Human-Robot Cooperation in Surface Inspection Aerial Missions

Martin Molina[1], Pedro Frau[1], Dario Maravall[1]
Jose Luis Sanchez-Lopez[2,3], Hriday Bavle[2], Pascual Campoy[2]
[1]Department of Artificial Intelligence, Technical University of Madrid (UPM), Spain
[2]Centre for Automation and Robotics, UPM-CSIC, Spain
[3]Interdisciplinary Centre for Security, Reliability and Trust, University of Luxembourg, Luxembourg

Abstract

The goal of the work presented in this paper is to facilitate the cooperation between human operators and aerial robots to perform surface inspection missions. Our approach is based on a model of human collaborative control with a mixed initiative interaction. In the paper, we present our human-robot cooperation model based on the combination of a supervisory mode and an assistance mode with a set of interaction patterns. We developed a software system implementing this interaction model and carried out several real flight experiments that proved that this approach can be used in aerial robotics for surface inspection missions (e.g., in vision based indoor missions). Compared to a conventional tele-operated inspection system, the solution presented in this paper gives more autonomy to the aerial systems, reducing the cognitive load of the operator during the mission development.

1 Introduction

Certain types of missions in aerial robotics may require special human-robot interaction with intermediate degrees of robot autonomy between manual teleoperation and complete autonomy. One example of this type of mission is surface inspection in which the operator uses aerial robots to inspect the state of a certain surface (e.g., an indoor wall, the surface of a dam, the facade of a building, etc.) to find defects (e.g., holes, fissures, mold, spots, humidity, etc.) as symptoms of potential problems due to, for example, structural imperfections.

In this type of scenario, the aerial robot may operate as an assistant for the human operator who delegates in the vehicle inspection tasks. The robot may have certain inspection abilities (e.g., path planning, defect recognition, etc.). These abilities may reduce significantly the workload of the operator and increase safety, compared to simple manual teleoperation. However, in this type of mission, it is difficult to have robots that operate fully autonomously because they may not have a complete understanding of the environment. Robots may have recognition abilities for certain defects but, sometimes, certain defects are difficult to classify automatically. In this case, the robot may ask for assistance to the operator, which requires a richer interaction model between operator and robot.

The goal of this paper is to present preliminary results of our ongoing research work to analyze more complex human-robot interaction in surface inspection missions. In our work, we have followed the general concept of collaborative control to formulate a specific human-robot interaction model designed for mission inspections. Our approach combines two interaction modes, supervisory and assistance (with a set of interaction patterns). We implemented this model using the software framework Aerostack (`www.aerostack.org`) [1, 2] and developed several flight experiments that proved the adequacy of this approach for aerial robotics.

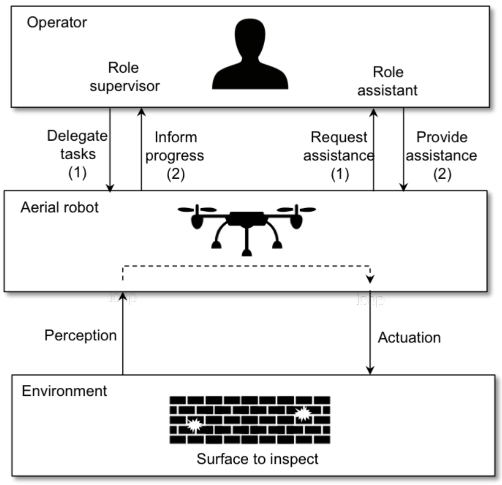


Figure 1: Collaborative control for surface inspection.

The remainder of the paper describes our model and the main results of our work. Section 2 describes the type of user-system collaboration that we have identified for this problem. Section 3 the required inspection abilities. Section 4 describes how we implemented it using Aerostack and, finally, section 5 describes real flight experiments that we performed to refine and evaluate our approach.

2 THE HUMAN-ROBOT INTERACTION MODEL

The problem that we consider in this work is the development of a surface inspection mission performed by a human operator and one or several aerial robots. The goal is to explore a spatial region of a given surface to detect the presence of certain defects. A simple example of this problem is to find imperfections such as fissures or holes in the surface of a wall.

