POLITÉCNICA

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES
UNIVERSIDAD POLITÉCNICA DE MADRID

José Gutiérrez Abascal, 2. 28006 Madrid
Tel.: 91 336 3060
info.industriales@upm.es

www.industriales.upm.es

Juan Jesús Roldán Gómez

05 TESIS DOCTORAL

INDUSTRIALES

TESIS DOCTORAL

ADAPTIVE AND IMMERSIVE INTERFACES TO IMPROVE SITUATIONAL AWARENESS IN MULTI-ROBOT MISSIONS

TESIS DOCTORAL
PARA LA OBTENCIÓN DEL
TÍTULO DE **DOCTOR**

JULIO 2018

Juan Jesús Roldán Gómez

DIRECTOR DE LA TESIS DOCTORAL:
Antonio Barrientos Cruz
Jaime del Cerro Giner

POLITÉCNICA

TECHNICAL UNIVERSITY OF MADRID

DOCTORAL THESIS

Adaptive and Immersive Interfaces to Improve Situational Awareness in Multi-Robot Missions

Author:

Juan Jesús Roldán Gómez

Advisors:

Dr. Jaime del Cerro
Dr. Antonio Barrientos

A thesis submitted for the program of PhD in Automation and Robotics

Robotics and Cybernetics Research Group (RobCib)
Centre for Automation and Robotics (UPM-CSIC)

May 11, 2018

Defensa de la tesis

Título:

Adaptive and Immersive Interfaces to Improve Situational Awareness in Multi-Robot Missions

Autor:

MSc. Juan Jesús Roldán Gómez

Directores:

PhD. Jaime del Cerro Giner

PhD. Antonio Barrientos Cruz

Tribunal nombrado por el Mgfco. y Excmo. Sr. Rector de la Universidad Politécnica de Madrid el día 5 de Junio de 2018:

Presidente:

PhD. Jose María Sebastian Zuñiga

Secretario:

PhD. Miguel Hernando Gutiérrez

Vocal:

PhD. Oscar Reinoso García

Vocal:

PhD. Miguel Ángel Salichs Sánchez-Caballero

Vocal:

PhD. João Ricardo Valente Pereira

Suplente:

PhD. Ramón Barber Castaño

Suplente:

PhD. José Javier Rainer Granados

Fecha de la defensa:**Calificación de la tesis:**

El presidente:

El secretario:

Los vocales:

“Para la libertad sangro, lucho, pervivo.
Para la libertad, mis ojos y mis manos,
como un árbol carnal, generoso y cautivo,
doy a los cirujanos.

Para la libertad siento más corazones
que arenas en mi pecho: dan espumas mis venas,
y entro en los hospitales, y entro en los algodones
como en las azucenas.

Para la libertad me desprendo a balazos
de los que han revolcado su estatua por el lodo.
Y me desprendo a golpes de mis pies, de mis brazos,
de mi casa, de todo.

Porque donde unas cuencas vacías amanezcan,
ella pondrá dos piedras de futura mirada
y hará que nuevos brazos y nuevas piernas crezcan
en la carne talada.

Retoñarán aladas de savia sin otoño
reliquias de mi cuerpo que pierdo en cada herida.
Porque soy como el árbol talado, que retoño:
porque aún tengo la vida.”

Miguel Hernandez

Acknowledgements

Una vez alcanzado el final de este camino, echo la vista atrás y veo cinco razones que me han llevado hasta aquí:

En primer lugar, el apoyo de la sociedad. Mi carrera ha transcurrido por un colegio público, un instituto público y una universidad pública. De otra manera es probable que me hubiera quedado por el camino. Me gustaría agradecer no sólo a los profesionales de la educación pública, sino a todos los que la sostenemos con nuestro trabajo. No dejemos que nos la arrebaten.

En segundo lugar, mi familia y amigos. En especial, a mis padres, que siempre han visto la formación como algo esencial, la han puesto por delante de todo y no han escatimado en recursos. También a Silvia y Nana, que han sufrido tanto mi falta de tiempo como mi abundancia de preocupaciones. En cierto modo he contraído una deuda que nunca podré pagar, aunque mejor con ellos que con un maldito banco.

En tercer lugar, mis compañeros de aventura:

- A Antonio y Jaime, por darme la oportunidad de protagonizar este viaje y ofrecerme su guía.
- A los que fueron por delante, João, David y, sobre todo, Mario, que me han enseñado una buena parte de lo que sé.
- A los que vienen por detrás, Pablo, Jorge, David, Andrés y Silvia, que me han ayudado a afrontar los problemas que han ido surgiendo en el día a día, y especialmente a Elena, que ha aportado su tiempo, trabajo e ideas para llevar a cabo los experimentos finales.
- A la gente del proyecto SAVIER de Airbus Defence & Space, tanto por la oportunidad de participar en el proyecto como por la experiencia de formar parte del equipo.
- To the people of the SnT of Luxembourg, specially Miguel and Amin, who made me feel like at home during my research stay.
- Y a mis estudiantes de TFG y TFM, que han hecho realidad algunas de mis ideas: Carmina, Bruno, Inmaculada, Alberto, Fran, Jorge, Félix, María, Santiago, Juan, Elena Peña, Víctor, Kala, Elena Crespo, Ángel, Silvia, Diego, Marina, Guillermo Rodríguez, Javier y Guillermo Robledo.

En cuarto lugar, el azar. Ese conjunto de decisiones, unas veces libres y otras condicionadas, y sucesos aparentemente aleatorios que me han traído hasta aquí.

Y por último, algunas ideas que trato de explicar a partir de aquí...

Graphical Abstract

Chapter 1: A game of drones

Chapter 2: Deconstructing mission

Chapter 3: There's something about Operator

Chapter 4: The lord of the UAVs

Abstract

Missions involving multiple robots have experienced an unprecedented growth over the last years. The explanation behind this fact may be that using a robot fleet is more effective, efficient, flexible and fault tolerant than employing a single robot for the task. However, this kind of missions entail a set of challenges related to human factors. Among these challenges, the most remarkable include managing workload, keeping situational awareness and managing stressful situations.

This PhD thesis aims at developing a new generation of interfaces that allow single operators to control robot fleets. For this purpose, data mining and machine learning tools are applied to discover relevant information and provide it to operators, whereas virtual and augmented reality technologies are used to create immersive interfaces that improve their situational awareness.

The thesis can be split into four blocks:

- **Mission modeling:** The objective is to generate mission models able to show what is happening at any moment and select relevant information. In this thesis, the use of process mining is proposed as a way to automatically model the missions through the data generated by them. Specifically, the use of Petri nets is proposed to determine the state of the mission, and decision trees are employed to predict its evolution. Among all the discovery algorithms of process mining that have been tested, the inductive miner provided the best results in the context of multi-robot missions.
- **Operator modeling:** The objective is to generate operator models to enable a prediction of their behavior during the mission, in order to provide them with the most adequate information at any moment. The classification of operators is proposed according to their mission control strategies, and it is followed by the prediction of their preferences, taking into account the data of previous missions. The first step is performed by applying clustering algorithms (Partition Around Medoids provided the best classifications), whereas the second one is developed by using probability distributions.
- **Adaptive interfaces:** Once the mission and operator models are obtained, the objective of this block is to discover the relevant information from the mission data and provide it to the operators through the interface. In this section of the thesis, the use of neural networks is proposed as a tool to transfer functions from operators to interfaces. Specifically, the experiments performed demonstrate that these models allow the interface to evaluate the relevances and risks of robots mirroring the way a human operator would.
- **Immersive interfaces:** Once the relevant information is selected, according to the mission's state and operator preferences; the objective of this last block is to transmit this information to the operators, while controlling their workload and keeping their situational awareness. In this thesis, the development of immersive interfaces is proposed based on virtual reality, which introduces the operators in the scenario where the robots are working. The experiments performed on

this phase demonstrate that these interfaces improve the situational awareness of operators without increasing their workload.

All the developments of the thesis have been validated through experiments with operators, who have monitored and commanded robot fleets (formed by ground, aerial and manipulator robots) to perform realistic missions (surveillance and intervention in disaster areas) in varied and relevant scenarios (both outdoors and indoors).

Keywords: Mission, Mission Modeling, Fleet, Swarm, Ground Robot, Aerial Robot, Manipulator Robot, Operator, Operator Modeling, Workload, Situational Awareness, Stress, Human-System Interaction, Autonomy, Monitoring, Commanding, Interface, Adaptive Interface, Immersive Interface, Virtual Reality, Augmented Reality, Multimodal, Robotics, Data Mining, Process Mining, Petri Net, Decision Tree, Discovery Algorithm, Inductive Miner, Clustering, Artificial Intelligence, Machine Learning, Neural Networks.

Resumen

Las misiones con múltiples robots han experimentado un gran crecimiento en los últimos años. Este hecho se debe a que una flota de robots aporta más eficacia, eficiencia, flexibilidad y tolerancia a fallos que un robot individual. Sin embargo, este tipo de misiones plantea una serie de retos relacionados con los factores humanos. Entre estos retos se pueden destacar la gestión de la carga de trabajo, el mantenimiento del conocimiento de la situación, la gestión de situaciones de estrés, etc.

Esta tesis doctoral tiene como objetivo el desarrollo de una nueva generación de interfaces que permitan a un solo operador controlar una flota de robots. Con este fin se emplean herramientas como la minería de datos y el aprendizaje automático para descubrir la información más relevante y proporcionarla al operador, así como tecnologías de realidad virtual y aumentada para crear interfaces inmersivas que mejoren su conocimiento de la situación.

La tesis se puede dividir en cuatro bloques:

- **Modelado de misiones:** El objetivo es generar modelos de las misiones que permitan conocer lo que está sucediendo en cada momento y seleccionar la información más relevante. La tesis propone el uso de la minería de procesos para modelar automáticamente las misiones a partir de los datos generados por las mismas. En concreto, se propone el uso de redes de Petri para determinar el estado de la misión y árboles de decisión para predecir su evolución. Tras probar todos los algoritmos de descubrimiento de modelos de la minería de procesos se llegó a la conclusión de que el algoritmo de minado inductivo es el mejor en el contexto de las misiones con múltiples robots.
- **Modelado de operadores:** El objetivo es generar modelos de los operadores para predecir su comportamiento durante la misión y poder proporcionarles la información que necesitan en cada momento. La tesis propone la clasificación de los operadores según sus estrategias para controlar las misiones y la predicción de sus preferencias teniendo en cuenta los datos de las misiones anteriores. El primer paso se realiza utilizando algoritmos de agrupamiento (el algoritmo de partición alrededor de los medoides proporciona la mejor clasificación), mientras que el segundo empleando distribuciones de probabilidad.
- **Interfaces adaptativas:** Una vez obtenidos los modelos de las misiones y los operadores, el objetivo es descubrir la información relevante a partir de los datos generados por la misión y proporcionarla a los operadores a través de la interfaz. La tesis propone el uso de redes neuronales para transferir funciones de los operadores a la interfaz. En concreto, los experimentos demuestran que estos modelos pueden ser utilizados para que la interfaz evalúe las relevancias o riesgos de los robots de la forma en que lo hacen los operadores.
- **Interfaces inmersivas:** Una vez seleccionada la información relevante según el estado de la misión y las preferencias del operador, el objetivo es mostrarla de manera que alivie la carga de trabajo y mantenga

el conocimiento de la situación del operador. La tesis propone el desarrollo de interfaces inmersivas basadas en realidad virtual, que introducen a los operadores en el escenario en el que están trabajando los robots. Los experimentos demuestran que estas interfaces mejoran el conocimiento de la situación de los operadores sin incrementar su carga de trabajo.

Todos los desarrollos de la tesis se han validado mediante experimentos con operadores, que tenían que supervisar y controlar flotas de robots (formadas por robots terrestres, aéreos y manipuladores) para llevar a cabo misiones realistas (vigilancia e intervención en entornos de desastre) en escenarios variados y relevantes (tantos exteriores como interiores).

Palabras clave: Misión, Modelado de Misiones, Flota, Enjambre, Robot Terrestre, Robot Aéreo, Robot Manipulador, Operador, Modelado de Operadores, Carga de Trabajo, Conocimiento de la Situación, Estrés, Interacción Humano-Sistema, Autonomía, Supervisión, Control, Interfaz, Interfaz Adaptativa, Interfaz Inmersiva, Realidad Virtual, Realidad Aumentada, Multimodal, Robótica, Minería de Datos, Minería de Procesos, Red de Petri, Árbol de Decisión, Algoritmo de Descubrimiento de Modelos, Algoritmo de Minado Inductivo, Agrupamiento, Inteligencia Artificial, Aprendizaje Automático, Redes Neuronales.

Contents

Acknowledgements	vii
Graphical Abstract	ix
Abstract	xi
Resumen	xiii
1 Introduction	1
1.1 Motivation	1
1.2 Objectives	2
1.3 Framework	2
1.4 Contributions	4
1.5 Structure	4
2 State of art	7
2.1 Multi-Robot Missions	9
2.1.1 Mission planning	9
2.1.2 Mission monitoring	12
2.2 Fleet	13
2.3 Operator	14
2.3.1 Workload	15
2.3.2 Situational awareness	16
2.3.3 Stress	17
2.3.4 Training	17
2.4 Human-system interaction	18
2.4.1 Control architecture	18
2.4.2 Autonomy	19
2.4.3 Operator role	21
2.4.4 Commanding	22
2.4.5 Trust in automation	23
2.5 Interfaces	23
2.5.1 Design guidelines	24
2.5.2 Multimodal interfaces	26
2.5.3 Immersive interfaces	26
2.5.4 Adaptive Interfaces	26
3 Mission modeling	29
3.1 Resources	30
3.1.1 Mission architecture	30
3.1.2 Modeling techniques	32
3.1.3 Process mining	34
3.2 Methodology	35
3.2.1 Mission study	36

3.2.2	Event log preparation	37
3.2.3	Model discovery	38
3.2.4	Model evaluation and enhancement	39
3.2.5	Time analysis	40
3.2.6	Resource analysis	40
3.3	Model discovery	41
3.3.1	Simulations	41
3.3.2	Algorithms	44
3.3.3	Comparison	46
3.4	Model evolution	49
3.4.1	Experiments	50
	Missions	50
	Simulator	51
	Information	51
3.4.2	Mission modeling	53
	Petri net	54
	Conversion of variables	55
	Decision tree	55
3.4.3	Mission monitoring	56
3.5	Experiments	58
3.5.1	Mission	60
3.5.2	Robots and equipments	61
3.5.3	Interface	62
3.5.4	Integration	62
3.6	Validation	63
3.6.1	Mission study	63
3.6.2	Event log preparation	64
3.6.3	Model discovery	69
3.6.4	Model evaluation and enhancement	69
3.6.5	Time analysis	71
3.6.6	Resource analysis	73
3.7	Contributions	74
4	Operator modeling	75
4.1	Resources	76
4.1.1	Clustering	76
4.1.2	Prediction	78
4.2	Methodology	78
4.2.1	Definition of profiles	79
4.2.2	Classification of operators	80
4.2.3	Prediction of preferences	80
4.3	Experiments	80
4.4	Application	82
4.4.1	Definition of profiles	82
4.4.2	Classification of operators	83
4.4.3	Prediction of preferences	86
4.5	Contributions	87

5	Adaptive interface	93
5.1	Resources	93
5.2	Discovering mission information	94
5.3	Transferring operator functions	98
5.4	Contributions	102
6	Immersive interfaces	103
6.1	Resources	103
6.2	Missions	107
6.3	Interfaces	109
6.3.1	Robots	109
	Aerial robots	109
	Ground robots	110
	Manipulator robots	110
6.3.2	Scenarios	110
6.3.3	Monitoring	111
6.3.4	Commanding	112
	Manipulator robot	112
	Mobile robots	115
6.4	Experiments	118
6.5	Results	122
6.6	Contributions	127
7	Experiments and results	129
7.1	Scenarios	129
7.1.1	Outdoor scenario	130
7.1.2	Indoor scenario	133
7.2	Interfaces	135
7.2.1	Conventional interface	135
7.2.2	Adaptive & immersive interface	135
7.3	Experiments	139
7.4	Results	143
7.4.1	Workload	144
7.4.2	Situational awareness	144
7.4.3	Performance	144
7.5	Contributions	149
8	Conclusions	151
8.1	Conclusions	151
8.2	Contributions	153
8.3	Achievements	153
8.4	Applications	155
8.5	Future works	156
	Bibliography	159

