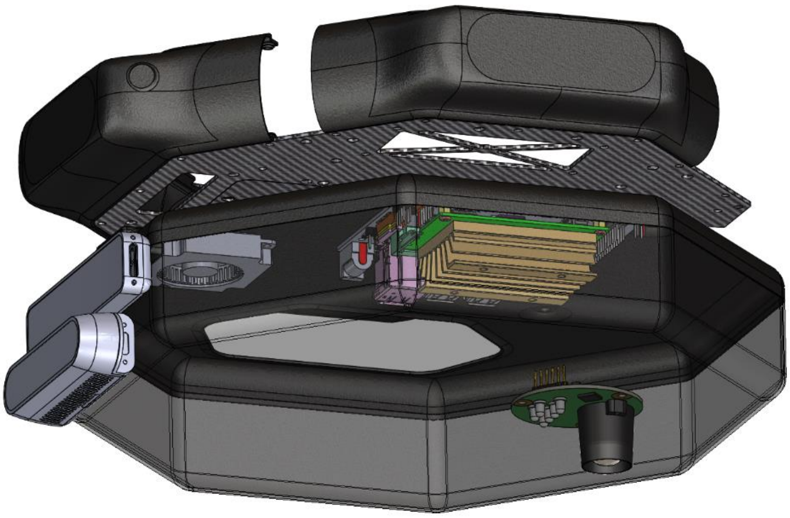


Figure 29: Modular case for collision avoidance sensors and GNC components

2.2 Ground Segment

The ground segment is the set of components and instruments that, together with the UAV, compose the Unmanned Aerial System (UAS). The ground segment is essential to carry out the activities necessary for the study, management and control of the mission, both before, during and after the flight mission. In the following paragraphs, the hardware and software components that compose the system will be described and their peculiarities and tasks will be studied.

2.2.1 Hardware Components

Hardware components essential to the vehicle management and control are mainly two: the radio controller and the Ground Control Station. There are different additional systems that may be required to perform specific flight missions. For example, for UAVs that load payloads such as monitoring cameras, it is necessary to have a Payload Control System to rotate the payload from the ground, independently from the vehicle control. For this project, it is not necessary to implement this system, because the camera is integrated in a fixed position and will move with the UAV, in order to avoid obstacles in agreement with its trajectory.

2.2.1.1 Radio controller

The radio controller allows the pilot to manage the control of the UAV, the transition from one flight mode to another, the start of automatic missions and the sending of commands to the payload, for example to perform a release, to take photographs or to record videos. For the BLUESLEMON project, the remote control developed by Hex, called Herelink (Figure 30) has been chosen.

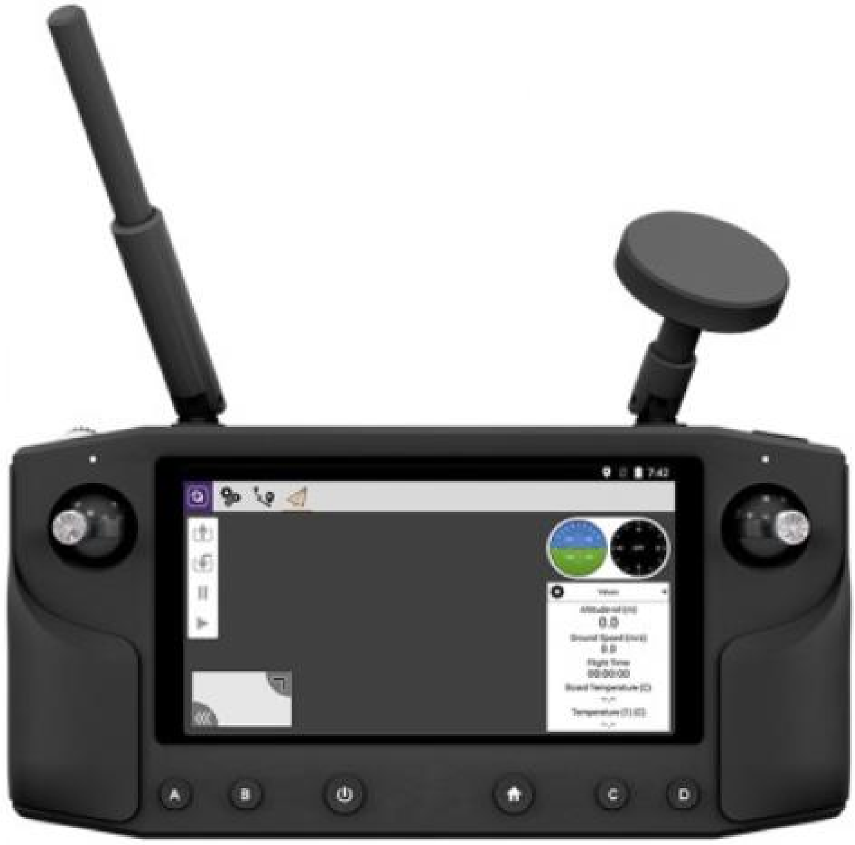


Figure 30: Hex Radio command, Herelink model (63)

The main features of this model are the integrated 5.5-inch color LCD touchscreen, the integration of the bidirectional telemetry, the long-range HD video transmission system and the flight control system, both at frequencies of 2.4 GHz. It is also designed and configured with the flight planner QGroundControl, based on Mavlink communication system and is equipped with two antennas to improve the power signal. Further important software features are wireless updates and wireless receiver configurations, which ensure a more reliable and secure connection between remote control and UAV.

2.2.1.2 Ground Control Station

All the ground activities, except the UAV flight control, are carried out using the Ground Control Station (GCS). The Ground Control Station can usually consist of a simple computer, or it can be developed and assembled specifically for different purposes. In this case, a commercial laptop will be used. It supports the use of software, the most important of which is QGroundControl (QGC), that will be described in the following paragraph. The computer is connected to the vehicle via telemetry, or as in this case via the Herelink receiver and ground module, allowing to observe in real-time the change of position, attitude and trajectory of the UAV during flights.

Considering the use of Herelink radio control, which is technologically very advanced, some typical GCS activities can already be carried out with the radio control, such as monitoring the vehicle vital parameters or its position on the QGC programme map, installed on Herelink by the developers. The main activities of mission programming, assessment and modification of aircraft flight parameters and design of automatic missions are carried out on the GCS. The following paragraph describes the features of the software used by the GCS to better understand its peculiarities.

2.2.2 Software Components

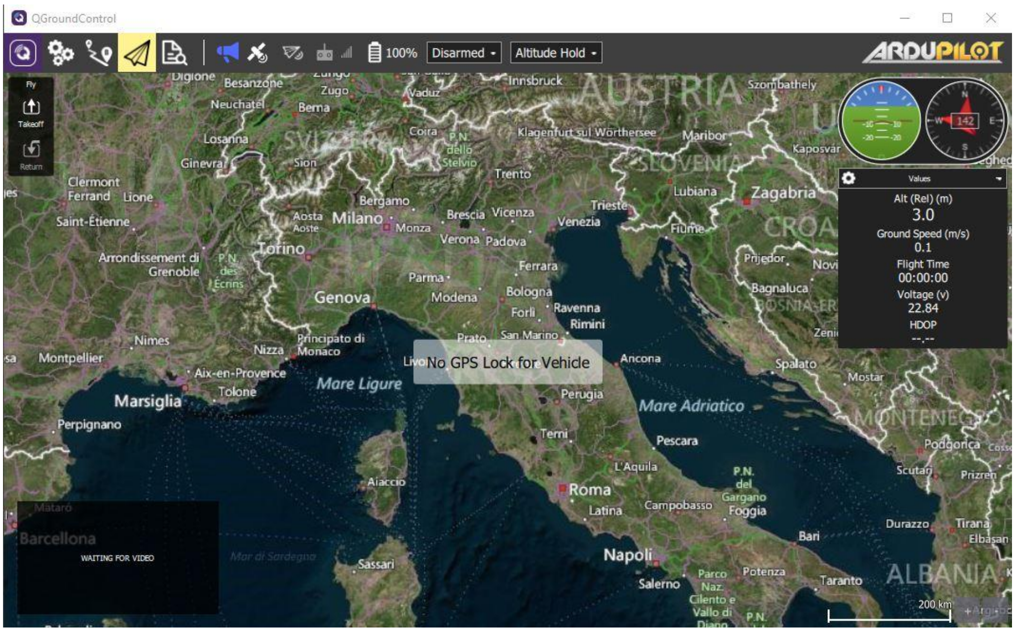
To follow and manage the mission from the ground it is necessary to access the mission planning and view the vital parameters of the UAV during the entire flight. To allow complete control of the flight, a software called QGroundControl is provided by the Dronecode developers and it is installed on the Ground Control Station. It serves primarily to load PX4 firmware easily on the flight controller hardware; moreover, it allows to perform many functions, both before and during the flight, by sending commands to any vehicle that is enabled to receive Mavlink signals. One of the main goals of QGroundControl is to ensure intuitive use for professional users and developers. In addition, the development code is entirely open source and therefore free of charge.

The main features of the software are listed below:

- Possibility to configure and perform the complete setup of PX4 (or possibly the Ardupilot firmware, for which it is compatible) on the autopilot of the vehicle;
- Flight support for UAVs with autopilot operated by PX4 or Ardupilot (or any other autopilot that communicates using the Mavlink protocol);
- Mission planning options for autonomous flights;
- Flight map display for real-time monitoring of position, trajectory and flight path, previously defined waypoints and instrument parameters on board;
- Video streaming with parallel display of instrument parameters on board;
- Support for managing multiple UAVs at the same time;
- Compatible with Windows, OS X, Linux, iOS and Android devices;
- Management of all aircraft types supported by PX4 and Ardupilot (multi-rotor, fixed wing, VTOL, etc...).

The QGC software allows to view the flight planner main screen with a wide availability of maps. Once the UAV is connected to the software through telemetry, it will be shown located on the map and will appear the vital parameters of the vehicle, such as altitude, battery voltage, current flight mode and so on (visible in Figure 31).

Through one of the main screens it is possible to design the flight mission fully automatic setting waypoints along the route. Parameters such as the altitude and flight speed of the UAV can be established for each vehicle.


QGroundControl

100% Disarmed Altitude Hold

Values

Alt (Rel) (m) 3.0

Ground Speed (m/s) 0.1

Flight Time 00:00:00

Voltage (v) 22.84

HDOP --.--

No GPS Lock for Vehicle

WAITING FOR VIDEO

Figure 31: QGC screen with flight map and UAV parameters

Through the main screen it is then possible to send commands to the vehicle manually, for example with the "take off" button (top left) it is possible to control the take-off of the multirotor and with the drop-down menu at the top it is possible to change flight modes, arm or disarm the UAV, read warnings and monitor the battery level. In the upper right corner, it is visible the change of attitude of the vehicle and its orientation.

Finally, during the completion of the missions it is possible to find flight information in real-time and monitor the variation of parameters, for example the level of oscillations along the three axes, parameters regarding the goodness of the GPS signal and many others.

CHAPTER 3: Testing Environment and Simulations

Parallel to the design of the experimental setup, the study and the choice of the most suitable collision avoidance method for this project have been carried out. In the following chapter, the reactive collision avoidance method will be described. The algorithm has been chosen as a result of the analysis provided in Chapter 1.

To implement this method in a simulation environment, different algorithms found in literature has been evaluated and then the most appropriate has been chosen and described in the next paragraph. The algorithm has been subsequently implemented in a simulation environment and personalized and optimized to simulate the system designed in Chapter 2. Moreover, additional analysis has been executed to select the simulation environment that has been used to simulate the UAV performing automatic flight missions and to test the collision avoidance system developed.

Finally, the entire system has been simulated and some assumptions have been made on different possible scenarios that the UAV can face in the real world and which could prove the functioning of the simulated collision avoidance system.

3.1 Collision Avoidance Approach

At the end of the analysis of collision avoidance methods developed and improved in recent years, an extension of the Vector Field Histogram method (1.2.4 Vector Field Histogram (VFH), page 22) has been chosen to apply to the following project. This methodology is based on an algorithm derived from the 2D VFH+ method and uses the octomap framework to define the three-dimensional environment in real-time. The algorithm that allow to extend the VFH approach to a three-dimensional method is called 3DVFH+. Its functioning is accurately explained in (64), were authors present the method for the first time.

The 3DVFH+ algorithm computes obstacle avoidance manoeuvres in a reactive manner. The algorithm uses an octomap to determine where the obstacles are located given the vehicle position in a 3D environment. It uses five stages to calculate a new vehicle motion. The concept of octomap can be summarised in a 3D occupancy grid mapping framework based on the octree structure. The octree data structure is a hierarchical structure containing multiple cubic volumes in space, also called nodes. The number of layers that compose the hierarchical structure define the size and precision of the octree (Figure 32).

The five steps that compose the algorithm functioning can be summarised as follows:

1. The octomap makes use of the octree data structure. When the vehicle moves in a large environment, the system has not enough processing power to explore and memorize all nodes of the three-dimensional space. Therefore, the first stage is called Octomap Exploring phase and it consists in a research of only nodes that lie within a bounding box around the vehicle.

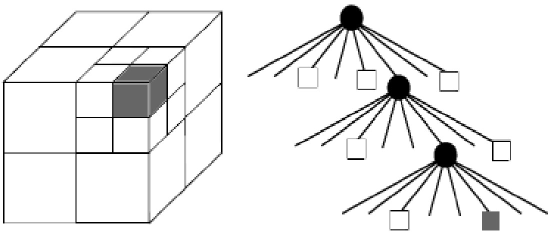


Figure 32: Octomap and Octree data structure (64)

   To determine which nodes need further exploration or which can be ignored, it is necessary to use the node location. The location of the nodes is not implemented in the octomap data structure to reduce the memory weight. Finally, nodes that are considered far from the vehicle box will not be explored. This improve the exploring speed without losing useful information. Nodes found in the Octomap Exploring phase are then used to create a 2D primary polar histogram.

2. The second stage is in fact called 2D Primary Polar Histogram phase. Information from the nodes are added into the 2D primary polar histogram. Figure 33 illustrates the polar histogram that shows the nodes position by two angles: the azimuth angle β_z (x-

axis of the 2D primary polar histogram) and the elevation angle β_e (y-axis of the 2D primary polar histogram).

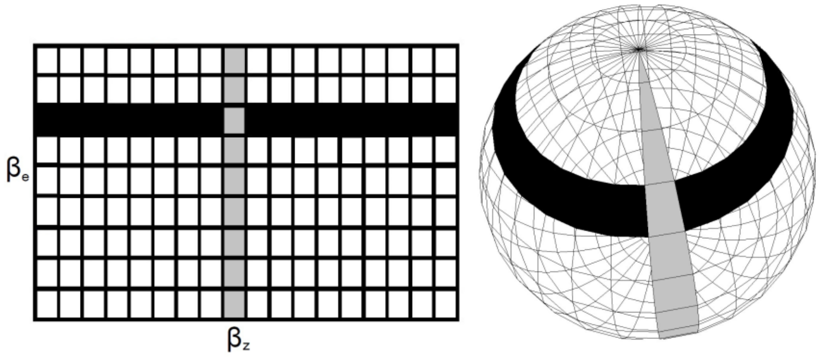


Figure 33: 2D Polar Histogram (64)

3. The third stage allow to calculate and add new information to the 2D primary polar histogram, especially the physical characteristics of the UAV and location of the nodes. This information regulates physical dynamic movements like the change of direction, which cannot change instantly. Indeed, the turning trajectory depends on the vehicle forward velocity, turning speed and climbing speed. All of these parameters are calculated during this phase inspiring to VFH+ algorithm calculations.

4. Following the generation of a 2D primary polar histogram based on the nodes of the octomap, the fourth stage reduce the information further by the creation of a 2D Binary Polar Histogram based on the previous one. Every cell of the 2D primary polar histogram is compared with two thresholds, τ_{low} and τ_{high}. At values higher than τ_{high} the value 1 is set; at values lower than τ_{low} the value 0 is set in the 2D binary polar histogram. If the values lie between the two thresholds, the next point will be used. The two thresholds allow the algorithm to distinguish real obstacles and to detect measurements errors.

5. The fifth stage is called Path Detection and Selection. It consists for searching available paths in the 2D binary polar histogram and selecting the path with the lowest path weight. The algorithm detects openings in the 2D binary polar histogram by creating a window that can move around the binary histogram. If all the elements in the window are equal to 0, the path is defined passable. When the window crosses the histogram boundaries,

the window uses elements connected by 2D polar histogram rules (Figure 34).

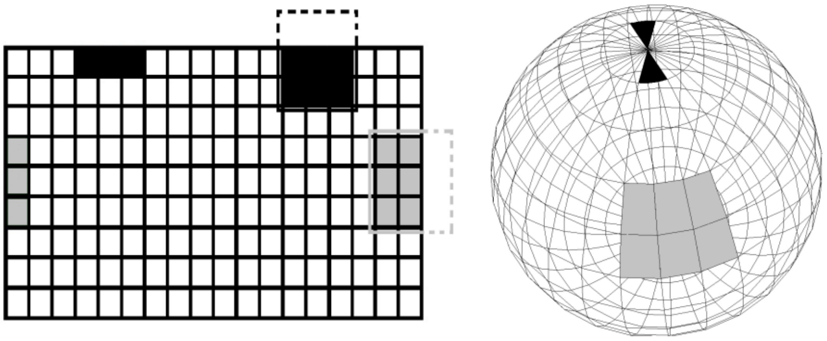


Figure 34: Moving window for path detection and selection phase (64)

When the path weight of all the candidate directions is calculated, the algorithm selects the direction with the lowest weight. This direction is then converted in a vehicle motion, using a decision tree that generates a vehicle motion based on the coordinates on the calculated direction.

Applying these five steps and using these techniques, the algorithm is able to calculate the UAV motion in real-time and in three dimensions. The method is sensor-agnostic because it was developed with the idea of not tailor the method for specific sensors. Actually, some technologies are more suitable than others. For the dissertation simulation phase, the sensor evaluated in the Chapter 2 has been simulated to be consistent with the experimental setup designed. Anyway, different sensors can be added to the algorithm following a data pre-processing phase.

The algorithm has been inserted in an open source library where several additional features have been added. They are described in the following paragraphs. The library has been defined *Obstacle Detection and Avoidance* (65) and has been used for this thesis in release version 0.3.1. In addition, the library has been developed specifically for simulations involving the use of PX4 autopilot firmware. Therefore, it has not been necessary to make any changes regarding the autopilot simulation, remaining consistent with the experimental setup defined in the previous chapter. Finally, the library is divided in two different collision avoidance approaches: Global Planner, based on the global obstacle avoidance and Local Planner, defined by the local obstacle avoidance. They are explained in the following paragraphs.

3.1.1 Global Obstacle Avoidance

In general, global obstacle avoidance algorithms use a map of the environment or build a map of the space around the vehicle using acquired sensor data during flight missions. An accurate representation of the three-dimensional world can require a lot of data. Considering a static environment, collision avoidance could be made only by mapping the environment and planning the best path around obstacles detected. These methodologies are generally computational heavy and they are complex to be implemented on small platforms such as light UAVs.

The Global planner is a global, graph-based planner that plans in a traditional octomap occupancy grid. Figure 35 shows an example of a Global Planner simulation at the beginning phase, in which the field of view of the sensor simulated is visible and the obstacles detected from the sensor define black squares in the map generated on the ground. It represents the occupancy grid and it remain memorized during all the flight mission.

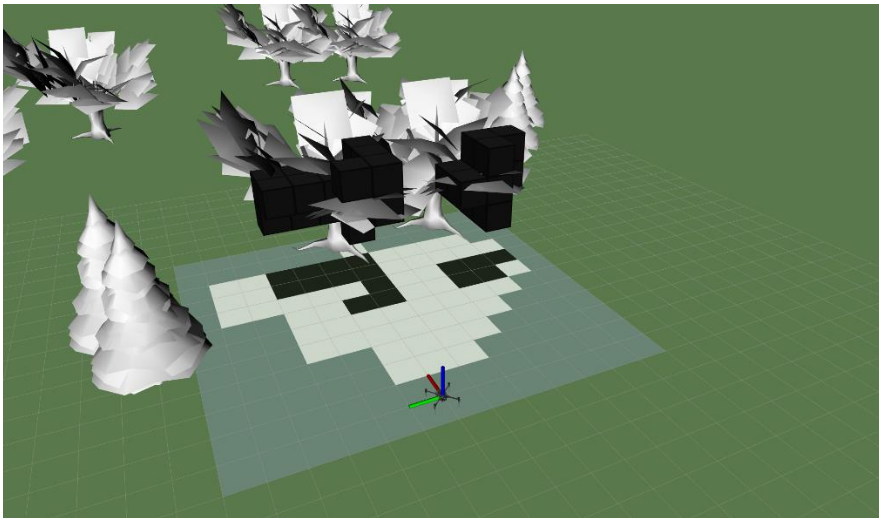


Figure 35: Global Planner algorithm functioning (65)

The Global Planner is computationally more expensive than the Local Planner because it builds a map of the environment. Moreover, accurate global position and heading are required to build a map that is good enough for navigation. For these reasons the Global Planner is not been applied for simulations for this dissertation. Indeed, for the project it is not important to generate a map of the environment but rather it is very important to avoid obstacles reactively and in real-time.

3.1.2 Local Obstacle Avoidance

The Local Planner is based on a local 3DVFH+* method that plans the best path using a vector field histogram. It is an extension of the 3DVFH+ algorithm because it considers also the VFH* extension (1.2.4 Vector Field Histogram (VFH), page 22). In (66) it is clearly explained how the VFH* algorithm has been combined with the 3DVFH+ algorithm to achieve a smooth UAV flight with a reliable obstacle avoidance behaviour. The main feature added to the 3DVFH+ algorithm is that the planner will have included some history during the path planning. This allow to optimize the trajectory control without add too much computational power and memory needed.

During an automatic flight mission, the UAV flies to mission waypoints dynamically, recomputing the path such that it is collision free. The algorithm can determine the position of the obstacles in real-time because it considers as obstacles only elements located close to the UAV. Therefore, the Local Planner requires less computational power then the other method implemented in the Obstacle Detection and Avoidance library, but it does not generate optimal paths towards the next waypoint because it doesn’t store information about the already explored environment. Finally, there is another feature in this library that is called the Safe Landing Planner. It is a local planner able to find safe area to land. This algorithm classifies the terrain under the UAV based on the mean and standard deviation of the z coordinate of point cloud points. This is then put into a 2D grid based on the xy point coordinates. This allow to find and locate flat areas, that are therefore considered safe landing zones.

The three algorithms presented are standalone and they are not meant to be used together. Therefore, the Safe Landing Planner could be considered an additional element to be implemented in a future algorithm that implements both local planner algorithms. Hence, the simulation phases of this elaboration will be carried out entirely with the aid of the Local Planner. It implements the method of greater interest, requiring less computational power and does not need to map the surrounding space. The following paragraphs are preparatory to the Software In The Loop (SITL) simulations phase, that will be reported in the next chapter.

3.2 Testing Environment

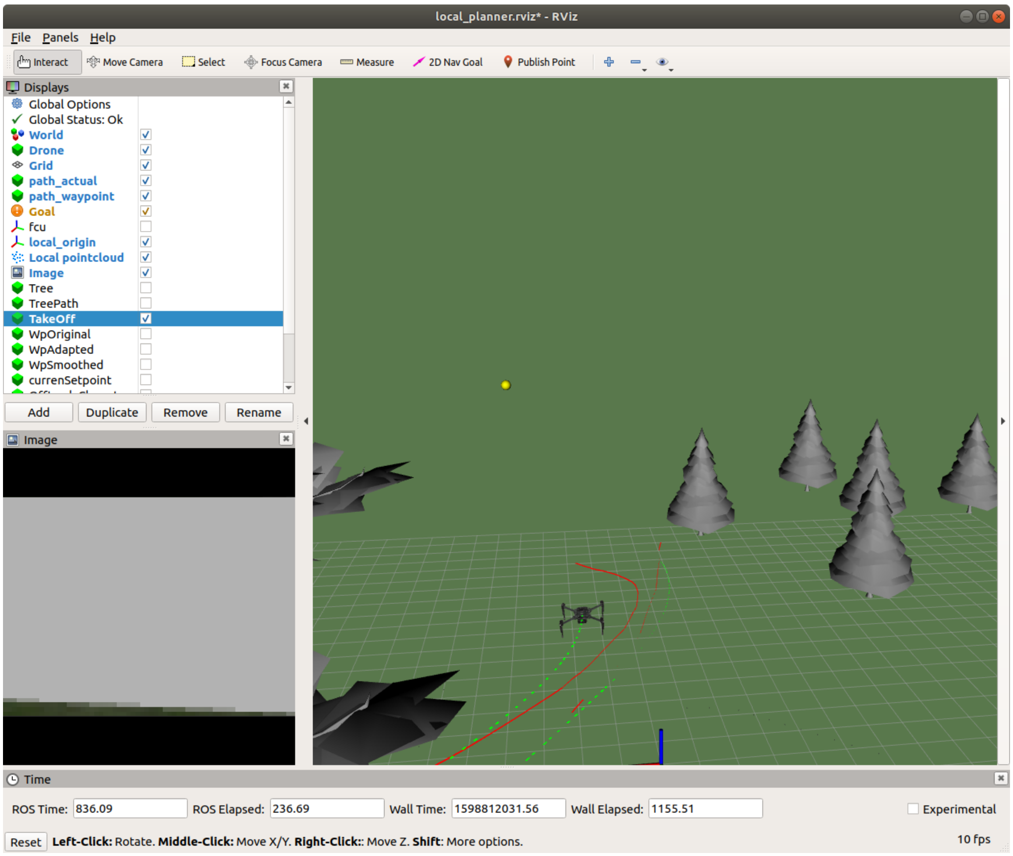
To study how the collision avoidance algorithm interacts with the system during automatic missions, the entire system has been simulated in an adequate testing environment. Experimental tests have been made firstly on simulations especially for security reasons and also because of the need to find an easy to deploy solution that was close to real UAV behaviours. For these reasons, a SITL approach has been preferred. it would allow to perform simulations, implementations and optimization of the collision avoidance algorithm without taking any risk.

In order to reproduce a 3D simulated environment, a suitable framework has been firstly chosen: Robot Operating System (ROS) is an open source library toolbox generally utilized to develop robot-based applications. It allows to operate with different environment simulators. For this thesis, Gazebo has been chosen. It is an open source 3D simulation environment that allows to simulate multi-robot behaviours on indoor and outdoor applications. These two softwares allow to test the collision avoidance algorithm, providing realistic scenarios with real physics simulation. Their implementation works as an external simulator for PX4 autopilot firmware allowing to simulate several activities, such as read sensor data from the virtual-created UAV, communicate between user inputs and the vehicle model, GPS information and 3D UAV physical stability simulations with user command responses.