Figure 1 summarizes this type of human-robot interaction. On the one hand, the operator can play the role of supervisor. This form of automation is related to the notion of supervisory control [3] in which a human operator is intermittently acting on the robot to delegate tasks. The robot closes an autonomous control loop through effectors to the environment. This concept has been used to design flexible interaction models, for example, for military mission planning of UAVs [4], swarming networks [5] or remote surveillance system [6].

But, on the other hand, the operator can also play the role of assistant. The human works as a resource for the robot, providing additional information. The robot may ask the human questions as it works, to obtain assistance with perception and cognition. This allows the human to compensate for limitations of autonomy. This is related to the idea of collaborative control in which human and robot work together [7]. The human and the robot dialogue to exchange information, to ask questions, and to resolve differences. This interaction scheme is a kind of mixed initiative approach [8]. Both, the operator and the robot, may take the initiative of the conversation during the dialogue.

Based on this collaborative scheme, we designed a general human-robot interaction model considering messages in categories according to the theory of speech acts [9, 10, 11]. We consider different illocutionary acts to distinguish the intention of the messages, and other subcategories defined by different schemes: DAMSL (Dialogue Act Markup In Several Layers) [12], KQML [13], Move Coding Scheme [14], etc. In particular we use the following categories:

- *Assertive messages*. These messages are sent to give certain information to the receiver (for example, the robot informs the operator the completion of a task).
- *Directive messages*. These messages cause the receiver to take a particular action. Within directive messages, we distinguish between two categories: action directives (requests for action) and information requests.

Our interaction model is divided in two interaction modes (supervisory mode and assistance mode) that are described in the following sections.

2.1 *Supervisory mode*

In this interaction mode, the operator sends action directives to the robot in order to delegate mission tasks to the robot. The operator may ask the aerial robot to perform an inspection mission, specifying the area to cover and the exploration strategy. In this case, the relation between operator and robot follows a hierarchical authority (as supervisor-subordinate schema) in which the operator delegates a set of tasks to the robot.

During the development of the mission, the operator observes the robot behavior and the robot sends assertive messages to inform about the mission execution progress (e.g., completed task or finished mission). These messages are useful for the operator to verify that the mission is developing as expected. The operator can interrupt the mission under certain circumstances (for example, to avoid wrong behaviors).

In this interaction mode, the robot shows autonomy to adapt to a dynamic environment while tries to reach its goal [15]. But the robot shows also autonomy to accept or reject the proposed actions according to characteristics of the environment and its own goals (e.g., safety goals) [16]. Its behavior is not completely determined by the influence of the operator.

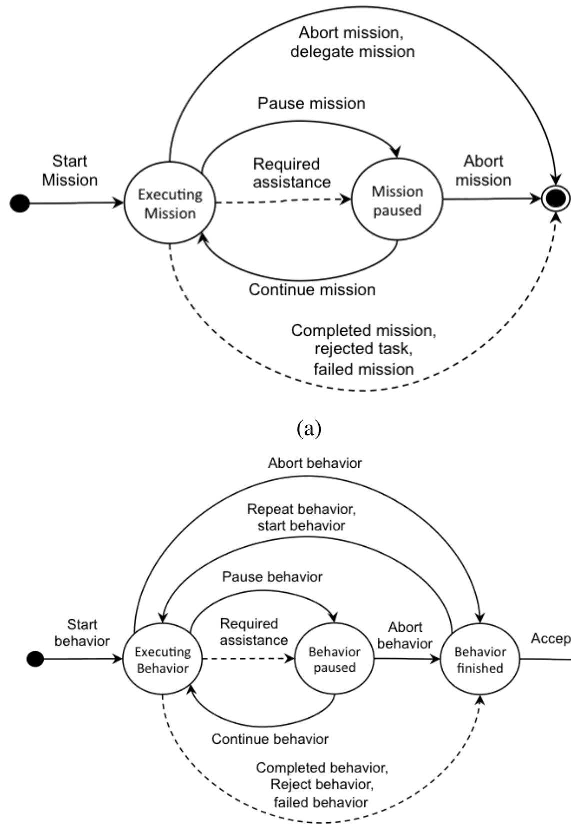


Figure 2: Dialog model for mission control (a) and behavior control (b).

Figure 2a shows the structure of the dialog for mission control. The figure is a state-transition diagram. Transitions with continue arrows correspond to messages sent to the robot by the operator: start mission, pause mission, continue mission, abort mission and delegate mission to other robot. Transitions with dashed arrows correspond to types of messages sent by the robot.

