Auton Robot
https://doi.org/10.1007/s10514-017-9686-1

Analyzing and improving multi-robot missions by using process mining

Juan Jesús Roldán[1] · **Miguel A. Olivares-Méndez**[2] · **Jaime del Cerro**[1] · **Antonio Barrientos**[1]

Received: 8 February 2017 / Accepted: 7 November 2017

Abstract Multi-robot missions can be compared to industrial processes or public services in terms of complexity, agents and interactions. Process mining is an emerging discipline that involves process modeling, analysis and improvement through the information collected by event logs. Currently, this discipline is successfully used to analyze several types of processes, but is hardly applied in the context of robotics. This work proposes a systematic protocol for the application of process mining to analyze and improve multi-robot missions. As an example, this protocol is applied to a scenario of fire surveillance and extinguishing with a fleet of UAVs. The results show the potential of process mining in the analysis of multi-robot missions and the detection of problems such as bottlenecks and inefficiencies. This work opens the way to an extensive use of these techniques in multi-robot missions, allowing the development of future systems for optimizing missions, allocating tasks to robots, detecting anomalies or supporting operator decisions.

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s10514-017-9686-1) contains supplementary material, which is available to authorized users.

✉ Juan Jesús Roldán
jj.roldan@upm.es

Miguel A. Olivares-Méndez
miguel.olivaresmendez@uni.lu

Jaime del Cerro
j.cerro@upm.es

Antonio Barrientos
antonio.barrientos@upm.es

[1] Centre for Automation and Robotics (CAR, UPM-CSIC), José Gutiérrez Abascal, 2, 28006 Madrid, Spain

[2] Interdisciplinary Centre for Security, Reliability and Trust (SnT, uni.lu), Richard Coudenhove-Kalergi, 6, 1359 Luxembourg, Luxembourg

Keywords Robotics · Multi-robot · Mission · Process mining · Modeling · Analysis

1 Introduction

In recent years, multi-robot missions have become popular and have been applied to new scenarios. The reasons for using fleets of robots instead of single robots are diverse. On the one hand, they can carry out the same missions with higher performance by using multiple robots to perform single tasks and choosing the optimal robot to accomplish each task. On the other hand, they can execute missions with more complexity that may require multiple agents and strong coordination. The robot teams can be heterogeneous or homogeneous and integrate ground, aerial, marine and underwater robots according to the mission requirements.

Multi-robot missions share some elements with industrial processes and public services. All these systems involve a problem of assigning some resources (e.g. raw materials, workers, machines or robots) to some processes (e.g. chemical reactions, projects, diagnoses or tasks). Furthermore, they pose another problem where single agents must perform series of operations in a limited time. The first perspective is important for the efficiency, i.e. to optimally use the resources to execute the process, whereas the second one is relevant for the effectiveness, i.e. to perform all the operations in the minimum time. A relevant difference between these systems is the level of uncertainty, which is higher in multi-robot missions than in industrial processes, since there are generally less repetitions, less rigid plans and more factors that could lead to deviations. This difference does not prevent the transference of techniques from one to another field, but could pose additional challenges that must be identified, addressed and

Published online: 23 November 2017

solved, such as the robustness of the models to deviations, the prediction of their evolution at decision points, etc.

The complexity of multi-robot missions depends on multiple factors, such as the number and types of robots, the number and types of tasks, the human agents (both operators and other participants), the volume of telemetry, the level of autonomy, the amount of commands, the requirements of coordination among agents... These missions can be compared to industries, organizations or public services in terms of complexity, agents and interactions.

Process mining is an emerging discipline that involves the modeling, analysis and improvement of processes through the information collected by event logs (Van der Aalst 2011). This discipline has been successfully applied to industries, organizations and public services. Some cases of use are the management of healthcare processes in hospitals (Mans et al. 2008), the analysis of customer behavior in web commerce (Poggi et al. 2013), the management of government offices (van der Aalst et al. 2007), and the detection of transaction fraud at early stages (Jans et al. 2011). Nevertheless, process mining is not extensively used in robotics, despite its potential in multi-robot missions. The most relevant of the few works that follow this approach can be found in Rozinat et al. (2009), which analyzes log files from a robot competition to extract both team strategies and robot actions. The present work goes further bringing more process mining techniques to robotics field (time and resource analysis), as well as introducing changes in other techniques to reach a better performance (event log preparation). Other works address similar problems but apply different techniques (Nair et al. 2004).

This paper explores the application of process mining to multi-robot missions. The main contributions of the work are the definition of a systematic procedure for the application of process mining in the context of robotics, and its application to the analysis of realistic missions of fire surveillance and extinguishing with a fleet of UAVs. Furthermore, the paper solves some practical issues that are not considered in other works, such as the preparation of event logs from telemetry logs in multi-robot missions and the generation of different models to perform different analysis.

Section 2 reviews the literature about multi-robot missions, taking into account the possibility to apply the proposed techniques. Section 3 describes the discipline of process mining, its main objectives and resources. Section 4 presents the systematic procedure developed for analyzing robot missions by means of process mining. Section 5 describes the multi-robot missions performed to generate realistic data. Section 6 shows the results of the analysis of these data. Section 7 discusses the results and proposes changes in the multi-robot missions. Finally, Sect. 8 summarizes the main conclusions of this study.

2 Multi-robot missions

As previously pointed out, multi-robot missions are becoming more common in recent years. Table 1 contains a diverse set of multi-robot missions collected by recent literature. The papers have been chosen giving priority to those with recent dates, variety of missions, robots and scenarios, and good descriptions of experiments. As it can be seen, they cover multiple missions surveillance, monitoring, tracking...) in

Table 1 Some multi-robot missions reported by literature

References	Mission	Robots	Operators	Scenario
Tully et al. (2010)	Area coverage	3 UGVs	No operator	Open field
Janchiv et al. (2011)	Area coverage	2 UGVs	No operator	Indoor
Lindemuth et al. (2011)	Surveillance	1 USV and 1 UAV	N operators	Sea
Valente et al. (2011)	Monitoring	N UAVs	1 operator	Open field
Tsokas and Kyriakopoulos (2012)	People tracking	3 UGVs	No operator	Indoor
Cantelli et al. (2013)	Surveillance	1 UGV and 1 UAV	1 operator	Urban area
De Cubber et al. (2013)	Search and rescue	1 UGV, 1 UAV and 1 USV	N operators	Land and sea
Garzón et al. (2013)	Exploration	1 UGV and 1 UAV	No operator	Urban area
Garzón et al. (2016)	Area coverage	3 UGVs	1 operator	Open field
Kapoutsis et al. (2016)	Mapping	2 UUVs	N operators	Ocean
Kruijff-Korbayová et al. (2015)	Search and rescue	N UGVs and N UAVs	N operators	Disaster area
Nestmeyer et al. (2017)	Exploration	6 UAVs	1 operator	Indoor
Lesire et al. (2016)	Securing	1 UAV and 1 UUV	6 operators	Lake
Roldán et al. (2016b)	Monitoring	1 UGV and 1 UAV	No operator	Greenhouse
Roldán et al. Here	Fire surveillance	2 UAVs	1 operator	Indoor

diverse scenarios (open fields, urban areas, disaster zones...). Additionally, the fleets are homogeneous and heterogeneous with ground, aerial, surface and underwater robots. Finally, the number of operators that control the missions vary from 0 to 6, although some works do not specify it.

There are some advantages in using fleets instead of single robots:

- *Diversity* Heterogeneous robot teams can be applied to scenarios with multiple domains.
- *Effectiveness* Robot teams are more effective than single robots, since they have more resources to perform the same tasks.
- *Efficiency* Robot fleets are also more efficient than single robots, because the allocation of tasks to robots can be optimized.
- *Flexibility* Robot fleets are more flexible than single robots, because they are able to adapt to different scenarios by only changing the assignation of tasks to robots.
- *Fault tolerance* Using robot teams instead of single robots reduces the impact of failures, because there are multiple robots available to perform each task.

On the other hand, there are some challenges in multi-robot missions (Cummings and Mitchell 2008):

- *Situational awareness* The operators must be able to discover the information in the data and understand how the mission is being developed.
- *Operator workload* The operators have to do an effort to perceive information, understand the mission, make decisions and control the robots in a limited time.

Finally, some important issues about multi-robot missions are listed below:

- *Coordination and control architecture* (Roldán et al. 2016c) It determines the roles of operators and robots during the mission, as well as the communication among them. There are three main schemes: centralized (the operators perform the majority of tasks), distributed (the robots perform the majority of tasks) and hybrid (some tasks are performed by operators and the rest by robots).
- *Level of automation* (Beer et al. 2014) It evaluates the capability of robot fleets to perform tasks that were previously carried out by human operators. The literature contains multiple scales of automation: from the scale of ten levels of Sheridan and Verplank (1978) to the one with four levels of Ruff et al. (2002). This last scale considers full manual control, management by consent (i.e. the robots suggest tasks and the operators can accept or reject them), management by exception (i.e. the robots suggest tasks and the operators have a certain time to reject them) and full automatic control.
- *Method of commanding* It defines the interaction among the operators and the robots when the level of automation is low. There are different possibilities such as the use of low-level controllers (e.g. joypad or joystick devices) or the use of high-level commands (e.g. task or waypoint commands).

In this paper we consider a hybrid coordination and control architecture, a manual level of automation and a medium or high-level method of commanding. In fact, one of the objectives of this analysis is to compare the performance of the multi-robot mission with task and waypoint commands.

3 Process mining

Process mining is a discipline that involves the discovery, evaluation and improvement of process models through the information collected by event logs (Van der Aalst 2011). Therefore, this discipline can be located between the conventional process analysis and modeling, and the modern techniques of data mining.

This discipline emerged in the context of business processes, but it has expanded to multiple areas as reported in Mans et al. (2008), Poggi et al. (2013), van der Aalst et al. (2007), Jans et al. (2011). However, the application of this set of techniques to the context of robot missions is still unexplored although it looks really promising.

As shown in Fig. 1, process mining works between models and processes and offers three possibilities: model discovery, reproduction and conformance/enhancement.

Model discovery or play-in techniques generate automatically process models through the behaviors collected by the event logs. These models usually describe the processes as sequences of activities, but they also can explain them according to their use of time or resources. In robotics, these techniques allow generating mission models through the experience of computer simulations or real experiments.

