Received June 9, 2021, accepted June 26, 2021, date of publication July 1, 2021, date of current version July 12, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3093978

A Generic ROS-Based Control Architecture for Pest Inspection and Treatment in Greenhouses Using a Mobile Manipulator

JON MARTIN[1], ANDER ANSUATEGI[1], IÑAKI MAURTUA[1], AITOR GUTIERREZ[1], DAVID OBREGÓN[2], OSKAR CASQUERO[3], AND MARGA MARCOS[3], (Senior Member, IEEE)

[1]Autonomous and Intelligent Systems Unit, Fundación Tekniker, 20600 Eibar, Spain
[2]Centro Tecnológico CTC, 39011 Santander, Spain
[3]Systems Engineering and Automatic Control Department, Faculty of Engineering, University of the Basque Country (UPV/EHU), 48940 Bilbao, Spain

Corresponding author: Jon Martin (jon.martin@tekniker.es)

This work was supported in part by the GreenPatrol European Project through the European GNSS Agency by the European Union's (EU) Horizon 2020 Research and Innovation Program under Grant 776324 [11].

ABSTRACT To meet the demands of a rising population greenhouses must face the challenge of producing more in a more efficient and sustainable way. Innovative mobile robotic solutions with flexible navigation and manipulation strategies can help monitor the field in real-time. Guided by Integrated Pest Management strategies, robots can perform early pest detection and selective treatment tasks autonomously. However, combining the different robotic skills is an error prone work that requires experience in many robotic fields, usually deriving on ad-hoc solutions that are not reusable in other contexts. This work presents *Robotframework*, a generic ROS-based architecture which can easily integrate different navigation, manipulation, perception, and high-decision modules leading to a faster and simplified development of new robotic applications. The architecture includes generic real-time data collection tools, diagnosis and error handling modules, and user-friendly interfaces. To demonstrate the benefits of combining and easily integrating different robotic skills using the architecture, two flexible manipulation strategies have been developed to enhance the pest detection in its early state and to perform targeted spraying in simulated and field commercial greenhouses. Besides, an additional use-case has been included to demonstrate the applicability of the architecture in other industrial contexts.

INDEX TERMS Precision agriculture, robotic control architecture, mobile manipulator, pest detection and treatment, greenhouse.

I. INTRODUCTION

The European agriculture land surface is decreasing due to deforestation and urbanization while population continues increasing. In order to achieve a more sustainable business model, protect the crops from adverse weather conditions and control the temperature and water of the plant, greenhouse production is growing a 22% accumulated increase in area since 2011 [1]. However, the presence of warm, humidity conditions and abundant food under protected structures provide favorable habitats for pest development, this being the main threat to production and productivity of greenhouse crops worldwide [2]. Digital farming [3]–[5] can help through sensors, robotics and data analysis to automatically maintain and monitor greenhouses, making cropping system smart and, thus, enhancing the agricultural productivity.

Traditional pest detection methods in tomato crops rely on farmers observing skills, which are very time-consuming and inefficient in large crops. Nowadays, robotic solutions combined with computer vision can be used to automate this repetitive inspection task, increasing the reliability, maximizing the health of crops and optimizing the use of pesticides to as little as 5%-10% [6]. For that purpose, robots need to implement plenty of different tasks such as localize

The associate editor coordinating the review of this manuscript and approving it for publication was Vincenzo Conti.

themselves [7] and navigate inside greenhouses [8], [9]; acquire quality pictures to identify pests and their locations [10]; or process the obtained results to generate efficient high-level instructions to command the robot according to an Integrated Pest Management (IPM) system [11]. However, most research works focus on individual problems neglecting its integration within a single complete solution. The combination of different robotic skills can be difficult and usually derive to ad-hoc solutions, but this is necessary to perform early pest detection. The insect in their early eggs state can measure as less as 0.3 mm and, to detect them, advance perception and dexterity skills need to be merged to automatically obtain close and good quality pictures of the pests from different sides of the leaves.

This work presents *Robotframework*, a novel robotic architecture that integrates navigation, manipulation, and perception skills while following high level instructions from an IPM decision support system for early pest detection and treatment in greenhouses. The architecture includes additional features that makes it easily applicable for similar precision agriculture applications where robot navigation, manipulation and perception skills are required. This generic architecture can remarkably reduce the development time required to perform Robot Operating System (ROS) based field robotic experiments due to efficient reuse of common modules across projects and robot platforms. To demonstrate the easy integration and the benefits of combining different robotic skills within the architecture, flexible manipulation strategies to enhance pest detection and targeted spraying have been developed. Finally to evaluate the architecture, several tests in simulated and field commercial greenhouses have been performed in the context of the European GreenPatrol project [13].

This paper is structured as follows: The related work is analyzed in Section 2. Section 3 introduces the challenges for performing autonomous pest detection and treatment and the main robotic system requirements. Section 4 describes the developed *Robotframework* architecture while Section 5 focuses on the manipulation strategies for enhancing pest detection and treatment. Section 6 introduces the simulated and field tests carried out to evaluate the system in greenhouse and industrial scenarios. Finally, the conclusions obtained from the assessment are discussed and the future work is presented.

II. RELATED WORK

Research on precision agriculture robotics has recently focused on two areas: (i) weed inspection and targeted spraying and (ii) fruit and vegetables harvesting robots [5]. The first area is mostly represented by outdoor robots for weed control such as the Graph Weeds Net [14], the RHEA project centered on both agriculture and forestry [15], BoniRob project dedicated to multipurpose farming [16], or CROPS project focused on precision spraying in vineyards [17]. The navigation of these outdoor robots is largely based on the use of satellite localization systems and their signal is much weaker and unprecise in indoor environments, making them less suitable for greenhouses [18]. Moreover, greenhouses are specially challenging for simultaneous localization and mapping solutions, as they are partially structured environments with constantly growing plants [19]. That is why most of the robots found in greenhouses use rails to navigate on it [20]. Some examples are the tomato harvesting robot [21], the pepper harvesting robot [22] or the cherry tomato harvesting robot [23]. Other examples are AURORA, a spraying robot that implements a wall following algorithm for navigation [24] or a greenhouse spraying robot that follows lines and QR codes for navigating [25]. The use of fixed paths such as rails for navigation in greenhouses has resulted in decoupling navigation from the manipulation and inspection tasks. This is the case of the CROPS robot framework [26], where the control architecture covers only the fruit localization and arm control functionalities. As demonstrated by the open source control architecture FroboMind [12], a common reusable architecture that combines different robotic skills and tailored to precision agriculture robots can significantly decrease development time and resources due to efficient reuse of existing work across projects. However, despite using ROS as communication middleware, BoniRob is outdated as it does not integrate the state-of-the-art accepted *navigation_stack* [27] for navigation or *MoveIt!* [28] for manipulation. The first package takes information from odometry, sensor streams, and a goal pose and outputs safe velocity commands that are sent to the mobile base. The second one is the most widely used software for manipulation and provides the latest advances in motion planning, manipulation, or 3D perception, among others. In order to combine both algorithms for mobile manipulators, some authors have tried to simultaneously plan and execute mobile manipulation goals [29], [30] but these are computationally complex methods tested in simulation or laboratory conditions and currently unfeasible for greenhouse-like unstructured environments.

Developing an effective IPM requires frequent and precise observations of plants. To build an early pest detection it may not be enough to focus on plants or leaves that are already infected with insects at adult stages as in [32]–[34], but it is necessary to detect the cause of the infection. In order to enhance the early pest detection, it is necessary to go a step further and detect the insects also in their egg and larva stages [10], [35]. Moreover, most pest detection works focus on the detection and classification of pests on already acquired pictures dataset neglecting the difficulties of automatically obtaining them with enough quality and closeness. In this sense, this work presents the manipulation strategies developed to get closer pictures to the surfaces of the leaves from above and from below, so as to inspect the surfaces of the leaves from both sides.

Finally, there are several European projects as DROPSA [36], ISEFOR [37], PALMPROTECT [38] or EMPHASIS [39] focused on the development of new fighting strategies against some specific pests, but the bridge between new pest

FIGURE 1. Tomato crop greenhouse evolution at the beginning (up) and at the end (down) of the season.

detection strategies and automated and robust management is barely addressed.

This work presents, similarly to [31], a decoupled mobile manipulation control for greenhouse related tasks using the ROS de-facto algorithms [27] and [28]. The navigation is based on latest robotic solutions which have proven to successfully use Galileo Satellites combined with IMU, odometry and range laser sensors for localization [8]. The control architecture follows the hybrid paradigm presented in [40], where rational and efficient deliberative decisions represented by an IPM strategy are combined with reactive behaviors represented by the different navigation, manipulation and vision modules.

III. PROBLEM DESCRIPTION AND ARCHITECTURE REQUIREMENTS

There are several challenges for developing a robotic system able to perform autonomous and continuous monitoring in greenhouses for the detection, identification, and control of pests. As shown in Figure 1, the plants grow remarkably during the growing season affecting: (1) the localization and navigation systems because of a constant change of the environment and the narrowing of the corridors; (2) the manipulation strategy, as the arm needs to approach the leaves to obtain good quality pictures while avoiding damaging the crops; and (3) the vision modules dealing with changes in illumination and focus distance. In addition, the system must be able to execute high-level instructions proposed by the IPM strategy, providing diagnosis and logging capabilities and offering an easy-to-use user interface.