List of Figures

2.1	Cloud of topics related to the control and monitoring of multi-robot missions.	8
2.2	Classification of main task allocation algorithms.	11
2.3	Classification of main motion planning algorithms.	12
2.4	Proposals of control architectures in literature. Models: Matellan et al. [144], Zheng et al. [245], Teichteil et al. [223], Marino et al. [143], Lang et al. [124], Khaleghi et al. (A and B) [111] and Roldán et al. [187]	19
3.1	Process mining and applications [1].	35
3.2	Flowchart of proposed methodology.	36
3.3	Situations for both teams over a match.	43
3.4	Path and autonomy of robots in a match.	43
3.5	Event log used in this work: representation with tasks (up) and with tasks and states (down).	45
3.6	Petri net obtained by Inductive Miner for the log with tasks.	49
3.7	Two scenarios considered for ISTAR missions. Map data ©2014 Google, based on BCN IGN Spain.	51
3.8	Examples of simulations with different scenarios, paths and events (loiters, contingencies and emergencies).	52
3.9	Event log of mission.	54
3.10	Petri net of mission.	55
3.11	Procedure for generating the mission trees.	56
3.12	Decision tree for the decision point in Surveillance between Reconnaissance, Emergency and Go to base.	57
3.13	Evolution of simulations.	57
3.14	Evolution of simulations.	58
3.15	Layout of experiments.	59
3.16	Experiments: (a) Scenario with robots, (b) Mission execution.	60
3.17	Multi-Robot Mission Interface: commanding window in front and monitoring window behind.	62
3.18	Procedure for generating event logs through telemetry and commands.	64
3.19	Examples of events generated by the location of UAVs.	65
3.20	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.	67

3.21	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.	70
3.22	Usage of areas during the mission.	72
4.1	The proposed methodology for modeling operators and predicting preferences.	79
4.2	Captures of USARSIM: (A) Main menu, (B) Selection of strategy, (C) Control of drone and (D) Control of rover.	81
4.3	Profiles of (A) operators and (B) missions.	84
4.4	Evolution value, mean, linear and quadratic strategy profiles.	85
4.5	Evolution of the mean of strategy variables of four players.	85
5.1	Discovery of information from data by using mission and operator models.	95
5.2	Diagram of SAVIER multi-UAV Demonstrator.	95
5.3	Captures of the interface: (a) Mission planning tag, (b) Mission monitoring window.	97
5.4	Contribution of prediction to interfaces.	98
5.5	Approximation of evaluation criteria of operator for robot relevance and risk.	100
5.6	Prediction errors of neural networks.	101
5.7	Evaluation of operator vs. prediction of neural networks. (a) Direct comparison; (b) Absolute error and mean value.	101
6.1	HTC Vive HMD, controllers and base stations.	105
6.2	Multi-robot missions performed in Luxembourg in 2016 and virtually reproduced in Madrid in 2017.	108
6.3	Non-predictive virtual reality interface.	111
6.4	Teleport to floor (left) and observation platform (right).	112
6.5	Predictive virtual reality interface.	113
6.6	Architecture to connect robot and VR worlds.	113
6.7	Commanding of the manipulator.	114
6.8	Multi-robot complete interface: operator enjoying the immersion in the mission (left), 2D view (top-right) and headset view (down-right).	115
6.9	Multi-robot commanding interface: teleporting closer to robots (top-left), selecting ground robot (top-right), selecting aerial one (down-left) and generating a waypoint for it (down-right).	116
6.10	(A) The architecture that connects simulated and real robots to conventional and virtual reality interfaces. (B) The section of the architecture dedicated to VR interfaces in further detail.	117
6.11	Conventional interface without the predictive component.	118
6.12	Conventional interface with predictive component.	119
6.13	Testing virtual reality interfaces: An operator working hard in real world (left) and what the operator is doing in virtual world (right).	119

6.14	NASA Task Load Index (NASA-TLX): English translation of the questionnaire of the experiments.	122
6.15	Situation awareness global assessment technique (SAGAT): English translation of the questionnaire of the experiments.	123
6.16	Box and whiskers diagrams for the workload and its variables. (A) Plot with the four interfaces: Conventional (CI), Predictive Conventional (PCI), Virtual Reality (VRI) and Predictive Virtual Reality (PVRI). (B) Plot with two groups of interfaces: Conventional (CIs) and Virtual Reality (VRIs).	125
6.17	Box and whisker diagrams for the situational awareness with four interfaces (left) and two groups of interfaces (right).	126
7.1	Scenarios considered for the experiments. Maps ©2018 Google and IGN España.	131
7.2	Mission in outdoor scenario.	131
7.3	Maps generated by SLAM algorithm.	132
7.4	Simplified ROS architectures of the ground robot. Squares represent nodes and ovals represent topics.	133
7.5	ROS architecture to manage simulated drones. Squares represent nodes and ovals represent topics.	134
7.6	Mission in indoor scenario.	134
7.7	Captures of conventional interface.	136
7.8	Capture of adaptive & immersive interface in outdoor scenario.	137
7.9	Captures of adaptive & immersive interface in indoor scenario.	138
7.10	Workload and situational awareness tests.	141
7.11	Workload and situational awareness tests.	142
7.12	Box and whiskers diagrams for workload and its related variables in outdoor scenario.	145
7.13	Box and whiskers diagrams for workload and its related variables in indoor scenario.	146
7.14	Box and whiskers diagrams for situational awareness in outdoor scenario.	147
7.15	Box and whiskers diagrams for situational awareness in indoor scenario.	147
7.16	Box and whiskers diagrams for performance and its related variables in outdoor scenario.	148
7.17	Box and whiskers diagrams for performance and its related variables in indoor scenario.	148
8.1	Cloud of topics related to multi-robot missions. The shaded area represents the ones covered by this thesis.	152
8.2	Applications of thesis developments to greenhouse agriculture, search and rescue and city monitoring.	157

List of Tables

2.1	A set of multi-robot missions from recent literature.	10
2.2	Human factors in multi-robot missions.	14
2.3	A diverse set of interfaces collected by literature.	24
2.4	The requirements for interfaces collected by literature.	24
3.1	Description of basic tasks	32
3.2	Analysis of modeling techniques	34
3.3	Situations of the game and tasks that respond to them.	42
3.4	Preliminary results of discovery algorithms.	46
3.5	Alignment of models with tasks to event logs.	46
3.6	Alignment of models with tasks and states to event logs.	47
3.7	Quality parameters for the Petri nets of simplified event log.	48
3.8	Quality parameters for the Petri nets of complete event log.	48
3.9	Events generated from telemetry.	65
3.10	Event logs considered in the work and their main features.	68
3.11	Models discovered from event logs.	69
3.12	Operator models and main features.	71
3.13	Duration of tasks.	73
3.14	Duration of commands.	73
4.1	Standard deviations for the variables of the profiles of players (operators) and levels (missions).	83
4.2	Comparison of clustering techniques with mean profiles.	88
4.3	Comparison of clustering techniques with linear profiles.	89
4.4	Comparison of clustering techniques with quadratic profiles.	90
4.5	Performance of predictions with different profiles (data in rows and methods in columns). Note: Performance with totally random prediction was 51.89%.	90
4.6	Performance of predictions for operators according to their number of games: Individual (I), Clusters (C), Population (P), Mean (M), Linear (L) and Quadratic (Q).	91
6.1	Study of available VR headsets.	105
6.2	Main considerations for developing immersive interfaces.	106
6.3	Design of experiments.	120
6.4	Summary of the results of the experiments.	123
6.5	T-test with pairs of interfaces in terms of NASA-TLX scores.	124
6.6	T-test with pairs of interfaces in terms of NASA-TLX scores.	126
7.1	Design of experiments.	139
7.2	Summary of results.	144
7.3	Results of performance for outdoor scenario.	149
7.4	Results of performance for indoor scenario. GR means Ground Robot, whereas MR means Manipulator Robot.	149

List of Abbreviations

2D	2 Dimensions
3D	3 Dimensions
ABM	Agent Based Model
A&II	Adaptive and Immersive Interface
AM	Alpha Minner
AMCL	Augmented Monte Carlo Localization
ANOVA	ANalysis Of VAriance
AR	Augmented Reality
BPMN	Business Process Modeling Notation
CI	Conventional Interface
DOF	Degree Of Freedom
DP	Decision Point
ETA	Estimated Time of Arrival
GCS	Ground Control Station
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
HFI	Human-Fleet Interaction
HM	Heuristic Minner
HMD	Head-Mounted Device
HMI	Human-Machine Interaction
HMM	Hidden Markov Machine
HRI	Human-Robot Interaction
HSI	Human-System Interaction
IAI	Intelligent Adaptive Interface
IM	Inductive Minner
IMU	Inertial Measurement Unit
ILPM	Integer Linear Programming Minner
ISTAR	Intelligence Surveillance Target Acquisition and Reconnaissance
LOA	Level Of Automation
MR	Mixed Reality
NASA-TLX	NASA-Task Load Index
NN	Neural Networks
PAM	Partition Around Medoids
PCI	Predictive Conventional Interface
PID	Proportional Integral Derivative
PN	Petri Net
PVRI	Predictive Virtual Reality Interface
ROS	Robot Operating System
SAGAT	Situation Awareness Global Assessment Technique
SAVIER	Situational Awareness VIRTUAL EnviRONment
SLAM	Simultaneous Location And Mapping
SM	State Machine
SOM	Self-Organizing Maps

STANAG	STAN darization A greement
TM	T ree M odel
UAS	U nmanned A erial S ystem
UAV	U nmanned A erial V ehicle
UGV	U nmanned G round V ehicle
USAR	U rbain S earch A nd R escue
USV	U nmanned S urface V ehicle
UUV	U nmanned U nderwater V ehicle
VR	V irtual R eality
VRI	V irtual R eality I nterface
WP	W ay P oint

Chapter 1

Introduction

*Caminante, son tus huellas
el camino y nada más;
Caminante, no hay camino,
se hace camino al andar.
Al andar se hace el camino,
y al volver la vista atrás
se ve la senda que nunca
se ha de volver a pisar.
Caminante no hay camino
sino estelas en la mar.*

Antonio Machado

This chapter introduces the present PhD thesis, presenting its motivation (Section 1.1), defining its main objectives (Section 1.2), describing its framework (Section 1.3), summarizing its main contributions (Section 1.4) and presenting the structure of document (Section 1.5).

1.1 Motivation

In the last years, multi-robot systems have become popular and have been applied to several scenarios. These systems can improve the capabilities of single robots without significantly increase their costs. Their strength is based on their effectiveness (they can do the same tasks in less time), efficiency (each task can be performed by the best robot), flexibility (they can adapt to different scenarios) and fault tolerance (they can recover a failure in a robot).

However, the control and monitoring of multi-robot systems poses multiple challenges to operators. Some of these challenges are managing a high workload (they have to receive data, understand mission, make decisions and send commands), keeping situational awareness (they have to know the mission state and evolution) and managing situations of stress (they have to avoid both boredom and anxiety).

Nowadays, each robot is controlled by at least one operator, but in a near future, a single operator must be able to control a robot fleet. There are multiple strategies to achieve this objective, such as increasing the level of autonomy of fleets, transferring functions from operators to interfaces, selecting the relevant information of mission and supporting the decisions of operators.

1.2 Objectives

This thesis aims at developing a new generation of interfaces in order to allow single operators to manage robot fleets. These interfaces will apply data mining and machine learning techniques to discover the relevant information, as well as immersive technologies as virtual and augmented reality to display it to the operators. Therefore, the purpose of this thesis is to explore the application of these techniques in the context of multi-robot missions.

More specifically, the thesis has four objectives:

- Mission modeling: The development of mission models that integrate all the actions of robots and operators and the interactions among them. These models will be used to determine the state of the mission at any moment and its potential evolution in the near future.
- Operator modeling: The development of models that predict the behaviors, states and preferences of operators during the missions. These models will be used to select the adequate information that the operators require to make the correct decisions.
- Adaptive interface: The development of an interface that is able to discover the relevant information from the whole data, taking into account the mission state and operator preferences.
- Immersive interfaces: The development of an interface that introduces the operators in the scenarios by means of virtual and augmented reality in order to improve their situational awareness.

This thesis aims to solve the problems of multi-robot missions and to follow an eminently practical approach. For these reasons, it is expected that all the developments of the thesis will be validated in a set of field experiments, where multiple operators will control heterogeneous robots to achieve varied objectives in different scenarios.

1.3 Framework

This PhD thesis is part of the SAVIER (Situational Awareness Virtual Environment) Open Innovation Project. This project was developed by Airbus Defence & Space in collaboration with five universities of Spain: Autonomous University of Madrid, Complutense University of Madrid, Charles III University of Madrid, University of Seville and Technical University of Madrid. This project aims to develop the technologies of the future Ground Control Stations for Unmanned Aerial Vehicle operation. The final goal is to allow human operators to manage UAVs' fleets and carry out complex missions. A list with the work packages of SAVIER Open Innovation Project is shown below:

- TD-01: "Multi-UAV coordination and control interface"

Author: Juan Jesús Roldán

Institution: Technical University of Madrid

- TD-02: "Minimizing search time of dynamic objects in uncertain environments"
Author: Sara Pérez
Institution: Complutense University of Madrid
- TD-03: "Collective intelligence applied to pattern detection and analysis in training systems and optimization problems"
Author: Víctor Rodríguez
Institution: Autonomous University of Madrid
- TD-04: "Multi-agent cognitive system for on-ground mission planning"
Author: Cristian Ramírez
Institution: Autonomous University of Madrid
- TD-05: "Advanced multimedia interfaces for multiple UAS"
Author: Juan Jesús Ruiz
Institution: University of Seville
- TD-06: "Stress level assessment with non-intrusive sensors"
Author: Francisco Hernando
Institution: Charles III University of Madrid
- TD-07: "Innovative AAA (Authentication, Authorization and Accounting) system for interoperable distributed architectures in GCS"
Author: Fidel Paniagua
Institution: Charles III University of Madrid
- TD-08: "Augmented reality over video stream acquired from UAVs for operations support"
Author: Susana Ruano
Institution: Technical University of Madrid
- TD-09: "Advanced gesture control and virtual objects interaction for immersive HMI"
Author: Tomás Mantecón
Institution: Technical University of Madrid
- TD-10: "Context-aware automatic speech recognition under cognitive stress aided by multimodal biometric detection"
Author: Fernando de la Calle
Institution: Charles III University of Madrid
- TD-11: "UAS air to air refueling"
Author: Jesús Martín
Institution: University of Seville

- TD-12: "Future intensive use of UASs for civil and military applications in non-segregated airspace"

Author: Ricardo Román

Institution: Technical University of Madrid

- TD-21: "Future HMS for UAV and human interface in GCS"

Author: Gabriel Barroso

Institution: Charles III University of Madrid

The PhD thesis corresponds to the TD-01 work package, which is focused on multi-UAV coordination and control interfaces. This research addresses the management of information in human-machine interfaces, in order to reduce the workload and improve the situational awareness of operators. This work package is closely related to other ones: TD-02 and TD-04 (which are involved in mission planning), TD-05 (which is related to operator interface) and TD-06 (which determines operator stress).

1.4 Contributions

The main contributions of the thesis to the state of art are listed below:

- As far as we know, this thesis is the first work that applies process mining to real robot missions, demonstrating that these techniques are suitable to model, analyze and improved this kind of missions.
- It proposes a new methodology to classify the operators and predict their preferences and validates it with a relevant amount of data provided by a serious game.
- It applies neural networks to model the evaluation of missions and successfully transfer this task from operators to interfaces.
- To our knowledge, this thesis is the first work to develop a complete immersive interface based on virtual reality and validate it in the monitoring and commanding of real robot missions.

1.5 Structure

The remainder of the document is organized as follows:

- Chapter 2 collects the state of art of multi-robot missions integrating multiple perspectives: missions, fleets, operators, human-system interaction and interfaces.
- Chapter 3 addresses the objective of mission modeling, which is required to select the information that is shown by the interface according to the mission state.
- Chapter 4 addresses the objective of operator modeling, which is needed to adapt the information that is shown by the interface to the operator state and preferences.

