WP7 Software Architecture and Validation Scenarios

August 30, 2021

Revision History

Version	Date	New Document	Author
0.01	20-07-2021	First scheme and draft	USE
0.02	30-08-2021	Add Scenario description. Document revision. Add Q&A.	CTU

List of Participants

Participant Organization Name	Acronym
Aristotle University of Thessaloniki	AUTH
Czech Technical University	CTU
Donecle	DON
University of Seville	USE
University of Twente	UT
Terabee	TER

Disclaimer

The aim of this document is to agree with the use case scenarios and start defining a road map for the integration. The research solutions presented in Sections 1.1 and 1.2 may undergo slight changes depending on the research results achieved and the project specifications.

Contents

1 Scenarios description

This section describes the scenarios proposed for validation of the methodologies and approaches developed in Work Package (WP) 7. The goal of this WP is to provide the methods and algorithms that will constitute the core of the software of the Aerial Co-Workers (ACWs)[1]. The power line inspection problem is taken as motivating example. Three types of ACWs are referred, each intended to provide different functionality: *Inspection-ACW*, *Safety-ACW*, and *Physical-ACW*. The use case scenarios can be summarized as follows:

- *Inspection*, where a fleet of ACWs (i.e., *Inspection-ACWs*) carries out a detailed investigation of power equipment autonomously, helping the human workers to acquire views of the power tower that are not easily accessible (see Fig. 1a);
- *Safety*, where a formation of ACWs (i.e., *Safety-ACWs*) provides the supervising team with a view of the humans working on the power tower in order to monitor their status and to ensure their safety (see Fig. 1b);
- *Physical*, where an ACW (i.e., *Physical-ACW*) physically interacts with the human worker and provides physical assistance to it, i.e., while in contact with the human it flies stably, reliably, and accomplishes the required physical task (e.g., handover of a tool) without becoming harmful for the human worker (see Fig. 1c).

1.1 Inspection-ACWs for inspection operations

This scenario aims to demonstrate the functionality of *Inspection-ACWs* within the project. It is meant to be used both as an academic and project scenario. Since the effort is mainly related to CTU activities, the academic scenario will be in Prague (CTU facilities, September 2021 - April 2022) while the project scenario will be in Lausanne (EPFL[2] facilities, July 2022) and in Spain (ATLAS[3], date to be defined).

Contributions: The scenario seeks to combine and use contributions coming from tasks T7.2, T7.3, and T7.4.

Description: The inspection system is in charge of computing feasible and constrained trajectories for a fleet of Inspection-ACWs (i.e., under-actuated vehicles) that have been assigned an inspection task together by a *High-level decision-making* sys-

[1] Aerial cognitive robotic systems that have capabilities on the operational range and safety in the interaction with people. In other words, under-actuated and fully-actuated multi-rotor aerial vehicles with cognitive capabilities, e.g., based on novel perception sensors, such as event cameras, advanced data fusion techniques, or fast on-line planning.

[2] École Polytecnique Fédérale Lausanne.

[3] ATLAS centre owned by FADA-CATEC (Fundacion Andaluza para el Desarrollo Aeroespacial). This centre is specialized in Unmanned Aerial Vehicles (UAVs) and aerial robot testing and has all the facilities needed for the testing of the high Technology Readiness Levels (TRL) modules and integrated AERIAL-CORE system facilities.