3.2.1 ROS

Robot Operating System, called ROS, is an open source flexible framework that provides tools and libraries to design and coordinate robot software (67). It provides standard operating system services and its functioning is principally based on nodes. A node represents a ROS-based process; different nodes can communicate by sending messages and commands through ROS, such as the path finding algorithm. It is composed by various ROS nodes and also the entire algorithm can be considered a ROS node for obstacle detection and avoidance, specifically developed for PX4 firmware. Indeed, the entire algorithm is defined as PX4 computer vision algorithms packaged as ROS nodes for depth sensor fusion and obstacle avoidance (65). It can communicate with other part of the system that are also composed by ROS nodes. For example, the autopilot can communicate with the rest of the system through a ROS node called MAVROS, by the use of MAVLink communication protocol. Indeed, it is a node used to convert ROS messages in MAVLink messages and vice versa allowing vehicles to communicate with ROS.

ROS provides an interface for message passing between different nodes and a primary visualization tool called RViz (Figure 36).



Global Options
✓ Global Status: Ok
World ☑
Drone ☑
Grid ☑
path_actual ☑
path_waypoint ☑
Goal ☑
fcu ☐
local_origin ☑
Local pointcloud ☑
Image ☑
Tree ☐
TreePath ☐
TakeOff ☑
WpOriginal ☐
WpAdapted ☐
WpSmoothed ☐
currenSetpoint ☐

ROS Time: 836.09 ROS Elapsed: 236.69 Wall Time: 1598812031.56 Wall Elapsed: 1155.51 ☐ Experimental

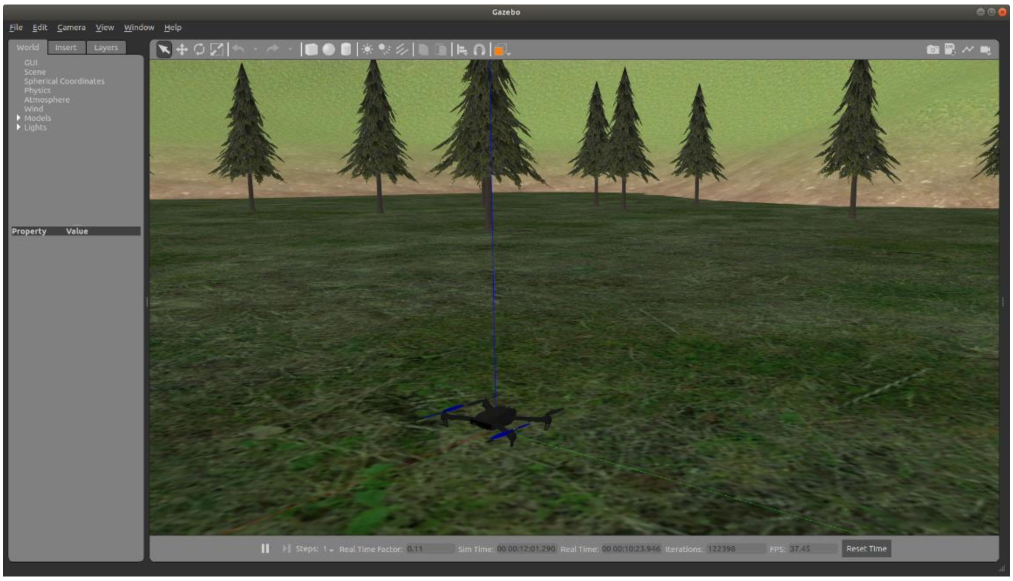
Reset **Left-Click:** Rotate. **Middle-Click:** Move X/Y. **Right-Click:**: Move Z. **Shift:** More options. 10 fps

Figure 36: RViz visualization tool

The general ROS configuration of the algorithm used is composed by four principal nodes: the UAV Node, in which is inserted the communication driver for autopilots, defined Mavros node; the Camera Node, directly connected to the Stereo Processing Node, only used in simulations; the Local Planner Node for running the entire obstacle avoidance algorithm, processes data and publishes new waypoints positions, then sent to the autopilot through the Mavros node.

3.2.2 Gazebo

Gazebo is an open source 3D simulation environment capable of simulating robots' behaviours, sensor readings, physical interactions between objects and some other different features (68). It is a complete environment with high resolution graphics and great simulating performances (Figure 37). Real physics simulations can detect collisions between the vehicle and the world simulated.



Steps: 1 Real Time Factor: 0.11 Sim Time: 00 00:12:01.290 Real Time: 00 00:10:23.946 Iterations: 122398 FPS: 37.45

Figure 37: Gazebo environment

Gazebo also provides sensors models and generates sensor feedback and physical interactions between sensors and other objects. Moreover, it is compatible with ROS, therefore for the Local Planner Node is easier to process sensor data whether the sensor is simulated or if the data come from a true sensor, such as in case of Hardware In The Loop (HITL) simulation. In this way, data from artificial sensors can be published as a ROS topic. In the next paragraphs, the vehicle and the collision avoidance sensor simulated to test the collision avoidance algorithm and simulate the entire system are described.

3.2.3 Platform simulation

Initially, it has been evaluated the idea of generating a Gazebo-compatible file from the UAV CAD model developed during the design phases, some of which are described in Chapter 2. But several factors, such as the high weight of the file in memory, the complexity of the model, the presence of details and components not essential to this type of simulation, which made the model even heavier at a computational level and the difficulty of integrating physical characteristics within the model, led to the choice to use a simulation model already fully implemented on Gazebo.

In order to choose the most suitable model of multirotor to best replicate the designed vehicle, a model with the same most important features for flight has been searched, in particular the autopilot and the general characteristics of the external frame, to guarantee the same flight properties. These are also the only features of the drone model showed in RViz tool, because for simulations it is not necessary to have such as a precise external case or specific shapes, but physical characteristics and flight control functioning are the most important features to simulate.

The simulated platform chosen for the tests is a quadcopter, 3DR Iris model (69). It has the same autopilot simulated, the Pixhawk 4 with PX4 firmware and the same physical characteristics of the UAV designed for the project. For the platform simulation, the Mavros node has been used to allow the communication between the UAV and the environment. It is a communication driver for autopilots that acquires information from the PX4 autopilot, such as state parameters, using the MAVLink protocol. This drone model does not have a camera integrated, therefore it has been necessary to simulate a sensor that allow to do obstacle avoidance.

3.2.4 Sensors simulation

Even if the method used is generally sensor-agnostic because it was not tailored for specific technologies, for the purpose some sensors are more suitable than others. Anyway, various different sensors can be integrated to the simulation model and to the algorithm. To be coherent to the designed system, a depth camera is simulated to inform the algorithm about the environment detected. The depth camera measures the distance to points detected in its Field Of View (FOV) and publish the data as a 3D point cloud.

In particular, the simulation of the Realsense D435i camera is carried out by the implementation of a stereo pair model attached to the UAV that provides the necessary simulated images. Then, an additional stereo processing node is used to convert stereo images into the required depth map. In a HITL simulation case, the real camera performs these computations internally and directly outputs the depth map.

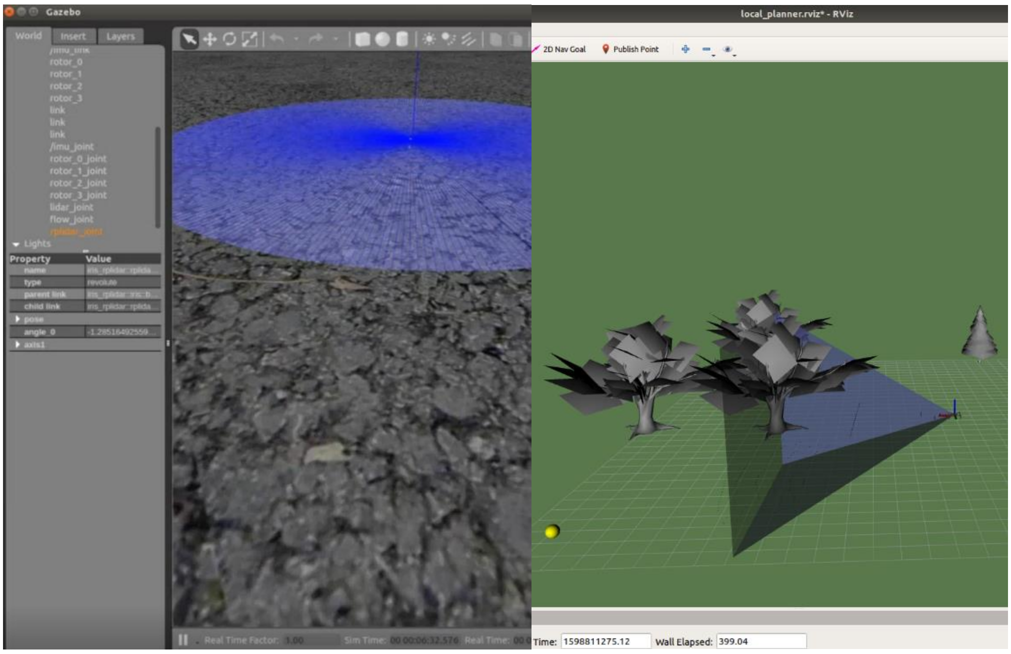


Figure 38: Sensors Field Of View simulations

Finally, in Figure 38, two different sensors are simulated to show different FOV during the obstacle detection phase. On the left a rotating lidar is simulated and integrated on the top of the drone; it has a 360 degrees horizontal FOV. On the right, the Realsense model utilized in the following simulations is simulated and integrated on the front of the vehicle; it has a horizontal FOV of 59 degrees and a vertical FOV of 46 degrees. Several sensors can define different collision avoidance performances. Moreover, the possibility of using several sensors at the same time could be evaluated effective from some points of view.

3.3 Obstacle Detection and Avoidance Evaluation

To best simulate the entire system designed, different environments were created to carry out flight missions. This analysis has been carried out to test the functioning of the algorithm, the collision avoidance sensor and the flight control system, simulating situations and environments that could lead to the evidence of system criticality. The different environments are defined as scenarios. For each scenario, the position of the UAV at the beginning of the flight mission and the original path are defined. The original path is designed with QGroundControl and then uploaded in the autopilot software. Then, different commands can be sent through QGC or directly through the Ubuntu Terminal, for example the rotor arming command or the starting automatic mission command. At the beginning of the mission, the UAV should be able to find a path allowing to move to the successive mission waypoint without colliding with any obstacle. During all flight missions, the flight mode of the UAV is set on Offboard, or on Mission mode. It means that the autopilot directly uses waypoints to navigate.

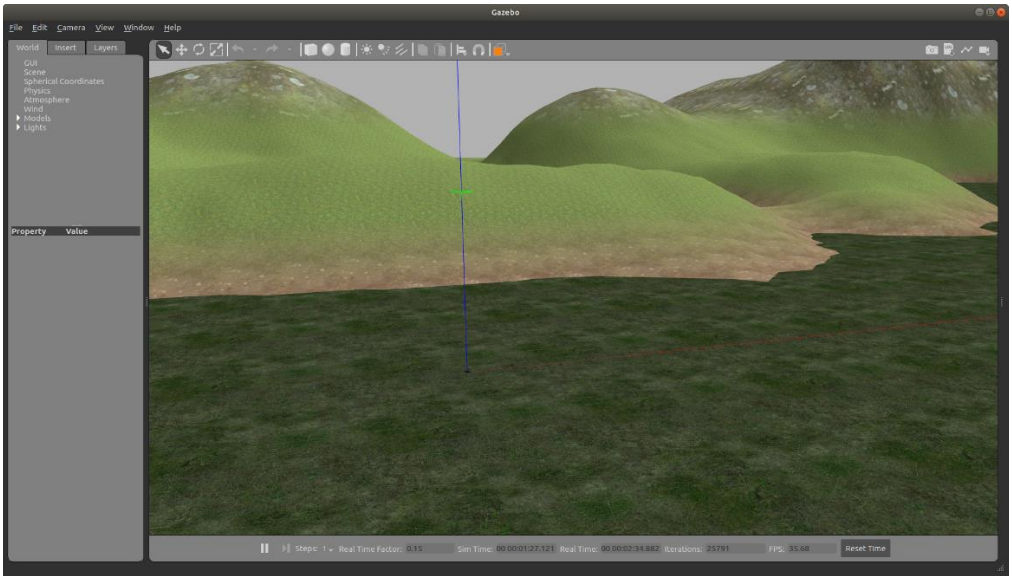
In general, scenarios have been created in outdoor environments and in typical alpine settings. These features have been defined to best simulate the scenario in which the BLUESLEMON project is developed, which is the monitoring of landslides in the alpine environment. The purpose of testing the system in these situations is to prove the ability of the method to solve critical problems that could originate in specific environments.

Finally, for all simulations, the basic functioning procedure of the algorithm is summarized as follows: the Local Planner Node runs the Local Planner algorithm and handles the communication to the other nodes. It receives vehicle state information from the Mavros node and the 3D point cloud from the Camera Node. Then the algorithm calculates next setpoints and publishes them through the Mavros node to the PX4 autopilot.

3.3.1 Scenario 1

The first scenario developed is an environment without obstacles. The first important factor to test for the collision avoidance system is to check when it works. In fact, it has been defined that it enters into action following the detection of obstacles within the FOV of the sensor. When no obstacle is detected, the collision avoidance algorithm must not redefine a new path and therefore the planned mission must be carried out entirely according to the trajectory defined during the mission planning phase on QGC.

The purpose of this simulation is therefore to observe the realization of the mission without any change of trajectory made by the collision avoidance system.



Steps: 1 Real Time Factor: 0.15 Sim Time: 00 00:01:27.121 Real Time: 00 00:02:34.882 Iterations: 25791 FPS: 35.68

Figure 39: Scenario 1 environment: no obstacles

In Figure 39, the environment can be observed without obstacles. However, it is clearly visible the presence of some areas of terrain with differences in altitude. These variations in altitude, in certain flight modes that do not follow the altimetric variation of the terrain, or for missions designed maintaining a constant altitude of waypoints close to ground level, can result as obstacles.

Therefore, on the basis of the simulated mission and the expected trajectory, two types of analysis can be defined: the verification of the non-interaction of the algorithm if any obstacle has been detected and the verification of the functioning of the collision avoidance system for the detection of vertical obstacles such as the variation in height of the terrain. The latter is an indispensable factor to be considered for landslide monitoring activities, because of the analysis of soils characterized by many height differences.

3.3.2 Scenario 2

The second scenario is the most inspired by a real environment. Indeed, for the BLUESLEMON project, several geographical sites in which to carry out the research project were examined and finally the Corvara site, in province of Bolzano (BZ), was chosen. The Corvara earth slide-earth flow is a large situ, more than three kilometers length, with a varying movement rates, from several metres to centimeters per year, constituted of a varying topography, from steep to flat, with open and forested areas (Figure 40). It is localised in an alpine environment, with an elevation from 1500 to 2150 metres.

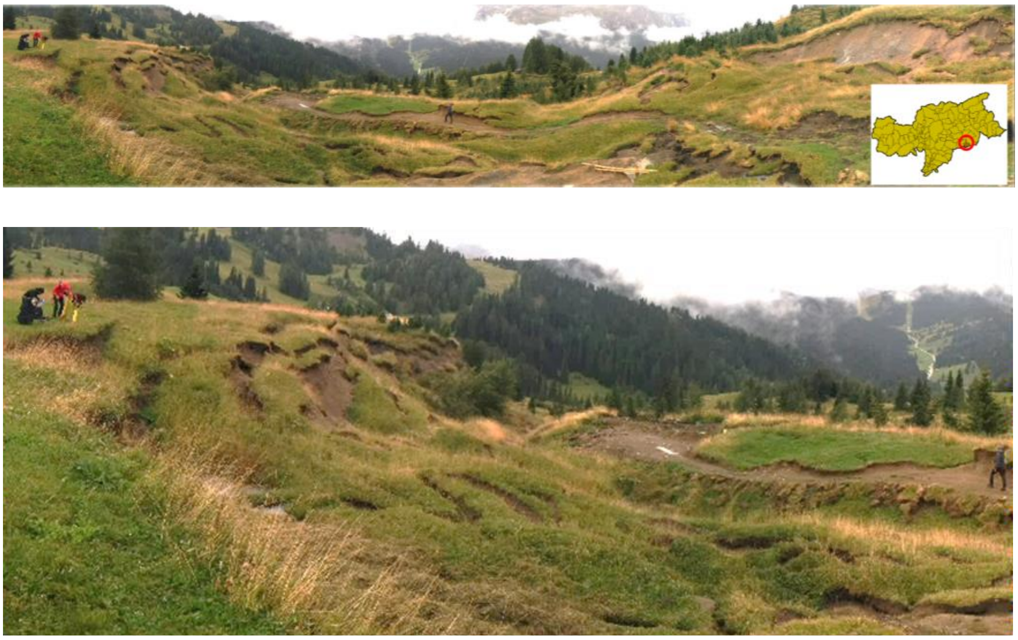


Figure 40: Corvara landslide – Bolzano (BZ)

In order to carry out monitoring activities, the UAV will have to move between various environments, steep, with vegetation and possible unforeseen obstacles along the flight path. Figure 40 shows the general trend of the landslide area. It is characterized by steep or semi-planar ground and trees scattered in a sparse manner along the ground, especially in flat areas. As a result of this analysis, Scenario 2 has been modelled taking into account the main characteristics of the area (Figure 41).

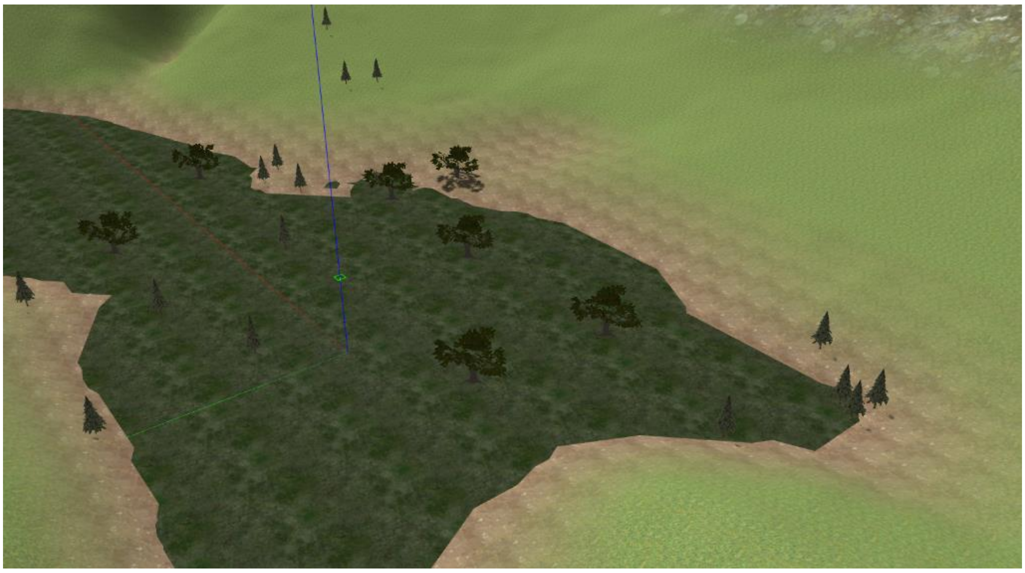


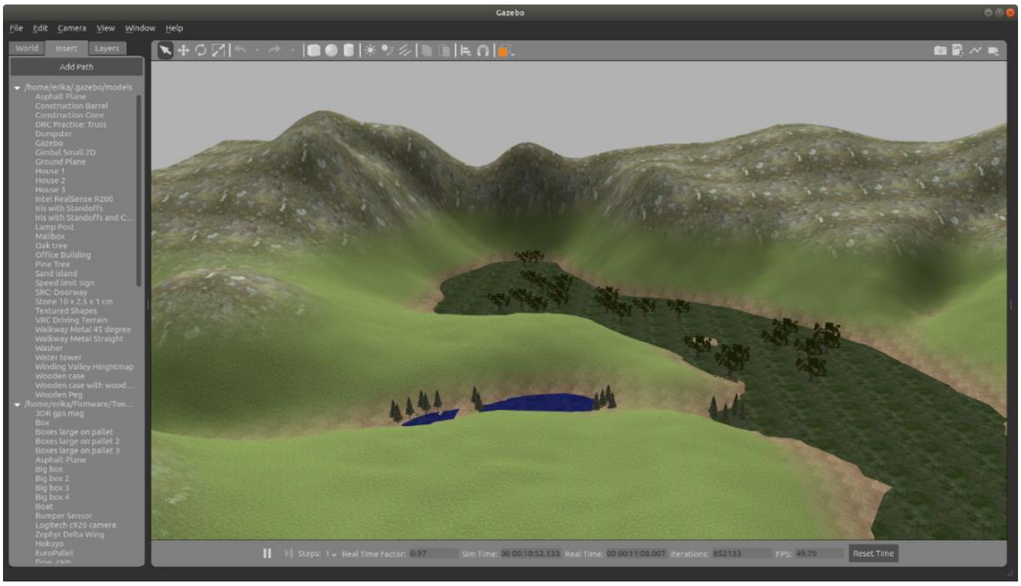
Figure 41: Scenario 2 environment: the landslide

Finally, during the mission planning phase, the geographical coordinates of Corvara were actually set so as to display on QGC the actual map of the landslide on which to plan the mission, in view of real flight missions that will be carried out on site.

3.3.3 Scenario 3

Scenario 3 has been designed as the most critical in term of environmental conditions: the slopes of the ground are much higher, the flat areas are narrower, forming valleys surrounded by mountainous areas. The obstacles are thicker; indeed, the vegetation is wooded and the trees have been positioned much closer, sometimes arranged in a way as to make the passage almost inaccessible.

The choice to make the environment more impervious was made to test the collision avoidance system even in the most critical areas of Corvara, thus allowing the UAV to monitor the entire scenario without limitations. This type of simulation needs to test the prediction of unexpected obstacles, which may appear suddenly and to test the most active and reactive part of the collision avoidance algorithm.



Steps: 1 Real Time Factor: 0.97 Sim Time: 00 00:10:52.133 Real Time: 00 00:11:08.007 Iterations: 652133 FPS: 49.79

Figure 42: Scenario 3 environment: rough environment

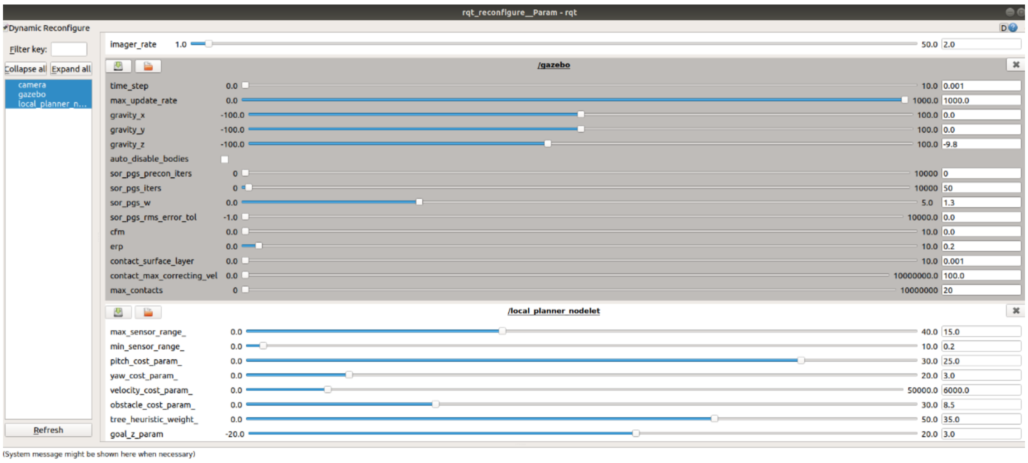
3.3.4 Evaluation of flight simulation parameters

Finally, keeping the scenarios developed and described in the previous paragraphs, a series of fundamental analyses has been possible performed to verify the variation of the algorithm performances as a function of the variation of different parameters.

There are a lot of parameters that affect the functioning of the collision avoidance algorithm; for example, during the 3DVFH+ algorithm initialization, it is necessary to configure some vehicle parameters for the UAV simulation, such as UAV mass, dimensions and speed, octomap resolution and multiple levels of thresholds to generate a 2D binary polar histogram. These parameters can be chosen empirically or can be set in function of the specific UAV simulated.

Other important parameters are related to the vehicle motion, such as drag, maximum acceleration constraints and time constants, or to the sensor simulation. Moreover, related to the obstacle detection and collision avoidance, parameters like safety radius and delta time resolution can affect simulations. With regard to computational constraints, time between updating the depth map, depth map resolution and similar parameters can be varied to analyse algorithm performances.

Considering the high number of parameters and combinations that can be reached to evaluate the algorithm functioning, analyses will be focused on specific parameters according to the type of simulation and scenario used. Instead, for the simulations carried out in the three scenarios without parameter evaluation, the standard parameters defined during the initialisation phase of the simulations will be maintained.



imager_rate	1.0	50.0	2.0

/gazebo

time_step	0.0	10.0	0.001
max_update_rate	0.0	1000.0	1000.0
gravity_x	-100.0	100.0	0.0
gravity_y	-100.0	100.0	0.0
gravity_z	-100.0	100.0	-9.8
auto_disable_bodies	□		
sor_pgs_precon_iters	0	10000	0
sor_pgs_iters	0	10000	50
sor_pgs_w	0.0	5.0	1.3
sor_pgs_rms_error_tol	-1.0	10000.0	0.0
cfm	0.0	10.0	0.0
erp	0.0	10.0	0.2
contact_surface_layer	0.0	10.0	0.001
contact_max_correcting_vel	0.0	10000000.0	100.0
max_contacts	0	10000000	20

/local_planner_nodelet

max_sensor_range_	0.0	40.0	15.0
min_sensor_range_	0.0	10.0	0.2
pitch_cost_param_	0.0	30.0	25.0
yaw_cost_param_	0.0	20.0	3.0
velocity_cost_param_	0.0	50000.0	6000.0
obstacle_cost_param_	0.0	30.0	8.5
tree_heuristic_weight_	0.0	50.0	35.0
goal_z_param	-20.0	20.0	3.0

Figure 43: RQT tool: parameters evaluation

In Figure 43, the RQT tool is shown with some of the standard parameters set automatically at the beginning of a simulation. RQT is a ROS tool that allow to visualize and reconfigure parameters of both Gazebo simulator, sensors and the Local Planner.

CHAPTER 4: Simulations Results

The following sections contain the results of simulations carried out in order to test the collision avoidance system developed. The 3DVFH+ algorithm has been evaluated considering its requirements: it needs to avoid obstacles in a 3D environment and to perform these calculations in real-time. The selected obstacle detection sensor has been simulated and tested to verify its effective operation in this specific application field.