The main robotic system requirements presented in Table 1 have been identified by observing a single robot needs for the GreenPatrol application. It is however noticeable that most requirements remarked in bold are desirable for almost any other mobile manipulator system. *Robotframework* takes all these requirements into account and presents an architecture that is not only valid for the current application but also for other agricultural or even industrial applications.

TABLE 1. Main mobile manipulator system requirements.

System Req.	Description
SR1	The robot can **localize** itself within the greenhouse **with or without** the help of **markers** and can **autonomously navigate**.
SR2	The system must provide **precise manipulation capabilities** to support dexterity tasks.
SR3	The system must provide **perception capabilities** to perceive its working environment, monitor interesting features or perform inspection tasks.
SR4	The robot can **execute parametrized high-level orders**.
SR5	The system must **detect and notify critical errors and warnings** triggering recovery behaviors or, ultimately, the complete application interruption.
SR6	The system must **log historical events** such as the route (positions) of the robot, the actions occurred there and unexpected incidents.
SR7	The system must provide an **easy-to-use GUI** to start the application, monitor its progress and present the general system status.

IV. SYSTEM ARCHITECTURE

This Section contains a description of the general control architecture presented in Figure 2. The four-layered architecture seeks the easy integration of the different robot functionalities ensuring the system requirements presented in Table 1. It follows a distributed computing design allowing several tasks to run in different computers while still appearing to its users as a single coherent system and allowing an easy extensibility.

ROS [41] is proposed as core communication middleware among the different modules. In recent years, ROS has become the de facto standard framework for the development of software in robotics. ROS is a flexible open-source framework for writing robot software that provides, collection of communication mechanisms, tools, libraries, and rules that aim to simplify the task of creating robot software for a wide variety of robotic platforms.

In the architecture, there are several modules that are common to any application. These are represented in turquoise color and include the robot user interface, as well as application layer, the error managing and logging modules and some common parts of the abilities layer. Some other modules composed by standard ROS modules or packages developed and tested by the GreenPatrol project are available in the architecture, but it is up to the user to use them or implement new modules using the available ones as templates. The drivers layer, for instance, depends on the robot used. Also, the high-level decision modules, here represented as an IPM system, depends on the application.

A. DECISION LAYER

The decision layer contains the high-level decision modules that generate new plans for the application. In this case,

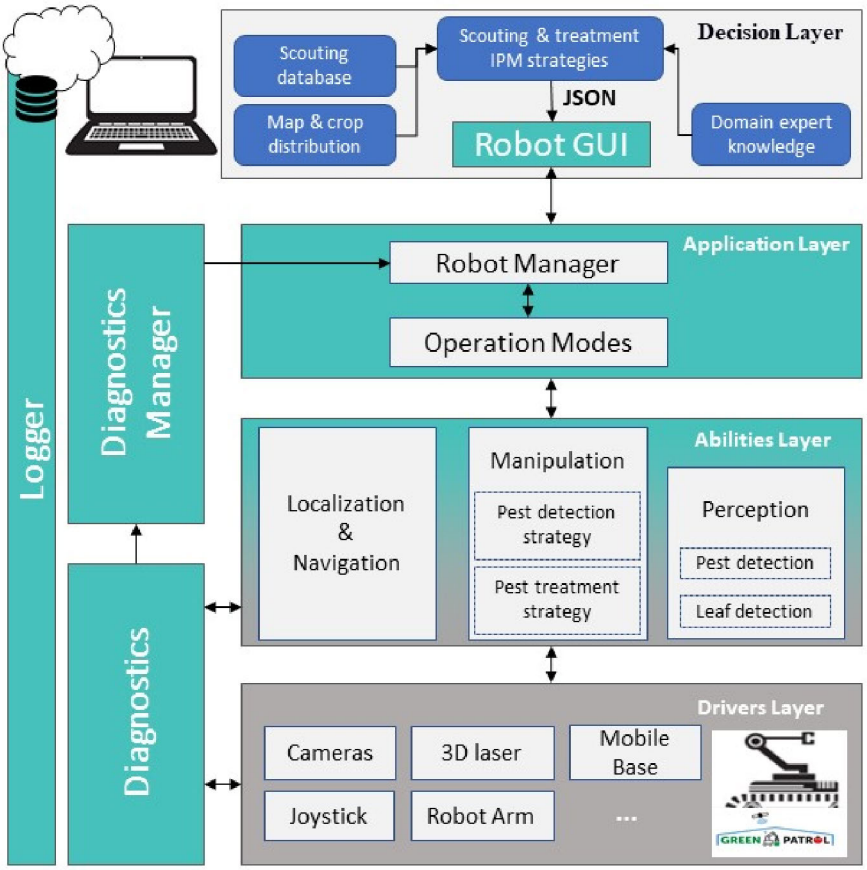


FIGURE 2. Four-layer Robotframework control architecture. Modules represented with turquoise color are common to any application. Grey modules are standard ROS modules while the rest have been developed and tested during the GreenPatrol project and could be used as templates.

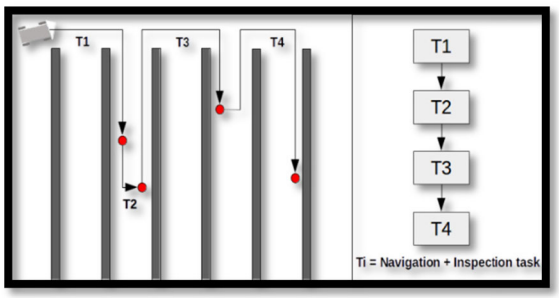


FIGURE 3. Greenhouse representation with a robot approaching the red circles representing the targets with navigation and pest inspection tasks (left). Representation of an IPM generated T1-T4 plan (right).

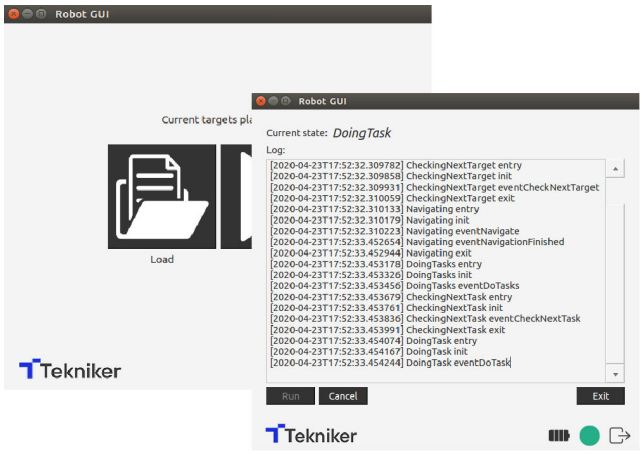


FIGURE 4. Robot GUI interface used to load and start a json plan (left) and the application workflow messages once the plan has been initialized (right).

an IPM strategy generates pest scouting and treatment plans based on domain expert knowledge, crops distribution in the greenhouse and information obtained from previous plan executions as seen in the top layer of Figure 2. The plans are composed by targets that contain a navigation goal to move the robot to a desired position and a task to be performed there. An example plan for the current application can be seen in Figure 3 where the robot must navigate to four different greenhouse zones and perform there an inspection task.

The **Robot GUI** module in Figure 2 is common for any application and provides an easy-to-use user interface to load, execute or cancel plans fulfilling system requirement SR7. The plan is then sent to the application layer to be interpreted and executed by the robot while the GUI displays the current state of the system including status, alerts, or the batteries level as shown in Figure 4.

The plans are implemented using a JavaScript Object Notation (JSON) format, which is a very common open standard

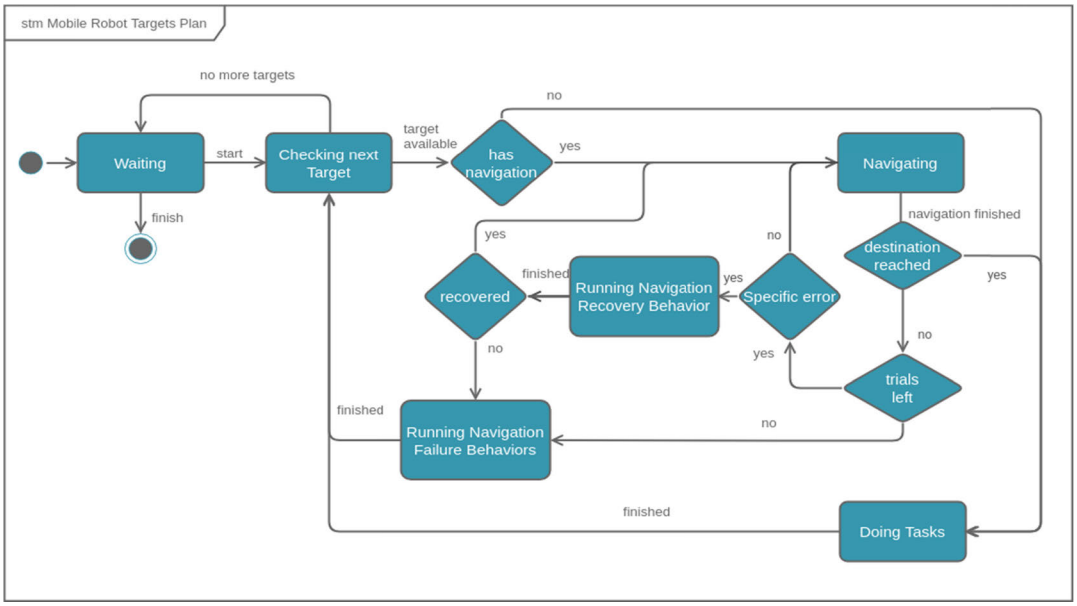


FIGURE 5. State machine representing the operation mode targets plan and its navigation, tasks, and error-handling behavior states.

and language-independent, that uses human-readable text to store and transmit data objects consisting of attribute–value pairs and array data.