The operator can also control the execution activating and canceling individual robot behaviors (e.g., land, go to point, turn lights on, etc.). Figure 2b shows the dialog model for this type of interaction in which the operator can send messages such as start behavior, pause behavior, continue behavior, repeat behavior and abort behavior.

Both models include a specific transition called required assistance. This transition represents that the robot has reached an impasse because there is an event that prevents from continuing the execution and needs operator assistance.

2.2 *Assistance mode*

In the presence of uncertainty, a robot may ask the operator for assistance. This interaction mode is required because robots have partial knowledge and are not completely self-sufficient. In this case, the robot works like the field technician (i.e., it is skilled, but may need help) and the operator is like the expert (i.e., she or he can provide assistance when needed) as it is considered in human collaborative control [17].

In the particular case of inspection missions, robots may have recognition abilities for certain defects but, sometimes, certain defects are difficult to classify automatically. In addition, unexpected changes of the environment (e.g., shadows, wind, etc.) may require attention from the operator to decide the appropriate response.

The interaction mode for assistance starts, for example, when the robot is not able to recognize the category of a detected defect in the surface. In this case, the robot sends an information request to ask the operator for the category of the detected defect. The operator answers the category or rejects the detection. In addition, the operator may help the robot proposing motion actions to have better views of the surface.

This interaction mode may also start when the robot is not able to complete a requested task because there is a problem in the environment such as: low visibility, low battery, lost position, high vibrations, impassable barrier, or unstable ground. This includes also the time out event that happens when the robot is not able to complete the task in the expected time due to unknown reasons. The operator helps the robot saying how to respond to these events.

We consider also that the robot may delegate certain specialized tasks to other robots. For example, the robot can transfer part of the mission to other robot because it does not have enough battery charge, or the robot can delegate a certain specialized task that require specialized actuators (e.g., use a special device to mark the detected defect on the wall).

To formulate a model for this type of dialog, we use interaction patterns (see table 1). Each interaction pattern expresses how the robot must interact with the operator in the presence of a certain event. Each pattern is a tuple consisting of two parts. The first part of the pattern is the event that generates an impasse in the development of the mission. The event is the reason why the assistance is required. The second part of the pattern is a list of suggested actions to be presented to the operator (for some of these actions, additional question-answers are needed).

For example, the third pattern in the table is used by the robot in the following way. When there is low visibility, the robot informs about this event to the operator and asks the operator for the next action to do, showing a list of suggested actions. In this case, the operator can select either (1) turn on the lights and continue mission or (2) abort mission and land. If one of these options is selected by the operator, it is performed automatically. The list of suggested actions is not closed. This means that, instead of the suggested actions, the operator can decide to do any other action using the supervisory mode.

Event	**Suggested actions**
Unknown object	• Zoom in • Zoom out • Learn new category • Horizontal exploration • Vertical exploration
Recognized category	• Confirm category and continue mission • Reject category and continue mission • Learn new category • Mark defect
Problem low visibility	• Turn on the lights and continue mission • Abort mission and land
Problem low battery	• Turn off the lights • Turn off the camera • Delegate mission to another robot and land
Problem time out	• Repeat behavior • Continue mission

Table 1: Example interaction patterns for operator assistance.

In general, the list of actions include different types of actions: (1) behavior control directives (e.g., zoom in, land, etc.), (2) mission control directives (e.g., abort mission, continue mission, repeat behavior), and (3) information requests to the operator (e.g., learn new category, a robot to delegate the mission).

The list of actions should not include more than a small number of options (e.g., 5 options) to facilitate an agile communication with the operator. In addition, the list is ordered according to its probability of selection (most probable action first). This list can be initially given by the programmer. But its components could be also learned and its probability updated according the interaction with the operator.

3 ROBOT BEHAVIORS FOR SURFACE INSPECTION

This section describes the specific robot behaviors that we have considered for surface inspection missions.

3.1 Visual recognition of surface imperfections

One of the requirements of an autonomous robot for surface inspection tasks is the ability to detect abnormal marks in a surface and classify the images in the corresponding category. For this purpose, it is possible to use computer vision algorithms.

In this work, we have used the method based on frequency histogram of connected elements (FHCE) [18, 19]. This method is useful to treat the image pixel by pixel and characterize the different types of flaws of the surface.