-
- Chapter 5 describes the development of adaptive interfaces, which are able to assume some functions of operators, as well as to support their decisions.
 - Chapter 6 describes the development of immersive interfaces, which introduce the operators in the scenarios to improve their situational awareness.
 - Chapter 7 explains the experiments performed to compare an interface that integrates the developments of the thesis to an interface of the state of art.
 - Chapter 8 summarizes the main conclusions of the thesis, as well as the future lines that it has opened.

Chapter 2

State of art

*If I have seen further it is by standing on
the shoulders of giants*

Isaac Newton

This chapter reports a complete state of art about the control and monitoring of multi-robot missions. The development and improvement of this kind of missions can be considered as a multi-disciplinary research, which involves multiple fields of study, such as robotics, mechanics, electronics, computing, human-machine interaction, artificial intelligence and computer vision. Figure 2.1 shows a cloud with the topics and trends related to the control and monitoring of multi-robot missions.

The chapter is inspired by our experiences applying multi-robot systems in agriculture and security published in the following articles:

- Roldán, J. J., Joossen, G., Sanz, D., del Cerro, J., & Barrientos, A. (2015). Mini-UAV based sensory system for measuring environmental variables in greenhouses. *Sensors*, 15(2), 3334-3350 [191].
- Roldán, J. J., Garcia-Aunon, P., Garzón, M., de León, J., del Cerro, J., & Barrientos, A. (2016). Heterogeneous multi-robot system for mapping environmental variables of greenhouses. *Sensors*, 16(7), 1018 [190].
- 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 [69].

The rest of the chapter exhaustively collects recent references that address this kind of missions from multiple points of view:

- Mission (Section 2.1): A complete set of recent multi-robot missions is studied, taking into account their main requirements and features, as well as the problems of mission planning and their current solutions.
- Fleet (Section 2.2): A study of fleet designs collected by literature is performed, including homogeneous robot teams, heterogeneous robot teams, robot swarms and human-robot teams.
- Operator (Section 2.3): A review of human factors in multi-robot missions is developed, considering relevant issues such as workload, situational awareness, stress, trust and training.

FIGURE 2.1: Cloud of topics related to the control and monitoring of multi-robot missions.

- Human-system interaction (Section 2.4): The extensive literature about human-system interaction is reviewed with the focus on important topics such as control architectures, scales of autonomy and commanding methods.
- Interface (Section 2.5): An analysis of interfaces is performed, including not only the design of conventional interfaces (guidelines about the display of information and alerts), but also the attempts to develop alternative interfaces (multimodal, immersive and adaptive interfaces).

2.1 Multi-Robot Missions

A first question that we can ask about multi-robot missions is “why should we use multiple robots instead of a single one?”. Here are multiple reasons:

- Effectiveness: A robot fleet is more effective than a single robot, because it has more resources to accomplish the same tasks. A good example is the surveillance of a large area, which can be performed by a robot fleet in less time than by a single unit.
- Efficiency: A robot team is more efficient than its members, because the allocation of tasks to robots can be optimized. For instance, the aerial robots can be assigned to exploration tasks, whereas the ground robots can be used for transportation tasks.
- Flexibility: In the same way, a robot team is more flexible than its members, because it can adapt to different scenarios just by changing the allocation of the tasks to robots.
- Fault tolerance. A robot fleet is more fault tolerant than a single robot, because when a robot has a breakdown, the fleet can continue the mission.
- Diversity: A heterogeneous team can be applied to scenarios with multiple domains (e.g. air, ground and sea), whereas most of the robots are only able to work in a domain.

A review of recent literature has provided the multi-robot missions collected by Table 2.1. This table contains information about the objectives of the missions, as well as the robots and operators that are involved.

The execution of this kind of missions has led to the development of several techniques for mission planning and monitoring. These techniques are analyzed in the following subsections: the ones related to mission planning in Section 2.1.1, whereas the ones about mission monitoring in Section 2.1.2.

2.1.1 Mission planning

Multi-robot mission planning is a well-known problem that involves multiple issues, such as the robotic systems (including both single-robot and multi-robot ones), multi-robot environments (taking into account cooperative and competitive ones), resource conflict, coordination among robots

TABLE 2.1: A set of multi-robot missions from recent literature.

Reference	Mission	Robots	Operators
[228]	Surveillance	2 UUVs / 2 USVs / 2 UAVs	No
[227]	Area coverage	3 UGVs	No
[102]	Area coverage	2 UGVs	No
[132]	Surveillance	1 USV / 1 UAV	N
[229]	Monitoring	N UAVs	1
[226]	People tracking	3 UGVs	No
[29]	Surveillance	1 UGV / 1 UAV	1
[43]	Search and rescue	1 UGV / 1 USV / 1 UAV	N
[70]	Exploration	1 UGV / 1 UAV	No
[69]	Area coverage	3 UGVs	1
[108]	Mapping	2 UUVs	N
[121]	Search and rescue	N UGVs / N UAVs	N
[78]	Mapping	2 UGVs	1
[159]	Exploration	6 UAVs	1
[130]	Securing	1 UAV / 1 UUV	6
[241]	Patrolling	20 UGVs	1
[190]	Monitoring	1 UGV / 1 UAV	No
[207]	Exploration	6 UAVs	1
[188]	Fire surveillance	2 UAVs	1

(which can be deliberative or reactive), communications among them (which can be explicit or implicit), decision-making structures (reviewing centralized and distributed ones), and task and motion planning problems [238]. Further information can be found in the surveys about this topic [98, 116, 153].

Task planning can be split into two problems: task decomposition and task allocation [238]. The first problem involves the division of the mission into tasks that can be performed by single robots. The second one addresses the assignment of these tasks to the robots looking for optimizing mission performance. There are some recent and complete surveys about this problem and the possible algorithms to solve it [153, 105, 106, 112].

The basic problem of task allocation consists of a set of tasks (T), a set of robots (R) and a set of utilities of robots to execute tasks (U). The objective of this problem is to find the allocation of tasks to robots that maximizes the utility of fleet [112].

Nevertheless, this problem can reach more complexity. On the one hand, the criteria to be optimized can vary, including mission time, covered distance, robot consumption and even a set of them. On the other hand, some restrictions can be added to the planning, generating a constraint satisfaction problem [173]:

- The features of tasks: e.g. their geographic areas and time intervals.
- The dependencies between tasks, which can be represented by Allen's Interval Algebra [11]: relationships of precedence, parallelism, meeting, overlapping, simultaneous starting or finishing...

FIGURE 2.2: Classification of main task allocation algorithms.

- The features of the robots: e.g. their dynamics and kinematics models, fuel or battery consumptions, initial positions and types of sensors and actuators.

As shown in Figure 2.2, there are two paradigms of task allocation algorithms: centralized and distributed. Centralized algorithms have a central agent that allocates the tasks to the rest of them. The main advantage of these algorithms is they can provide optimal solutions in terms of costs, time and resources, whereas the main drawbacks are they are not robust (if the central agent fails, the task allocation is not possible) and scalable (they could not deal with huge amounts of tasks and robots). In contrast, distributed algorithms are based on the communication of information and allocation of tasks between equal agents. These algorithms are more flexible, robust and fault tolerant, but the quality of solutions depends on the information managed by agents.

The centralized algorithms can apply deterministic or stochastic techniques. Deterministic techniques follow rigid procedures, which means that given certain initial conditions, they always follow the same paths and reach the same results. A good example is the formulation of task allocation as a stable marriage problem, as well as the application of Gale-Shapley algorithm to solve it [187]. Stochastic techniques integrate random factors, so multiple solutions are generated combining heuristic and randomness and the best of them are selected. Two of these techniques are genetic algorithms [174] and particle swarm optimization [240].

The distributed algorithms include market, society and agent-based methods. Market-based approaches consider the robots follow different interests and establish market mechanisms to distribute tasks. Some examples are game theoretical algorithms [38] and auction mechanisms [212]. Society-based approaches propose collective decision making in the fleet by using mechanisms as strategy diffusion [107] and coalition formation [164]. Agent-based approaches distribute the robots in the scenario and provide them with the capability to take nearby tasks [25].

Multi-robot motion planning involves not only generating continuous collision-free paths from origins to destinations, but also avoiding the possible interferences between robots. Therefore, the motion planning of every robot should take into account the movements of the rest. However, the techniques for multi-robot motion planning are usually extensions of the techniques for single-robot motion planning.

Multiple surveys about motion planning can be found in recent literature, each one focused on a specific type of robot, task or scenario: motion planning for UAVs [74], coverage path planning [67], three-dimensional

FIGURE 2.3: Classification of main motion planning algorithms.

path planning [241] and heuristic techniques [138]. Figure 2.3 shows a classification of the main motion planning algorithms collected by literature.

On the one hand, classic algorithms encompass cell decomposition, potential field, sampling and subgoal methods. Cell decomposition methods divide the space into cells and generate paths through these cells [8]. Potential field methods generate attractive forces to goals and repulsive forces from obstacles, in order to move robots to goals avoiding obstacles [46]. Sampling methods build graphs with random samples and search the minimum path from current to goal positions. The most common of these methods are probabilistic road-map (PRM) [35] and rapidly-exploring random trees (RRT) [118]. Finally, subgoal methods generate lists of reachable positions from initial to target positions [133].

On the other hand, heuristic algorithms involve neural networks, fuzzy logic, genetic and swarm algorithms. These algorithms overcame the classic ones in terms of application in research works at the beginning of the 21st century [149]. Neural networks are able to learn the localization and navigation skills of robots, receiving as inputs the targets and obstacles and providing as outputs the movements of robots [45]. Fuzzy logic techniques use the inputs from sensors (e.g. speed of robot and distance to obstacles) and a set of fuzzy rules (e.g. follow target, avoid obstacle, keep direction...) to generate the outputs to actuators (e.g. attitude and speed) [30]. Genetic algorithms evolve a population of possible paths through crossover, mutation, evaluation and selection operations, in order to obtain the minimum obstacle-free path [161]. Finally, particle swarm optimization starts from a random population of particles and update them with their experiences, looking for maximizing an objective function [244].

Motion planning problems are more complex when they take into account multiple criteria (time, distance, consumption...), as well as multiple constraints (robot and target dynamics, static and mobile obstacles...). Some tasks present special requirements for motion planning, such as surveillance and tracking. For instance, motion planning in surveillance should consider the dynamics of robots, models of sensors and probabilistic distributions of targets [170].

2.1.2 Mission monitoring

Once there is a plan with the tasks and paths of robots, the focus is on monitoring the mission. This activity involves a set of robots with payloads that collect information, a set of operators that control the mission, and a set of interfaces that communicate robots and operators.

Mission modeling is useful not only for mission specification but also for mission monitoring, since it provides tools to know the state of mission at any time and make the best decisions according to this information. An objective can be to develop a common language for operators and robots or, at least, a family with different languages for different domains, such as mission monitoring, robot and behavior languages [47].

Other proposals address the automatic modeling through the experience of previous missions. The study of modeling techniques in the context of robot missions reveals the Petri nets and hidden Markov are the most promising, due to their ability to manage concurrency and uncertainty respectively [184]. The discovery algorithms of process mining can be used to automatically generate Petri nets from event logs of multi-robot missions [186]. Additionally, decision trees can be added to Petri nets in order to manage uncertainty: Petri nets estimate the state of the mission, whereas decision trees are able to predict its evolution. [189].

2.2 Fleet

Section 2.1 showed a diverse set of multi-robot missions performed during the last decade. These references apply different fleet configurations and robot types to adapt to different scenarios and missions. The design of the fleets implies a series of decisions (single robot vs robot team, class of team, types of robots...) and requires an analysis of the mission [27, 72]. The fleets applied in this kind of mission can be classified according to multiple criteria:

According to their size, the fleets can be classified into single units (which is the minimum number of robots), two unit systems (which is the minimum size to form a team), multi-unit systems (where robots perform one or multiple tasks that require or not coordination) and swarms (where robots have simple behaviors and tasks are accomplished by their interactions).

According to their morphology, the multi-robot systems can be divided into identical fleets (where all the members present the same mechanics and capabilities), homogeneous fleets (where all the members have similar morphology but differences in their capabilities, such as minor changes in their sensors or actuators) and heterogeneous fleets (where the robots have differences in morphologies and capabilities, such as combinations of ground and aerial robots).

Finally, the fleets can be classified into not coordinated, weakly coordinated and strongly coordinated, according to the level of cooperation between robots; aware and unaware if the robots know or do not know that they are cooperating; and centralized, distributed and hybrid, according to the coordination architecture [27, 63, 88].

Furthermore, the mobile robots that take part in this kind of missions can be classified according to their application domain and locomotion method into the following categories:

- Unmanned Ground Vehicles (UGVs).

Wheeled robots.

Legged robots.

Tracked robots.

TABLE 2.2: Human factors in multi-robot missions.

Issue	Problem	Detection	Solution
Workload	Inefficiency and errors	Physiological signals Test (NASA-TLX)	Adjust autonomy Transfer functions
Situational Awareness	Inefficiency and errors	Actions and performance Test (SAGAT)	Immersive interface Filter information
Stress	Anxiety / Boredom: Human errors	Physiological signals Test (NASA-TLX)	Adjust autonomy Filter information
Trust	Mistrust: Human errors Overtrust: Machine errors	Reactions Survey	Adjust autonomy Train operators

- Unmanned Aerial Vehicles (UAVs).
Fixed-wing robots.
Multi-rotor robots.
- Unmanned Surface Vehicles (USVs).
- Unmanned Underwater Vehicles (UUVs).

Note that industrial and service robots are excluded in this classification, since they are not applied in the considered multi-robot missions.

The robots can be equipped with a huge variety of payloads to perform a wide range of missions. Among these payloads, we can find multiple types of cameras (both 2D and 3D), laser or sonar scanners and environmental sensors (temperature, humidity, luminosity, gases...). These payloads are used to collect information about the environment and be able to actuate over it.

Two trends can be detected in recent literature: on the one hand, the rise of swarms and, on the other hand, the human-robot teaming. In the first case, the works are focused on the development of new techniques for human-swarm interaction [115]. In the second one, the challenge is in designing systems for peer to peer human-robot interaction [120]. A review of intelligent systems in these scenarios with their strengths and weaknesses can be found in [31].

2.3 Operator

Human factors emerged as a key issue in the development of multi-robot missions. These missions revealed that human operators face several challenges with autonomous systems, which unfortunately may lead to accidents with different causes and consequences. Table 2.2 summarizes the main human factor problems in multi-robot missions, as well as the methods for detecting and preventing them.

This section analyzes multi-robot missions from the point of view of operators. For this purpose, the following subsections are focused on the main issues of human factors in these missions: operator workload (Section 2.3.1), situational awareness (Section 2.3.2), stress assessment (Section 2.3.3) and operator training (Section 2.3.4).

2.3.1 Workload

Workload can be defined as the sum of the amount of work, the working time and the subjective experience of the operator [137]. Additionally, this concept can be divided into three attributes: input load, operator effort and work performance [151]. Finally, multiple dimensions of workload can be considered, such as physical, mental and temporal demands [91]. A good survey about workload in the context of multi-robot supervision can be found in [236].

The control of multiple robots implies a rise of workload for single operators [40]. This fact can be explained by the multitasking nature of multi-robot missions [50]. The operators of this kind of missions have to perceive data, discover information, understand situation, make decisions and generate commands. An excessive workload of operators can increase waiting times, cause errors in decision making and, therefore, decrease mission performance [51].

The most common method to determine workload is the NASA Task Load Index (NASA-TLX) [91]. It is a questionnaire where the operators can perform a subjective evaluation of their workload by answering a set of questions. In the first part of the questionnaire, the operators compare some variables according to their influence on workload: mental demand (low–high), physical demand (low–high), temporal demand (low–high), performance (good–poor), effort (low–high) and frustration (low–high). In the second one, they rate these variables regarding their experience during the mission. The results of twenty years of experiments with NASA-TLX questionnaire have been compiled in [90].

However, there are other methodologies to estimate the workload of operators. An early review included not only subjective methods, but also analytical, empirical and physiological techniques [137]. Some aspects of workload can be inferred from the actions of operators (e.g. the distribution of attention among the robots or the time spent in monitoring and commanding activities), whereas other ones have to be estimated from physiological variables (e.g. cardiac response, eye movement, brain activity, blood pressure and galvanic skin response).

