



H2020-ICT-25-2016-2017



**HYbrid FLying rollIng with-snake-aRm robot for contact inSpection**

## **HYFLIERS**

### **D4.2**

#### *Navigation support system and operator interface*

<b>Contractual date of delivery</b>	30 Jun 2020, agreed postponed date 31 Dec 2020
<b>Actual date of delivery</b>	31 Dec 2020
<b>Editor(s)</b>	Ulrico Celentano (UOULU)
<b>Author(s)</b>	Marko Kauppinen, Ulrico Celentano (UOULU), Alvaro Caballero, Manuel Bejar, Guillermo Heredia, Anibal Ollero (USE), Miguel Ángel Trujillo, Antidio Viguria (FADA-CATEC)
<b>Workpackage</b>	WP4
<b>Estimated person-months</b>	20
<b>Dissemination level</b>	PU
<b>Type</b>	R
<b>Version</b>	1.0
<b>Total number of pages</b>	68

#### **Abstract:**

This document reports about the progress about the path planning and navigation support for hybrid robot autonomous operation, the human operation supporting function by augmented reality, and the mobile platform for remote control support and battery management.

#### **Keywords:**

Augmented reality. Autonomous motion. Battery management. Dynamic awareness. HoloLens. Human interaction. Hybrid robot. Mixed reality. Navigation support. Operations support. Optimisation. Path planning. Pipe inspection. Reactive behaviour. Remote control. Robot operating system. Support platform. Unmanned aerial vehicle.

## Executive summary

This deliverable extends the developments presented in the earlier deliverable [D4.1].

First, developments concerning the planning algorithms of the movement of the hybrid robot (HR) and its associated navigation support (obstacle detection, reactive behaviours, etc.) are reported in Section 2. These algorithms pave the way for autonomous operation of the HR in its inspection tasks. With reference to the general overview of the three main research areas defined in D4.1 for this topic (see Section 2), this deliverable presents developments.

One is about the generation of motion plans that allow autonomous motion along complete sequences of inspection points. Here, the obstacles included in the industrial environment maps are considered for the generation collision-free trajectories, where the hybrid motion capabilities (flying, rolling) are exploited to achieve optimal trajectories according to operation time or power consumption. Additionally, the motion planner is extended to guarantee safer trajectories for autonomous operation in presence of a dynamic behaviour of the system. Finally, navigation support enhances the level of autonomy offered by the system by implementing reactive behaviours in case the on-board sensors reveal the presence of obstacles not considered in the planning algorithms (not included in the pre-existing map, for example due to plan updates or repairs).

Supplemental material about the above part is provided in the form of videos, which are listed in the dedicated appendix.

Further (in Section 3), this deliverable presents a 3D (three-dimensional) augmented reality system to be part of the human operation supporting functions. This new type of interface for the hybrid robot operator will enrich the information available during the operation visually showing the status of the hybrid robot and its mission. This information will be provided to the operator in such a way that they do not have to change its field of view and hence, they can always maintain a look to the hybrid robot and its operation at any time.

The system uses Microsoft HoloLens goggles that allow mixed reality, basically combining the real world with a virtual environment where the physical and digital objects coexist both together and can even interact between them. The system supports sight, voice, and hand gestures for human interactions and it is enriched by sounds and spatial mapping for immersion and control. HoloLens run on Windows 10 on which also runs Unity (used to e.g. create videogames). On the other side a ROS (robot operating system) module runs on Linux Ubuntu for communication with the unmanned aerial vehicle (UAV). Ros-Sharp on the Windows side is used to engage Unity with ROS.

This deliverable also reports (Section 4) about the developments of the remote mobile support platform ground control station computer, including mapping support such as occupancy mapping (visual tracking odometry for the light control unit through ROS and point cloud measurements) and deep learning object detection system from RGBD (red, green, blue, depth) image, currently using available modules with custom solutions being considered.

The mobile support platform (Section 5), accommodating all the needed components, is realised as a pushable cart that can be transformed into an easily transferrable form (like a large bag). When opened, the cart has two trays. At the bottom, the electrical part box includes the AC-DC (alternate current to direct current) power supply unit (1 kW with input range of 85 - 265 V) and the main battery (a safe and reliable 8-cell LiFePO<sub>4</sub> battery pack). The total energy capacity is 2.5 kWh (with a voltage range of 20 to 28.8 V) and it is able to charge the HR several times (in situations where an

AC power is not available) while simultaneously running the computer and communications hardware. A microcontroller unit performs real time monitoring (equipped with an external OLED screen) and cuts off the battery outputs if the available energy level drops too low. A custom solution enables operating uninterruptedly when disconnected and connected from AC power. At the top of the cart, a detachable ABS plastic water-resistant box contains all the required components for communications (Ethernet and WiFi with internal antennas) and support functions related to HR operation (including database and navigation support), waterproof keyboard and mouse, a water and dust protected multi-touch screen, and waterproof HDMI, Ethernet and USB ports.

## Abbreviations and symbols

1D	one-dimensional
2D	two-dimensional
3D	three-dimensional
ABS	acrylonitrile butadiene styrene
AC	alternating current
API	application programming interface
BLDC	brushless DC
BMS	battery management system
BRIEF	binary robust independent elementary feature
CAD	computer-aided design
COVID	coronavirus disease
DA	dynamic awareness
DC	direct current
DOF	degree of freedom
FCU	flight control unit
GATT	generic attribute
GCS	ground control station
HD	high-definition
HDMI	high-definition multimedia interface
HID	human interface device
HMR	hybrid mobile robot
HR	hybrid robot (either HMR or HRA)
HRA	hybrid robot with arm
HYFLIERS	hybrid flying rolling with-snake-arm robot for contact inspection
IC	integrated circuit
IR	infrared
LiDAR	light detection and ranging
LiFePO <sub>4</sub>	lithium iron phosphate
LiPo	lithium polymer
MCU	microcontroller unit
MOSFET	metal-oxide semiconductor field-effect transistor
MP	motion planner
MSP	mobile support platform
NVDLA	Nvidia Deep Learning Accelerator
OLED	organic light-emitting diode
ORB	oriented fast and rotated BRIEF
OS	operating system
PC	personal computer
PCB	printed circuit board
PDI	proportional-integral-derivative

PMS	power management system
PSU	power supply unit
PVC	polyvinyl chloride
RAM	random access memory
RGBD	red, green, blue, depth
ROS	robot operating system
RRT	rapidly-exploring random tree
SLAM	simultaneous localization and mapping
UAV	unmanned aerial vehicle
UGV	unmanned ground vehicle
UI	user interface
USB	universal serial bus
VSLAM	visual SLAM
Wi-Fi	Wireless Fidelity (synonym for WLAN, wireless local area network)
$E_{fly}$	electrical energy consumption when flying between aerial nodes
$E_{roll}$	electrical energy consumption when rolling
$E_{tran}$	electrical energy consumption when performing a hybrid transition
$e_{fly}$	energy consumptions per unit of displacement when the system is flying between aerial nodes
$e_{roll}$	energy consumptions per unit of displacement when the system is rolling
$e_{tran}$	energy consumptions per unit of displacement when the system is performing a hybrid transition
$F_x$	lifting force
$F_r$	friction force
$g$	gravity acceleration
$I$	electrical current
$I(t)$	current over time
$I_{xx}, I_{yy}, I_{zz}$	principal moment of inertia
$m$	mass of the robot
$n_{air}$	number of air nodes
$n_{pipe}$	number of pipe nodes
$n_r$	number of rotors
<b><math>p</math></b>	inspection node
$P_a$	aerodynamic power
$P_e$	electrical power
$P_m$	mechanical power
$r_0$	radius conical truncated volume
$r_r$	radius of rotor
$T$	operation time

$\mathbf{T}$	torque
$T_x, T_y, T_z$	torque along $x, y, z$
$V_{fly}$	battery voltage when flying
$V_{roll}$	battery voltage when rolling
$u_{fly}$	HR velocity when the system is flying between aerial nodes
$u_{roll}$	HR velocity when the system is rolling
$u_{tran}$	HR velocity when the system is performing a hybrid transition
$u_x, u_y, u_z$	velocity along $x, y, z$
$u_\phi, u_\theta, u_\psi$	velocity along $\phi, \theta, \psi$
$x$	position coordinate
$y$	position coordinate
$z$	position coordinate
$\beta$	semi-angle of conical truncated volume
$\Delta c$	cost increment
$\Delta t$	time of (rolling, flying or hybrid transition) operation phase
$\delta$	sampling resolution
$\Gamma = \{\gamma_{i,j}\}$	connection matrix (connection between pipe arrays $i$ and $j$ ), Boolean
$\eta$	orientation angle
$\eta_a$	electrical to aerodynamic conversion efficiency
$\eta_m$	electrical to mechanical conversion efficiency
$\theta$	pitch angle
$\lambda$	air-pipe exploration ratio
$\mu$	exploration ratio correcting factor
$\mu_f$	friction coefficient
$\rho_a$	air density
$\phi$	roll angle
$\psi$	yaw angle

## Table of Contents

Executive summary.....	2
Abbreviations and symbols.....	4
List of Figures .....	9
List of Tables .....	12
1. Introduction.....	13
2. Path planning and navigation support for autonomous operation (T4.1) .....	15
2.1. Introduction.....	15
2.2. Description of the hybrid robot.....	16
2.3. Architecture of the navigation system .....	17
2.4. Planning algorithm for autonomous hybrid motion.....	18
2.4.1. Hybrid planning space .....	18
2.4.2. Optimization capabilities in hybrid planning: operation time and energy consumption .....	21
2.4.3. Robustness in hybrid planning: air-pipe and pipe-pipe transitions.....	24
2.5. Extension of planning algorithm based on Dynamics Awareness for safer operation .....	26
2.5.1. Modelling and control.....	26
2.5.2. Dynamics Awareness.....	27
2.6. Navigation support to reinforce system autonomy .....	28
2.7. Simulation results.....	29
2.7.1. Planning algorithm for autonomous hybrid motion.....	31
2.7.2. Extension of planning algorithm based on Dynamics Awareness for safer operation .....	33
2.7.3. Navigation support to reinforce system autonomy .....	37
2.8. Experimental results.....	39
2.8.1. Planning algorithm for autonomous hybrid motion.....	39
3. Augmented reality applications (T4.4) .....	43
3.1. Microsoft HoloLens Goggles.....	44
3.1.1. Sight block .....	45
3.1.2. Gestures block.....	46
3.1.3. Voice block .....	47
3.1.4. Spatial mapping.....	47
3.1.5. Spatial sound .....	48
3.1.6. Coordinates systems.....	48
3.2. Software architecture .....	48
3.2.1. Unity.....	49
3.2.2. ROS.....	53

---

3.3. Experiments and results .....	54
3.3.1. Simulations.....	55
3.3.2. Simulations in Gazebo .....	58
3.3.3. Real application experiments .....	59
4. Remote control support on the mobile support platform PC (T4.2) .....	61
5. Mobile ground support platform (T4.3) .....	63
5.1. MSP cart.....	63
5.2. Main power battery system overview .....	63
5.3. MSP power supply box .....	64
5.4. Custom power management system board.....	64
5.5. MSP HR support PC box .....	65
5.6. MSP development and testing.....	66
References.....	67
Appendix: Supplemental material.....	68



## List of Figures

Figure 1. Hybrid rolling-aerial platform with manipulation capabilities.....	16
Figure 2. Hybrid rolling-aerial platform integrating the 5-DOF robotic arm supported by a 1-DOF linear guide system.....	17
Figure 3. Overall navigation architecture. ....	17
Figure 4. Scheme with both aerial and pipe states during the hybrid expansion of the proposed MP-HR planning algorithm. Aerial states in green colour and pipe states in purple colour. ....	19
Figure 5. Effect of the weighting parameter $\mu$ in the process of discretisation and sampling of the hybrid planning space: the complete set of <b><i>npipe</i></b> pipe and <b><i>nair</i></b> aerial states corresponding to the discretisation (left), the selected subset with uniform sampling (centre) and the selected subset corresponding to the sampling with compensating effect (right) that results in a more balanced distribution. ....	20
Figure 6. Illustrative examples of the criteria to select yaw angles for the different types of nodes in the hybrid planning space. ....	21
Figure 7. Basic behaviours corresponding to the optimization of the proposed cost functions. Flying is prioritised over rolling when minimising the operation time $T$ ( $ufly > uroll$ ) whilst rolling is prioritised over flying when minimising the energy consumption $E$ ( $eroll < efly$ ).....	22
Figure 8. Feasibility checking in transitions between aerial nodes and pipe nodes. Transitions taking place out of the region of safe transition are discarded.....	25
Figure 9. Block diagram of the control scheme. ....	27
Figure 10. Operation basis of the MP-HR-DA algorithm: feasibility analysis of new air-air branch (left) and feasibility analysis of new air-pipe branch (right). In contrast to the geometrical interpolation (blue dashed lines associated with MP-HR), the dynamic simulation of the system (green solid lines associated with MP-HR-DA), which is closer to the performance of the real system, leads to unsafe conditions. Consequently, the potential new nodes <b><i>xnew</i></b> should be discarded in both cases. ....	28
Figure 11. Highly-cluttered industrial environment with many pipe arrays.....	28
Figure 12. Example of application scenarios for hybrid reactivity.....	29
Figure 13. Example scenario illustrating the criteria followed to represent the different application scenarios.....	30
Figure 14. Application scenario proposed to validate the MP-HR planner: topology definition (left) and the corresponding safety regions (right). Moderately-cluttered environment with two pipe arrays, two isolated pipes and three inspection points.....	31
Figure 15. Trajectory planned with the MP-HR algorithm when optimising the operation time (dashed light blue) and its corresponding execution by the controlled hybrid robot (solid dark blue).....	32
Figure 16. Overshooting in reference waypoints extracted from Figure 15: landing on the inspection point 3 (left) and final approaching to the start / final point (right). The extended version of the motion planner that incorporates Dynamics Awareness will overcome these undesired behaviours of the closed-loop dynamics.....	32

Figure 17. Trajectory planned with the MP-HR algorithm when optimising the energy consumption (dashed light blue) and its corresponding execution by the controlled hybrid robot (solid dark blue). .....	33
Figure 18. Application scenario proposed to validate the MP-HR-DA planner: topology definition (left) and the corresponding safety regions (right). Highly-cluttered environment with three pipe arrays in complex topologies, a tank (red vertical cylinder) and four inspection points. ....	34
Figure 19. Trajectory planned with the MP-HR algorithm when optimising the operation time (dashed light blue) and its corresponding execution by the controlled hybrid robot (solid dark blue). ....	35
Figure 20. Detailed views of Figure 19: violations of margin of safety (left) and region of safe transition (right). ....	36
Figure 21. Trajectory planned with the MP-HR-DA algorithm when optimising the operation time (dashed light green) and its corresponding execution by the controlled hybrid robot (solid dark green). ....	37
Figure 22. Detailed views of Figure 21: the MP-HR-DA overcomes the violations of the margin of safety (left) and the region of safe transition (right). ....	37
Figure 23. Application scenario proposed to validate the reactive capabilities: topology definition (left) and the corresponding safety regions (right). It is a modified version of the scenario shown in in Figure 14, where the pipe highlighted in green that crosses horizontally from (-10m, 3m, 2.5m) to (10m, 3m, 2.5m) will not be included in the map provided to the planner. ....	38
Figure 24. Reactive capabilities for rolling manoeuvres. Trajectory initially planned by the MP-HR algorithm without considering the green pipe (left) and re-planned trajectory when the green pipe is detected by on-board sensors (right). ....	39
Figure 25. Experimental scenario proposed to validate the MP-HR planner. Scarcely-cluttered environment with two arrays of three PVC pipes and two inspection points. The Leica MS50 robotic total station is used to measure the motion of the HR. ....	40
Figure 26. Aerial manipulator with rolling base inspecting an array of pipes. ....	40
Figure 27. Trajectory planned with the MP-HR algorithm (dashed light blue) and executed by the experimental prototype of the controlled hybrid robot (solid dark blue). Optimization of the energy consumption. ....	41
Figure 28. Sequence of images showing the execution of the inspection task. ....	42
Figure 29: Mixed reality concerns. ....	43
Figure 30: Mixed reality devices. ....	43
Figure 31: Microsoft HoloLens Goggles. ....	44
Figure 32: HoloLens Clicker. ....	45
Figure 33: HoloLens Hardware schematic. ....	45
Figure 34: Air tap. ....	46
Figure 35: Bloom gesture. ....	46
Figure 36: Vision frame. ....	47
Figure 37: Example of a spatial allocation mesh covering a room. ....	48
Figure 38: Schematic of the software architecture. ....	49

Figure 39: Markers to be placed in the initialization point. ....	49
Figure 40: Plane for initialization. ....	50
Figure 41: Initialization interface.....	50
Figure 42: Interface of the application. ....	51
Figure 43: Map display. ....	52
Figure 44: Artificial Horizon display component: a) Base, b) Roll indicator, c) Yaw indicator and d) Artificial horizon. ....	53
Figure 45: Unity software architecture. ....	53
Figure 46: ROS topics and data flow. ....	54
Figure 47: Tree of the relationship between all the positions. ....	54
Figure 48: Battery indicator static test. ....	56
Figure 49: Initialization point. ....	57
Figure 50: Waypoint display and communications intensity.....	57
Figure 51: Position, speed, height and distance indicators test.....	58
Figure 52: Static test with a real UAV.....	59
Figure 53: Real flights experiments. ....	60
Figure 54. Remote GCS control testing with the mobile robot in the dark green circle. Set route shown with green arrow, end position with the light green box, and obstacle avoiding planned route output shown with orange arrows. ....	61
Figure 55. Octomapping [OctoM] performed with a test UAV, using RealSense D435 camera for depth measurements and T265 for odometry, which is also given as optical flow stream input to the Pixhawk FCU. On the right, the UAV was flying on the first floor (blueish), and then traversed by walking up to the third floor (yellow – red) via a staircase and to an office space at the end of a corridor.....	62
Figure 56. Object detection testing on the UOULU hexacopter test UAV.....	62
Figure 57. The collapsible and pushable MSP cart for mounting the HR support related hardware.....	63
Figure 58. The computer-aided design (CAD) drawing and the realized main power supply box of the MSP with the mounted hardware.....	64
Figure 59. Simplified overview of the PMS board functionality.....	65
Figure 60. The MSP PC box connected to the hexacopter over Wi-Fi, visualizing the occupancy grid map being generated onboard the UAV.....	66

**List of Tables**

Table 1: Goggles specifications .....44

Table 2: Voice command for UAV camera control.....51

Table 3: ROS message .....55

Table 4: Waypoints ROS message.....55

Table 5: Data type and Gazebo topics relationship with the interface.....58

## 1. Introduction

This deliverable extends the developments presented in the earlier deliverable [D4.1] and in particular it reports about the progress about the path planning and navigation support for hybrid robot autonomous operation, the human operation supporting function by augmented reality, and the mobile platform for remote control support and battery management.

Correspondingly, Section 2 reports about the developments concerning the planning algorithms of the movement of the hybrid robot (HR) and its associated navigation support. These algorithms (obstacle detection, reactive behaviours, etc.) pave the way for autonomous operation of the HR in its inspection tasks. More in particular, one development is about the generation of motion plans that allow autonomous motion along complete sequences of inspection points. Here, the obstacles included in the industrial environment maps are considered for the generation collision-free trajectories, where the hybrid motion capabilities (flying, rolling) are exploited to achieve optimal trajectories according to operation time or power consumption. As an addition, the motion planner is extended to guarantee safer trajectories for autonomous operation in presence of a dynamic behaviour of the system. Finally, navigation support enhances the level of autonomy offered by the system by implementing reactive behaviours in case the on-board sensors reveal the presence of obstacles not considered in the planning algorithms (not included in the pre-existing map, for example due to plan updates or repairs). Supplemental material about the above part is provided in the form of videos, which are listed in the dedicated appendix.

Further, Section 3 presents a 3D (three-dimensional) augmented reality system to be part of the human operation supporting functions. This new type of interface for the hybrid robot operator will enrich the information available during the operation visually showing the status of the hybrid robot and its mission. This information will be provided to the operator in such a way that they do not have to change its field of view and hence, they can always maintain a look to the hybrid robot and its operation at any time. The system uses Microsoft HoloLens goggles that allow mixed reality, basically combining the real world with a virtual environment where the physical and digital objects coexist both together and can even interact between them. The system supports sight, voice, and hand gestures for human interactions, and it is enriched by sounds and spatial mapping for immersion and control. HoloLens run on Windows 10 on which also runs Unity (used to e.g. create videogames). On the other side a ROS (robot operating system) module runs on Linux Ubuntu for communication with the unmanned aerial vehicle (UAV). Ros-Sharp on the Windows side is used to engage Unity with ROS.

Section 4 reports about the developments of the remote mobile support platform ground control station computer, including mapping support such as occupancy mapping (visual tracking odometry for the light control unit through ROS and point cloud measurements) and deep learning object detection system from RGBD (red, green, blue, depth) image, currently using available modules with custom solutions being considered.

Finally, Section 5 presents the developments about the mobile support platform accommodating all the needed components and realised as a pushable cart that can be transformed to an easily transferrable form (like a large bag). When opened, the cart has two trays. At the bottom, the electrical part box includes the AC-DC power supply unit (1 kW with input range of 85 - 265 V) and the main battery (a safe and reliable 8-cell LiFePO<sub>4</sub> battery pack). The total energy capacity is 2.5 kWh (with a voltage range of 20 to 28.8 V) and it is able to charge the HR several times (in situations where an AC power is not available) while simultaneously running the computer and communications

hardware. A microcontroller unit performs real time monitoring (equipped with an external OLED screen) and cuts off the battery outputs if the available energy level drops too low. A custom solution enables operating uninterruptedly when disconnected and connected from AC power. At the top of the cart, a detachable ABS plastic water-resistant box contains all the required components for communications (Ethernet and WiFi with internal antennas) and support functions related to HR operation (including database and navigation support), waterproof keyboard and mouse, a water and dust protected multi-touch screen, and waterproof HDMI, Ethernet and USB ports.

## 2. Path planning and navigation support for autonomous operation (T4.1)

This section presents the new advances concerning the path planning algorithms for autonomous hybrid motion that have been made in the framework of Task 4.1 “3D navigation support” and Task 4.3 “Mobile ground support”. The final objective of these tasks is the generation of motion plans that allow the hybrid robot to inspect, in a safe and efficient manner, by means of a set of measurements at points localized in different pipe arrays. For that purpose, the robot can navigate through the environment either flying or rolling on the pipes.

### 2.1. Introduction

This initial subsection introduces the work-lines that have undergone significant progress. Furthermore, the navigation architecture where these new developments have to be integrated, is also retrieved here from D4.1 for the sake of completeness. This general overview will bring a better assessment of the autonomy level that can be achieved in the final stage of the project. Once this overall frame has been clarified, subsequent subsections will present the particular developments covered by this deliverable.

The related research lines covered here are the following:

- Generation of motion plans that allow autonomous motion along complete sequences of inspection points.
  - Consideration of the obstacles included in the maps of the industrial environments to generate trajectories free of collisions.
  - Consideration of the hybrid motion capabilities (flying, rolling) offered by the hybrid robot (HR) to generate optimal trajectories according to different indices (operation time, energy consumption).
- Extensions of motion planner to guarantee safer trajectories for autonomous operation
  - Consideration of dynamic behaviour of the system.
- Navigation support to enhance the level of autonomy offered by the system
  - Implementation of reactive behaviours in case the on-board sensors reveal the presence of obstacles not considered in the planning algorithms.