This method uses the concept of neighborhood, i.e. for a given pixel $p(i,j)$ of an image, its neighborhood is formed by a set of pixels which distances to p are not greater than two integer values r, s and is defined as $\phi^{r,s}_{(i,j)}$. A connected element is the neighborhood selected such as the intensity I of a pixel is a subset of a given grayscale range $[T-\epsilon,\ T+\epsilon]$:

$$C_{(i,j)}(T) = \phi^{r,s}_{(i,j)} \ : \ I(k,l) \subset [T-\epsilon,\ T+\epsilon],\ \forall(k,l) \in \phi^{r,s}_{(i,j)}$$

Given the previous definitions, $H(T)$ is defined as the sum of all the connected elements for each pixel of an image on different gray levels T, where T is greater than 0 and inferior than the maximum intensity minus one:

$$H(T) = C_{(i,j)}(T),\ 0 \leq T \leq I_{max} - 1 \qquad (1)$$

We use this algorithm to separate the flaws from the background, dividing the process into several steps. We defined the neighborhood shape with a square kernel to fit both the hole and fissure needs. Our square kernel is formed by a 3×3 matrix (i.e. $r = 1$ and $s = 1$) of neighboring pixels where the center pixel is the target.

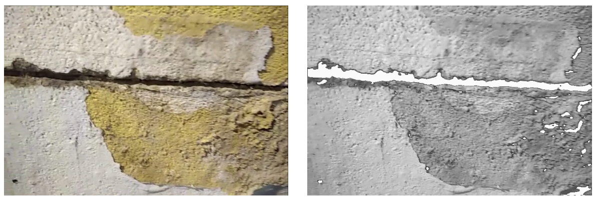


Figure 3: Example of original vs resulting image after flaw separation.

To find the connected elements in the kernel, we calculate the standard deviation of the grayscale values between pixels within the same neighborhood. Then, given a threshold th (in our case 0.1), if the standard deviation is inferior than th, we assign to the pixel the mean gray value of all the pixels of the kernel. If not, we preserve its original grayscale value.

Then, to calculate FHCE we apply equation 1 to the resulting image after separating connected elements. According to FHCE result, we see that the region related to a flaw is characterized by a range of gray levels between 1 and 50 (i.e. dark gray). If we assign a white value to all the pixels in that range, we have as a result an image with all the flaws separated from the background (Figure 3).

Finally, we have to distinguish between categories of flaws. In this work, two main flaws were considered on the inspected surface: fissures and holes. Fissures can be characterized as a linear flaw, which is why we search for areas where white values are concentrated along a certain linear direction. Holes have to be defined as a square (or circular) flaw, which is why we search for areas where white values are concentrated along a square area of a certain dimension. As a result, we will get delimited areas for both flaw types (see Figure 4).

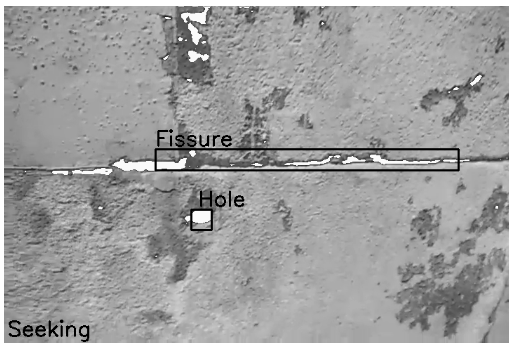


Figure 4: Example of image classification.

3.2 Motion behaviors

In addition to visual recognition of flaws, the robot requires specifc motion behaviors to develop a safe exploration search with an appropriate degree of autonomy. Table 2 shows a list of behaviors that the robot may need for inspection tasks.

We have designed this set of behaviors to be reusable for different inspection missions. There are general behaviors (e.g., TAKE OFF, LAND) and more specific behaviors for inspection tasks (e.g., ZOOM, EXPLORE). For example, the behavior EXPLORE develops an exploration search following a prefixed trajectory to cover the complete surface. The trajectory may follow different strategies such as vertical search or horizontal search (see Figure 5).