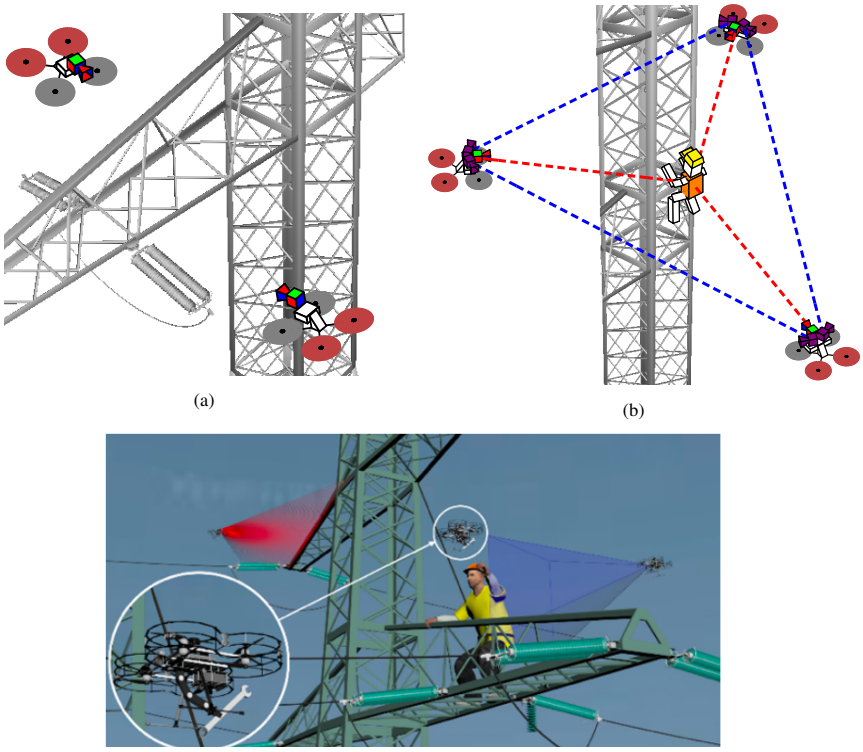

Figure 1: From left to right: *Inspection* (a), *Safety* (b), and *Physical* (c) scenarios. Onboard cameras and sensors acquire data of the power equipment. A mutual localization system helps to maintain the formation avoiding contact with the tower [1].

tem[4] (see. Sec. 2). In Figure 2 snapshots of a *power tower inspection* scenario are shown.

The Inspection-ACWs operate in a known environment, represented by a map (i.e., an occupancy grid map) that also includes the position of obstacles and the power tower, and are located using RTK GPS sensors. In the ACW hardware setup, cameras are mounted in an *eye-in-hand* configuration, i.e., rigidly attached to the body frame. An offline running planner [2] computes trajectories that avoid obstacles and maintain a safe distance between ACWs while also accounting for their velocity and acceleration constraints. The algorithm divides the inspection tasks among the available Inspection-ACWs to generate optimal strategies that satisfy the mission specifications.

The High-level decision-making system monitors the plan execution, integrating information perceived by the ACWs (i.e., their battery level, localization, and target state

[4]Inspection tasks are indicated with the sequence $\mathcal{I}_P = \{x, y, z, \psi\}$ along with the action to take (e.g., record a video or take a picture), where x, y, and z represent the spatial coordinates of the target in 3D space and ψ is the heading angle that ACWs use to orient themselves with respect to (w.r.t.) the target.

estimation), in order to re-plan online the mission if the initial plan is no longer feasible. In addition, this information is used to decide when an ACW has failed and needs to be landed urgently. For instance, to replace an ACW that may run out of battery before finishing its current task. A video with the preliminary numerical simulations in Gazebo and real flight tests are available at `https://youtu.be/KJL3Z2UmAVM` and `https://youtu.be/NJTujhn1feE`, respectively.

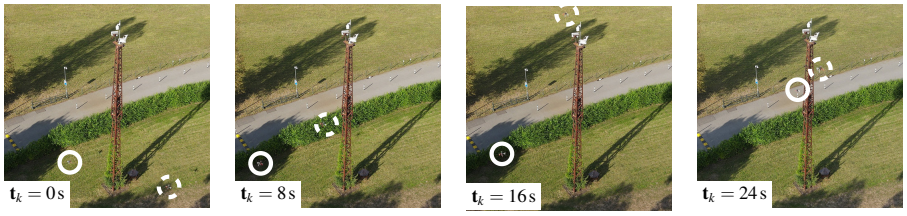

Figure 2: Snapshots of a *power tower inspection* scenario. The system evolution at different time instants $\mathbf{t}_k$ is reported. Solid and dashed circles are used to indicate the ACWs used to perform the mission [2].

Alternative scenarios: Here is a list of possible alternative scenarios to the one described above.

1. The Inspection-ACWs are located using the Terabee Real-Time Positioning System (RTPS) (beacon localization) instead of RTK GPS sensors. This scenario is meant to prove the connection with task T3.2.