The first development corresponds to a consolidated implementation of the planning algorithm for autonomous hybrid motion that was introduced in D4.1. Additionally, further advances in the reactive capabilities of the planner have endowed the system with full hybrid reactivity; that is the possibility of switching between pipe and air to avoid obstacles. Finally, an extension of the planning algorithm for robust collision avoidance based on Dynamics Awareness has also been derived. All of these new functionalities reinforce considerably the level of robustness and autonomy offered by the system.

The remaining items of the work-lines that will be addressed in deliverable D4.3 are:

- Extensions of motion planner to guarantee safer trajectories for autonomous operation
  - Consideration of aerodynamic effects produced by the proximity of the elements that require inspection (horizontal arrays of pipes, vertical pipes / tanks, etc.).
- Navigation support to enhance the level of autonomy offered by the system
  - Implementation of re-planning strategies in case that modifications of the inspection tasks arise during the execution of the global mission.

## 2.2. Description of the hybrid robot

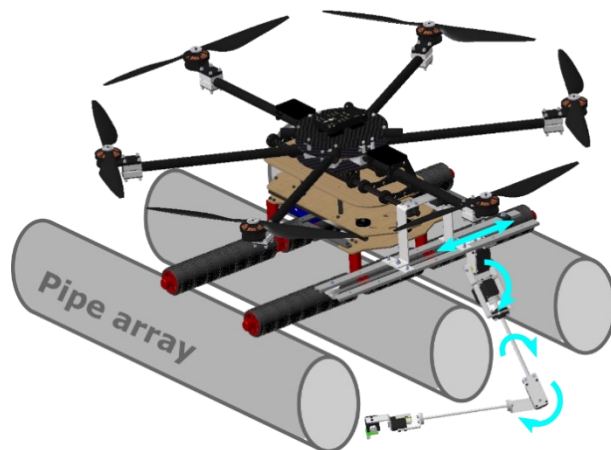
In general terms, the hybrid systems considered in HYFLIERS are flying-rolling inspection platforms capable of landing and moving along the pipe arrays to perform the inspection without wasting energy for the propellers. In this way, the robot overcomes the limitations of flying robots in terms of operation time and position accuracy when applied to industrial inspection and maintenance tasks.

The planning strategies included in this deliverable are based on general design principles and hence can be applied to the different hybrid systems developed in HYFLIERS. In this document, one particular setup of these hybrid robots has been used to guide the presentation of the proposed methods (see Figure 1). This platform has also been used for the first experimental tests with the developed algorithms for path planning and navigation support that have been included in Section 2.8.



**Figure 1.** Hybrid rolling-aerial platform with manipulation capabilities.

Figure 1 brings a general overview of the platform. The flying-rolling robot consists of a standard hexarotor platform, a 5-DOF (degrees of freedom) robotic arm supported by a 1-DOF linear guide system, and a customised rolling base that replaces the conventional landing gear.





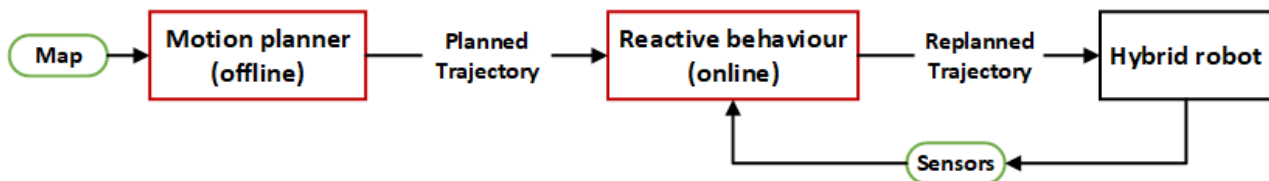
**Figure 2.** Hybrid rolling-aerial platform integrating the 5-DOF robotic arm supported by a 1-DOF linear guide system.

As it can be seen in Figure 2, the rolling base is designed to facilitate the displacement of the robot along pipe arrays. Two rollers have been integrated in order to control these displacements. Moreover, the design of the robotic manipulator enables the access to the contour of the pipes, either the side part or the lower part, regions where the defects are usually concentrated. The prototype consists of a lightweight robotic arm with three joints for end-effector positioning, two joints for wrist orientation, and an actuated linear guide system.

Since the linear actuator will be transversely moving along the gaps existing between the pipes, the most suitable configuration for inspection will be a centered position of the HR with respect to the parallel array of pipes that is under inspection, as suggested in Figure 2. This central location increases the stability and accuracy of the operation. Consequently, the planner will assume the same configuration within its operation.

### 2.3. Architecture of the navigation system

The navigation architecture proposed in Figure 3 to meet the requirements of hybrid planning capabilities is composed of two main subsystems, the *Motion planner subsystem* and the *Reactive behaviour subsystem*.



**Figure 3.** Overall navigation architecture.

A synthetic description of these subsystems is provided below:

- *Motion planner subsystem (offline)*

Given a map of the environment, a description of its navigation constraints, and a set of inspection points, this block is in charge of generating a global trajectory that allows autonomous hybrid navigation through the inspection targets. The resultant trajectory will be given to the hybrid robot as control reference. The Motion Planner for Hybrid Robots (MP-HR) presented in Section 2.4 corresponds to the implementation of these basic functionalities of the planner.

This subsystem will also include any extension of the motion planner that could bring more safety and robustness to the system operation. The consideration of the dynamics and aerodynamics of the hybrid robot during the expansion of the planning tree would therefore be framed within this subsystem. The Motion Planner for Hybrid Robots with Dynamics Awareness (MP-HR-DA) presented in Section 2.5 corresponds to the implementation of the functionality that allows considering the dynamics within the planning process.

- *Reactive behaviour subsystem (online)*

During the execution of the planned trajectory, the *Reactive behaviour subsystem* will be continuously monitoring the information delivered by on-board sensors. If any obstacle not considered in the motion plan appeared, or any dynamic change concerning elements included in the map arose, this subsystem would check if the latter could provoke a collision. In that case, the *Reactive behaviour subsystem* will locally modify the global trajectory for collision avoidance, minimising as much as possible the deviations with respect to the global plan. The reactive approach presented in Section 2.6 corresponds to the implementation of this functionality.

Additionally, it may occur that the global trajectory generated by the *Motion planner subsystem* becomes outdated during the execution of the global mission. This could be motivated by changes in the inspection tasks that may arise during the mission (e.g. re-inspection of certain areas could be necessary after online post-processing of data gathered in previous phases of the mission). In this case, the *Reactive behaviour subsystem* will use re-planning strategies that adapt the global trajectory to the new requirements.

## 2.4. Planning algorithm for autonomous hybrid motion

In order to generate a trajectory that steers the hybrid system to perform its operation in a safe and efficient manner, a novel motion planning methodology that takes advantage of the flying and rolling capabilities offered by the system has been developed.

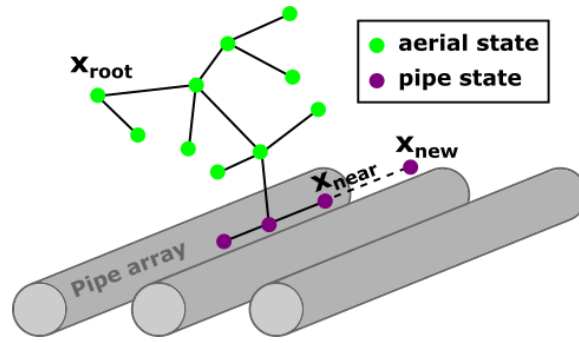
Given a map of the environment and a set of inspection points located in the different pipes, the developed planning method computes the trajectory that allows autonomous hybrid navigation through the inspection targets to complete the mission. Afterwards, the resultant trajectory will be given to the hybrid robot as control reference.

Moreover, the algorithm brings the possibility to optimise metrics in the generated trajectories like operation time or energy consumption. This is a remarkable feature since it allows better adaptation of the plans to specific missions. For instance, urgent inspections motivated by emergency situations would benefit from optimising the time operation, whereas routine inspections could better fit with energy optimization in order to maximize the coverage.

The developed motion planner, hereinafter referred to as MP-HR (Motion Planner for Hybrid Robots), is built over the basis of the RRT\* algorithm (Optimal version of the Rapidly-exploring Random Tree [KarFra2011]). However, its algorithmic modules have been completely adapted within the framework of HYFLIERS project in order to efficiently integrate the hybrid nature of the system into the motion planning technique [SuaEtal2020]. Next subsections detail the main novelties.

### 2.4.1. Hybrid planning space

The exploitation of the hybrid capabilities of the system makes it necessary to consider jointly aerial and pipe states within the planner operation. Figure 4 illustrates this new hybrid paradigm during the expansion of the planning method. As it can be seen, the pipe states are restricted to the upper part of the pipes conforming the arrays since this is the location from which the hybrid robot should operate, either rolling along the pipe or deploying the robotic arm.



**Figure 4.** Scheme with both aerial and pipe states during the hybrid expansion of the proposed MP-HR planning algorithm. Aerial states in green colour and pipe states in purple colour.

The variables considered within this hybrid planning space, whether in air states or in pipe states, include the system position -coordinates  $x, y, z$ - in an Earth-fixed frame as well as the heading orientation -yaw angle  $\psi$ - with respect to the same frame. The exploration of these subspaces -position and yaw orientation- can be addressed sequentially to minimise the computational load. The global approach followed to integrate these sequential explorations will guarantee in any case the consistence of the global exploration. That is, the fulfilment of the position and yaw orientation associated with each inspection point in the generated trajectory. This strategy eases considerably the reactive capabilities presented in subsequent sections.

The next subsection introduces the specific characteristics of the exploration of position sub-space. The following subsection is devoted to the particularities of the yaw exploration, including the procedure to bring into line the trajectory in yaw subspace with that previously generated for position subspace, at the same time that the yaw orientation associated with each inspection point is guaranteed.

### ***Position sub-space***

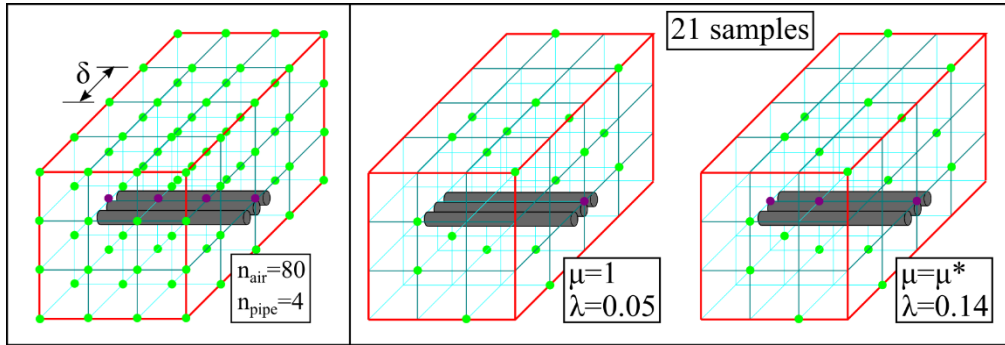
In standard RRT\*-based algorithms, the exploration of the planning space is based on uniform sampling. However, this approach is not suitable for the hybrid planning space considered in HYFLIERS since the ratio between air and pipe nodes is very prone to be unbalanced. This situation could provoke that pipe nodes are not sufficiently explored and, as a result, that the hybrid capabilities of the system are not fully exploited. This subsection proposes a smart process of discretisation and sampling to tackle this problem.

The discretisation provides a two-fold benefit. Firstly, it eases the smart sampling of the hybrid planning space that will be described later to counteract the air-pipe unbalance. Secondly, it contributes to bound the computation time. This will be a valuable feature in the context of the adaptation of the planning algorithm to implement reactive behaviours that is presented in Section 2.6. The discretisation has been performed uniformly with the same resolution  $\delta$  for both the 3D aerial space and the 1D operation lines in the upper part of the pipes.

The smart sampling is the part of the process that allows to adjust the original ratio between air and pipe nodes that is typically very unbalanced. The corrective parameter  $\mu$  will be in charge of this purpose. It will be introduced to steer the sampling process in such a way that the ratio between air and pipe nodes is transformed into the new ratio  $\lambda \in (0,1)$ :

$$\lambda = \mu \frac{n_{pipe}}{n_{pipe} + n_{air}} \quad (1)$$

Figure 5 illustrates previous concepts in an example scenario. As it can be seen, when  $\mu = 1$  the sampling of aerial states and pipe states is uniform (as in the standard approach) and the illustrative sample shown in the figure reveals the unbalanced condition anticipated in this subsection (only 1 of 21 samples corresponds to pipes). In contrast, when the sampling is steered by a correcting parameter  $\mu = \mu^*$ , whose selection takes into account the density of pipe arrays, the resultant sample is more balanced (for example in Figure 5, 3 of 21 samples correspond to pipes) and hence the hybrid capabilities of the system are better explored.



**Figure 5.** Effect of the weighting parameter  $\mu$  in the process of discretisation and sampling of the hybrid planning space: the complete set of  $n_{pipe}$  pipe and  $n_{air}$  aerial states corresponding to the discretisation (left), the selected subset with uniform sampling (centre) and the selected subset corresponding to the sampling with compensating effect (right) that results in a more balanced distribution.

### *Yaw sub-space*

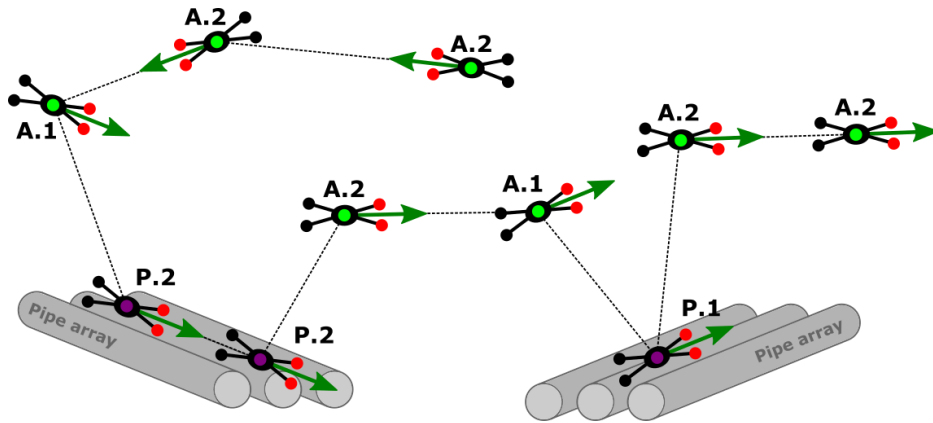
The algorithm to derive the yaw trajectory will enforce two directives when selecting the yaw value for a certain node. Firstly, the alignment with the displacement directions that can be inferred from the position trajectory generated for the hybrid robot. Secondly, the alignment with the orientation of the pipes where the hybrid robot has to land and operate. As it will be seen later, these objectives could come into conflict with each other and the algorithm should provide mechanisms to solve these situations.

The resultant criteria to select the yaw values for each kind of node are presented below. They are also graphically illustrated in Figure 6. As it can be anticipated from this figure, the branches that join the yaw values selected for the different nodes will be built in such a way that smooth transitions at constant velocity are guaranteed.

- P)** In pipe nodes, the yaw angle is enforced to be aligned with the orientation of the pipe. Since two orientations fulfilling this requirement can be considered for each pipe, the final selection is based on the following considerations:

- P.1)** In case that the pipe node is surrounded by two aerial nodes (landing, inspection on the landing point and finally, take-off), the selected orientation will be the one that minimises the aggregated angle variation produced along the two transitions air-pipe and pipe-air.

- P.2) In the remaining cases (another pipe node before or after), the selected orientation will be given by the direction of the rolling movement that allows the system to reach, or to arrive from, that other pipe node.
- A) In aerial nodes, the final selection presents again two branches:
  - A.1) When the aerial node is associated with landing transitions, the yaw angle is equal to that of the following pipe node. This allows a safer landing with constant yaw during the complete manoeuvre.
  - A.2) In the remaining cases (whether in take-off transitions or in free flight), the yaw angle is aligned with the direction of the vector that connects the node under consideration with the next node in the position trajectory. This ensures that the robot is always flying in areas that have been covered by the perception sensors located on-board since they are typically forward-facing.



**Figure 6.** Illustrative examples of the criteria to select yaw angles for the different types of nodes in the hybrid planning space.

#### 2.4.2. Optimization capabilities in hybrid planning: operation time and energy consumption

##### *Description of the optimization logic*

For the optimization of the system operation at the planning level, two different cost functions that allow better adaptation of the plans to specific missions have been defined: the operation time and the energy consumption. For instance, urgent inspections motivated by emergency situations would benefit from optimising the time operation, whereas routine inspections could better fit with energy optimization in order to maximise the coverage.

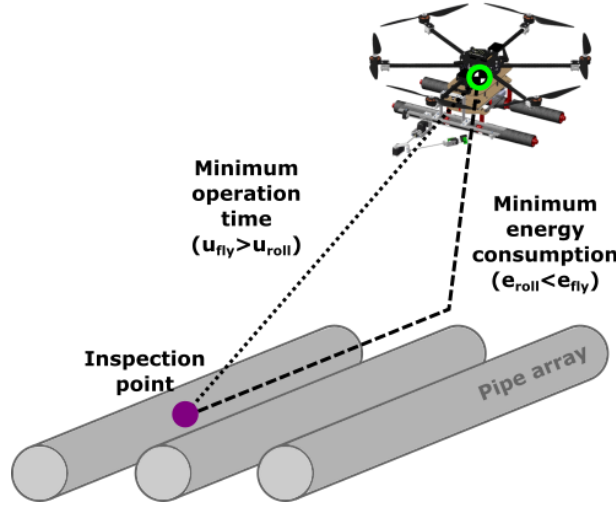
These cost functions consider the hybrid capabilities of the robot through the associated velocities  $u_{roll}$ ,  $u_{fly}$ ,  $u_{tran}$  and energy consumptions per unit of displacement  $e_{roll}$ ,  $e_{fly}$ ,  $e_{tran}$  when the system is rolling, flying between aerial nodes or performing a hybrid transition (landing/take-off). Thus, the increment in cost  $\Delta c$  associated with a branch from node  $i$  to node  $j$  when minimising the operation time  $T$  or the energy consumption  $E$  can be defined, respectively, as:

$$\Delta c_{i,j}^T = \frac{\|p_i - p_j\|}{u_k} \quad k = roll, fly, tran \quad (2)$$

$$\Delta c_{i,j}^E = \|p_i - p_j\| e_k \quad k = roll, fly, tran \quad (3)$$

where  $k$  denotes the manoeuvre associated with the branch: rolling (*roll*), flying (*fly*) or hybrid transition (*tran*) either for landing or take-off. Since a normal operation complies with  $u_{fly} > u_{roll}$

and  $e_{roll} < e_{fly}$ , the planner will prioritise flying branches when minimising operation time  $\Delta c_{i,j}^T$  and rolling branches for energy consumption  $\Delta c_{i,j}^E$ . These basic behaviours are illustrated in Figure 7. The derivation of an approximate estimation for the energy consumptions per unit of displacement that are associated with these basic behaviours, that is  $e_{roll}$  and  $e_{fly}$ , will be addressed in next subsection.



**Figure 7.** Basic behaviours corresponding to the optimization of the proposed cost functions. Flying is prioritised over rolling when minimising the operation time  $T$  ( $u_{fly} > u_{roll}$ ) whilst rolling is prioritised over flying when minimising the energy consumption  $E$  ( $e_{roll} < e_{fly}$ ).

Additionally, a smoothing technique is applied at the end of the planning process. This technique follows a classic approach but incorporates some adaptations to operate in the hybrid planning space. In particular, only aerial nodes could be considered for this smoothing process. Pipe nodes are not processed because the 1D characterization of the pipe arrays already assures sub-optimal trajectories on the pipes and, even more important, because the elimination of transition pipe nodes could negatively affect the optimality of the global trajectory.

### ***Estimation of the energy consumption***

This section is devoted to calculate an approximated estimation of the ratios of energy consumption  $e_{roll}, e_{fly}, e_{tran}$  per unit of displacement that are included in Equation (3).

In general, the electrical energy consumed by an aerial robot while rolling, flying between aerial nodes or performing a hybrid transition (landing/take-off)  $E_{roll}, E_{fly}, E_{tran}$  can be defined as:

$$E_{roll,fly,tran} = \int_0^{\Delta t} I(t)dt \quad (4)$$

where  $I(t)$  is the demanded current over time. Hereinafter, two different models have been derived to estimate the function  $I(t)$  that allows to solve Equation (4): one for rolling phases and one for flying phases. Both of them are based on energetic principles. However, while the first focuses on the analysis of the mechanical power, the second makes use of the concept of the aerodynamic power.

### **Energy consumption in rolling phases**

According to the law of conservation of energy, the electric power  $P_e$  generated by the robot batteries will be transformed to the mechanical power  $P_m$  that will be used to roll on the pipes, with certain power losses quantified through the efficiency parameter  $\eta_m$ . Consequently:

$$\begin{aligned} P_m &= \eta_m P_e = \eta_m V_{roll} I \\ I &= \frac{P_m}{\eta_m V_{roll}} \end{aligned} \quad (5)$$

where  $V_{roll}$  is the battery voltage and  $P_e = V_{roll} I$  by definition. Additionally, considering that the mechanical power  $P_m$  will be used to counteract the friction force  $F_r$  between the robot wheels and the pipes, this can be modelled as:

$$P_m = F_r u_{roll} = \mu_f m g u_{roll} \quad (6)$$

where  $\mu_f$  is the coefficient of friction,  $m$  is the mass of the robot,  $g$  is the gravity acceleration,  $P_m = F_r u_{roll}$  by definition, and  $F_r = \mu_f m g$  is the used friction model. Substituting, Equation (6) in Equation (5):

$$I = \frac{\mu_f m g u_{roll}}{\eta_m V_{roll}} \quad (7)$$

At this point, all the parameter in  $I(t)$  can be considered constant except  $V_{roll}$ . However,  $V_{roll}$  can be approximated, as a first approach, by its mean value between initial and final voltages, which are usually well defined. Integrating Equation (4) after substituting Equation (7):

$$E_{roll} = \frac{\mu_f m g u_{roll}}{\eta_m V_{roll}} \Delta t \quad (8)$$

Considering constant rolling velocity between nodes  $i$  and  $j$ :

$$u_{roll} = \frac{\|\mathbf{p}_i - \mathbf{p}_j\|}{\Delta t} \quad (9)$$

and substituting Equation (9) in Equation (8):

$$E_{roll} = \frac{\mu_f m g}{\eta_m V_{roll}} \|\mathbf{p}_i - \mathbf{p}_j\| \quad (10)$$

Finally, comparing Equation (10) with Equation (3):

$$e_{roll} = \frac{\mu_f m g}{\eta_m V_{roll}} \quad (11)$$

#### Energy consumption in flying phases

According to the law of conservation of energy, the electric power  $P_e$  generated by the robot batteries will be transformed to the aerodynamic power  $P_a$  that will be used to lift the robot, with certain power losses quantified through the efficiency parameter  $\eta_a$ . Consequently:

$$\begin{aligned} P_a &= \eta_a P_e = \eta_a V_{fly} I \\ I &= \frac{P_a}{\eta_a V_{fly}} \end{aligned} \quad (12)$$

Additionally, according to the Momentum Theory of Fluid Dynamics, the aerodynamic power  $P_a$  of a multicopter with  $n_r$  rotors of radius  $r_r$  can be modelled as:

$$P_a = \frac{(mg)^{3/2}}{\sqrt{2\rho_a \pi n_r r_r^2}} \quad (13)$$

where  $\rho_a$  is the air density. This model is derived for aerial vehicles while hovering. However, its validity has also been demonstrated, as a first approach, for smooth flights [DorEtal2017]. In any case, this model supposes an overestimation of the aerodynamic power, which is slightly lower when the vehicle is moving. The latter allows an estimation on the side of safety. Substituting, Equation (13) in Equation (12):

$$I = \frac{(mg)^{3/2}}{\eta_a V_{fly} \sqrt{2\rho_a \pi n_r r_r^2}} \quad (14)$$

Integrating Equation (4) after substituting Equation (14) and assuming again a constant value of  $I$  following the same reasoning as in the derivation for rolling segments:

$$E_{fly} = \frac{(mg)^{3/2}}{\eta_a V_{fly} \sqrt{2\rho_a \pi n_r r_r^2}} \Delta t \quad (15)$$

Considering constant flight velocity between nodes  $i$  and  $j$ :

$$u_{fly} = \frac{\|\mathbf{p}_i - \mathbf{p}_j\|}{\Delta t} \quad (16)$$

and substituting Equation (16) in Equation (15):

$$E_{fly} = \frac{(mg)^{3/2}}{\eta_a V_{fly} u_{fly} \sqrt{2\rho_a \pi n_r r_r^2}} \|\mathbf{p}_i - \mathbf{p}_j\| \quad (17)$$

Finally, comparing Equation (17) with Equation (3):

$$e_{fly} = \frac{(mg)^{3/2}}{\eta_a V_{fly} u_{fly} \sqrt{2\rho_a \pi n_r r_r^2}} \quad (18)$$

The same approach also allows the estimation of  $e_{tran}$ :

$$e_{tran} = \frac{(mg)^{3/2}}{\eta_a V_{fly} u_{tran} \sqrt{2\rho_a \pi n_r r_r^2}} \quad (19)$$

### 2.4.3. Robustness in hybrid planning: air-pipe and pipe-pipe transitions

In standard derivations of RRT\*-based planners, the admissibility of a new branch is analysed based on possible collisions of the vehicle with obstacles within the planning space.