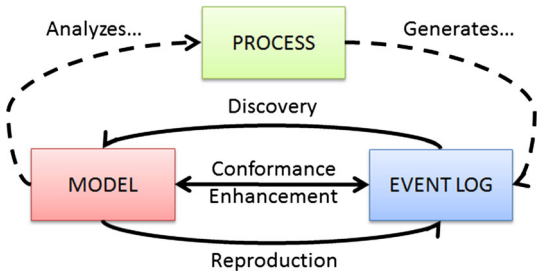


Fig. 1 Process mining and applications (Van der Aalst 2011)

Model reproduction or play-out techniques automatically generate event logs through the process models. These event logs follow the different behaviors that are described by the process models. This information is useful to analyze processes, design components, assign resources or train operators. In the context of robotics, these techniques allow to perform simulations instead of experiments to reduce the costs.

Model conformance and enhancement or replay techniques compare the process models with the event logs. The aim of these techniques is to evaluate the adaptation of process models to event logs and correct their possible deviations: e.g. to discover states or transitions that are not considered by the model. In robotics, these techniques allow the enrichment of analytic models through the experience of missions.

3.1 Event logs

An event log is an ordered set of events that take place during the execution of a process. Each event of the log is defined by a set of fields according to the following structure:

- *Case* It contains a unique identifier of the event.
- *Activity* It is the operation performed in the event.
- *Timestamp* It is the date and time of the event.
- *Resource* It is the agent that performed the event.

3.2 Models

Process mining uses different types of models. The most common models are Petri nets (Van der Aalst 1998) and Business Process Model and Notation (Dijkman et al. 2008). However, there are other alternatives such as the transition systems (van der Aalst et al. 2006) and causal nets (Van Der Aalst et al. 2011).

4 Methodology

The previous section explained process mining, including a series of resources and techniques for modeling and analysis. This section presents a systematic procedure for the application of these techniques in the context of robotics. This procedure is summarized in Fig. 2 and explained in the following sections. Section 4.1 addresses the preliminary mission study, Sect. 4.2 involves the event log preparation, Sect. 4.3 describes the discovery of models, Sect. 4.4 addresses the model evaluation and improvement, Sect. 4.5 reports the analysis from time perspective, and Sect. 4.6 reports the analysis from resource perspective.

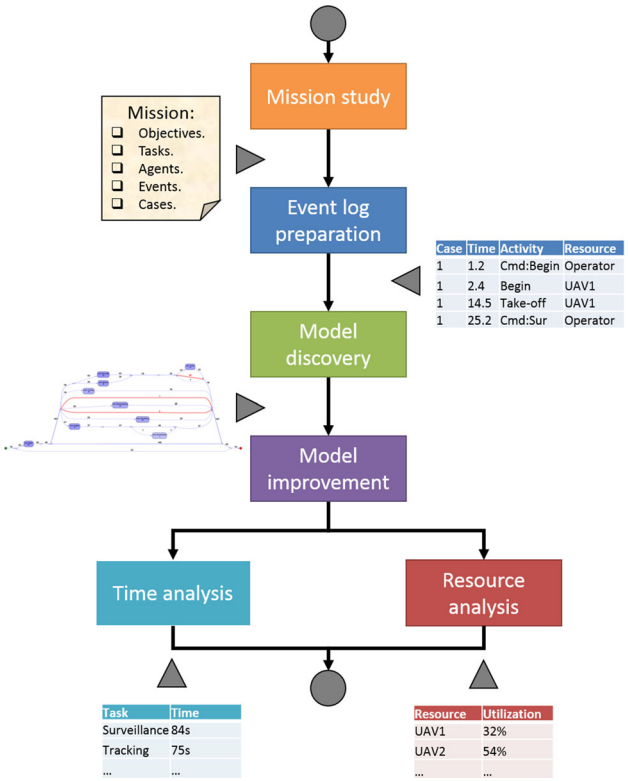


Fig. 2 Flowchart of proposed methodology

4.1 Mission study

First of all, we need to get an overview of the mission to analyze the experience correctly. This overview should include the basic information about the mission, which can be obtained by answering the following questions:

- What is the aim of the mission?
- Which are the different objectives of the mission?
- Which are the events of the mission? How are their types and features?
- How many cases are available?
- Which are the tasks of the mission?
- Which agents are involved in the mission? Which are their actions and interactions?

Additionally, we must define the objectives that we want to achieve with the analysis. Some possible objectives for this kind of analysis are listed below:

- A mission study from time perspective, which may allow to reduce the duration, detect bottlenecks and improve the mission effectiveness.
- A mission study from agent perspective, which may provide information about resource allocation and improve the mission efficiency.

- An analysis of operator, which may allow to understand the behavior and disclose problems of performance or training. This analysis can be performed considering one operator and studying his/her skills and habits, as well as taking into account multiple operators and looking for common strategies and possible deviations.
- An analysis of human-fleet interfaces, which may allow to test them and provide information for future developments.

This preliminary analysis is also conducted when process mining is used in other fields. However, the particularities of robot missions must be taken into account in the questions. A first one is that the analysis should consider "objectives", "robots", "operators" and "tasks", instead of only "resources" and "activities", which can be considered equivalent to "robots + operators" and "tasks" respectively. A second one is that there are usually fewer cases with more variants than in other applications such as industries and services.

4.2 Event log preparation

Multi-robot missions usually generate a huge amount of data like industrial processes or public services. In contrast to them, these data are sometimes incomplete, unstructured or stored in multiple logs. As previously mentioned, process mining works from structured event logs. Therefore, the second step of the procedure, an original contribution of this paper, is the preparation of adequate event logs. Let's distinguish between some possible cases:

- An event log collects all the mission events.
- There are multiple event logs. When the different agents generate different event logs (e.g. base station, interface, robots...), these event logs must be synchronized and merged into a unique one.
- The event log is not complete. When the event log collects some mission events, but it does not consider the rest, these events must be identified in the telemetry log and included in the event log.
- There is not mission event log. The mission events must be generated by discretizing the telemetry variables. For instance, we can identify take-off and landing by looking for changes in the altitude, emergencies by comparing real and goal positions...

Section 3 described the general form of the event logs in process mining. However, the event logs in the context of multi-robot missions may have particular features. Additionally, the generation of different event logs (e.g. considering different criteria for case and resource fields) lead to the discovery of different models and the development of different studies. The structure of these event logs is described below:

- *Case* It is a unique identifier that can represent the mission repetition if we want general mission models to study the relations between the agents (e.g. "M20" for the twentieth mission), or the mission and agent if we want mission models particularized to the agents to study their behaviors "M20A2" for the second agent of twentieth mission.
- *Activity* It can represent the robot tasks (e.g. surveillance and reconnaissance) and actions (e.g. go to waypoint and take a picture), the commands of operator, and the situations of emergency (e.g. the battery is low).
- *Timestamp* It still indicates the date and time when the event took place. Nevertheless, in the context of robotics, we suggest to set separately starting and finishing times of the events to facilitate the future time analysis.
- *Resource* It is the robot that performs the task or the operator that sends the command. In the context of multi-robot missions, this parameter is important to generate social models, as well as to study the resource allocation.

The events can be generated by the algorithms involved in the mission, which are executed by the robots and the interface, or by the scripts that process the telemetry after the mission. The event logs are stored in spreadsheets (usually, XLS, ODS or CSV files) and can be edited with *Disco* (Günther and Rozinat 2012). This program can be used to filter the event logs by timestamps (e.g. removing old or recent cases), activities (e.g. removing the irrelevant activities or directly the cases without relevant activities) and other criteria.

4.3 Model discovery

Once we have complete and structured event logs, the next step is the discovery of models. Mission models provide more information than event logs and, therefore, they are more powerful for the analysis. The models organize better the cases than the event logs, since they separate common and exceptional behaviors and reduce the number of states and transitions. Furthermore, the event logs only represent precedence relationships between the activities, whereas the models are able to represent relationships of causality and parallelism too.

As mentioned above, process mining works with different types of models, such as Petri nets, Business Process Modeling Notation, transition systems and casual nets. Additionally, there are other types of models that are common in robotics, such as agent-based models (Dudek et al. 1996), hidden Markov models (Zhu 1991) and state machines (Belta et al. 2007).

These models were analyzed in Roldán et al. (2015), which concluded that Petri nets and hidden Markov models are the most promising. Petri nets are interesting due to their ability to manage concurrency, which allows to represent the actions and interactions of multiple robots. Hidden Markov models are interesting because they can manage uncertainty, which can be a challenge in complex multi-robot missions (Rodríguez-Fernández et al. 2016). Petri nets were used in Roldán et al. (2016a) considering their strong relationship with process mining, as well as the possibility to complement them with decision trees to make predictions. The present work goes further establishing a systematic procedure, combining explored techniques with new ones, and using massive data of real multi-robot missions.

According to their objectives, there are different types of models:

- *Complete mission model* This model represents the global development of the mission. Additionally, it combines operator commands and robot actions and shows the relations among them.
- *Operator model* This model shows the strategy of the operator to command the robots during the mission. Among other things, it allows to detect deviations in the mission execution and to support the operator decisions.
- *Agent model* This model considers the actions of robots and allows to study their performance and collaboration during the mission. This study can detect mission bottlenecks, as well as inefficiencies in resource allocation.
- *Combined operator and agent model* This model is obtained by considering as case the mission and the agent. It is interesting because integrates both operator and robot actions but segregates them.

Process mining offers multiple discovery algorithms to build Petri nets through event logs (Van Dongen et al. 2009), such as Alpha miner (Van der Aalst et al. 2004), heuristic miner (Weijters and Van der Aalst 2003), Integer Linear Programming (ILP) miner (Van der Werf et al. 2008) and Inductive miner (Leemans et al. 2013). All this algorithms are implemented in the process mining framework *ProM 6* (Verbeek et al. 2010). These algorithms were studied in the context of robotics in Roldán et al. (2017a), which shown Inductive miner has the best results working on multi-robot missions. Inductive miner applies a divide-and-conquer strategy, which means it makes partitions of states and splits the log recursively until the complete model is obtained.

4.4 Model evaluation and enhancement

Once we have a set of models that explain the multi-robot mission, the next step is their evaluation and improvement.

Process mining usually evaluates the quality of models by means of four parameters: fitness, simplicity, generalization and precision (Buijs et al. 2012). Fitness is the ability of the model to explain the observed behavior. Simplicity is the ability to do it in an easy-to-understand way. Generalization measures the capability to avoid overfitting, which occurs when the models are adapted to training cases but cannot explain new samples. And precision measures the capability to avoid underfitting, which occurs when the models are able to represent many behaviors but are not useful to make predictions. Fitness and simplicity, as well as generalization and precision, are opposite concepts. Therefore, the models should reach a compromise between these pairs of parameters.