Involved partners: CTU + Terabee (external WP connection, i.e., T3.2) + USE + DON. CTU will lead the task. Terabee will provide the sensor equipment (i.e., the RTPS beacon localization). USE will cooperate with CTU to integrate the High-level decision-making system (see Sec. 2) within the software architecture. DON will provide the under-actuated vehicles (i.e., quad-rotors) for the demonstrator and integrate CTU's algorithms on their platforms. As backup solution, CTU will use its own UAVs to cope with the task.

1.2 Safety-ACWs for surveillance tasks

This scenario aims to demonstrate the functionality of *Safety-ACWs* within the project. It is meant to be used both as an academic and project scenario. Since the effort is mainly related to CTU activities, the academic scenario will be in Prague (CTU facilities, September 2021 - April 2022) while the project scenario will be in Lausanne (EPFL facilities, July 2022) and in Spain (ATLAS facilities, date to be defined).

Contributions: The scenario seeks to combine and use contributions coming from tasks T7.2, T7.3, and T7.4.

Description: A fleet of Safety-ACWs (i.e., under-actuated vehicles) have been assigned a surveilling task together by the *High-level decision-making* system (see Sec. 2). The Safety-ACWs are required to monitor a human being working on an electrical tower ensuring its safety, i.e., images and videos are provided to the ground

supervision team who assist the operator working on the tower. In Figure 3 snapshots of the scenario along with the trajectories for the fleet of Safety-ACWs are shown.

The Safety-ACWs operate in a known environment, represented by a map (i.e., an occupancy grid map) that also includes the position of obstacles and the power tower. Both the Safety-ACWs and the human worker are located using RTK GPS sensors. In the ACW hardware setup, cameras are mounted in an *eye-in-hand* configuration, i.e., rigidly attached to the body frame. An online running planner computes feasible trajectories for the ACWs working in a leader-follower scheme. Obstacle avoidance capabilities guarantee compliance with safety requirements between ACWs and the human worker.

The High-level decision-making system monitors the plan execution, integrating information perceived by the ACWs (i.e., their battery level, localization, and target state estimation), in order to re-plan online the mission if the initial plan is no longer feasible. In addition, this information is used to decide when a ACW has failed and needs to be landed urgently. For instance, to replace an ACW that may run out of battery before finishing its current task. Moreover, the human worker can use gestures to interact with the ACWs by requesting for high-level actions, such as scaling the formation or change the camera viewing angles. In this case, the High-level decision-making system takes care of recomputing the mission plan. A video with the preliminary numerical simulations in Gazebo and real flight tests is available at `https://youtu.be/v4f7kb_fxjA`.

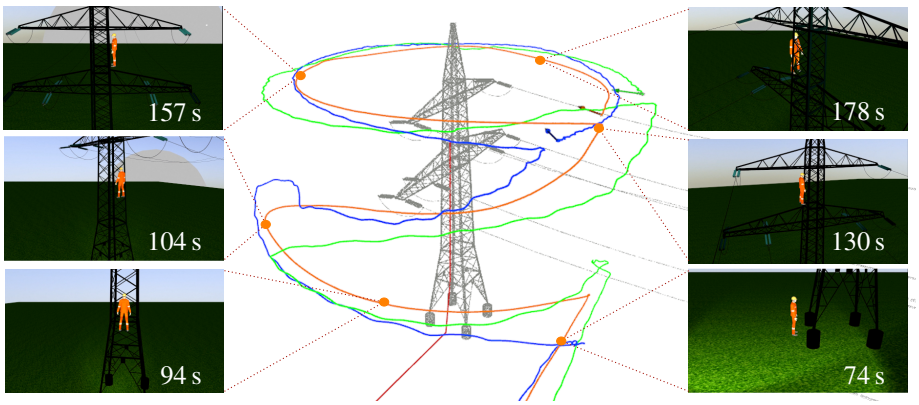

Figure 3: An illustration of the Safety-ACWs while monitoring a human being working on the power tower [3].

Alternative scenarios: Here is a list of possible alternative scenarios to the one described above.

1. Both Safety-ACWs and the human worker are located using the absolute and relative Terabee sensor (i.e., Follow Me sensor and RTPS) instead of RTK GPS sensors. This scenario is meant to prove the connection with task T3.2.