In HYFLIERS, the hybrid operational environment of the robot introduces additional aspects that can also affect whether or not it is possible to admit a new branch. These new aspects arise from the need to switch between air and pipes, or between different pipe structures connected to each other. The integration of these new situations into the planning space requires specific conditions with regard to the corresponding transitions that guarantees safe operation at all times. Therefore, the tree expansion will now account not only for the collisions but also for these conditions that guarantee robustness in hybrid planning. In order to better encompass all these aspects, the admissibility of new branches will hereinafter be denoted by the new term feasibility. A more detailed description of the application of this new concept is presented below.



- Feasibility of transitions between aerial and pipe states.

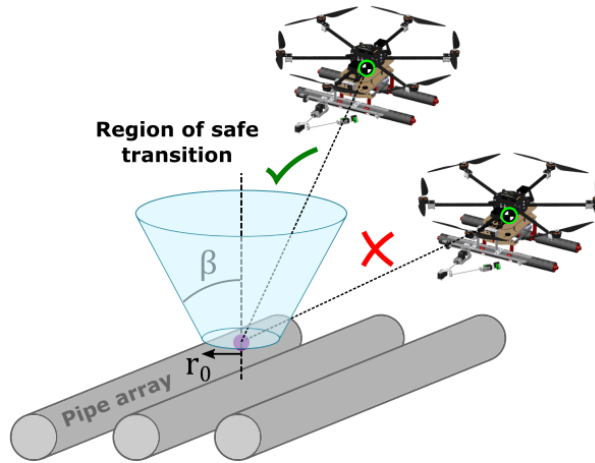
Within planning framework, take-off and landing manoeuvres correspond to branches that link aerial and pipe nodes. These manoeuvres will be considered safe when the reference trajectories associated with those branches fall within a volume that is compatible with the dynamic constraints of the vehicle. This region will be hereinafter referred to as *safety region for hybrid transitions*.

More in detail, these safety regions will correspond to conical truncated volumes with aperture semi-angle  $\beta$  and radius  $r_0$  at the intersection with the pipe (see Figure 8). Please note that the velocity imposed to these hybrid transitions will be smaller than the corresponding reference to the normal flight; that is,  $u_{tran} < u_{fly}$ .

Mathematically, these feasibility conditions are given by:

$$\sqrt{(x_{air} - x_{pipe})^2 + (y_{air} - y_{pipe})^2} < r_0 + \tan \beta (z_{air} - z_{pipe}) \quad (20)$$

where  $(x, y, z)_{air,pipe}$  are the position coordinates of the aerial and pipe nodes that form the transition branch that is under analysis.



**Figure 8.** Feasibility checking in transitions between aerial nodes and pipe nodes. Transitions taking place out of the region of safe transition are discarded.

- Feasibility of transitions between pipe states.

During the expansion of the search tree, a potential new branch linking two pipe nodes can be classified into two categories: branches linking pipe nodes in the same pipe array or branches linking pipe nodes in different pipe arrays. Whereas in the first group rolling transitions can be assumed feasible, in the second group it is important to take into account whether both nodes belong to connected or unconnected pipe arrays. In the first case, the transition between arrays can be executed through the intersection region. However, in the second case, the potential new branch must be discarded because it is not possible to link two nodes in different unconnected pipe arrays without an intermediate transition through aerial nodes.

All these situations can be characterized using a connection matrix  $\mathbf{\Gamma} \in \mathcal{B}^{N \times N}$  where  $N$  is the number of pipe arrays in the environment. The different elements  $\gamma_{i,j}$  of this boolean matrix takes the logical values  $\{0, 1\}$  and represent the connection between pipe arrays  $i$  and  $j$ . Consequently,  $\mathbf{\Gamma}$  is a symmetric matrix with ones in the main diagonal:

$$\mathbf{\Gamma} = \begin{bmatrix} 1 & \gamma_{1,2} & \gamma_{1,3} & \dots & \gamma_{1,N} \\ \gamma_{1,2} & 1 & \gamma_{2,3} & \dots & \gamma_{2,N} \\ \gamma_{1,3} & \gamma_{2,3} & 1 & \dots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \gamma_{N-1,N} \\ \gamma_{1,N} & \gamma_{2,N} & \dots & \gamma_{N-1,N} & 1 \end{bmatrix} \quad \gamma_{i,j} = \{0, 1\} \quad (21)$$

This characterization allows to establish a simple criterion to analyse feasibility in pipe-to-pipe transitions by simply checking the logical value of  $\mathbf{\Gamma}$  for the pipe arrays associated with a specific branch.

## 2.5. Extension of planning algorithm based on Dynamics Awareness for safer operation

This section proposes an additional capability for the basic planning algorithm for autonomous hybrid motion presented in Section 2.4. An enhancement that pursues safer operation in highly cluttered areas: Dynamics Awareness. The awareness of the flying dynamics of the system within the internal calculations of the motion planner allows addressing scenarios where the obstacle proximity narrows considerably the safety operation margins. In these conditions, the correlation between the planned and the finally executed trajectory is even more critical.

Since the proposed extension relies on the knowledge of the system dynamics, the implementation of closed-loop simulations of the controlled system when following the commanded trajectories are required for the new operation basis of the motion planner. Accordingly, the dynamic model of the hybrid system while flying, as well as its associated controller, will be presented in next subsection.

### 2.5.1. Modelling and control

The behaviour of the hybrid robot will be described by means of an elaborated mechanical model. Kane's method [KanLev1985] has been commonly used for modelling multi-body systems since it holds some unique advantages when addressing complex robotic systems like the hybrid robot. Of the latter, the most remarkable are the derivation of a compact model in first order differential equations that are uncoupled in the generalized velocity derivatives as well as the easy computerization and the computational efficiency of the resulting equations of motion.

The configuration variables selected as system generalized coordinates are the position of the centre of mass  $\mathbf{p} = [x, y, z]^T$  in an Earth-fixed frame and the orientation angles  $\boldsymbol{\eta} = [\phi, \theta, \psi]^T$  (roll, pitch, yaw). These configuration variables lead to the following kinematic differential equations where  $u_x, u_y, u_z, u_\phi, u_\theta, u_\psi$  are the generalized velocities associated with the configuration variables:

$$\begin{aligned} [\dot{x}, \dot{y}, \dot{z}]^T &= [u_x, u_y, u_z]^T \\ \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} &= \begin{bmatrix} (u_\phi \cos(\psi) - u_\theta \sin(\psi)) / \cos(\theta) \\ u_\phi \sin(\psi) + u_\theta \cos(\psi) \\ u_\psi - \tan(\theta) (u_\phi \cos(\psi) - u_\theta \sin(\psi)) \end{bmatrix} \end{aligned} \quad (22)$$

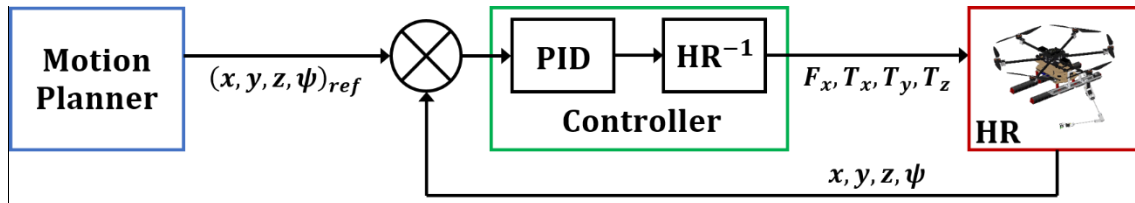
Regarding forces and torques exerted on the system, the rotors generate a resultant lifting force  $F_x$  applied at the multirotor center of mass as well as a torque  $\mathbf{T} = [T_x, T_y, T_z]^T$  applied to the rigid body.

Application of Kane's method leads to the following dynamic differential equations for translation and rotation:

$$\begin{bmatrix} m\ddot{u}_x \\ m\ddot{u}_y \\ m\ddot{u}_z \\ I_{xx}\dot{u}_\phi \\ I_{yy}\dot{u}_\theta \\ I_{zz}\dot{u}_\psi \end{bmatrix} = \begin{bmatrix} F_x \sin(\theta) \\ -F_x \sin(\phi) \cos(\theta) \\ F_x \cos(\phi) \cos(\theta) - mg \\ T_x - (I_{zz} - I_{yy})u_\theta u_\psi \\ T_y + (I_{zz} - I_{xx})u_\phi u_\psi \\ T_z - (I_{yy} - I_{xx})u_\phi u_\theta \end{bmatrix} \quad (23)$$

where  $m$  is the mass of the hybrid robot,  $I_{xx}, I_{yy}, I_{zz}$  are its principal moment of inertia and  $g$  is the gravity acceleration. For more details about this modelling approach based on Kane's method, please refer to [CabEtal2018].

Concerning the on-board controller, a control scheme that makes use of nonlinear strategies based on model inversion has been derived to provide the hybrid system with the capacity of executing navigation manoeuvres. The approach, schematised in Figure 9, consists in linearising the system through model inversion ( $HR^{-1}$ ) and applying Proportional-Integral-Derivative (PID) control laws to the resultant dynamics. For more details about the control approach, please refer again to [CabEtal2018].



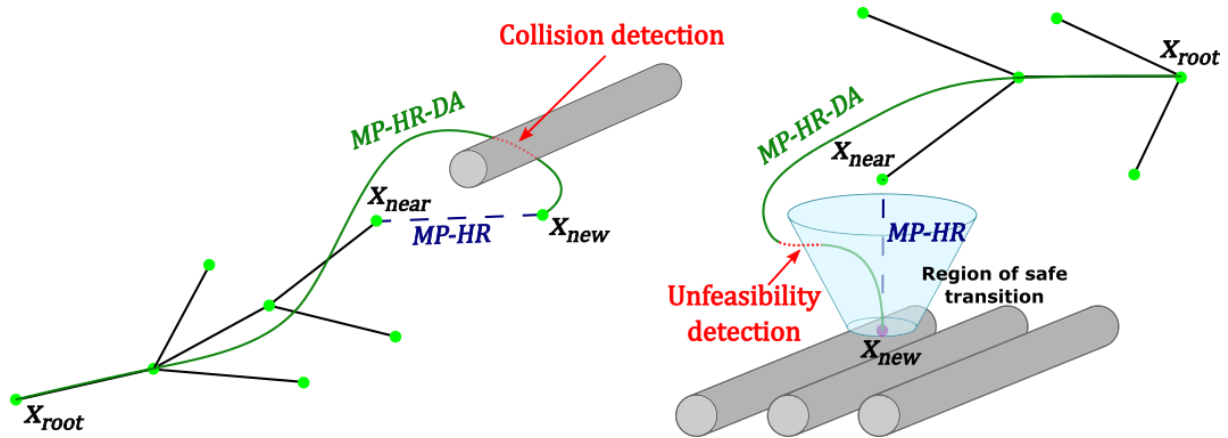
**Figure 9.** Block diagram of the control scheme.

### 2.5.2. Dynamics Awareness

The developments presented in Section 2.4 guarantee planned trajectories that are kinematically feasible as well as efficient in terms of operation time or energy consumption. However, the complex dynamics that govern aerial systems requires further attention since they provoke considerable differences between planned and executed trajectories. The collision risk introduced by these differences is especially critical for cluttered environments like industrial sites.

In order to overcome this undesired influence of the dynamics, the Motion Planner for Hybrid Robots (MP-HR) previously presented is transformed into a more advanced algorithm that incorporates Dynamics Awareness (MP-HR-DA). This extension is inspired by a previous contribution [CabEtal2018], where Dynamics Awareness was applied to derive robust planning strategies for inspection tasks using aerial manipulators. Following the same approach, the expansion of the search tree for aerial branches in the new MP-HR-DA will be based on the behaviour of the controlled system, which means that the feasibility of these branches is checked through closed-loop simulations of the controlled system (green solid lines in Figure 10) instead of using geometrical interpolation between states (blue dashed lines in Figure 10). Moreover, the scope for this dynamical analysis of the tree extension is a root-to-candidate validation. Thus, not only the dynamical feasibility of the possible new branch reaching the candidate node (from node  $\mathbf{x}_{near}$  to node  $\mathbf{x}_{new}$  in Figure 10) is analysed, but also the complete path from the tree-root node  $\mathbf{x}_{root}$ .

This whole approach guarantees that the resultant planned trajectories are compatible with the dynamics constraints previously mentioned and hence reinforces considerably the level of robustness and autonomy offered by the system.



**Figure 10.** Operation basis of the MP-HR-DA algorithm: feasibility analysis of new air-air branch (left) and feasibility analysis of new air-pipe branch (right). In contrast to the geometrical interpolation (blue dashed lines associated with MP-HR), the dynamic simulation of the system (green solid lines associated with MP-HR-DA), which is closer to the performance of the real system, leads to unsafe conditions. Consequently, the potential new nodes  $x_{new}$  should be discarded in both cases.

## 2.6. Navigation support to reinforce system autonomy

The industrial scenarios considered in the project are in certain cases highly-cluttered environments -e.g. an industrial refinery with many pipe arrays that intersect with each other as in Figure 11- and it is very likely that the map provided to the planner do not include all the potential obstacles that the HR will come across in the inspections.



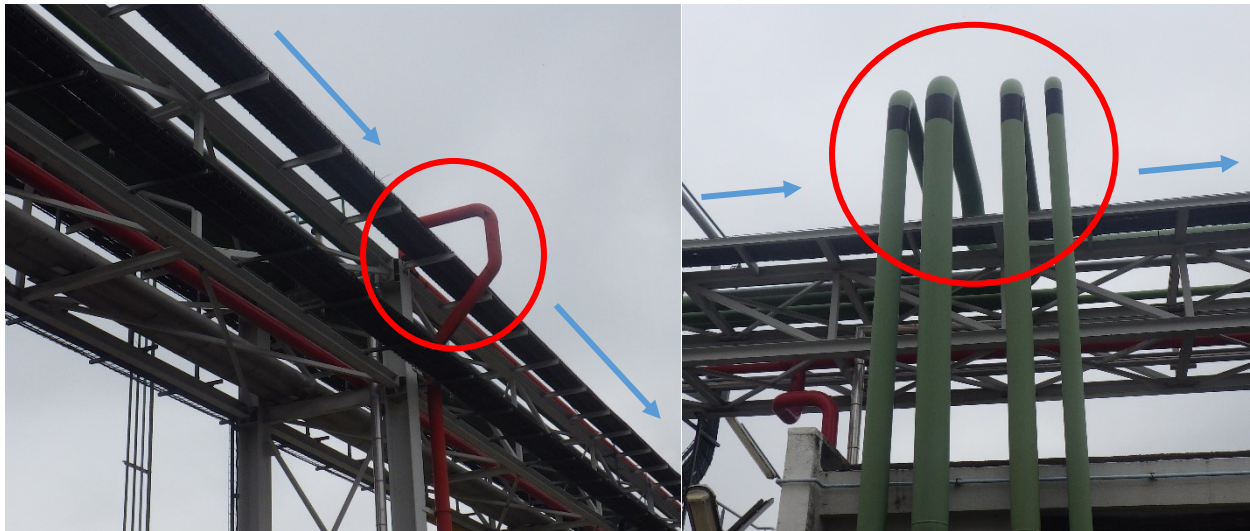
**Figure 11.** Highly-cluttered industrial environment with many pipe arrays.

In order to address this uncertainty, the on-board sensors should be capable of detecting the unexpected elements. This detection in real-time will allow to locally modify the global trajectory for

collision avoidance. The motion planner plays an important role in this reaction since it is in charge of that local re-planning.

This kind of reactive behaviours was firstly explored in D4.1. However, in that case, the reactive response was considered under standard conditions for aerial systems; that is, in flight segments. This deliverable explores the hybrid reactivity that is specific to HYFLIERS project. In other words, the unmapped obstacles will affect a rolling path on the pipes and hence the reactive avoidance will imply switching from pipe to air. After that, the system shall return to the pipe in order to keep following the original global inspection plan. This capacity of hybrid reaction will reinforce the level of robustness and autonomy offered by the system.

Figure 12 represents two scenarios that benefit from hybrid reactivity. In both cases, the pipes circled in red would not have been included in the initial map provided to the planner. This is a realistic assumption since it is difficult to keep updated the map of the pipes after the frequent operations of repairing and substitution of the piping layout. As a result, the generated trajectory would command the HR to roll along the main arrays (blue arrows), which would produce a collision with the pipes circled in red. The hybrid reactivity presented in this section should allow the system switching from pipe to air to avoid the unmapped pipes and, after that, from air to pipe, to continue with its navigation.



**Figure 12.** Example of application scenarios for hybrid reactivity.

In case that the HR has to enable the reactive response and to locally modify the global trajectory for collision avoidance, the same algorithmic basis of the global planner will be used. However, this new scenario requires lower computation times for assuring real-time reactions. Although the reduction of the planning space associated with the local nature of the reactive behaviour has contributed to satisfy this requirement, more elaborated approaches like the usage of specific tools to improve code efficiency, are being considered to better meet the real-time requirements of the system.

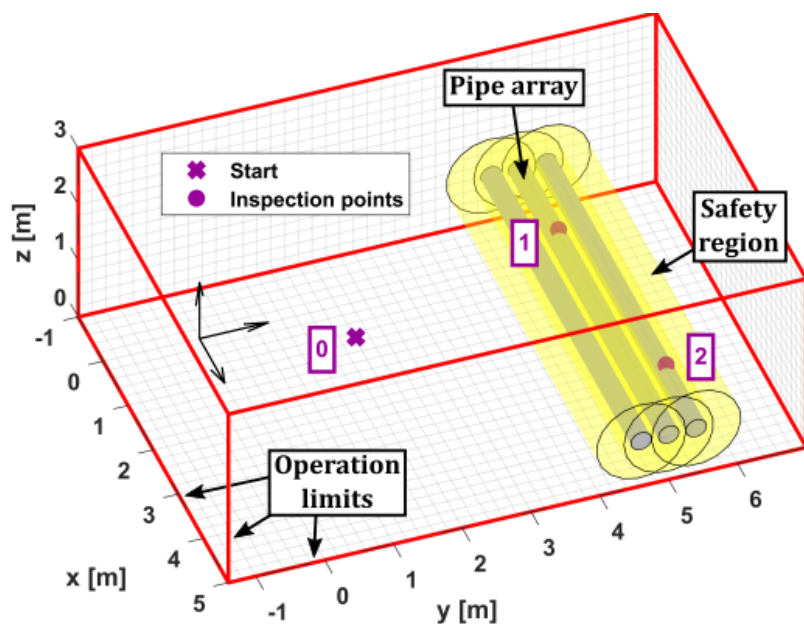
## 2.7. Simulation results

In order to demonstrate the validity of the planning strategies presented in previous sections, the algorithms will be tested through simulation in realistic industrial scenarios. All these scenarios require capabilities of hybrid motion to perform safely the desired operations in the associated cluttered environments.



The expected results will be inspection trajectories that allow navigating through the provided sequence of inspection points while avoiding the pipes and other restricted zones (operation limits, etc.). Additional requirements such as metrics optimization or reactive behaviours will be introduced in the corresponding scenarios.

When illustrating graphically the scenarios, certain common criteria have been followed. Figure 13 shows an example scenario to illustrate these criteria. Grey elements correspond to the existing pipe structures whereas the yellow volumes denote the safety regions associated with these pipe structures whose violations will be treated as collisions. In particular, these safety regions define the non-allowable regions for the displacement of the centre of mass of the HR. The purple points are reserved to highlight the inspection points while purple crosses represent the initial and final position of the HR. All these reference points are numbered according to the inspection sequence (0, 1, 2, ..., 0). Finally, the red lines stand for the operational limits of the HR system for a particular inspection mission.



**Figure 13.** Example scenario illustrating the criteria followed to represent the different application scenarios.

In general terms, the validation tests have been carried out following the same sequence. Firstly, the algorithm under analysis has been executed to generate the motion plan. Then, the resultant plan has been provided to the closed-loop simulation of the controlled hybrid system. The objective is therefore to analyse not only the planned trajectories, but also the closed-loop behaviour when the system follows such trajectories.

The simulation work has been performed in a Matlab-Simulink framework that provides the graphical evolution of the system variables together with the corresponding virtual reality animations. Both graphical outputs will be used throughout this section to illustrate the obtained results. Additionally, intuitive 3-D snapshot diagrams have also been included. In these diagrams, the coloured lines represent, respectively, the complete planned (dashed light colours) and the simulated (solid dark colours) movements of the multirotor centre of mass. In contrast, the snapshots themselves only cover some intermediate configurations to intuitively illustrate the behaviour of the system (air/pipe

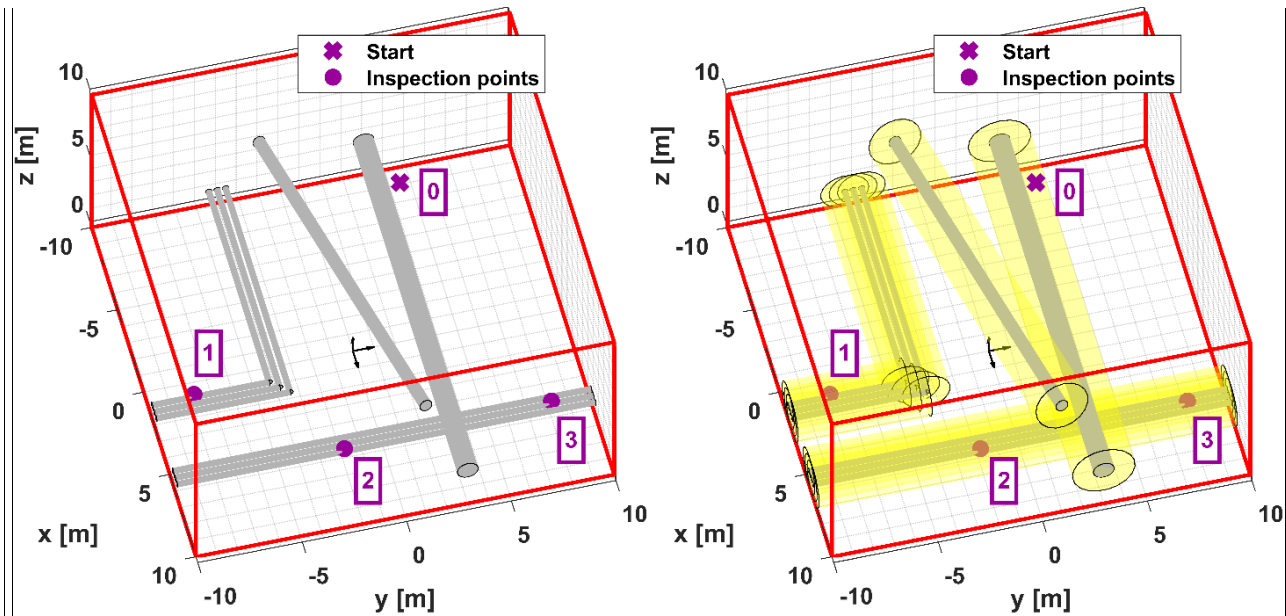
location, yaw angle, ...). These representative configurations are time-ordered by identifying labels that go from the initial position (label 0) to the final position (label 0 again) passing through a set of intermediate positions (labels 1,2,3, ...).