The four parameters can be obtained by different ways depending on the scenario. Sometimes, these parameters can be obtained quantitatively from the information of models and event logs (e.g. comparing the traces of models and event logs or studying the features of states and transitions). However, there are cases where some of these variable has to be studied qualitatively by the users, which could limit the use of these techniques to expert users.

The enhancement of models can be addressed by taking a step back and adjusting again their detail, which can be performed basically adding and removing states and transitions. For instance, if the model is complex or tends to underfitting, some infrequent transitions may need to be removed, whereas if the model is simple or tends to overfitting, some discarded transitions should be considered.

4.5 Time analysis

The time analysis uses the models previously obtained to compute the average, minimum and maximum times of the states and transitions. Among other things, this study can find the bottlenecks of the mission and the tasks that require more effort. Let's see some examples of time analysis:

- *Operator actuation times* This study allows to determine the times required by the operator to execute the commands. These times can be measured from when the operator opens the command window until when the command is sent. The results can be used to improve the design of commanding interfaces and the training of operators.
- *Operator decision times* This study provides information about the reaction times of operators. These times can be measured from when the mission requires an intervention until when the operator realizes it. The results can be relevant to correct problems in the design of information interfaces and the training of operators.

– *Robot execution times* This study determines the times that the robots need to execute their tasks. It allows to detect which activities have the greatest time costs and to introduce changes to improve the effectiveness of the mission.
– *Transition times* This study determines the waiting times between the different mission tasks. It allows to detect and correct both excessive waiting times and inefficient resource allocations.

4.6 Resource analysis

The resource analysis shows a different perspective of mission than the time analysis. This study allows to know which resources perform each task, to find the relationships between the agents that take part in the mission and to maximize the synergies between the resources and the tasks.

A first step in the resource analysis is the construction of resource-activity matrices, where the rows contain the resources, the columns contain the activities and each cell contains the number of times that a resource perform an activity per case. These tables show easily which resources collaborate in certain tasks or which tasks are performed by some resources.

Another step is the development of social networks (Van der Aalst and Song 2004), where the nodes are the resources implied in the mission (e.g. robots, operators and other agents), and the arcs are their relationships (e.g. collaboration in common or dependent tasks). Nevertheless, these social networks are more interesting when the number of agents is high and understanding their roles in the mission is not easy.

5 Experiments

The experiments were designed to reproduce a multiple robots-single operator scenario, in order to generate valuable data to apply the developed procedure. In this way, the techniques can be applied to study the performance of the mission, the commands of operator, the actions of agents and the interactions among them. Although the experiments were performed inside the laboratory to allow to perform multiple repetitions, they represented an outdoor disaster scenario to provide realistic data with uncertainty.

A total of thirty-six experiments were conducted: twenty with a general scenario to analyze the whole mission, and sixteen with a specific one to analyze two methods of commanding. All the missions were performed by the same operator, who monitors the mission and commands the robots by means of a simple interface. The Fig. 3 shows the layout

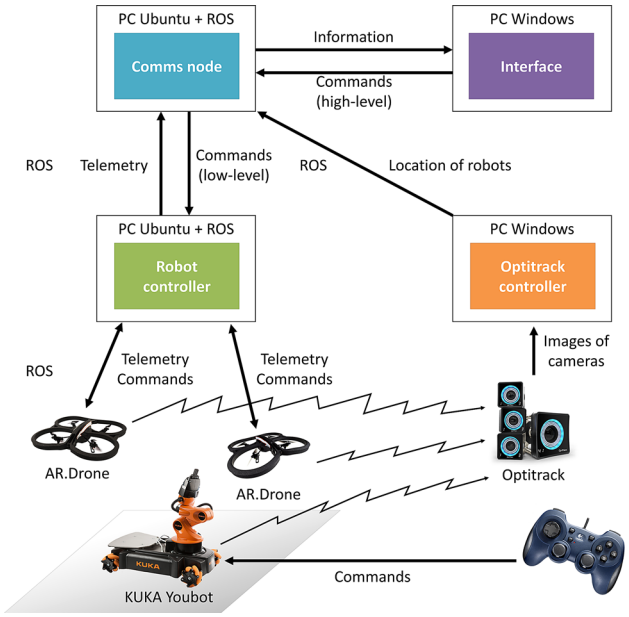


Fig. 3 Layout of experiments

of the experiments with the different components: scenario, robots, computers and interface.

5.1 Mission

The aim of the mission is to search and extinguish fires, as well as to detect and follow intruders. For this purpose, a scaled but relevant scenario of 5.35 m × 6.70 m × 5.00 m was recreated in the laboratory. As shown in Fig. 4, this scenario had three areas of interest: the first one is a fixed circle that represents the base where the robots start and finish the mission, the second one is a fixed square that represents the reservoir where the robots can load the water, and the third one changes its position depending on the mission and represents the fire that must be detected and extinguished. An ordered list with the tasks of mission and their short descriptions is shown below:

– *Begin* The robot switches on and takes-off.
– *Surveillance* The robot flies over an area at high altitude (1.6 m) with a back and forth pattern to find the potential fires.
– *Reconnaissance* The robot flies over a list of points at low altitude (0.8 m) to check the potential fires.
– *Capture* The robot flies to the reservoir, descends (0.6 m) and loads the water.
– *Release* The robot flies to the fire, ascends (1.2 m) and discharges the water.
– *Go to WP* The robot flies to a waypoint with other purpose: e.g. to leave free the way of the other robot.

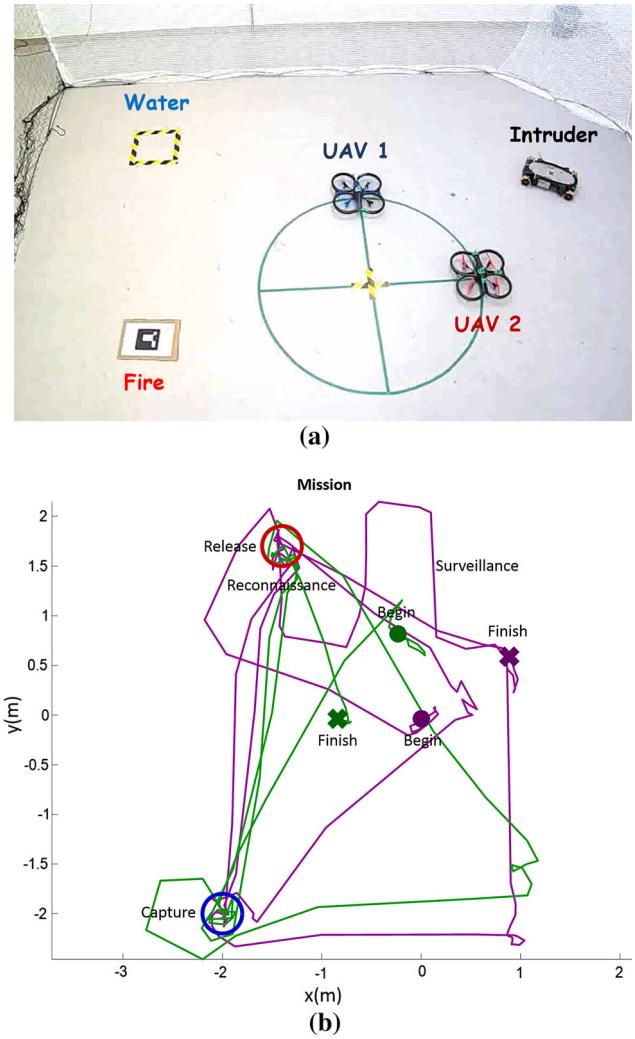


Fig. 4 Experiments: **a** scenario with robots, **b** mission execution

– *Tracking* The robot follows the suspect across the scenario at low altitude (0.8 m).
– *Finish* The robot lands and switches off.

5.2 Robots and equipments

The experiments employed two *Parrot AR.Drone 2.0* quadcopters (Krajník et al. 2011) to perform the mission, and a *KUKA Youbot* (Bischoff et al. 2011) robot to perform the role of intruder. In this way, the aerial robots were controlled by the mission operator and their information is available for the mission execution and analysis, whereas the ground robot was controlled by a person that was not involved in the mission and its information must be acquired by the aerial robots.

The Parrot AR.Drone 2.0 quadcopter is a low-cost and small-size solution for this kind of indoor flights. This small quadrotor has a size of 525 mm × 515 mm × 120 mm, a weight of 420 g and an autonomy of around 18 min with 1500 mAh batteries.

The quadrotor uses its four propellers to stabilize, change roll (going to the left or right), change pitch (going forward or backward), change yaw (rotating in place) and move vertically (ascending or descending). It is controlled by an embedded processor and can connect to WiFi networks or generate its own one. It has an IMU, a ultrasonic altimeter and two cameras (one in front and another down the robot).

The quadcopter provides a full telemetry that consists of state, position and orientation estimations based on visual odometry, angular and linear velocities and accelerations based on IMU readings, battery level, and motor voltage.

The quadrotors receive speed commands (linear and angular references) that are generated by the robot controllers. These controllers are PID regulators that use the current and goal poses to generate the adequate speed commands. Two commanding methods are used in the experiments: waypoint commands, which are directly sent to the robot controllers, and task commands, which are converted to waypoints by the communications node. As shown in Fig. 3, this node acts as an interface between robots (Ubuntu + ROS) and interface (Windows), extracting the information from telemetry and converting the high-level commands into low-level commands.

On the other hand, the *KUKA Youbot* robot has a size of 580 mm × 380 mm × 140 mm, a weight of 20 kg and a load capacity of 20 kg. It has an embedded PC, where the algorithms of guidance, navigation and control can be executed, and a router, which allows the communications with other computers to exchange telemetry and commands.

As previously mentioned, the ground robot was not part of the robot team. It was remotely controlled with a joypad by a person that was not involved in the mission. The information of this ground robot must be acquired by the aerial robots, which supposes a challenge for the mission planning, monitoring and analysis.

A motion capture system *Optitrack* was used to get the accurate position and orientation of robots (Dentler et al. 2013). This system uses a series of cameras located around the room and a set of markers attached to the robots with certain patterns, in order to capture and track the robots in a space with 6 degrees of freedom (DoF).