2. This task involves both Safety-ACWs and Physical-ACW. Safety-ACWs could oversee a tool delivery operation. Safety-ACWs provide visual feedback of the

operation to the supervision team on the ground who is in control of aborting the mission if something goes wrong (i.e., harmful scenario for the worker). As before, the ACWs are localized using RTK GPS (first attempt) or Terabee sensors (second attempt). The scenario can be meant as an academic and/or project scenario. CTU could provide the no-fly zone coordinates (equivalently the fly-zone coordinates) to the Physical-ACW. The coordinates could delimit an area in which Physical-ACW can safely move.

3. The same as 2), but now the AUTH module and Terabee VGA camera are used as feedback from the controller to know the distance between Safety-ACWs.

Involved partners: CTU + Terabee (external WP connection, i.e., T3.2) + USE + AUTH + DON. CTU will lead the task. AUTH will help with the human detection and gesture recognition. Terabee will provide the sensor equipment (i.e., the VGA Camera, Follow Me, and RTPS sensors). USE will cooperate with CTU to integrate the High-level decision-making system (see Sec. 2) within the software architecture. DON will provide the under-actuated vehicles (i.e., quad-rotors) for the demonstrator and integrate CTU's algorithms on their platforms. As backup solution, CTU will use its own UAVs to cope with the task.

1.3 Physical-ACWs for human-robot interaction

This scenario aims to demonstrate the functionality of *Physical-ACW* within the project. It is meant to be used both as an academic and project scenario. Since the effort is mainly related to UT activities, the academic scenario will be in Twente (UT facilities, September 2021 - April 2022) while the project scenario will be in Lausanne (EPFL facilities, July 2022) and in Spain (ATLAS facilities, date to be defined).

Contributions: The scenario seeks to combine and use contributions coming from tasks T7.1, T7.2, and T7.3.

Description: The use case scenario plans to show the following capabilities: (i) physical interaction (with a human worker or the environment in general), and (ii) the switching between under-actuated and fully-actuated configurations when necessary for safety issues or to optimize energy consumption. Tool delivery is an example of such scenario (see Section 5.3.4 of the WP2 deliverable D2.1). Here is a reminder of the steps:

- The ACW approaches the human worker within a predefined approach envelope, ensuring visibility and safety for the human. The approach is performed by combining the Terabee RTPS and RTK GPS sensors.
- The ACW deploys the tray containing or to contain the tool and waits for the human worker's permission to approach further. This permission is given by a press of the button of the Terabee beacon.
- The ACW approaches the human worker and waits for the start to perform the manipulation. During the final approach and throughout this phase, strongly damped control laws will exploit the over actuation to provide an elastic response to the interactions.

- When the transfer is performed, the human worker will signal that he has finished by a push back physical interaction. This interaction will be detected by the ACW as a command to move back and execute the rest of the mission plan.
- During these phases a flight quality checker (i.e., the High-level decision-making, see Sec. 2) is running on the low-level control board, to ensure that the ACW will stay safely in the predefined region and inside a feasible flight envelope, if this is not the case, for example due to extreme external disturbances, or an expected behavior from the operator, the ACW is able to pause (i.e., hovering) or stop the mission (i.e., safe landing)

Alternative scenarios: Here is a list of possible alternative scenarios to the one described above.

-

Involved partners: UT + Terabee (external WP connection, i.e., T3.2) + USE + DON. UT will lead the task. Terabee will provide the sensor equipment (i.e., the RTPS beacon localization). USE will cooperate with UT to integrate the High-level decision making system (see Sec. 2) within the software architecture. DON will provide the fully actuated platform for the demonstrator and integrate UT's algorithms on their platform.

1.4 Measurable goals

These are the measurable goals from Table 2 of the project proposal.

- Coordination of a fleet of ≥ 3 ACWs with different abilities (such as, Physical-ACW, Safety-ACW, and Inspection-ACW specified in Task 7.4) around the mockup: 60 min time of the cooperative ACW mission.
- Precision of ACW state estimation relatively to other ACWs and to workers: 2 % of relative distance in range 0 m to 10 m.
- Reliability of behavioral state estimation for cognitive interaction as in task 7.3 (≥ 5 behavioral patterns): 98 %.

These are the measurable goals from Table 3 of the project proposal.

- Deliver and collect tools and part up to 250 g that can easily and safely be grasped by workers.
- Detect and handle obstacles down to 0.02 m, such as cables or ropes.
- Uninterrupted support of workers activities: 20 min (exceeding a single ACW fly-time).