The JSON keywords related to navigation are:

- **navigate**: Indicates whether the target contains a navigation step. The following navigation keys are only considered if navigate is true.
- **navigationType**: Three types of robot navigation are available. (1) *Natural navigation* making use of the well-known *navigation_stack* from ROS; (2) *Relative navigation* to perform a continuous motion to reach a position relative to the robot's current position; and (3) *Precise navigation* to generate a continuous motion to position the robot accurately with respect to an artificial mark. Only the first type is used in the GreenPatrol context.
- **navigationTrials**: Number of trials for navigation in case of failure.
- **targetPose**: Navigation destination (x, y, theta) in the given *frame_id* sent to the navigation node.

The JSON keywords related to task definition:

- **tasks**: An array of tasks to be executed.
  - **name**: Name of the task.
  - **type**: Type of the task plugin that will be loaded and executed. In this case it may be an inspection or a spraying task.
  - **params**: Necessary parameters to perform the task. This enables a high configurability of the IPM strategy to define the zones to be inspected or the amount of pesticide to use depending on the infection level of the plant.

This method permits orders parametrization that covers a wide range of mobile robotics applications. The plans can be generated manually or automatically by different high-level decision support systems, addressing system requirement SR4. Section 5 presents two simple plan examples for pest inspection and treatment operations and subsection 6-C presents an additional plan for an aileron inspection in an industrial use case.

B. APPLICATION LAYER

The application layer includes the **robot manager module**, which is responsible for controlling the overall robotic system, maintaining its status continuously. This layer also includes the **operation mode**, which interprets the high-level plans and implements a specific robotic application. The *operation mode* is designed to be as general as possible, easily configurable for a variety of robotic processes and, thus, avoiding an ad-hoc implementation only useful for specific workspace configurations. A plan can be specified by a set of targets composed by a *navigation destination* for the mobile platform and a set of *tasks* to be performed at each destination. Figure 5 shows the state machine implemented for the operation mode. It starts in a *Waiting* state until a new-plan event indicates the beginning of the operation. The system switches then to *Checking next target* state, analyzing the next target in the sequence.

The *Navigating* state is responsible for coordinating the autonomous movements of the mobile platform. The architecture permits an easy integration of different navigation modules and provides the management of their results. If the navigation is not able to reach the target pose due to obstacles on the way or localization problems, this will be notified in the result and, if required, a *Navigation Recovery Behavior* is triggered. Due to the criticality of the satellite-based localization in greenhouses to reach a position accurately, it has been

necessary to include an additional feature in the navigation stack to provide information about the localization quality. In case of a remarkable localization quality loss, a specific error recovery behavior can be triggered. This behavior consists in sending the robot to a well-known greenhouse position where the localization signal is known to be strong and retrying from there the previous navigation goal. There is a second recovery behavior triggered when, despite having a proper localization signal, the robot does not reach the destination with enough accuracy. This can happen because a slightly better localization is needed or because there are obstacles on the way that the navigation module cannot overcome. In both cases, the recovery behavior consists off waiting for a predefined time still, while playing an advertisement sound. Waiting may help improving the localization while the sound notifies the operators in the vicinity about the current robot state and, if needed, about the need of removing the obstacle on the way.

The number of trials to perform the navigation are configurable. If there are no more trials left, meaning the robot failed to reach the destination, the failure is notified in the *running navigation failure behavior* state, and the tasks to be performed at this point are skipped, addressing the following target. The navigation state has been developed in a generic way to support different global, relative, and precise navigation modules as it will be explained later.

Once the navigation finishes correctly, the configured set of tasks are executed in the *Doing Tasks* state. A *Task* is the implementation of a robots' specific set of actions. The proposed architecture is designed to implement new robotic tasks by using the ROS *pluginlib* mechanism. The tasks are developed as plugins which are parametrized, dynamically loadable and executed from a runtime library. The plan generated by the high-level decision module must contain enough information for the state machine to understand where to go and which actions to take at each place. There is therefore no need to touch or recompile the core of the framework. This is useful for extending/modifying the application and provide a great extendibility to the system. Section V present two task implementations in the context of precision agriculture for pest detection and treatment while Section VI-C illustrates an additional task example for an aileron inspection in an industrial context. The benefits and reusability of the architecture is finally described in the discussions section. Moreover, the here presented task plugins can be used as templates and be adapted for future tasks.

C. ABILITIES LAYER

This layer is composed by the ROS nodes involved in the basic control functionalities of a robot. These nodes manage the sensor and actuator components, and provide robot capabilities such as autonomous navigation, manipulation, and inspection. At this level, ROS provides a wide range of state-of-the-art robotic algorithms: *GMaping* [42] for generating maps using the on board 2D laser scanners. The maps can be manually modified to include, for instance, forbidden areas

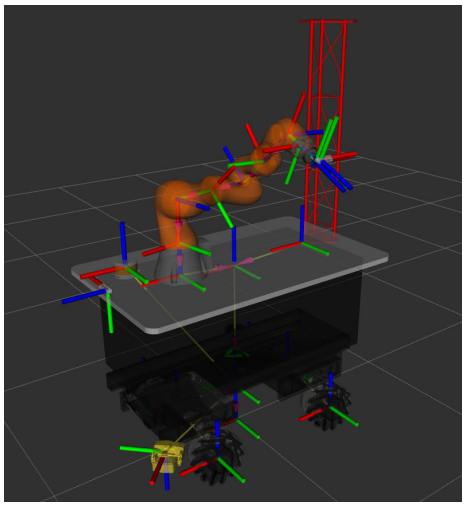


FIGURE 6. GreenPatrol robot description in ROS-visualization RViz. It presents the main platform components (mobile platform, arm, sensors…) and the transformation links between them.

for the robot; the *Navigation Stack* used in the *Navigation State* for planning global and local paths. It uses combined 2D laser scanners and satellite based localization to generate the velocity commands for the mobile base while avoiding the obstacles on the unstructured greenhouse environment; the Unified Robot Description URDF for generating a combined robot description as presented in Figure 6; *MoveIt!* is used for generating and executing collision free manipulation trajectories in the *Doing Tasks State* as it will be later presented in Section V. *MoveIt!* tools can be also used, to integrate 3D point cloud based obstacles or useful simulation tools among other utilities.

On top of them, several additional nodes have been developed. To ensure the system requirement SR1 and freely navigate within the greenhouse, the localization module benefits from the multiple signal frequencies and the higher accuracy provided by the European Global Navigation Satellite System (EGNSS) of the Galileo constellation as explained in [8]. The system requirement SR3 is achieved using a deep learning model for detecting the most harmful pests in greenhouse tomato crops: Bemisia Tabaci, Tuta Absoluta and Whitefly [10]. In addition, a leaf detection deep learning model has been implemented to safely and accurately detect and approach individual leaves using a 3D camera. Then, closer and high resolution pictures of the pests are taken as presented in Section 5, answering to system requirement SR2. The architecture includes additional modules for relative and precise navigation which are valuable for a wide range of mobile robotic applications as shown in subsection 6-C.

D. DRIVERS LAYER

The drivers layer includes the modules that allow interacting with the robot platform sensors and actuators. An overview of the specific robotic system used for validation purposes is shown in Figure 7.

FIGURE 7. GreenPatrol robotic platform entering the greenhouse.

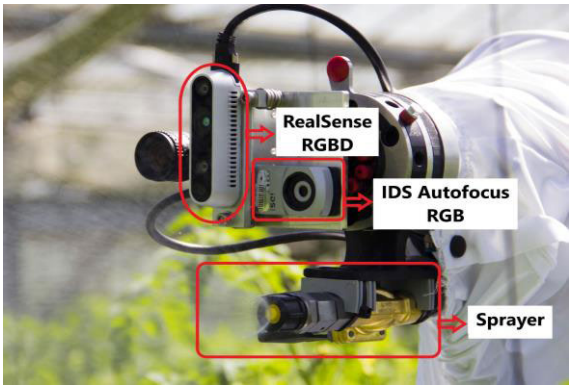


FIGURE 8. GreenPatrol pest inspection and treatment tools mounted at the robot arms end-effector.

The mobile platform consists on the Segway RMP 440 Omni Flex [43] with mecanum wheels to improve mobility in greenhouse narrow corridors. The platform is equipped with an on-board PC, a Velodyne 3D laser scanner [44] for obstacle detection and two OS32C safety laser scanners [45] for obstacle detection, mapping and navigation. The absolute localization unit consist of a multi-constellation, GNSS receiver, IMU and odometry. A KUKA LBR iiwa manipulator [46] has been mounted on the middle of the platform to allow inspecting the leaves on the right and left sides. The vision system consists of a 3D RealSense camera [47] to find leaves positions and an IDS RGB autofocus camera [48] to acquire good quality pictures of the pests as seen in Figure 8.