### 2.7.1. Planning algorithm for autonomous hybrid motion

This section is focused on the validation of the MP-HR planner presented in Section 2.4 that serves as the basis for the more advanced developments derived in subsequent sections (Dynamics Awareness and Reactivity). The scenario proposed below has been selected for that purpose. Both cost functions defined in Equations (2) and (3) have been used to optimise respectively the operation time and the energy consumption. The objective is to evaluate the performance of the generated trajectories under such metrics as well as their robustness against potential collisions with the pipes.

#### *Scenario 1: Inspection plan in moderately cluttered areas*

The application scenario considered for this validation is depicted in Figure 14. The inspection plan consists of three measurement points and the operation area can be classified as moderately-cluttered since it comprises several pipe arrays and isolated pipes with different diameters and inclinations.

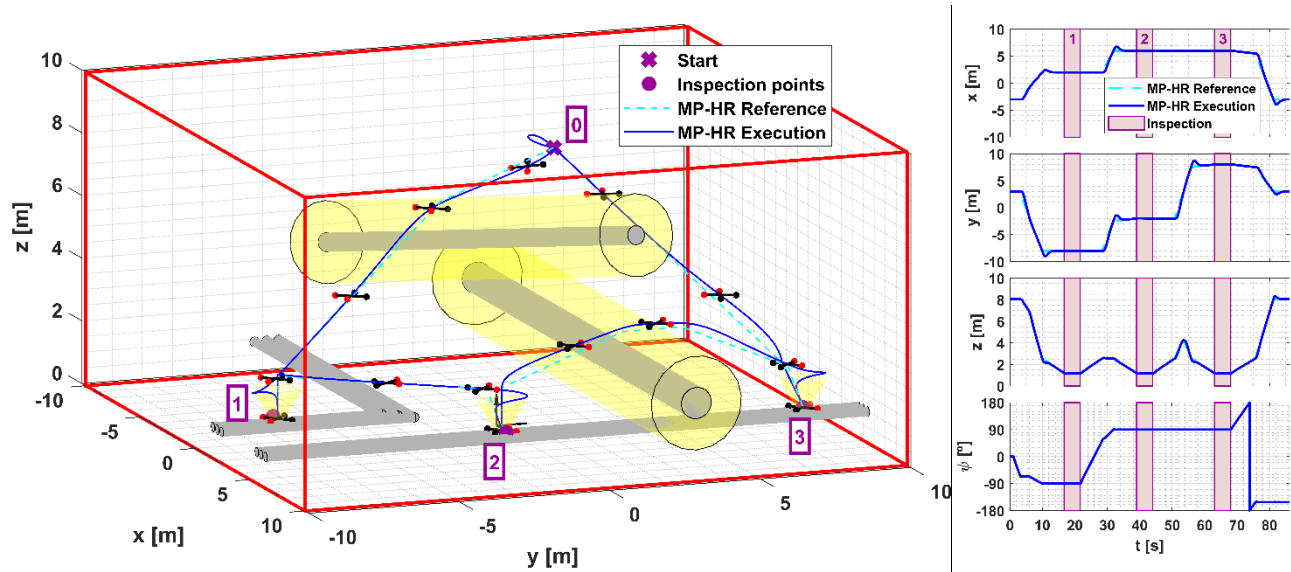


**Figure 14.** Application scenario proposed to validate the MP-HR planner: topology definition (left) and the corresponding safety regions (right). Moderately-cluttered environment with two pipe arrays, two isolated pipes and three inspection points.

#### *Efficiency in operation time*

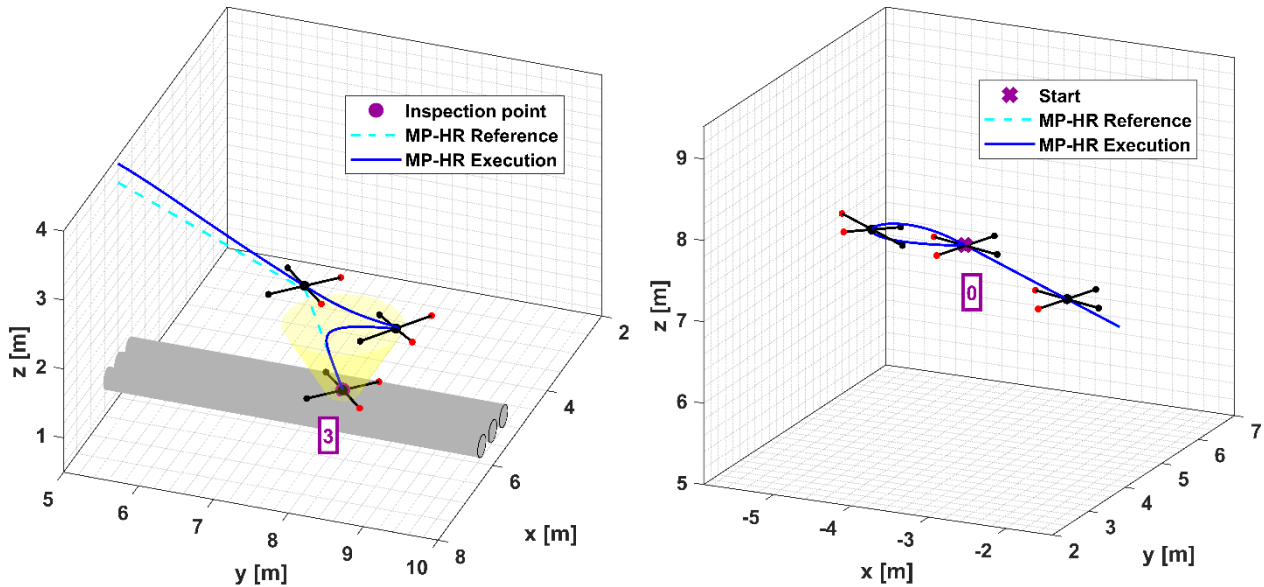
The validation simulations of the MP-HR algorithm when minimising the operation time have been included in Figure 15 as well as in the animation video “*Scen\_1\_MPHR\_time.mp4*” provided in [Vid-S]. As it can be seen, the HR tends to fly directly between inspection points. This behaviour is consistent with the higher velocity -and hence shorter time- associated with the flying mode (in comparison with rolling mode). Additionally, the restricted areas (safety margins of pipes, operation limits, ...) are properly avoided.

Concerning the yaw angle, it is properly aligned with the flight path that guarantees optimal time execution. This alignment also allows better detection of unmapped obstacles since the common setups of on-board sensors maximise the perception capabilities on the front side of the system.



**Figure 15.** Trajectory planned with the MP-HR algorithm when optimising the operation time (dashed light blue) and its corresponding execution by the controlled hybrid robot (solid dark blue).

Finally, it must also be noted that the system (first HR prototype presented in Section 2.8) exhibits some overshooting around the reference waypoints (see Figure 16) whose magnitude depends on the controller and parameters adopted for this simulation work.



**Figure 16.** Overshooting in reference waypoints extracted from Figure 15: landing on the inspection point 3 (left) and final approaching to the start / final point (right). The extended version of the motion planner that incorporates Dynamics Awareness will overcome these undesired behaviours of the closed-loop dynamics.



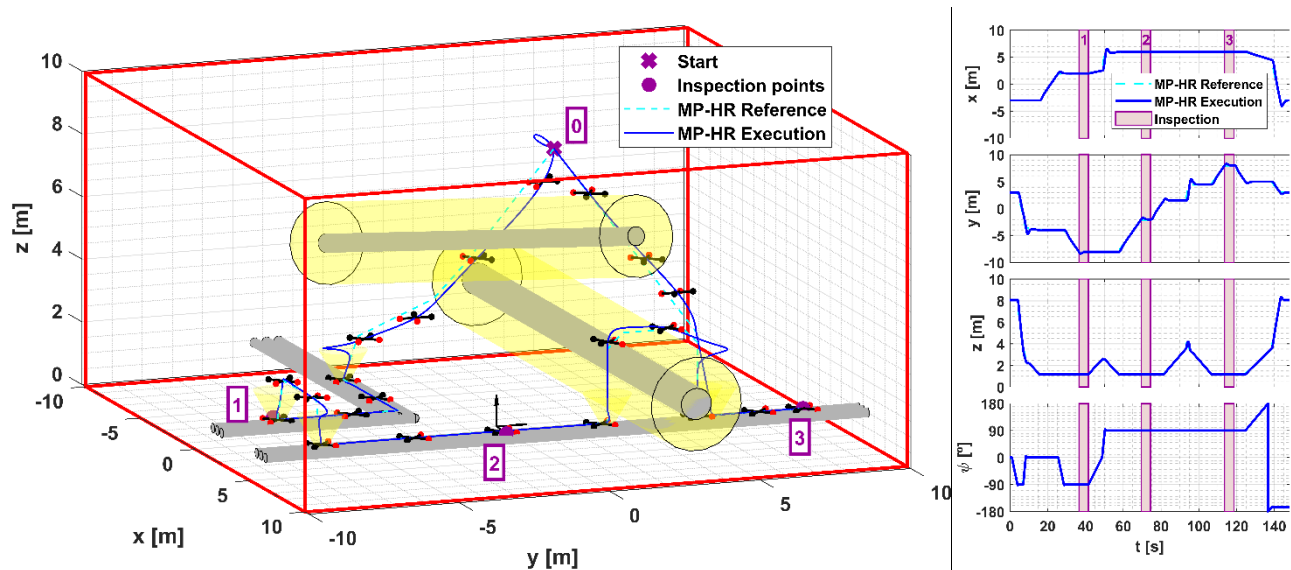
Following this analysis, it can be concluded that the algorithm is capable of generating both safe and time-efficient trajectories for the hybrid robot. However, the existence of the overshooting previously mentioned is something that requires further follow-up. Subsequent scenarios will illustrate that this degraded performance of the controller could lead to collisions and other not allowed behaviours. The extended version of the motion planner that incorporates Dynamics Awareness will be the solution to overcome these undesired behaviours of the closed-loop dynamics, as will be presented later in Section 2.7.2.

### ***Efficiency in energy consumption***

The validation simulations of the MP-HR algorithm when minimising the energy consumption have been included in Figure 17 as well as in the animation video “*Scen\_1\_MPHR\_energy.mp4*” provided in [Vid-S]. In contrast to previous case, the HR tends here to roll on the pipes as much as possible. This new behaviour is consistent with the smaller energy consumption associated with the rolling mode (in comparison with flying mode). Additionally, the restricted areas (safety margins of pipes, operation limits, ...) are properly avoided.

The alignment of yaw angle in flight segments is fulfilled in the same manner as before. Concerning the pipe segments, these results endorse the proper selection of yaw references that are compatible with rolling displacements along the pipes.

Following this analysis, it can be concluded that the algorithm is capable of generating both safe and energy-efficient trajectories for the hybrid robot, with the same remarks concerning the overshooting that were discussed in previous simulations.



**Figure 17.** Trajectory planned with the MP-HR algorithm when optimising the energy consumption (dashed light blue) and its corresponding execution by the controlled hybrid robot (solid dark blue).

### **2.7.2. Extension of planning algorithm based on Dynamics Awareness for safer operation**

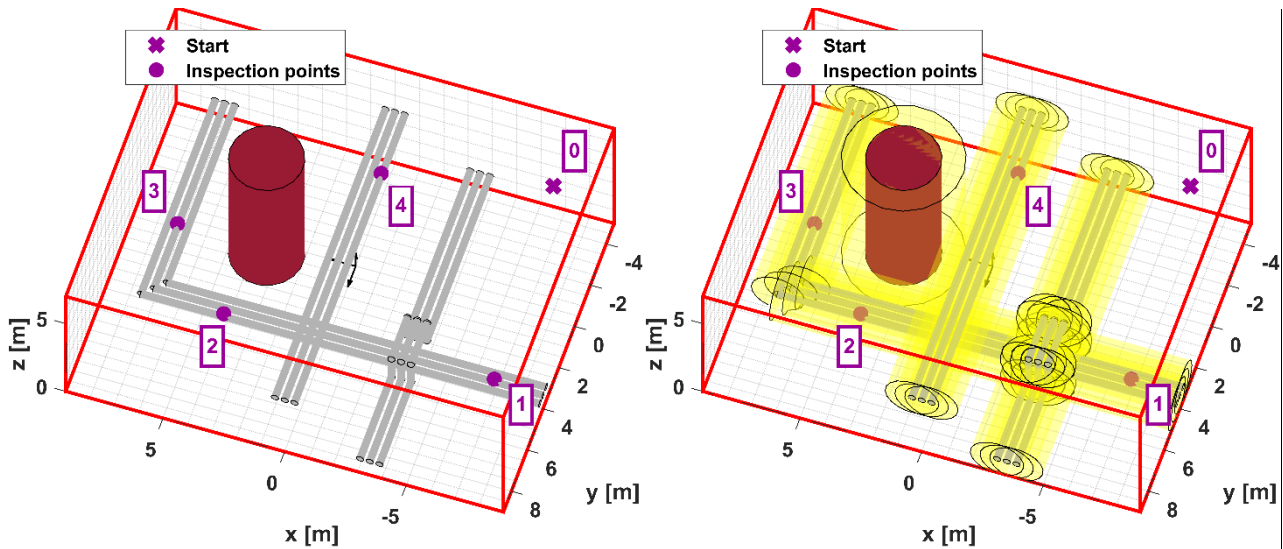
This section validates the planner extension presented in Section 2.5 that considers the dynamic constraints of the hybrid system when flying. In order to better illustrate the need of this feature, a new highly-cluttered scenario is proposed. Moreover, the selected metric for planner optimization is the operation time. The usage of this metric, which yields faster trajectories by maximising the flying

segments, together with the new highly-cluttered scenario, narrow the security margins of the generated flying trajectories with respect to the obstacles. The simulations developed under such critical situation reveal the importance of taking into account the expected closed-loop behaviour of the system in planner operation.

Finally, it must be clarified that this safety enhancement of the algorithm is equally suitable for flying phases when minimising the energy consumption. The choice of the time-consumption metric for this section is simply made to maximise the number of situations that will benefit from the algorithm extension.

### ***Scenario 2: Inspection plan in highly cluttered areas***

The application scenario is presented in Figure 18. The new inspection plan consists of 4 measurement points and the operation area can be classified as highly-cluttered since it comprises more pipe structures than previous scenarios. Their relative topology is also more complex since they are located at different heights and there also exist some connections amongst them. Additionally, a tank (red vertical cylinder) has been included in the central area, which will also limit considerably the manoeuvrability of the HR when flying.



**Figure 18.** Application scenario proposed to validate the MP-HR-DA planner: topology definition (left) and the corresponding safety regions (right). Highly-cluttered environment with three pipe arrays in complex topologies, a tank (red vertical cylinder) and four inspection points.

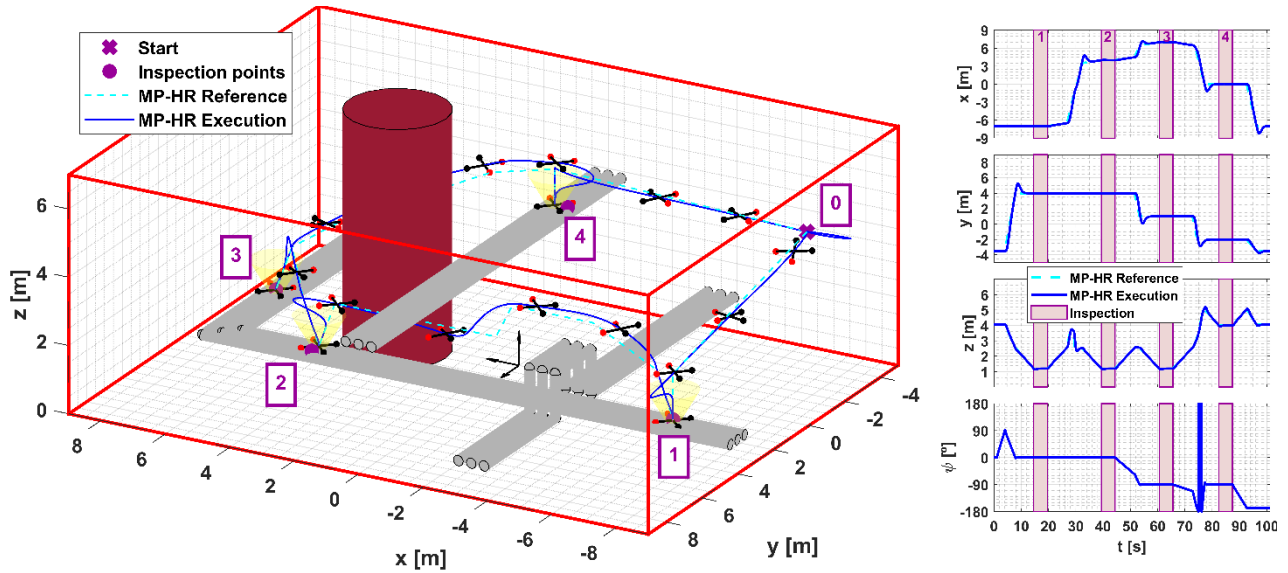
### ***Dynamics Awareness in time-efficient plans***

The validation simulations have been structured around two phases. On a first step, the performance of the basic MP-HR algorithm has been evaluated in the scenario described before. On a second step, the extended MP-HR-DA algorithm is applied to the same scenario in order to illustrate the benefits associated with this enhanced version of the planner.

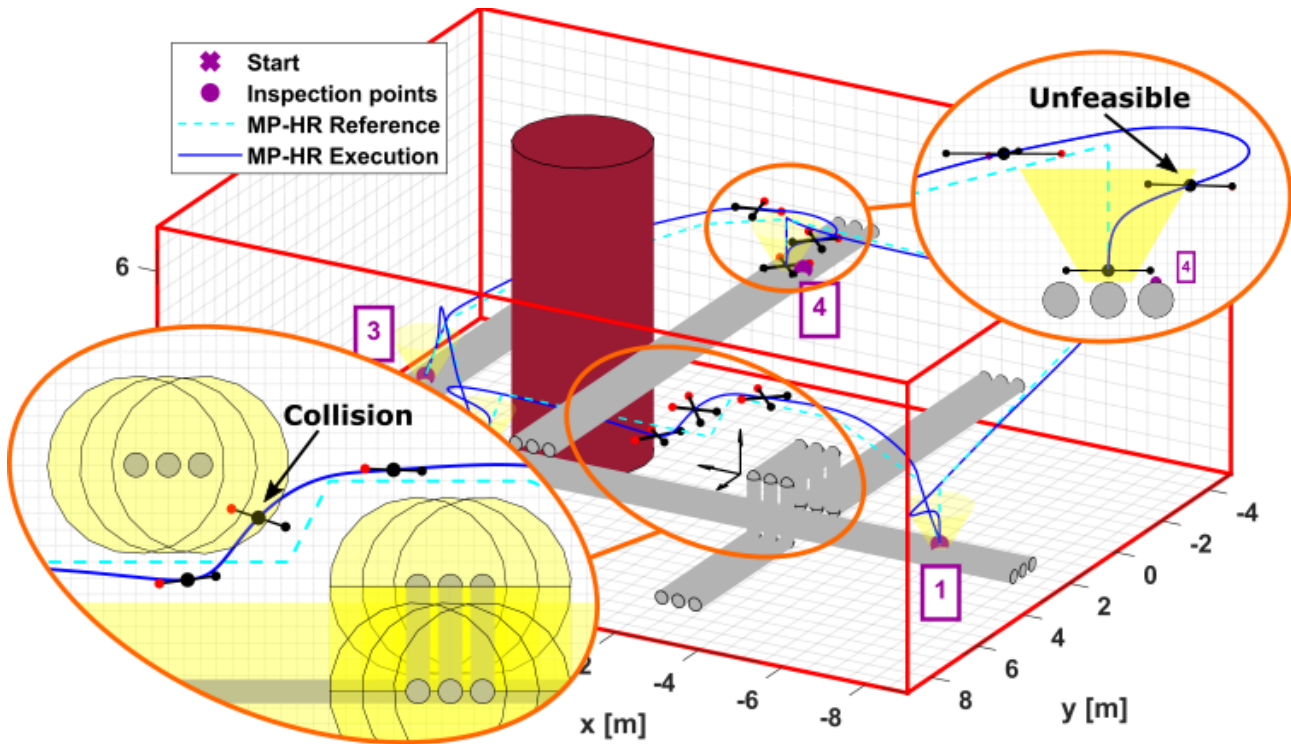
#### Time optimization without Dynamics Awareness

The trajectory generated by the basic MP-HR algorithm for the new highly-cluttered scenario has been included in Figure 19 as well as in the animation video “*Scen\_2\_MPHR\_time.mp4*” provided in [Vid-S]. According to this result, the algorithm plans both safe and time-efficient trajectories for the proposed inspection sequence. However, the corresponding closed-loop trajectory described by the

controlled system does not satisfy the safety conditions. Firstly, it violates the safety margin of the pipe array located above, which is already considered a collision to prevent closer approximations (see detailed view in the left side of Figure 20). Secondly, the condition required to safely landing on the same array is not fulfilled either (see detailed view in the right side of Figure 20). These undesired behaviours are consistent with the fact that the basic MP-HR algorithm does not consider the dynamics of the system and therefore dynamical effects like overshooting are not accounted for during the feasibility-checking phase of the planner.



**Figure 19.** Trajectory planned with the MP-HR algorithm when optimising the operation time (dashed light blue) and its corresponding execution by the controlled hybrid robot (solid dark blue).