Most of recent works propose to estimate mental workload from physiological signals. Some examples are the application of functional near infrared (fNIR) spectroscopy to monitor the brains of air traffic control and UAV piloting operators [14], use of electroencephalogram (EEG) and peripheral physiological measures (PPMs) and application of several models of classification and regression to predict the mental workload of the players of a video game [215], and utilization of heart rate variability (HRV) to estimate mental workload of UAV operators [104].

The literature collects multiple strategies to reduce operator workload in multi-robot missions. Some works look for transferring functions from operators to robots to lighten their workload, such as the navigation and tracking skills [50]. Other ones propose providing the operator with support in perception, comprehension, decision making and execution of tasks [100].

2.3.2 Situational awareness

Situational awareness can be described as the perception of the elements in the environment within a volume of time and space (level 1), the comprehension of their meaning (level 2), and the projection of their status in the near future (level 3) [58]. More generally, five types of awareness in multi-robot missions can be defined according to the subject and object [53]:

- Human-human: Humans know identities, activities and locations of other humans.
- Human-robot: Humans know identities, activities, locations and surrounding of robots.
- Robot-human: Robots know human commands and constraints.
- Robot-robot: Robots know tasks and plans of other robots.
- Human-mission: Humans know the mission objectives and their degree of achievement.

Some examples of variables that have impact on human-robot awareness [52] are the spatial relationships between agents and scenario, weather in the scenario, health and status of robots, operational threats, information related to mission, etc.

There are multiple factors that have influence on situational awareness. A good example is the operator workload: situational awareness can decrease when workload is high due to lack of resources, but it can also decline when workload is low due to boredom [39]. Another example is the system autonomy: increasing the autonomy to a certain level can improve the situational awareness, but increasing it from this level could have a negative impact [60]. Other relevant variables are the complexity of mission, number of robots and type of interface.

The consequences of the lack of situational awareness in multi-robot missions can vary from mistakes to accidents. A compilation of accidents of UAVs and their potential causes related to the lack of situational awareness of operators can be read in [52].

The most common method to estimate situational awareness is the Situational Awareness Global Assessment Technique (SAGAT) [60]. This method provides an objective measurement and works in the following way:

1. The operator is monitoring a simulation of a multi-robot mission.
2. At some random time, the simulation is stopped and the interface is blanked.
3. The operator has to answer a series of questions about the situation of mission.
4. At the finishing of mission, the real and perceived situations are compared.
5. A score is determined in three zones (immediate, intermediate and long-range).

Other techniques estimate situational awareness through the performance of operators in certain situations along the missions [206]. For this purpose, the simulations must introduce mission deviations, unexpected agents or anomalous measurements. In this moment, the operators have to react to these contingencies by leaving the plans, making decisions and generating new commands. Therefore, a correlation between the performance of their actions and their situational awareness can be established [61].

The literature collects multiple strategies to improve situational awareness in multi-robot missions. Most of these proposals are related to the design of interfaces and focus on the management of information [214] and immersion of operator [131].

2.3.3 Stress

The stress of operator is other variable that impacts on the success of mission. Neither high nor low levels of stress are desirable: the first ones because could indicate boredom, whereas the second ones because could point out anxiety [41]. Therefore, medium levels of stress are optimal in multi-robot operation, since they indicate the operators are involved in the mission and can achieve the objectives.

Although there are subjective techniques to evaluate stress, such as the questions related to effort and frustration of NASA-TLX questionnaire, most of the studies propose objective measurements.

Multiple physiological signals are used to estimate stress, such as the heart rate variability [146], galvanic skin response [123] and brain readings [113]. Additionally, several studies detect states of stress and relaxation by studying the pupil diameter [177, 167]. Finally, a work in the context of social robotics detects some affects including stress through the body language and vocal intonation [96].

2.3.4 Training

The first operators were the people involved in the research, design, development and application of robots. In military applications, the earliest operators of UAVs were the former pilots of manned aircrafts, whereas in civilian applications, the operators had a wider variety of backgrounds. In both cases, training procedures were developed to obtain native and professional robot operators.

The literature collects multiple simulation environments for training multi-robot operators. A simulator for training up to ten operators in a shared scenario is developed in [148], whereas a training platform for operators without prior experience in the operation of multiple robots is presented in [218]. Both works look for a quick training of workers without experience with robots to intervene in disaster scenarios.

Another relevant fact is the study of operators during training, in order to determine if they are ready to operate robots. For this purpose, data mining techniques as clustering are used to obtain operator profiles [181], whereas Petri nets and hidden Markov models are applied to model their behaviors [179].

2.4 Human-system interaction

Human-System Interaction (HSI) is a discipline that involves studying, designing, developing and evaluating robot systems that work with humans. This term is used in order to encompass the interactions of humans with robots, fleets, computer and interfaces. The objective of HSI is to allow humans and systems to work together to accomplish tasks.

There are multiple types of human-system interaction. According to the location of agents, they can be remote (if humans and robots are separated spatially or temporally) or close (if they are in the same place and time). According to the nature of tasks, they can be classified as teleoperation, supervision, social interaction, etc. Further information can be found in this excellent survey about human-robot interaction [75].

The next subsections address multiple concepts of human-system interaction, such as control architectures (Section 2.4.1), levels of autonomy (Section 2.4.2), operator roles (Section 2.4.3), commanding methods (Section 2.4.4) and trust in automation (Section 2.4.5).

2.4.1 Control architecture

Control architectures define the functions of agents in multi-robot missions, as well as the interactions among them and the communications of the whole system. A diverse set of proposals is collected by references [144, 245, 223, 143, 124, 111, 187], which are classified in Figure 2.4. According to the distribution of functions, there are three possible schemes:

- **Centralized architectures:** The base station performs all the functions of data processing, decision-making, mission planning, task allocation, path planning, payload control and command sending; whereas the robots only follow the paths, execute the payload actions and collect data. On the one hand, these architectures are easy to design, implement and handle, but, on the other hand, they have problems of scalability to multiple robots due to the potential overload of the base station.
- **Distributed architectures:** The robots perform the functions of data collection and processing, decision-making, task allocation, path planning and following and payload control and execution; whereas the base station is only an interface among robots and operator, which provides operator with information of mission and sends operator commands to fleet. These architectures are efficient due to their modularity and adaptability to complex scenarios, but they often require strong coordination among the agents and have high dependency on communications.
- **Hybrid architectures:** They are a compromise solution that share elements with both centralized and distributed schemes. These proposals consider centralized and distributed functions and assign them to the base station and fleet respectively. These approaches allow to adapt the control architecture to the particular scenarios, missions, fleets and operators. As shown in Figure 2.4, most of the proposals of multi-robot control architectures are hybrid.

FIGURE 2.4: Proposals of control architectures in literature. Models: Matellán et al. [144], Zheng et al. [245], Teichtell et al. [223], Marino et al. [143], Lang et al. [124], Khaleghi et al. (A and B) [111] and Roldán et al. [187]

In addition, these architectures are structured into multiple layers that perform different functions. These layers can be located in base station or robots and may develop reactive or deliberative behaviors. Commonly, reactive behaviors are used to robot control, whereas deliberative ones are applied for mission planning.

2.4.2 Autonomy

The literature about robotics contains diverse definitions of autonomy [17]. A compromise among them leads to define autonomy as "the capability of systems to carry out some tasks without the control of humans". The most common way to describe the autonomy of multi-robot systems are the levels of autonomy (LOAs). These LOAs establish the roles of operators and robots, taking into account their actions and interactions.

The first reference to the LOAs in the literature, as well as the most analyzed and applied, is the work of Sheridan and Verplank in 1978 [217]. This scale considers ten LOAs according to the roles of computers and humans at the time of generating the options, selecting an action, launching it and informing the other agents involved in the mission:

1. Human provides the options, selects the action and launches it.
2. Computer provides the options, whereas human selects and launches the action.
3. Computer suggests one of the options, whereas human selects and launches the action.

4. Computer selects the action and human can approve or reject it.
5. Computer selects the action and launches it if human approves.
6. Computer selects the action and launches it if human does not reject in a certain time.
7. Computer performs the action and informs human.
8. Computer performs the action and informs if human requests.
9. Computer performs the action and informs if it decides.
10. Computer performs the action and ignores human.

Another relevant work about LOAs was developed by Endsley and Kaber in 1999 [59]. This work also considers ten LOAs, but assigns different meanings to them:

1. Manual control: Human performs all tasks.
2. Action support: System assists to human with performance of actions.
3. Batch processing: Human generates the options and selects the tasks, whereas system performs them.
4. Shared control: Human and system generate the options, human selects the tasks, and both perform them.
5. Decision support: System generates the options, human selects the tasks and system performs them.
6. Blended decision making: System generates the options, selects the tasks and performs them if human consents.
7. Rigid system: System presents a limited set of options and human selects among them.
8. Automated decision making: System generates the options and selects the best one.
9. Supervisory control: System selects and performs the tasks, whereas human monitors system and intervenes if necessary.
10. Full automation: System performs all tasks.

A third interesting work about LOAs is the developed by Cummings et al. in 2007 [40]. This work reduces the scale of autonomy from ten to six levels and focuses on the control of multiple UAVs by a single operator:

1. Human makes all decisions and actions.
2. Computer offers the complete set of decisions/actions.
3. Computer offers a selection of decisions/actions.
4. Computer suggests an alternative and executes it if human accepts.

5. Computer suggests an alternative and executes it unless human rejects.
6. Computer makes all decisions and actions.

Other works [200, 162] summarize the possible configurations into four LOAs: fully manual, management by consent (system performs the tasks if human consents), management by exception (system performs the tasks if human does not reject in a certain time) and fully autonomous.

The first one [200] compares these LOAs by means of a set of multi-UAV missions. The management by consent provides the best results in terms of performance, avoidance of errors and situational awareness (self-rating questionnaires). Nevertheless, the results of management by exception in terms of workload (NASA-TLX questionnaire) are better as more UAVs are used in the mission.

The second one [162] makes a survey to discover the preferences of pilots about autonomy. The results show they prefer management by consent against higher levels of autonomy. This preference is stronger when the tasks are complex or critical and weaker when the workload is high.

In addition, the work of Parasuraman et al. in 2000 [165] provides an interesting perspective about LOAs. They distinguish between different functions (data acquisition, information analysis, decision making and action execution) and propose different LOAs for each one of them. This approach leads to the development of flexible systems, where humans and systems can distribute tasks according to their performance.

Finally, the work of Beer et al. [17] provides some additional guidelines to determine the autonomy of robots and provide adequate human-robot interactions. These guidelines are related to the tasks, components of tasks (i.e. sense, plan and act), performance of robots, levels of autonomy, and influence of autonomy on human-robot interaction.

2.4.3 Operator role

The roles of operators are mainly conditioned by the levels of automation. For instance, in low levels the operators have to teleoperate the robots, whereas in high levels they only have to supervise the mission. Nevertheless, the roles of operators may have certain variability even within defined levels of automation.

The current operation of UAVs often involves multiple operators for each vehicle with different roles [12]. The most common are the flight operators, who control the flight of the UAVs during the mission, and payload operators, who command the payloads and manage their information [52]. However, there are other roles, such as mission commanders, who supervise the mission and coordinate the operators, and analysts, who provide information of scenarios, resources and targets [145]. Furthermore, the maintenance operators, who are responsible of the vehicles, stations and communications, must be considered as an important part of the crew [94].

Similar schemes are proposed for the application of heterogeneous air-ground teams to urban search and rescue missions [155]. However, the research works recently consider future scenarios with more robots controlled by less operators and even human-robot teamwork [77].

The studies of supervisory control pose five possible roles for operators during missions: plan, teach, monitor, intervene and learn [216]. First, the operators take part in mission planning and teach the systems to perform tasks. Later, they monitor the systems to ensure their success and, if required, intervene to correct the errors. Finally, they learn from the results of mission to improve the planning of future ones.

Other works define five roles for operators that imply different ways to control mobile robots: operator, teammate, supervisor, bystander and mechanic [213]. The operator can adjust the control systems of robots or directly take control of them. The teammate collaborates with robots on equal terms to accomplish tasks. The supervisor monitors the operation of robots and intervenes if necessary. The bystander does not interact with robots but needs to understand their behaviors. Finally, the mechanic can adjust the physical components of robots.

2.4.4 Commanding

When the autonomy of systems is high and operators have supervisor roles in missions, a method of commanding is not required. The reason is that systems make decisions and execute tasks without the participation of operators. Despite this fact, the operators can usually intervene in various ways: e.g. replanning when the context has changed and the plan is not valid, or changing the plan by adding and removing tasks or modifying the robot-task allocations.

Nevertheless, when the autonomy of systems is low and the operators have an active role in the control of robots, the method of commanding is important. An adequate method of commanding can reduce the time that operator spends generating and sending commands and, therefore, improve the performance of mission [188]. The different methods of commanding are listed below sorted according to increasing autonomy [195]:

- Direct control: The operators have to directly command the motions of robots and payloads. Commonly, they use a device to control them: e.g. a RC controller, keyboard, joystick or joypad. In this way, they can control translation in X, Y and Z axis, rotation in roll, pitch and yaw, translational and rotational speeds, etc. This method establishes the closest relationship between operators and robots, which can optimize the performance in the control of robots, but limits the ratio between robots and operators to one or even less.
- Action commands: The operators have to command the actions of robots and payloads. In the case of robots, these actions are usually waypoints that contain at least the target position and orientation, but also can include more information: e.g. speed, time and actions to perform. In the case of payloads, there are more commands such as take a picture, record a video, take a measure, pick or place a load, etc. This method allows an operator to manage more than one robot, but still implies a high workload, which may produce an inefficient use of robots.
- Task commands: The operators have to command the tasks of robots, defining their names, locations, resources and targets. These tasks

use robots to cover certain paths and activate their payloads. Some examples of tasks are the surveillance of areas, reconnaissance of targets, tracking of mobile targets, transport of loads, etc. This method reduces the workload of operators and allows them to manage more robots, but can affect the performance of some tasks. For instance, when the robots generate their paths in surveillance, they may not take into account the knowledge of scenario and the intuitions about targets of operators.

2.4.5 Trust in automation

A last relevant topic in human-system interaction is the trust in automation of operator. Two extreme situations must be avoided: overtrust and mistrust [165]. In the first case, the operator could stop paying attention to mission and do not detect the failures of automation. Overtrust is more common when the level of automation is high, the error rate of automation is low and the operator does not have previous experiences of errors [242]. In the second case, the operator could reject the support of automation, increasing his/her workload and reducing the performance of mission. In contrast to overtrust, mistrust occurs when the level of automation is medium, the error rate of automation is significant and the operator is aware of the consequences [242].

The literature distinguishes among extrinsic and intrinsic factors regarding trust in automation [32]. An important extrinsic factor is the workload of operator (more workload generally implies more trust in automation) [208], whereas an intrinsic factor that must be remarked is the operator self-confidence (less self-confidence implies more trust in automation) [208]. The experiments show the operators have more confidence with low autonomy systems than with high autonomy ones, this difference is bigger when the number of robots is higher, and the presence of failures has a remarkable influence on this perception [201].

2.5 Interfaces

Interfaces are a key element in multi-robot missions, since they manage the interactions between operators and robots. They have influence not only on discovery and display of information, but also on generation and transmission of commands. A diverse set of interfaces with their features is collected in references [147, 85, 115, 41, 65, 66, 176] and Table 2.3.

A survey of Ground Control Stations (GCSs) for UAVs can be seen in [12], which includes both portable, mobile and fixed stations and analyzes aspects such as hardware, software, interfaces and communications. Another survey about interfaces, in this case for supervisory control of robot fleets, can be found in [32]. Among other issues, it analyzes multimodal interfaces, planning tools, visualization tools, attention management tools, trust calibration tools, adaptive automation and intelligent agents.

This section summarizes the main contributions and challenges in multi-robot interfaces. The rest of section is organized as follows: Section 2.5.1 collects some guidelines for the design of interfaces, Section 2.5.2 analyzes

TABLE 2.3: A diverse set of interfaces collected by literature.

Reference	Robots	Operators	Multimodal	Immersive	Adaptive
[147]	1	1	Yes	No	No
[85]	40	1	Yes	No	No
[115]	50-200	1	No	No	No
[41]	4	1	No	No	No
[65]	3	1	No	No	No
[66]	4	1	No	No	No
[203]	3	1	No	Yes	No
[176]	10	1	No	Yes	No
[199]	1	1	No	Yes	No
[152]	N	1	No	No	Yes

TABLE 2.4: The requirements for interfaces collected by literature.