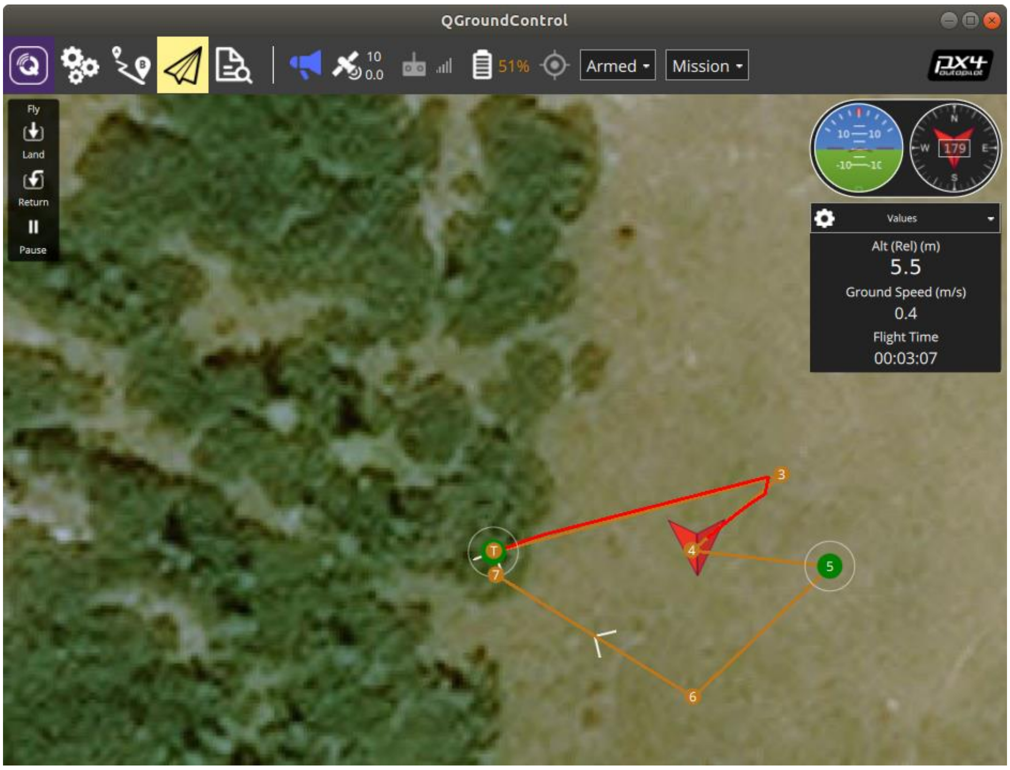
For each simulation, the main phases will be described: mission planning, simulation execution and evaluation of the obtained results. For each of them, crucial parameters will be highlighted and, subsequently, the mission will be fully analysed. In section 4.4 the most significant parameters of each simulation previously conducted are analysed, varied throughout iterative simulations and then optimised.

Finally, Appendix C collects additional images and graphic results for each simulation.

4.1 Scenario 1 Simulation

The first set of simulations has been carried out in the obstacle-free environment. For this reason, the UAV flight has been planned along a non-linear path, in order to test the dynamics of the drone in three-dimensional space, during a flight mission.

Figure 44 shows the QGC screen during the flight. The mission has been designed by inserting several waypoints connected by the yellow line. Afterwards, the mission is uploaded and ready to be accomplished if the UAV is set with the Mission or the Offboard flight mode. For each waypoint different parameters can be set, such as altitude and flight speed. In this case, the mission is carried out entirely at an altitude of 5 metres. In the figure, the relative altitude is displayed on the right and as can be seen, the UAV oscillates slightly in altitude (5.5 metres) between a waypoint and the next one. The ground speed is set for the entire mission to 5 m/s and it is visible in the parameters window on the right. During the turning phases, after the waypoint has been reached, the speed decreases according to UAV dynamics and reaches a value of 0.4 m/s when the UAV performs very narrow turns, with rapid change of trajectory direction. The red arrow in fact indicates the running position of the drone and its flight direction (in the image it has just reached Waypoint number 4), while the red line identifies the actual path travelled by the drone.

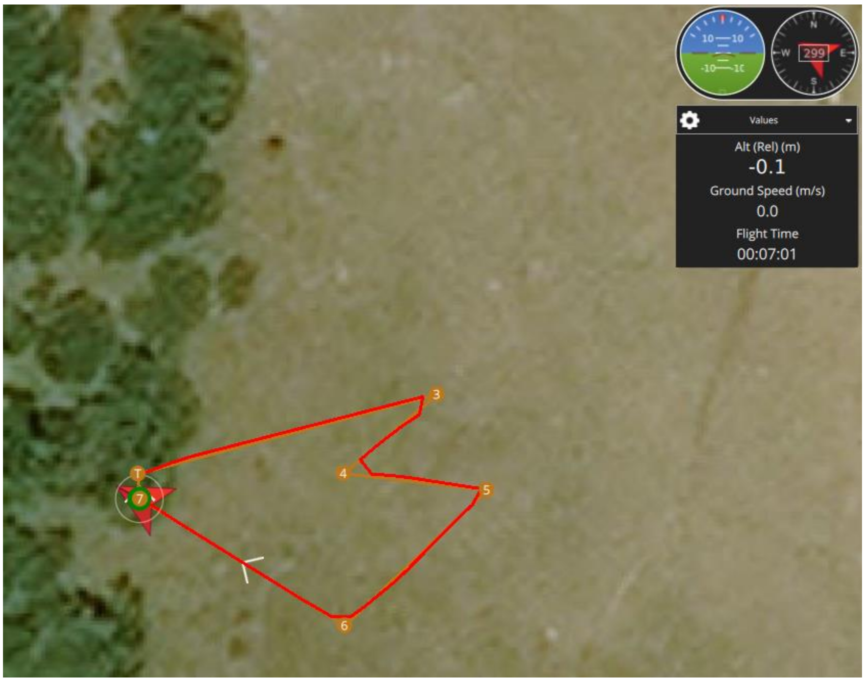


Values
Alt (Rel) (m)
5.5
Ground Speed (m/s)
0.4
Flight Time
00:03:07

Figure 44: Scenario 1 - Mission planning and flight visualization on QGC

According to Figure 45, the UAV has completed the mission and it has followed the initial path established during the mission design phase. The only difference in route is in proximity of waypoints. Indeed, a waypoint is considered reached as soon as the drone enters within a certain radius (NAV_ACC_RAD parameter: the standard value is set on 10 metres, but for the simulations performed it has been set on 2 metres) from the specific point. For this reason, the UAV does not complete the entire route around the targets, thus avoiding abrupt manoeuvres, especially around sudden trajectory changes, as in Waypoints 3 and 4. The parameter that adjusts the radius within which the target is reached can be changed, for example can be reduced if it is necessary to achieve greater accuracy of the target position.

During this simulation it is also possible to observe on Gazebo and on RViz how the vehicle moves with a good flight dynamic: it is able to fly forwards, up, down, right and left while maintaining a desired velocity. It is also able to move backward, but this type of movement is not used during the implementation of automatic missions as the collision avoidance sensor is placed on the front side of the drone.



Values

Alt (Rel) (m)
-0.1

Ground Speed (m/s)
0.0

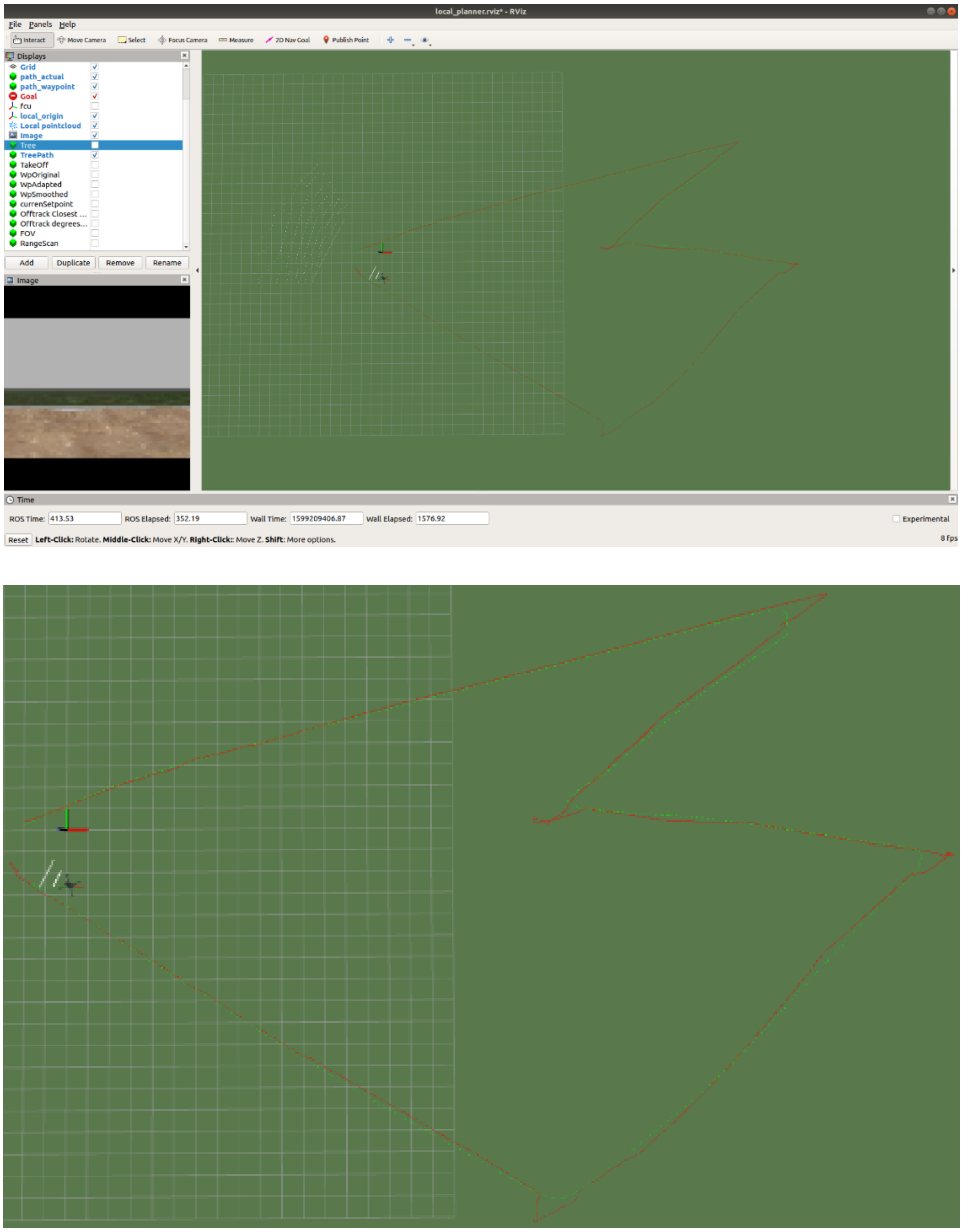
Flight Time
00:07:01

Figure 45: Scenario 1 results - Desired path execution

Figure 46 shows the entire path travelled by the drone on the RViz display tool. The red path represents the mission established to reach the waypoints avoiding obstacles detected, while the green dotted line identifies the path actually completed by the UAV. In this case they are almost coincident. On the RViz screen is also displayed the FOV of the camera and the real-time image of what the sensor sees (bottom left).

The route designed for the simulation just described has been placed entirely in a flat area of the scenario. Then, a second mission has been planned to verify the behaviour of the system in approaching areas no longer flat, characterized by hill terrain. The second path starts in a flat area and then moves to a soft hill, showed in Figure 47. The drone approaches the sloping ground with the aim of reaching the predetermined waypoint, keeping a minimum distance from the ground. The parameter responsible for maintaining a minimum standard distance set for all simulations will be described in

Paragraph 4.4. It is used to maintain a safety distance between the UAV and the terrain.



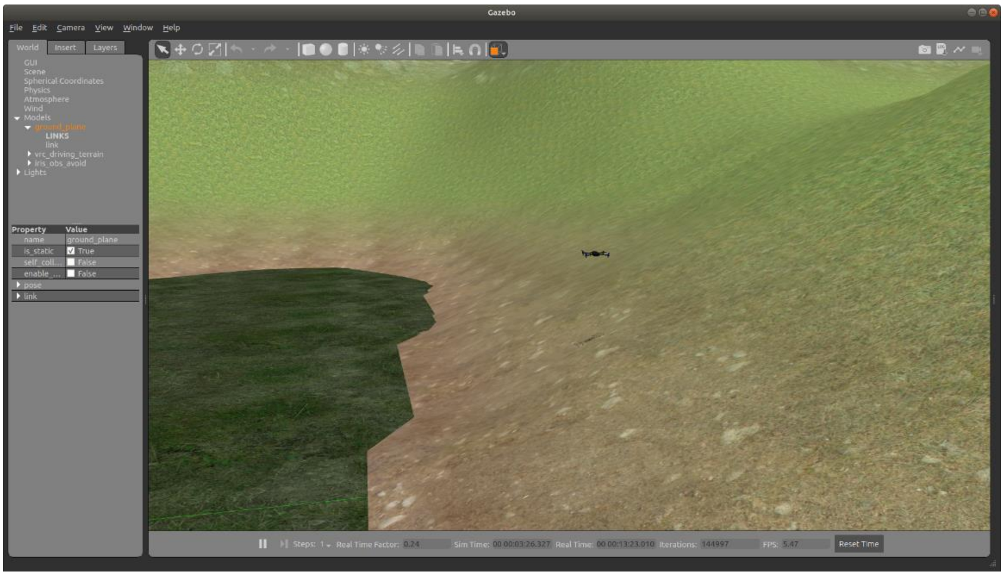
☑ Grid
☑ path_actual
☑ path_waypoint
☑ Goal
□ fcu
☑ local_origin
☑ Local pointcloud
☑ Image
□ Tree
☑ TreePath
□ TakeOff
□ WpOriginal
□ WpAdapted
□ WpSmoothed
□ currenSetpoint
□ Offtrack Closest ...
□ Offtrack degrees...
□ FOV
□ RangeScan

ROS Time: 413.53 ROS Elapsed: 352.19 Wall Time: 1599209406.87 Wall Elapsed: 1576.92 □ Experimental

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click:**: Move Z. **Shift**: More options. 8 fps

Figure 46: Scenario 1 results - Mission accomplished; top view, RViz visualization

The vehicle attempts to reach the waypoint located very close to the ground (5 metres from ground level 0, then placed just over 2 metres above the hilly ground), slows down, and stops at the minimum distance that the drone maintains from the obstacle detected inside the FOV of the camera, in this case represented by the terrain. Without interrupting the mission, as soon as the vehicle detects the terrain as an obstacle, the collision avoidance algorithm continuously recalculates the optimal path to reach the target respecting the flight and collision avoidance parameters set.



Property	Value
name	ground_plane
is_static	☑ True
self_coll...	□ False
enable_...	□ False

Steps: 1 Real Time Factor: 0.24 Sim Time: 00 00:03:26.327 Real Time: 00 00:13:23.010 Iterations: 144997 FPS: 5.47

Figure 47: Scenario 1 - Terrain altitude variation tests, Gazebo visualization

Figure 48 shows how the UAV changes direction multiple times before being able to generate the optimal path that allows to reach Waypoint 3. On the other hand, as far as Waypoint 4 is concerned, the vehicle is not able to reach it even after several attempts, as it is too close to the ground from all possible directions of arrival. In this case, the mission has been aborted early and an automatic mission abort command has been sent, followed by a landing command on the position reached as close as possible to Waypoint 4. Therefore, the mission has not been completed due to the hill too close to Waypoints 3 and 4.

Finally, it is noted that the route actually carried out (red line) differs greatly in the closeness of an obstacle, compared to the route initially planned (yellow line), as the UAV maintains the safety distances set by redefining a new trajectory. To solve this problem, the mission path or the standard flight and obstacle avoidance parameters can be modified according to each specific mission.

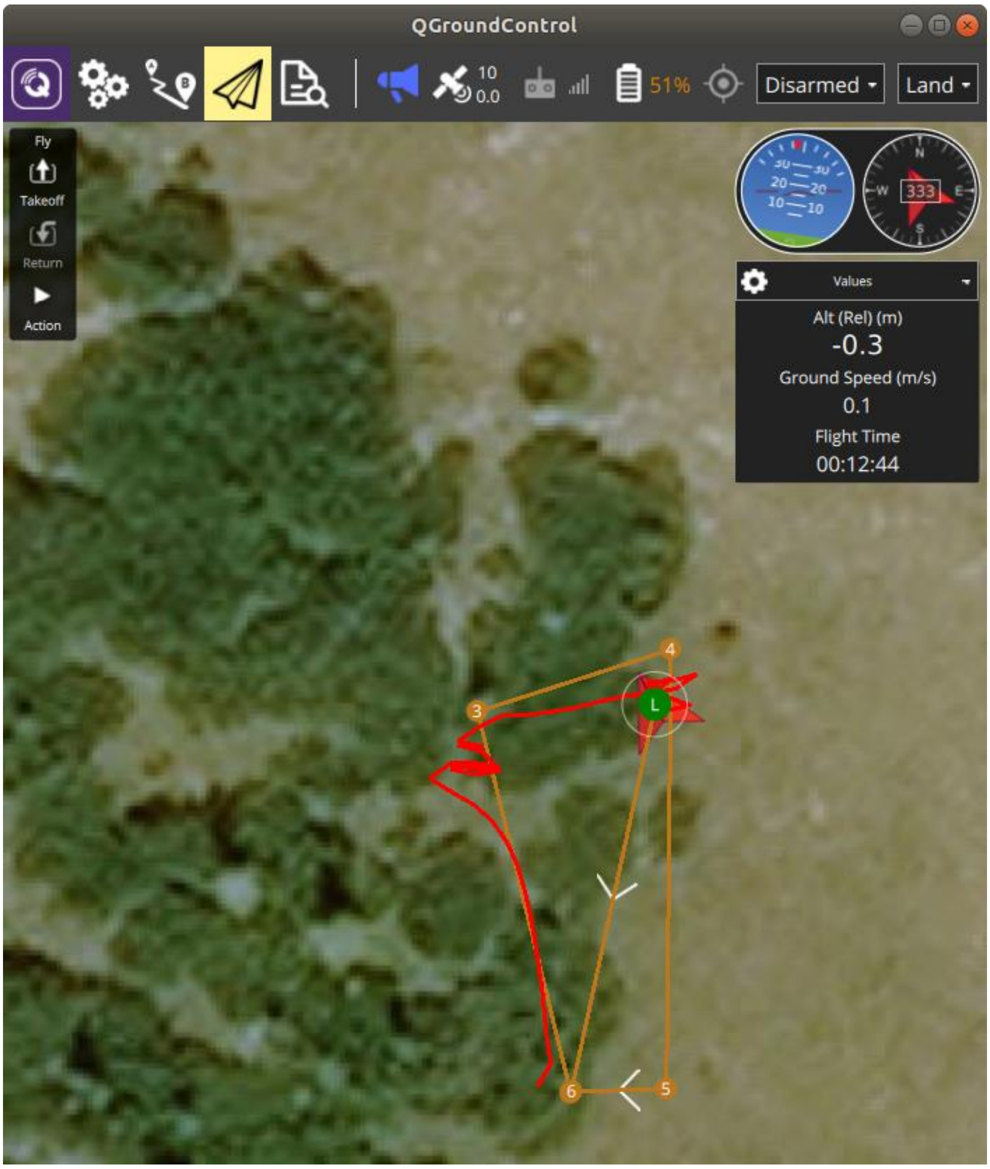
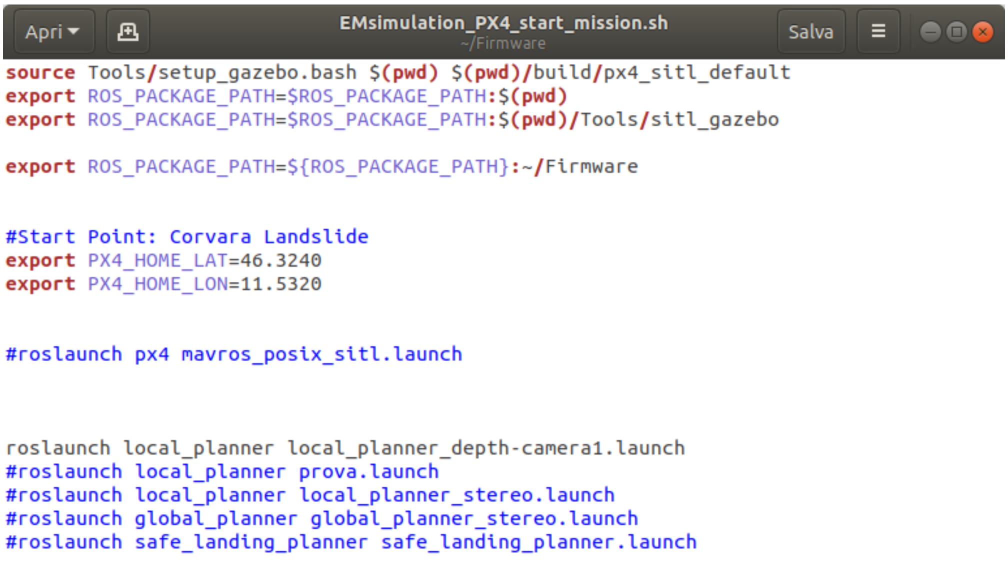


Figure 48: Scenario 1 results - Terrain altitude variation, mission not completed

4.2 Scenario 2 Simulation

The path for the second scenario is planned in order to simulate in the best possible way a real standard flight mission, defined according to the BLUESLEMON Project objectives. First of all, on QGC the geographical coordinates of the Corvara site are set up. This step has been performed through a simulation launch script that allows to place the UAV in specific geographical coordinates (Figure 49). The vehicle will appear in the centre of the simulation environment on Gazebo, but with specific geographical coordinates; its true position is then visible on the QGC map (Figure 50).



```
source Tools/setup_gazebo.bash $(pwd) $(pwd)/build/px4_sitl_default
export ROS_PACKAGE_PATH=$ROS_PACKAGE_PATH:$(pwd)
export ROS_PACKAGE_PATH=$ROS_PACKAGE_PATH:$(pwd)/Tools/sitl_gazebo

export ROS_PACKAGE_PATH=${ROS_PACKAGE_PATH}:~/Firmware


#Start Point: Corvara Landslide
export PX4_HOME_LAT=46.3240
export PX4_HOME_LON=11.5320


#roslaunch px4 mavros_posix_sitl.launch



roslaunch local_planner local_planner_depth-camera1.launch
#roslaunch local_planner prova.launch
#roslaunch local_planner local_planner_stereo.launch
#roslaunch global_planner global_planner_stereo.launch
#roslaunch safe_landing_planner safe_landing_planner.launch
```

Figure 49: Simulation launch script for Corvara geographical coordinates

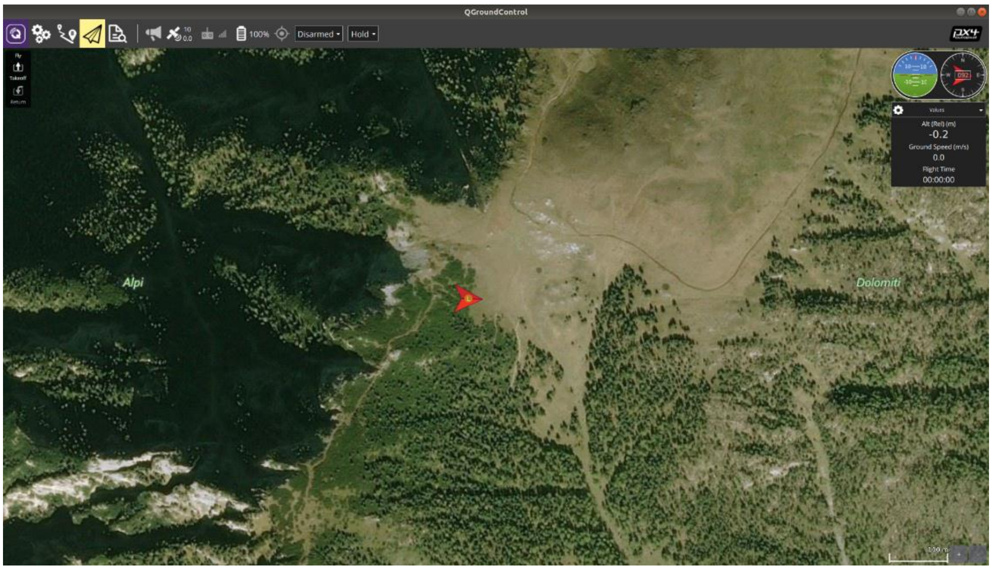


Figure 50: UAV launch positioning on QGC

Secondly, the waypoints positioning has been defined to simulate a monitoring mission of a medium size area, creating a semi-rectangular-shaped route, according to the project's provisions. The mission lasts about 10 minutes, which is on average half the endurance of the designed UAV. This choice has been carried out considering a safety margin for which the vehicle, in case it encounters several obstacles, has enough endurance to be able to recalculate a hypothetical longer route, without the risk of reaching a critical battery value.

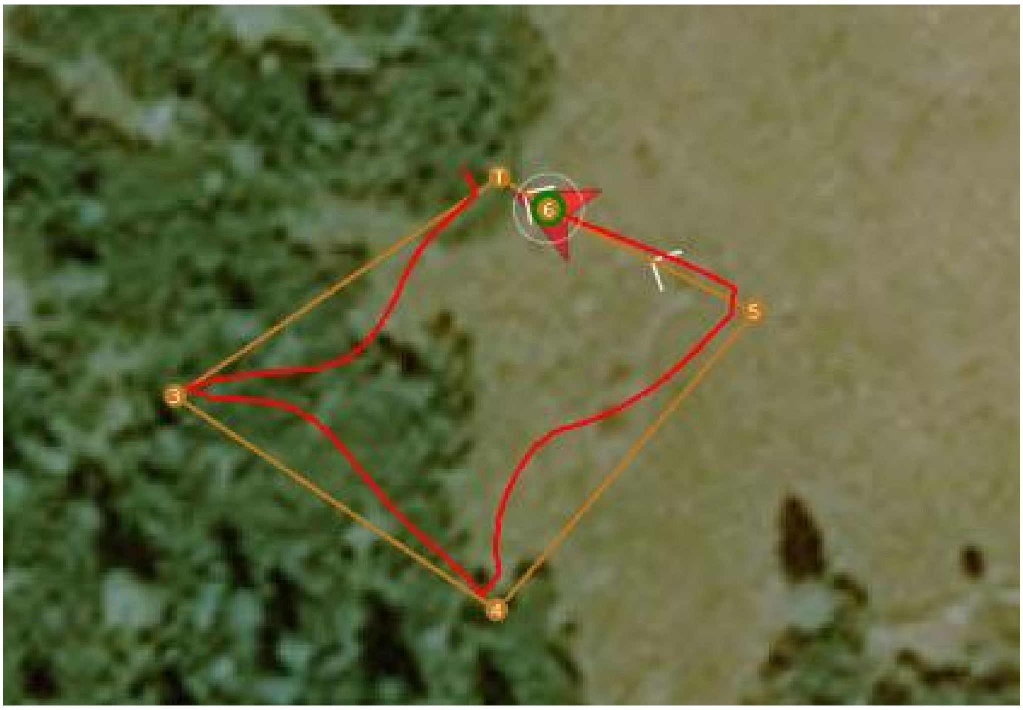


Figure 51: Scenario 2 results - Obstacle avoidance trajectory variations

Figure 51 shows the route designed in the mission planning phase (yellow) and the route actually taken by the UAV (red) thanks to the use of the active collision avoidance system. The first two waypoints identify the launch position and the take-off coordinate (L ant T overlapping points), while waypoints 3,4,5 and 6 identify the geographical coordinates of different areas of the Corvara landslide that the vehicle has to fly over (it is possible to glimpse, from the QGC map in the background of the mission, a green wooded area and a brown area; the latter is a slightly sloping area of the landslide with trees scattered inside).