5.3 Interface

An intuitive operator interface was developed to receive the information of mission and send the commands to the robots. This interface can show in real-time the information of UAVs, as well as the images of their cameras. Figure 5 shows the main window, which provides the information of mission and fleet, and the command window, which allows the operator

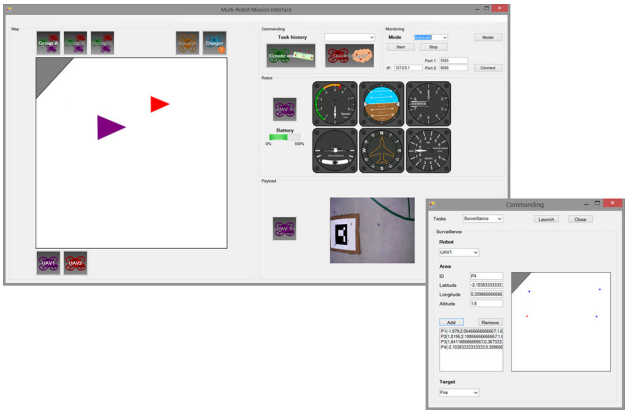


Fig. 5 Multi-robot mission interface: commanding window in front and monitoring window behind

to generate different types of commands. Additionally, this interface is able to record all the information and commands during the missions.

The main window consists of two components: a map of the scenario with the robots, where the user can select the most interesting one, and a panel that shows the information of the selected robot (orientation, speed, acceleration, battery...) and its payload (the images of the cameras).

The command window allows two commanding methods: waypoint commanding, where the operator sends the waypoints that the UAVs have to reach one by one, and task commanding, where the operator sends tasks (e.g. surveillance, reconnaissance, capture and release) and the system splits these tasks into waypoints and sends them one by one.

5.4 Integration

The robots, the motion capture system and the interface were integrated by means of Robot Operating System (ROS) (Quigley et al. 2009). Specifically, a ROS architecture with the following four nodes was developed:

- *Information node* It receives the messages from robots and motion capture system through ROS topics and sends them to the interface via TCP socket.
- *Commanding node* It receives the commands from the interface via TCP socket and sends them to the robots through a ROS topic.
- *Task node* It receives the high-level commands (tasks), splits them into medium-level commands (waypoints) and sends them to the controller node. This node only works when the mission is controlled by means of task commands.
- *Controller node* It receives the current and goal positions of the robot, generates speed commands by means of a PID controller and sends them to the robot.

6 Results

This section describes the application of the methodology of Sect. 4 to the experiments of Sect. 5. For this purpose, it follows the same scheme of Sects. 4, 6.1 studies the mission, Sect. 6.2 prepares the event log, Sect. 6.3 discovers the models, Sect. 6.4 evaluates and improves them, Sect. 6.5 analyzes it from time perspective, and Sect. 6.6 from resource perspective.

6.1 Mission study

First of all, we are going to collect some information about the mission by answering the previous questions.

- *What is the aim of the mission?* To search and extinguish fires, as well as search and follow intruders.
- *Which are the different objectives of the mission?* Fire surveillance, extinguishing of fires, search of suspects and suspect tracking.
- *Which are the events of the mission?* The launch of robots, the tasks performed by robots, the possible failures... *How are their types and features?* The events are characterized by an unique identifier, an activity, an agent that performs the activity, an agent that command it, a starting time and a finish time.
- *How many cases are available?* Thirty-six missions were performed to generate information for the analysis: twenty to analyze the mission and sixteen to compare two command methods.
- *Which are the tasks of the mission?* Begin, surveillance, reconnaissance, go to waypoint, capture, release and finish. It must be noted that in the context of this work, an objective is a result that must be achieved for completing the mission, whereas a task is a set of actions performed by a robot to achieve an objective.
- *Which agents are involved in the mission?* Two aerial robots, a ground robot and an operator. *Which are their actions and interactions?* The aerial robots cooperate to perform the mission, the operator controls the aerial robots and the ground robot is independent.

Then, we can define the objectives of the analysis. In this case, we have a multi-robot mission controlled by a human operator. Therefore, the following studies are interesting:

- Reaction, decisions and commands of operator.
- Performance of robots in tasks.
- Mission times.
- Task allocation to robots.
- Performance of interface.

6.2 Event log preparation

The second step is to collect all the available mission data and generate one or several adequate mission event logs. As shown in Fig. 6, this process addresses preparing telemetry and command logs and merging them into mission event log.

The telemetry log collects the data generated by the aerial robots during the mission. Each row contains the time in milliseconds and the following variables for each UAV: state (e.g. 1 is initializing, 2 is landed, 3 is flying, 6 is taking-off, 8 is landing...) position (X, Y and Z), orientation (roll, pitch and yaw), linear velocity, angular velocity, linear acceleration, angular acceleration, battery level, power of motors, and magnetometer measurements.

The process to generate events from telemetry consists of two steps and their results are collected by Table 2. First, we collect the events that we have observed during the planning and execution of previous missions. This kind of multi-robot missions may have events of various types, such as tasks (e.g. *Surveillance*, *Reconnaissance*...), actions (e.g. *Take-off*, *Landing*...), situations (e.g. *Detection*, *Accident*...) and locations. Some examples of events related to the location of UAVs are shown in Fig. 7: this is the case of the changes of quadrant, the transitions between center and periphery, the entrance and exit of relevant areas, and the distance between UAVs. Second, we look for the beginning and finishing conditions of these events. These conditions establish a relationship between the mission events and the telemetry variables. Some events present unique conditions (e.g. *Waiting* occurs when the speed of UAV is close to zero), whereas others may have multiple conditions (e.g. *Take-off* may start when the state changes to 6 and when the altitude starts to increase).

An issue that must be taken into account is the influence of uncertainty on the generation of events. For instance, if the poses of robots are not known with enough accuracy,

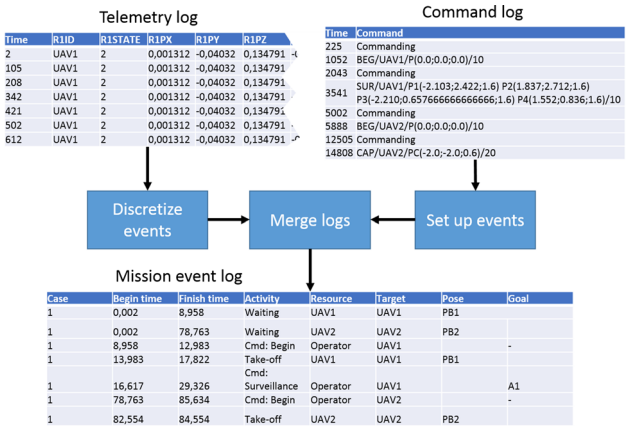


Fig. 6 Procedure for generating event logs through telemetry and commands

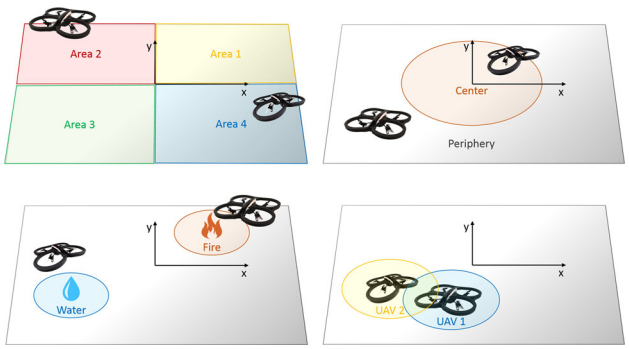


Fig. 7 Examples of events generated by the location of UAVs

Table 2 Events generated from telemetry

Activity	Begin condition	Finish condition
Take-off	$State = 6$	$Z > 0.5\,\text{m}$
Land	$State = 8$	$Z < 0.1\,\text{m}$
Waiting	$t = 0\,\text{s}$	$\|\|(V_x, V_y, V_z)\|\| > 0\,\text{m/s}$
Stopped	$\|\|(V_x, V_y)\|\| < 0.05\,\text{m/s}$	$\|\|(V_x, V_y)\|\| > 0.05\,\text{m/s}$
Capture	$\|\|(X, Y) - (X_c, Y_c)\|\| < 0.2\,\text{m}$	$\Delta t = 2\,\text{s}$
Release	$\|\|(X, Y) - (X_r, Y_r)\|\| < 0.2\,\text{m}$	$\Delta t = 2\,\text{s}$
Detection	$\|(X, Y) - (X_t, Y_t)\| < 0.2$	$\Delta t = 1\,\text{s}$
Tracking	$\|\|(X, Y) - (X_{mt}, Y_{mt})\|\| < 0.5\,\text{m}$	$\|\|(X, Y) - (X_{mt}, Y_{mt})\|\| > 0.5\,\text{m}$
Quadrant	$(X, Y) \in Q_i$	$(X, Y) \in Q_j$
Center/periphery	$\|\|(X, Y)\|\| < 1.0\,\text{m}$	$\|\|(X, Y)\|\| < 1.2\,\text{m}$
Fire area	$\|\|(X, Y) - (X_r, Y_r)\|\| < 0.5\,\text{m}$	$\|\|(X, Y) - (X_r, Y_r)\|\| > 0.75\,\text{m}$
Water area	$\|\|(X, Y) - (X_c, Y_c)\|\| < 0.5\,\text{m}$	$\|\|(X, Y) - (X_c, Y_c)\|\| > 0.75\,\text{m}$
Accident	$\|\|(V_x, V_y)\|\| = 0\,\text{m/s}$	$Z = 0\,\text{m}$

(X_c, Y_c) capture point (water), (X_r, Y_r) release point (fire), (X_t, Y_t): target (fire), (X_{mt}, Y_{mt}) mobile target (suspect)
$State$ state of robot, (X, Y) location of robot, (V_x, V_y) velocity of robot, $\|V\|$ norm of vector, Δt time interval

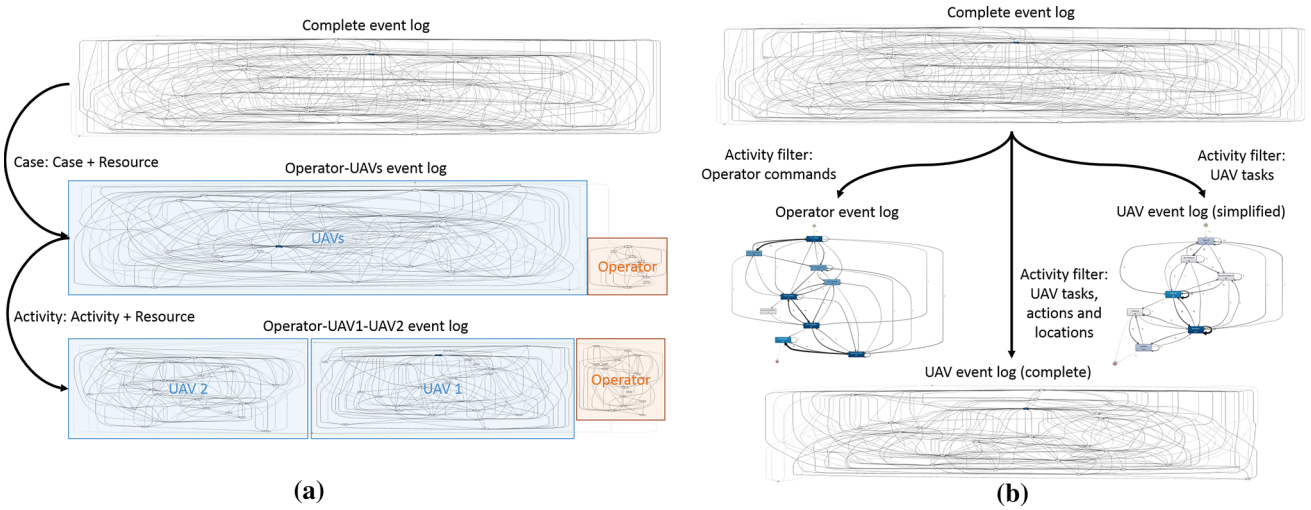


Fig. 8 Event log preparation: **a** modifying the case and activity tags to distinguish between operator and UAVs in the event log, **b** filtering activities related to operator and UAVs to obtain independent event logs of them

some real events can be missed and some false events may be considered. The consequences of these errors highly depend on the level of uncertainty, from minor deviations that are discarded in the discovery of models to critical errors that cannot be solved automatically.