References	Requirements
[125]	Resistance to weather, environment and conditions.
[214]	Integration of robot position, health, status and measurements in the same displays.
[157]	Reduction of the amount of information.
[95]	Adaptation to the preferences of operator.
[163]	Guidance of operator attention to relevant information.
[9]	Use of maps to show information of robots and mission.

multimodal displays and commands, Section 2.5.3 addresses some proposals of immersive interfaces, and Section 2.5.4 explores the proposals of adaptive interfaces.

2.5.1 Design guidelines

This section tries to answer the question "How should a good multi-robot interface be?". For this purpose, a review of the literature about interfaces was performed and the main design guidelines are collected. The main hardware and software requirements obtained from this review are collected in Table 2.4 and explained above.

The literature discusses more about software than hardware. Nevertheless, there are interesting contributions related to devices. In urban search and rescue (USAR) missions, the operators usually demand control units that resist the weather (heat, cold, rain and snow) and other conditions of these environments, can be used with firefighters gloves and are visible in sun and dark [125].

Ruiz et al. [202] consider a scenario where the GCS monitors the UAVs, a pilot controls them, and the communication between the GCS and pilot is critical. They explore three possibilities: radio communication (the pilot asks for information and the GCS provides it), tablet interface (the pilot has a tablet with the video and information of UAV), and monocular head-mounted display (the pilot has this information while keeping the eye contact with UAV). The results show the head-mounted display is the best

option in terms of workload and situational awareness. However, these results may change in a scenario with multiple UAVs, where the impact of visual contact is less important.

A systematic review of the literature that analyzes twenty-seven UAV interfaces can be found in [95]. This work provides a set of guidelines for future human-swarm interfaces: reduction of the information of UAVs to preserve situational awareness, adaptation to user preferred control and observation modes, design for specific swarms and missions to reduce cognitive workload, allowance of peer-to-peer work with operator and swarm, and use of multimodal feedback to improve situational awareness.

Olson et al. [163] define some guidelines for designing interfaces for supervisory control of UAVs. They point out the necessity to provide the operators with knowledge about the goals of UAVs, in order to make their actions easy to understand. Additionally, the displays must reduce the amount of information that is visible and guide the operator attention to the critical information. Furthermore, they indicate the need to provide the operators with methods to intervene during the mission, such as replanning or reprogramming systems.

Scholtz et al. [214] compare three interfaces in the context of a USAR competition: one that uses augmented reality, other one that only shows the images of two cameras, and a last one that shows both information and video. The conclusions of this experience are the following requirements: the interface should reference the position of robot in the environment, show robot health and status, integrate the information of sensors in the same display, self inspect the robot body, and present the relevant information in the mission context.

Other requirements for interfaces in USAR missions are the use of fluid videos with high-quality images, the integration of multiple robots and operators, and the reduction of required training or knowledge [125].

The Astute project applies the concept of map-centric interfaces to emergency scenarios [9]. These interfaces combine multimodal interactions and adaptation to situation, in order to provide the operators with relevant information in proper mode. The studies point out that map-centric displays easily convey complex information to field operators.

Other proposals for multi-agent systems' interfaces can be found in [157]: a unique window with relevant information, a user-friendly layout, the selection of information for aiding decision making, a rapid user interaction, and the required information for a dynamic change of roles.

Finally, several metrics to evaluate interfaces in the context of multi-robot missions can be found in [214]. The proposed variables are task effectiveness (time to accomplish task), neglect time (performance decline when operator is not attending), robot attention demand (fraction of time that operator is paying attention to robot), free time (fraction of time that operator does not pay attention to any robot), fan out (estimation of number of robots that operator can manage at once) and interaction effort (time to interact plus cognitive demand).

2.5.2 Multimodal interfaces

A review about the problems of teleoperation and the effects of multimodal displays can be found in [33]. The problems are the limited field of view, difficulty of perceiving orientation, complexity of managing diverse cameras with different points of view, degradation in depth perception, degradation in received video, time delays in perception and actuation, and reduction of accuracy with motion.

As reported, the effects of multimodal displays in teleoperation performance are diverse. The combination of visual and audio spatial information leads to an enhancement of situational awareness [219]. Haptic displays can be applied in visual interfaces to provide the alerts with the locations and speeds of the robots [211]. The influence of visual, aural and haptic feedbacks on the spatial ability of operator is significant, but their effects in the performance of teleoperation are not clear [126].

Multimodal commands offer even more possibilities than multimodal displays. The idea is to combine voice, touch and gestures to reach simple, natural and fast interactions between operators and robots [150]. The literature contains multiple approaches for commanding multi-robot missions by means of speech commands [110]. Additionally, the gesture commands are employed in this context, applying not only hand gestures [141], but also combinations of face poses and hand gestures [156].

2.5.3 Immersive interfaces

An interesting resource to deal with human factor problems like the lack of situational awareness is immersion. Immersive interfaces take advantage of multiple technologies (e.g. cameras and glasses) and disciplines (e.g. computer vision) to introduce operators in scenarios. There are three main immersive technologies: virtual reality (VR), augmented reality (AR) and mixed reality (MR).

Virtual reality generates artificial environments based on real or imaginary scenarios and integrates the elements that take part in the missions: robots, targets, threats... This technology uses images, sounds and touch to simulate the presence of operators in scenarios and allow them to control and monitor robots. The first studies of VR interfaces pointed out they could improve the situational awareness of operators [203, 192].

Augmented Reality integrates relevant information about the mission in videos streamed by the robots or from the operator's point of view. This information usually includes maps, terrain elevation, obstacles, robot paths, target locations and other data [32]. Multiple experiments show the potential of AR video stream in comparison to conventional video stream in the context of multi-robot missions [54].

Mixed Reality is a combination of AR And VR that creates new scenarios and allows the interaction with both real and virtual objects. Although the application of this technology to immersive interfaces looks promising, there are only a few cases of use in the context of robotics [64].

2.5.4 Adaptive Interfaces

As previously introduced, other approach to address human factor challenges is adaption. Adaptive interfaces apply data mining and machine

learning algorithms to adapt the information to mission and operator. The objective is to provide the agents with the information that they require for making decisions and executing actions, improving their speed, accuracy, understanding, coordination and workload [89].

Some guidelines to design intelligent adaptive interfaces (IAIs) can be found in [97]. They propose the use of multiple types of models (knowledge, operators, robots, tasks, world...) to manage the information of missions. Additionally, they define an adaption process with four steps: knowledge acquisition (what is happening?), attention (what adaptation is going to occur?), reasoning (why is it necessary?) and decision making (how is it happening?).

Chapter 3

Mission modeling

I'm afraid. I'm afraid, Dave. Dave, my mind is going. I can feel it. I can feel it. My mind is going. There is no question about it. I can feel it. I can feel it. I can feel it. I'm a... afraid. Good afternoon, gentlemen. I am a HAL 9000 computer. I became operational at the H.A.L. plant in Urbana, Illinois on the 12th of January 1992. My instructor was Mr. Langley, and he taught me to sing a song. If you'd like to hear it I can sing it for you.

HAL 9000 (2001, A Space Odyssey)

This chapter addresses the objective of selecting the relevant information according to the mission state. For this purpose, multi-robot missions must be modeled and the models must be used to determine their states. The proposal of this chapter is to use the data generated by robots during missions to automatically build the models.

The chapter collects and expands the content published in the following articles:

- Roldán, J. J., del Cerro, J., & Barrientos, A. (2015, June). A proposal of methodology for multi-UAV mission modeling. In Control and Automation (MED), 2015 23th Mediterranean Conference on (pp. 1-7), IEEE [184].
- Roldán, J. J., Olivares-Méndez, M. A., del Cerro, J., & Barrientos, A. (2017). Analyzing and improving multi-robot missions by using process mining. *Autonomous Robots*, 1-19 [188].
- Roldán, J. J., del Cerro, J., & Barrientos, A. (2017). Using process mining to model multi-UAV missions through the experience. *IEEE Intelligent Systems*, 32(4), 40-47 [186].
- Roldán, J. J., Garcia-Aunon, P., del Cerro, J., & Barrientos, A. (2016, October). Determining mission evolution through UAV telemetry by using decision trees. In Systems, Man, and Cybernetics (SMC), 2016 IEEE International Conference on (pp. 000188-000193), IEEE [189].

The rest of the chapter is organized as follows: Section 3.1 describes the resources used to solve the problem: modeling techniques and process mining, Section 3.2 addresses the systematic methodology developed

to analyze multi-robot missions, Section 3.3 searches the best algorithm to discover mission models, Section 3.4 looks for techniques to estimate the evolution of the mission, Section 3.5 describes the multi-robot missions performed to generate realistic data, Section 3.6 shows the results of applying the methodology to the data obtained from experiments, and Section 3.7 summarizes the main contributions of this study.

3.1 Resources

This section addresses the resources used for multi-robot mission modeling: Section 3.1.1 describes the architecture that was established to analyze multi-robot missions, Section 3.1.2 analyzes multiple modeling techniques and selects the most appropriate for multi-robot missions, and Section 3.1.3 describes the discipline of process mining that allows to automatically discover models from event logs.

3.1.1 Mission architecture

On the one hand, a multi-robot mission can be analyzed from the point of view of a complex system. In other words, there are some objectives that may appear and disappear, as well as a set of resources that may be assigned or available during the mission. Therefore, this kind of mission can be addressed as a problem of allocating resources for achieving objectives.

On the other hand, a multi-robot mission can be studied from the perspective of a robot operation. There are a set of objectives defined before the mission and a set of tasks that are launched to achieve them. Therefore, the mission can be treated as series of tasks that consist of sequences of actions.

An architecture for multi-robot missions must consider both high and low-level concepts. The high-level concepts are related to the mission perspective as a complex system: the mission consists of tasks that use resources to achieve objectives. Meanwhile, the low-level concepts are related to the mission perspective as a robot operation: the mission consists of tasks that are sequences of states with actions. These concepts are presented below:

- Mission: A mission is a set of objectives that requires a series of resources: e.g. "Fire control in a country".
- Objective: An objective is a result that must be achieved for the completion of a mission: e.g. "Fire surveillance in a region" and "Fire extinguishing in a location".
- Resource: A resource is an element (robot, payload, people, object, place...) that is available for achieving an objective and/or performing a task: e.g. "robot", "camera", "policeman" and "airport".
- Task: A task is a set of actions performed to achieve an objective: e.g. "Fire surveillance", "Fire extinguishing" and "Fire monitoring".
- State: A state is a situation of a task that is defined by some actions and observations: e.g. "Take-off", "Go to mission area" and "Emergency".

- **Transition:** A transition is the change between two states when some conditions are satisfied: e.g. "Transition from 'Take-off' to 'Fly to mission area' when 'Drone altitude is higher than 1,000 feet'" and "Transition from 'Fly to mission area' to 'Emergency' when 'Drone battery is low'".
- **Action:** An action is a robot maneuver or payload utilization: e.g. "Drone goes to waypoint (lat., long., alt., time)" and "Drone camera takes a picture".
- **Observation:** An observation is an information that can be used to determine the state of a task: e.g. "Drone altitude is higher than 1,000 meters" and "Drone battery is low".

This architecture allows addressing the multi-robot mission as two independent problems. The first one (mission level) is related to the generation of tasks that apply the resources and reach the objectives. Meanwhile, the second one (task level) considers the execution of each task as a sequence of actions performed by single robots and their payloads.

A set of realistic multi-robot scenarios was studied in order to identify a list of basic tasks. These scenarios include both civilian and military missions and different applications such as fire control or maritime surveillance. The basic tasks are collected in Table 3.1, where "area of interest" means "the area defined by the coordinates of the consecutive vertices", "description" includes "class, features and potential location" and information refers to "images, videos, measures or other things".

- **Surveillance:** This task involves covering an area with a certain pattern of movement to detect some potential objectives.
- **Mapping:** This task involves covering an area with a certain pattern of movement to build a map.
- **Reconnaissance:** This task involves visiting a number of points to check potential objectives.
- **Monitoring:** This task involves visiting a point to supervise other tasks or collect information from the environment.
- **Tracking:** This task involves following a mobile target for acquiring information and/or supervising it.
- **Capture:** This task involves visiting a location for picking a resource.
- **Release:** This task involves visiting a location for placing a resource.
- **Support:** This task involves supporting other tasks by providing information of their environments.
- **Maintenance:** This task includes the use of robots to perform operations of maintenance to other robots.
- **Relay:** This task involves visiting a location with a specific equipment to expand the range of communications.

TABLE 3.1: Description of basic tasks

Task	Input	Output
Surveillance	Area of interest Target description	Target list
Mapping	Area of interest	Map
Reconnaissance	List of points Target description	Target list
Monitoring	List of points	Information
Tracking	Target description	Information
Capture	Resource description	Operation result
Release	Objective description	Operation result
Support	Agent description	Information
Maintenance	Resource description	Operation result
Relay	Objective description	Operation result

3.1.2 Modeling techniques

The selection of an appropriate modeling technique is fundamental not only for the mission specification but also for the mission control and monitoring. This technique should generate suitable models for diverse tasks (e.g. surveillance and reconnaissance) that include both regular (e.g. take-off, fly to point and land) and exceptional states (e.g. emergency, failures and threats).

A review of modeling techniques in robotics literature provided the following results:

- Agent-based model (ABM): The agent-based models are used to represent complex systems as a collection of simple agents [73]. They can represent multiple agents, environment, interactions among the agents and actions of the agents over the environment. This technique is frequent in social science, but it is also used in multi-robot systems [55].
- Business Process Modeling Notation (BPMN): The Business Process Modeling Notation is a tool for modeling and representing business processes [49]. BPMN consist of events (similar to transitions), activities (similar to states) and gateways (logical operators and time events). As its name suggests, this technique is applied in business processes but it is not common in other contexts.
- Hidden Markov model (HMM): A hidden Markov model is a stochastic model for Markov processes that manages hidden states [57]. In this model, the state is not observable, but has influence in the observable output. Therefore, the current state can be inferred through the previous states and outputs. A HMM consist of states, transitions and observations, which can be qualitative or quantitative information that define the state. Both the transitions and observations have different probabilities that allow the estimation of the state. HMMs

have been used in the context of robotics to provide robots with capabilities related to intelligence and adaptation ([247] and [239]). However, the use of HMMs in multi-robot missions is not common.

- Petri net (PN): A Petri Net is a mathematical and graphical tool for system modeling [154]. This tool defines a directed bipartite graph with places (circles), transitions (bars), arcs (arrows) - which connect places to transitions or vice versa - and marks (dots) - which are in places, active transitions and define the configuration of network. There are several types of PN: they can be discrete, continuous or hybrid according to their marking. In addition, there are different extensions to be applied in different contexts: e.g. colored PNs, which have tokens that contain information codified in color. They have been widely applied in robotics, including mission control and monitoring applications ([166] and [127]).
- State Machine (SM): A state machine is a well-known and widely used technique for modeling a system behavior [117]. SMs consist of states, which represent the possible situations, and transitions, which manage the changes of states when some conditions are satisfied. This technique has been used in the context of robotics for tasks related to mission specification and robot control [18].
- Tree Model (TM): A tree model is a common modeling tool for different kind of systems [117]. These models can be classified according to their structure in two classes: hierarchical trees and sequential trees. In the hierarchical trees, each element in the n level can be split into multiple elements in the $n+1$ level that are parts of it. In the sequential trees, each node is a decision and each branch from root to leaves is a possible path. The application of these models is quite usual in some fields such as project management and software engineering, but currently it is not explored in multi-robot missions.

An analysis of these modeling techniques was performed in order to discuss which ones are suitable to model multi-robot missions.

ABM is a promising technique for multi-robot scenarios, but presents limitations with complex tasks. Furthermore, if the tasks are performed by single robots, the potential of this technique is not exploited.

TM is useful for systems consisting of a set of predictable decisions that generates a set of potential paths. However, this technique is weak in complex scenarios with multiple agents, such as the ones considered in this work, because it might generate unexpected situations.

BPMN is focused on business processes, but it can be exported to other areas. In fact, this technique is versatile due to it has multiple resources (e.g. the gateways allow multiple kinds of concurrency). Nevertheless, the BPMN models can be converted to PNs [135]: both directly and through an event log.