The mission is carried out at an altitude of about 5 metres for the entire route, so as not to reach altitudes too close to the ground, thus avoiding, in view of the flight missions that will be carried out in the real world, the formation of effects such as the ground effect. On the other hand, altitudes are not set too high to thoroughly test the collision avoidance system and allow the drone to face any type of tree, thus remaining at heights comparable to those of the vegetation.

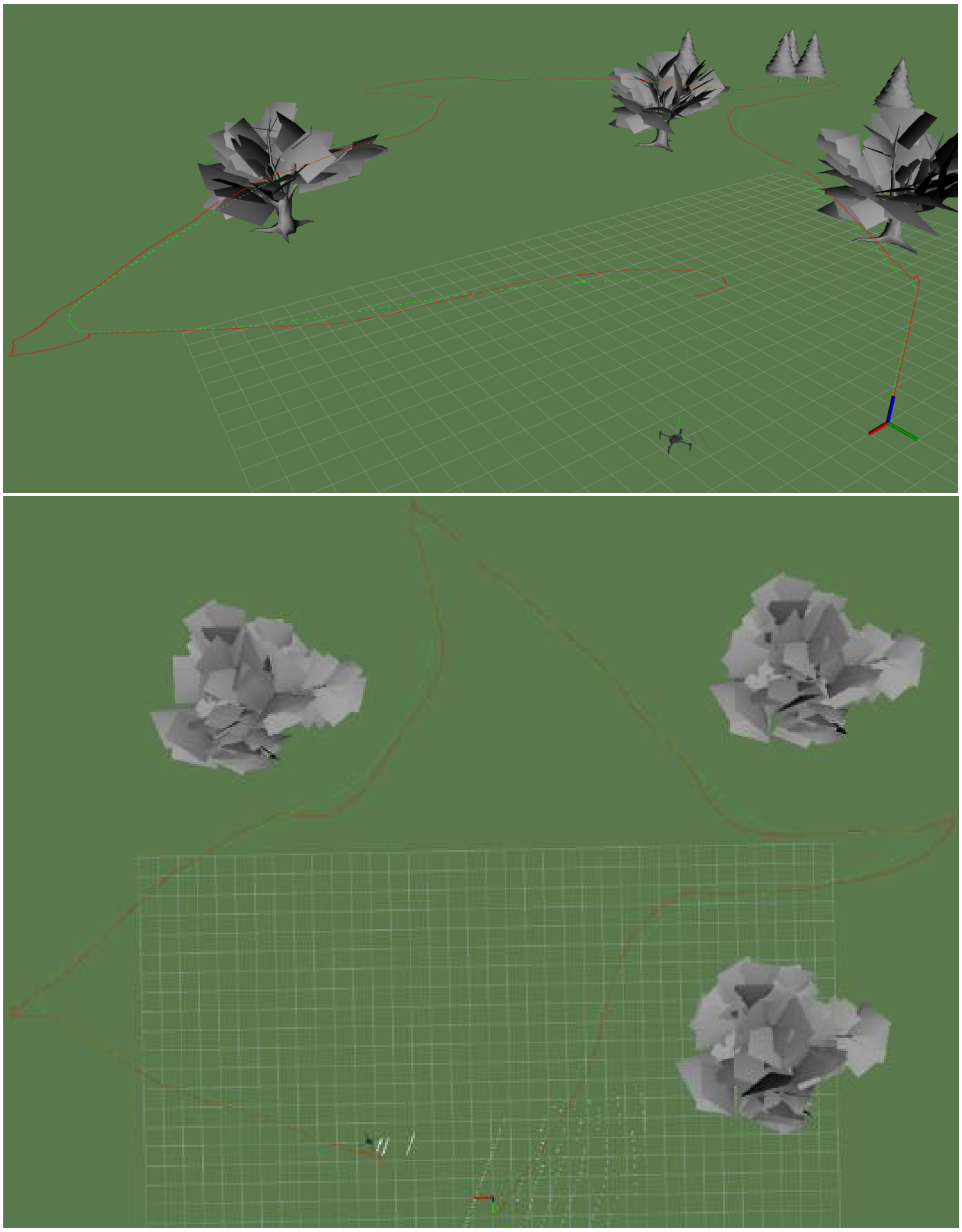


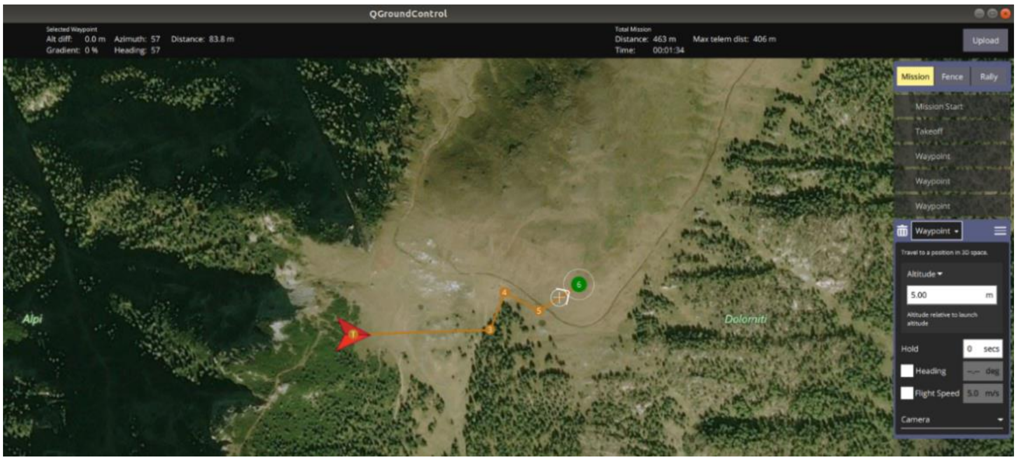
Figure 52: Scenario 2 results - Mission accomplished; 3D view and top view, RViz visualization

The cause of the large gap between the desired path and the actual path is clearly visible in Figure 52: the drone takes off from the vehicle generation point in the scenario (bottom right, where the three-axis reference system is displayed), reaches the second waypoint in vertical ascent and starts the mission towards the third waypoint, which, however, is in the middle of several trees. The collision avoidance system generates the new optimised path (red line in RViz) and the UAV moves avoiding obstacles until it reaches all targeted waypoints (green dotted line). At the end of the mission, a land command has been manually sent and the drone has landed below Waypoint 6. It is noted that the collision avoidance system has not been implemented for the landing phase, indeed, the red line calculation of the optimal path is interrupted and the landing procedure is highlighted only by the green line that indicates the route taken.

Finally, it has been possible to verify how slight unevenness of the terrain do not affect obstacle detection. Indeed, in this simulation, only trees are avoided and there are no variations in altitude or in trajectory due to the slightly inclined terrain. This is due to standard values of some parameters that regulate the sensitivity of the obstacle detection sensor. For example, by modifying them, it is possible to vary the camera detection range and the minimum distance at which the UAV must be at each detected object, including the ground.

4.3 Scenario 3 Simulation

The third set of simulations has been carried out maintaining the geographic coordinates of the landslide site chosen for the project. Instead, the flight mission, with its relative waypoints, has been redesigned in function of the new simulation environment, created with a greater number of trees, disposed closer than the previous scenario and with many portions of steep terrain.



QGroundControl

Selected Waypoint
Alt diff: 0.0 m Azimuth: 57 Distance: 83.8 m
Gradient: 0 % Heading: 57

Total Mission
Distance: 463 m Max telem dist: 406 m
Time: 00:01:34

Travel to a position in 3D space.

Altitude: 5.00 m

Altitude relative to launch altitude

Hold: 0 secs

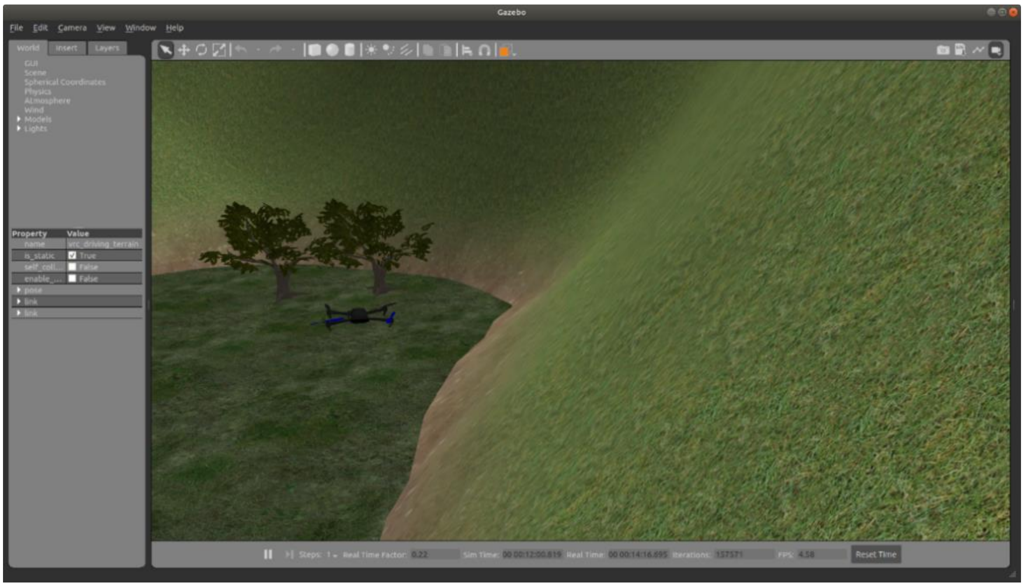
□ Heading: --.-- deg

□ Flight Speed: 5.0 m/s

Figure 53: Scenario 3 - Mission planning phase on QGC

The path shown in Figure 53 has been planned in order to test the maximum performances of the collision avoidance algorithm; for example, waypoints have been placed between trees very close to each other, or very close to obstacles like big trees or steep terrain. It is also shown the actual landslide of Corvara, on the map below the mission and, on the right of the QGC screen, a mission design drop-down list is open, in which it is possible to insert waypoints customizing their features.

Initially, the mission has been developed by inserting waypoints in the most critical positions; Figure 54 shows the UAV flying very close to the semi-vertical walls of the simulated environment. Maintaining the standard flight and avoidance parameters, the mission cannot be completed because the last waypoint has been placed too close to the ground surrounding the valley. Indeed, the vehicle manages to accomplish the mission almost entirely, avoiding the trees in various ways, either by circumventing them while maintaining a constant altitude, or by flying above them and then resuming the flight altitude set by the original route.



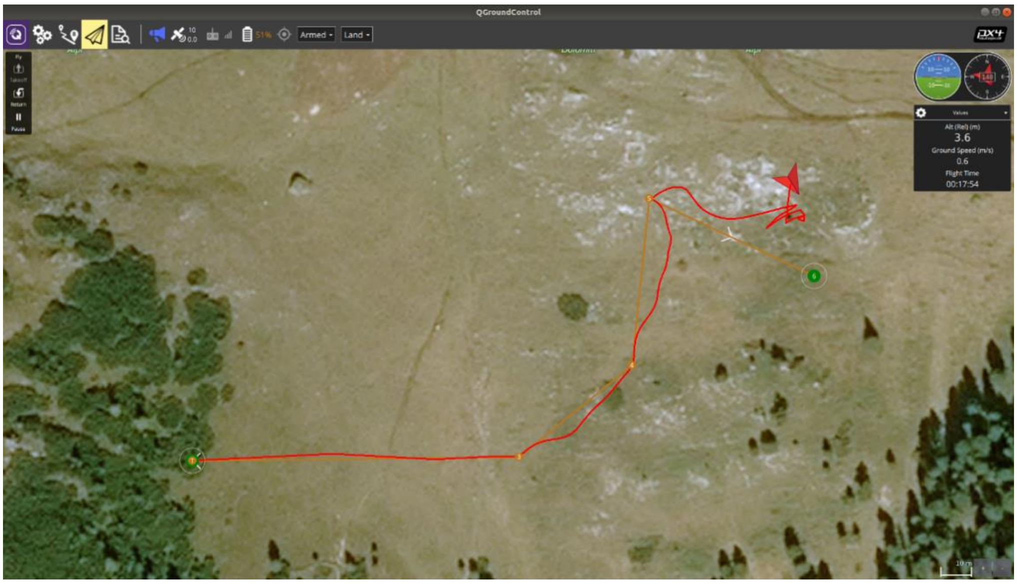
Property	Value
name	vrc_driving_terrain
is_static	☑ True
self_coll...	□ False
enable_...	□ False
pose	
link	
link	

Steps: 1 Real Time Factor: 0.22 Sim Time: 00 00:12:00.819 Real Time: 00 00:14:16.695 Iterations: 157571 FPS: 4.58

Figure 54: Scenario 3 - Rough environment tests, Gazebo visualization

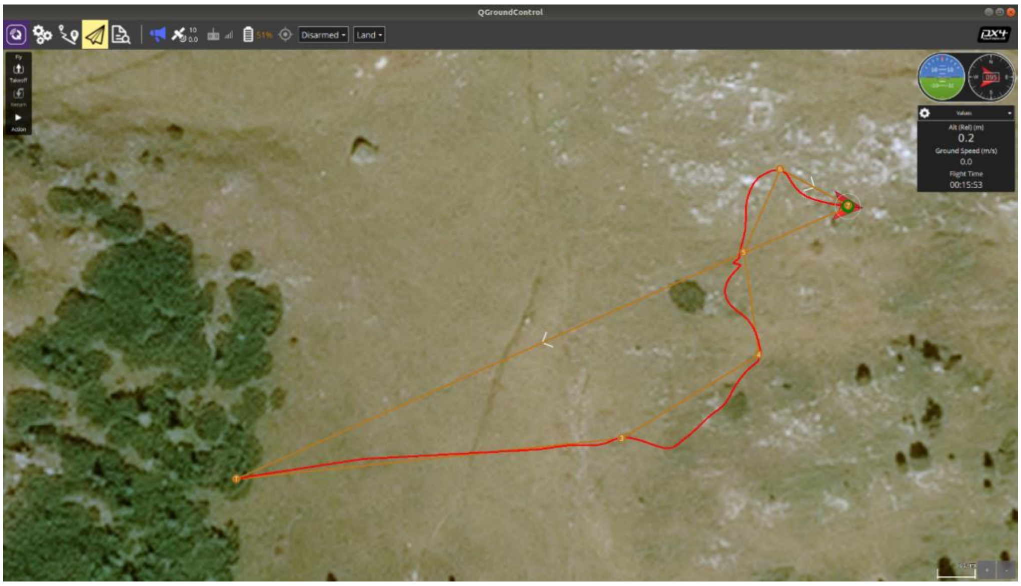
However, once it flies near an inlet in the terrain, it cannot reach the target even after several attempts. The algorithm recalculates the route several times and tries the approach from the right, left and top, but the imposed parameters of distance from the detected obstacles prevent it from completing the mission.

Figure 55 shows a simulation performed with the most critical path. It is evident how the UAV recomputes a route whenever it detects an obstacle in the camera's field of view and attempts to return to the desired route to reach the next target. From Waypoint 5 to Waypoint 6, the vehicle is forced to move away from the desired route to cross the cliffs, but it is unable to get close enough to the target to complete the mission. The various attempts are sometimes overlapping (red line of the final path).



Alt (Rel) (m)
3.6
Ground Speed (m/s)
0.6
Flight Time
00:17:54

Figure 55: Scenario 3 results - Terrain altitude variation, mission not completed

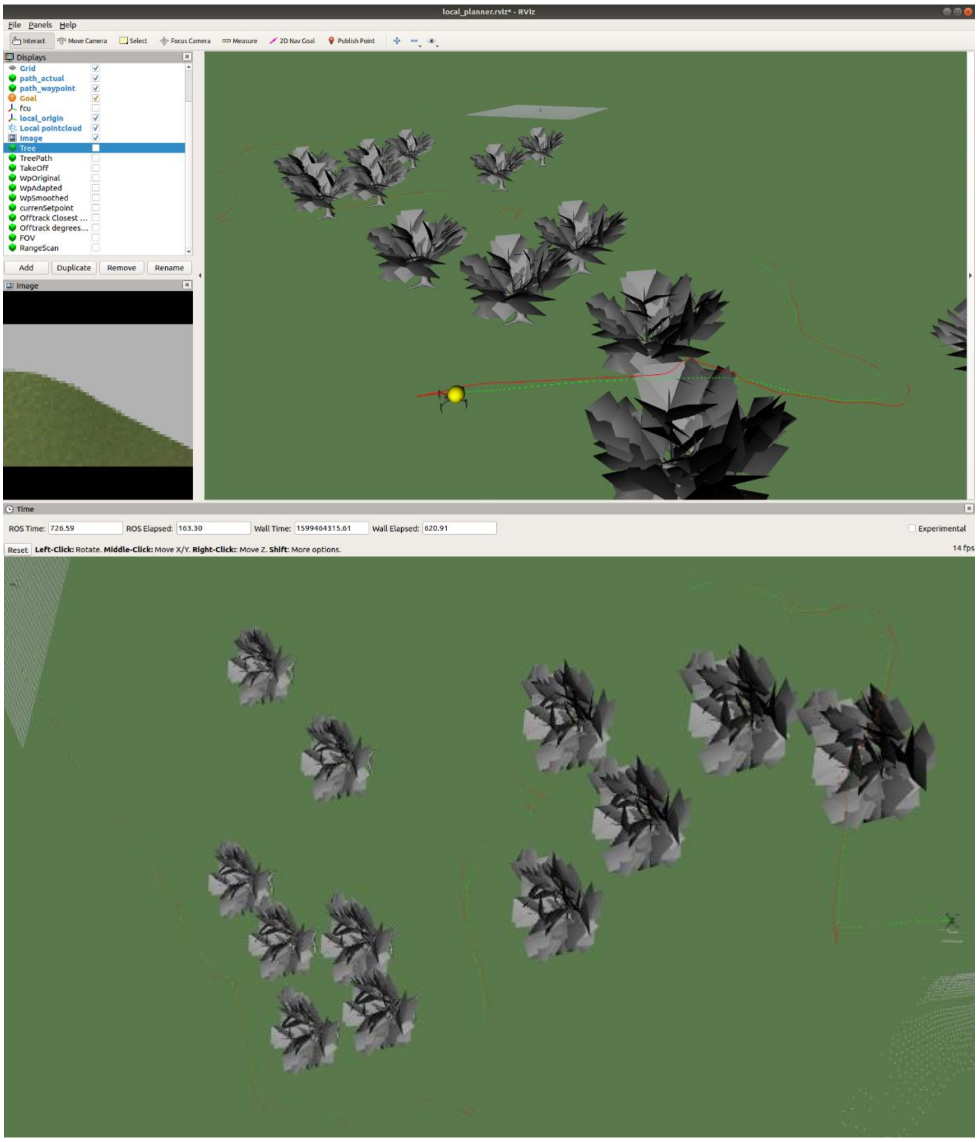


Alt (Rel) (m)
0.2
Ground Speed (m/s)
0.0
Flight Time
00:15:53

Figure 56: Scenario 3 results - Mission accomplished

Later, the mission has been replanned by moving the last waypoints slightly further away from the rocks (Figure 56). In this way, the UAV is able to reach all waypoints on the first attempt, calculating the optimal route continuously and making trajectories at minimum distance radius from the desired path, according to the parameters set.

Figure 56 and Figure 57 demonstrate how the drone moves among obstacles. In RViz, only trees are visualized and not the terrain, but the algorithm considers also altitude variations and all the ground features, such as the difference between grass and rock, during the landing phase, and the physics of all objects.



ROS Time: 726.59 ROS Elapsed: 163.30 Wall Time: 1599464315.61 Wall Elapsed: 620.91

Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click:: Move Z. Shift: More options.

Figure 57: Scenario 3 results - Mission accomplished; 3D view and top view, RViz visualization

RViz is very useful to observe in real time the path computed by the algorithm (red line) to avoid obstacles and the path actually travelled by the drone (green dotted line). Indeed, the algorithm only takes into account the specific parameters for obstacle detection and collision avoidance, but does not considers the flight parameters of the UAV, such as the rate of turns, the ascent and descent speed in relation to turns and so on. Therefore, the autopilot has to manage the incoming commands from the collision avoidance system and combine them with the vehicle dynamics. This is why the two lines differ, especially in turns or in changes of trajectory. The route begins at the centre of the white grid at the top of the 3D view figure, where the reference system is fixed. The path is then articulated in the middle of the vegetation positioned in the narrow valley and ends in the foreground, where the last waypoint (yellow marker) is displayed, just reached by the UAV.

Finally, in this type of simulations, the path has been changed when the vehicle has not been able to complete the mission. The other solution is the variation of the parameters that affect the functioning of the collision avoidance system, while maintaining the same mission. This solution will be evaluated in the following paragraph. It will be studied how the simulation changes by varying the most significant parameters. The aim is to optimize them for the specific study application of this thesis, to make the UAV able to carry out the very different kind of missions, regarding the type of environment treated in the previous paragraphs. In addition, it is interesting to highlight the random component with which the algorithm assigns values to nodes not directly adjacent to the UAV. Indeed, according to Figure 55 and Figure 56, the paths generated for the first four waypoints are slightly different, despite same parameters and scenario have been adopted for these different simulations.

4.4 Optimisation of flight simulation parameters

Starting from general considerations, some standard parameters have been set and maintained constant for all the simulations, which are not modified even in this phase, such as weight, dimensions and endurance of the simulated vehicle. There are some differences between the parameters of the simulated vehicle, consistent with the model displayed in Gazebo and the values that will then have to be considered in future phases of HITL simulations. In fact, the UAV designed in BLUESLEMON project has not been simulated. The table below contains the main parameters that define the simulated vehicle (3DR Iris model) and the comparison with real values of the designed UAV.

	3DR Iris model	**Designed UAV**
Diagonal Wheelbase	550 mm	620 mm
Landing Gear Size (Height)	100 mm	110 mm
Takeoff Weight	1282 g	2800 g
Average Flight time	15 min	24 min

Instead, other UAV-specific parameters will be modified, according to realistic ranges and adapted to the simulated model, to evaluate system performance, such as maximum speed, drag and maximum acceleration. In addition, the empirically chosen parameters, e.g. octomap resolution and the ranges chosen to generate the 2D binary polar histogram, will not be changed in the next paragraphs.

Finally, as far as the sensor simulation is concerned, the only parameter that is considered variable is the so-called imager_rate, which is imposed at a value of 2, but can be varied, always remaining consistent with the real model chosen. Therefore, in the following paragraphs, simulation parameters of the UAV and its sensors will be varied and optimised, keeping fixed the internal parameters that affect the deep functioning of the algorithm.

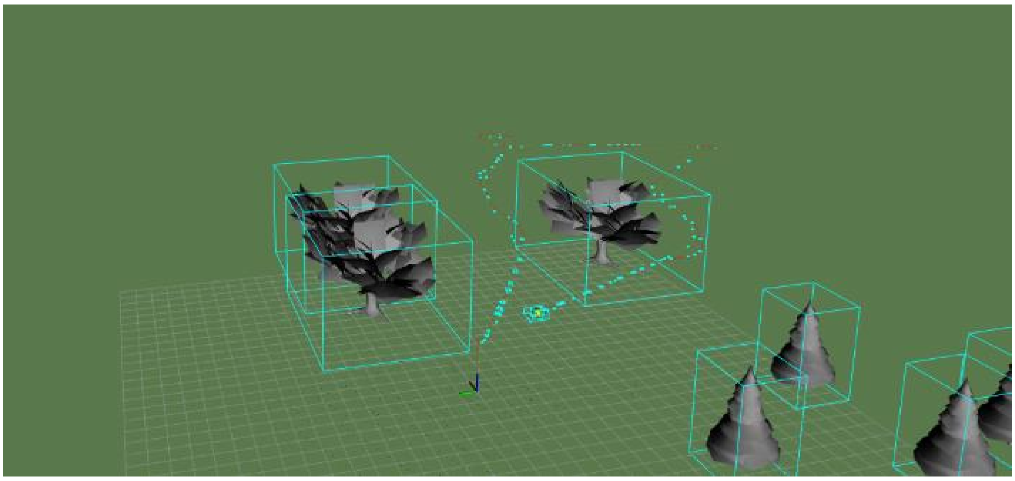
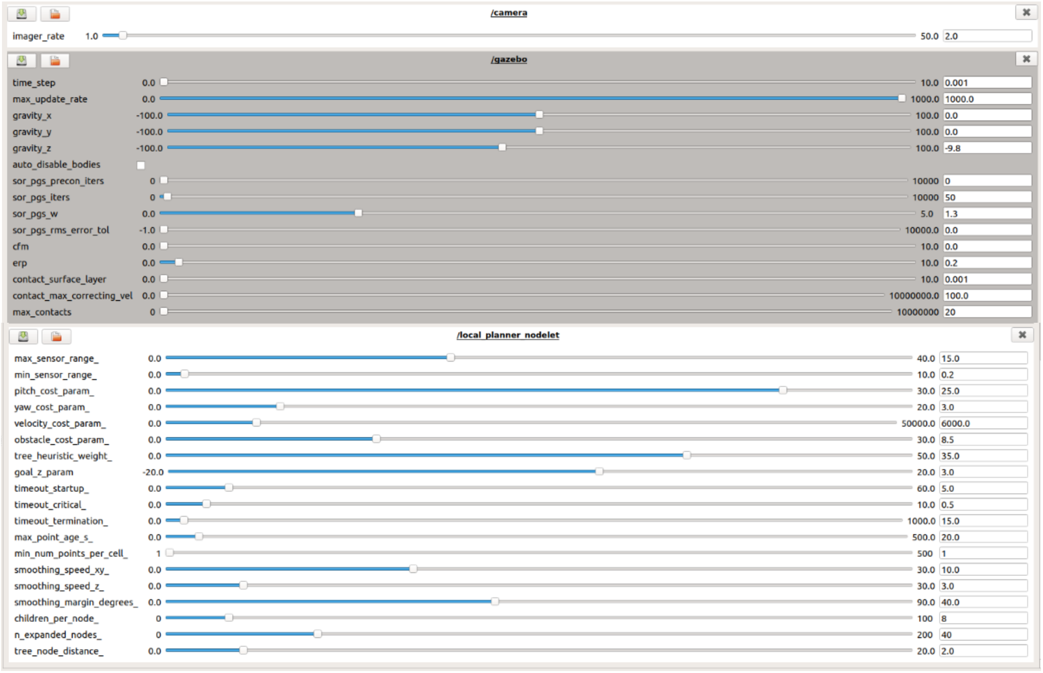


Figure 58: Obstacles and vehicle hitboxes

Figure 58 shows a fundamental concept that affects the performance of the collision avoidance system: the presence of hitboxes. Indeed, each simulated object, except the terrain, is surrounded by a virtual box that defines its physical properties of weight and size.