The command log collects the commands sent by the operator to the robots during the mission. Each row contains the time in milliseconds and the action of operator: commanding, when the operator launches the commanding window, and the command, when the operator sends it to the robot. The commands follow this pattern: task, receiver, point list and target. For instance, the command for performing a fire surveillance is as follows: surveillance, UAV, area (represented by the list of vertices) and fire.

The commands of operator, in contrast to the telemetries of UAVs, are structured into events. Therefore, we only have to add the information of commands to some fields of events and complete the rest of them. In this case, the beginning time corresponds to the opening of the command window, whereas the finishing time to the sending of the command. Additionally, the resource role is performed by the operator and the agent role by the target UAV.

As shown in Fig. 6, each event is defined by the case (number of mission), begin time, finish time, activity (task, action, command, situation or location), resource (the agent that performs the activity), target (the agent that receives the activity), pose (the current position of the resource) and goal (the target position for the activity). We add to the conventional event logs the fields of target, which is useful to distinguish the robot that must receive and execute the operator commands, pose and goal, which are useful to study if some tasks have been successfully accomplished. In addition, we have separated the beginning and finishing times to easily estimate the duration of events.

As previously mentioned, we performed thirty-six multi-robot missions and uses twenty of them to study multiple aspects about the robots, the operator and the interface. Therefore, the complete event log of these missions has 2023 events grouped in 20 cases (a case per mission), 32 activities (an activity per operator command, as well as robot task, action and location) and 3 resources (an operator and two UAVs). This event log is loaded with *Disco* (Günther and Rozinat 2012), in order to generate graphic representations and to condition and filter the events. The complete event log contains all the events of the mission, but sometimes it is not the best resource to study the mission, since it may not be easy to understand and some sequences of events may be hidden. For these reasons, it is interesting to work with partial event logs that are focused on specific elements of the missions. Figure 8 shows some examples of event log preparation: the separation of operator and UAVs in the event log to study their actions and interactions (a), and the generation of operator and UAV event logs to study their behaviors independently (b). The different event logs generated from the complete event log are collected in Table 3 and described in the following paragraphs.

- *Complete event log (EL-A)* This event log is obtained from the mission telemetry and commands and used as the basis for later conditioning and filtering.
- *Event log with operator commands and robot tasks (EL-B)* This event log is the result of selecting the events related to operator commands and robot tasks. It allows to study how the robots execute the commands of operator.
- *Event log with operator commands (EL-C)* This event log is obtained by selecting the events related to the commands of operator. It allows to know the operator performance in mission commanding.

Table 3 Event logs considered in the work and their main features

Event log	Cases	Events	Activities	Resources
EL-A	20	2023	32	3
EL-B	20	565	14	3
EL-C	20	260	9	1
EL-D	20	349	7	2
EL-E	20	1116	11	2
EL-F	20	1763	23	2
EL-G	60	2023	32	3
EL-H	60	2023	62	3
EL-I	4	379	31	3

Table 4 Models discovered from event logs

Event log	Model	Traces	Deviations	Ratio (%)
EL-A	M-A	2063	105	94.91
EL-B	M-B	605	70	88.43
EL-C	M-C	260	49	81.15
EL-D	M-D	368	16	95.65
EL-E	M-E	1156	47	95.93
EL-F	M-F	1802	106	94.12
EL-G	M-G	2140	627	70.70
EL-H	M-H	2137	706	66.96
EL-I	M-I	57	24	42.11

- *Event log with robot tasks (EL-D)* This event log is the result of selecting the events of UAVs related to tasks and removing those related to actions and locations. It allows to easily see the evolution of mission.
- *Event log with robot tasks and actions (EL-E)* This event log is the result of selecting the events of UAVs related to tasks and actions and removing only those related to locations. It allows to see the evolution of mission with more detail. For instance, it shows not only the begin and end of tasks like surveillance, but also the intermediate results such as the detections.
- *Event log with robot tasks, actions and locations (EL-F)* This event log is the result of selecting all the events of UAVs. It allows to establish relationships between events of UAVs. For instance, the locations where the tasks are executed.
- *Complete event log with separated operator and UAV events (EL-G)* This event log is generated by considering as case the mission and resource. Therefore, it shows separately the evolutions of operator and UAVs because their activities are different.
- *Complete event log with separated operator, UAV 1 and UAV 2 events (EL-H)* This event log is generated by considering as case the mission and resource, and as activity the action and resource. Therefore, it shows separately not only the operator and the UAVs, but also UAV 1 and UAV 2.
- *Event log of accident cases (EL-I)* This event log is obtained by selecting the cases that present accidents. It is useful to investigate the causes of accidents.

6.3 Model discovery

The third step is to discover suitable models from the event logs. As mentioned above, these models not only organize better the information but also discover some relations between events. All the models generated during this step and used in the following ones are Petri nets. The size and detail of models can be adjusted by defining the percentages of activities and connections between them of the event log. The model will consider the relevant activities and paths and ignore the infrequent ones. In this context, the activities represent tasks, changes of location and other mission events, whereas the paths are sequences of two or more activities. In this work we apply the Inductive Miner implemented in the *Mine with Inductive Visual Miner* plugin of *ProM 6* (Leemans et al. 2014).

Table 4 contains the discovered models and their features. The traces are generated by playing the event log over the model and may be representations of actual events or products of the model. The deviations are paths that are contained in the event log but are not considered by the model. For instance, if the event log has twenty times a behavior (e.g. *Surveillance–Reconnaissance–Tracking*) and once another behavior (e.g.*Surveillance–Tracking*), the model could represent only the common behavior and ignore the infrequent one. In this case, if we play the event log over the model, we will get sixty-three traces (twenty one times *Surveillance*, *Reconnaissance* and *Tracking*) and one deviation (one of the *Reconnaissance* is produced by the model because it does not represent the direct path between *Surveillance* and *Tracking*. The ratio is the percentage of traces generated by the model that are actually in the event log.

6.4 Model evaluation and enhancement

The fourth step is to evaluate and improve the discovered models. For this purpose, we can apply the notions of fitness, simplicity, generalization and precision. Nevertheless, these notions are qualitative and could be difficult to quantify.

The complete mission model (M-A), which includes all the activities related to operator commands and robot tasks, actions and locations, is shown in Fig. 9. We can estimate that this model has a good fitness because the deviations suppose around 5% of traces according to Table 4, but in a quick view we can see that it is not simple. Additionally, the

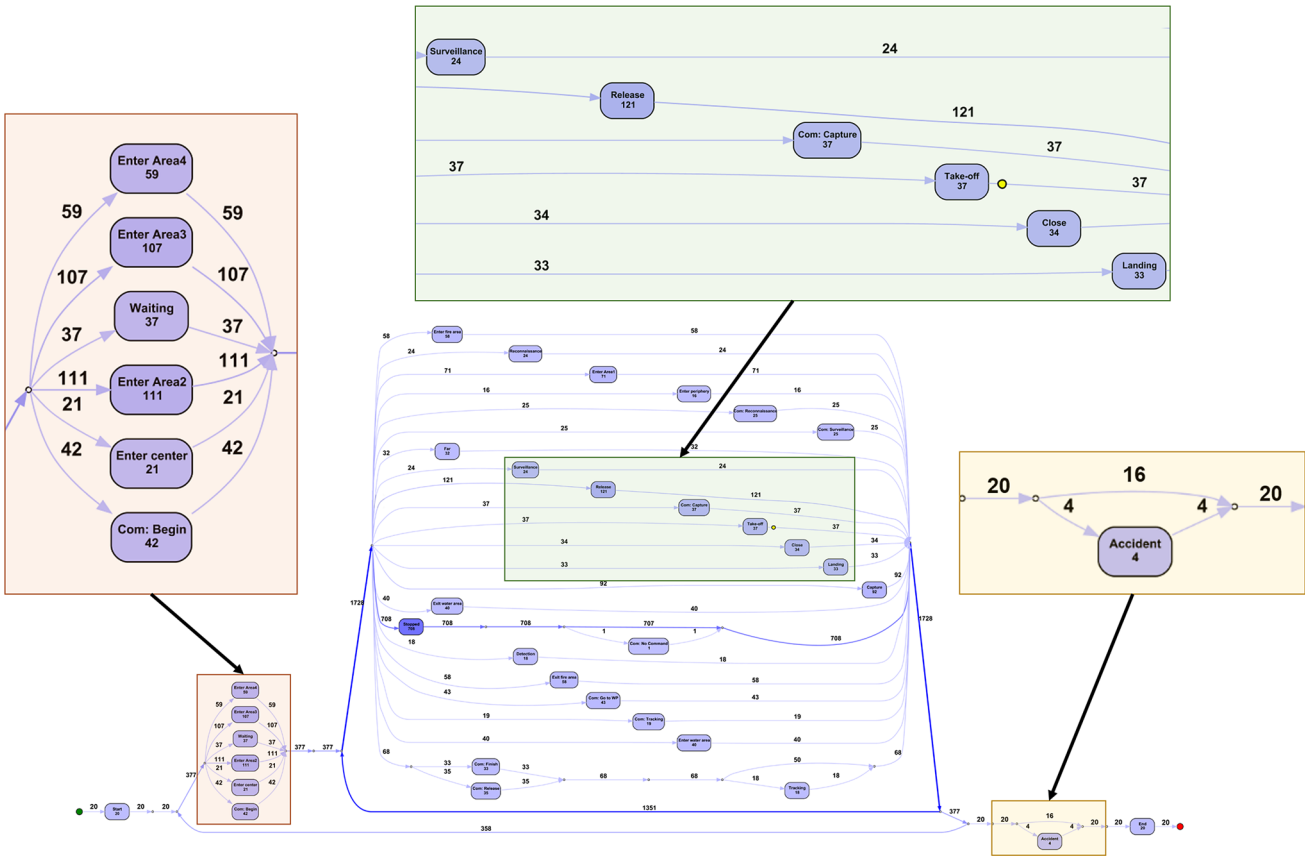


Fig. 9 Representation of complete model with some highlighted limitations: the distortion caused by location events (red), the parallelism between operator commands and robot tasks (green) and the inability to study the sequences of accident (orange). The boxes represent activities, whereas the arcs represents paths. The numbers on boxes and arcs mean the repetitions of these activities and paths (Color figure online)

model has good generalization, as it represents the majority of cases, but also bad precision, as it does not allow to make predictions. The figure highlights some phenomena that are discussed below:

- The location events distort the model because they are independent of the rest of events. The tasks or commands are not related to the quadrants or areas where the robots are located. This causes the algorithms cannot determine if the location events precede or succeed the other events.
- The operator commands and robot tasks do not appear in a defined order and the model is not able to discover the relations of causality. For instance, we know a robot performs a task after the operator commands it. However, there may be events between the command and the task: commands to other robots, tasks of other robots, situation or location events, etc.
- The lack of precision is a problem to determine the events that precede another specific event. For example, we can determine the causes of accident by studying the events that precede it. Nevertheless, the complete model does not show the events that happen before the accident.

In conclusion, this model allows to study the frequencies and durations of tasks, but it is not useful to determine the evolution of the mission. For this reason, we discovered a diverse set of models both general of mission and focused on operator or robots.

The operator model also presents an interesting behavior. As shown in Table 5, the ratio between deviations and traces depends on the percentage of paths (M-C-1 includes 100%, M-C-2 includes 90% and so on). These paths are removed ordered by their frequencies, from the less to the most common. Therefore, if we look for the operator model with the best fitness, we should consider the 90% of paths instead of all of them.

6.5 Time analysis

The fifth step is to analyze the missions from the perspective of time. The duration of mission is between 3 min 28 s and 10 min 49 s with a mean value of 5 min 36 s and a median value of 5 min. Therefore, the variability of the mission is high: e.g. the longest mission is three times longer than the

Table 5 Operator models and main features

Event log	Model	Traces	Deviations	Ratio (%)
EL-C	M-C-1	260	49	81.15
EL-C	M-C-0.9	259	23	91.12
EL-C	M-C-0.8	238	32	86.55
EL-C	M-C-0.7	241	34	85.89
EL-C	M-C-0.6	165	105	36.36
EL-C	M-C-0.5	167	105	37.13

Table 6 Duration of tasks

Task	Mean time (s)	Min time (s)	Max time (s)
Take-off	3.63	2.55	5.02
Surveillance	18.12	1.02	62.72
Reconnaissance	19.01	1.08	66.60
Capture	2	2	2
Release	2	2	2
Tracking	114.97	69.25	175.43
Landing	1.05	0.29	2.23

shortest one. Tables 6 and 7 show the duration of robot tasks and operator commands respectively.

On the one hand, the longest task is *Tracking* with a mean duration of 115 s, because it starts when the mobile target is detected and it finishes when the mission is completed or the UAV is required for another task. The next tasks in terms of duration are *Reconnaissance* and *Surveillance* with 19 and 18 s respectively. In these tasks there are remarkable differences between the shortest and longest executions. This behavior can be due to *Surveillance* and *Reconnaissance* strongly depend on the area size and the number of points respectively.

On the other hand, the commands that require more effort of the operator are *Reconnaissance* and *Surveillance* with mean times of 17.59 and 16.23 s. Additionally, there are other commands that sometimes take longer times, such as *Begin* (39.50 s), *Finish* (46.45 s), *Capture* (28.84 s) and *Release* (21.37 s). These extreme cases occur when the operator have doubts about the situation of mission or make mistakes during the commanding.

Finally, the Fig. 10 shows the occupancy of multiple areas during the mission. The quadrants 2 and 4 are more transited than the quadrants 1 and 3 (30 vs. 20%), as well as the center is about twice busier than the periphery (65 vs. 35%).

6.6 Resource analysis

The sixth step is to analyze the missions from the perspective of resources. The event log shows the UAV 1 performs 1340 events (63.51%), the UAV 2 develops 510 events (24.17%)

Table 7 Duration of commands

Task	Mean time (s)	Min time (s)	Max time (s)
Begin	9.76	3.17	39.50
Surveillance	16.23	12.16	22.50
Reconnaissance	17.59	9.04	58.07
Go to WP	9.47	5.04	16.84
Capture	7.45	3.51	28.84
Release	6.21	2.61	21.37
Tracking	7.49	3.79	18.32
Finish	6.16	3.27	46.45

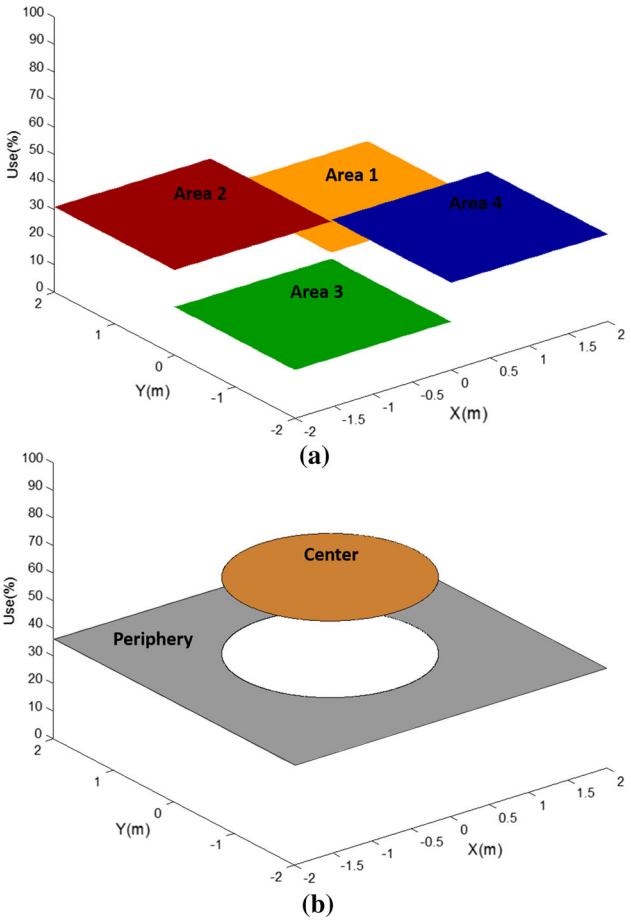


Fig. 10 Usage of areas during the mission

and the operator executes 260 events (12.32%). A first study could conclude that the operator takes advantage of UAV 1 but does not use enough UAV 2. Nevertheless, the study of operator commands disclose that 131 (50.38%) are sent to UAV 2 and 128 (49.23%) are sent to UAV 1. Therefore, the difference is not in the number but in the type of tasks: UAV 1 performs more times *Surveillance* than UAV 2 (15 vs 9), whereas UAV 2 performs more times *Reconnaissance* than UAV 1 (14 vs. 11). This fact explains the different number of

events: e.g. quadrant changes (183 vs 165), center-periphery transitions (39 vs. 12)...

The waiting times for UAVs, which are measured from the beginning of mission to the take-off of UAVs, have a mean value 94.67 s and suppose a 36.24% of the whole mission. In addition, the most common event in the mission is *Stop* (707 repetitions), which occurs when a UAV finishes a task and waits for a new command. The stop events have a mean duration of 45.97 s per mission, which supposes a 18.07% of its duration. If we study the *Wait* and *Stop* events according to the UAVs, we find the UAV 1 is used 66.57% of time, whereas the UAV 2 is only used 42.07% of time.

7 Discussion

First of all, it must be remarked that the experiments and the analysis have been designed to allow the comparison among the results of process mining and visual inspection of missions. Nevertheless, the results show that process mining is a powerful tool that can extract some information that is not evident in manual analysis. Logically, if the missions are more complex, visual inspection will be more difficult, whereas process mining will be more useful. Furthermore, the application of process mining allows to automate a considerable part of the analysis, which may imply a reduction in the consumption of time and resources, since it avoids the manual analysis of huge amounts of data.

This section discusses the results of the analysis developed in the previous one. The conclusions are organized as follows: Sect. 7.1 collects the discovered information about the mission, whereas Sect. 7.2 contains the conclusions about the operator performance.

7.1 Mission

The most evident problem in the missions is the use of resources. The operator usually launches the first UAV in the first 30 s and the second one after 1 min 30 s. Therefore, this excessive deployment time impedes to take advantage of the multi-robot system in the beginning of the mission. A possible solution of this problem is an autonomous system for the deployment of fleet, which could manage the simultaneous take-off of both UAVs and their placement in the adequate areas. Furthermore, the UAVs are stopped many times during the mission and this is an important waste of time. This happens when one of the UAVs finishes its task, but the operator is paying attention to the another. In this case, a potential solution is to show an alert of free UAV in the interface.

Another relevant problem are the accidents, which can be studied through the accident event log and model. As it can be seen in Fig. 11, the accidents are related to situations where the UAVs get close in fire or water areas. A possible strategy to prevent these accidents is to automatically limit the distance between the UAVs, as well as the number of UAVs in fire or water areas.