4 SYSTEM IMPLEMENTATION

We implemented an experimental software prototype to refine and evaluate the approach for surface inspection missions described in this paper. For this purpose, we used and extended the software framework Aerostack (`www.aerostack.org`) [1, 2]. Aerostack is a general software framework for aerial robotics that provides different soft-

Behavior	Description
TAKE OFF	The robot takes off
EXPLORE	The robot explores the surface following a search strategy. Argument: strategy VERTICAL, HORIZONTAL
LAND	The robot lands
GO HOME	The robot returns to the home base
ZOOM	The robot moves forward to zoom in or backwards to zoom out. Argument: IN, OUT
MOVE	The robot moves a prefixed distance (e.g., 0.5 m) in the specified direction. Argument: direction UP, DOWN, RIGHT, LEFT
KEEP HOVERING	The robot pauses its movement
TURN LIGHT	The robot turns the light on or off. Argument: ON, OFF
MARK SURFACE	The robot draws a colored circle around the flaw
APPROACH	The robot goes to the position of another robot. Argument: robot identifier.

Table 2: Example behaviors for inspection tasks.

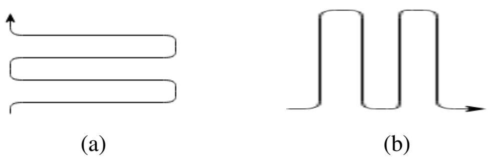


Figure 5: Example of exploration strategies: (a) horizontal and (b) vertical.

ware components and a combination scheme for building the software architecture for autonomous operation of an aerial robotic system. For example, Aerostack provides software components with perception algorithms, SLAM algorithms, controllers, mission plan interpretation methods, multi-robot communication and a general graphical user interface for human-robot interaction.

Figure 6 shows a block diagram with the main software components of our implementation. In the figure, blue blocks correspond to processes provided by Aerostack and orange blocks are new processes that we programmed and added to Aerostack for inspection problems. For instance, we programmed the process called surface defect recognizer that implements the FHCE algorithm presented previously, using the images captured by the front camera of the drone. In this implementation of FHCE we also used simple routines provided by the OpenCV library (e.g., for representation of images, pixel manipulation, etc.). Aerostack provides a library of behaviors such as general motion behaviors (take off, land, got to point, etc.). We extended this library with specific

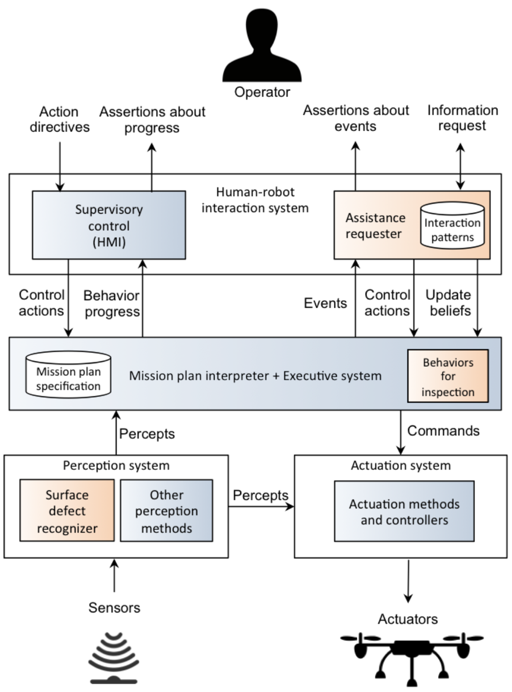


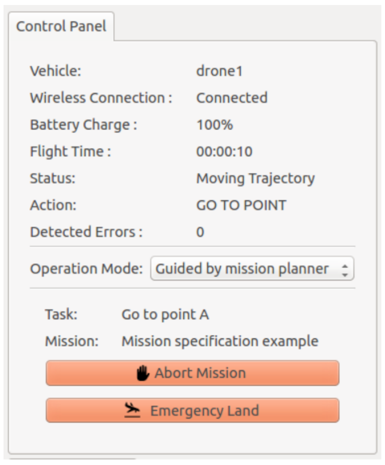
Figure 6: Block diagram with the main software components.

behaviors useful for inspection missions. For example, we implemented the behavior explore with different exploration strategies, or the behavior zoom, to move closer or farther to the surface.