The spraying equipment consists of a plastic tank on the back-right corner of the platform, an electric compressor with a pipe and a spraying nozzle at the arm's end-effector shown in Figure 8.

A benefit of using ROS is the availability of a wide variety of robotic components drivers such as mobile robots, manipulators, cameras and, lasers. This makes the architecture hardware agnostic enabling the possibility to replace them without affecting the rest of the architecture.

E. MONITORING

The three modules shown on the left side of Figure 2 are available with the architecture to monitor the functional state of the system. The **Diagnostics** module has been designed for collecting and preprocessing specific data from drivers and abilities layers which are then passed to the **Diagnostics Manager** for automatic decision making and incidents notification fulfilling system requirement SR5. These two modules must be adapted to the application on demand. Moreover, the generated DEBUG, INFO, WARNING and ERROR messages are recorded by the **Logging** module using a RabbitMQ [49] queue that implements the Advanced Message Queuing Protocol (AMQP). The logs are used to record historical track of the process, ensuring SR6, and can either be stored locally in the robot or in the cloud using a non-relational Elasticsearch database [50]. This data has been later used to obtain tests results and statistics.

The following Section shows how to include new ad-hoc modules and tasks within the architecture. In particular, the integration of manipulation strategies for enhancing pest detection and treatment operations are presented.

V. MANIPULATION STRATEGIES FOR PEST DETECTION AND TREATMENT

The high-level decision support system (in this case the IPM strategy) defines the manipulation, inspection, and treatment tasks to be performed. First, the mobile platform needs to navigate to the target plants as seen in Section 4-A. Once in front of the plant, the robotic arm mounted on the middle-top of the mobile platform performs the corresponding pest inspection or treatment task on right and left sides of the platform. This Section presents the strategies taken, the execution workflow and examples of simple plans for each manipulation task. The tasks have been developed as *plugins* and represent standalone integration cases withing the architecture presented here.

A. PEST INSPECTION TASK

The plant zones to be inspected and the number of pictures that need to be taken at each zone are represented as the Pest Monitoring Index (PMI) in Table 2 and Figure 9. Lower and darker zones tend to provide more suitable habitats for the pests, resulting on a higher number of pictures required. As an example, in the high-up zone the robot must inspect leaves above 1m (PMI6) and requires two pictures to be taken, while in the middle-bottom zone the robot must inspect leaves from bellow in between 0.5 m and 1 m (PMI2) and requires 4 pictures to be taken. A simple inspection plan is shown in Figure 11.

TABLE 2. Pest monitoring index for defining the number of pictures to take at each plant zone.

PMI	1	2	3	4	5	6
N° of Photos	5	4	3	3	3	2

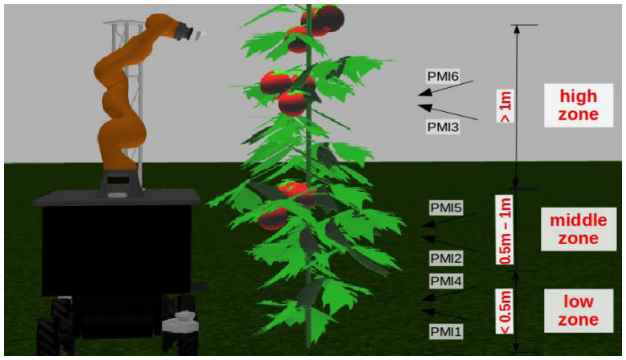


FIGURE 9. The GreenPatrol robot appears facing the plant at the high-up zone inspection position in Gazebo simulation.

The manipulation strategy for pest detection consists of the workflow defined in Figure 10 (up). First, the arm is moved to the next inspection zone. Second, the leaf detector model and the RGBD image are used to find leaves poses. If no leaf is found, the arm is moved to the following inspection zone. Third, the arm approaches the leaves found in the previous step and takes closer pictures of them using the RGB autofocus camera. An algorithm determines the quality of the picture. If the quality is not good enough, the arm makes a predefined small movement, and takes a new picture from there. This process is repeated until all required plant zones have been inspected.

The pictures taken in this process are saved locally on the robot. After completing the plan, the pictures are sent to the cloud, where a Deep Learning (DL) model has been deployed to identify infection areas in the greenhouse offline. The IPM strategy module uses the DL module results along with additional information such as the current harvest season conditions, the working area size, the size of the plant or legal aspects on pesticides on the working country. As a result, new inspection (Figure 11) and treatment (Figure 12) plans are generated.

```
[
 {
  "navigate": true,
  "navigationType": 1,                    # GLOBAL NAVIGATION
  "navigationTrials": 3,
  "targetPose": {
   "frameId": "map",
   "x": 12.0,
   "y": 3.0,
   "theta": 1.5708
  },
  "tasks": [                              # INSPECTION TASK
   {
    "name": "inspection",
    "type":"greenpatrol_inspection_task::GreenpatrolInspectionTaskPlugin",
    "params":"high_up;high_bottom; middle_up;middle_bottom;low_up;low
_bottom; plant_height:1.5;low_side:right;id_inspection:1;
image_name:-1; date:yyyy-mm-dd"
   }
  ]
 }
]
```

FIGURE 11. Example of a simple GreenPatrol inspection plan.

```
[
 {
  "navigate": true,
  "navigationType": 1,                    # GLOBAL NAVIGATION
  "navigationTrials": 3,
  "targetPose": {
   "frameId": "map",
   "x": 12.0,
   "y": 3.0,
   "theta": 1.5708
  },
  "tasks": [                              # SPRAYING TASK
   {
    "name": " spraying ",
    "type":" greenpatrol_spraying_task::GreenpatrolSprayingTaskPlugin",
    "params":"high_up;high_bottom; middle_up;middle_bottom;low_up;lo
w_bottom; plant_height:1.5;low_side:right;id_inspection: -1; dosage:0.5"
   }
  ]
 }
]
```

FIGURE 12. Example of a simple GreenPatrol spraying plan.

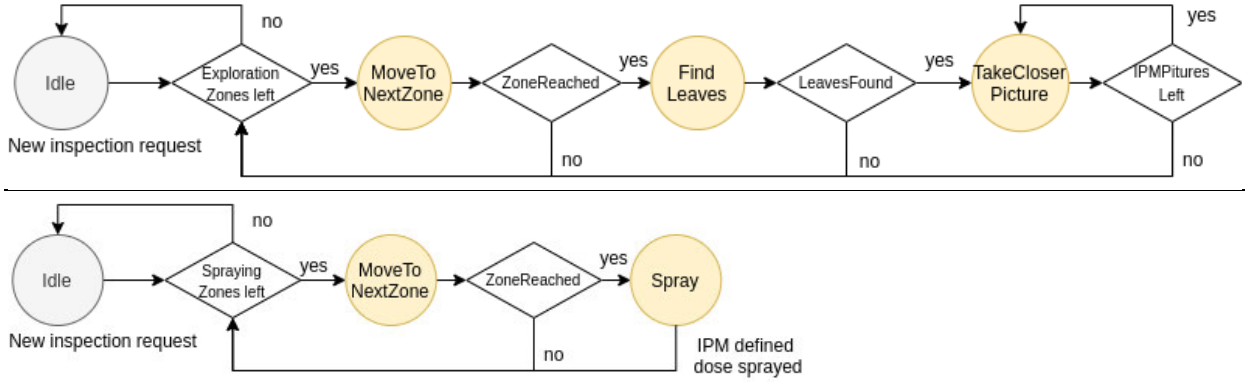


FIGURE 10. Manipulation strategies workflows for pest detection task (up) and pest treatment task (down).

B. PEST TREATMENT TASK

The pest treatment process can be defined as the precise spraying of pesticide on different plant zones (high, middle and low), being the pesticide spraying dose at each plant determined by IPM strategy as shown in the parameters field in Figure 12.

The manipulation strategy is represented by the workflow defined in Figure 10 (down). First, the arm is moved to the next spraying zone. Second, the sprayer is activated and in order to cover the whole plant zone, the manipulator performs small, controlled movements until the complete dose has been sprayed. This process is repeated until all required plant zones have been sprayed.

VI. SYSTEM VALIDATION

This section presents the validation tests performed within the simulated and real greenhouses of 52 × 30m and 31 corridors between plants shown in Figure 13 and Figure 1 respectively. The aim of these tests has been the assessment of the following features: first, the correct integration of *Robotframework* with the different robotic modules; second, successful execution of pest inspection and treatment plans; third, the logging capabilities of the system to generate and use the collected data; finally, the system requirements proposed in Table 1.

FIGURE 13. Details of the simulated environment in Gazebo simulator with the robot performing navigation and inspection tasks.

The results and the most remarkable conclusions are detailed at the end of each test. Furthermore, the use of Robotframework in an industrial application is presented to demonstrate its adaptability and generalization.

A. GREENHOUSE SIMULATION TESTS

Gazebo simulator [51] has been used to simulate the crops, robot sensory information (laser, images…), physics involved (collisions, inertia…) and localization data (global coordinates, errors…). The leaves, despite realistic, do not perfectly represent the real world and do not contain insects on them. Thus, a simulated vision module for leaf detection provides their position. Also, the images used for validating the pest detection and identification modules are semi-randomly acquired from our custom dataset of labelled images (a set not used for training the model) with infected and healthy images. The same software as in the real scenario has been used.