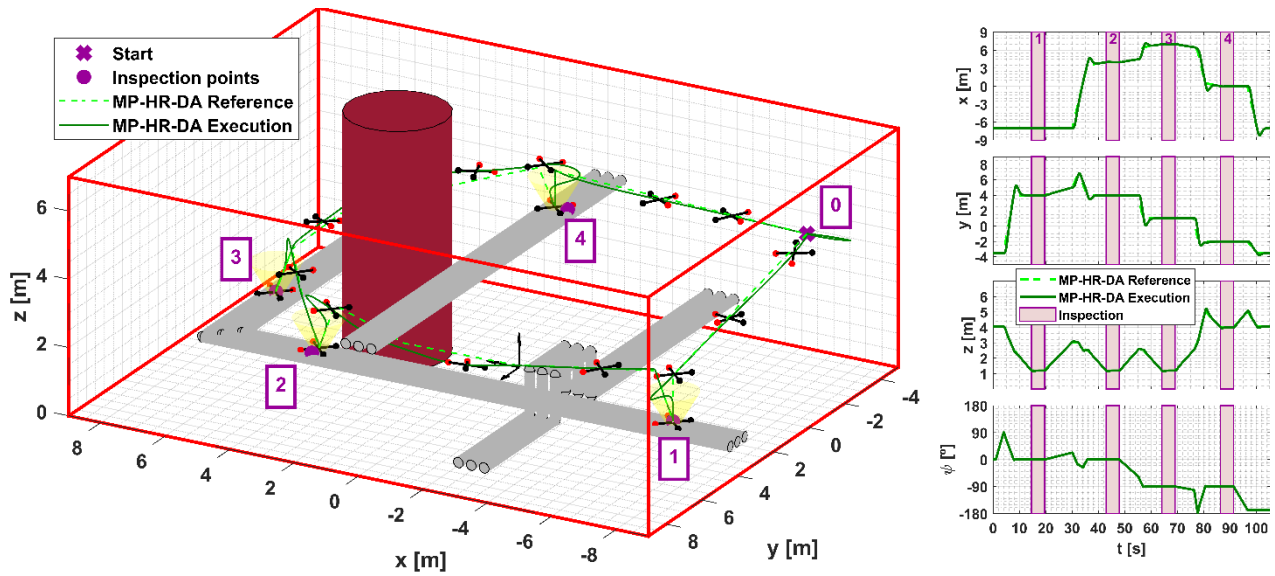


**Figure 20.** Detailed views of Figure 19: violations of margin of safety (left) and region of safe transition (right).

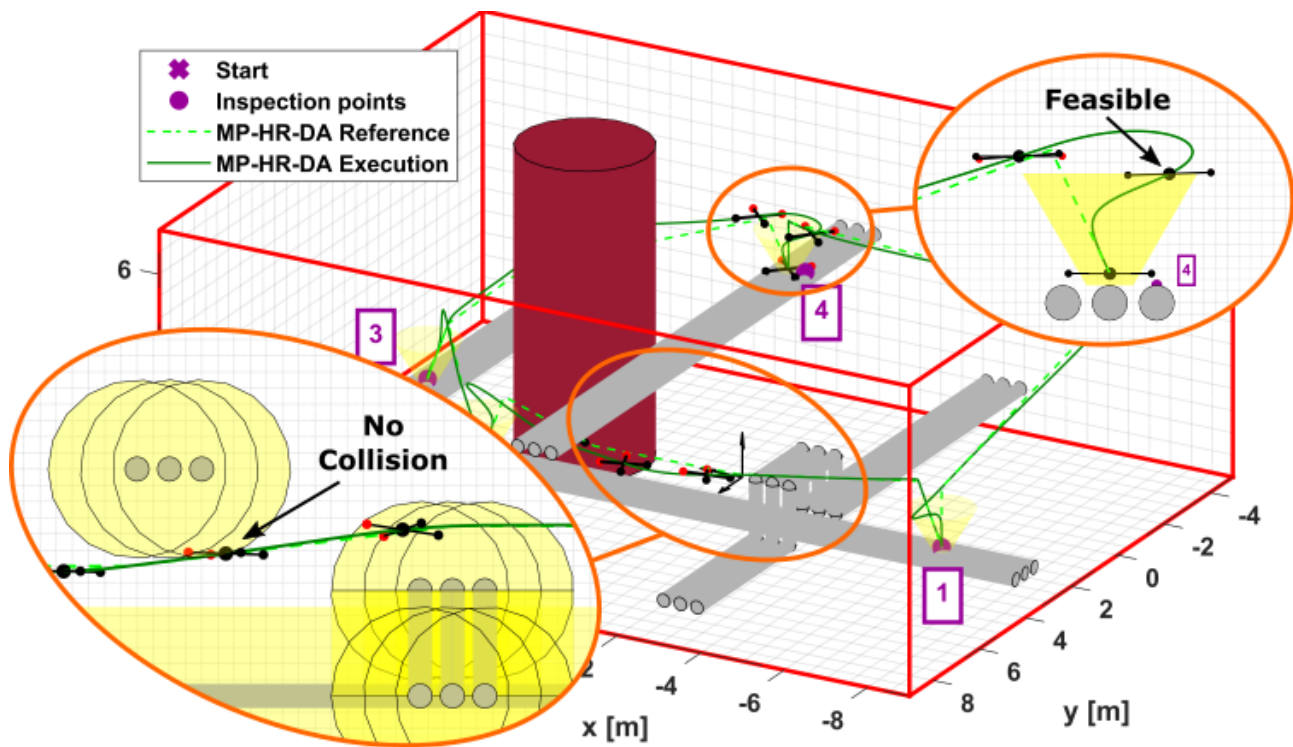
#### Time optimization with Dynamics Awareness

Previous results illustrated the need to consider the dynamics of the system when applying the MP-HR algorithm in highly-cluttered environments. The simulation results corresponding to the MP-HR-DA method that meets this requirement are presented in Figure 21 as well as in the animation video “*Scen\_2\_MP-HR-DA\_time.mp4*” provided in [Vid-S]. As it can be observed, the Dynamics Awareness allows the algorithm to solve the problematic situations described previously: neither collisions with the safety margins for obstacles (please note that the safety regions define the non-allowable regions for the displacement of the centre of mass of the HR) nor violations of the safety conditions for landing manoeuvres appear (see Figure 22). The analysis of these results allows to conclude that the Dynamics Awareness is required for generating both time-efficient and safe trajectories when the hybrid system operates in cluttered environments.





**Figure 21.** Trajectory planned with the MP-HR-DA algorithm when optimising the operation time (dashed light green) and its corresponding execution by the controlled hybrid robot (solid dark green).



**Figure 22.** Detailed views of Figure 21: the MP-HR-DA overcomes the violations of the margin of safety (left) and the region of safe transition (right).

### 2.7.3. Navigation support to reinforce system autonomy

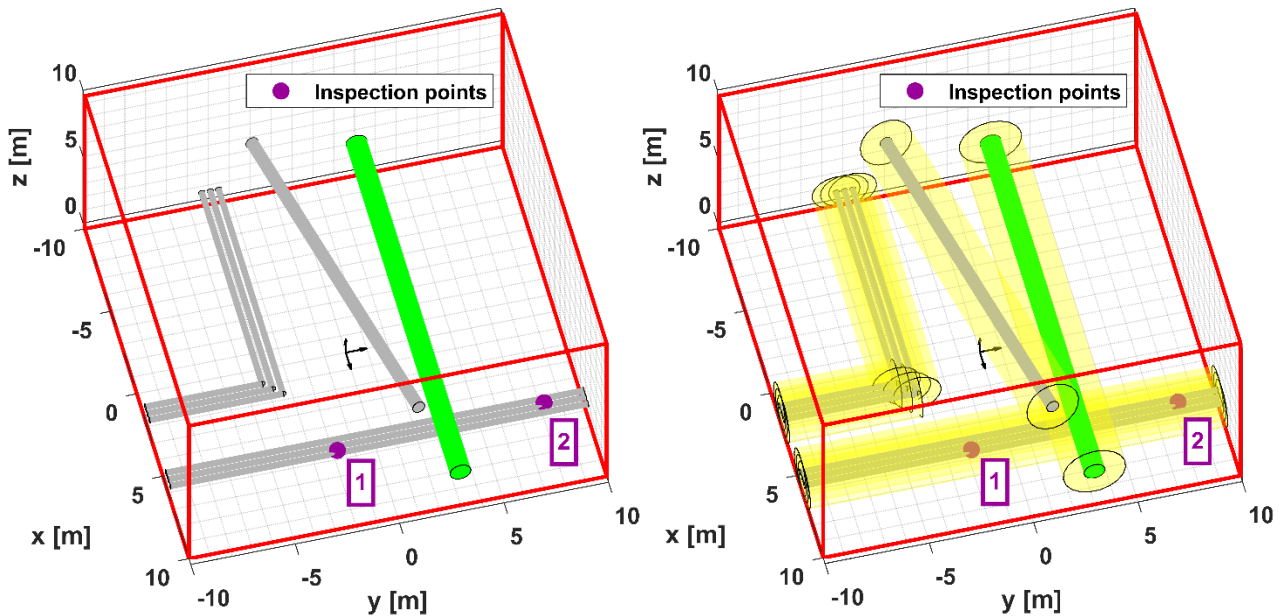
This section illustrates the reactive capabilities of the MP-HR planner that have been presented in Section 2.6. With this motivation, a modified version of the scenario given in Figure 14 is presented below.

The first simulations for this topic were included D4.1. In that case, the validation was focused on aerial phases. In order to complete the validation, the analysis is concentrated here on the hybrid reactivity required for rolling phases.

Finally, it must be noted that, since the need of hybrid reactivity only arises when rolling on the pipes, the selected metric for planner optimization in this scenario will be the energy consumption which is more prone to that kind of manoeuvres. In any case, this reactive enhancement of the algorithm is equally suitable for rolling phases when minimising the time consumption.

### ***Scenario 3: Inspection plan in moderately cluttered areas with unmapped obstacles***

As advanced before, a modified version of scenario given in Figure 14 has been proposed. In this case, some pipes of the moderately-cluttered area are not included in the map provided to the planner. Specifically, the horizontal pipe that passes crosswise through the scenario from  $(-10\text{m}, 3\text{m}, 2.5\text{m})$  to  $(10\text{m}, 3\text{m}, 2.5\text{m})$  has been removed (see Figure 23). This is a realistic assumption since it is difficult to keep updated the map of the pipes after the frequent operations of repairing and substitution of the piping layout. As a consequence, the on-board sensors must reveal the presence of these unmapped obstacles to trigger the online re-planning that will allow the reactive response of the system.



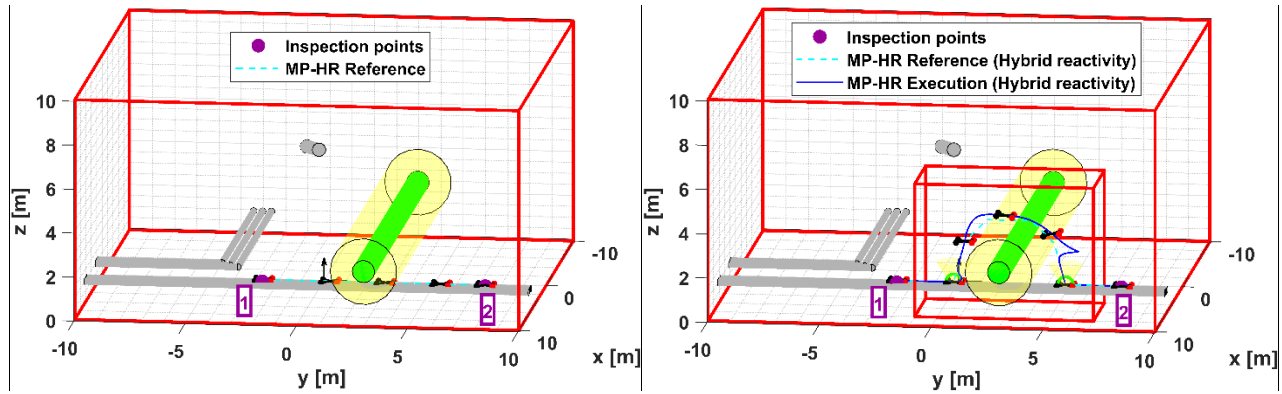
**Figure 23.** Application scenario proposed to validate the reactive capabilities: topology definition (left) and the corresponding safety regions (right). It is a modified version of the scenario shown in in Figure 14, where the pipe highlighted in green that crosses horizontally from  $(-10\text{m}, 3\text{m}, 2.5\text{m})$  to  $(10\text{m}, 3\text{m}, 2.5\text{m})$  will not be included in the map provided to the planner.

### ***Reactivity for rolling manoeuvres in energy-efficient plans***

The validation simulations of the hybrid reactivity have been presented in Figure 24 as well as in the animation video “*Scen\_3\_MPHR\_reactive.mp4*” provided in [Vid-S]. On the left side of Figure 24, the trajectory computed when the unmapped pipe is not considered. On the right side, the re-planned trajectory generated during the inspection that allows the reactive avoidance of the pipe detected by on-board sensors. In this right side, the initial and final positions (green circles) as well as the operation area (red subset of the operational limits) of the re-planning have been highlighted. From a more general view, it can also be concluded that the HR maintains its tendency to roll on the pipes as

much as possible. Additionally, the restricted areas (safety margins of pipes, operation limits, ...) apart from the unmapped pipe, are still properly avoided. Therefore, the generation of both safe and energy-efficient trajectories is also validated when facing unexpected obstacles during rolling manoeuvres.

Concerning the yaw angle, the reactivity response robustly maintains the alignment that is required for rolling displacements along the pipes.



**Figure 24.** Reactive capabilities for rolling manoeuvres. Trajectory initially planned by the MP-HR algorithm without considering the green pipe (left) and re-planned trajectory when the green pipe is detected by on-board sensors (right).

## 2.8. Experimental results

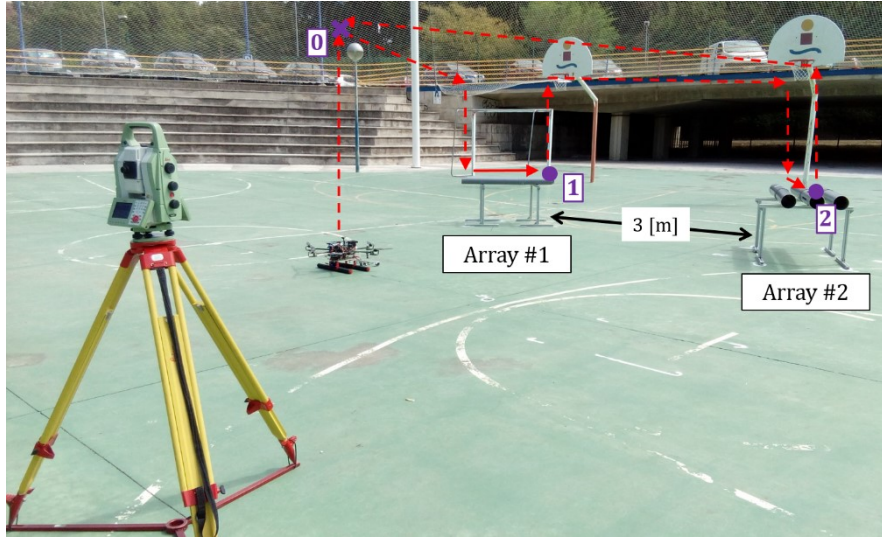
Once the different strategies for planning hybrid motion have been validated through simulation, the next step is the experimental validation in real outdoor scenarios.

### 2.8.1. Planning algorithm for autonomous hybrid motion

First experiments have been carried out with the basic MP-HR algorithm in a scarcely-cluttered area consisting of several mock-ups made of plastic that closely resemble the real pipe arrays. These conditions are considered safe enough to guarantee the absence of collisions even in the case that any deviation from expected operation is produced.

#### *Experimental scenario: Inspection plan in scarcely cluttered area*

The proposed outdoor scenario is shown in Figure 25. It consists of two arrays of three PVC (polyvinyl chloride) pipes (2 m length, 20 cm Ø, 10 cm separation) supported by a frame structure built with Rexroth bars. The inspection plan comprises the two measurement points that are highlighted in the figure.



**Figure 25.** Experimental scenario proposed to validate the MP-HR planner. Scarcely-cluttered environment with two arrays of three PVC pipes and two inspection points. The Leica MS50 robotic total station is used to measure the motion of the HR.

### *Experimental setup*

Following the general design presented in Section 2.2, the robot used in the experimental work consists of a hexarotor platform integrating a rolling base and a 5-DOF robotic arm supported by a 1-DOF linear guide. This guiding system facilitates the deployment of the arm in the array of pipes to inspect their contour once the platform has landed. The experimental prototype previously described is presented in Figure 26 while performing the inspection of a pipe array.



**Figure 26.** Aerial manipulator with rolling base inspecting an array of pipes.

The architecture of the HR comprises three main parts. Firstly, the multirotor platform integrates a Pixhawk autopilot with the position/trajectory controller as well as a Raspberry Pi 3B+ computer with the remaining software modules. Secondly, the rolling base is controlled by two servos using a STM32 microcontroller. Finally, the robotic manipulator integrates two groups of servos: three standard Herkulex DRS-0101 actuators used for positioning the end effector, and the customised micro servos of the linear guide and the wrist joints.

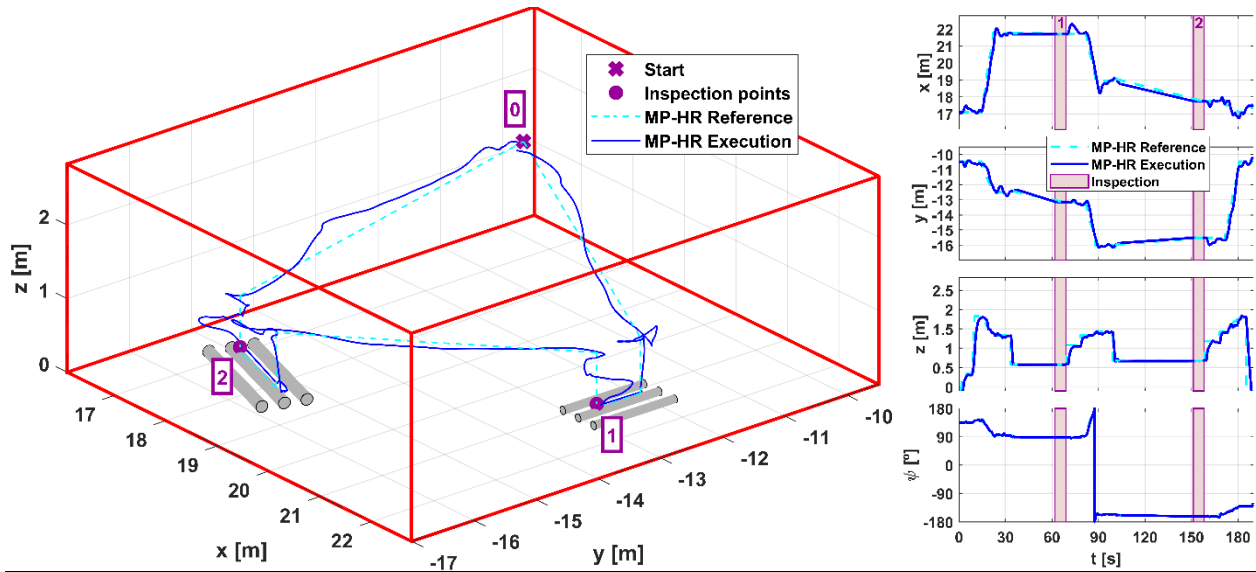


The motion of the HR is measured using the robotic total station (Leica MS50 [LeiMS50]) that can be seen in Figure 25. This laser tracking system provides accurate estimations of the position of the robot.

### *Experiments with energy-efficient plans*

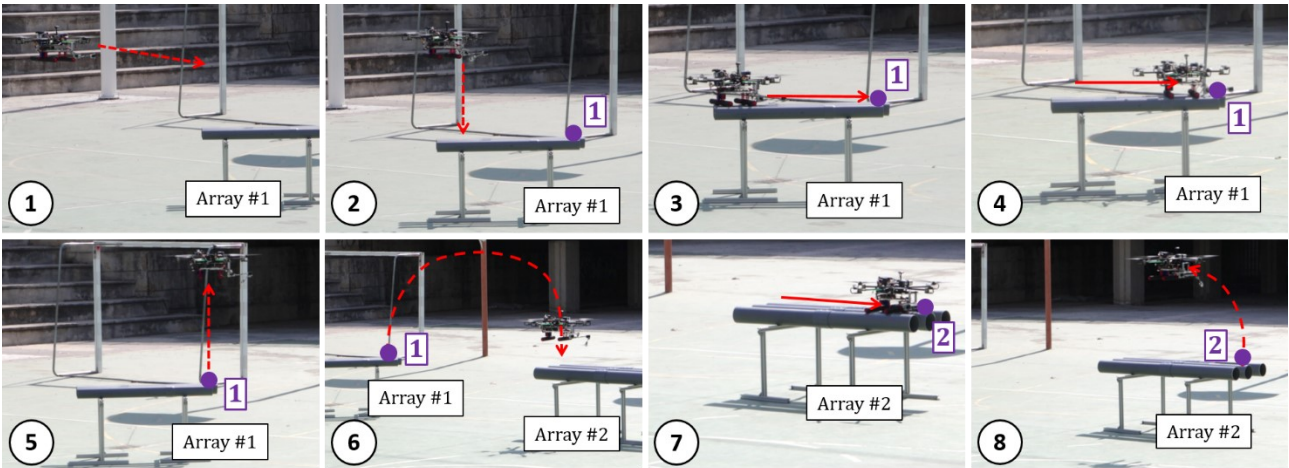
Since one of the main objectives of these first experiments is the validation of the algorithm capabilities to leverage the hybrid motion of the system, the selected optimization metric has been the energy consumption. This choice will make the planner tending to maximise the rolling manoeuvres and, hence, the hybrid operation of the system.

Figure 27 represents the trajectory planned by the MP-HR algorithm (dashed light blue) and the real trajectory executed by the HR (solid dark blue). As it can be observed, the planner computes a trajectory that properly guides the robot along the selected inspection points. Furthermore, the safe and energy-efficient behaviour that could be expected from simulation work, is also met with these experimental results. Indeed, the trajectory is energy-efficient since the three sub-trajectories connecting the inspection points prioritise rolling segments along the pipe arrays and only incorporates flight segments when it is unavoidable. Moreover, the trajectory also guarantees a safe execution of the inspection since the safety conditions concerning obstacle avoidance as well as landing / take-off manoeuvres, are properly fulfilled.



**Figure 27.** Trajectory planned with the MP-HR algorithm (dashed light blue) and executed by the experimental prototype of the controlled hybrid robot (solid dark blue). Optimization of the energy consumption.

The execution of the experiment can also be followed in the experimental video “Exp\_MPHR\_energy.mp4” provided in [Vid-S] as well as in the sequence of images included in Figure 28.



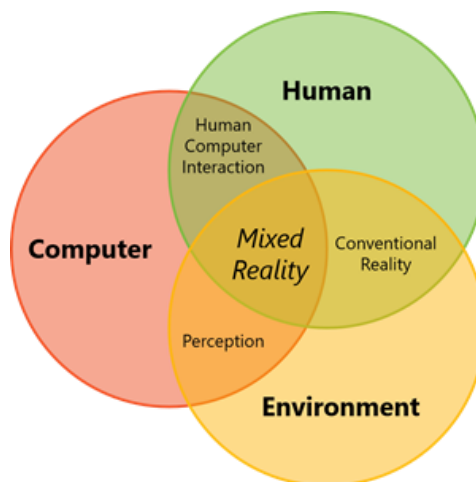
**Figure 28.** Sequence of images showing the execution of the inspection task.

### 3. Augmented reality applications (T4.4)

In this section an augmented reality system will be presented. This system will be part of the human operation supporting functions of the overall HYFLIERS system. This 3D augmented reality system will be employed to enrich the information available to the pilot during the operation. It is part of Task 4.4.

During the first period of the project, this task was focused on the development of a new type of interface for the hybrid robot operator. Among other advanced functionalities that have been integrated in this interface, augmented reality goggles will show to the operator visual information about the status of the hybrid robot and its mission. This information will be provided to the operator in such a way that they do not have to change its field of view and hence, they can always maintain a look to the hybrid robot and its operation at any time. By doing this, lacks of information about the Hybrid Robot state and operation will be avoided.