SM is probably the most well known and widely used technique, but it might not be the best model in the considered scenario. In fact, HMM provides the same functions as SM and further other ones.

In summary, PNs and HMMs are the most interesting techniques and represent two opposite conceptions. The strength of HMMs is the management of uncertainty, while the strength of PNs is the concurrency.

TABLE 3.2: Analysis of modeling techniques

MT	MA	CT	C	U
ABM	+	-	-	-
BPMN	-	+	+	-
HMM	-	+	-	+
PN	-	+	+	-
SM	-	+	-	-
TM	-	-	-	-

The management of uncertainty is useful in the context of multi-robot missions. At least, the robots, other agents and the environment may behave as sources of uncertainty. For instance, communication failures and emerging threats may affect the development of tasks and the performance of monitoring. A robust model against the uncertainty, such as a HMM, can help the operator to manage these situations.

The concurrency is also practical in multi-robot missions. This feature allows to take into account the interactions among robots and other agents in the models of tasks. A model that is able to represent the concurrency, such as a PN, can help the operator to carry out complex missions.

The conclusions of this preliminary study are summarized in Table 3.2, where MT means Modeling Technique, MA means Multiple Agents, CT means Complex Tasks, C means Concurrency and U means Uncertainty. Petri nets and hidden Markov models are promising modeling techniques for multi-robot missions and each one presents a potentiality (PNs can model concurrency and HMMs can work under uncertainty).

3.1.3 Process mining

Process mining is a discipline that involves the discovery, evaluation and improvement of process models through the information collected by event logs [1]. Therefore, this discipline can be located between the conventional process analysis and modeling, and the modern techniques of data mining.

Process mining arose in the context of business processes, but it has expanded to multiple areas as reported in [140, 171, 6, 103]. However, the application of these techniques to the context of robot missions is still unexplored although it looks really promising. An exhaustive review about the process mining topics and trends in the last ten years can be found in [139].

As shown in Figure 3.1, process mining works between processes, models and event logs, and it offers three possibilities: model discovery, reproduction and conformance/enhancement.

- **Model discovery (play-in techniques):** It automatically builds process models from the behaviors collected by event logs. These models usually describe the processes as sequences of activities, but they can also explain them according to their use of time or resources. In robotics, these techniques allow generating mission models through the experience of simulated or field experiments.
- **Model reproduction (play-out techniques):** It automatically generates event logs through the process models. These event logs contain the

FIGURE 3.1: Process mining and applications [1].

different behaviors that are allowed 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 and reduce their costs.

- Model conformance/enhancement (replay techniques): It compares 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 the potential deviations. For instance, to discover states or transitions that are not considered by the model. In robotics, these techniques allow the enrichment of preliminary models with the experience of missions.

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: Unique identifier of the event.
- Activity: Operation performed in the event.
- Timestamp: Date and time of the event.
- Resource: Agent that performs the event.

The most common models in process mining are Petri nets [4], Business Process Model and Notation [49], transition systems [7] and causal nets [230].

3.2 Methodology

This section presents the systematic methodology developed for applying the techniques of process mining in the context of robotics. This methodology allows to model, analyze and improve multi-robot missions through the generated data.

The methodology is summarized in Figure 3.2 and explained in the following sections: Section 3.2.1 addresses the preliminary mission study,

Section 3.2.2 involves the event log preparation, Section 3.2.3 describes the discovery of models, Section 3.2.4 addresses the model evaluation and improvement, Section 3.2.5 reports the analysis from time perspective, and Section 3.2.6 reports the analysis from resource perspective.

FIGURE 3.2: Flowchart of proposed methodology.

3.2.1 Mission study

The first step addresses a preliminary study the mission, in order to acquire the relevant information for its subsequent modeling, analysis and improvement. This information can be acquired by asking and answering some 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?

In addition to this, the objectives of the study must be well defined to perform the appropriate steps and reach the desired conclusions. Some possible objectives for this kind of analysis are listed below:

- A mission study from time perspective, which may allow to reduce duration, detect bottlenecks and improve effectiveness.
- A mission study from agent perspective, which may provide information about the 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 the mining process is applied to other fields. However, two particularities of multi-robot missions must be considered at the time of posing the questions. The 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. The second one is that there are usually fewer cases with more variants in multi-robot missions than in other applications such as industries and services.

3.2.2 Event log preparation

Multi-robot missions share the generation of huge amounts of data with industrial processes or public services. However, these amounts of data are sometimes incomplete, unstructured or stored in multiple logs, which is a challenge for the application of process mining. Therefore, the second step of the methodology is the preparation of adequate event logs. The possible types of event logs are listed below:

- 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 other logs 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, the take-off and landing events of aerial robots can be identified by looking for changes in the altitude.

Section 3.1 described the general form of the event logs in process mining. However, the event logs of multi-robot missions usually have particular features. Additionally, the generation of different event logs lead to the discovery of different models. This can be done by considering multiple criteria for defining cases and resources and can be used to analyze different properties of the missions. The structure of these event logs is described below:

- **Case:** It is a unique identifier that can represent not only the mission repetition to obtain general mission models and study the relations between the agents (e.g. "M17" for the seventeenth mission), but also the mission and agent if mission models particularized to the agents are required to study their behaviors (e.g. "M17A1" for the first agent of seventeenth 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 potential emergencies (e.g. low battery).
- **Timestamp:** It contains the date and time when the event takes place. Nevertheless, in the context of robotics, setting separated starting and finishing times for the events can be a good idea to facilitate the future time analysis.
- **Resource:** It is the robot that performs the task or the operator that generates 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 during the mission by monitoring algorithms, which are executed by the robots and interfaces, or after it by data processing scripts, which can discover them from the telemetries. The event logs are stored in spreadsheets (usually, XLS, ODS or CSV files) and can be edited with *Disco* application [82]. This program can be used to filter the event logs by timestamps (e.g. removing old or recent cases) and activities (e.g. removing the irrelevant activities or cases), among other criteria.

3.2.3 Model discovery

Once the event logs are completed and structured, the next step is to discover the 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 are able to distinguish between common and exceptional behaviors and reduce the number of states and transitions. Furthermore, the event logs only represent relations of precedence between the activities, whereas the models can also represent relations of causality and parallelism.

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, hidden Markov models and state machines. Petri nets and hidden Markov models are the most promising among these models, since they are able to manage concurrency and uncertainty respectively. Petri nets are chosen because they are used in process mining and can be complemented with decision trees to address uncertainty.

There are different types of models according to their application:

- Complete mission model: This model represents the global development of the mission. Additionally, it combines operator commands with robot actions and shows the relations among them.
- Operator model: This model shows the way the operator commands the robots during the mission. Among other things, it allows to detect deviations in the mission and support the operator decisions.
- Agent model: This model considers the actions of robots and allows to study their performance and coordination during the mission. This study can detect bottlenecks in missions and inefficient allocation of resources.
- Combined operator and agent model: This model is obtained by considering as case the mission and agent. It is interesting because integrates both operator and robot actions but segregating them.

Process mining offers multiple discovery algorithms to build Petri nets through event logs [231], such as Alpha miner [3], heuristic miner [234], Integer Linear Programming (ILP) miner [235] and Inductive miner [128]. These algorithms are analyzed in the context of multi-robot missions and tested with data obtained from simulations in Section 3.3.

3.2.4 Model evaluation and enhancement

Once the models of the multi-robot mission are obtained, the next step is their evaluation and improvement.

Process mining usually evaluates the quality of models through four parameters: fitness, simplicity, generalization and precision [26]. 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, so the models should reach a compromise between them.

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 the number of states and transitions that are considered. For instance, if the model is complex or tends to underfitting, the infrequent transitions can be removed, whereas if the model is simple or tends to overfitting, the discarded transitions should be considered.

3.2.5 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 may reveal the bottlenecks of the mission and the tasks that require more effort. Some examples of time analysis are listed below:

- Operator actuation times: This study allows to determine the times required by the operator to execute the commands. 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. 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.

3.2.6 Resource analysis

The resource analysis looks the mission from another point of view 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 [5], 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.

3.3 Model discovery

The objective of this study was to determine the best discovery algorithm of process mining in the context of multi-robot missions. For this purpose, four algorithms with their variants were selected: Alpha Miner (AM), Heuristic Miner (HM), ILP Miner - per casual dependency (ILPM-C) and before-after transition (ILPM-BA) - and Inductive Miner - basic (IM-B), for infrequent behavior (IM-Inf), for incomplete logs (IM-Inc) and exhaustive k-sucessor (IM-EkS). In order to compare these algorithms in a relevant environment, a multi-UAV simulator based on a game was developed.

The rest of the section is organized as follows: Section 3.3.1 describes the simulator developed to generate mission data, Section 3.3.2 explains the algorithms used to discover the mission models and Section 3.3.3 evaluates the models and selects the best algorithm.

3.3.1 Simulations

The simulator considers a competitive scenario with equality of forces, in order to produce adverse conditions and uncertainty sources. Specifically, it poses a multi-UAV game inspired by *Quidditch* (Harry Potter's saga), where two teams of three UAVs coordinated by coaches compete to achieve the maximum score. They have to control three balls: a *Quaffle* that has to be launched to goal to score ten points, a *Bludger* that can be launched to opponents to stop them, and a *Snitch* that has to be captured to score fifty points. Each simulation represents a match and has a duration of 200 seconds. The multiplicity of agents, objectives, strategies, actions and interactions generates a complex scenario, where the techniques and models can be tested under harsh conditions.

This simulator was developed by using Robot Operating System (ROS) [172] to meet the requirements of modularity. ROS allows to build a modular structure with nodes that process the information and topics that perform the communication. Therefore, the simulator can reproduce multiple scenarios just by adding, removing or reconfiguring the nodes. The different nodes, their number and their functions are listed below:

- Coach (1 node per team): It analyzes the information of the game, generate complex tasks for achieving the objectives and allocate them to the robots.
- Robot (1 node per robot): It converts the complex tasks into sequences of basic tasks.
- Task (1 node per robot): It performs the planning and monitoring of basic tasks.
- Navigation (1 node per robot): It computes the path to the waypoints and generates speed commands to follow it.
- Controller (1 node per robot): It executes the speed commands to move the robot.
- Simulator (1 node): It collects all the information, manages the game, time and score, handles the movement of balls and manages the interaction among UAVs and targets.

TABLE 3.3: Situations of the game and tasks that respond to them.

Situation	Complex task	Basic tasks
Begin	Begin	Begin
Attack	Drive	Go to Waypoint
	Pass	Release
	Shoot	Release
	Get Away	Go to Waypoint
	Support	Support
	Bludger	Go to Waypoint-Capture-Release
Defense	Snitch	Sur.-Rec.-Tracking-Capture
	Keep	Go to Waypoint
	Mark	Tracking
	Press	Tracking-Capture
	Block	Go to Waypoint-Capture
	Bludger	Go to Waypoint-Capture-Release
Transition	Quaffle	Go to Waypoint-Capture
	Bludger	Go to Waypoint-Capture-Release
	Snitch	Sur.-Rec.-Tracking-Capture
Emergency	Maintenance	Maintenance
Finish	Finish	Finish

- Recorder (1 node): It stores the information, commands and events of the game in a data base for the subsequent analysis.

The coach node analyzes the situation of the game and classifies it into six categories: *Begin*, *Attack*, *Defense*, *Transition*, *Emergency* and *Finish*. Each of these categories presents a set of possible tasks that can be launched according to the context of the game. For instance, when the team is attacking, the UAV that controls the *Quaffle* can advance with it, pass it to a teammate or kick it to goal. Meanwhile, the UAVs that do not control the *Quaffle* can get away to receive a pass, support their teammates, use the *Bludger* against an opponent or directly search the *Snitch*. Table 3.3 contains all the situations of the game and the potential tasks to respond them. The robot and task nodes decompose the complex tasks into basic tasks (as shows the Table 3.3) and perform their planning and monitoring. The navigation and controller nodes control the movement of robots, as well as the work of sensors and actuators.

The first tests with the simulator were made to analyze the game and optimize the performance of agents. Figure 3.3 shows the evolution of a match with the situations of both teams. These situations are the result of the analysis of mission performed by the coaches. As can be seen, the situations of attack and defense are alternative, which reveals the strategies are working correctly. Figure 3.4 depicts the paths of robots and the evolution of their autonomy. As shown, most of the game is developed in central positions and the robots must perform maintenance operations to keep their autonomy during the match.

The simulator generates mission event logs that contain the events related to UAVs during the mission. Half of the event logs were used for mission modeling, while the rest were used for validating the models. In

FIGURE 3.3: Situations for both teams over a match.

FIGURE 3.4: Path and autonomy of robots in a match.

particular, 96 cases were used for modeling and validation (i.e. a case per UAV, 3 UAVs per team, 2 teams per match and 16 matches).

3.3.2 Algorithms

As previously mentioned, mission models are obtained from event logs by applying discovery algorithms of process mining. The following works are performed with two detail levels: an event log with tasks that is easier to understand, and another one with tasks and states that is more realistic, both of them shown 3.5.

Multi-robot mission event logs have some particularities that are important in model discovery. The presence of loops with one or multiple tasks poses a difficulty for some algorithms, and the existence of transitions that can be performed in both directions cannot be processed by other ones. Finally, the longitude of the traces (the cases have an average of 95 events) complicates the conformance between models and event logs.

As previously explained, there are different discovery algorithms to build Petri nets from an event logs [231]. Specifically, four algorithms and their variants were used to discover the models and their results were compared. These algorithms, which are implemented in the process mining framework *ProM 6*, are explained below:

- Alpha Miner (AM) [3] is the basic algorithm of process mining for discovering Petri nets from event logs. This algorithm is a powerful tool if the model is a structured workflow Petri net. However, when the event log presents short loops, it may leave disconnected states and generate void or duplicated states. In addition, this algorithm and its variants are prone to overfitting.
- Heuristic Miner (HM) [234] is a suitable alternative for mining models because it is able to find the relations between states and avoid the extreme models: both disconnected and flower models (i.e. the models that allow all transitions between all states). However, this algorithm is focused on the main behaviors and may ignore the infrequent and exceptional ones, which can reduce the conformance of models to event logs.
- ILP Miner [235] applies Integer Linear Programming to find the places of the Petri net. In contrast to Alpha Miner, this algorithm does not tend to overfitting, but it can generate underfitted models if it cannot find dependencies. In this work, two variants are used: ILP per causal dependency (ILPM-C) and ILP before-after transition (ILPM-BA).
- Inductive Miner [128] uses a divide-and-conquer strategy, making partitions of states and splitting the log recursively until it finds the model. In this work, four variants are used: Inductive Miner basic (IM-B), Inductive Miner variant for infrequent behavior (IM-Inf), Inductive Miner variant for incomplete logs (IM-Inc) and Inductive Miner exhaustive k-successor (IM-EkS).

Table 3.4 shows a preliminary evaluation of the models generated by the algorithms. This evaluation shows if the miners generate suitable models or, by contrast, they are not able to obtain them: for example, the algorithm

TABLE 3.4: Preliminary results of discovery algorithms.

Algorithm	Simplified event log	Complete event log
AM	Disconnected model	Model
HM	Model	Model
ILPM-BA	Flower Model	Model
ILPM-C	Flower Model	Model
IM-B	Model	Model
IM-Inf	Model	No model
IM-Inc	Model	Model
IM-EkS	Model	No model

TABLE 3.5: Alignment of models with tasks to event logs.

Algorithm	Trace fitness	Move-model fitness	Move-log fitness
HM	0	1	0
IM-B	0.999	0.999	1
IM-Inf	0.624	1	0.624
IM-Inc	0.569	1	0.569
IM-EkS	1	1	1

does not discover a model (noted as 'No model'), discovers a model where some states are disconnected (noted as 'Disconnected model'), or generates a flower model (noted as 'Flower model').

3.3.3 Comparison

This section involves the evaluation of the models discovered in the previous one. As already pointed out, the evaluation uses different event logs than the discovery, in order to detect and avoid the overfitting.

A first approach to evaluate model quality is to obtain the alignment between models and event logs. For this purpose, the event log is replayed in the model to compare the traces [2]. If the model is correct, most of the movements on the event log correspond to movements on the model, and the fitness is close to one. Otherwise, most of the movements are only in the model or only in the log, and the fitness is close to zero.