2 Software architecture description

Figure 4 shows the software architecture in WP7 from a task planning perspective, including the modules involved and their interfaces. High-level decision-making is carried out by two modules: the *High-Level Planner* and the *Agent Behavior Manager*.

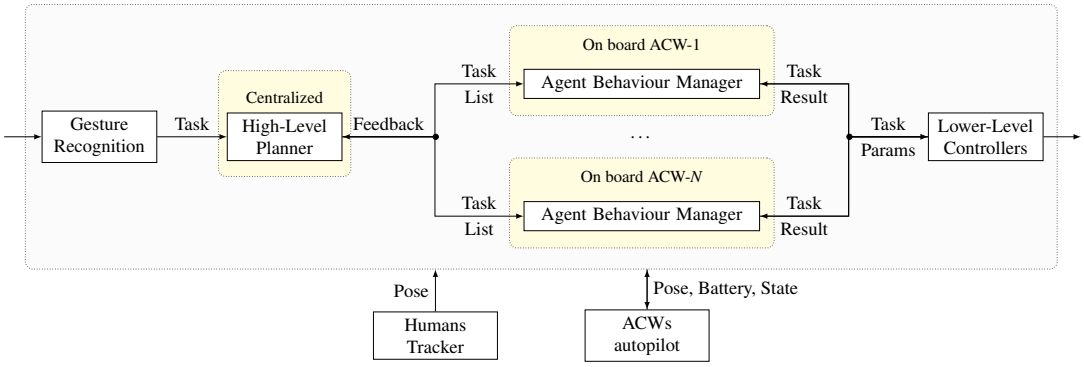

Figure 4: WP7 Software architecture: Modules and interfaces.

The *High-level Planner* is the centralized module of the software architecture and runs on a ground station. This module is in charge of task planning, i.e., it decides which tasks will be assigned to each ACW. The system knows the available ACWs thanks to the feedback provided by the Terabee RTPS. At this stage, battery constraints and ACW positions and capabilities are taken into consideration. Such information is supplied at a constant rate (to be defined). The number and type of available ACWs can be specified before the mission starts via a configuration file.

The *Agent Behavior Manager* is a distributed module running on board each ACW. This module mainly runs a state machine implemented as a Behavior Tree (BT) which governs the ACW to perform each of the assigned tasks. Each BT monitors the ACW's battery and task status and reacts to any possible failure or unexpected event, requesting a new re-planning to the High-Level Planner in case of need. In addition, the *ACW autopilot* and *Human Tracker* nodes supply for the ACW state and human worker pose, respectively. This information is used for monitoring and tool delivery tasks.

Each Agent Behavior Manager implements several interconnected BT. The *Main tree* is depicted in Fig. 5. This BT checks whether the mission is over (a mission would represent the working session, not a single task, i.e., whether the ACW is ready to be turned off) and otherwise, whether the ACW has any task to perform. If so, the battery level is checked and, depending on the result, either the corresponding task is executed (sub-tree represented in Fig. 6) or the ACW is guided to a recharging station[5]. The BT is also prepared to be safe against a loss of connection with the centralized module. Both unexpected events are managed flushing the task queue for the ACW to

[5]Both Safety, Inspection, and Physical-ACW provide an input interface to guide the ACW to the charging station. In other words, among the low-level controller capabilities, there is a "reach this point" feature. The location of the charging stations is known in advance or provided as input by the High-Level Planner/Behavior Tree.

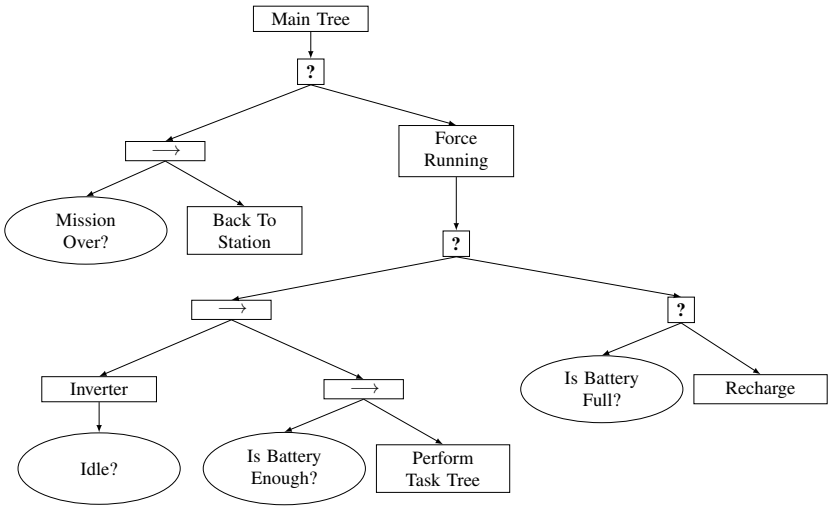

Figure 5: Behavior Tree: Main tree.