The different types of hitboxes identify different types of obstacles. In this case, the pine model can be identified as a type of obstacle that is easy to detect and circumvent, as the box is compact and the foliage is uniform. On the other hand, the oak tree, represents a much more difficult obstacle, since the box must contain all the foliage, which however is composed of more jagged and inconstant branches. The drone, also surrounded by a small hitbox, is able to get around the tree, possibly passing close to the hitbox, but, if it exceeds that threshold, it risks getting stuck in the tree or crashing even in the simulation.

Finally, below Figure 59 shows the tool screen with all the main simulation parameters set with the standard values considered and used until this phase, which will be changed and optimised in the following paragraphs.



/camera

imager_rate	1.0	50.0	2.0

/gazebo

time_step	0.0	10.0	0.001
max_update_rate	0.0	1000.0	1000.0
gravity_x	-100.0	100.0	0.0
gravity_y	-100.0	100.0	0.0
gravity_z	-100.0	100.0	-9.8
auto_disable_bodies	□		
sor_pgs_precon_iters	0	10000	0
sor_pgs_iters	0	10000	50
sor_pgs_w	0.0	5.0	1.3
sor_pgs_rms_error_tol	-1.0	10000.0	0.0
cfm	0.0	10.0	0.0
erp	0.0	10.0	0.2
contact_surface_layer	0.0	10.0	0.001
contact_max_correcting_vel	0.0	10000000.0	100.0
max_contacts	0	10000000	20

/local_planner_nodelet

max_sensor_range_	0.0	40.0	15.0
min_sensor_range_	0.0	10.0	0.2
pitch_cost_param_	0.0	30.0	25.0
yaw_cost_param_	0.0	20.0	3.0
velocity_cost_param_	0.0	50000.0	6000.0
obstacle_cost_param_	0.0	30.0	8.5
tree_heuristic_weight_	0.0	50.0	35.0
goal_z_param	-20.0	20.0	3.0
timeout_startup_	0.0	60.0	5.0
timeout_critical_	0.0	10.0	0.5
timeout_termination_	0.0	1000.0	15.0
max_point_age_s_	0.0	500.0	20.0
min_num_points_per_cell_	1	500	1
smoothing_speed_xy_	0.0	30.0	10.0
smoothing_speed_z_	0.0	30.0	3.0
smoothing_margin_degrees_	0.0	90.0	40.0
children_per_node_	0	100	8
n_expanded_nodes_	0	200	40
tree_node_distance_	0.0	20.0	2.0

Figure 59: Parameters variation tool

4.4.1 Scenario 1 parameters optimisation

In the first scenario, thanks to the absence of obstacles, it has been tested how much the ground affects the obstacle detection system. For example, if a cruise altitude of 5 metres is set, but the parameter defining the minimum distance the UAV must maintain from obstacles is set at 10 metres, the algorithm could detect the ground as an obstacle from which the vehicle does not respect the minimum distance. This could represent an enormous limit of operation, defining limitations on the altitude of the UAV, which should always remain greater than the minimum distance parameter. According to the first tests, this behaviour did not take place. The parameter responsible for the minimum distance between UAV and obstacle has been iteratively varied from 3 metres to 15 metres, maintaining a constant flight altitude at 5 metres, and the results show that the flat ground does not affect the recalculation of the optimal route in any way. This parameter is called Obstacle_cost_param; it identifies an approximate distance from obstacles (metres) when the obstacle distance term dominates the cost function. Its standard value is set to 8.5 metres.

Figure 60 shows the result of a series of simulations performed with different values of Obstacle_cost_param. The scenario threated in Paragraph 4.1 has been simulated, in which the terrain altitude variation is used to test the algorithm functioning. The first attempt has been carried out with the standard parameters and the mission has not been completed. The second attempt has been carried out with the parameter set at 3.5 metres and the UAV has been able to complete the mission, even if with several attempts to reach the most critical waypoints. The third simulation has been carried out by imposing a minimum distance from obstacles of only 1.5 metres. The mission was completed with excellent results.

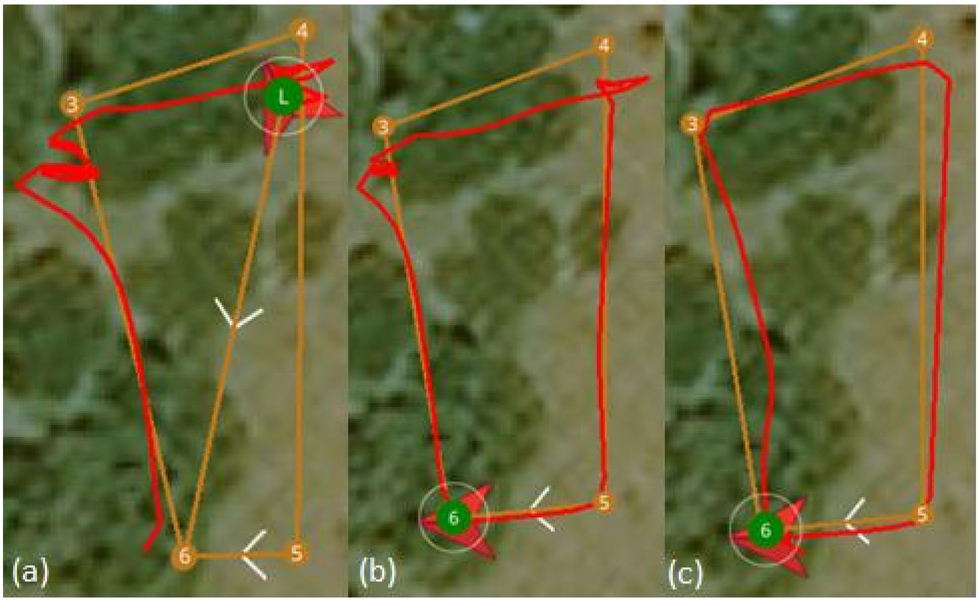


Figure 60: Obstacle distancing parameter varying tests: (a) 8.5 m; (b) 3.5 m; (c) 1.5 m

However, it must be considered that, due to factors of flight safety in the real world, a distance of 1.5 metres from an obstacle can be very dangerous, as far as possible rock walls are concerned, due to the aerodynamic and turbulence effects that can be created, putting the stability of the drone at risk. Furthermore, this specific simulation does not consider obstacles, such as trees, and the risk of hitting branches not detected by the camera is very high. This latter aspect will be analysed more closely in the next results.

Finally, the variation of precision in reaching waypoints is a further aspect that requires an additional evaluation. Among the simulation parameters there is not a precise parameter that identifies the radius within which the waypoint is considered reached, but some flight parameters can be varied to make the drone more manoeuvrable in some conditions. Indeed, when a waypoint is followed by a sudden change in trajectory, the UAV reaches the target with low levels of precision (Figure 61, image (a)), because of some flight limitations. The three parameters that can be changed to solve this problem are the Pitch_cost_param, the Yaw_cost_param and the Velocity_cost_param, to change respectively the pitch, yaw and velocity rate, especially in their maximum values. The values set during the simulation performed are reported in the table below; being cost function weights, they are defined without units of measurement. To obtain greater accuracy in reaching the targets, these three values have been iteratively incremented (image (b) and image (c)). The vehicle can perform manoeuvres most rapidly and with higher turn rates, therefore the waypoints are better reached.

	Range	(a)	(b)	(c)
Pitch_cost_param	0÷30	25	27	30
Yaw_cost_param	0÷20	3	8	15
Velocity_cost_param	0÷50000	6000	22500	40000

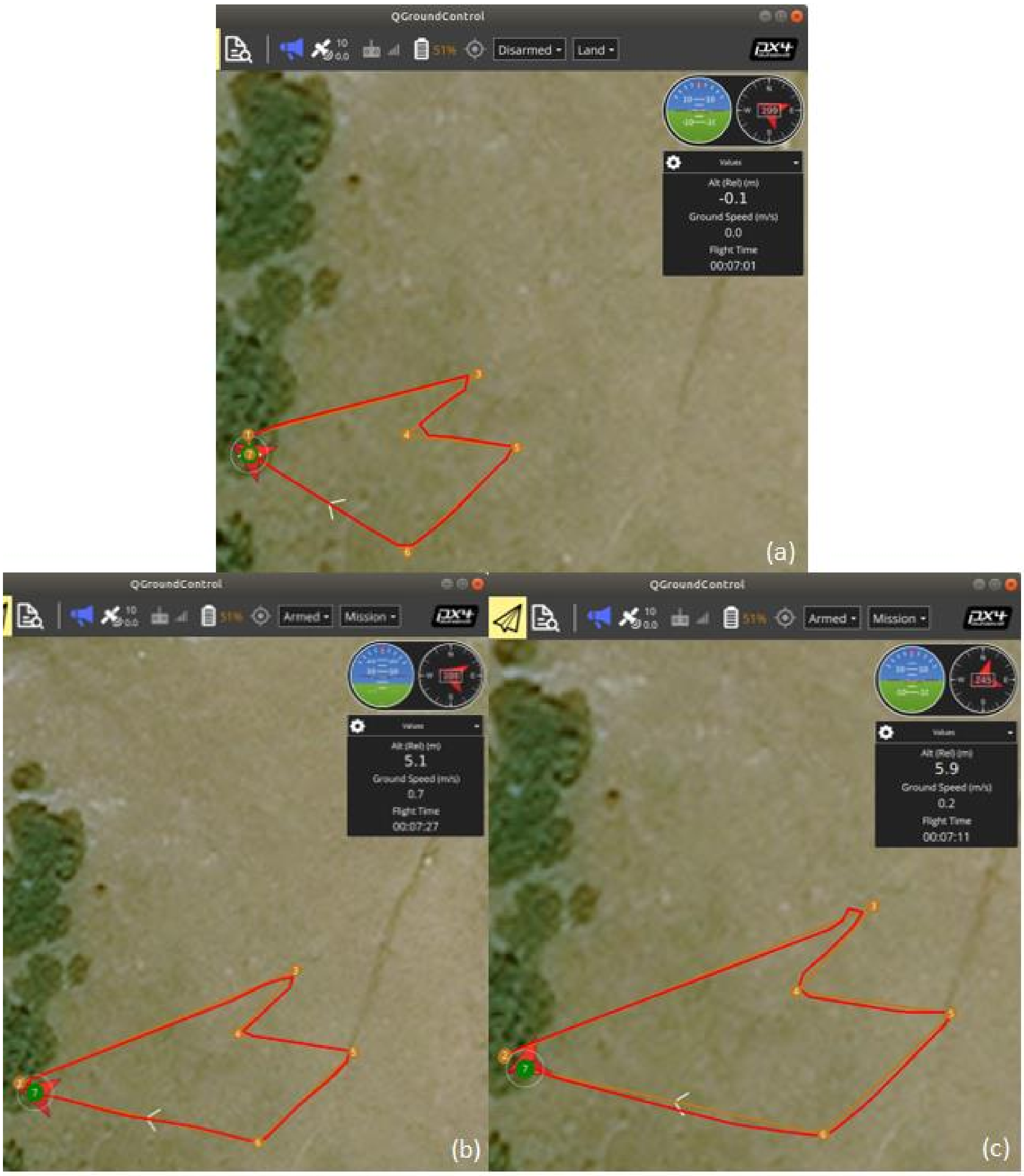


Figure 61: No obstacle simulations – reaching waypoints with greater accuracy

4.4.2 Scenario 2 parameters optimisation

The second scenario has been very useful to evaluate the Obstacle_cost_param parameter and to observe how the definition of the path varies in relation to the decrease in the minimum distance between the UAV and the obstacle. A lower distance results in a route more similar to the desired one, but the risk of collision between the drone and unexpected obstacles increases.

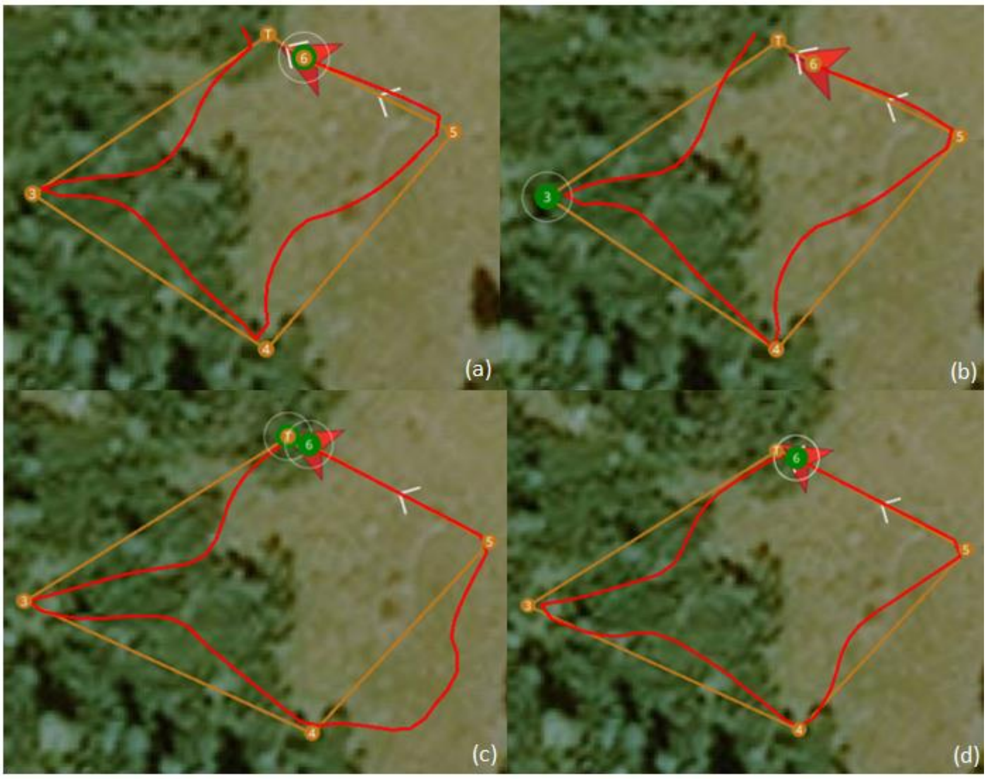
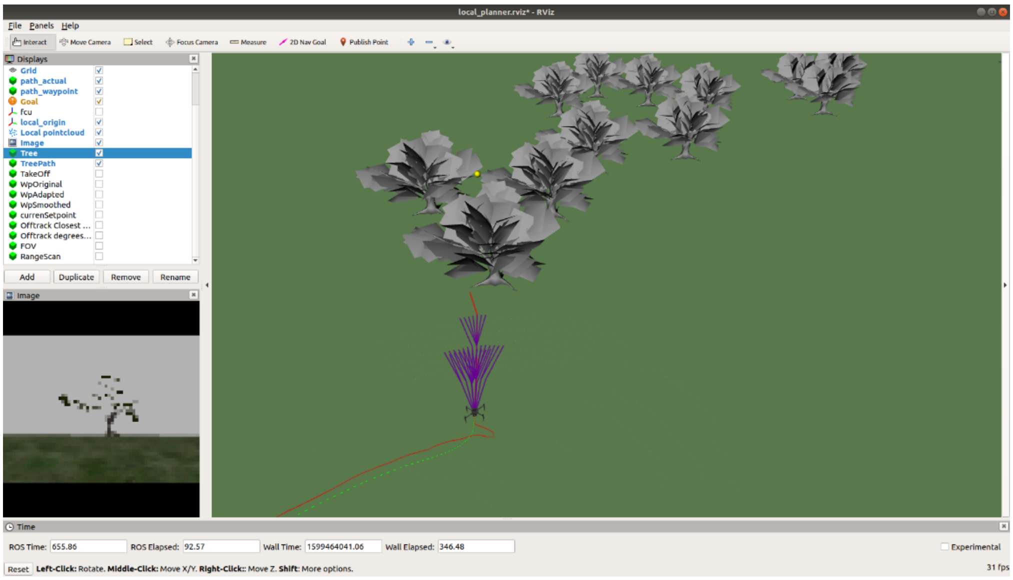


Figure 62: Obstacle distancing parameter varying tests: (a) 10 m; (b) 8.5 m; (c) 5 m; (d) 3 m

The images in Figure 62 are the results of the variation of the above-mentioned parameter, respectively (from left to right, from up to down) for 10, 8.5, 5 and 3 metres as the minimum distance, also called safety radius. As the safety radius decreases, the route will be more and more superimposed on the desired mission path and therefore the functioning of the system is optimised. The value of 3 metres is the minimum value with which the UAV does not run the risk of colliding with portions of obstacle not well detected.

In these simulations the accuracy with which the drone reaches the waypoints has been simultaneously optimised, as explained in the previous paragraph. In addition, another parameter has been evaluated that concerns the maximum operating range of the sensor: Max_sensor_range_ parameter. Increasing the amplitude of the FOV has a double feedback: the sensor has a higher performance, being able to detect many more obstacles, but at the same time the algorithm risks to be severely tested both from a computational point of view and as regards the management of obstacles at very different distances, during the redefinition of the optimal path. For these reasons, with high FOVs the simulation went haywire and performance dropped considerably. By setting the value equivalent to the FOV of the chosen camera model, it has been verified that the choice made during the design phase is appropriate.

Finally, the degree of randomness of the optimal route definition is visible in these simulations, for example for the totally different approach in path redefining between waypoints 4 and 5 in the third image, where the UAV passes on the other side of the desired trajectory, avoiding the tree flying on its right side. The reason of this difference is visualized in Figure 63. New feasible paths are created (purple tree generated from the vehicle) but only one of these solutions is selected as the optimal path to be followed. This decision is made by the algorithm depending on values assigned randomly to nodes not directly adjacent to the UAV.



Grid ☑
path_actual ☑
path_waypoint ☑
Goal ☑
fcu □
local_origin ☑
Local pointcloud ☑
Image ☑
Tree ☑
TreePath ☑
TakeOff □
WpOriginal □
WpAdapted □
WpSmoothed □
currenSetpoint □
Offtrack Closest ... □
Offtrack degrees... □
FOV □
RangeScan □

ROS Time: 655.86 ROS Elapsed: 92.57 Wall Time: 1599464041.06 Wall Elapsed: 346.48 □ Experimental

Figure 63: RViz visualization of possible paths tree

4.4.3 Scenario 3 parameters optimisation

Also in the third scenario, the parameter that regulates the size of the safety radius has been varied, in order to complete the mission. In Figure 64, three simulations are shown: the first is the one carried out with standard parameters already analysed in the previous paragraphs. By setting a minimum safety distance radius of 8.5 metres, the UAV cannot reach the last waypoint of the mission planned, even after several attempts. Therefore, it has been decreased this value iteratively in subsequent simulations.

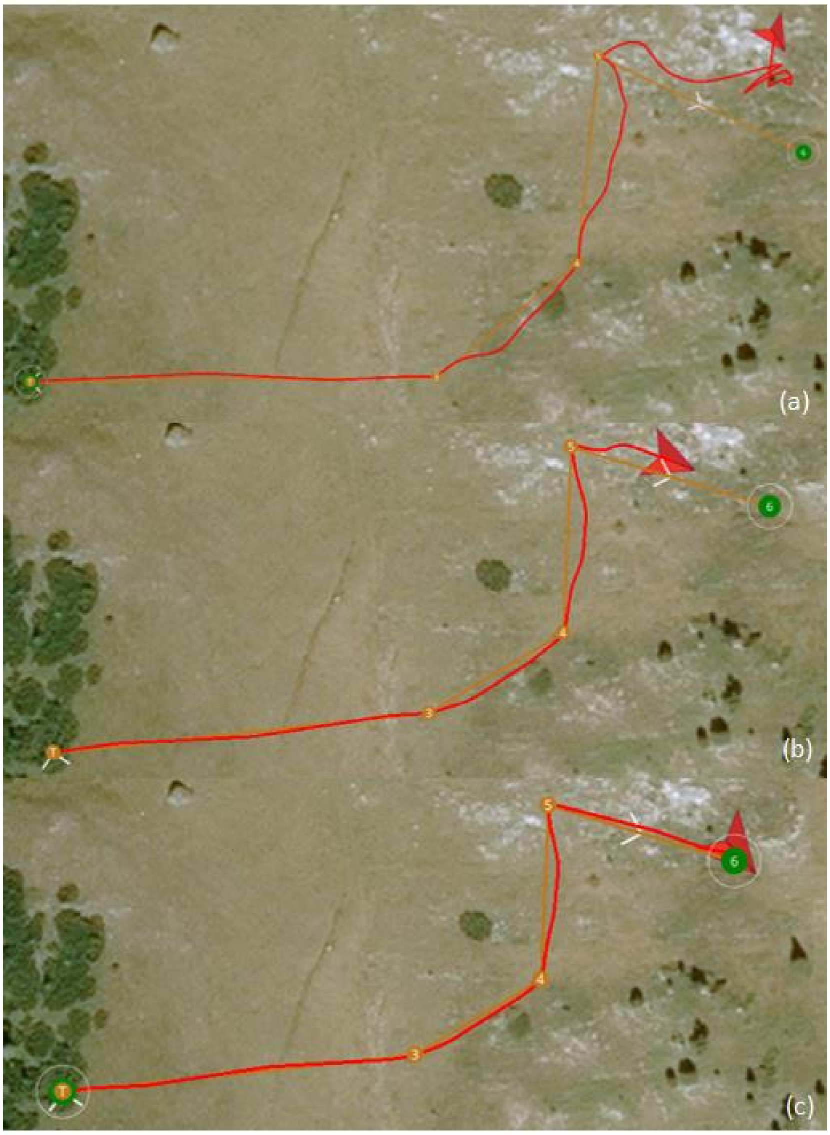
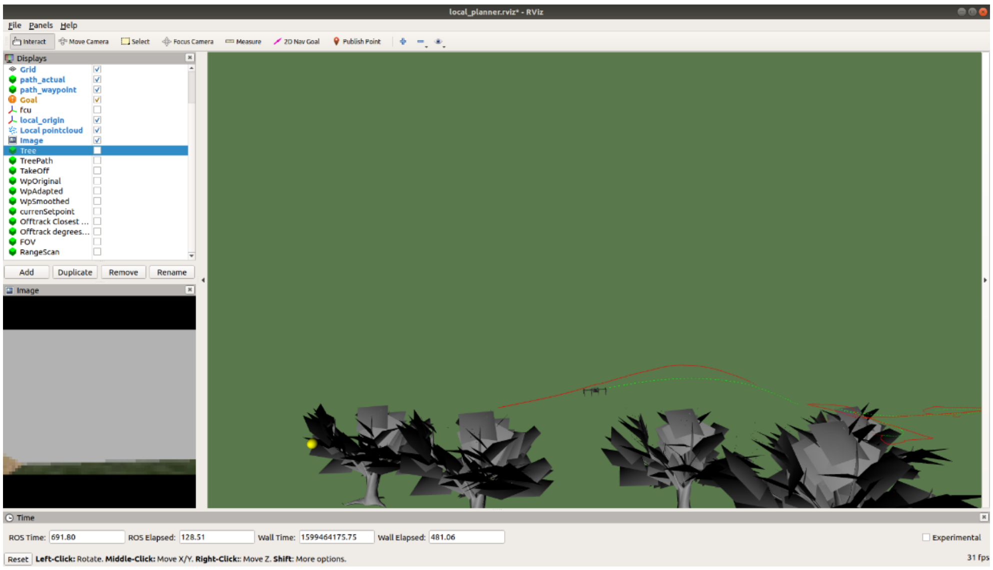


Figure 64: Optimisation of obstacles distancing parameter: (a) 8.5 m; (b) 3 m; (c) 1.5 m

In the second simulation reported in Figure 64, a value of 3 metres has been set, but even in this case the mission has not been completed; it has been aborted before the vehicle makes several attempts to reach the waypoint because it runs into some problems of excessive proximity to a tree. In this respect, the third case is very interesting. The third simulation has been carried out by varying the parameter up to a value of 1.5 metres, in real-time during the execution of the mission. This method has been necessary because, by setting such a small safety radius value from the beginning of the mission, the UAV can't detect all the branches that protrude from the main canopy of trees along the path and consequently it crashes. If, on the other hand, a safety radius of at least 3.5 metres is maintained for the entire area where the UAV passes through trees, it is then possible to modify it only in the last part of the mission, where there are no trees but rock walls. In this way, the mission results completed, but it must be considered that even in the vicinity of rock walls, 1.5 metres of distance can be dangerous. In this case, it would therefore be advisable to modify the mission, maintaining adequate safety parameters for any possible obstacle that the vehicle may encounter along the way.



Grid ☑
path_actual ☑
path_waypoint ☑
Goal ☑
fcu ☐
local_origin ☑
Local pointcloud ☑
Image ☑
Tree ☐
TreePath ☐
TakeOff ☐
WpOriginal ☐
WpAdapted ☐
WpSmoothed ☐
currenSetpoint ☐
Offtrack Closest ... ☐
Offtrack degrees... ☐
FOV ☐
RangeScan ☐

ROS Time: 691.80 ROS Elapsed: 128.51 Wall Time: 1599464175.75 Wall Elapsed: 481.06 ☐ Experimental

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click:**: Move Z. **Shift**: More options. 31 fps

Figure 65: Obstacle avoidance along z axis - RViz visualization

Figure 65 shows an instant of the mission previously described, in which the UAV, finding on the right a very high rock wall (not displayed on RViz) and on the left many dense trees difficult to detect due to the shape and variegated structure, it recalculates the optimal path doing collision avoidance vertically and thus avoiding the trees increasing its altitude and then descend until it reaches the next waypoint displayed in a yellow point.

Finally, in Figure 66, it is still possible to observe the randomness of the algorithm in defining some stretches of the ideal route to be performed. For example, in the route between the last two waypoints or between the first two; the two simulations are performed with the same identical desired route and with identical parameters, but the resulting route performed is different.