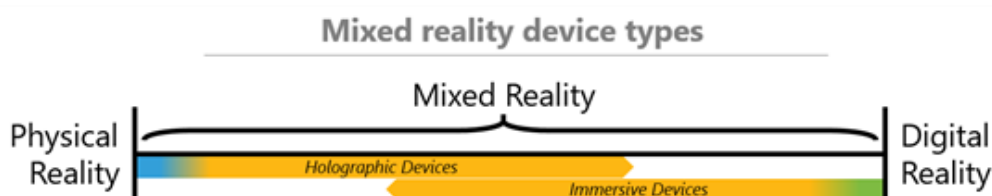
The device used for this development was Microsoft HoloLens. These goggles make possible mixed reality, which basically combined the real world with a virtual environment where the physical and digital objects coexist both together and can even interact between them. These definitions are graphically represented in Figure 29.



**Figure 29:** Mixed reality concerns.

Mixed reality represents the majority of market possibilities, being the augmented reality and virtual reality a minor part of the available devices. In particular, Microsoft has two different devices of mixed reality (see Figure 30):

- Holographic devices: these gadgets can place digital content in the real world.
- Immersive devices: these gadgets directly hide the real world and a digital experience replaces it.



**Figure 30:** Mixed reality devices.

As it was said before, the hardware selected was the Microsoft HoloLens goggles. This hardware allows placing holograms within the holographic framework, light objects and sounds, all of them mixed with the real-world image. All of them can react to the sight, gestures and voice commands, and even interact with the real world. Apart from that, it has different possibilities for the hologram, since they can remain static in a given position or follow the user eyes. Thus, different shapes and behaviours of the holograms are possible with HoloLens. Head-phones placed on the user's ears enable the goggles to emit sound, enhancing the system with an additional functionality.

In this section, both hardware and software architecture will be presented, as well as the results of the first experiments carried out at CATEC's facilities.

### 3.1. Microsoft HoloLens Goggles

The Microsoft HoloLens are the hardware selected for this application, and they are shown in Figure 31.



**Figure 31:** Microsoft HoloLens Goggles.

**Table 1:** Goggles specifications

Display	
Optics	See-through holographic lenses (waveguides)
Resolution	2 HD (high-definition) 16:9 light engines
Holographic density	>2.5 radiants (light points per radian)
Eye-based rendering	Display optimization for 3D eye positno
Sensors	
Head tracking	4 visible light cameras
Eye tracking	2 IR (infra-red) cameras
Depth	1-MP time-of-flight (ToF) depth sensor
IMU	Accelerometer, gyroscope, magnetometer
Camera	HD, with 2MP picture
Audio and speech	
Microphone array	Five channels
Speakers	Built-in spatial sound

Apart from all the specifications in Table 1, these goggles have a flash memory of 64GB, with 2GB of RAM. Their weight is 579g and the battery allows between 2 and 3 hours of active use. They can follow the user sight, respond to the spatial sound. Its operating system (OS) is Windows 10, and they have applications in Windows Store, such as Holograms, Microsoft Edge, Photos, Windows

Feedback, Calibration, Learn Gesture, etc. Moreover, they have multiple compatible accessories, as the one shown in Figure 32, the HoloLens clicker which enables the system with new features. Apart from that, they can be connected with standard Bluetooth keyboards and gamepads, and even with any peripheral that is compatible with HID (human interface device) or GATT (generic attribute) profiles, although they may need an extra application for its use.



**Figure 32:** HoloLens Clicker.

To build any type of application, there are several blocks for construction available, including sight, gesture, voice, spatial mapping, spatial sound and coordinates system block. The following subsections will explain each of them.



**Figure 33:** HoloLens Hardware schematic.

### 3.1.1. Sight block

It is the first input for the goggles, and it is considered the primary way to focus in the mixed reality. This block indicates where the user is looking at, and hence, it can predict its intentions. The first action to do is to focus on the user's hands. By doing this, the interaction between the goggles and the user starts and then, hand movements can be used to control the menu. Some other uses for this block are:

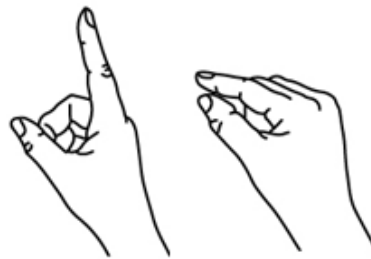
- The application can cross user sight with the holograms of the scene to determine where the user's attention is.
- Holograms can be targeted, allowing actions such as selection, activation, grabbing, displacements and the interaction with them.
- In addition to the spatial mapping (which will be explained later), the user can place holograms in the real-world surfaces.
- By means of the user's look, the HoloLens can know if the user is pointing the object or not. In that case, they can give to the user visual and audio indications to recover the targeting to the object.

For the majority of the applications, a pointer is used to give the user enough confidence to know what they are interacting with. Once the hologram or the real-world object is targeted, the following step is to interact with gestures and voice commands.

### 3.1.2. Gestures block

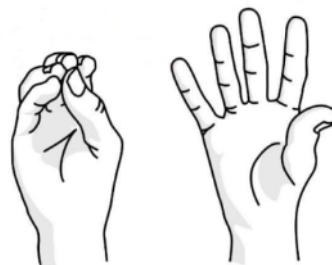
Gestures are input by user's hands, which simplify the interaction with the content, although it does not provide the system with a precise positioning in space. It is necessary to combine both look and gesture inputs, since the HoloLens does not support only movement controls. The basic gesture inputs are the following ones:

- **Air tap (Figure 34):** it is a tapping gesture with the hand, similar to a mouse click. This gesture input is normally used after pointing an object or hologram with user's eyes. It can be substituted with the bottom of the HoloLens Clicker or saying "Select" with the voice control.



**Figure 34:** Air tap.

- **Bloom gesture (Figure 35):** this gesture is reserved for the "Home" tab. It is used to return to the Start Menu. As it is seen in the figure below, it consists of joining the fingertips together and then opening the palm. As for the previous gesture, it can be replaced by a voice command, in that case "Hey Cortana, go Home".



**Figure 35:** Bloom gesture.

However, applications can work with more than simple gestures as the one presented above. There are composite gestures and you can even customize gestures, for example a double tapping. Some of these composite gestures are:

- Tap and hold, holding the tapping position as it would be a grab.
- Manipulation, used to move holograms following hand's movement.
- Navigation, which work as a joystick, moving the hand along the axes creating zooms and displacements.

Therefore, it is possible to make a specific hologram to respond to some types of gestures, which is a very interesting possibility in terms of coding and design. It must be mentioned that to recognize a gesture, it must be inside the named "Vision frame", as illustrated in Figure 36. This frame is defined by the viewing aperture of the HoloLens cameras. It does not apply in the case of being using the

HoloLens clicker. This constrain must be taken carefully in consideration, since it could be possible that during a continuous movement the user takes their hands out of the frame and hence the process would be interrupted. The user must be aware of that.



**Figure 36:** Vision frame.

### 3.1.3. Voice block

Voice input is a human natural way of communication, and hence, HoloLens implements this feature. It allows the user to shorten nested menus with a single command. For instance, users can activate holograms saying only the command *Select*. This feature is habilitated by a keywords detection algorithm, whose consumption is very low. Thus, it is always enabled, without any impact on the battery life. Moreover, it enables the popular Microsoft Voice Command: *Hey, Cortana*. After that, the user can formulate any type of question or give any instruction.

Furthermore, HoloLens have a “*See it, say it*” model for voice inputs, where labels on the buttons inform to the user the voice command that they can use for that purpose. In addition, some commands are allowed while the user is watching a hologram, as well as voice dictation can be used to substitute the writing by mean of Air Taps.

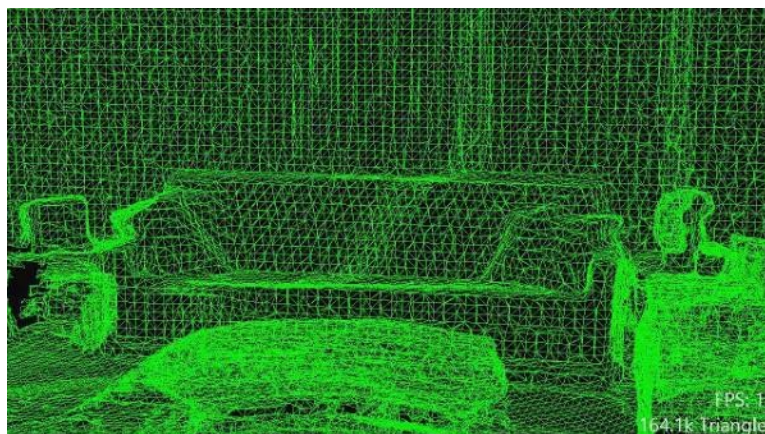
From the application perspective, there are different categories in terms of transmission depending on the customized processing and audio optimization.

### 3.1.4. Spatial mapping

These goggles merge the real world with a hologram. To do so, the HoloLens maps the space in which they are working, and hence, the user could have a more realistic experience.

The application provides the Spatial Surface Observer with one or more bounding volumes, to define the regions of space in which the application wishes to receive spatial mapping data. These volumes can be either stationary, i.e., in a fixed location with respect to the real world, or they could be attached to the HoloLens and their movement. However, in this case, they will not rotate.





**Figure 37:** Example of a spatial allocation mesh covering a room.

### 3.1.5. Spatial sound

As if it were a real object, the HoloLens goggles provide with a spatial sound that can come from all the directions: above, below, behind, etc. This increases the user experience and improves the integration of both the virtual and real world.

### 3.1.6. Coordinates systems

The holograms need to be placed in the world, implying the placement and orientation of them with accuracy. This geometry can be represented within different reference frames. Windows provides with several reference systems of the real world and they are known as Spatial Coordinates Systems. Cartesian coordinate systems are the ones used in all 3D graphics applications. However, in mixed reality the case is different, due to the fusion of both virtual and real world. That is the reason for using spatial coordinate system.

It is important to highlight that these coordinates systems express their coordinates in meter. It means that an object placed 2 units far from another one, either in the X, Y or Z axis, will appear 2m apart one from another once rendered in mixed reality. That must be taken into consideration when rendering objects and environment in the real world. Moreover, the spatial coordinates systems defined in Windows are always right-handed.

These coordinates systems can be either stationary or attached. The first ones work to maintain the position of the objects near of the user, as stable as possible in relation with the world, taking into account the changes of the user's head position and orientation. On the other hand, an attached reference frame moves with the user while they are walking.

## 3.2. Software architecture

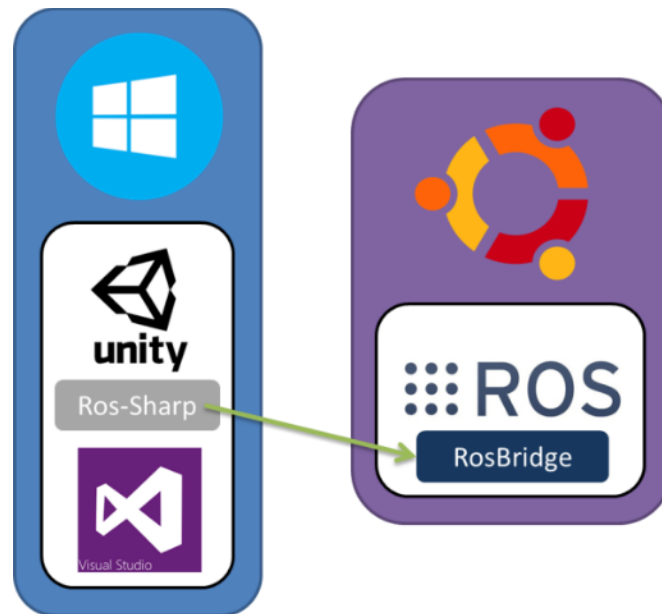
Once the hardware and its possibilities have been explained, it is time to develop the software architecture. As a summary, the environment used for this development has been ROS (Robot Operating System), and the application has been developed with Unity and Visual Studio. In the following sections, both of them are going to be explained. A basic schematic is shown in Figure 38. As it can be seen, two clearly differentiated modules compose the architecture:

- On the left-hand side, it is the Windows OS (in particular, Windows 10, since it is the only one compatible with HoloLens). Here, it has been used Unity and Visual Studio. The first one is a multiplatform thought to create videogames. It allows the design of 2D and 3D



videogames. It provides with a main scripting API (application programming interface), in C#, both for the editor and for the games. Thus, it was needed Visual Studio to code.

- On right side, it is the ROS module, which needs to operate in Linux OS, in its version Ubuntu. ROS is required to communicate with the unmanned aerial vehicle (UAV). ROS is an open-source meta-operative system. It provides with all the expected features of an OS, such as hardware abstraction, low level devices control, implementation of common use functionalities, message passing between processes and package management.

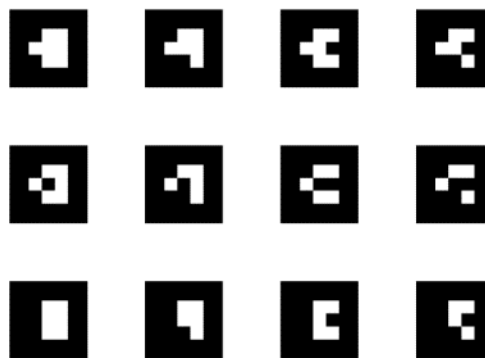


**Figure 38:** Schematic of the software architecture.

In order to use Unity with ROS, it is needed a particular package, whose name is ROS-Sharp. This package contains several libraries, coded in C#, to engage Unity with ROS.

### 3.2.1. Unity

To create mixed reality scene with unity, it is necessary the package HoloToolkit. Taking into account that Unity draws the object with respect to its initialization point, the possibility of being in constant communication with ROS since the beginning is key for the good functioning of the application. The application developed in Unity has at every moment since its initialization an active communication with ROS, through a script provided by ROS-Sharp.



**Figure 39:** Markers to be placed in the initialization point.

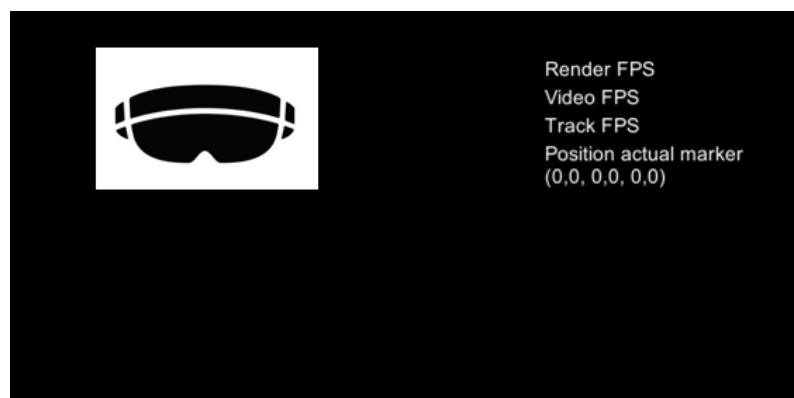
Therefore, we must merge the initialization position of the goggles, their provided current position, and the area in which the UAV is flying. For this purpose, the user has to calibrate the goggles position in order to know where they are located with respect to the UAVs. This operation is essential to show the UAVs information in the correct position, so the user must perform this task before starting the operation. The package ArtToolKit is used for that purpose. It provides a base for the scripts that allows the detection of a marker by HoloLens. These markers have been modified and the final result is shown in Figure 39.

This set of markers is recognized by the goggles, and then, the set gives rise to a plane, which has an associated image as the one shown in Figure 40. This image enables to know the pose (both orientation and position) of the plane and, hence, the pose of the goggles.



**Figure 40:** Plane for initialization.

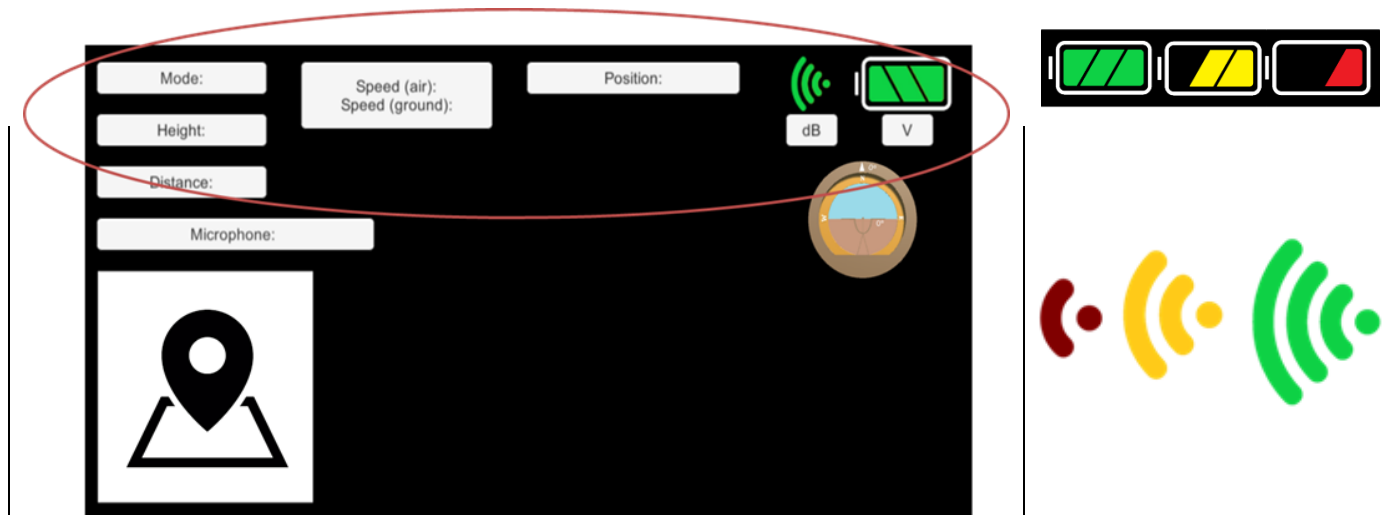
From the point of view of the application, this procedure is seen as in Figure 41. In the upper left corner, it will be seen the image coming from the HoloLens, and in the right side of it will be seen the information related to with the goggles. The position of the marker will always be the (0, 0, 0) point until the camera recognizes them. Once the markers have been detected, the user must wait for at least 7 seconds, staring at the markers.



**Figure 41:** Initialization interface.

Since the marker is recognized, the goggles can be positioned in the environment in which the UAV is going to fly. Hence, the transformation and merging of the positions mentioned before can be made. Then, the application goes to the next view and starts receiving information through a message created specifically for this purpose, coming from a ROS subscription. Figure 42 shows the interface and how these data are shown to the user. This message contains the following information:

- Flight mode
- Distance of the UAV to the HoloLens
- Ground and air speed
- UAV position and pose
- The status of UAV-HoloLens communications
- Status of the battery



**Figure 42:** Interface of the application.

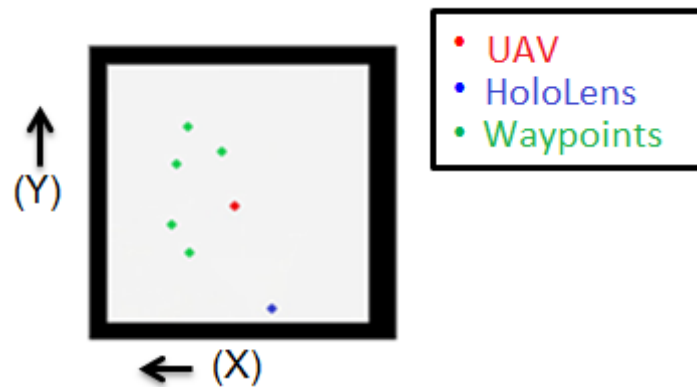
As it can be seen in the previous figure, there is a microphone label. The application has the possibility of being subscribed to the UAV camera. For simplification of the system, it can only be activated by voice commands. The commands to interact with the UAV camera and the actions performed on the interface are shown in Table 2.

**Table 2:** Voice command for UAV camera control

Voice command	Interface
<i>Show/remove camera:</i> It enables/disenable the view of the UAV CAMERA	
<i>Interface camera:</i> Allows the camera view to be placed in a full plane. In this case, the rest of the information from the drone is no longer received.	
<i>Exit:</i>	As Figure 42

Remove the vision of the camera in any case	
---	--

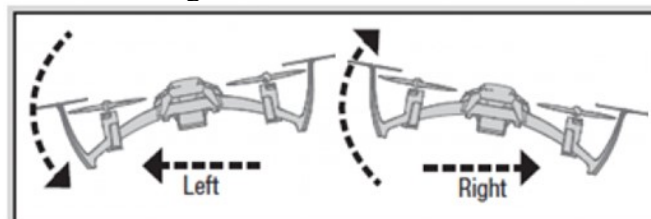
Following with the interface description, in the bottom left of Figure 42 appears a map, that can include the UAV position, HoloLens position and the waypoints through which the UAV is expected to flight, if received (see Figure 43).



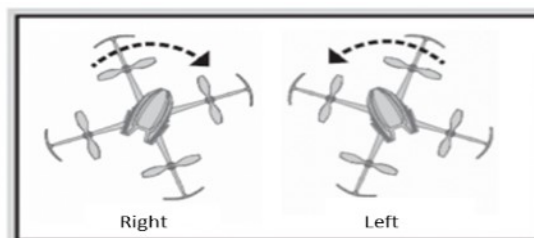
**Figure 43:** Map display.

And finally, there is an artificial horizon display, that follows user's look but it rotates depending on the UAV pose. It has three different parts, shown in Figure 44. It was one of the most interesting developments for the application, and one of the most appreciated by the pilots. Its functionality is as follows:

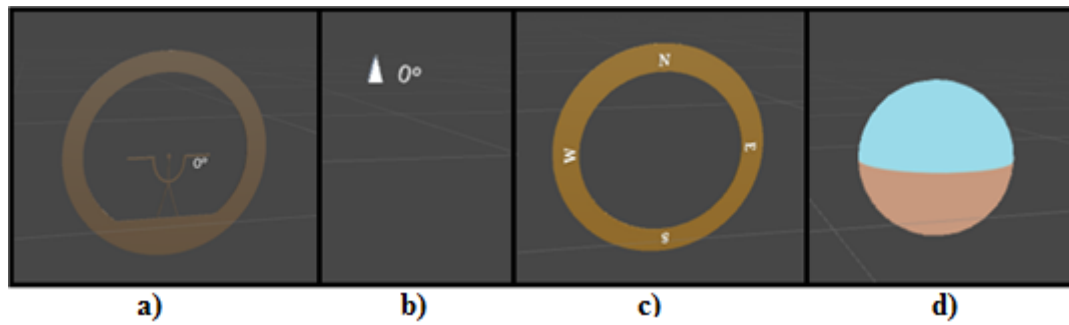
- The base (Figure 44, a), will stay fixed in its position, and it will display the degrees that the UAV has moved in yaw.
- Figure 44, b) shows the Roll indicator. It will display the number of degrees that the UAV is moving in that direction following the direction of the image below, being positive when the UAV moves to the left side and negative otherwise.