7.2 Operator

As previously mentioned, we performed thirty-six multi-robot missions and uses sixteen of them to study the methods of commanding. All the missions were performed by the

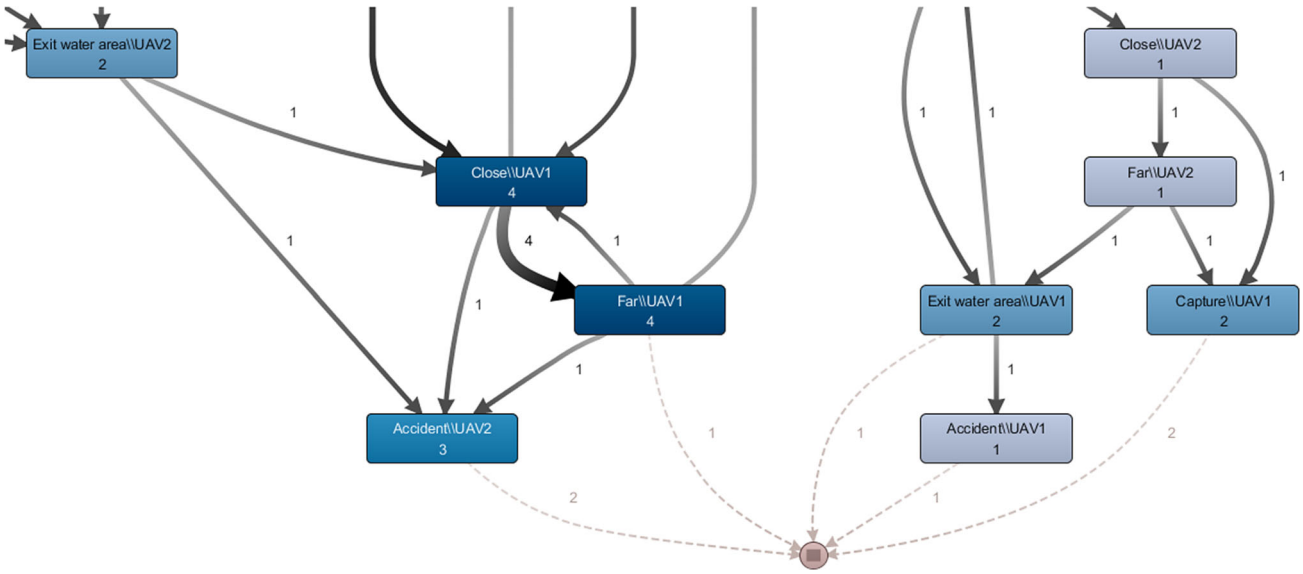


Fig. 11 Fragment of the event log with accident cases (EL-I) that shows the events before the accidents

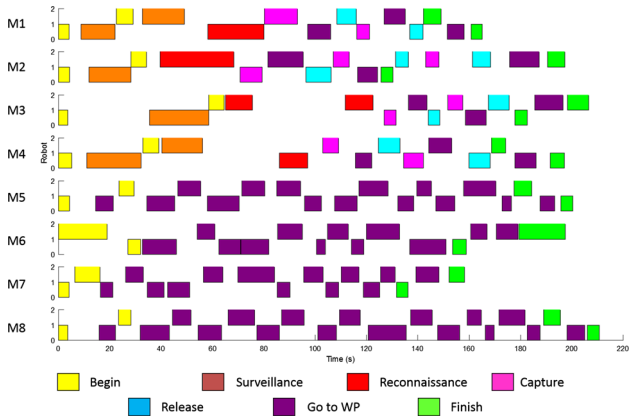


Fig. 12 Commands from operator to robots in eight missions: from M1 to M4, task commands, and from M5 to M8, waypoint commands

Table 8 Comparison among task and waypoint commands

Parameter	Task Com.	WP Com.
Mission time (mean)	195.7 s	184.7 s
Mission time (std)	17.3	20.8
Number of commands (mean)	14.3	18.2
Number of commands (std)	0.5	2.2
Commanding errors (mean)	0.0	0.25
Commanding errors (std)	0.0	0.5
Commanding time (abs)	129.3 s	140.2 s
Commanding time (rel)	66.35%	75.92%
Monitoring time (abs)	66.5 s	44.5 s
Monitoring time (rel)	33.65%	24.08%

same operator: half of them by using task commands, whereas the rest by sending waypoint commands. The Fig. 12 shows the evolution of commands in eight of these sixteen missions.

Additionally, the Table 8 contains some average measures of the missions. As it can be seen, the mission times were lower with waypoint commanding than with task commanding, but this difference is not significant according to an ANOVA test with $alpha = 0.05$. This fact is because the operator has closer control of the UAVs. For instance, he can send directly the points where he believes there is a fire, instead of sending the whole area and waiting until the UAV is flying over these points.

Nevertheless, the rest of variables show that task commanding is better than waypoint commanding. The operator has to use less commands (14.3 vs. 18.2 commands per mission) and makes less mistakes (0.0 vs. 0.25 errors per mission): e.g. to send an incorrect command or to send a correct command to an incorrect robot. The ANOVA tests with $alpha = 0.05$ show the difference in the number of commands is significant, whereas the difference in the number of errors is not significant. Furthermore, the operator spends less time in commanding (66.35 vs 75.92%, an advantage of 12.61%) and, therefore, he has more time to understand the mission and make decisions.

The operator workload is still high with task commands, because the scenario is fast and changing, which requires a considerable effort to the operator. The Table 7 shows that the operator spends more time entering surveillance and reconnaissance commands than the rest. This fact can be explained by the commanding interface: the points must be entered by clicking in the map and confirmed by pressing a button. These commands would be faster and the operator would save more time if the areas are defined by clicking and dragging and the point list is confirmed after introduced.

8 Conclusions

This paper explores the application of process mining to multi-robot missions. It not only proposes a systematic procedure to use this discipline in the context of robotics, but also applies this procedure in a set of realistic multi-robot missions. This practical experience provides more details to know how to apply the theoretical procedures in other scenarios.

The main contributions of the methodology are related to the generation of event logs and the discovery of models. The first one allows to apply these techniques to a wide range of missions, even if the available data is not complete or structured. The second one allows to generate the adequate models to obtain the desired information. For instance, it is easier to obtain the information of operator from the model of operator than from the complete model.

The main conclusion of the work is that process mining is a powerful tool for the analysis of multi-robot missions. If we use this tool adequately, we will make useful conclusions, but if we do not use it properly, we can lose relevant information. Some important issues discovered in the multi-robot missions were the causes of accident, the inefficiencies in the management of resources and the limitations of the commanding interface. In fact, the analysis performed in this work was the starting point for the design of an adaptive and immersive interface in a later one (Roldán et al. 2017b).

This paper opens the way to a future extensive use of process mining in multi-robot missions. These techniques will contribute to improve their efficiency, to reduce their risks and to detect and solve their problems. The set of missions where they can be applied is huge and includes missions with diverse domains, robots and operators.

The main challenge for this future application of process mining in robotics is related to the automation of this methodology. At this moment there are steps that are fully automated (e.g. event log preparation and model generation), whereas

there are others that needs an expert user (e.g. model evaluation and enhancement). Future works will study techniques to solve these challenges and allow an autonomous analysis.

Acknowledgements This work is framed on SAVIER (Situational Awareness Virtual EnviRonment) Project, which is both supported and funded by Airbus Defence & Space. The research leading to these results has received funding from the RoboCity2030-III-CM project (Robótica aplicada a la mejora de la calidad de vida de los ciudadanos. Fase III; S2013/MIT-2748), funded by Programas de Actividades I+D en la Comunidad de Madrid and cofunded by Structural Funds of the EU, and from the DPI2014-56985-R project (Protección robotizada de infraestructuras críticas) funded by the Ministerio de Economía y Competitividad de Gobierno de España. The experiments were performed in the facilities of Interdisciplinary Centre for Security, Reliability and Trust (SnT) of the University of Luxembourg (uni.lu).

References

Beer, J., Fisk, A. D., & Rogers, W. A. (2014). Toward a framework for levels of robot autonomy in human-robot interaction. *Journal of Human-Robot Interaction*, *3*(2), 74.

Belta, C., Bicchi, A., Egerstedt, M., Frazzoli, E., Klavins, E., & Pappas, G. J. (2007). Symbolic planning and control of robot motion [grand challenges of robotics]. *IEEE Robotics & Automation Magazine*, *14*(1), 61–70.

Bischoff, R., Huggenberger, U., and Prassler, E. (2011). Kuka youbot—A mobile manipulator for research and education. In *2011 IEEE international conference on robotics and automation (ICRA)* (pp. 1–4). IEEE.

Buijs, J. C., Van Dongen, B. F., & van der Aalst, W. M. (2012). On the role of fitness, precision, generalization and simplicity in process discovery. In *OTM confederated international conferences "on the move to meaningful internet systems"* (pp. 305–322). Springer.

Cantelli, L., Mangiameli, M., Melita, C. D., & Muscato, G. (2013). UAV/UGV cooperation for surveying operations in humanitarian demining. In *2013 IEEE international symposium on safety, security, and rescue robotics (SSRR)* (pp 1–6). IEEE.

Cummings, M. L., & Mitchell, P. J. (2008). Predicting controller capacity in supervisory control of multiple UAVS. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, *38*(2), 451–460.

De Cubber, G., Doroftei, D., Serrano, D., Chintamani, K., Sabino, R., & Ourevitch, S. (2013). The EU-ICARUS project: Developing assistive robotic tools for search and rescue operations. In *2013 IEEE international symposium on safety, security, and rescue robotics (SSRR)* (pp. 1–4). IEEE.

Dentler, J., Kannan, S., Mendez, M. A. O., & Voos, H. (2016). A real-time model predictive position control with collision avoidance for commercial low-cost quadrotors. In *2016 IEEE conference on control applications (CCA)* (pp. 519–525). IEEE.

Dijkman, R. M., Dumas, M., & Ouyang, C. (2008). Semantics and analysis of business process models in BPMN. *Information and Software Technology*, *50*(12), 1281–1294.

Dudek, G., Jenkin, M. R., Milios, E., & Wilkes, D. (1996). A taxonomy for multi-agent robotics. *Autonomous Robots*, *3*(4), 375–397.

Garzón, M., Valente, J., Roldán, J. J., Cancar, L., Barrientos, A., & Del Cerro, J. (2016). A multirobot system for distributed area coverage and signal searching in large outdoor scenarios. *Journal of Field Robotics*, *33*(8), 1087–1106.

Garzón, M., Valente, J., Zapata, D., & Barrientos, A. (2013). An aerial-ground robotic system for navigation and obstacle mapping in large outdoor areas. *Sensors*, *13*(1), 1247–1267.

Günther, C. W., & Rozinat, A. (2012). Disco: Discover your processes. *BPM (Demos)*, *940*, 40–44.

Janchiv, A., Batsaikhan, D., Hwan Kim, G., & Lee, S.-G. (2011). Complete coverage path planning for multi-robots based on. In *2011 11th international conference on control, automation and systems (ICCAS)* (pp. 824–827). IEEE.

Jans, M., van der Werf, J. M., Lybaert, N., & Vanhoof, K. (2011). A business process mining application for internal transaction fraud mitigation. *Expert Systems with Applications*, *38*(10), 13351–13359.

Kapoutsis, A. C., Chatzichristofis, S. A., Doitsidis, L., de Sousa, J. B., Pinto, J., Braga, J., et al. (2016). Real-time adaptive multi-robot exploration with application to underwater map construction. *Autonomous Robots*, *40*(6), 987–1015.