The simulation test presented in Figure 14 consists of the following steps: First, an IPM algorithm generates a new *semi random scouting plan* based on the greenhouse dimensions, which in this case corresponds to 31 rows (R1-R31) in the horizontal axis and 6 vertical zones that, in turn, consist of

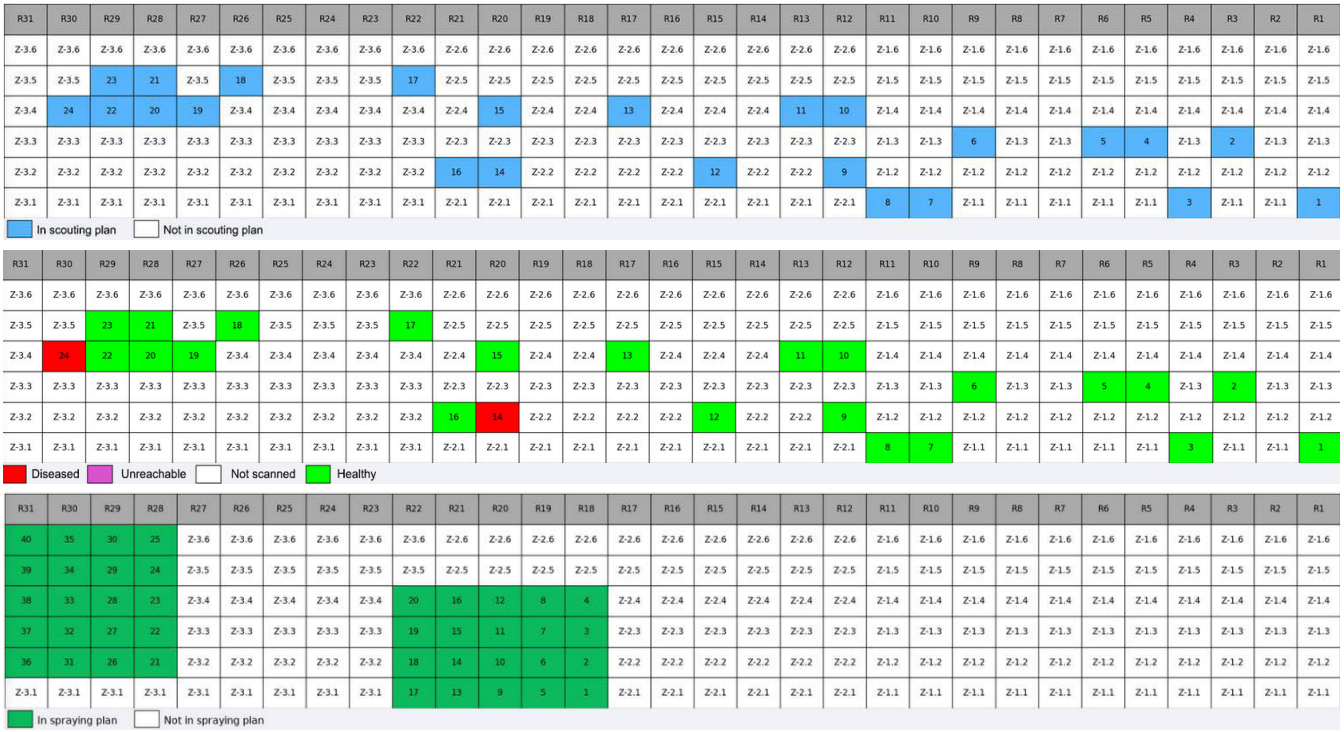


FIGURE 14. Representation of the greenhouse inspected/sprayed zones during the different simulation test-steps: Initial *semirandom scouting plan* (up). *Scouting results* representing the infected and healthy zones (middle). *Spraying plan* generated for infected zones (down).

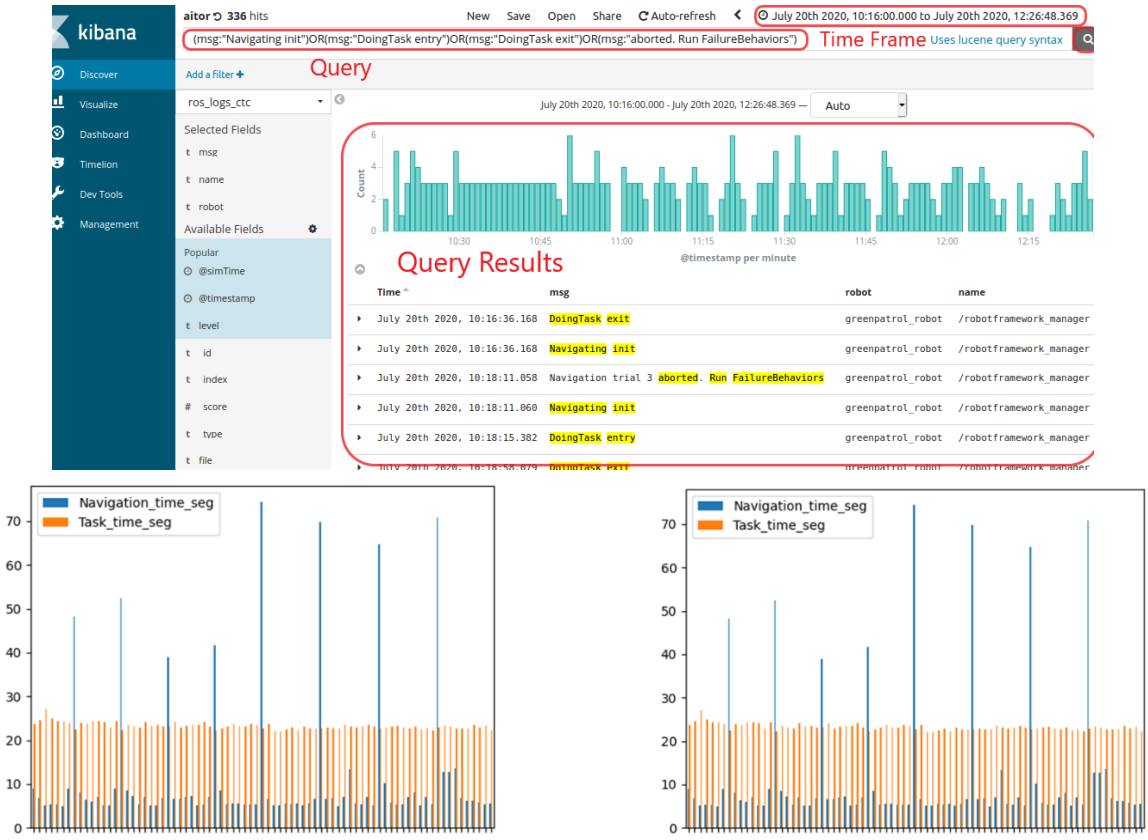


FIGURE 15. Log query results visualized in the Kibana user interface for the semirandom pest inspection test (up). The lower graphs represent timing metrics obtained from navigation and inspection tasks during the scouting (down-left).

a bunch of plants. This plan presented in Figure 14 (up), consists of 120 (navigation + inspection) targets to be performed in the zones marked in blue. The plan is executed, the log messages are generated and sent to the cloud through the logger module. The results obtained from the data analyzed is presented in Figure 14 (middle) where the green color represents the healthy zones and the red color the infected ones. From these results, the IPM algorithm can generate a new *spraying plan* with 80 targets as seen in Figure 14 (down). The plan is executed, the results are logged, and the results are analyzed again.

The log data is saved in non-relational databases and can be accessed through elastic-search queries as seen in the Kibana user interface presented in Figure 15 up marked in red. First, the time frame at which the test was performed must be defined. To filter the logs and search for specific information, queries can be done to the database.

On the one hand, the graphs in Figure 15 present timing metrics obtained from the inspection (down-left) and spraying (down-right) plans execution. The observed navigation time peaks occur when moving from one row to another. Smaller navigation times reflect the movements to plants nearby. We could for example capture the exact moment at which the robot failed to reach a destination after aborting the maximum number of trials (in this case three trials as shown in Figure 15 up). When a navigation fails, the following task is not performed, resulting on 119 successful inspection and a single task skipped during the scouting plan execution. Also, minimum, maximum, or average times for the different navigation, inspection or, spraying tasks can be easily obtained to analyze the system performance. The total simulation task time consists of 1h 40min of the semi-random scouting plan execution, 2h 24min of image analysis, and 47min of spraying plan execution, resulting on 4h 51min test.

On the other hand, 1547 pictures were acquired and processed during the test, resulting on a 98.25% of the greenhouse being pest-free (green zones) and a 1.75% being infected (red zones). Being the leaves positioning detection and image acquisition modules simulated, the arm movements are controlled resulting on all inspection tasks success. Similarly, the spraying manipulation strategy is based on prefixed arm movements and these movements are therefore always successful. The semi-randomly acquired pictures could be used to partially validate the pest detection and identification module allowing the IPM module to generate new spraying and treatment plans based on these results.

Finally, it is interesting to remark that logs are available as long as the databases are maintained allowing to analyze information ex post. This enhances the traceability capabilities by for example allowing to include new required queries not overseen before.