The "Replay log on PN for conformance analysis" plug-in of *ProM 6* was used to obtain the alignments of Tables 3.5 and 3.6. These tables only contain the algorithms that generated suitable models for each event log. As shown, the behaviors of algorithms in simplified (tasks) and complete (tasks and states) logs are different. This fact is explained by the properties of logs: the simplified log contains loops of one task, whereas the complete log only contains loops of multiple states.

In the log with tasks, the best result is obtained with IM-EkS, but the result with IM-B is similar (trace fitness of 0.999 vs 1). The result of HM is noteworthy because this algorithm does not correctly and completely reproduce any trace. In the log with tasks and states, the best result is obtained with ILPM-C, but ILPM-BA and IM-B also provide good results (trace fitnesses of 0.989 and 0.999 versus 1). Therefore, the best algorithm in both event logs is IM-B, which can be applied to both logs reaching fitnesses close to 1.

TABLE 3.6: Alignment of models with tasks and states to event logs.

Algorithm	Trace fitness	Move-model fitness	Move-log fitness
AM	0.847	0.099	0.649
HM	0.209	0.655	0.143
ILPM-BA	0.989	0.990	0.999
ILPM-C	1	1	1
IM-B	0.999	0.999	1
IM-Inf	0.915	0.980	0.925

The alignment only considers a perspective of model quality (fitness) and it does not take into account another relevant issues (complexity, underfitting and overfitting). Therefore, an approach that takes into account these measurements is required to make conclusions. Process mining usually evaluates the quality of the models through four parameters: fitness, simplicity, generalization and precision [26].

- **Fitness:** It is the ability of the model to explain observed behavior, which is the fundamental requirement of models.
- **Simplicity:** It is related to the complexity of models; a simple model is easier to understand and employ than a complex one.
- **Generalization:** It measures the capability of the model to avoid overfitting. Overfitting occurs when the model is too adapted to the event log and fails with new samples.
- **Precision:** It measures the capability of the model to avoid underfitting. Underfitting appears when the model is too general and can represent any behavior.

Fitness and simplicity are usually opposite concepts: a model with good fitness may be complex, whereas a model with good simplicity can be useless. In the same manner, generalization and precision are commonly opposite variables: a model with good generalization may be underfitted, whereas a model with good precision can tend to overfitting. Consequently, a compromise between these pairs of variables should be reached in order to obtain a high-quality model.

These concepts are applied to this scenario through some simple equations, which take into account the available information: number of states considered by model, number of states with and without activities, number of states not considered by model, number of states duplicated in the model and number of transitions considered by model.

Equation 3.1 determines the value of simplicity through the weighted sum of the total states (S_i) and transitions (T_i). The weights (f_t and f_s) depend on the ratio of states to transitions and look for normalizing the value between 0 and 1.

$$Simplicity_i = \left(f_t * \frac{S_i}{S_m} + f_s * \frac{T_i}{T_m} \right)^{-1} \quad (3.1)$$

TABLE 3.7: Quality parameters for the Petri nets of simplified event log.

Algorithm	Fitness	Simplicity	Generalization	Precision
HM	0	0.357	0.380	0.641
IM-B	0.999	0.346	0.345	0.663
IM-Inf	0.624	0.876	0.852	0.166
IM-Inc	0.569	0.725	0.667	0.301
IM-EkS	1	0.361	0.256	0.729

TABLE 3.8: Quality parameters for the Petri nets of complete event log.

Algorithm	Fitness	Simplicity	Generalization	Precision
AM	0.847	0.640	0.562	0.568
HM	0.209	0.329	0.275	0.596
ILPM-BA	0.989	0.863	0.860	0.238
ILPM-C	1	0.637	0.564	0.763
IM-B	0.999	0.284	0.249	0.674
IM-Inf	0.915	0.711	0.552	0.230

Equations 3.2 and 3.3 compute the values of generalization and precision respectively. Generalization will be better when there are less transitions (T_i) and empty states (VS_i), whereas precision will be better when there are more transitions (T_i) and less ignored states (DiS_i). The variables T_m , VS_m and DiS_m are the average values of transitions, empty states and ignored states in the discovered models; whereas the constants k_{g1} , k_{g2} , k_{p1} and k_{p2} are used to balance the components of equations and normalize the values between 0 and 1.

$$Generalization_i = k_{g1} * \frac{VS_m}{VS_i} + k_{g2} \frac{T_m}{T_i} \quad (3.2)$$

$$Precision_i = k_{p1} * \frac{DiS_m}{DiS_i} + k_{p2} * \frac{T_i}{T_m} \quad (3.3)$$

Tables 3.7 and 3.8 show the values of fitness, simplicity, generalization and precision of the models. These metrics have been adapted to the context of multi-robot mission models, in order to extract valuable information for making conclusions.

In the simplified event log, the IMs are clearly the best algorithms. Specifically, IM-EkS and IM-B have better fitness, but IM-Inf and IM-Inc have better simplicity and IM-B has the best balance between generalization and precision (it does not tend to underfitting nor overfitting). IM-B algorithm is selected because its fitness and simplicity are near IM-EkS, while its balance between generalization and precision is better, although it also tends to overfitting.

In the complete event log, ILPM and variants show better performance than the rest of algorithms. In particular, ILPM-C reaches the best fitness, while ILPM-BA obtains a similar fitness but is quite simpler. In addition, both algorithms have an acceptable relation between generalization and precision, as well as they avoid both underfitting and overfitting. Thus, the ILPM-BA is selected over the ILPM-C and the rest of algorithms.

FIGURE 3.6: Petri net obtained by Inductive Miner for the log with tasks.

Finally, if a single algorithm has to model both logs, IM-B is clearly the best option: it is the only algorithm that is able to generate suitable models with competitive performances in both contexts. Figure 3.6 shows the Petri net obtained by using this algorithm with the event log with tasks.

3.4 Model evolution

The objective of this study was to develop a model-based method to estimate mission state through robot telemetry. This system integrates Petri nets for determining the state and decision trees for estimating the evolution. Both the Petri nets and the decision trees are generated automatically from the telemetry of previous missions. The whole system is validated by monitoring a set of UAV missions in a realistic simulator.

This work assumes the mission state has to be estimated in the ground station instead of received from the air vehicles. This assumption is justified because the required information of the mission may not be available or may not be complete. Some possible reasons are listed below:

- The air to ground data link is temporarily interrupted, due to communications failure or range limitation.
- The control and the monitoring of the air vehicle are performed by different ground stations.
- The air vehicle is executing semi-automatic commands sent by the ground station.
- The air vehicle is not following the mission plan and performing the assigned task.

In this case, the UAVs do not report any data about the status of the tasks that they are performing. Therefore, mission monitoring must be addressed by using only mission model and vehicle telemetry.

This work starts from an analytical model that represent the possible sequences of tasks in the mission. A complete set of simple simulations of the mission is performed in order to generate telemetry. Decision trees are

generated to relate this telemetry to the model and determine the mission state at any time. Afterwards, another set of complex simulations is carried out in order to check the performance of decision trees.+

The rest of the section is organized as follows: Section 3.4.1 describes the experiments, Section 3.4.2 explains mission modeling and Section 3.4.3 addresses mission monitoring.

3.4.1 Experiments

Since the simulations of UAV missions are an essential part of this work, this section describes the experimental layout. The following sections show with detail the missions performed, the simulators used and the telemetry generated.

Missions

A set of missions of Intelligence, Surveillance, Target Acquisition and Reconnaissance (ISTAR) are performed for mission modeling and monitoring. In these missions, the UAV starts from an airport, searches the potential targets, checks them in case of detection, and returns to the airport. These are the tasks identified for these missions:

- Begin: Preparation and take-off.
- Go to area: Fly to mission area.
- Surveillance: Fly over an area to detect targets.
- Reconnaissance: Fly over a point to check targets.
- Go to base: Fly to base.
- Finish: Landing and shutdown.
- Contingency: Fly over a point waiting for a runway.
- Emergency: Diagnose and correct the fault.

The mission plan includes the mission areas and the routes of navigation and contingency. These routes are defined by lists of waypoints with latitude, longitude and altitude. Nevertheless, the operators can send semi-automatic commands to modify the flight in order to achieve the objectives: e.g. to change the speed or altitude and to perform a loiter (flight in circles over certain location).

Thirty-two simulations were performed to generate telemetry for mission modeling. These simulations cover eight flight paths, three mission areas and two scenarios (as seen in Figure 3.7). In addition, each simulation contains a certain behavior; for instance the detection of targets followed by their reconnaissance, use of contingency routes or deviation from planned path (as shown in Figure 3.8).

On the other hand, four simulations with different flight paths and events have been used for mission monitoring. These simulations allow to check the conformance between model prediction and real evolution.

FIGURE 3.7: Two scenarios considered for ISTAR missions.
Map data ©2014 Google, based on BCN IGN Spain.

Simulator

Two simulators were used to reproduce these missions: one for model training and another for model validation.

- Training simulator: This simulator is developed and works under Matlab/Simulink environment. The aircraft model is based on a high wing aircraft with conventional empennage. It has an empty weight of $460kg$, $100kg$ of maximum fuel weight, a reference surface of $10m^2$ and a wingspan of $9.6m$. All the aerodynamic and inertial properties are similar to other aircrafts of its class, flying at $Re \sim 1.5 * 10^6$ and $M < 0.2$. This model has been implemented with a stabilization and path-following control, providing as output several telemetry variables.
- Validation simulator: This simulator is the kernel of SAVIER Demonstrator, which is the environment where all the developments of SAVIER Project are validated. The whole demonstrator also integrates GCS environment, mission planning and monitoring application, payload simulator and other components. The simulator can reproduce the realistic dynamic and kinematic behavior of medium-size tactical UAV. It provides a complete and realistic telemetry based on NATO's STANAG 4586 that is described in next section.

Information

The communications among UAV and GCS follow the NATO's STANAG 4586 [10]. This standardization aims to promote the interoperability of present and future UAV systems within the NATO rules. For this purpose, the document establishes a series of messages to exchange the telemetry of UAVs and the commands of GCSs.

Among the hundreds of messages with thousands of variables defined by STANAG 4586, the simulator provides 11 messages with 108 variables.

FIGURE 3.8: Examples of simulations with different scenarios, paths and events (loiters, contingencies and emergencies).

This telemetry contains the following information related to the UAV health and flight:

- Operating mode (1 variable): It specifies the method for controlling the vehicle, which can be manual or automatic.
- Air/ground relative states (15 variables): It contains information about the relative state of UAV to GCS, such as the wind speed and altitude.
- Inertial states (17 variables): It defines the inertial state of the UAV through the variables of position, linear speed, linear acceleration, attitude and angular speed.
- Waypoint states (20 variables): It describes the path following status through the information of previous, target and next waypoints (latitude, longitude, altitude, time and speed).
- Air comms (6 variables): It reports the status and mode of the communications system, including all the data links.
- Air/ground additional relative states (12 variables): It contains additional information about the relative state of UAV to GCS, such as the airspeed and temperature.
- Operating states (10 variables): It specifies the semi-automatic commands of the GCS that are executed by the UAV, e.g. altitude, heading or speed.
- Additional air segment data (3 variables): It contains information about weight on wheels and flight modes.
- Fuel system (4 variables): It reports the information of consumed and remaining fuel.
- Mission ID (6 variables): It describes the current mission of the UAV to the GCS.
- Ground Comms Status (14 variables): It reports the status of the communications between UAV and GCS, including both the features (frequency, channel and antenna) and the alarms (failures).

3.4.2 Mission modeling

As previously stated, Petri nets are a suitable alternative for mission modeling, specially when multi-robot systems instead of single robots are considered. In contrast to other models, Petri nets can manage the concurrency [154], which is a key issue in the context of complex tasks with multiple agents and strong coordination.

Nevertheless, basic Petri nets work with deterministic transitions, so their performance could decrease under uncertainty. The literature contains several proposals to provide the Petri nets with the ability to manage these situations. Two examples are the use of stochastic Petri nets [34] and the combination of Petri nets and decision trees [198]. The first method is widely known and used in the context of robotics missions, but the second

FIGURE 3.9: Event log of mission.

one allows to use the telemetry to determine the mission state and evolution.

The combination of Petri nets and decision trees is common in process mining [1]. In this discipline, a decision point is a place of the Petri net that can be followed by multiple transitions [198]. If this decision between some alternatives depends on the variables of the process, the problem could be solved by generating and managing a decision tree. These models use the experience to establish relations between some response variables and some predictor variables, which in this case are the following place of Petri net and the variables of environment respectively.

In the context of robotics, a Petri net can represent the mission: each place corresponds to a task and each transition to a change of task. Then, a decision point occurs when two or more tasks can follow the current one. This work aims to generate decision trees linking the tasks of missions to the telemetry of UAVs.

Petri net

An event log of the mission was generated by the training simulations. This event log is shown in Figure 3.9 and contains all the possible sequences of tasks. Later, Inductive Miner [128] was applied to obtain the model shown in Figure 3.10. This discovery algorithm of Process Mining is implemented in *PROM 6.5.1* toolkit and provides the Petri net of the mission.

As seen in Figures 3.9 and 3.10, the model has two decision points: one in Surveillance between Reconnaissance, Emergency and Go to base; and another in Go to base between Contingency and Finish. These decision points are mined with decision trees in the following sections.

FIGURE 3.10: Petri net of mission.

Conversion of variables

Some considerations about the variables must be attended before developing the decision trees. In general, the best strategy is to work with dimensionless, discrete and relative variables.

On the one hand, if we carefully examine the variables of telemetry, we find that some of them are not significant. That is the case of some absolute variables (e.g. latitude and longitude), which depend on the scenario and cannot be used to make decisions. Nevertheless, this variables can be converted to relative variables (e.g. latitude and longitude referred to mission area), in order to exploit them to make decisions.

On the other hand, if we think about the control and monitoring of UAV missions, we will realize that some complex variables are more interesting than the simple ones. For instance, a variable that shows if the UAV is in the mission area can be computed by comparing the location of UAV to the vertices of the area, and it is more useful than these separated variables.

Some variables considered in the work are the following:

- Relative Latitude and Longitude: Latitude and longitude referenced to mission area (the north border is 100%, the south border is -100%, the east border is 100%, the west border is -100%, and the center is 0%).
- Relative Altitude: Altitude referenced to mission plan (the sea level is 0% and the maximum altitude is 100%).
- UAV in Route: Is the UAV following the route?
- UAV in Area: Is the UAV in the mission area?
- UAV in Loiter: Is the UAV performing a loiter?
- Distance to Base and Contingency: Absolute straight distances between UAV and base and UAV and contingency points.
- Covered Distance: Absolute distance covered by UAV in mission.

Decision tree

Decision trees are a common tool of classification and regression in data mining and machine learning [101]. These models are able to establish relations between some target variables and some observed variables [183]. The leaves of the trees represent the labels of target variables, whereas their branches correspond to the values of the observed variables that lead to these labels.

The literature contains multiple algorithms to obtain decision trees. These algorithms can be classified in two categories [136]: methods that generate

FIGURE 3.11: Procedure for generating the mission trees.

single trees through greedy approaches (e.g. algorithms that search minimum cost [136]) and methods that generate multiple trees without them (e.g. evolutionary algorithms [16]).

The procedure for generating mission trees applied in this work is depicted in Figure 3.11. The target variable is the task that follows the current task, whereas the observed variables are the primary and secondary variables obtained from the telemetry. Therefore, the first step is filtering the telemetry to get only the samples that correspond to the possible tasks. Then, the decision trees are obtained from these samples by a recursive partitioning algorithm. This algorithm is implemented in *RapidMiner Studio 7.0* and consists of multiple steps, where it applies all the variables, computes their performance, selects the variable with best performance and split the samples according to it. In this work, the performance has been evaluated by means of the information gain (the entropy of the samples), gain ratio (the gradient of information gain) and Gini index (a measure of the impurity of samples).

This procedure has been used to obtain two decision trees for the two mentioned decision points. In the first decision point (DP1 in Figure 3.9), the task could change to Reconnaissance, Emergency and Go to base. The decision tree obtained for this point from the telemetry of previous missions is shown in Figure 3.12. As seen, a dozen of variables (e.g. attitude, speed, distance to base, relative location...) determines the following task. In the second decision point (DP2 in Figure 3.9), which separates Contingency and Finish, the algorithm provided a simpler decision tree. The task is Finish if the altitude is lower than 600 meters and Contingency in the opposite case.