Krajník, T., Vonásek, V., Fišer, D., & Faigl, J. (2011). AR-drone as a platform for robotic research and education. In *International conference on research and education in robotics* (pp. 172–186). Springer.

Kruijff-Korbayová, I., Colas, F., Gianni, M., Pirri, F., Greeff, J., Hindriks, K., et al. (2015). Tradr project: Long-term human-robot teaming for robot assisted disaster response. *KI-Künstliche Intelligenz*, *29*(2), 193–201.

Leemans, S. J., Fahland, D., & van der Aalst, W. M. (2013). Discovering block-structured process models from event logs-a constructive approach. In *International conference on applications and theory of Petri nets and concurrency* (pp. 311–329). Springer.

Leemans, S. J., Fahland, D., & van der Aalst, W. M. (2014). Process and deviation exploration with inductive visual miner. *BPM Demo Sessions*, *1295*, 46.

Lesire, C., Infantes, G., Gateau, T., & Barbier, M. (2016). A distributed architecture for supervision of autonomous multi-robot missions. *Autonomous Robots*, *40*(7), 1343–1362.

Lindemuth, M., Murphy, R., Steimle, E., Armitage, W., Dreger, K., Elliot, T., et al. (2011). Sea robot-assisted inspection. *IEEE Robotics & Automation Magazine*, *18*(2), 96–107.

Mans, R., Schonenberg, M., Song, M., van der Aalst, W. M., & Bakker, P. J. (2008). Application of process mining in healthcare—A case study in a dutch hospital. In *International joint conference on biomedical engineering systems and technologies* (pp. 425–438). Springer.

Nair, R., Tambe, M., Marsella, S., & Raines, T. (2004). Automated assistants for analyzing team behaviors. *Autonomous Agents and Multi-Agent Systems*, *8*(1), 69–111.

Nestmeyer, T., Giordano, P. R., Bülthoff, H. H., & Franchi, A. (2017). Decentralized simultaneous multi-target exploration using a connected network of multiple robots. *Autonomous Robots*, *41*(4), 989–1011.

Poggi, N., Muthusamy, V., Carrera, D., & Khalaf, R. (2013). Business process mining from e-commerce web logs. In *Business process management* (pp. 65–80). Berlin, Heidelberg: Springer.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., et al. (2009). Ros: An open-source robot operating system. In *ICRA workshop on open source software* (Vol. 3, p. 5). Kobe, Japan.

Rodríguez-Fernández, V., Gonzalez-Pardo, A., & Camacho, D. (2016). A method for building predictive HSMMS in interactive environments. In *2016 IEEE congress on evolutionary computation (CEC)* (pp. 3146–3153). IEEE.

Roldán, J. J., del Cerro, J., & Barrientos, A. (2015). A proposal of methodology for multi-UAV mission modeling. In *2015 23th mediterranean conference on control and automation (MED)* (pp. 1–7). IEEE.

Roldán, J. J., del Cerro, J., & Barrientos, A. (2017a). Using process mining to model multi-UAV missions through the experience. *IEEE Intelligent Systems*, *32*(4), 40–47.

Roldán, J. J., Garcia-Aunon, P., del Cerro, J., & Barrientos, A. (2016a). Determining mission evolution through UAV telemetry by using decision trees. In *2016 IEEE international conference on systems, man and cybernetics (SMC)* (pp. 108–103). IEEE.

Roldán, J. J., Garcia-Aunon, P., Garzón, M., de León, J., del Cerro, J., & Barrientos, A. (2016b). Heterogeneous multi-robot system for mapping environmental variables of greenhouses. *Sensors*, *16*(7), 1018.

Roldán, J. J., Lansac, B., del Cerro, J., & Barrientos, A. (2016c). A proposal of multi-UAV mission coordination and control architecture. In *Robot 2015: Second Iberian robotics conference* (pp. 597–608). Springer.

Roldán, J. J., Peña-Tapia, E., Martín-Barrio, A., Olivares-Méndez, M. A., Del Cerro, J., & Barrientos, A. (2017b). Multi-robot interfaces and operator situational awareness: Study of the impact of immersion and prediction. *Sensors*, *17*(8), 1720.

Rozinat, A., Zickler, S., Veloso, M., van der Aalst, W. M., & McMillen, C. (2009). Analyzing multi-agent activity logs using process mining techniques. *Distributed Autonomous Robotic Systems*, *8*, 251–260.

Ruff, H. A., Narayanan, S., & Draper, M. H. (2002). Human interaction with levels of automation and decision-aid fidelity in the supervisory control of multiple simulated unmanned air vehicles. *Presence: Teleoperators and Virtual Environments*, *11*(4), 335–351.

Sheridan, T. B., & Verplank, W. L. (1978). *Human and computer control of undersea teleoperators*. DTIC Document: Technical report.

Tsokas, N. A., & Kyriakopoulos, K. J. (2012). Multi-robot multiple hypothesis tracking for pedestrian tracking. *Autonomous Robots*, *32*(1), 63–79.

Tully, S., Kantor, G., & Choset, H. (2010). Leap-frog path design for multi-robot cooperative localization. In *Field and service robotics* (pp. 307–317). Berlin, Heidelberg: Springer.

Valente, J., Sanz, D., Barrientos, A., del Cerro, J., Ribeiro, Á., & Rossi, C. (2011). An air-ground wireless sensor network for crop monitoring. *Sensors*, *11*(6), 6088–6108.

Van der Aalst, W. M. (1998). The application of petri nets to workflow management. *Journal of Circuits, Systems, and Computers*, *8*(01), 21–66.

Van der Aalst, W. (2011). *Process mining: Discovery, conformance and enhancement of business processes*. Berlin: Springer.

Van Der Aalst, W., Adriansyah, A., & Van Dongen, B. (2011). Causal nets: A modeling language tailored towards process discovery. In *International conference on concurrency theory* (pp. 28–42). Springer.

Van Dongen, B., Alves de Medeiros, A., & Wen, L. (2009). Process mining: Overview and outlook of petri net discovery algorithms. *In transactions on petri nets and other models of concurrency II*, 225–242.

van der Aalst, W. M., Reijers, H. A., Weijters, A. J., van Dongen, B. F., De Medeiros, A. A., Song, M., et al. (2007). Business process mining: An industrial application. *Information Systems*, *32*(5), 713–732.

van der Aalst, W. M., Rubin, V., van Dongen, B. F., Kindler, E., & Günther, C. W. (2006). Process mining: A two-step approach using transition systems and regions. BPM Center Report BPM-06-30, BPMcenter.org, 6.

Van der Aalst, W. M. & Song, M. (2004). Mining social networks: Uncovering interaction patterns in business processes. In *International conference on business process management* (pp. 244–260). Springer.

Van der Werf, J. M. E., van Dongen, B. F., Hurkens, C. A., and Serebrenik, A. (2008). Process discovery using integer linear programming. In *International conference on applications and theory of petri nets* (pp. 368–387). Springer.

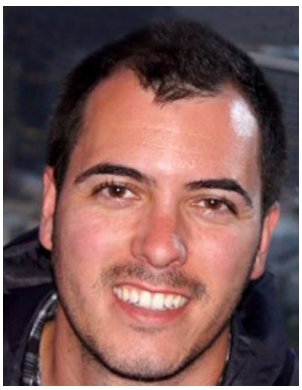
Verbeek, H., Buijs, J., Van Dongen, B., & van der Aalst, W. M. (2010). Prom 6: The process mining toolkit. In *Proceedings of BPM demonstration track* (Vol. 615, pp. 34–39).

Van der Aalst, W., Weijters, T., & Maruster, L. (2004). Workflow mining: Discovering process models from event logs. *IEEE Transactions on Knowledge and Data Engineering*, *16*(9), 1128–1142.

Weijters, A. J., & Van der Aalst, W. M. (2003). Rediscovering workflow models from event-based data using little thumb. *Integrated Computer-Aided Engineering*, *10*(2), 151–162.

Zhu, Q. (1991). Hidden markov model for dynamic obstacle avoidance of mobile robot navigation. *IEEE Transactions on Robotics and Automation*, *7*(3), 390–397.

Juan Jesús Roldán was born in 1988 in Almería (Spain). He studied B.Sc. + M.Sc. on Industrial Engineering with specialization in Automation and Electronics (2006–2012) and M.Sc. on Automation and Robotics (2013–2014) in Technical University of Madrid (UPM). Currently, he is Ph.D. candidate of Centre for Automation and Robotics (CAR, UPM-CSIC) and Airbus Defence & Space, whose research is focused on the multi-robot coordination and control interfaces.



Dr. Miguel A. Olivares-Méndez received the Diploma in Computer Science Engineering in 2006 (University of Malaga, UMA, Spain), and received the M.Sc. degree in Robotics and Automation and Ph.D. degree in Robotics and Automation (Technical University of Madrid, UPM, Spain), in 2009 and 2013, respectively. He got the Best Ph.D. Thesis award of 2013 by the European Society for Fuzzy Logic and Technology (EUSFLAT). In May 2013 he joined the Interdisciplinary Center for Security Reliability and Trust (SnT) at the University of Luxembourg (Uni.Lu), as Associate Researcher in the Automation & Robotics Research Group. In December 2016 he becomes Research Scientist and major responsible of the research activities on mobiles robotics in the Automation & Robotics Research Group at the SnT-University of Luxembourg. His main research interests are on UAVs, computer vision, sensor fusion, vision-based control, soft-computing control techniques and robotics. He has published over 60 book chapters, scientific journals and conferences papers.

Dr. Jaime del Cerro received his Ph.D. degree in Robotics and Computer vision at Technical University of Madrid (UPM) in 2007. Currently, he is an assistant Professor of Robotics, basic control subjects and Guidance, Navigation and Control of autonomous robots in this university. He has participated in several European Framework projects and projects funded by ESA (European Space Agency) and EDA (European Defense Agency), as well as commercial agreements with relevant national companies

Dr. Antonio Barrientos received the M.Sc. Engineer degree in Automation and Electronics from the Technical University of Madrid (UPM) in 1982, and the Ph.D. in Robotics from the same University in 1986. In 2002 he obtained the M.Sc. Degree in Biomedical Engineering by National Distance Education University (UNED). Since 1988 he is Professor on robotics, computers and control engineering at the Technical University of Madrid. He has worked in robotics for more than 30 years, developing industrial and service robots for different areas. Antonio Barrientos is currently the head of the Robotics and Cybernetics Research Group (RobCib) of the Centre for Automation and Robotics (CAR, UPM-CSIC).