B. GREENHOUSE FIELD TESTS

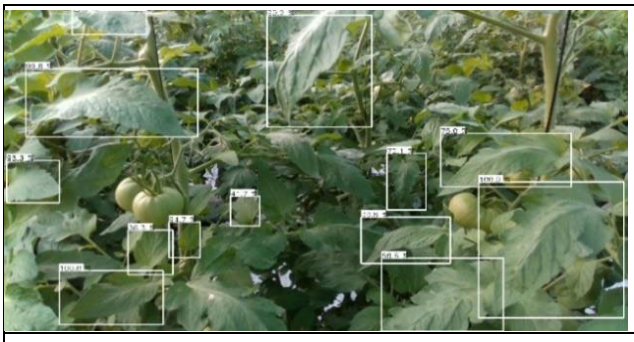
The objective of the field tests is to ensure the integration of the robot manipulation and perception modules with the architecture in real operational conditions. The robot shown in Figure 7 replaces the simulated one and the leaves detection and pest inspection DL models are included. The rest of the software remains the same as in the simulation tests. To test the integration of robot manipulation, inspection, and spraying functionalities two different tests-sets have been performed:

The first tests-set consists of executing simple scouting plans for inspecting the high zone of plants on the right and on the left sides of the robot. As explained in Section 5-A, the robot navigates to a plant, finds tomato leaves poses and approaches them to take good-quality, closer pictures. According to the Pest Monitoring Index in Table 2, the robot should take 5 pictures in total, 2 from above the leaves (PMI6) and 3 from bellow them (PMI3). The robot is teleoperated to the next plant and the process is repeated 28 times resulting in the acquisition of 140 pictures.

The log data is again and used to obtain different metrics similarly as in the previous test. The total valid number of pictures acquired depends first on the number of leaves found during the *FindLeavesStep*. Most of the times plenty of leaves are found as seen in the first row of Table 3.

However, bad illumination or closeness to leaves may cause that not enough leaves are found as seen in Table 3 row 2. In that case, the subsequent closer approach to leaves cannot take place resulting on 38 pictures missed during this step. Among the closer acquired pictures, 27 had not enough quality and where not valid for the pest detection and identification model. This quality is measured through different parameters such as blur, noise, and distortions of different intensities. Despite most of the analyzed pictures were healthy images (64/75), it was possible to detect and identify some Whitefly insects and Tuta Absoluta damaged areas on

TABLE 3. Image acquisition results acquired during the greenhouse field inspection tests.




Successfully found leaves.


Illumination changes reduce the number of detected leaves (left). Closeness to leaves reduce the field of view and number of detected leaves (right).


Successful White Fly (left) and Tuta Absoluta damaged areas (right) detection and identifications.

tomato leaves as seen in Table 3 third row. A summary with the number of pictures desired, missed and finally acquired is presented in Table 4.

TABLE 4. Acquired number of images and causes of missed pictures.

Desired number of pictures	140
Missed pictures during *Find Leaves* step	38
Missed pictures during the *Take Closer Picture* step	27
Total acquired healthy pictures	64
Total acquired infected pictures	11

The second tests-set consists of executing 10 simple pest treatment plans in which the high zone of plants on the left and right sides of the robot have been sprayed as explained in Section 5-B. The manipulation movements during the spraying task are based on prefixed arm movements and have been always successful.

C. ARCHITECTURE GENERALIZATION AND ADAPTABILITY ASSESSMENT ON AN INDUSTRIAL USE CASE

The previous tests have shown how different tasks for pest detection and treatment can share navigation functionalities thanks to the architectures' modular design based on plugins. During the tests only the first navigation type available in the presented framework has been used being 10 cm accuracy sufficient for a greenhouse navigation solution. There are however other applications such as a precise drilling or inspection operations, in which global navigation must be supported by a more accurate precise navigation. For these cases, *Robotframework* supports the possibility to use relative and/or precise navigation to enhance the maneuverability and robot platform's positioning accuracy.

This is the case of the CRO-INSPECT European project [52], which provides flexible assistive robotic inspection for complex composite parts (e.g. aileron). In this context, the robot composed of the same Segway mobile platform and KUKA iiwa robot configuration shown in Figure 16, uses the *Robotframework* architecture to perform the workflow presented in Figure 17: First, the robot freely navigates through an industrial workspace to position in front of an aileron with an average accuracy of 20 cm using the *global navigation*.

Second, the robot improves its positions in front of the aileron with an accuracy under 1 cm using markers and the *precise navigation* module. Both navigation types have been exhaustively tested, comparing different algorithms' limitations and capabilities in [53]. Third, the robot performs an aileron *inspection task* making use of the iiwa's force sensors and an ultrasonic inspection tool. Within a fourth step, the robot uses the *relative navigation* module to move straight and parallel to the aileron position. Then, the steps 2 to 4 are repeated until the complete aileron has been inspected.

FIGURE 16. The robotic solution for advanced inspection of complex composite parts within the CRO-INSPECT project context.

```
[
 {
  "navigate": true,
  "navigationType": 1,                    # GLOBAL NAVIGATION
  "navigationTrials": 3,
  "targetPose": {
    "frameId": "map",
    "x": 12.5,
    "y": 25.46,
    "theta": 1.57
  }
 },
 {
  "navigate": true,
  "navigationType": 3,                    # PRECISE NAVIGATION
  "navigationTrials": 2,
  "poseOffset": {
   "x": 0.015,
   "y": -0.95,
   "theta": 3.102
   },
  "tasks": [                              # INSPECTION TASK
   {
    "name": "inspection",
    "type": "croinspect_inspection_task::CroinspectInspectionTaskPlugin",
   }
  ]
 },
 {
  "navigate": true,
  "navigationType": 2,                    # RELATIVE NAVIGATION
  "navigationTrials": 2,
  "targetPose": {
   "frameId": "map",
   "x": 1.0,
   "y": -0.2,
   "theta": 0.0 }
   }
  }
]
```

FIGURE 17. Example of a simple CROINSPECT inspection plan.

We would like to remark that this application uses the same here presented architecture, reusing the robot manager, GUI, logging, and monitoring modules without any modification.

VII. DISCUSSION AND SUGGESTED IMPROVEMENTS

The greenhouse simulation tests have been used to validate:

The generation of pest inspection and treatment plans by the IPM; the integration of *Robotframework* with navigation and manipulation modules; the execution of the IPM generated plans (SR1, SR2, SR4); the proper notification and continuation of the plan when a navigation fails (SR5); manipulation strategies workflow for pest inspection and treatment tasks.

The greenhouse field tests have been used to validate the execution of simple pest inspection and treatment plans on robot's right and left sides using the real perception modules and spraying equipment (SR2, SR3, SR4).

In both cases the following additional validations have been carried out: use of the GUI for starting the plan, visualizing system status (SR7), and pausing, stopping, or restarting the plan on demand; storage of logs in the cloud and subsequent use of the logs to obtain metrics, monitor the correct performance of the system and detect incidents (SR6).

The system requirements presented in Table 1 have been obtained through the analysis of needs of a mobile manipulator in a greenhouse environment. However, the architecture has shown to be generic and not limited to the application or robot configuration presented here. New agricultural use cases could similarly make use of the architecture to integrate their satellite-based open-field localization and navigation modules reusing mechanisms such as the operation mode or the recovery behaviors. Once the navigation target is reached, new inspection or dexterity tasks such as weed removal or harvesting could be integrated using previously implemented tasks as templates. The architecture is also valid for industrial robotic applications working in more structured environments as presented in the CRO-INSPECT aileron inspection use case.

The plans generated by the high-level decision modules must keep a similar structure as shown in the plan examples in Figures 11, 12 and 17. The parametrization of the proposed solution provides however a great extendibility and adaptability. On the one hand, an application can decide whether to skip the navigation step by setting the *navigate* variable to *false* or using its different variances by setting the corresponding *NavigationTypes*. Most common global, relative, and precise navigation functionalities are already provided by the framework and can be used on demand. The number of *NavigationTrials* or the specific *targetPoses* can be also set on demand for each individual step. On the other hand, application specific tasks are developed independently to the architecture without needing to modify or recompile the core of the framework. New tasks are implemented using the ROS *pluginlib* concept. These are dynamically loaded and executed in runtime and provide behavior flexibility through the *param* variables as shown in the pest inspection and treatment plans. It is remarkable that the use cases presented here completely decouple the navigation and manipulation steps, while a combined solution is yet possible if required within the task step of the state machine.

The use of ROS makes the system hardware agnostic, being possible to replace the mobile base, arm, sensors, or end-effectors without affecting the architecture. In addition, the available framework infrastructure with GUI, logging or application management modules will significantly reduce the time to build up a new robotic prototype with similar requirements.

Finally, valuable observations have been obtained from the greenhouse field tests. First, the number of movements and pictures to be acquired depends on the number of leaves found. In the real scenario this depends on the leaf detector module which has faced the following challenges: closeness to leaves reduces the camera field of view and therefore the number of detected leaves; changes in the illumination also reduces the number of detected leaves. Therefore, the leaf detector DL model should be retrained with closer images, perspectives, and illuminations in greenhouse environment. Second, it was observed that most pictures analyzed during the tests where healthy images despite some more plants where actually infected. It should be possible to increase the probabilities of acquiring infected leaves pictures by including a Single Shot Detector (SSD) real time pest detector model. This model could be combined with the current leaf detector model to approach the most probable leaves with pests first. In addition, some pictures had not enough quality and where not valid for the pest detection model. In the future the quality threshold must be increased.