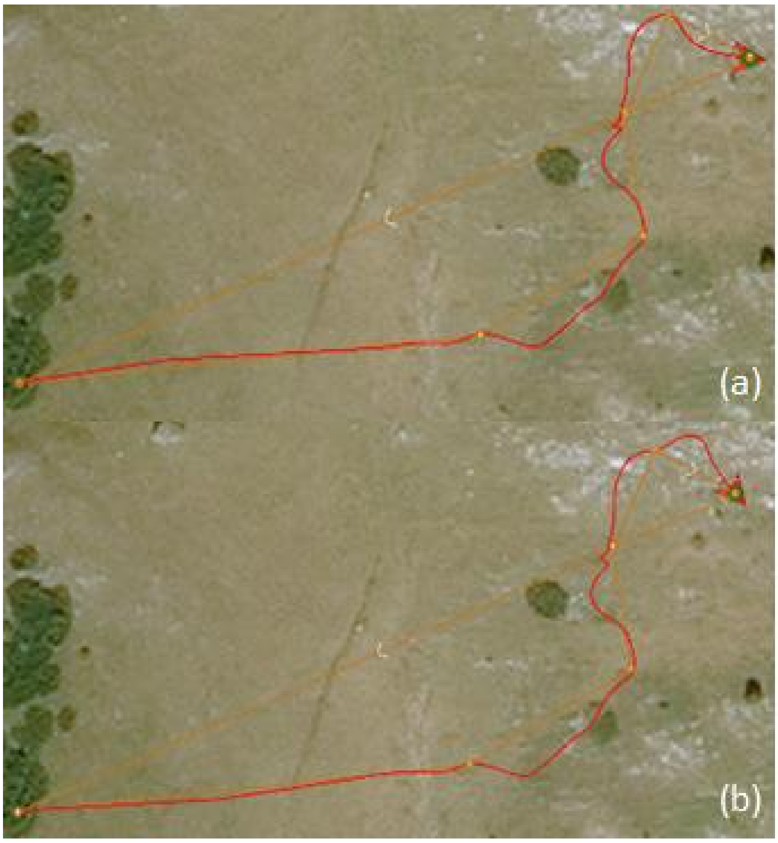


Figure 66: Example results of the randomness degree of the optimal route definition

4.5 Discussion

At the end of the test phase of the algorithm and of the whole simulated system, several observations can be reported. In general, all the simulations led to positive results, including missions not completed due to crashes or to the impossibility to perform critical manoeuvres by the UAV. The latest have been very useful in defining the current limits of the system, giving the cue for possible and useful future implementations. In addition, many simulations have been carried out in each environment examined. One of the main reasons for the high number of simulations has been to verify that the random behaviour of the optimal route calculation could not lead to negative factors or critical results. For all the simulations performed this never happened, therefore, statistically the route recalculation mechanism is safe. The level of safety can be varied by modifying the flight and simulation parameters of the system. For the three main simulation environments used in this dissertation, a general assessment about the system performances will be summarised below.

The unobstructed environment has been fundamental to verify that the system do not continuously interact with the desired route even when no obstacle have been detected. It has been also possible to understand how and when the algorithm come into action, depending on the type of terrain around the UAV. A type of simulation such as those carried out in this environment is also useful to study the reliability of the system and its general behaviour, being able to detect possible operating errors or bugs in the algorithm.

The testing of the system under the condition as similar as possible to the real-world scenario of the BLUESLEMON project has been carried out in the second simulation environment. Most of the simulation results in this scenario can be considered excellent, because in all the case tested the UAV is able to complete the entire mission reaching all the waypoints and avoiding obstacles continuously and without the need to try several times new approaches of overcoming the obstacle. However, it has been evident in this type of simulation that it would be very useful to expand the algorithm operation to all the other phases of the mission, to guarantee obstacle avoidance even in phases such as take-off and landing.

The third simulation environment has tested the designed system the very deeply. For critical situations, in fact, the algorithm worked properly but with average performances lower than the previous scenario. The purpose of the most critical scenario has been precisely to verify the limits of the algorithm performance and most of the missions have been completed even in critical conditions. However, it is of great interest for the future implementation in the real world, to guarantee high performances also in cases of critical environment, for example by implementing an automated variation of the most significant parameters during the execution of missions in real-time, to accomplish different kind of flight missions.

CHAPTER 5: Conclusions and Future Work

This thesis has been dedicated to design and simulate a system for a quadrotor UAV, capable of performing obstacle detection and collision avoidance during the accomplishment of automatic missions. Several design phases have been carried out to obtain a complete system that could be tested in a simulation environment.

Following the study of the state of the art, a three-dimensional extension of the Vector Field Histogram method has been selected, as it is able to guarantee a reactive collision avoidance in real-time, with performance suitable for the specific application. Other 3D methods might have been appropriate, but more experimental research is still necessary. Within this dissertation, it has been also necessary to consider the BLUESLEMON project business approach, such as timing and risk assessment, so it has been considered more appropriate to use an already established method, which was to be adapted and optimized. Moreover, company policy leads to a compromise between better choices of timing and risk.

The chosen method conveyed the search for an algorithm that would allow the method to be applied on a hardware and on a simulated system. The 3DVFH+ algorithm has been adopted and optimised to achieve the objective. It is a method with low computational cost for ground detection that can use data from a single sensor only, for example, as the dissertation application, a forward-facing camera. In this regard, a custom-build UAV has been designed for the project and the Guidance, Navigation and Control system has been designed and adapted to manage the collision avoidance algorithm.

The entire system has been subsequently tested in the Gazebo simulator. 3D simulations with realistic physics, carried out in simulated environments with different characteristics, highlighted some operating features of the system. The simulation experiments suggest a great potential of the algorithm for complex environments, where performances benefit from the look-ahead functionality. More specifically, the detection and avoidance method is able to maintain a minimum safety distance to obstacles, continuously redefining the optimal path when obstacles are detected near the vehicle.

By analysing the test results and evaluating the overall thesis work, several possible future implementations and extensions of the system potential have been evaluated.

The 3DVFH+ algorithm offers various possibilities for further improvements, such as about the UAV dynamics. Furthermore, as far as system simulation is concerned, it is of interest to simulate the specific drone model developed in the project, importing the original CAD in the simulation environment and providing the physics and sensor features to the simulated UAV, as already done with the not-specific simulated vehicle in this thesis. Moreover, the development of specific simulations to emulate obstacles appearing in motion would allow to better test the reactive and active functionality of the algorithm.

Secondly, the collision avoidance system could be extended and tested with the use of additional sensors. It is interesting, for example, to evaluate and compare the use of several sensors at the same time to ensure a higher vehicle FOV for the obstacle detection, or hybrid solutions, with different types of sensors, to evaluate additional detection functionalities. Indeed, the 3DVFH+ performance is potentially limited by the available sensor information. For this study, the algorithm has been implemented on a relatively simple hardware. Furthermore, the safety and the flight speed could be increased using additional sensors. For these improvements it is necessary to evaluate the computational potential of the companion computer on board the designed UAV, which must be able to support the use of multiple sensors.

The main future steps for the project will also be the development of Hardware In The Loop (HITL) simulations, with which it will be possible to test different hardware components without flying the drone, such as the autopilot, companion computer, GPS, depth camera and their mutual communications. Finally, future works will focus on improving the reliability of the system and carrying out the same tests and simulations in the real word, testing the complete real system developed.

Acknowledgments

Thank you Professor Guglieri for giving me this fantastic opportunity and, together with Gianluca, for always being supportive. Thank you MAVTech, for being an incredible team. Alex and Fabio, the time has come to say: Aprite Pure!

Thank you **M** & **P** for always supporting and motivating me, you are very precious. Thank you **A** for being an amazing sister and a powerful role model.

Thank you **E** for being my world, my beloved Cret, my favourite teammate.

Thank you **G** & **S** for being true, incredible friends, even from a distance. Forever and ever.

Thank you **F** for our talks, the most interesting in the last 418 years.

Thank you **I** for being the best Tremun of the world and for encouraging me to write my thesis to the sound of parties. Together with **V**, **J** and **D**, thank you for making me feel at home from the first day.

Thank you, all my friends, to be part of my life every day and to fill my heart with happiness.

Thank you **E** for having got it, not always but mostly with a smile. Good girl, keep it up and always remember: Forza e coraggio, all’arrembaggio.

♡

Bibliography

1. *Collision Avoidance of a Mobile Robot for Moving Obstacles Based on Impedance Force Control Algorithm.* **Eun Soo Jang, Seul Jung, T. C. Hsia.** s.l. : IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005.

2. *A Survey on Unmanned Aerial Vehicle Collision Avoidance Systems.* **Hung Pham, Scott A. Smolka, Scott D. Stoller, Dung Phan, Junxing Yang.** Stony Brook, NY, USA : Stony Brook University.

3. *Quadcopter Control System.* **Daniel Gheorghita, Ionut Vintu, Letitia Mirea, Catalin Braescu.** 2015, 2015 19th International Conference on System Theory, Control and Computing (ICSTCC), pp. 421-426.

4. *Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor.* **Robert Mahoney, Vijay Kumar, Peter Corke.** s.l. : IEEE Robotics & Automation Magazine, 2012, Vol. 19, pp. 20-32.

5. *Backstepping and sliding-mode techniques applied to an indoor micro quadrotor.* **Samir Bouabdallah, Roland Siegwart.** s.l. : In Proceedings of IEEE Int. Conf. on Robotics and Automation, 2005, pp. 2247–2252.

6. *Obstacle avoidance approaches for quadrotor uav based on backstepping technique.* **Qingbo Geng, Qiong Hu Huan Shuai.** s.l. : 5th Chinese Control and Decision Conference, 2013, pp. 3613–3617.

7. *Full control of a quadrotor.* **Samir Bouabdallah, Roland Siegwart.** s.l. : IEEE/RSJ Int. Conference on Intelligent Robots and Systems, 2007.

8. *Minimum-time approach to obstacle avoidance.* **Prasad, Jongki Moon and J.V.R.** s.l. : Mechatronics, 2011, Vol. 21, pp. 61—875.

9. *UAV collision avoidance based on geometric approach.* **Jung-Woo Park, Hyon-Dong Oh, Min-Jea Tahk.** s.l. : SICE Annual Conference, 2008, pp. 2122-2126.

10. *Obstacle Avoidance Based-Visual Navigation for Micro Aerial Vehicles.* **Wilbert G. Aguilar, Verónica P. Casaliglla, José L. Pólit.** s.l. : electronics, 2017.

11. *Cooperative Sense and Avoid Implementation in Simulation and Real World for Small Unmanned Aerial Vehicles.* **Armin Strobel, Marc Schwarzbach.** s.l. : Research Training Group Cooperative, Adaptive and Responsive Monitoring in Mixed Mode Environments.

12. *UAV collision avoidance using cooperative predictive control.* **Boivin E., Desbiens A., Gagnon E.** s.l. : Control and Automation, June 2008, Vol. 16th Mediterranean Conference, pp. pp.682-688.

13. *Visual Flight Rules-Based Collision Avoidance Systems for UAV Flying in Civil Aerospace.* **Hamid Alturbeh, James F. Whidborne.** s.l. : robotics, MDPI, 2020.

14. *Comparative Study of Collision Avoidance Techniques for Unmanned Aerial Vehicles.* **Alexopoulos A., Kandil A., Orzechowski P., Badreddin E.** s.l. : Systems, Man, and Cybernetics (SMC), October 2013, Vol. 2013 IEEE International Conference, pp. pp.1969-1974.

15. *Real-time obstacle avoidance for manipulators and mobile robots.* **Khatib, Oussama.** 1, s.l. : The International Journal of Robotics Research, 1986, Vol. 5, pp. 90-98.

16. *UAV trajectory design using total field collision avoidance.* **Sigurd K., How J.** Austin, TX, USA : In Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit, August 2003.

17. *High-speed obstacle avoidance for mobile robots.* **Johann Borenstein, Y. Koren.** s.l. : Proceedings IEEE International Symposium on Intelligent Control, 1988, pp. 382–384.

18. *UAS collision avoidance algorithm minimizing impact on route surveillance.* **F.G, Smith A.L. Harmon.** Chicago, IL, USA : AIAA Guidance, Navigation, and Control Conference and Exhibit, 2009.

19. *Potential field methods and their inherent limitations for mobile robot navigation.* **Koren, Johann Borenstein and Y.** s.l. : IEEE Conference on Robotics and Automation, 1991, pp. 1398–1404.

20. *The vector field histogram - fast obstacle avoidance for mobile robots.* **J. Borenstein, Y. Koren.** 3, s.l. : IEEE Transaction on Robotics and Automation, 1991, Vol. 7, pp. 278 – 288.

21. *Obstacle Avoidance.* **Ribeiro, Maria Isabel.** s.l. : Navigation/Collision Avoidance, 2005.

22. *Histogramic in-motion mapping for mobile robot obstacle avoidance.* **J. Borenstein, Y. Koren.** 4, s.l. : IEEE Transaction on Robotics and Automation, 1991, Vol. 7, pp. 535 – 539.

23. *Vfh+: Reliable obstacle avoidance for fast mobile robotics.* **Johann Borenstein, Iwan Ulrich.** s.l. : In Proceedings of the International Conference on Robotics and Automation, 1998, Vol. 2, pp. 1572-1577.

24. *Vfh*: Local obstacle avoidance with look-ahead verification.* **Ulrich, Johann Borenstein Iwan.** s.l. : In Proceedings of the IEEE International Conference on Robotics and Automation, 2000, Vol. 3, pp. 2505-2511.

25. *Nearness diagram navigation (nd): A new real time collision avoidance approach.* **Javier Minguez, Luis Montano.** s.l. : In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2000, pp. 2094–2100.

26. *Nearness Diagram (ND) Navigation: Collision Avoidance in Troublesome Scenarios.* **Javier Minguez, Luis Montano.** 1, s.l. : IEEE Transactions on robotics and automation, 2004, Vol. 20.

27. *The curvature-velocity method for local obstacle avoidance.* **Simmons, Reid.** s.l. : In Proceedings of the 1996 IEEE International Conference on Robotics and Automation, 1996, pp. 3375–3382.

28. *Robust obstacle avoidance in unknown and cramped environments.* **W. Feiten, R. Bauer, G. Lawitzky.** s.l. : In Proceedings of the IEEE International Conference on Robotics and Automation,, 1994, pp. 2412–2417.

29. *Dynamic window based approach to mobile robot motion control in the presence of moving obstacles.* **Marija Seder, Ivan Petrovic.**

30. *The dynamic window approach to collision avoidance.* **Dieter Fox, Wolfram Burgard, Sebastian Thrun.** 1, s.l. : IEEE Robotics & Automation Magazine, 1997, Vol. 4, pp. 23-33.

31. *A survey of collision avoidance approaches for unmanned aerial vehicles.* **B.M. Albaker, N.A. Rahim.** Technical Postgraduates (TECHPOS) : International Conference, Dec 2009, pp. 1-7.

32. *Predictive potential field-based collision avoidance for multicopters.* **Matthias Nieuwenhuisen, Mark Schadler, Sven Behnke.** s.l. : Proc. of the 2nd Conference on Unmanned Aerial Vehicles in Geomatics, 2013.

33. *Obstacle detection and collision avoidance for a uav with complementary low-cost sensors.* **Nils Gageik, Paul Benz, Sergio Montenegro.** s.l. : IEEE Access, 2015, Vol. 3, pp. 599–609.

34. *Flying fast and low among obstacles.* **Sebastian Scherer, Sanjiv Singh, Lyle Chamberlain, Srikanth Saripalli.** s.l. : In International Conference on Robotics and Automation (ICRA), 2007, p. 2023.

35. *Obstacle avoidance for unmanned air vehicles using optical flow probability distributions.* **Paul Merrell, Dah jye Lee, Al Beard.** 5609, s.l. :

Mobile Robots XVII, 2004, pp. 13–22.

36. *Biologically inspired uav obstacle avoidance and control using monocular optical flow & divergence templates.* **John Stowers, Michael Hayes, Andrew Bainbridge-Smith.** s.l. : Proceedings of the 5th International Conference on Automation, Robotics and Applications, 2011, pp. 378–383.

37. *A survey of optical flow techniques for uav navigation applications.* **Haiyang Chao, Yu Gu, Marcello Napolitano.** Grand Hyatt Atlanta, Atlanta, GA : International Conference on Unmanned Aircraft Systems, 2013, pp. 710–716.

38. *A biomimetic reactive navigation system using the optical flow for a rotary-wing uav in urban environment.* **Laurent Muratet, Stéphane Doncieux, Jean arcady Meyer.** s.l. : Proceedings of ISR, 2004.

39. *A contribution to vision-based autonomous helicopter flight in urban environments.* **Laurent Muratet, Stephane Doncieux, Yves Briere, Jean arcady Meyer A, Place Emile Blouin.** s.l. : Robotics and Autonomous Systems, 2005, Vol. 50, pp. 195–209.

40. *Safe teleoperation of a quadrotor using fastslam.* **João Mendes, Rodrigo Ventura.** s.l. : IEEE International Symposium on Safety, Security, and Rescue Robotics, 2012, pp. 1-6.

41. *Real-time Obstacle Avoidance and Mapping for AUVs Operating in Mapping for AUVs Operating in.* **Jacques C. Leedekerken, John J. Leonard, Michael C. Bosse, Arjuna Balasuriya.** s.l. : Massachusetts Institute of Technology.

42. *Decentralized Navigation of Multiple Agents Based on ORCA and Model Predictive Control.* **Hui Cheng, Qiyuan Zhu, Zhongchang Liu, Tianye Xu, Liang Lin.**

43. *Reciprocal n-body Collision Avoidance.* **Jur van den Berg, Stephen J. Guy, Ming Lin, and Dinesh Manocha.**

44. *LSwarm: Efficient Collision Avoidance for Large Swarms with Coverage Constraints in Complex Urban Scenes.* **Senthil Hariharan Arul, Adarsh Jagan Sathyamoorthy, Shivang Patel, Michael Otte, Huan Xu, Ming C Lin, Dinesh Manocha.** 2019.

45. *Obstacle avoidance approaches for quadrotor uav based on backstepping technique.* **Qingbo Geng, Qiong Hu Huan Shuai.** s.l. : 25th Chinese Control and Decision Conference, 2013, pp. 3613–3617.

46. *Minimum-time approach to obstacle avoidance constrained by envelope protection for autonomous uavs.* **Jongki Moon, J.V.R. Prasad.** 5, s.l. : Mechatronics, 2011, Vol. 21, pp. 861-875.

47. Micro Aerial Vehicle Technologies. *Mavtech.* [Online] https://www.mavtech.eu.

48. Dronecode foundation. *the linux foundation.* [Online] 2019. https://docs.px4.io/v1.9.0/en/flight_controller/pixhawk4.html .

49. Dronecode. [Online] PX4 Dev Team. https://docs.px4.io/master/en/power_module/holybro_pm07_pixhawk4_power_module.html.

50. Proficnc - Hex. [Online] http://www.proficnc.com/content/12-here.

51. *Drotek store.* [Online] Drotek Electronics. https://store-drotek.com/911-sirius-rtk-gnss-rover-f9p.html .

52. Herelink store. [Online] https://www.onedrone.com/store/herelink-hd-video-transmission.

53. Lightware products. [Online] https://lightwarelidar.com/products/sf11-c-120-m .

54. Hawkeye navigation camera store. [Online] https://it.banggood.com/Hawkeye-Firefly-Split-4K-160-Degree-HD-Recording-DVR-Mini-FPV-Camera-WDR-Single-Board-Built-in-Mic-Low-Latency-TV-Output-for-RC-Drone-Airplane-p-1528048.html .

55. RPLidar store. *Slamtech.* [Online] https://www.slamtec.com/en/Lidar/A3 .

56. ROS. *RPLidar results.* [Online] https://answers.ros.org/question/206185/why-rplidar-gmapping-has-a-bad-result/ .

57. **H. Alvarez, L.M. Paz, J. Sturm, D. Cremers**. Collision Avoidance for Quadrotors with a Monocular Camera.

58. Stereolabs. *ZED camera .* [Online] https://www.stereolabs.com/zed/.

59. Intel Reasense D435i. [Online] https://www.intelrealsense.com/depth-camera-d435i/.

60. Diydrones. *Intel Realsense D435 .* [Online]

https://diydrones.com/profiles/blogs/first-flight-intel-realsense-d435-on-jetson-tx2-uav.

61. Nvidia . *Jetson NANO store.* [Online] https://www.nvidia.com/it-it/autonomous-machines/embedded-systems/jetson-nano/.

62. Airframes. *PX4.* [Online] https://docs.px4.io/master/en/airframes/airframe_reference.html .

63. ProfiCNC products. [Online] http://www.proficnc.com/all-products/211-gps-module.html .

64. *3DVFH+: Real-Time Three-Dimensional Obstacle Avoidance Using an Octomap.* **Simon Vanneste, Ben Bellekens, Maarten Weyn**. Paardenmarkt 92, B-2000 Antwerpen : CoSys-Lab, Faculty of Applied Engineering.

65. GitHub. *Obstacle Detection and Avoidance.* [Online] https://github.com/PX4/avoidance.

66. **Baumann, Tanja**. Obstacle Avoidance for Drones Using a 3DVFH* Algorithm. *Master Thesis.* 2018.

67. ROS - Robot Operating System. [Online] https://www.ros.org/about-ros/.

68. Gazebo Simulator. [Online] http://gazebosim.org/.

69. *Iris Quadcopter.* [Online] http://www.arducopter.co.uk/iris-quadcopter-uav.html.

70. *TerraXcube.* [Online] eurac research, 2020. https://terraxcube.eurac.edu/.

71. eurac research. *Large Cube - technical sheet.* s.l. : terraXcube, 2020.

Appendix A: BLUESLEMON Project Overview

BLUESLEMON
BT Beacon and Unmanned Aerial System technologies for Landslide Monitoring

The BLUESLEMON project (BT Beacon and Unmanned Aerial System technologies for Landslide Monitoring) foresees the creation of a system for the automatic monitoring of landslide areas, based on the following technologies: Bluetooth (BT) Beacon, RFID and a Multi-Purpose Unmanned Aerial Vehicle (UAV). The objective of the project is to increase the ability to prevent landslide damage by a ubiquitous and pervasive monitoring of landslide areas.

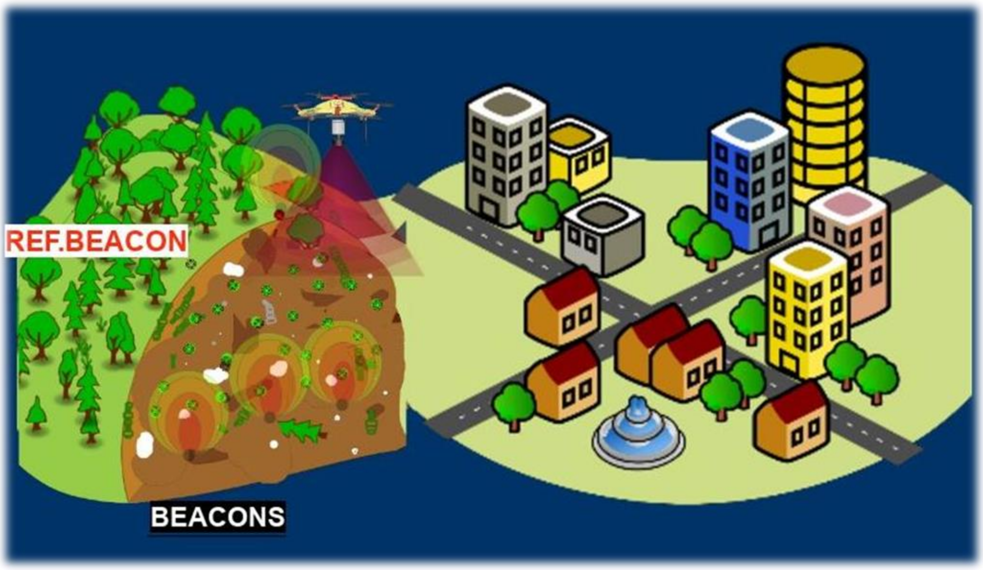


Figure 67: BLUESLEMON project scenery

The UAV will be designed to manage the beacons of the landslide movement monitoring system. The system developed can be used in risky situations (such as on rock faces or slopes), allowing safe and quick data detections. Compared to traditional GPS, it will be characterized by low hazard for operators, low cost, high spatial coverage (i.e. several sensors controlled in a flight). The accuracy of the sensors developed is determined using DGPS data recorded directly in the field. The main technological idea is to use the combination of electromagnetic induction power supply with unambiguous identification technologies, managed on the territory with UAV specially designed and tested, in order to periodically detect its position and, at the end of the monitoring period or in case of faults, inform operators correctly to perform a recovery/replacement of the component in the field.

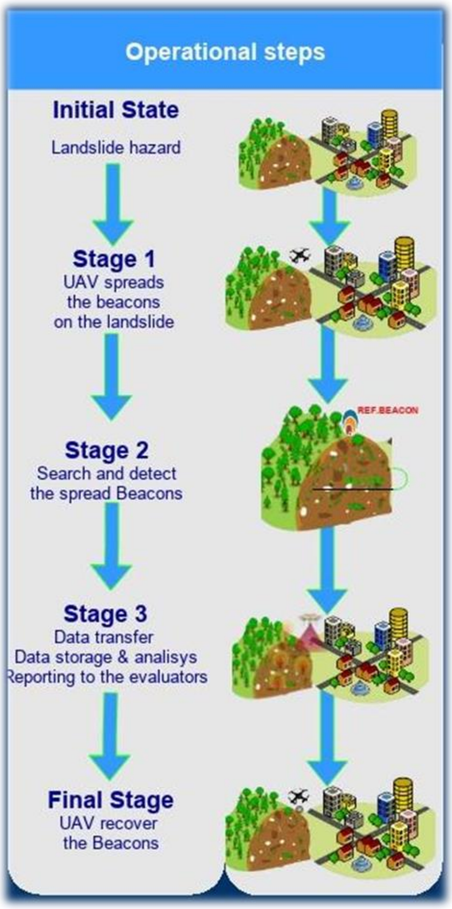


Figure 68: BLUESLEMON project operational steps

During all the operational steps showed in Figure 68, the vehicle shall be able to avoid possible obstacles along the trajectory established during mission planning. Flight missions can be carried out autonomously from the UAV, therefore the collision avoidance system will have to be operating continuously. The tests of the monitoring system, in particular of the UAV properly equipped, will be performed both in controlled conditions both directly in situ. The environmental conditions at the measuring site may be first simulated under controlled conditions in the terraXcube (70) at the NOI Techpark in Bolzano. In particular the LargeCube (71) will be equipped for testing individual components or the whole machine, to verify how some parameters environmental conditions may affect the performance of the flight platform subsequently used in operations. During the design phases, it will be searched for the best strategies to implement in the features of the UAV a system to identify and avoid obstacles along the planned flight path. Mission planning procedures will be defined according to the accessible databases of digital terrain models and flight obstacles. Moreover, the technologies to be implemented in avionics will be searched and studied to detect obstacles and to generate the manoeuvres to avoid obstacles during the flight mission.

Finally, new materials, components and new design techniques will be studied to obtain a system able to operate in environments with adverse climatic conditions (will be tested in the terraXcube laboratory in different altitude conditions even high, over 2500 m, humidity and temperature even low, below -20°C). New sensors and systems will be implemented to be able to navigate automatically, avoiding any obstacles in the planned route.

The three partners involved in the BLUESLEMON project are a research institution and two companies: Eurac research is the research partner of the project; FOS S.p.A. is the lead partner of the project; MAVTech S.r.l. is the technology partner of the project. This last one has allowed the collaboration and the realization of this thesis.

MAVTech s.r.l. (47) was founded as a spin-off company of Politecnico di Torino (2005-2014), currently also located in Bolzano/Bozen as a Technology Company of NOI Techpark Südtirol/Alto Adige. The main focus of MAVTech s.r.l. is the prototyping and production of Remotely Piloted Aircraft System (RPAS) with competitive performance and costs (including customer support and end-user training) resulting from the development of projects based on the transfer of new aerospace technologies from the research field to the operational and industrial sector.