recharging, while giving the High-Level Planner control to decide when it is the best time to stop recharging (the High-Level Planner just needs to assign tasks again so that the ACW start working back).

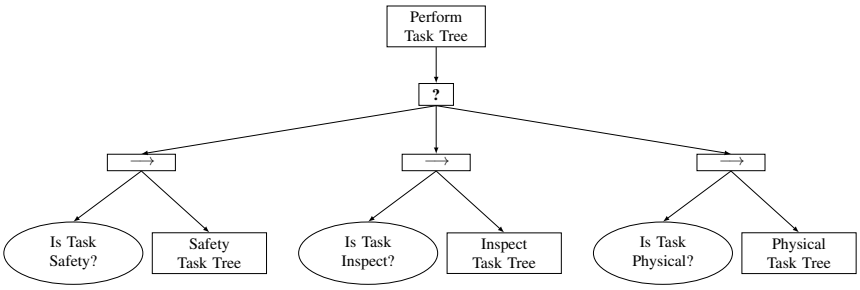

Figure 6: Behavior Tree: Perform Task Tree.

Figures 8, 7 and 9 represent the sub-trees that run Safety (see Sec. 1.2), Inspection (see Sec. 1.1), and Physical tasks (see Sec. 1.3), respectively. They all guide the ACW close to where the tasks needs to be performed (e.g., close to a worker to monitor or a place to inspect) and then, the corresponding Low-Level controller is called. These Low-Level controllers run on board the corresponding ACWs and must communicate their results (success or failure) asynchronously back to the Agent Behavior Manager, so that the Behaviour Tree could continue running.

The data interfaces for each module in the software architecture are listed in Table 1, while the data explanation is provided in Table 2.

Table 1: Description of the data interfaces for each software module.

Module Name	Input Data	Output Data
Gesture Recognition	Images	**Task, defined by:** Task ID, Task Type, Monitoring Distance, Monitoring Number, WP List, Tool ID (some task parameters will be ignored depending on Task Type)
High-Level Planner	Task, Feedback (Agent Beacon, BatteryEnough, BT info), Human Pose, ACWs Pose, Battery and State	Task List adding to each one its extra parameters result of the planning (Formation and/or List of ACW's IDs)
Agent Behaviour Manager	Task List, Result (Low-Level Result), Human Pose, ACWs Pose, Battery and State	Params needed by Low-Level Controllers (depending on Task Type)
Low-Level Controllers	Params (depending on Task Type)	Result
Humans Tracker		Pose
ACW autopilot	Low-Level orders	Pose, Battery and State

2.1 Tasks explanation

Once a task has been delivered to the High-Level Planner, in order to change any of its parameters, the Gesture Recognition node should send again the task, keeping the same Task ID and updating only the corresponding parameters. Thus, the High-Level Planner would overwrite it and allocate it again with the new parameters. In the following, a brief description on how to carry out each of the available tasks in the system. It is assumed that there are Low-Level Controllers running on the ACWs to perform basic navigation actions, formation control for human worker monitoring, inspection operations, and physical interaction with the human worker (e.g., to pick or deliver a tool). These Low-Level Controllers operate in a known environment, represented by a map (i.e., an occupancy grid map) that also includes the position of obstacles and the power tower.

2.1.1 Inspection-ACWs

High-Level Planner inputs: Task ID, Task Type and Waypoint List (the rest will be ignored).

Description: the High-Level Planner, from the list of waypoints, will decide how many ACWs are required and which part of the waypoint list is assigned to each one. Then, it would send to each corresponding Agent Behavior Manager the task parameters, including a list with the IDs of the selected ACWs. The same information is forwarded to the Low-Level Controllers when the BT calls them (see Fig. 7). Basically, the list of waypoints to be inspected are covered by the assigned ACWs, stopping at each of them to take images.