VIII. CONCLUSION

The *Robotframework* architecture presents an innovative and efficient solution that combines centralized high-level decision system, here represented by the IPM strategy, with a robot able to navigate inside greenhouses without additional infrastructure while performing early pest detection and control in an autonomous way. *Robotframework* includes additional logging, monitoring and error handling modules and an intuitive GUI to manage instructions coded in JSON notation which are human understandable and easily configurable to load navigation or customized tasks on demand.

The architecture is based on ROS, and its modular design in conjunction with the *pluginlibs* design makes it easy to be reused in other contexts without needing to change the main core of the architecture. The state machine presented in the application layer is common for all applications but the parametrizable plan generation and execution methods makes it highly adaptable and extensible for new applications. The architecture supports three different types of navigation modes by default. Besides, three real scenario tasks were presented: two agricultural tasks to detect and treat pests in greenhouses, and an additional industrial task for an aileron inspection.

Greenhouse simulation and field tests have been performed to validate the architecture. Although a single simulation test has been presented here, more than 60 hours of simulation have been performed during the validation of the GreenPatrol project. The field tests have been used to evaluate manipulation, leaves detection and pest identification. The project's YouTube channel [54] contains audiovisual material presenting the robot during the simulated and greenhouse validation tests, the user interface and the achievements reached during this almost 3 years project. Finally, the obtained results have been discussed and used to identify the main challenges entailing autonomous pest detection and treatment tasks with robots in greenhouses and to propose future work and improvements based on the experience acquired.

We believe that the use of Robotframework can significantly reduce the time to build up new mobile manipulator robotic applications for other agriculture or industry related tasks.

REFERENCES

[1] *Galileo Enhanced Solution for Pest Detection and Control in Greenhouse Fields with Autonomous Service Robots*. Accessed: Jul. 2, 2021. [Online]. Available: https://cordis.europa.eu/project/id/776324/reporting

[2] M. Rathee, P. K. Dalal, and S. Mehra, "Integrated pest management under protected cultivation: A review," *J. Entomol. Zool. Stud.*, vol. 6, no. 2, pp. 1201–1208, 2018.

[3] A. Khanna and S. Kaur, "Evolution of Internet of Things (IoT) and its significant impact in the field of precision agriculture," *Comput. Electron. Agricult.*, vol. 157, pp. 218–231, Feb. 2019.

[4] M. S. Farooq, S. Riaz, A. Abid, K. Abid, and M. A. Naeem, "A survey on the role of IoT in agriculture for the implementation of smart farming," *IEEE Access*, vol. 7, pp. 156237–156271, 2019.

[5] R. Shamshiri, C. Weltzien, I. A. Hameed, I. J. Yule, T. E. Grift, S. K. Balasundram, L. Pitonakova, D. Ahmad, and G. Chowdhary, "Research and development in agricultural robotics: A perspective of digital farming," *Int. J. Agric. Biol. Eng.*, vol. 11, no. 4, pp. 1–14, 2018.

[6] S. L. Young and D. K. Giles, "Targeted and microdose chemical applications," in *Automation, The Future of Weed Control in Cropping Systems*. Amsterdam, The Netherlands, 2014, pp. 139–147.

[7] M. Pattinson *et al.*, "GNSS precise point positioning for autonomous robot navigation in greenhouse environment for integrated pest monitoring," in *Proc. 12th Annu. Baška GNSS Conf. (BASKA)*, Baška, Croatia: Zenodo, 2018, doi: 10.5281/zenodo.2620125.

[8] D. Obregón, R. Arnau, M. Campo-Cossio, J. G. Arroyo-Parras, M. Pattinson, S. Tiwari, I. Lluvia, O. Rey, J. Verschoore, L. Lenza, and J. Reyes, "Precise positioning and heading for autonomous scouting robots in a harsh environment," in *Proc. Int. Work-Conf. Interplay Between Natural Artif. Comput.*, vol. 2019, pp. 82–96.

[9] R. F. Carpio, C. Potena, J. Maiolini, G. Ulivi, N. B. Rossello, E. Garone, and A. Gasparri, "A navigation architecture for ackermann vehicles in precision farming," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 1103–1110, Apr. 2020.

[10] A. Gutierrez, A. Ansuategi, L. Susperregi, C. Tubío, I. Rankić, and L. Lenža, "A benchmarking of learning strategies for pest detection and identification on tomato plants for autonomous scouting robots using internal databases," *J. Sensors*, vol. 2019, pp. 1–15, May 2019.

[11] Y. Dong, F. Xu, L. Liu, X. Du, B. Ren, A. Guo, Y. Geng, C. Ruan, H. Ye, W. Huang, and Y. Zhu, "Automatic system for crop pest and disease dynamic monitoring and early forecasting," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 4410–4418, 2020.

[12] K. Jensen, M. Larsen, S. Nielsen, L. Larsen, K. Olsen, and R. Jørgensen, "Towards an open software platform for field robots in precision agriculture," *Robotics*, vol. 3, no. 2, pp. 207–234, Jun. 2014.

[13] *GreenPatrol EU Project*. Accessed: Jul. 2, 2021. [Online]. Available: http://www.greenpatrol-robot.eu/

[14] K. Hu, G. Coleman, S. Zeng, Z. Wang, and M. Walsh, "Graph weeds net: A graph-based deep learning method for weed recognition," *Comput. Electron. Agricult.*, vol. 174, Jul. 2020, Art. no. 105520.

[15] *RHEA EU Project*. [Online]. Available: http://www.rhea-project.eu/

[16] A. Ruckelshausen, P. Biber, M. Dorna, H. Gremmes, R. Klose, A. Linz, F. Rahe, R. Resch, M. Thiel, D. Trautz, and U. Weiss, "BoniRob: An autonomous field robot platform for individual plant phenotyping," *Precis. Agric.*, vol. 9, pp. 841–847, Jan. 2009.

[17] S. Best, J. Hemming, E. Pekkeriet, W. Saeys, Y. Edan, A. Shapiro, M. Hočevar, R. Oberti, M. Armada, H. Ulbrich, and J. Baur, "CROPS: Clever robots for crops," *Eng. Technol. Ref.*, vol. 1, no. 1, pp. 1–11, 2015.

[18] Y. Wang, X. Chen, and P. Liu, "Statistical multipath model based on experimental GNSS data in static urban canyon environment," *Sensors*, vol. 18, no. 4, p. 1149, Apr. 2018.

[19] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte Carlo localization for mobile robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 1999, pp. 1322–1328.

[20] K. Fue, W. Porter, E. Barnes, and G. Rains, "An extensive review of mobile agricultural robotics for field operations: Focus on cotton harvesting," *AgriEngineering*, vol. 2, no. 1, pp. 150–174, Mar. 2020.

[21] W. LiliZ. Bo, F. Jinwei, H. Xiaoan, W. Shu, L. Yashuo, Q. Zhou, and W. Chongfeng, "Development of a tomato harvesting robot used in greenhouse," *Int. J. Agricult. Biol. Eng.*, vol. 10, no. 4, pp. 140–149, 2017.

[22] B. Arad, J. Balendonck, R. Barth, O. Ben-Shahar, Y. Edan, T. Hellström, J. Hemming, P. Kurtser, O. Ringdahl, T. Tielen, and B. Tuijl, "Development of a sweet pepper harvesting robot," *J. Field Robot.*, vol. 37, no. 6, pp. 1027–1039, Sep. 2020.

[23] F. Qingchun, W. Zou, P. Fan, C. Zhang, and X. Wang, "Design and test of robotic harvesting system for cherry tomato," *Int. J. Agricult. Biol. Eng.*, vol. 11, no. 1, pp. 96–100, 2018.

[24] A. Mandow, J. M. Gomez-de-Gabriel, J. L. Martinez, V. F. Munoz, A. Ollero, and A. Garcia-Cerezo, "The autonomous mobile robot AURORA for greenhouse operation," *IEEE Robot. Autom. Mag.*, vol. 3, no. 4, pp. 18–28, Dec. 1996.

[25] W. Peng, W. Pengbo, and G. Changxing, "A combined visual navigation method for greenhouse spray robot," in *Proc. IEEE 9th Annu. Int. Conf. CYBER Technol. Autom., Control, Intell. Syst. (CYBER)*, Jul. 2019, pp. 604–608.

[26] T. Hellström and O. Ringdahl, "A software framework for agricultural and forestry robots," *Ind. Robot, Int. J.*, vol. 40, no. 1, pp. 20–26, Jan. 2013.

[27] *Navigation Stack*. Accessed: Oct. 5, 2020. [Online]. Available: http://wiki.ros.org/navigation

[28] *MoveIt! Motion Planning Framework*. Accessed: Apr. 10, 2021. [Online]. Available: https://moveit.ros.org/

[29] J. Liao, F. Huang, Z. Chen, and B. Yao, "Optimization-based motion planning of mobile manipulator with high degree of kinematic redundancy," *Int. J. Intell. Robot. Appl.*, vol. 3, no. 2, pp. 115–130, 2019, doi: 10.1007/s41315-019-00090-7.

[30] H. Deng, J. Xiong, and Z. Xia, "Mobile manipulation task simulation using ROS with MoveIt," in *Proc. IEEE Int. Conf. Real-time Comput. Robot. (RCAR)*, Jul. 2017, pp. 612–616, doi: 10.1109/RCAR.2017.8311930.