Appendix B: GPS RTK Comparative Analysis

The choice of the GPS tracking device is very important for maintaining adequate positioning performances during flight missions. Following a preliminary analysis, in which the conventional Here2 and the RTK Here+ models (50) were compared, the hypothesis of using a conventional GPS was discarded due to errors in positioning too much high (the Here2 model analysed has a positioning accuracy of 2.5 m). Considering the need to detect Beacon position changes for distances of the order of one centimetre, it was necessary to consider GPS that supported RTK technology. The GPS RTK operating principle will be presented below and the various models analysed for the BLUESLEMON project will be subsequently analysed.

GPS RTK working principle

Real Time Kinematic positioning is a localization technique based on the use of two receivers: the "base", which is kept fixed on a known geographical point and the "rover", which is transported to the points where the position is to be measured, for example for detailed ground measurements, or which is taken on board vehicles to be able to detect its position with high accuracy. A positive aspect of the use of this technique on board UAVs is the possibility of making surveys in inaccessible scenarios, for example a landslide, where, being able to have high accuracies can be an essential advantage.

This technique, in fact, thanks to the presence of the base, whose geographical positioning is known a priori, allows to make corrections in real time, thus determining a GPS location with accuracy of the order of a few centimeters. In this way, the data provided does not require processing and differential correction in post processing but is available and usable in real time. The use of this technique is however limited by the effects generated by the ionosphere and the troposphere on the signal that passes through them; these areas of the atmosphere generate systematic errors in the raw data of the signal. From this problem arises the need to work keeping short distances between the base receiver and the rover receiver; this distance is defined baseline.

In Figure 69 it is possible to observe an example of base (left) and of rover (right) used for topographical surveys on the ground. It is evident the necessity to use much more compact rover receivers in order to make them embarkable on UAVs. Some RTK GPS systems specific to aircraft, compact and light weight will be analysed below.

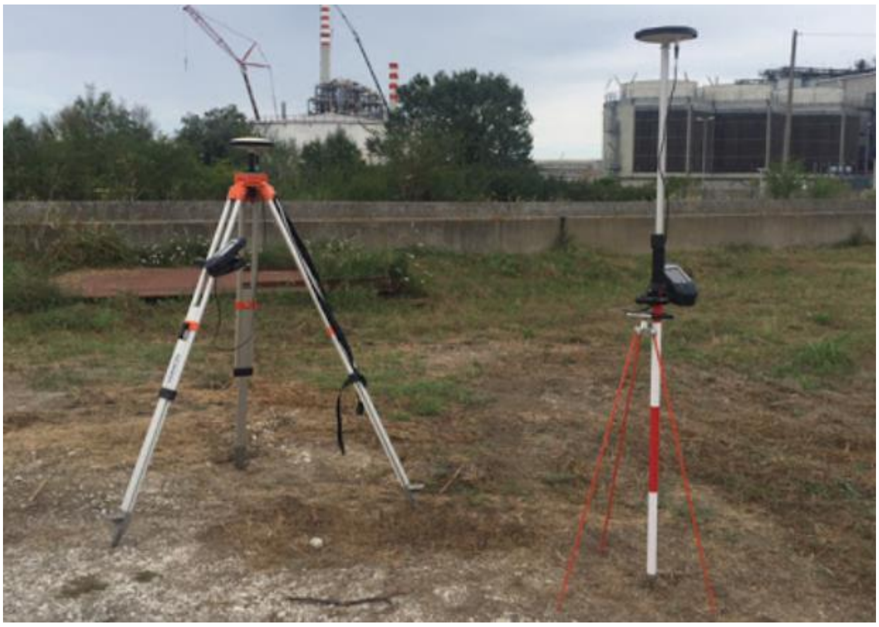


Figure 69: GPS RTK base and rover[1]

Here+ V2 RTK GNSS

The product Here+V2, developed by Hex Technology, Proficnc and Ardupilot, is able to provide positioning information with centimetric accuracy and allows multiple GNSS reception of current systems, including the latest, such as GPS, GLONASS, Beidou and Galileo, ensuring high satellite reception. The model is specifically designed to be integrated with the Pixhawk autopilot and compatible with autopilot management software such as Mission Planner.

The complete kit includes the GPS rover (Figure 70 in the center) that

[1] http://www.indago-rovigo.it/a54_rilievi-fotografici.html

must be embarked on the SAPR and a smaller GPS (Figure 70 on the left) that must be connected both to the computer of the ground station and to an appropriate external antenna (Figure 70 on the right) defining all together the base receiver. The model is also equipped with a compass, an IMU and an integrated barometer that allow to increase the accuracy of the measurements and related positioning data.

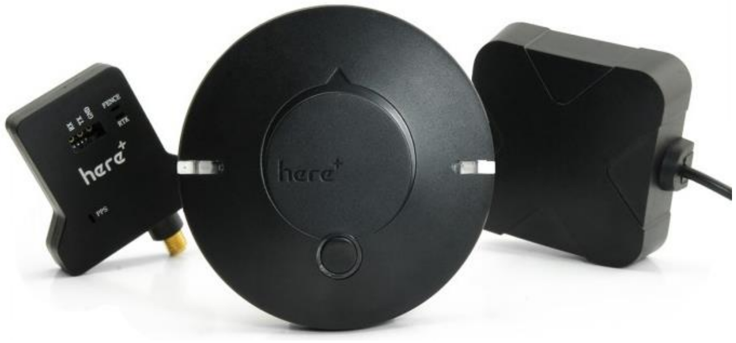


Figure 70: GPS RTK Here+V2 model[2]

The product has already been tested by MAVTech to verify its functionality on former prototypes, as can be seen in Figure 71. The rover module is clearly visible on the front back of the aircraft, while the antenna of the base was placed on a tripod and connected to the base, in turn connected to a computer of the ground station.

Specifications

- Accuracy: 2.5 cm + 1 ppm CEP
- Update rate during navigation: RTK: up to 8 Hz
- Working voltage: 5V
- Usage environment: maximum altitude 5000m, maximum velocities 500 m/s
- Operative temperatures: from -40°C to 85°C
- Dimensions: 67 x 17 mm (ø x h)
- Weight: 49 g
- Cost: 720 €

[2] http://www.proficnc.com/system-kits/77-gps-module.html

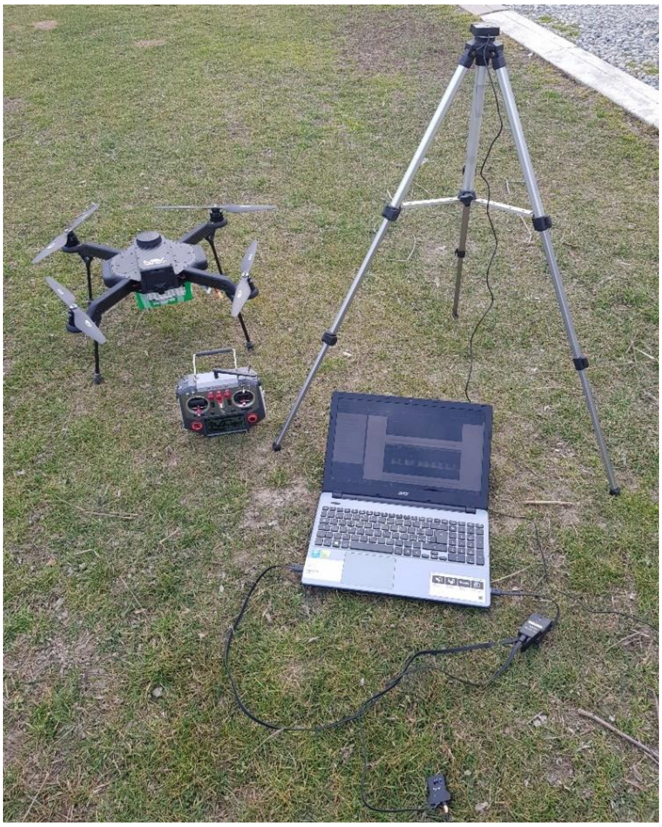


Figure 71: Base and rover configuration – integration onboard a Mavtech UAV

Septentrio AsteRx-m2 UAS

Septentrio has developed different types of RTK GPS receivers for applications such as logistics, surveillance, mapping and automation. Specifically, for the integration on UAS there are three models, which will be considered Asterx-m2 UAS as specific for the integration with autopilots such as Pixhawk (also of interest is the Asterx-m2a, slightly more compact in size and weighing less than 10 g). The main feature that makes it different in operation than the previous model is the ability to work in double frequency (L1 and L2 bands). Multifrequency GPS allows to obtain positioning data that are less affected by atmospheric disturbances and therefore are much more precise.

In fact, by obtaining data from two different bands, it is possible to provide more information to the receiver, therefore the possibility of an error occurring due to the crossing of the ionosphere, the troposphere and the reflection of the signal in urban environments is greatly reduced, for example close to buildings. In addition, if the reception of one of the two bands proves impossible, the other also performs the backup functionalities. For this reason and for some software libraries integrated in the model (such as AIM+: Advanced Interference Mitigation), the product does not need to communicate with a fixed receiver, but is able to manage the positioning corrections autonomously. Hence, it is not necessary to purchase a receiving base, but only the rover is required within the aircraft (Figure 72). However, the use with a base is an available option.

Figure 72: GPS RTK AsteRx-m2 UAS model[3]

Specifications

- Multifrequency (L1, L2) and multi-constellation (GPS e GLONASS) receiver
- Horizontal accuracy: 0.6 cm + 0.5 ppm
- Vertical accuracy: 1 cm + 1ppm
- Output rate: 20 Hz
- Working voltage: 5V
- Initializing time: 7s
- Operative temperatures: from -40°C to +85°C
- Pixhawk compatible
- Dimensions: 47.5x70x14.9 mm
- Weight: 38 g

[3] https://www.septentrio.com/en/products/gnss-receivers/rover-base-receivers/oem-receiver-boards/asterx-m2-uas

North Surveying and Positioning Systems RTKite GNSS

The RTKite system is also multi-frequency and works in the L1 and L2 bands as the model previously described. It is designed to be compatible with Pixhawk and other autopilots and is able to communicate both in UHF band and on mobile network. Unlike the previous model it requires the use of a base receiver as well as the rover receiver.

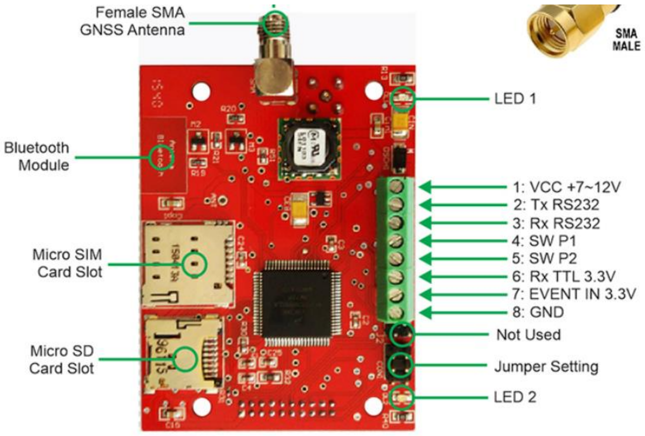


Figure 73: GPS RTK RTKite model[4]

Specifications

- Multifrequency (L1, L2) and multi-constellation (GPS, SBAS, COMPASS, Galileo, GLONASS) receiver
- Horizontal accuracy: 0.8 cm + 1 ppm
- Vertical accuracy: 1,5 cm + 1ppm
- Input power: 7-12VDC
- Initializing time: <10s
- Operative temperatures: from -40°C to +85°C
- Pixhawk compatible
- Dimensions: 74x54x25.4mm (without antenna)
- Weight: 55 g (without antenna; antenna's weight: 25g)
- Cost: ~2200 €

[4] https://gnssrtkmodule.com/index.php/manuals/8-rtkite-gnss-receiver-user-guide

Drotek Sirius RTK GNSS rover (F9P) and base

Finally, the model of the Drotek Electronics developer is analysed, which provides the complete kit of both rover module and RTK base (Figure 74). The Sirius rover module, based on ZED-F9P U-blox technology, offers a multi-band GNSS for high performance and reliability in various industrial applications. The F9P module provides a positioning accuracy of about 1 cm, a convergence time of less than 10 seconds and a navigation update speed of up to 20 Hz.

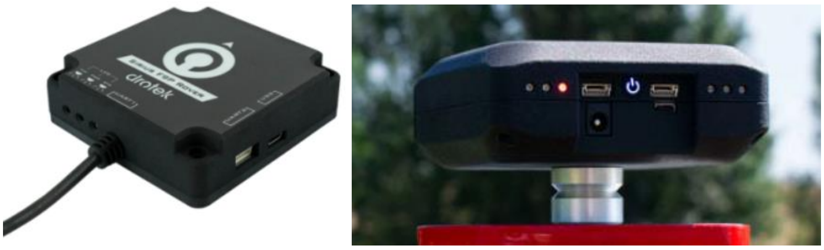


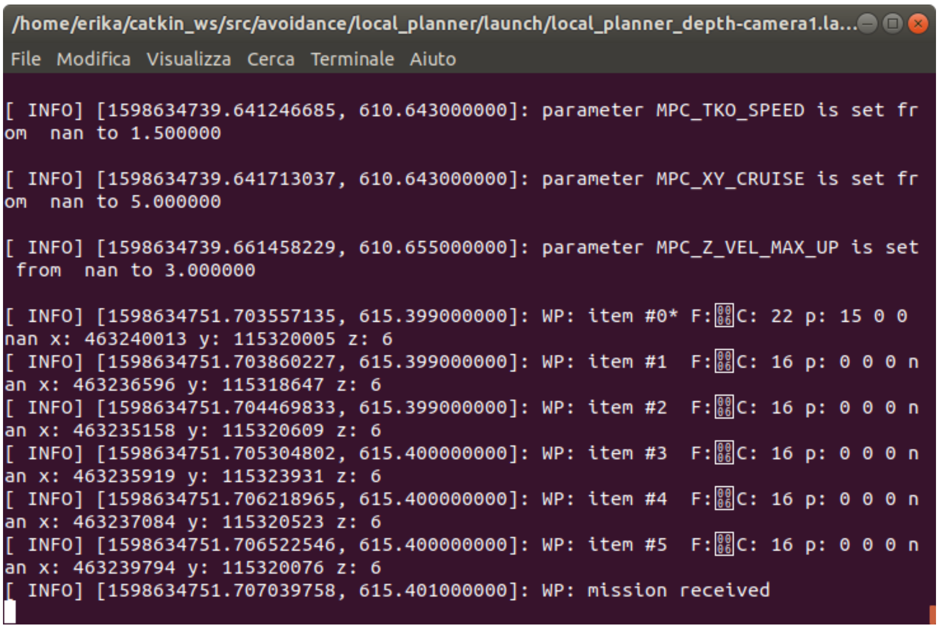
Figure 74: Drotek RTK GPS Sirius model, rover (left) and base (right) (51)

Following an in-depth research and a comparison with the performance of the RTK Here+ GPS, previously tested by the company, this model has been selected for the project, as it reflects the best compromise between performance required, market price and technological offer. Below are listed the specifications of the system, in particular weight and dimensions related to the rover, which will be embarked on the UAV.

Specifications

- Multifrequency (L1, L2) and multi-constellation (GPS, BeiDou, Galileo, GLONASS)
- Accuracy: 1 cm
- Convergence time: 10 s
- Update rate during navigation: up to 20 Hz
- Pixhawk compatible
- Working power: 5V – 75 mA
- Operative temperatures: from -20°C to 70°C
- Dimensions: 74 x 74 x 22 mm
- Weight: 124 g
- Cost: 720 € (base + rover)

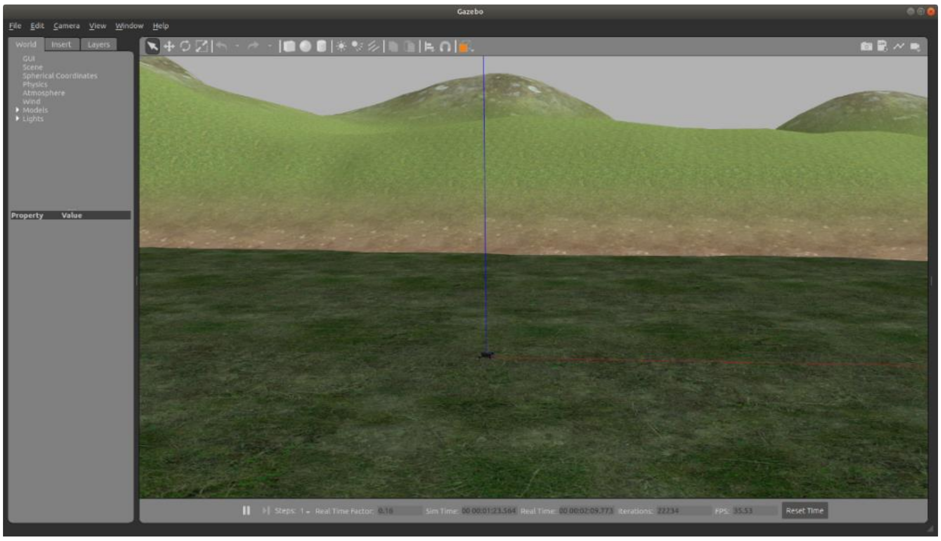
Appendix C: Additional Graphic Results


/home/erika/catkin_ws/src/avoidance/local_planner/launch/local_planner_depth-camera1.la...
File
Modifica
Visualizza
Cerca
Terminale
Aiuto

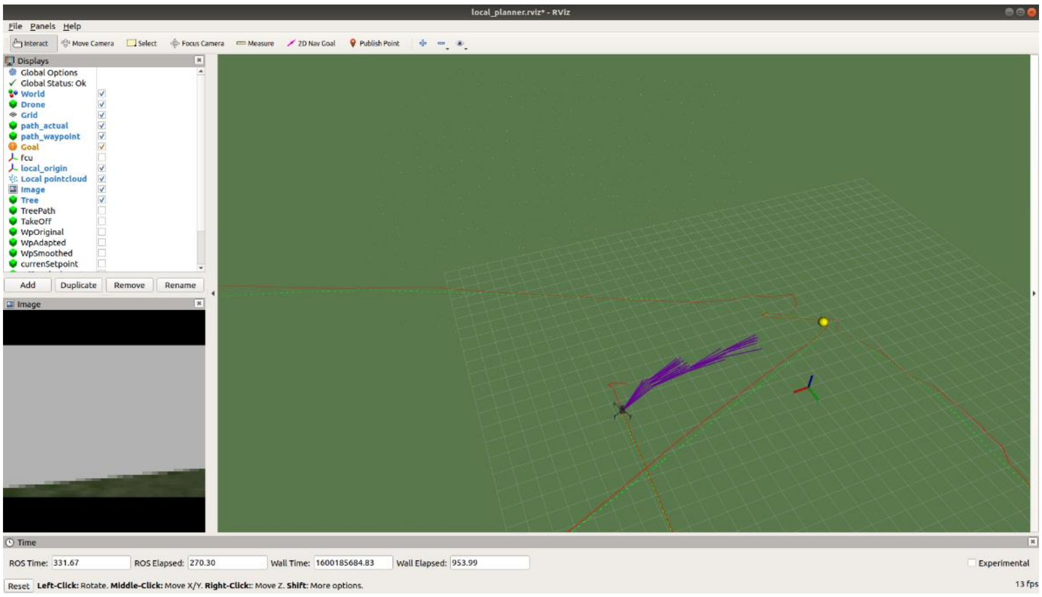
```
[ INFO] [1598634739.641246685, 610.643000000]: parameter MPC_TKO_SPEED is set fr
om  nan to 1.500000

[ INFO] [1598634739.641713037, 610.643000000]: parameter MPC_XY_CRUISE is set fr
om  nan to 5.000000

[ INFO] [1598634739.661458229, 610.655000000]: parameter MPC_Z_VEL_MAX_UP is set
 from  nan to 3.000000

[ INFO] [1598634751.703557135, 615.399000000]: WP: item #0* F:␆C: 22 p: 15 0 0
nan x: 463240013 y: 115320005 z: 6
[ INFO] [1598634751.703860227, 615.399000000]: WP: item #1  F:␆C: 16 p: 0 0 0 n
an x: 463236596 y: 115318647 z: 6
[ INFO] [1598634751.704469833, 615.399000000]: WP: item #2  F:␆C: 16 p: 0 0 0 n
an x: 463235158 y: 115320609 z: 6
[ INFO] [1598634751.705304802, 615.400000000]: WP: item #3  F:␆C: 16 p: 0 0 0 n
an x: 463235919 y: 115323931 z: 6
[ INFO] [1598634751.706218965, 615.400000000]: WP: item #4  F:␆C: 16 p: 0 0 0 n
an x: 463237084 y: 115320523 z: 6
[ INFO] [1598634751.706522546, 615.400000000]: WP: item #5  F:␆C: 16 p: 0 0 0 n
an x: 463239794 y: 115320076 z: 6
[ INFO] [1598634751.707039758, 615.401000000]: WP: mission received
```

Sending the planned mission from the QGC interface to the autopilot PX4 firmware by using waypoints coordinates; it is displayed on the command prompt.

Scenario 1 Simulation

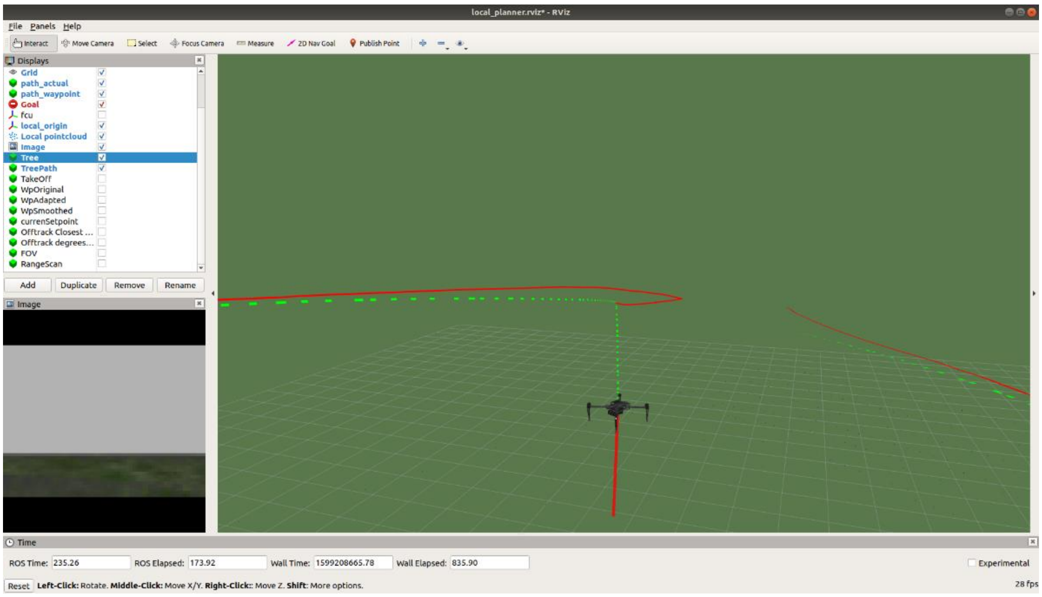


Simulation environment constituted by a flat area, hilly terrain around and the drone generated at the centre of the environment, in the origin of the axes of the reference system.


local_planner.rviz* - RViz
File Panels Help
Interact Move Camera Select Focus Camera Measure 2D Nav Goal Publish Point
Displays
Global Options
Global Status: Ok
World
Drone
Grid
path_actual
path_waypoint
Goal
fcu
local_origin
Local pointcloud
Image
Tree
TreePath
TakeOff
WpOriginal
WpAdapted
WpSmoothed
currenSetpoint
Add Duplicate Remove Rename
Image
Time
Experimental
Reset Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click: Move Z. Shift: More options.

ROS Time: 331.67 ROS Elapsed: 270.30 Wall Time: 1600185684.83 Wall Elapsed: 953.99 13 fps

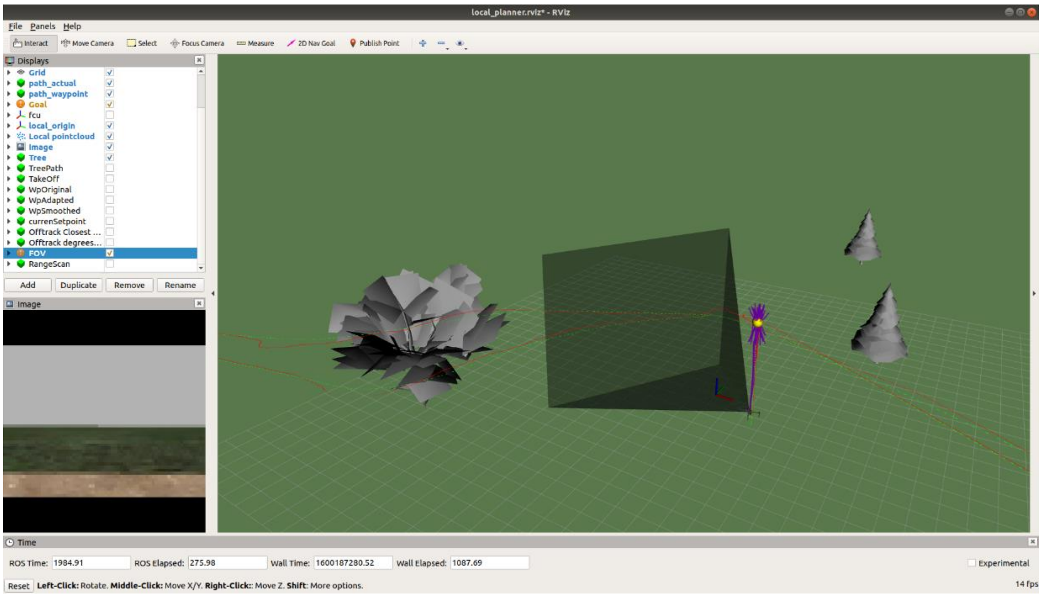
Possible path tree visualization towards a waypoint in an environment without obstacles.



ROS Time: 235.26 ROS Elapsed: 173.92 Wall Time: 1599208665.78 Wall Elapsed: 835.90 28 fps

Difference between the path imposed and the trajectory actually completed (green dotted line). End of the definition of the optimal path (thin red line) at the beginning of the landing phase, because of the collision avoidance system functioning only during automatic missions.