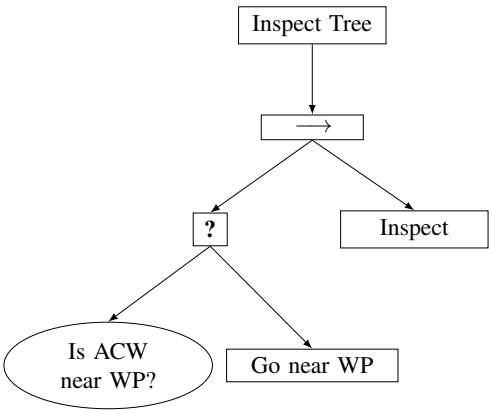

Figure 7: Behavior Tree: sub-tree that controls the inspect tasks.

2.1.2 Safety-ACWs

High-Level Planner inputs: Task ID, Task Type, Human Target ID, Monitoring Distance, and Monitoring Number (the rest will be ignored).

Description: the High-Level Planner will assign this task to as many Safety-ACWs as specified Monitoring Number. The formation will be chosen by the High-Level Planner from a set of fixed formations (to be listed) depending on the number of ACWs. Each selected ACW will know a list with the IDs of the ACWs selected for the task and the formation that they must take. As shown in Fig. 8, each Agent Behaviour Manager would individually navigate each ACW near the human target and then, it would call the corresponding Low-Level Controller for formation control. Extra ACWs could even be added to the formation at any time, just updating the task parameters sending a new task from Gesture Recognition.

2.1.3 Physical-ACWs

High-Level Planner inputs: Task ID, Task Type, Human Target ID and Tool ID (the rest will be ignored).

Description: After task allocation, the High-Level Planner will send the information to the corresponding Agent Behavior Manager and from there, the Low-Level Controllers will be called sequentially as shown in Fig. 9. Basically, the ACW needs to

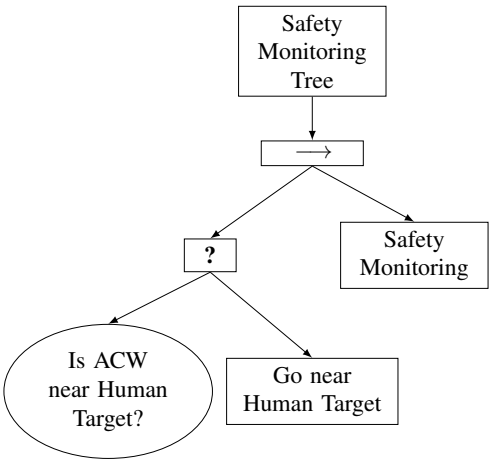

Figure 8: Behavior Tree: sub-tree that controls the safety monitoring tasks.

navigate to the station where the tool is, pick it up, navigate back to where the worker is, and start physical interaction to deliver the tool.

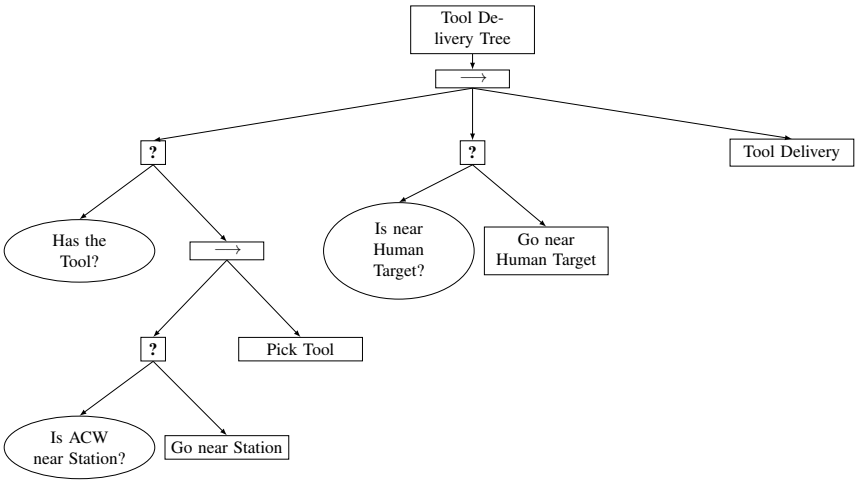

Figure 9: Behavior Tree: sub-tree that controls the tool delivery tasks.