The dialog between the operator and the robot uses three communication channels. First, Aerostack provides a graphical user interface (GUI) to interact with the drone. This interface allows the operator to do supervisory control. Figure 7 shows an example of window presented by the Aerostack GUI which can be used by the operator to control the mission execution (start mission, abort mission, etc.). This window is called control panel, with specific buttons and fields, where operator can control the state of the mission. The robot uses also this panel to show the current action (e.g., take off, go to point, land, etc.). This is useful to help the operator to supervise the correct execution of the mission.

The Aerostack GUI also provides another channel that the operator can use to send behavior directives to the robot. In this case, the operator selects the specific behavior to do (e.g., land, take off, etc.).

The version of Aerostack that we used for this work did not provide the assistance request for human-robot interaction, as we defined in this paper. Therefore, we programmed a new process, called assistance requester, to provide this service. This process receives events corresponding to a mission impasse (e.g., the presence of an unrecognized image or the presence of a problem) and interacts with the operator requesting the appropriate information according to the inter-


Control Panel

Vehicle:	drone1
Wireless Connection :	Connected
Battery Charge :	100%
Flight Time :	00:00:10
Status:	Moving Trajectory
Action:	GO TO POINT
Detected Errors :	0

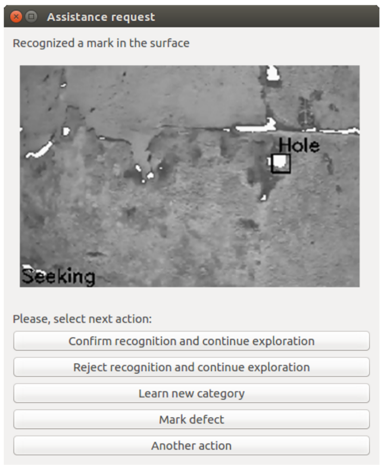
Operation Mode: Guided by mission planner

Task:	Go to point A
Mission:	Mission specification example

Figure 7: Window for mission control provided by Aerostack.

action patterns.

We programmed the assistance requester using a command line interpreter with simple commands with two letters (e.g., cm for continue mission, etc.). However, for a future more complete implementation, we designed two alternative options for assistance request. The first option is based on a interruption window presented to the operator (a pop-up window) asking for operator attention and requesting an answer. Figure 8 shows an example of the design of such a window. We asume that while this window is presented, the robot keeps hovering and, if the operator does not answer after a limited time T (e.g., $T = 45$ seconds), the robot continues the mission.


Assistance request

Recognized a mark in the surface

Please, select next action:

Hole
Seeking
Confirm recognition and continue exploration
Reject recognition and continue exploration
Learn new category
Mark defect
Another action

Figure 8: Example window presented to the operator for assistance request.

An alternative option for the previous window is based on using voice messages and speech recognition used in natural user interfaces (as we propose in [20]). In this case, the robot generates the following text message:

> *The mission is stopped because I have recognized the mark shown on the monitor. Please, say what to do next: (1) confirm recognition and continue exploration, (2) reject recognition and continue exploration, (3) learn a new category, (4) mark defect, and (5) do another action.*

5 EXPERIMENTS

This section shows a preliminary set of scenarios that we analyzed to evaluate our interaction model. Table 3 shows the set of flight experiments that we considered to cover different situations of interaction. The experiments illustrate the kind of collaborative human-robot interaction presented in this paper.

Scenario	Description
Low visibility and mission delegation	Robot R1 finds a dark area which does not allow the proper recognition. The operator orders R1 to turn on the light and continue with the mission. Then, the drone has a low battery charge. The operator transfers the mission to robot R2.
Distributed specialized tasks	Robot R1 finds a hole. The operator orders to delimitate the hole zone. The drone asks for painter drone help. Painter drone draws a circle around the hole. The first drone finishes the mission.
Reconstruction of large fissure	Robot R finds a large fissure. The operator orders to change the inspection strategy to up/down. The drone starts moving taking several pictures of the fissure. Once finished, it makes a reconstruction of the fissure and classifies it correctly.
Zoom out for large fissure	Robot R finds a large fissure. The operator orders to zoom out. The robot gets a better (complete) view and classifies it correctly.
Lost position	Robot R loses its position with respect to the wall. The operator relocates the drone in the inspection area using movement orders and orders to continue the mission.
New object to recognize	Robot R finds an unknown imperfection that cannot be classified following the trained imperfections. The drone asks the operator what to do, and the operator decides to create a new class stain.

Table 3: Example scenarios corresponding to flight experiments.