- The yaw indicator is the one in Figure 44, c). It follows the direction indicated in the following figure:

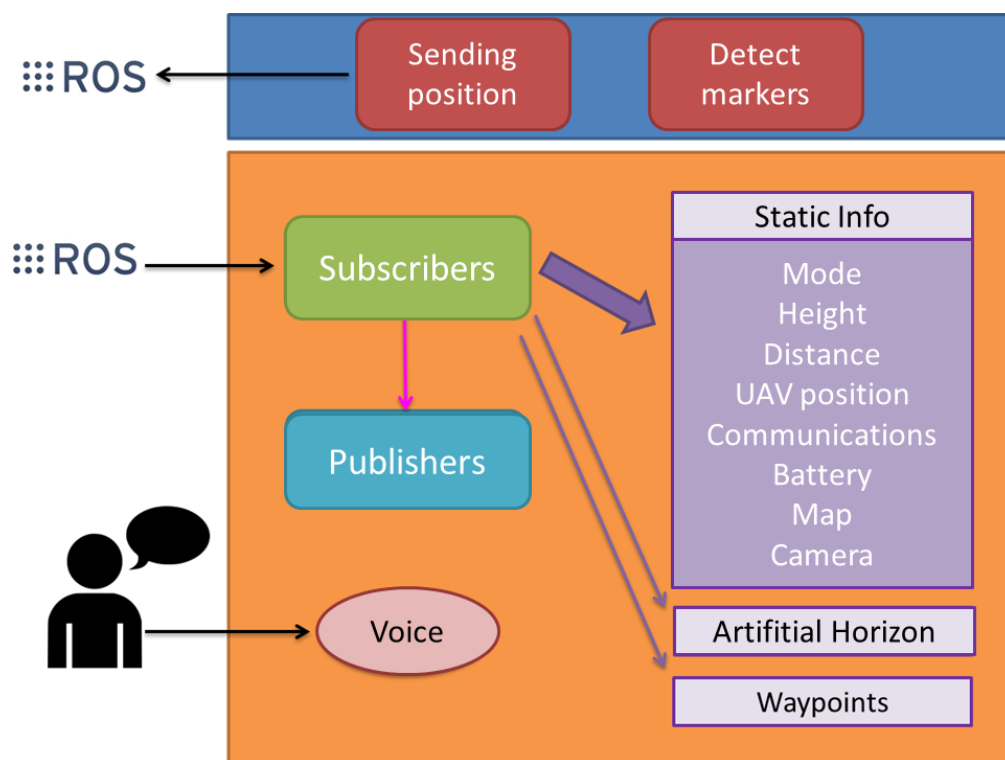


- Finally, Figure 44, d) shows the artificial horizon. It mainly moves with the UAV pitch inclination, but it is also affected by the roll movement.



**Figure 44:** Artificial Horizon display component: a) Base, b) Roll indicator, c) Yaw indicator and d) Artificial horizon.

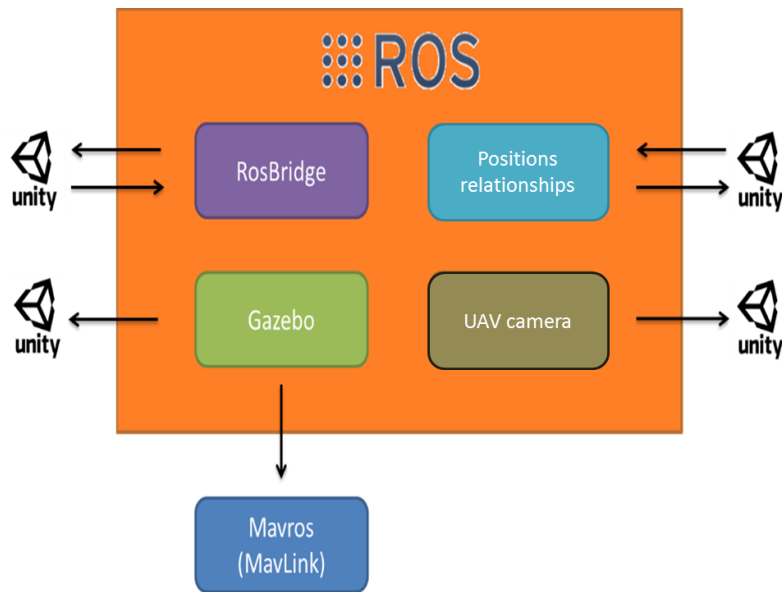
To conclude, it has been explained the interface designed for the purpose of this application, which has been developed with Unity in Windows OS, as summarized in Figure 45.



**Figure 45:** Unity software architecture.

### 3.2.2. ROS

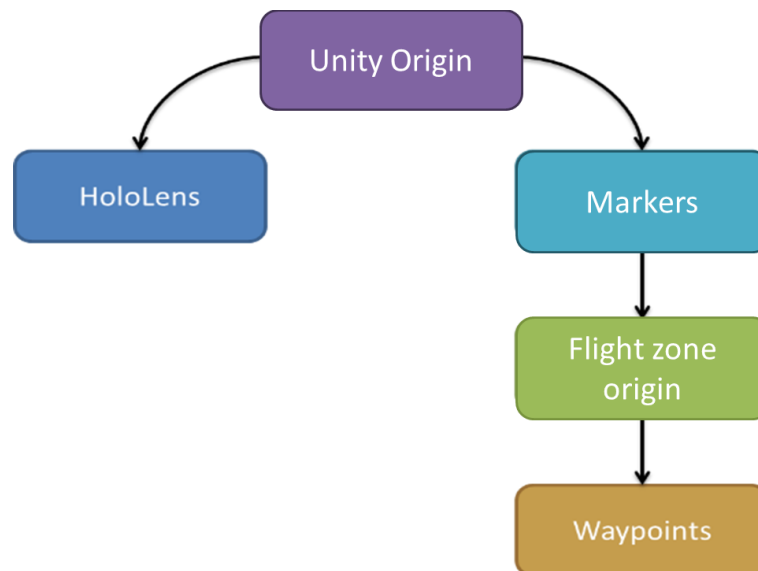
The main objective of the implementation carried out within the ROS environment is to allow the exchange of data between the UAV and the goggles. ROS works with publishers and subscribers. Several topics were declared to exchange the required data. These topics and the communication flow are shown in Figure 46. Gazebo is a simulator that allows recreating a 3D environment to test the application before the real flights. It is an independent ROS node.



**Figure 46:** ROS topics and data flow.

All the scripts are launched at the same time, although until the RosBridge is not activated by the user, the data coming from Gazebo is not possible to be gathered.

The tree that is generated for the relationship between all the positions is illustrated in Figure 47. In this way, it is possible to know the position and orientation of either the goggles or the UAV, in any of the available reference systems (Flight zone and Unity). Due to that tree, it is possible to subscribe Unity to any conversion that it needs for the construction of the interface.



**Figure 47:** Tree of the relationship between all the positions.

### 3.3. Experiments and results

After the application design process, several experiments were carried out. Before getting to the real flights, which were carried out in CATEC's testbed, the different cases that could occur were simulated. Since it has been a novel application development, the simulation phase was carefully

reviewed. It was not until all the simulations were passed that the glasses were tested with a UAV in a real flying environment.

Before the explanation of the experiments, in Table 3 it can be seen the composition of the ROS message passed to Unity. As it can be seen, this message contains practically all the data needed for the application.

**Table 3: ROS message**

<b>Data type</b>	<b>Message</b>
String	Mode
Float32	Height
Float32	Distance
Float32	Speed air
Float32	Speed ground
Float32	Communication
Float32	Battery
Float32	Position x drone
Float32	Position y drone
Float32	Position z drone
Float32	Rotation x drone
Float32	Rotation y drone
Float32	Rotation z drone

Another message was set, this one related to the waypoint. This message allowed displaying the waypoints, indicating where these marks are located. The message has the structure of Table 4.

**Table 4: Waypoints ROS message**

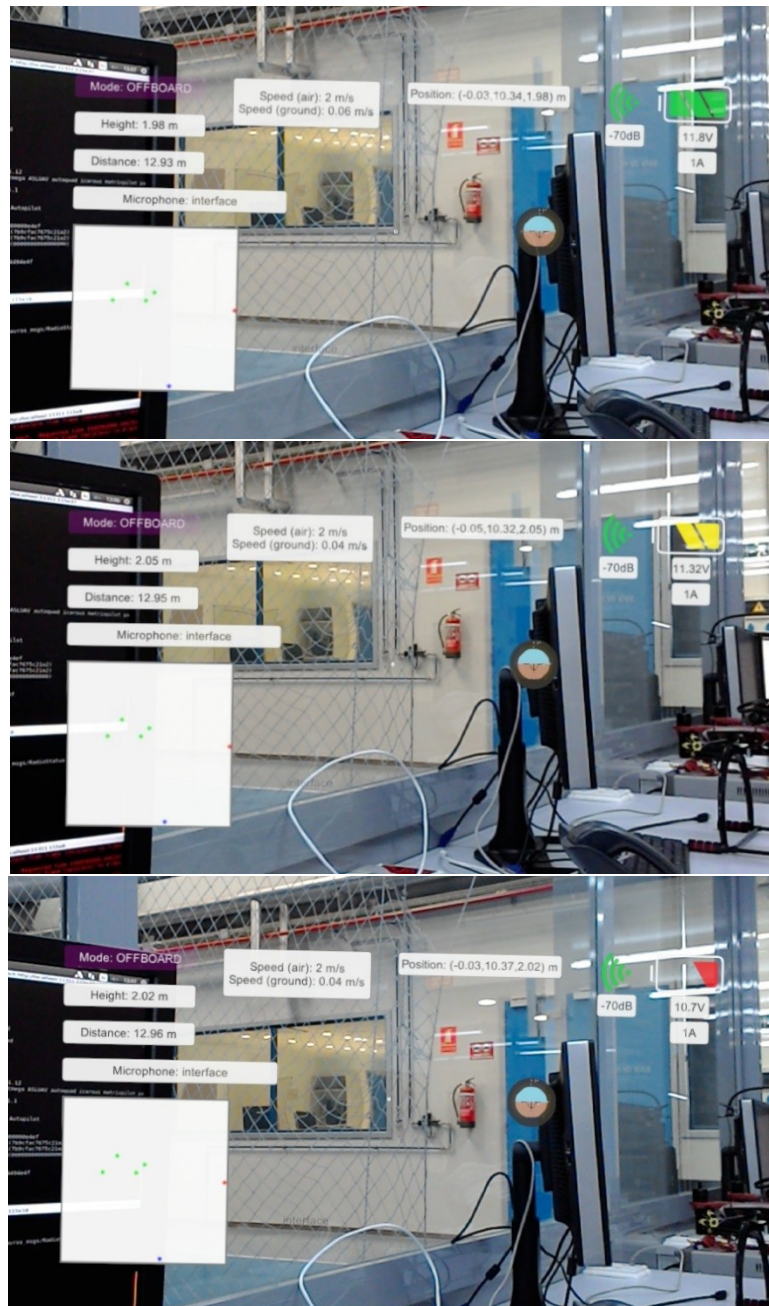
<b>Data type</b>	<b>Message</b>
Header	Header
geometry_msgs/Pose[]	Waypoints

It was not until all the simulations were overcome that the goggles were tested with a real UAV flying. The following are the results of both phases.

### 3.3.1. Simulations

Firstly, the experiments consisted in the recreation of several simulated scenarios. The first one verified the correct communication and data exchange between Unity and ROS. For that purpose, a publishing node was created, with the full message mentioned before. In that way, the topic was manually modified. Hence, it could be checked that Unity responded correctly to the change of the information and thus, the interface was updated. For instance, in Figure 48, it is shown the change in the battery indicator.





**Figure 48:** Battery indicator static test.

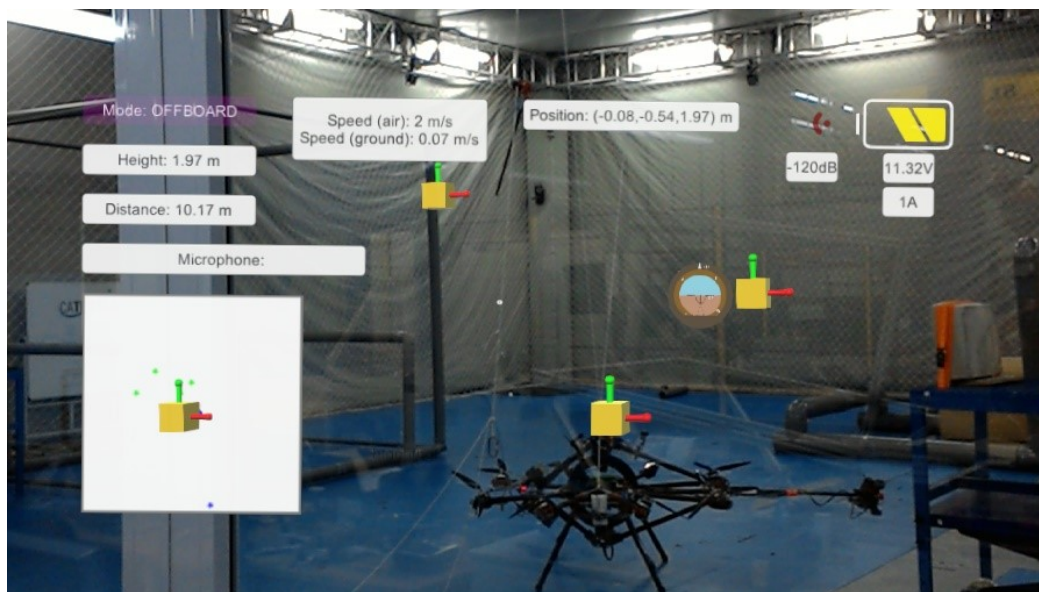
On the other hand, as it was explained in subsection 3.2.1, first of all the goggles must identify a marker to set its position in the real world, as well as its orientation (see Figure 49). For the application performed, it was needed to set the (0, 0, 0) to be assigned to the centre of CATEC's testbed. The system could then determine the initial point and hence Unity was able to calculate the conversions needed for positioning.





**Figure 49:** Initialization point.

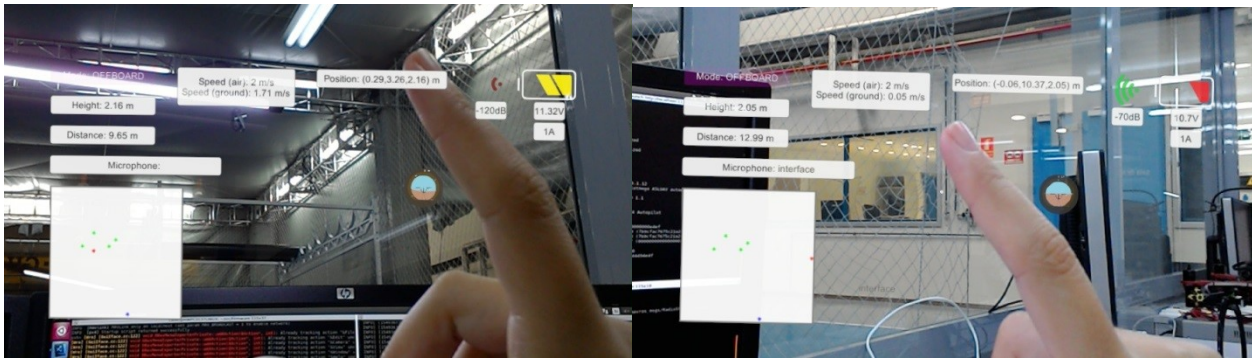
Moreover, Figure 50 shows the test that was performed on the way point indicator and in the communications intensity indicator. Starting with the last ones, it can be seen as when the intensity falls below the set threshold, it changes according to the design presented in Figure 42. The waypoints are displayed as the yellow boxes on the interface. It is a 3D representation of the waypoints in green shows on the map in the bottom left side of the interface.



**Figure 50:** Waypoint display and communications intensity.

After that, all the ROS topics were modified manually, and since Unity was subscribed to them, the information was displayed on the interface and it was uploaded on time. For instance, in Figure 48, it is shown the change in the battery indicator.

Finally, the labels related to the position of the UAV, its height and distance to the goggles, as well as its air and ground speed were tested. As it was mentioned, this information was manually changed on its ROS topics. Unity changed directly the information on the corresponding label (see Figure 51).





**Figure 51:** Position, speed, height and distance indicators test.


Tests on the voice block and on the gesture block where as well performed. Once that the entire interface was tested, the application was tested in a simulated flight. It is explained in the following section.

### 3.3.2. Simulations in Gazebo

Gazebo is a simulator that allows obtaining information from the simulated drone. It has been used to corroborate that the previous manual tests had a correct update with the drone in motion. In that case, the application was subscribed to the proper Gazebo topics, which substituted the ROS message, and cover the whole labels and indicators of the interface. The relationship between the Gazebo topics and the labels, as well as the data type, is shown in Table 5.

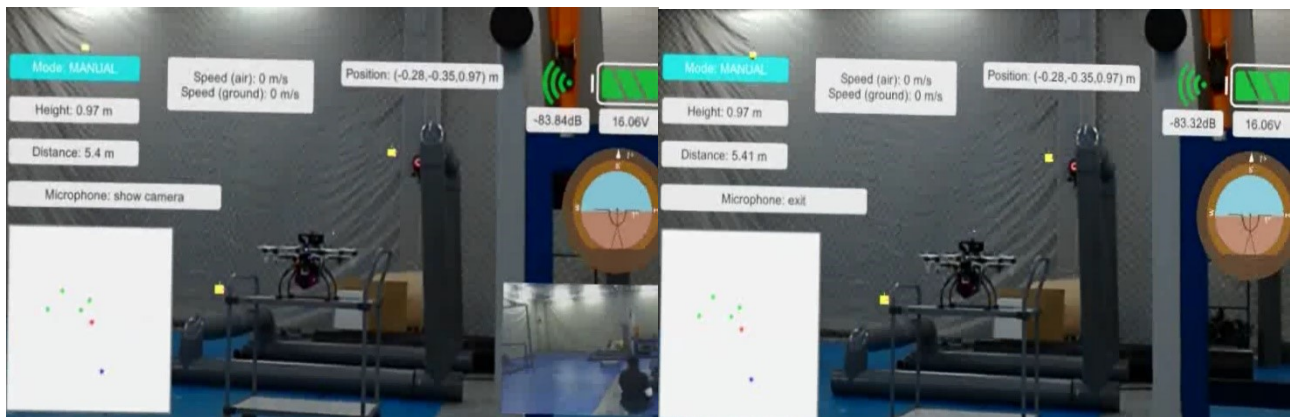
**Table 5:** Data type and Gazebo topics relationship with the interface

Data type & name	Gazebo topic	Label/Indicator
string mode	mavros/state	Mode:
float32 height	mavros/local_position/velocity	Height:
float32 distance	mavros/local_position/pose *	Distance:
float32 speed_air float32 speed_ground	mavros/wind_estimation mavros/local_position/velocity	Speed (air): Speed (ground):
float32 communication	mavros/radio_status	 dB
float32 battery	mavros/battery	 V

float32 position_x_drone float32 position_y_drone float32 position_z_drone	mavros/local_position/pose	Position:
float32 rotation_x_drone float32 rotation_y_drone float32 rotation_z_drone	mavros/local_position/pose	

### 3.3.3. Real application experiments

Once the tests in Gazebo were passed, the next step was to test the application with a real UAV. The first test with a real UAV (see Figure 52) was static, and personnel from CATEC moved it to check that the data changed correctly. In addition, during this test the microphone block was also checked, and as it can be seen in the Figure below, the *show camera* and *exit* voice commands worked correctly.



**Figure 52:** Static test with a real UAV.

This allowed checking all the topics and the map, as well as the correct functioning of the microphone, thanks to a USB camera that was connected to a computer. This camera simulated the one that is supposed to be mounted on the UAV. The interface was shown to CATEC's pilots, who made their contributions. It made the interface design more robust and suitable for real flights.

Several real flights were carried out. The correct operation of the application was checked thanks to several simple flights in which the UAV had to vary its pose (position and orientation). The flight mode was also changed and the voice control block and the gesture control block were checked. The application uploaded the information in real time. Some images of these experiments are shown in the following figures.

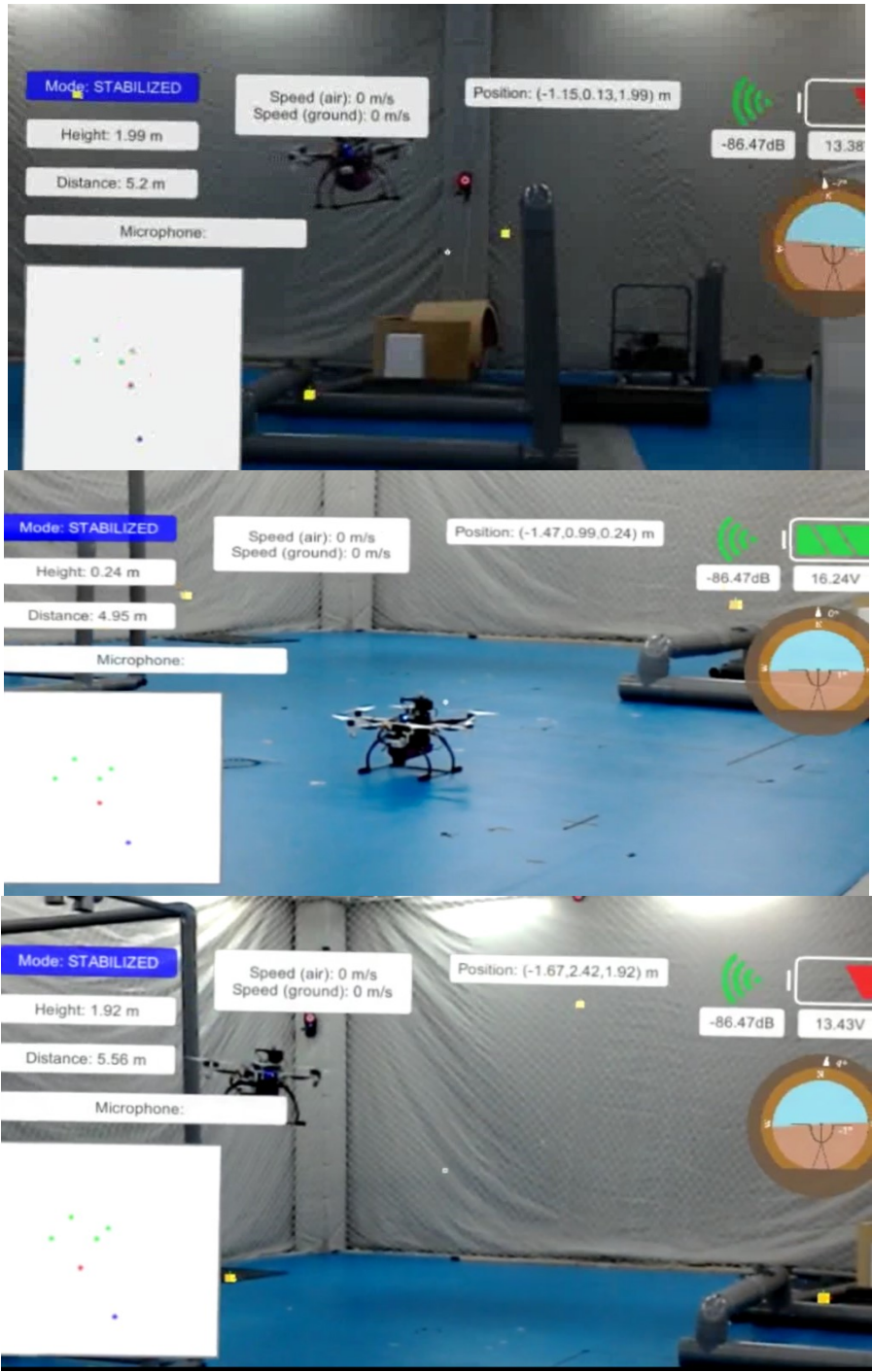


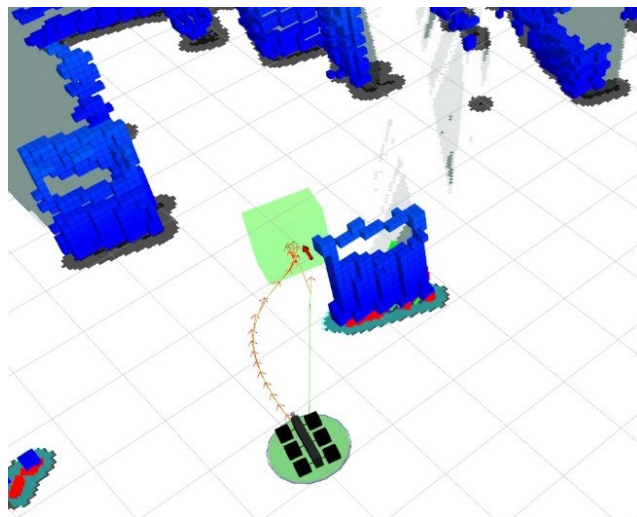
Figure 53: Real flights experiments.