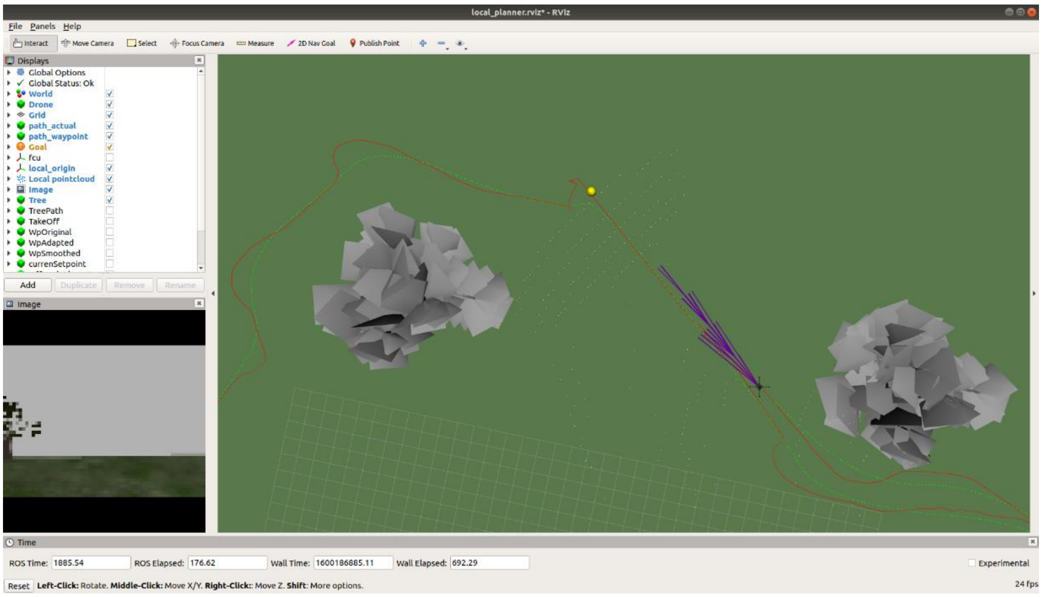
Scenario 2 Simulation



ROS Time: 1984.91 ROS Elapsed: 275.98 Wall Time: 1600187280.52 Wall Elapsed: 1087.69 □ Experimental

Reset **Left-Click:** Rotate. **Middle-Click:** Move X/Y. **Right-Click:**: Move Z. **Shift**: More options. 14 fps

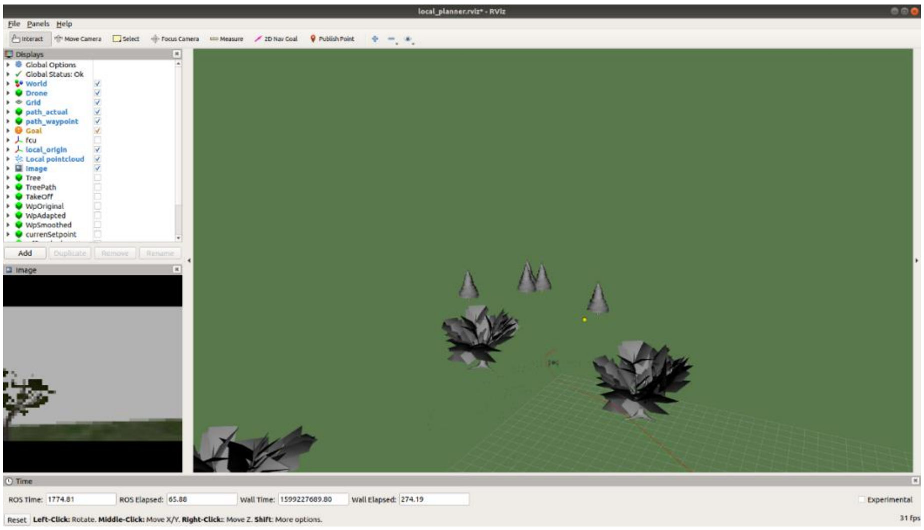
Camera Field Of View and possible paths tree visualization towards the first waypoint.



ROS Time: 1885.54 ROS Elapsed: 176.62 Wall Time: 1600186885.11 Wall Elapsed: 692.29 □ Experimental

Reset **Left-Click:** Rotate. **Middle-Click:** Move X/Y. **Right-Click:**: Move Z. **Shift**: More options. 24 fps

Path redefinition among obstacles towards the next waypoint.

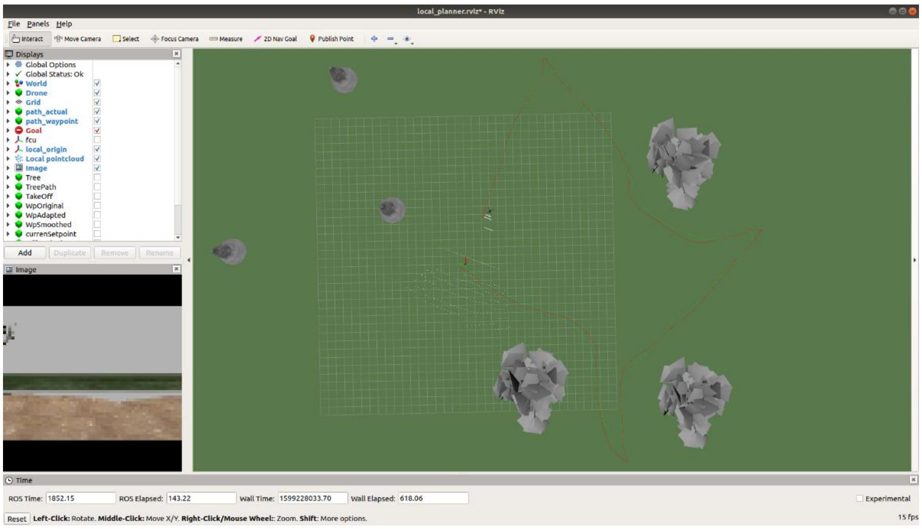


- Global Options
- ✓ Global Status: Ok
- World ☑
- Drone ☑
- Grid ☑
- path_actual ☑
- path_waypoint ☑
- Goal ☑
- fcu ☐
- local_origin ☑
- Local pointcloud ☑
- Image ☑
- Tree ☐
- TreePath ☐
- TakeOff ☐
- WpOriginal ☐
- WpAdapted ☐
- WpSmoothed ☐
- currentSetpoint ☐

ROS Time: 1774.81 ROS Elapsed: 65.88 Wall Time: 1599227689.80 Wall Elapsed: 274.19 ☐ Experimental

Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click:: Move Z. Shift: More options. 31 fps

Obstacle avoidance to reach the waypoint.

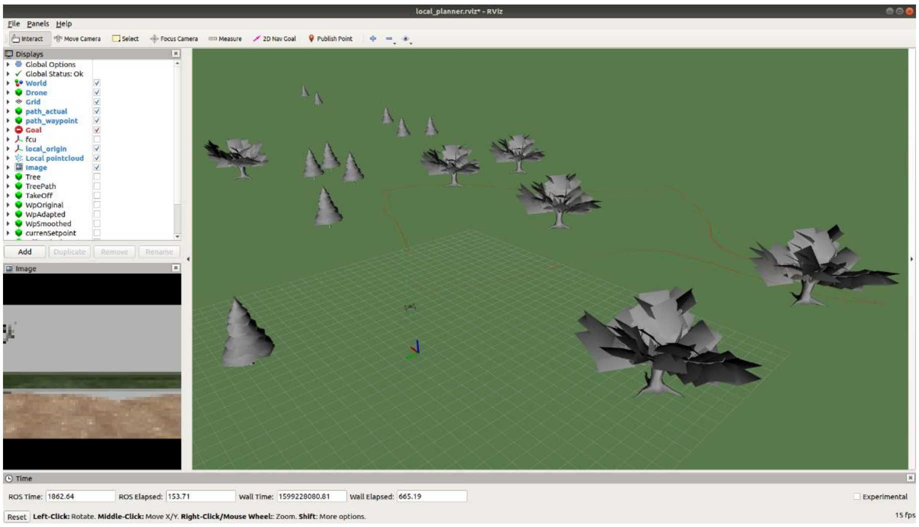


- Global Options
- ✓ Global Status: Ok
- World ☑
- Drone ☑
- Grid ☑
- path_actual ☑
- path_waypoint ☑
- Goal ☑
- fcu ☐
- local_origin ☑
- Local pointcloud ☑
- Image ☑
- Tree ☐
- TreePath ☐
- TakeOff ☐
- WpOriginal ☐
- WpAdapted ☐
- WpSmoothed ☐
- currentSetpoint ☐

ROS Time: 1852.15 ROS Elapsed: 143.22 Wall Time: 1599228033.70 Wall Elapsed: 618.06 ☐ Experimental

Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel:: Zoom. Shift: More options. 15 fps

Final path completed avoiding obstacles. Top view.



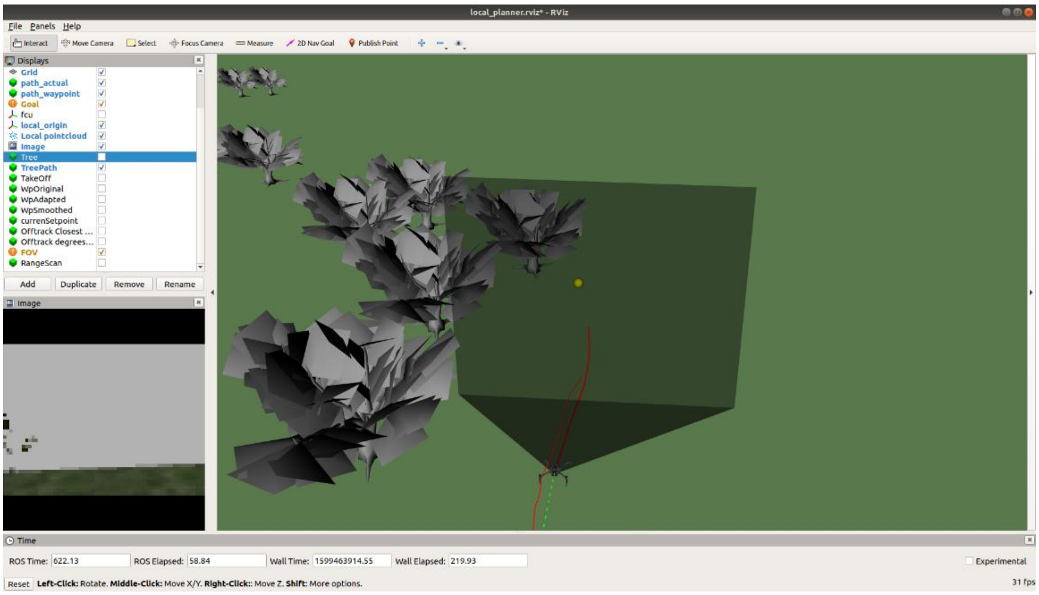
- Global Options
- ✓ Global Status: Ok
- World ☑
- Drone ☑
- Grid ☑
- path_actual ☑
- path_waypoint ☑
- Goal ☑
- fcu ☐
- local_origin ☑
- Local pointcloud ☑
- Image ☑
- Tree ☐
- TreePath ☐
- TakeOff ☐
- WpOriginal ☐
- WpAdapted ☐
- WpSmoothed ☐
- currentSetpoint ☐

ROS Time: 1862.64 ROS Elapsed: 153.71 Wall Time: 1599228080.81 Wall Elapsed: 665.19 ☐ Experimental

Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel:: Zoom. Shift: More options. 15 fps

Final path completed avoiding obstacles. Three-dimensional view.

Scenario 3 Simulation

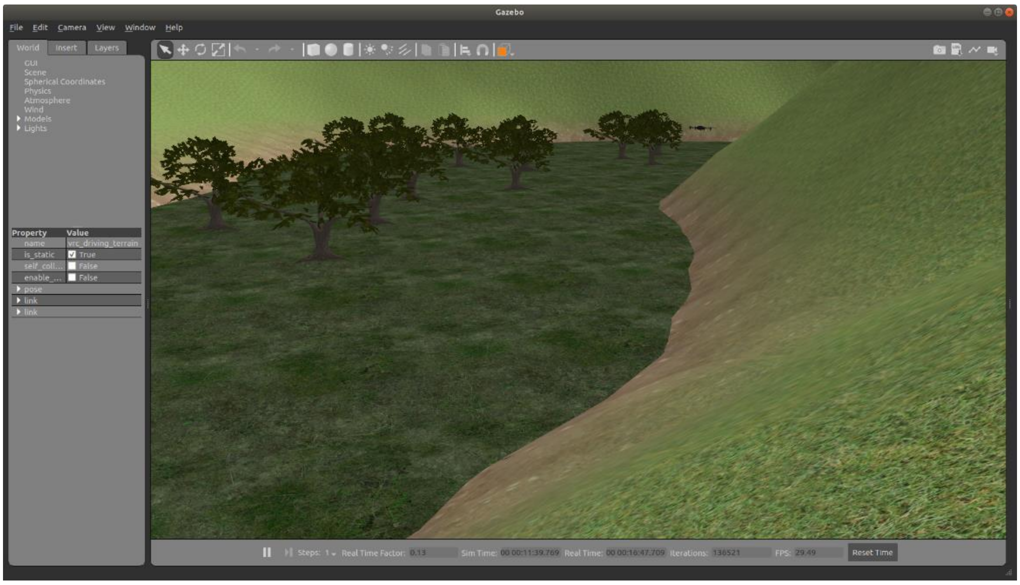


Grid	☑
path_actual	☑
path_waypoint	☑
Goal	☑
fcu	□
local_origin	☑
Local pointcloud	☑
Image	☑
Tree	□
TreePath	☑
TakeOff	□
WpOriginal	□
WpAdapted	□
WpSmoothed	□
currenSetpoint	□
Offtrack Closest ...	□
Offtrack degrees...	□
FOV	☑
RangeScan	□

ROS Time: 622.13 ROS Elapsed: 58.84 Wall Time: 1599463914.55 Wall Elapsed: 219.93 □ Experimental

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click::** Move Z. **Shift:** More options.

Camera FOV visualization and obstacle detection.

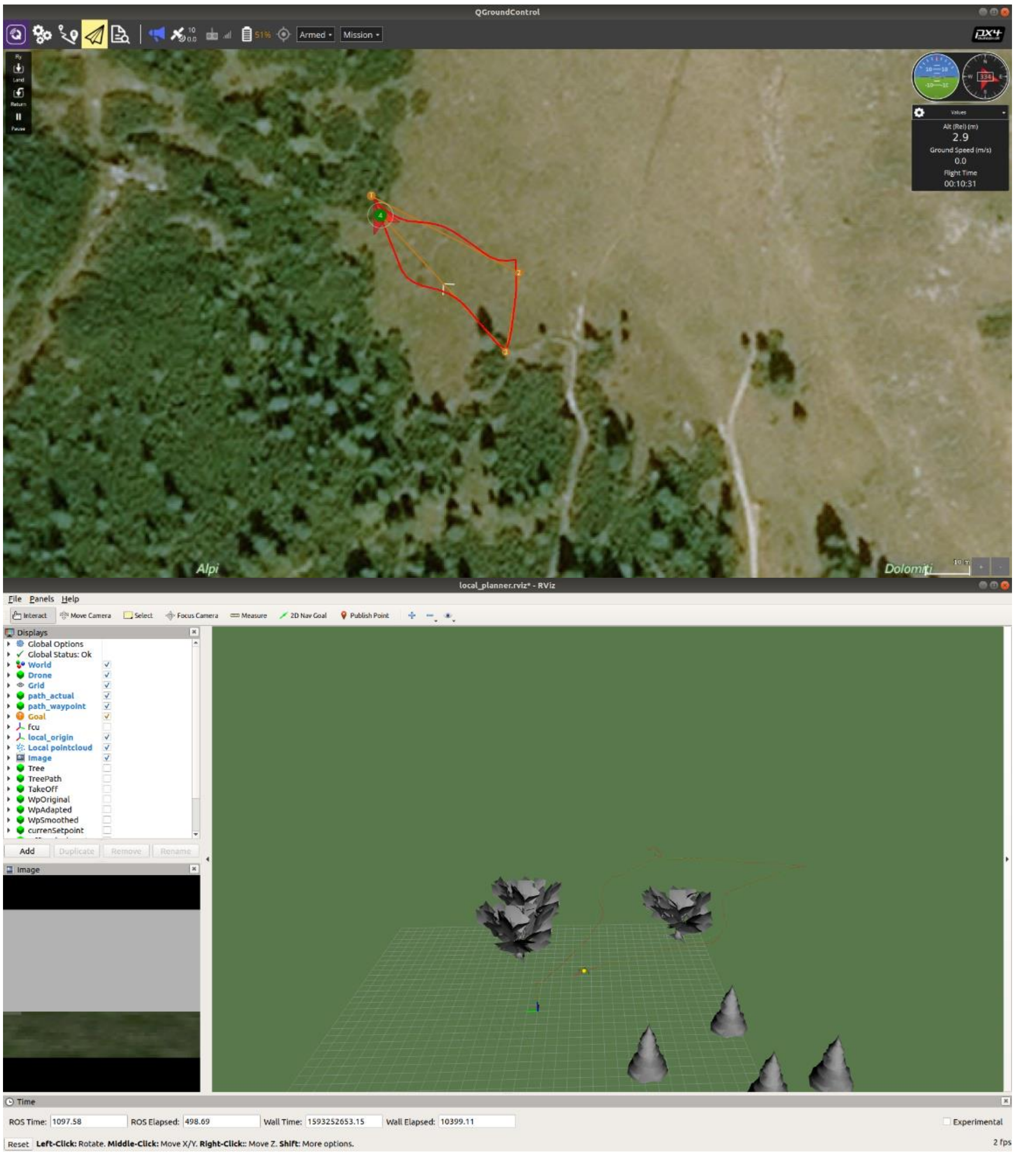


Property	Value
name	vrc_driving_terrain
is_static	☑ True
self_coll...	□ False
enable_...	□ False

Steps: 1 Real Time Factor: 0.13 Sim Time: 00 00:11:39.769 Real Time: 00 00:16:47.709 Iterations: 136521 FPS: 29.49

Drone flying near a rock wall in an impervious environment.

Other Simulations



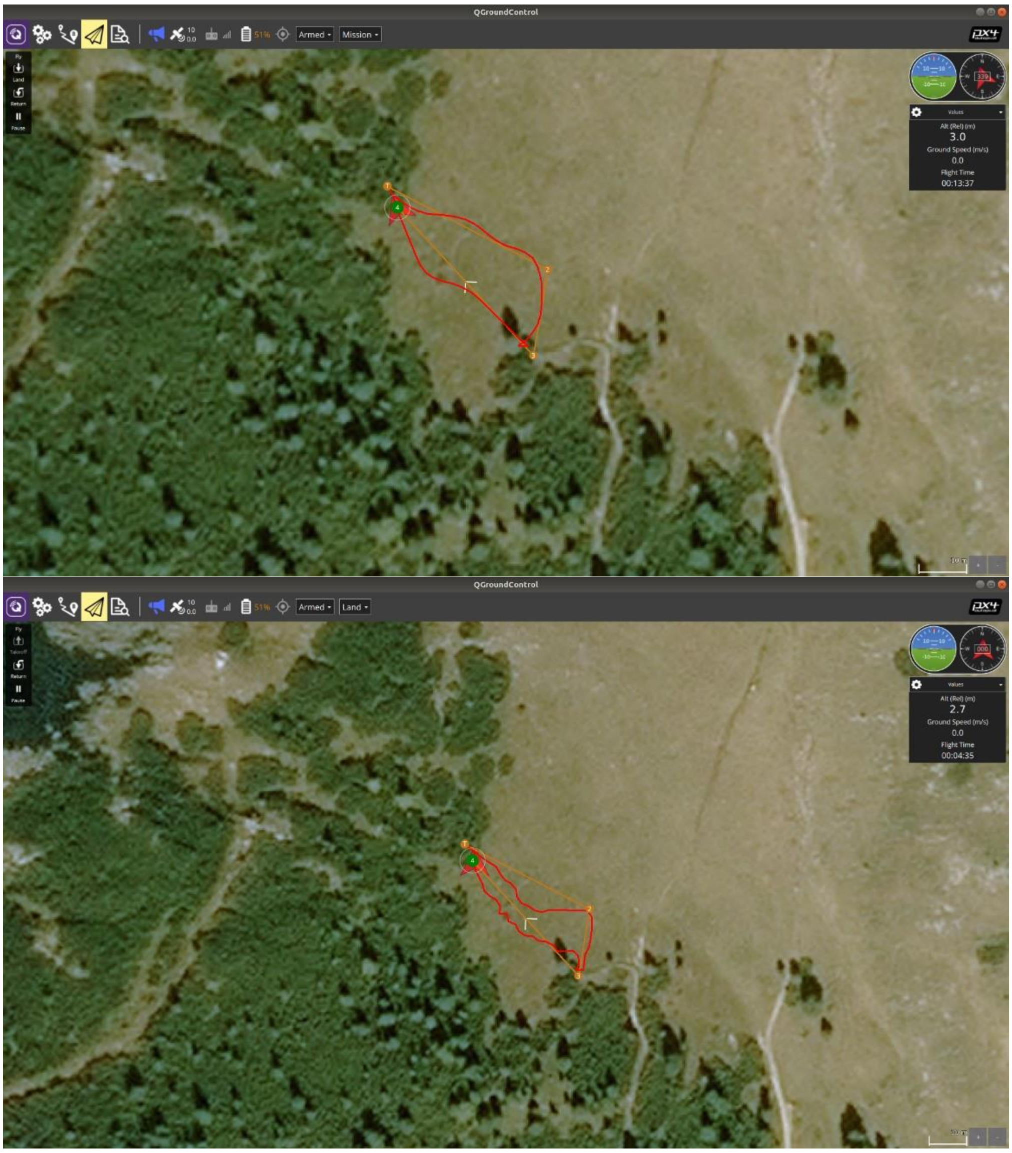
Alt (Rel) (m) 2.9
Ground Speed (m/s) 0.0
Flight Time 00:10:31

ROS Time: 1097.58 ROS Elapsed: 498.69 Wall Time: 1593252653.15 Wall Elapsed: 10399.11

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click:**: Move Z. **Shift:** More options.

2 fps

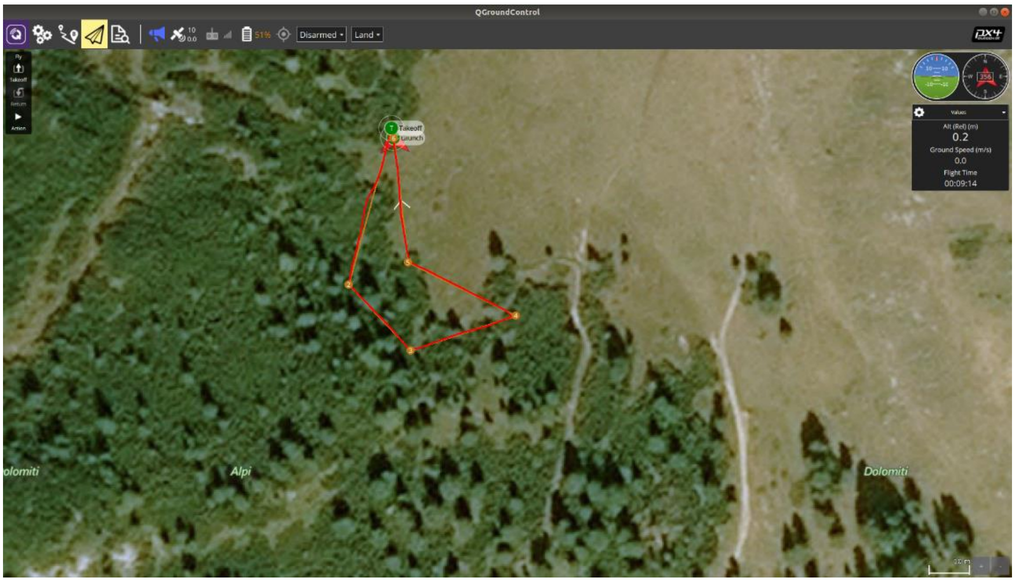
Obstacle avoidance among trees; QGC and RViz visualization.



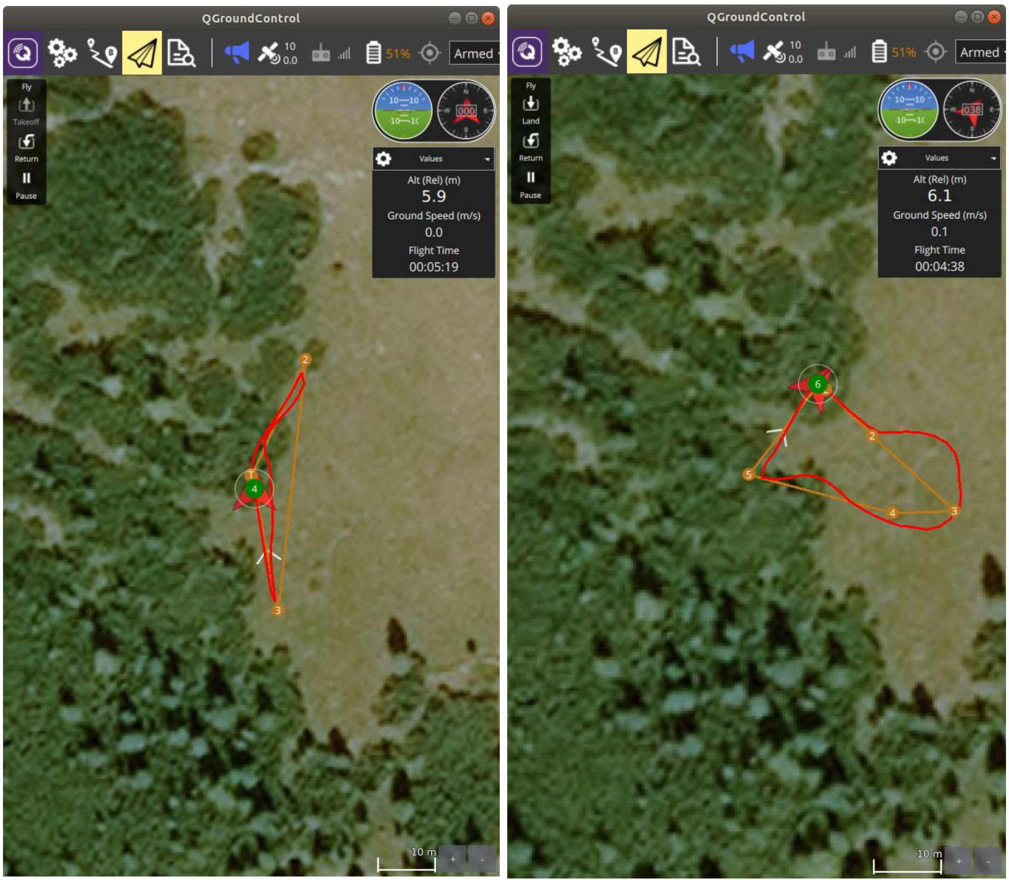
Alt (Rel) (m) 3.0
Ground Speed (m/s) 0.0
Flight Time 00:13:37

Alt (Rel) (m) 2.7
Ground Speed (m/s) 0.0
Flight Time 00:04:35

Same mission, different paths accomplished: degree of randomness of the optimal route definition



Alt (Rel) (m) 0.2
Ground Speed (m/s) 0.0
Flight Time 00:09:14



Alt (Rel) (m) 5.9
Ground Speed (m/s) 0.0
Flight Time 00:05:19

Alt (Rel) (m) 6.1
Ground Speed (m/s) 0.1
Flight Time 00:04:38

Other missions completed: the collision avoidance system is active and allows to accomplish missions.