For example, in the third scenario, the robot develops the inspection until it finds a large fissure. The robot recognizes an unknown object but it is not able to classify it. Then, the operator orders to zoom out. Then, the drone zooms out from the fissure to get a better (complete) view and then, classifies correctly the defect. Then the drone continues the mission and no more objects are recognized.

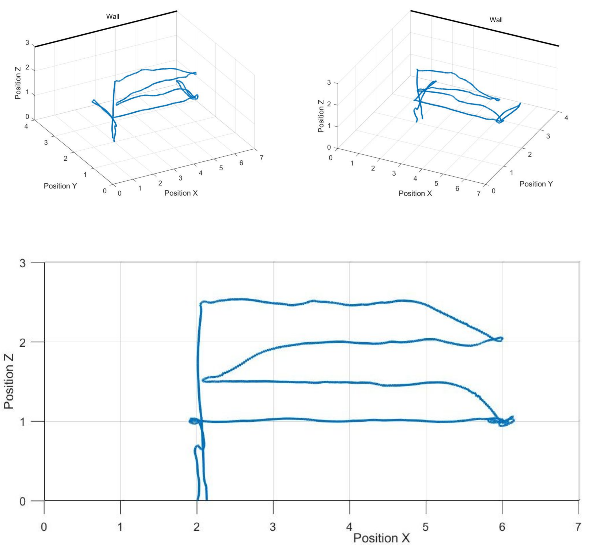


Figure 9: Example exploration trajectory followed by the robot.

Figure 9 shows the trajectory developed by the robot in one of the flight experiments. This example corresponds to an aerial platform AR Drone 2.0 (Figure 10) performing a indoor inspection to find holes and fissures on a wall. The whole trajectory is covered in 1 minute and 19 seconds. The robot develops autonomously an horizontal exploration starting from the takeoff point $(2,\ 2,\ 0)$ and landing at the same point after the exploration. There are two points where the robot zooms in and out to have closer views of the wall: point $(2,\ 3,\ 1)$ and point $(6,\ 3,\ 1)$. In this particular example, the robot performs an autonomous inspection without any interaction with the operator.

Figure 10: The aerial robot (AR Drone 2.0) during wall inspection.

Figure 11 shows examples of recognized defects on the wall, corresponding to a fissure and a hole. In general, the experiments proved that the recognition method was able to classify correctly the 88.5% of fissures and the 49.5% of the holes (evaluated with a sample of 200 images). The recognition performance can be acceptable for the interaction goal presented in this paper, where the robot requires the confirmation of the operator. Especially, the algorithm gives good results on well differentiated flaws from the background. A high performance of the recognition process was outside the goal of this work and may be achieved with future research work.

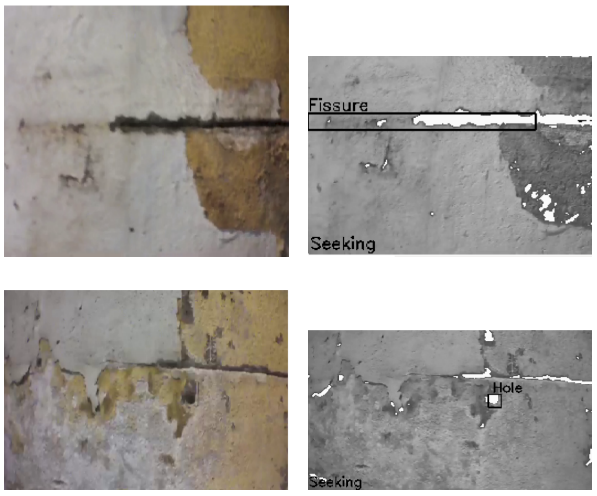


Figure 11: Example of recognized defects on the wall.

6 CONCLUSIONS

In this paper we have presented our human-robot approach for surface inspection tasks based on a collaborative control with a flexible dialogue between operators and robots with a mixed initiative interaction. The human-robot dialog scheme is defined with two main interaction modes, supervision and assistance, with generic interaction patterns. The presented model is generic to be applied for different surface inspection missions with one operator and one or more aerial robots.

Compared to a conventional tele-operated inspection system, the solution presented in this paper leaves more autonomy to the aerial robot, which can reduce the cognitive load of operator during the mission development.