3 Hardware integration

This section will describe briefly the target hardware (i.e., the vehicle we are going to use) where the software architecture will have to run.

4 Q&A

- **Q1:** *The Human Tracker detector, will be running on board each ACW? Will it be based on images or also on Terabee's sensors?*
- **Q2:** *Will ROS be used for software integration? Which version? Which simulation platform? Are MAVROS messages used to exchange data with drones?*
- **Q3:** *Safety-ACW. It would be nice to have a list of possible drone configurations (e.g., circle, triangle, etc.) that the operator can ask to the fleet of quad-rotors. Are the formation camera viewing angles (i.e., bearing angle w.r.t. the human worker) are input for the system? Is the formation scale an input for the task?*
- **Q4:** *Both for the Safety and Inspection-ACWs, it would be useful to compare the hardware equipment used for the experiments (see Secs. 1.1 and 1.2) and that onboard the drones supplied by DON. CTU can provide the ROS and Gazebo setup.*
- **Q5:** *Inspection-ACWs. It would be nice to set up the number of drones that should perform the autonomous inspection of a power tower. If there are no project specifications to meet (see recent amendment), two/three drones would be fine for CTU. Moreover, how are the waypoints to be inspected divided among the available ACWs?*
- **Q6:** *Gestures. It would be nice to have a list of gestures with their respective meaning. These are inputs for the High-level decision-making system.*
- **Q7:** *Are we thinking of creating a Gazebo environment where we can test the different algorithms and emulate the scenario of the demonstrator?*
- **Q8:** *Questions for CTU from USE: does the High-Level Planner have to group/cluster the n-waypoints (e.g., using K-mean algorithm) by m-UAVs or does the algorithm (CTU Low-Level Planner) independently manage the assignment of waypoints to the Inspection-ACWs?*

References

[1] G. Silano, J. Bednar, T. Nascimento, J. Capitan, M. Saska, and A. Ollero, "A Multi-Layer Software Architecture for Aerial Cognitive Multi-Robot Systems in Power Line Inspection Tasks," in *2021 International Conference on Unmanned Aircraft Systems*, 2021, pp. 1624–1629.

[2] G. Silano, T. Baca, R. Penicka, D. Liuzza, and M. Saska, "Power Line Inspection Tasks With Multi-Aerial Robot Systems Via Signal Temporal Logic Specifications," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 4169–4176, 2021.

[3] V. Krátký, A. Alcántara, J. Capitán, P. Štěpán, M. Saska, and A. Ollero, "Autonomous Aerial Filming with Distributed Lighting by a Team of Unmanned Aerial Vehicles," *IEEE Robotics and Automation Letters*, 2021, To Appear.

Table 2: Description of data types.

Data name	Data type	Comment
Task ID	String	Unique identifier of each task
Task Type	Integer	Task type indicator: m/M, i/I or d/D
Human Target ID	String	Unique identifier of each human worker. The position of the human target and other needed info is supposed to be known and accessible via its ID.
Monitoring Distance	Float	Distance from which the ACW surveil the worker during a safety monitoring task
Monitoring Number	Integer	Number of ACWs that are required in formation for a certain safety monitoring task
WP List	List of 3 float tuples (x, y, and z)	List of waypoints to be inspected
List of ACW's IDs	List of Strings	List of the unique identifiers of the ACWs that have been selected for a task that requires multiple ACWs
Formation	Integer	Indicates which of the predefined types of formations should be used for monitoring (e.g., circle, triangle)
Tool ID	String	Unique identifier of the tool to be delivered
ACW Pose	geometry_msgs /PoseStamped	ACW Position and orientation
ACW Battery	sensors_msgs /BatteryState	Percentage of battery in the ACW
Battery Enough	Boolean	Result of computing if an ACW will have enough battery for its current task
Agent Beacon	String, Integer	ACW unique ID and integer defining the type of ACW (Safety, Inspect, or Physical Interaction). Used as heartbeat to detect new ACWs
BT info	String list	Status of each BT node: which are Running, which are IDLE, which just returned SUCCESS, and which just returned FAILURE
Lower-Level Result	Boolean	Result of the Low-Level Controllers after a call