[31] H. Zhou, W. Chou, W. Tuo, Y. Rong, and S. Xu, "Mobile manipulation integrating enhanced AMCL high-precision location and dynamic tracking grasp," *Sensors*, vol. 20, no. 22, p. 6697, Nov. 2020, doi: 10.3390/s20226697.

[32] G. P. Prathibha, T. G. Goutham, M. V. Tejaswini, P. R. Rajas, and K. Balasubramani, "Early pest detection in tomato plantation using image processing," *Int. J. Comput. Appl.*, vol. 96, no. 12, pp. 22–24, Jun. 2014, doi: 10.5120/16847-6707.

[33] A. F. Fuentes, S. Yoon, J. Lee, and D. S. Park, "High-performance deep neural network-based tomato plant diseases and pests diagnosis system with refinement filter bank," *Frontiers Plant Sci.*, vol. 9, p. 1162, Aug. 2018.

[34] P. Boissard, V. Martin, and S. Moisan, "A cognitive vision approach to early pest detection in greenhouse crops," *Comput. Electron. Agricult.*, vol. 62, no. 2, pp. 81–93, Jul. 2008.

[35] J. R. Fiona and J. Anitha, "Automated detection of plant diseases and crop analysis in agriculture using image processing techniques: A survey," in *Proc. IEEE Int. Conf. Electr., Comput. Commun. Technol. (ICECCT)*, Feb. 2019, pp. 1–5, doi: 10.1109/ICECCT.2019.8869316.

[36] K. Steffen, F. Grousset, F. Petter, M. Suffert, and G. Schrader, "EU-project DROPSA: First achievements regarding pathway analyses for fruit pests," *EPPO Bull.*, vol. 45, no. 1, pp. 148–152, Apr. 2015.

[37] *Isefor—European State Forest Association*. Accessed: Jul. 2, 2021. [Online]. Available: https://eustafor.eu/eu-projects/isefor/

[38] *PALM PROTECT EU Project*. Accessed: Jul. 2, 2021. [Online]. Available: https://secure.fera.defra.gov.uk/palmprotect/

[39] *EMPHASIS EU Projec*. Accessed: Jul. 2, 2021. [Online]. Available: http://www.emphasisproject.eu/

[40] M. J. Matarić, "Situated robotics," in *Encyclopedia of Cognitive Science*. London, U.K.: Macmillan, 2002.

[41] *Robot Operating System (ROS)*. Accessed: Jul. 2, 2021. [Online]. Available: https://www.ros.org/

[42] *Gmapping, Laser-Based SLAM*. Accessed: Jul. 2, 2021. [Online]. Available: http://wiki.ros.org/gmapping

[43] *Segway RMPv3 Drivers ROS Wiki*. Accessed: Jul. 2, 2021. [Online]. Available: http://wiki.ros.org/Robots/RMPv3

[44] *Velodyne Laser Drivers ROS Wiki*. Accessed: Jul. 2, 2021. [Online]. Available: http://wiki.ros.org/velodyne

[45] *Omron os32c Laser Drivers ROS Wiki*. Accessed: Jul. 2, 2021. [Online]. Available: http://wiki.ros.org/omron_os32c_driver

[46] *Kuka Lbr Iiwa Drivers ROS Wiki*. Accessed: Jul. 2, 2021. [Online]. Available: http://wiki.ros.org/kuka_lbr_iiwa_support

[47] *Intel RealSenseTM Drivers ROS Wiki*. Accessed: Jul. 2, 2021. [Online]. Available: http://wiki.ros.org/RealSense
[48] *UEye Cameras IDS Drivers ROS Wiki*. Accessed: Jul. 2, 2021. [Online]. Available: http://wiki.ros.org/ueye_cam
[49] *RabbitMQ*. Accessed: Jul. 2, 2021. [Online]. Available: https://www.rabbitmq.com/
[50] *ElasticSearch*. Accessed: Jul. 2, 2021. [Online]. Available: https://www.elastic.co/
[51] *Gazebo Simulator*. Accessed: Jul. 2, 2021. [Online]. Available: http://gazebosim.org/
[52] *CRO-INSPECT*. Accessed: Jul. 2, 2021. [Online]. Available: https://cordis.europa.eu/project/id/716935
[53] I. Lluvia, A. Ansuategi, C. Tubí, L. Susperregi, I. Maurtua, and E. Lazkano, "Optimal positioning of mobile platforms for accurate manipulation operations," *J. Comput. Commun.*, vol. 7, no. 5, pp. 1–16, 2019.
[54] *GreenPatrol Youtube Chanel*. Accessed: Jul. 2, 2021. [Online]. Available: https://www.youtube.com/channel/UCuI_1ySFqs26byudAeMzQAQ

JON MARTIN received the M.Sc. degree in industrial electronics and automatic control from the University of the Basque Country, in 2014, where he is currently pursuing the Ph.D. degree. From 2014 to February 2019, he worked at Technische Hochschule Nürnberg Georg Simon Ohm, as a Research Assistant, working on mobile robotics and coordinating the university roboCup@work team. In March 2019, he was a Researcher with the Smart and Autonomous Systems Unit, Tekniker, taking part in recent European projects, such as CROINSPECT, GREENPATROL, and PICKPLACE. His research interests include artificial intelligence, planning and navigation in mobile robots, and industrial robot manipulation.

ANDER ANSUATEGI received the M.Sc. and Ph.D. degrees in computer science from the University of Basque Country, in 2005 and 2012, respectively. In October 2006, he was a Researcher at the Tekniker. With more than ten years of experience in robotics research, he is currently the Coordinator of the research activities in robotics at the Tekniker. During these ten years, he has taken part in numerous European projects. He has some articles in peer-reviewed international journals and contributions to several congresses related to his field of expertise. His interests include artificial intelligence, data management, planning and navigation in mobile robot, and machine learning. He has taken part in European projects, such as MAINBOT, ROBO-PARTNER, FOURBYTHREE, and GREENPATROL, which are related with sensors data management and robotics. He is coordinating the experiment PRODETECT running as part of the second open call of the COVR H2020 Project.

IÑAKI MAURTUA is currently an electrical engineer with more than 30 years of experience. He is the Head of the Smart and Autonomous System Unit and has participated and coordinated several European and national projects dealing with robotics and machine vision, AI, wearable computing, system monitoring, and flexible manufacturing systems (taking part in the development of seven flexible manufacturing cells, including the programming of the robotic elements). He was in charge of the UCD process in the FP6 WEARIT@WORK Project and the specific scenario for production. He has been the Coordinator of FP7-ROBOFOOT, FP7-MAINBOT, and H2020-FourByThree Projects. Nowadays, he is the Technical Manager of ESMERA Project.

AITOR GUTIERREZ received the degree in computer science engineering from the University of the Basque Country (UPV/EHU), in 2010. He started working as an IOS developer in a startup in West Virginia, USA. In 2012, he joined the Vicomtech, a research and development center focused on computer vision, working in several research projects about computer vision, machine learning, the IoT software architectures, and open hardware. In January 2018, he was a Researcher at the Smart and Autonomous Systems Unit, Tekniker, working in different project related to deep learning and computer vision. He is coordinating several industrial projects in these fields.

DAVID OBREGÓN is currently pursuing the Ph.D. degree in intelligent systems with UNED, with a focus on location systems in robotics, probabilistic algorithms and navigation in greenhouses, with additional training in areas of artificial intelligence, such as neural networks or evolutionary computing. Since 2017, he has been working in the navigation and robotics area at the Centro Tecnoló gico de Componentes (CTC). He is an Engineer in computer science at UNED. He is a Technical Engineer in computer management at UCLM. His main research interests include being autonomous location and navigation systems in robotic environments. Previously his career was framed in the analysis and development of business software, where he accumulated more than 15 years of experience.

OSKAR CASQUERO received the B.S. degree in telecommunication engineering, in 2003, and the Ph.D. degree in engineering, in 2013. From 2004 to 2007, he worked as an IT Architecture Analyst with the University of the Basque Country, Virtual Campus. Since 2007, he has been working as an Assistant Professor with the Systems Engineering and Automatic Control Department. He is currently at the Faculty of Engineering, Bilbao. He investigates on smart and flexible manufacturing systems using digital twins, model-driven engineering, multi-agent systems, and cloud computing technologies.

MARGA MARCOS (Senior Member, IEEE) received the B.Sc. degree and the M.Sc. and Ph.D. degrees in control engineering from the University of the Basque Country (UPV/EHU), Spain, in 1983, 1984, and 1988, respectively. She is currently a Professor in control engineering at Basque Country University, where she was the Vice-Dean of the Faculty of Engineering, from 1990 to 1993. From 1995 to 2005, she was the Chairman of the Control Department. She has authored and coauthored more than 200 technical articles in international journals and more than 200 technical papers in conference proceedings. She has acted as the main researcher of more than 80 research projects funded by national and European research and development programs. Her main research interests include application of model driven engineering to automation systems and the application of industrial agents in manufacturing. She is a member of EUCA and NOC of IFAC Spain. She is a member of the IFAC Council for the 2020–2023 triennium. She has served and serves on TCs of IFAC (AARTC, WRTP, and CC) and IEEE (NBCS, ICPS, and IA). From 2014 to 2017, she was the Chair of the IFAC TC Computer for Control. She is the Publication Co-Chair for IEEE CDC2005 and the General Co-Chair for IEEE ETFA 2010. From 2018 to 2020, she has served as an Associate Editor for IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING (T-ASE) journal.