We developed an experimental prototype of a software system to refine and validate this model using the framework Aerostack. The experiments proved that this approach can be used in aerial robotics for surface inspection missions (e.g., vision based indoor missions). However, this preliminary design and implementation still needs to be extended with additional components and more extensive evaluation to show in more detail its contributions and its practial utility.

We plan to extend this approach in the future with more types of behaviors (e.g, other complex exploration strategies) and other types of categories of defects to be used in related scenarios.

ACKNOWLEDGEMENTS

This research work has been partially supported by the Spanish Ministry of Economy and Competitiveness through the project VA4UAV (Visual autonomy for UAV in Dynamic Environments), reference DPI2014-60139-R.

REFERENCES

[1] J. L. Sanchez-Lopez, R. A. Suárez Fernández, H. Bavle, C. Sampedro, M. Molina, J. Pestana, and P. Campoy. Aerostack: An architecture and open-source software framework for aerial robotics. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 332–341, June 2016.

[2] J. L. Sanchez-Lopez, M. Molina, H. Bavle, C. Sampedro, R. A. Suárez Fernández, and P. Campoy. A multi-layered component-based approach for the development of aerial robotic systems: The aerostack framework. *Journal of Intelligent & Robotic Systems*, May 2017.

[3] T. B. Sheridan. *Telerobotics, automation, and human supervisory control*. MIT press, 1992.

[4] C. A Miller and R. Parasuraman. Designing for flexible interaction between humans and automation: Delegation interfaces for supervisory control. *Human factors*, 49(1):57–75, 2007.

[5] M. L. Cummings. Human supervisory control of swarming networks. In *2nd annual swarming: autonomous intelligent networked systems conference*, pages 1–9, 2004.

[6] J.-S. Lee, M.-C. Zhou, and P.-L. Hsu. An application of petri nets to supervisory control for human-computer interactive systems. *IEEE Transactions on Industrial Electronics*, 52(5):1220–1226, 2005.

[7] T. Fong, C. Thorpe, and C. Baur. Multi-robot remote driving with collaborative control. *IEEE Transactions on Industrial Electronics*, 50(4):699–704, 2003.

[8] E. Horvitz. Principles of mixed-initiative user interfaces. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 159–166. ACM, 1999.

[9] J. R. Searle. *Speech acts: An essay in the philosophy of language*, volume 626. Cambridge university press, 1969.

[10] J. R. Searle and K. Gunderson. Language, mind, and knowledge. *Minneapolis: University of Minnesota*, 1975.

[11] J. L. Austin. *How to do things with words*. Oxford university press, 1975.

[12] M. G. Core and J. Allen. Coding dialogs with the damsl annotation scheme. In *AAAI fall symposium on communicative action in humans and machines*, volume 56. Boston, MA, 1997.

[13] Y. Labrou and T. Finin. A proposal for a new kqml specification. Technical report, Technical Report Technical Report TR-CS-97-03, University of Maryland Baltimore County, 1997.

[14] J. Carletta, S. Isard, G. Doherty-Sneddon, A. Isard, J. C Kowtko, and A. H Anderson. The reliability of a dialogue structure coding scheme. *Computational linguistics*, 23(1):13–31, 1997.

[15] R. Murphy. *Introduction to AI robotics*. MIT press, 2000.

[16] C. Castelfranchi. Guarantees for autonomy in cognitive agent architecture. In *International Workshop on Agent Theories, Architectures, and Languages*, pages 56–70. Springer, 1994.

[17] T. Fong, C. Thorpe, and C. Baur. Robot as partner: Vehicle teleoperation with collaborative control. In *Multi-robot systems: From swarms to intelligent automata*, pages 195–202. Springer, 2002.

[18] D. Maravall and M. A. Patricio. Image segmentation and pattern recognition: a novel concept, the histogram of connected elements. In *Pattern recognition and string matching*, pages 411–463. Springer, 2003.

[19] M. A. Patricio and D. Maravall. A novel generalization of the gray-scale histogram and its application to the automated visual measurement and inspection of wooden pallets. *Image and vision computing*, 25(6):805–816, 2007.

[20] R. A. Suárez Fernández, J. L. Sanchez-Lopez, C. Sampedro, H. Bavle, M. Molina, and P. Campoy. Natural user interfaces for human-drone multi-modal interaction. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1013–1022, June 2016.