## 4. Remote control support on the mobile support platform PC (T4.2)

There has been some further testing with the remote MSP (mobile support platform) GCS (ground control station) PC (personal computer), which is planned to be integrated into the MSP computer case, using ROS for visualizing and controlling robot navigation. The navigation related visualizations have been initially tested utilizing mostly the Mörrri unmanned ground vehicle (UGV) platform at UOULU, Figure 54, but the visualization is transferrable to the flying robot case. For these tests, both robots were equipped with similar sensors, but Mörrri also has LiDAR that the HR may not have, due to weight limitations.

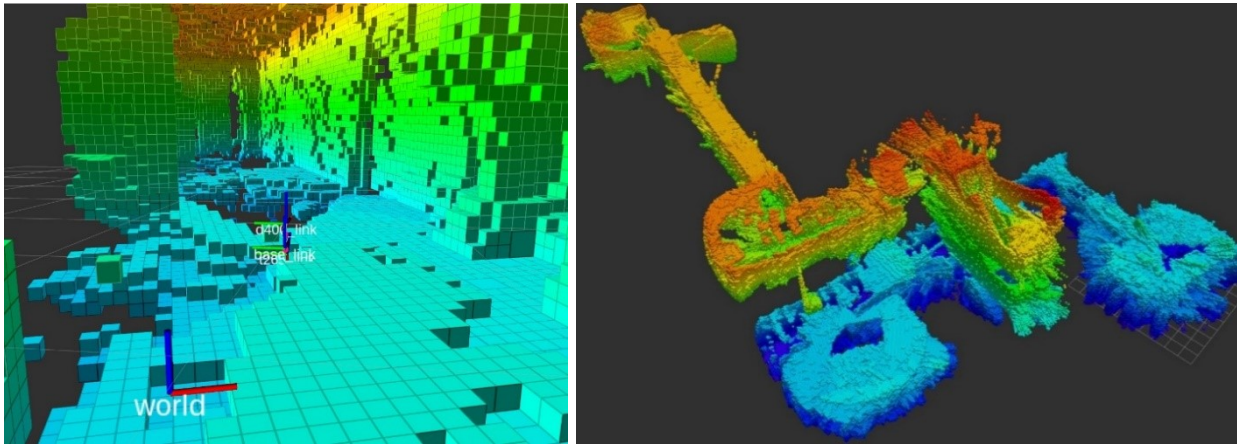
We have tested with UAV the occupancy mapping using the Intel RealSense T265 and D435 cameras, Figure 55, where the T265 provides the visual tracking odometry for the Pixhawk based flight control unit (FCU) through ROS vision\_to\_mavros package [Vis2MR], and the D435 is used for point cloud measurements. For collision free robot path planning testing, using the MSP computer in UOULU, we are currently testing the NanoMap for ROS package [NaMaR] for the UAV control environment. For localization support without GPS, we are looking into using the recently published ORB-SLAM3 [OrbSL3], or OpenVSLAM [OpVSL] depending which is better. These should be usable to generate a map reference in new environments, when combined with IMU and visual odometry tracking information, at least for testing purposes.



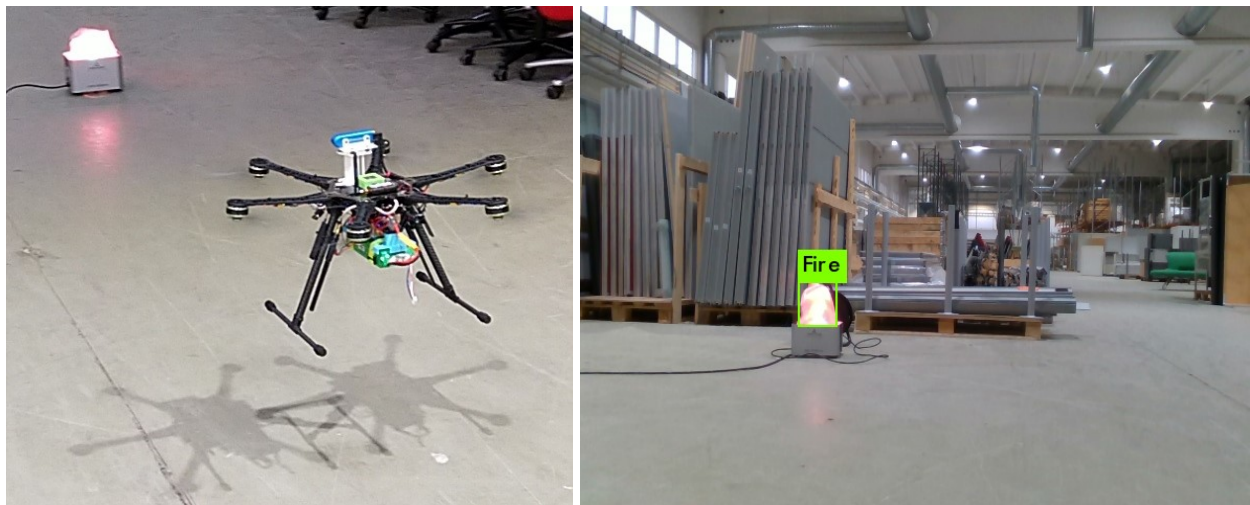
**Figure 54.** Remote GCS control testing with the mobile robot in the dark green circle. Set route shown with green arrow, end position with the light green box, and obstacle avoiding planned route output shown with orange arrows.

We have also done some experimentation, Figure 56, using the YOLOv3 based deep learning object detection system [YOLOv3] on the Nvidia Jetson TX2 to detect and 3D-localize specific objects in relation to the UAV, using RGBD (red, green, blue, depth) image. However, we are working towards a custom implementation for improved performance on UAV platforms. We are currently upgrading our test UAVs with Nvidia Jetson Xavier NX onboard computers, which have roughly twice the processing power of Jetson TX2, and feature dual NVIDIA Deep Learning Accelerator (NVDLA) cores to significantly accelerate deep learning related functions. Also, we are looking at transferring some of the deep neural networks and mapping related computational tasks off the UAV onto the GCS PC, which may be later extended further to other cloud computational services available in

mobile edge networks, such as 5G, although utilizing edge computing is not included in HYFLIERS project.



**Figure 55.** Octomapping [OctoM] performed with a test UAV, using RealSense D435 camera for depth measurements and T265 for odometry, which is also given as optical flow stream input to the Pixhawk FCU. On the right, the UAV was flying on the first floor (blueish), and then traversed by walking up to the third floor (yellow – red) via a staircase and to an office space at the end of a corridor.



**Figure 56.** Object detection testing on the UOULU hexacopter test UAV.

## 5. Mobile ground support platform (T4.3)

In this section are described the advancements in the development of the mobile support platform. The finalized choices are presented and the current stage of the physical implementation of the necessary hardware are presented. The MSP includes the support functions required in the HR remote operation and sufficient capabilities required in the use cases defined in [D1.2] and [D1.4].

### 5.1. MSP cart

The MSP cart, Figure 57, is a pushable cart, based on customized Filmcart SMARTONE MICRO production cart. The cart has “offroad” terrain wheels, for improving the movement capabilities of the cart. The top part has a detachable GCS PC box, which includes support for the PC and UI devices related to HR control and monitoring, and the power supplies for the related hardware. The bottom part has an electrical box including the main battery and AC power system, referred as the MSP power supply box, that supplies power for the PC box and HR charging related equipment. The top part can be detached so that the cart can be transformed to an easily transferrable form. Two of the wheels could be customized to have BLDC (Brushless DC) motors, to aid in moving the cart around. However, this would require some minor modifications to the current implementation of the MSP power box. Also, as the motorized wheels have been considered an optional feature from the start, they are currently not being implemented, but this choice may be revised if it is necessary.



**Figure 57.** The collapsible and pushable MSP cart for mounting the HR support related hardware.

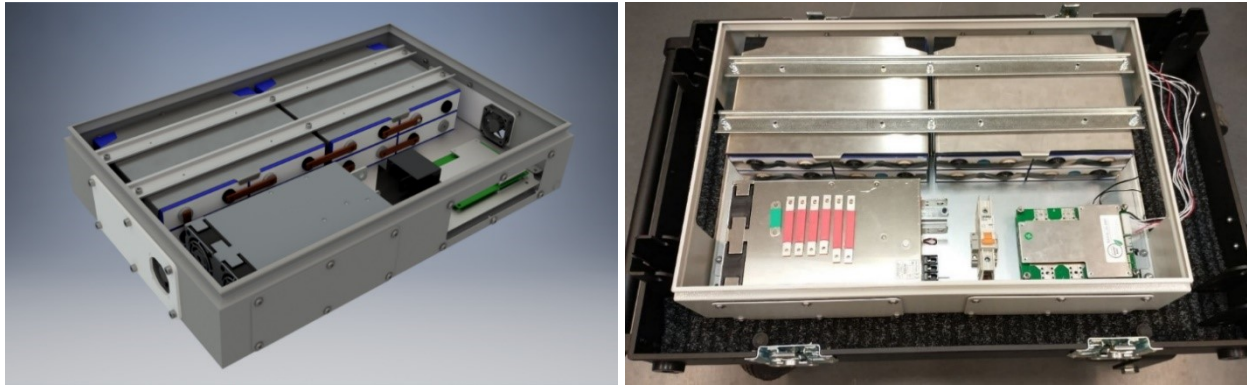
### 5.2. Main power battery system overview

The main power system of the cart is an 8-cell LiFePO<sub>4</sub> (Lithium Iron Phosphate) battery pack, with 105 Ah capacity. Originally, LiPo (Lithium Polymer) batteries were intended to be used, see [D4.1]. However, the availability of reputable good LiPo batteries is poor. Moreover, implementing the related electronics for constructing a high-capacity modular battery, which would have had to consist of several multi-cell LiPo batteries in parallel to achieve high capacity, would have been complex and reduced the system overall reliability and safety. For these reasons, LiFePO<sub>4</sub> batteries have been selected as they are significantly safer and more reliable, compared to other commonly available LiPo battery chemistries. The battery management system (BMS) was initially thought to be a fully custom solution developed earlier at UOULU, but a ready-made battery pack management module by Overkill Solar (BMS 100a 8s) was selected, which also circumvents the potential usage right issues related to the above UOULU intelligent battery modules.



### 5.3. MSP power supply box

The MSP's current total main battery energy capacity is 2.5 kWh, with a voltage range of 20 to 28.8 V, which is subsequently the voltage output range of the MSP power box to the PC box and the HR charger. The power limit for the PC, including the user interface (UI) and communications related hardware, is set as 500 W. The HR charger DC power supply is limited by a fuse to be 1000 W. With the 2.5 kWh main battery, the system should be able to be used to charge the HR several times, in situations where an AC power is not available, while simultaneously running the GCS PC and communications hardware. The power supply box includes the main battery pack, the custom power & charging power management system (PMS) module, the BMS module for the battery, and a 1 kW AC-DC power supply unit (TDK Lambda SWS1000L-24 PSU), with mains AC input range of 85 - 265 V. The power box internal structure is shown in Figure 58.



**Figure 58.** The computer-aided design (CAD) drawing and the realized main power supply box of the MSP with the mounted hardware.

### 5.4. Custom power management system board

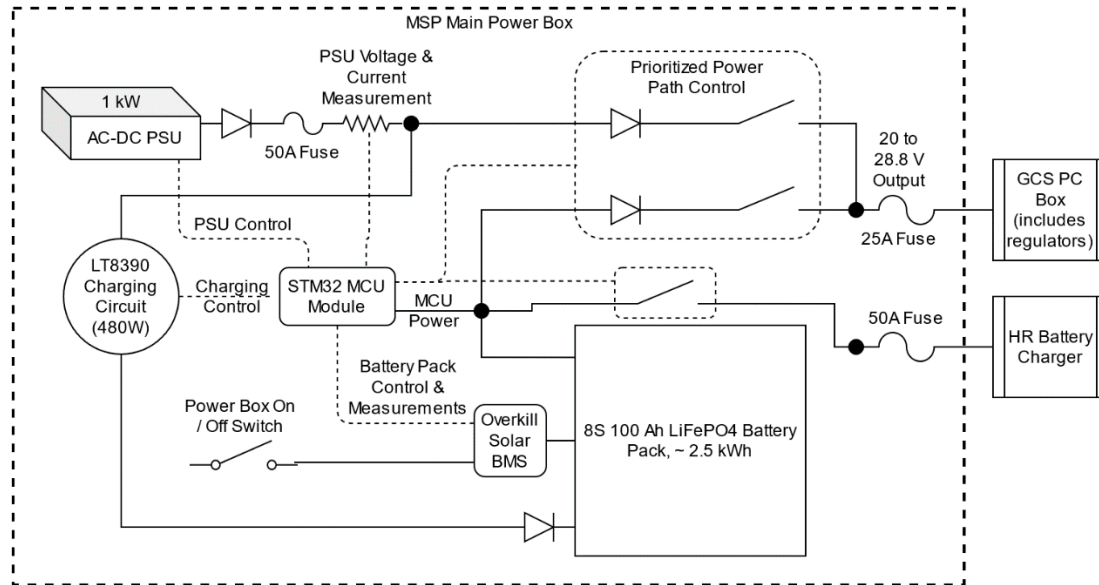
The custom power management system board (PMS board) handles the features that were not implementable with off-the-shelf components. The biggest difficulties were related to solving the uninterrupted power supply requirements and system management requirements. The MSP should be able to work uninterrupted when it is connected and disconnected from AC power during operation and this is handled with the custom solution presented here, with an overview presented in Figure 59.

The PMS board includes an LT8390 based charger, which charges the 2.5 kWh main  $\text{LiFePO}_4$  battery at maximum 480 W power. During operation, the power taken by the GCS PC and the HR charger are monitored, so the maximum allowed main  $\text{LiFePO}_4$  battery charging power is determined from how much power reserve is available from the 1 kW AC-DC PSU. For simplicity, the HR charger takes the power directly from the MSP main battery, which can output a large power for the charger without having to worry about supply capabilities of the PSU and the consumption by other support hardware.

In the PMS board, the STM32F446RET6 microcontroller unit (MCU) is controlling the power output and the real time monitoring and charging control of the MSP main battery. More specifically, the MCU handles the control of the LT8390EFE based battery charging circuitry, communication with the Overkill Solar BMS, controlling the output of the AC-DC PSU and the real-time control of the power bus matrix between all the connected components within the power supply box. Other main controllers in the power bus system structure are the LTC4421 power path controller for driving power path control N-MOSFET switches, and LM74700-Q1 for driving the N-MOSFET based low-



loss diodes. Additionally, the board has a connector for the external OLED screen, for showing the power box UI elements and measurements, that are relevant to observe the correct functioning and status of the power supply box. There are also high-side current and voltage sensing integrated circuits (ICs) used in the bus matrix, like the INA240A3D current sense amplifier, plus other smaller components not detailed here. The simplified functional overview of the custom PMS board and main power box related electronics is shown in Figure 59. The “HR Battery Charger” component shown on the right-hand side here refers to all the charging cases, either a battery charging dock or a ‘contactless’ charger that have been presented in [D1.4]. The battery charger output will be cut off automatically by the MCU, if the available energy level of the MSP drops too low.



**Figure 59.** Simplified overview of the PMS board functionality.

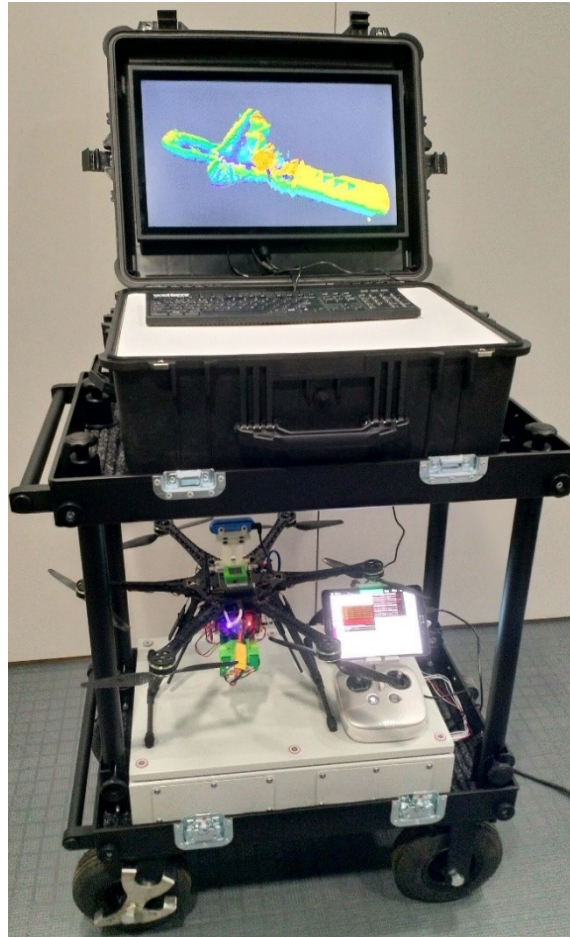
The printed circuit board (PCB) design still requires some finishing touches, especially on the side of thermal management of the board components.

### 5.5. MSP HR support PC box

The box is reasonably large so that all the required MSP elements could be contained in the case that are required for communications and support functions related to HR operation. The box contains the additional support computer, for database and possible navigation support functions, Wi-Fi and Ethernet networking support, waterproof WetKeys keyboard and mouse, and a 22-inch Beetrionics 22TS7M Full HD multi-touch screen with IP65 rating (water and dust protection). The box also has waterproof input for HDMI, Ethernet and USB, for attaching additional external devices to the support computer, such as a separate GCS (ground control station) for the HR and pilot's data stream.

The PC box is not permanently fixed to the cart and only receives the DC power through a 2-pin rugged plug from the MSP power supply box, described in Section 5.3 and shown on top of the cart in Figure 60. The PC box dimensions are 626×492×330 (mm), and it is a water-resistant carry case, which is customized to include the electronics required for the remote operation support functions of the HR. The computer was initially planned to be a fully-fledged PC, Zotac ZBOX with either GTX 1070 or RTX 2070, but may be updated with a newer model. The Wi-Fi & Ethernet router is currently Asus ROG Rapture GT-AC5300, with 8 RJ-45 ports and 8 Wi-Fi antennas, and the antennas are mounted inside the box as it is ABS plastic and should have little effect on the Wi-Fi signal strength.

The screen has a 9 to 36 V input range, therefore it does not need a regulator and is supplied directly by the output voltage from the MSP power box. The PC power is supplied via a dedicated DC-DC converter, and additional regulators can be added as needed, for example 19 V for Wi-Fi. If other high-power regulators are required in the PC box, additional custom DC-DC converters with any output voltage (up to 60 V) may be realized using the same LT8390 based controller, that is being utilized in the charging circuit of the main battery.



**Figure 60.** The MSP PC box connected to the hexacopter over Wi-Fi, visualizing the occupancy grid map being generated onboard the UAV.

### 5.6. MSP development and testing

The progress with the MSP support cart was slowed down significantly by the COVID-19 (coronavirus disease) pandemic, and several critical laboratories were in lockdown for about four months, preventing the design and construction of some of the mechanical parts of the MSP cart. Also, there were significant delays in getting some of the components for the MSP, both electrical and mechanical. Therefore, the MSP cart has not yet been used to complete extensive testing for the software and hardware implemented.

## References

- D1.2 Zesch W, Garrido FJ and Trujillo MÁ, eds. (2020) System concept and architecture. H2020-ICT-25-2016-2017 779411 HYFLIERS project deliverable D1.2 (CO). Version 2.0. Feb 2020.
- D1.4 Garrido J and Trujillo MÁ, eds. (2019) Final system specification, concept and architecture. HYFLIERS project deliverable D1.4 (CO). Dec (Revised Mar 2020).
- D4.1 Celentano U, ed. (2019) Measurement data management service and path planning algorithm. H2020-ICT-25-2016-2017 779411 HYFLIERS project deliverable D4.1 (CO). Jun (Revised Feb 2020).
- CabEtal2018 Caballero A, Bejar M, Rodriguez-Castaño A, and Ollero A. (2018). Motion planning with dynamics awareness for long reach manipulation in aerial robotic systems with two arms. *The International Journal of Advanced Robotics Systems*, 15(3), 2018.
- DorEtal2017 Dorling K, Heinrichs J, Messier G.G, and Magierowski S. (2017). Vehicle Routing Problems for Drone Delivery. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(1), pp. 70-85, Jan 2017.
- KanLev1985 Kane TR, and Levinson DA. (1985). *Dynamics, theory and applications*. New York, NY, USA: McGraw-Hill.
- KarFra2011 Karaman, S. and Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7), 846–894.
- KiCd <https://kicad-pcb.org>
- LeiMS50 [https://w3.leica-geosystems.com/downloads123/zz/tps/nova\\_ms50/white-tech-paper/leica\\_nova\\_ms50\\_tpa\\_en.pdf](https://w3.leica-geosystems.com/downloads123/zz/tps/nova_ms50/white-tech-paper/leica_nova_ms50_tpa_en.pdf).
- NaMaR [https://github.com/peteflorence/nanomap\\_ros](https://github.com/peteflorence/nanomap_ros)
- OctoM <https://octomap.github.io>
- OpVSL <https://github.com/xdspacelab/openvslam>
- OrbSL3 [https://github.com/UZ-SLAMLab/ORB\\_SLAM3](https://github.com/UZ-SLAMLab/ORB_SLAM3)
- SuaEtal2020 Suarez A, Caballero A, Garofano A, Sanchez-Cuevas P.J, Heredia G, and Ollero A. (2020). Aerial manipulator with rolling base for inspection of pipe arrays. *IEEE Access*, vol. 8, pp. 162516-162532, 2020.
- Vid-S <https://hdvirtual.us.es/discovirt/index.php/s/siHzJLMcqXKiF3s> (see also Appendix for details and individual links)
- Vis2MR [https://github.com/thien94/vision\\_to\\_mavros](https://github.com/thien94/vision_to_mavros)
- YOLOv3 <https://pjreddie.com/darknet/yolo/>

## Appendix: Supplemental material

For ease of use, here below it is reported the list of videos available in [Vid-S]. Links to the videos are embedded.

[Scen\\_1\\_MPHR\\_time.mp4](#)

Simulation of the MP-HR algorithm minimising the operation time in a moderately cluttered area, see Section 2.7.1.

[Scen\\_1\\_MPHR\\_energy.mp4](#)

Simulation of the MP-HR algorithm minimising the energy consumption in a moderately cluttered area, see Section 2.7.1.

[Scen\\_2\\_MPHR\\_time.mp4](#)

Simulation of the MP-HR algorithm in a highly cluttered area, see Section 2.7.2.

[Scen\\_2\\_MPHR-DA\\_time.mp4](#)

Simulation of the MP-HR-DA method in a highly cluttered area, see Section 2.7.2.

[Scen\\_3\\_MPHR\\_reactive.mp4](#)

Simulation of the MP-HR hybrid reactivity in a moderately cluttered area with unmapped obstacles, see Section 2.7.3.

[Exp\\_MPHR\\_energy.mp4](#)

Experiment with the MP-HR algorithm, see Section 2.8.1.