3.4.3 Mission monitoring

The set of training simulations is shown in Figure 3.13. As seen, these simulations include several sequences of tasks and unexpected situations (emergencies and contingencies).

Additionally, the set of validation simulations is depicted in Figure 3.14. For each simulation, the graphic shows three sequences of tasks, the first one is the real behavior (noted as "Real"), the second one is the behavior observed by Petri net (noted as "Model-PN") and the third one is the behavior

FIGURE 3.12: Decision tree for the decision point in Surveillance between Reconnaissance, Emergency and Go to base.

FIGURE 3.13: Evolution of simulations.

FIGURE 3.14: Evolution of simulations.

predicted by decision trees (noted as "Model-DT"). This Figure contains much information about the performance of models.

On the one hand, it can be seen that Petri net works correctly except at a certain moment in the third mission. At this time, the real mission presents a situation of emergency that is solved quickly (the UAV deviates from path but immediately retrieves it), but the model does not consider this transition of back from Emergency to Surveillance. Therefore, the Petri net keeps the situation of emergency until the vehicle comes back to base.

On the other hand, some features can be appreciated in decision trees. In the states with a single possible evolution, the prediction is obvious, but in the decision points, their behavior can be observed. Although the application of these models to decision points is successful, the predictions before these points are not accurate and vary during the execution. This behavior is extreme in the first mission, where the decision tree makes the correct prediction (Reconnaissance) only a few seconds before the transition (from Surveillance to Reconnaissance). This fact is due to the decision trees, which are trained in the decision points under a certain conditions and cannot work at other moments under other conditions.

3.5 Experiments

The experiments were designed to reproduce a multiple robots-single operator scenario, in order to generate valuable data to apply the developed methodology. In this way, the techniques can be applied to study the performance of the mission, the commands of operator, the actions of robots and the interactions among all of 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 monitored the mission and commanded the robots by means of a simple interface. The Figure 3.15 shows the layout of the experiments with its different components: scenario, robots, computers and interface.

FIGURE 3.15: Layout of experiments.

3.5.1 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 x 6.70 m x 5.00 m was recreated in the laboratory. As shown in Figure 3.16, this scenario had three areas of interest: a fixed circle that represents the base where the robots start and finish the mission, a fixed square that represents the reservoir where the robots can load the water, and the fire that changes its location in the mission and 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.
- **Tracking:** The robot follows the suspect across the scenario at low altitude (0.8 m).
- **Finish:** The robot lands and switches off.

FIGURE 3.16: Experiments: (a) Scenario with robots, (b) Mission execution.

3.5.2 Robots and equipments

The experiments employed two *Parrot AR.Drone 2.0* quadcopters [119] to perform the mission, and a *KUKA Youbot* [21] 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 525mm x 515mm x 120mm, a weight of 420g and an autonomy of around 18 min with 1,500 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 Figure 3.15, 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 580mm x 380 mm x 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 [44]. 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).

3.5.3 Interface

An intuitive operator interface was developed to receive the information of mission and send the commands to the robots. This interface can show real-time information from the UAVs, as well as images from their cameras. Figure 3.17 shows the main window, which provides the information of mission and fleet, and the command window, which allows the operator to generate various types of commands. Additionally, this interface is able to record all the information and commands during the missions.

FIGURE 3.17: Multi-Robot Mission Interface: commanding window in front and monitoring window behind.

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 waypoints to UAVs 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.

3.5.4 Integration

The robots, the motion capture system and the interface were integrated by means of Robot Operating System (ROS) [172]. 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.

3.6 Validation

This section describes the application of the methodology of Section 3.2 to the experiments of Section 3.5. For this purpose, it follows the same scheme of Section 3.2: Section 3.6.1 studies the mission, Section 3.6.2 prepares the event log, Section 3.6.3 discovers the models, Section 3.6.4 evaluates and improves them, Section 3.6.5 analyzes it from time perspective and Section 3.6.6 from resource perspective.

3.6.1 Mission study

First of all, the fundamental information of mission is collected by answering the previously asked 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 a 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 this 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.

In addition, the objectives of the analysis must be defined. In this case, the following studies of the multi-robot mission controlled by the human operator are interesting:

FIGURE 3.18: Procedure for generating event logs through telemetry and commands.

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

3.6.2 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 Figure 3.18, 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 3.9. In the first step, the events that are observed during the planning and execution of previous missions are collected. 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 Figure 3.19: this is

TABLE 3.9: Events generated from telemetry.

Activity	Begin condition	Finish condition
Take-off	$State = 6$	$Z > 0.5m$
Land	$State = 8$	$Z < 0.1m$
Waiting	$t = 0s$	$\ (V_x, V_y, V_z)\ > 0m/s$
Stopped	$\ (V_x, V_y)\ < 0.05m/s$	$\ (V_x, V_y)\ > 0.05m/s$
Capture	$\ (X, Y) - (X_c, Y_c)\ < 0.2m$	$\Delta t = 2s$
Release	$\ (X, Y) - (X_r, Y_r)\ < 0.2m$	$\Delta t = 2s$
Detection	$\ (X, Y) - (X_t, Y_t)\ < 0.2$	$\Delta t = 1s$
Tracking	$\ (X, Y) - (X_{mt}, Y_{mt})\ < 0.5m$	$\ (X, Y) - (X_{mt}, Y_{mt})\ > 0.5m$
Quadrant	$(X, Y) \in Q_i$	$(X, Y) \in Q_j$
Center / Periphery	$\ (X, Y)\ < 1.0m$	$\ (X, Y)\ < 1.2m$
Fire area	$\ (X, Y) - (X_r, Y_r)\ < 0.5m$	$\ (X, Y) - (X_r, Y_r)\ > 0.8m$
Water area	$\ (X, Y) - (X_c, Y_c)\ < 0.5m$	$\ (X, Y) - (X_c, Y_c)\ > 0.8m$
Accident	$\ (V_x, V_y)\ = 0m/s$	$Z = 0m$

(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

FIGURE 3.19: Examples of events generated by the location of UAVs.

the case of the changes of quadrant, transitions between center and periphery, entrance and exit of relevant areas, and distance between UAVs. In the second step, the beginning and finishing conditions of these events are established. These conditions set 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, 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 are structured into events in contrast to the telemetries of UAVs. Therefore, the preparation involves adding 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 Figure 3.18, 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). Three fields were added to the conventional event logs: 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, the beginning and finishing times were separated to easily estimate the duration of events.

As previously mentioned, thirty-six multi-robot missions were performed and twenty of them were used to study some aspects about the robots, operator and interface. Therefore, the complete event log of these missions has 2,023 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 (two drones and one operator). This event log is loaded with *Disco* [82], in order to generate graphic representations and to 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 be hard 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 3.20 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 3.10 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.

(A)

(B)

FIGURE 3.20: 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.

TABLE 3.10: Event logs considered in the work and their main features.

Event Log	Cases	Events	Activities	Resources
EL-A	20	2,023	32	3
EL-B	20	565	14	3
EL-C	20	260	9	1
EL-D	20	349	7	2
EL-E	20	1,116	11	2
EL-F	20	1,763	23	2
EL-G	60	2,023	32	3
EL-H	60	2,023	62	3
EL-I	4	379	31	3

- 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.
- 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.

TABLE 3.11: Models discovered from event logs.

Event Log	Model	Traces	Deviations	Ratio
EL-A	M-A	2,063	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	1,156	47	95.93%
EL-F	M-F	1,802	106	94.12%
EL-G	M-G	2,140	627	70.70%
EL-H	M-H	2,137	706	66.96%
EL-I	M-I	57	24	42.11%

3.6.3 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, the Inductive Miner algorithm implemented in the *Mine with Inductive Visual Miner* plugin of *ProM 6* [129] is used.

Table 3.11 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 the behavior (e.g. *Surveillance - Reconnaissance - Tracking*) and once the behavior (e.g. *Surveillance - Tracking*), the model could represent only the common behavior and ignore the infrequent one. In this case, if the event log is played over the model, sixty-three traces will be obtained (twenty one times *Surveillance, Reconnaissance and Tracking*) with 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.

3.6.4 Model evaluation and enhancement

The fourth step is to evaluate and improve the discovered models. For this purpose, the notions of fitness, simplicity, generalization and precision can be applied. 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

FIGURE 3.21: 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.

shown in Figure 3.21. It can be estimated that this model has a good fitness because the deviations suppose around 5% of traces according to Table 3.11, but in a quick view it can be seen that it is not simple. Additionally, the 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, the robots always perform their tasks after the operator commands them. 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, the causes of accident can be determined by studying the events that precede it. Nevertheless, the complete model does not show the events that happen before the accident.

TABLE 3.12: 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%

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, a diverse set of models were discovered: some of them were general of mission and the rest were focused on operator or robots.

The operator model also presents an interesting behavior. As shown in Table 3.12, the ratio between deviations and traces depends on the percentage of paths (M-C-1 includes 100%, M-C-0.9 includes 90% and so on). These paths are removed ordered by their frequencies, from the less to the most common. Therefore, if the operator model with best fitness is searched, the 90% of paths should be considered.

3.6.5 Time analysis

The fifth step is to analyze the missions from the perspective of time. The duration of mission is between 3 min 28 secs and 10 min 49 secs with a mean value of 5 min 36 secs 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 shortest one. Tables 3.13 and 3.14 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 secs, 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 secs 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 secs. Additionally, there are other commands that sometimes take longer times, such as *Begin* (39.50 secs), *Finish* (46.45 secs), *Capture* (28.84 secs) and *Release* (21.37 secs). These extreme cases occur when the operators have doubts about the situation of mission or make mistakes during the commanding.

Finally, Figure 3.22 shows the occupancy of multiple areas during the mission. The second and fourth quadrants are more occupied than the first and third ones (30% versus 20%), as well as the center is about twice busier than the periphery (65% versus 35%).

(A)

(B)

FIGURE 3.22: Usage of areas during the mission.

TABLE 3.13: Duration of tasks.

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

TABLE 3.14: Duration of commands.

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

3.6.6 Resource analysis

The sixth step is to analyze the missions from the perspective of resources. The event log shows the UAV 1 performs 1,340 events (63.51%), the UAV 2 develops 510 events (24.17%) 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 versus 9), whereas UAV 2 performs more times *Reconnaissance* than UAV 1 (14 versus 11). This fact explains the different number of events: e.g. quadrant changes (183 versus 165), center-periphery transitions (39 versus 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 secs 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 secs per mission, which supposes a 18.07% of its duration. If *Wait* and *Stop* events are studied according to the UAVs that perform them, it is found that UAV 1 is used 66.57% of time, whereas the UAV 2 is only used 42.07% of time.

3.7 Contributions

This chapter explores the application of process mining to model multi-robot missions. In order to achieve this objective, it not only proposes a systematic methodology (Section 3.2, but also applies it to multiple realistic missions (Section 3.6). Additionally, it studies multiple discovery algorithms to automatically generate Petri nets from mission event logs (Section 3.3) and proposes the addition of decision trees to Petri nets to predict mission evolution (Section 3.4).

The main contributions of this chapter are summarized below:

- It performs an analysis of multi-robot missions that leads to define an architecture that covers two approaches. The first one considers a set of tasks that consume a set of resources to achieve a set of objectives. The second one considers the tasks as sequences of states with actions and observations.
- It evaluates multiple modeling techniques in the context of multi-robot missions. This evaluation reveals that Petri nets and hidden Markov models are the most promising, since they are able to manage concurrency and uncertainty respectively. It also shows that Petri nets are better than hidden Markov models because they are integrated in process mining and can be combined with decision trees to address uncertainty.
- It performs one of the first works to apply process mining in the context of robotics, the most exhaustive in terms of methodology and the first one that uses data of real experiments.
- It defines novel methods to prepare the event logs for a proper discovery of models in the context of multi-robot missions. It describes with detail the actions that must be performed to achieve this objective, such as to discover events from telemetry and to create different event logs focused on different aspects of missions.
- It compares the main discovery algorithms of process mining in the context of multi-robot missions. This study reveals Inductive Miner is best in terms of robustness and performance, since it is able to generate adequate models for all the logs, and it provides the best results in terms of fitness, simplicity, generalization and precision.
- It proposes the addition of decision trees to Petri nets to estimate their evolution at decision points. The experiments show they are a powerful tool because take advantage of robot data to accurately predict mission evolution.

Chapter 4

Operator modeling

Look at you. I say this in no spirit of contempt, but look at you. The material you are made of is soft and flabby, lacking endurance and strength, depending for energy upon the inefficient oxidation of organic material. Periodically you pass into a coma and the least variation in temperature, air pressure, humidity, or radiation intensity impairs your efficiency. You are makeshift. I, on the other hand, am a finished product. I absorb electrical energy directly and utilize it with an almost one hundred percent efficiency. I am composed of strong metal, am continuously conscious, and can stand extremes of environment easily. These are facts which, with the self-evident proposition that no being can create another being superior to itself, smashes your silly hypothesis to nothing.

Cutie (I, Robot)

This chapter addresses the objective of adapting the information of interface to the preferences of operator. In order to achieve it, the behavior of operators must be modeled and the models must be used for making predictions. The proposal of this chapter is to classify the operators according to their tactics, techniques and performances in previous missions, in order to predict their strategies, actions and performances in the following ones.

The chapter collects and expands the content of the following publication:

- Roldán, J. J., Diaz-Maroto, V., Real, J., del Cerro, J., & Barrientos, A. Work hard, play hard: predicting the strategies of multi-robot operators through a videogame (review).

The rest of the chapter is organized as follows: Section 4.1 describes the resources used to solve the problem: mainly, clustering algorithms and prediction methods, Section 4.2 presents the methodology applied to analyze and model operators, Section 4.3 describes the videogame developed to obtain data of multi-robot operators, Section 4.4 addresses the application of the proposed methodology to the collected data, and Section 4.5 summarizes the main contributions of this chapter.

4.1 Resources

This section addresses the resources that are available to model operator behavior: Section 4.1.1 describes the techniques of clustering, whereas Section 4.1.2 analyzes the available resources for prediction.

4.1.1 Clustering

Cluster analysis aims at grouping series of objects in the way of the ones in the same group are more similar than the rest in the other groups [15]. This set of techniques is commonly used for unsupervised learning, since it is able to automatically classify unlabeled data. Clustering is a well known resource that has been widely applied in diverse fields such as biology [56], social sciences [205], medicine [86] and computer science [19].

The proposal of this work is to apply clustering techniques to classify people according to their behavior. In this way, the use of cluster analysis to study human behavior is really common in psychology [20]. Furthermore, there are studies that specifically apply clustering techniques to the operators of UAV missions [180].

There are several types of clustering algorithms according to their work and application, such as hierarchical, centroid-based, distribution-based and density-based clustering [62]. The objective of this work is to choose a robust technique for modeling multi-robot operators and be able to predict their actions, instead of exhaustively evaluate all the available techniques to find the best one in the mentioned context. Therefore, four really well known and widely used techniques have been considered for this study:

- K-means [93]: It is the most common centroid-based clustering algorithm, which allows to classify N observations with D dimensions into K clusters, in the way that each observation belongs to the cluster with the nearest mean. Although the problem is computationally hard (NP-hard), there are efficient heuristic algorithms to solve it. Most of the proposals of K-means algorithms initialize the centroids with a certain criterion and perform iterations with two steps called assignment and update. The first one involves the classification of observations in clusters defined by centroids. The second one addresses the computation of the means of observations to be the centroids of new clusters. The algorithms converge when there are not significant changes among iterations, but they do not ensure that these solutions are optimal.
- Self-Organizing Maps (SOM) [114]: This technique is based on an unsupervised neural network that is trained to produce a low-dimensional discretized representation of the space of observations. This algorithm has two modes called training and mapping. The first one addresses the generation of maps from inputs by using competitive learning. The second one involves the classification of new inputs by means of the mentioned maps. Two relevant parameters to be considered are the method for initializing variables and the metric for computing distances in optimization.
- Partition Around Medoids (PAM) [109]: It is a centroid-based clustering like K-means, but it chooses medoids instead of means as centers

of clusters. Similarly to means, medoids are representative points of clusters, so the differences between observations and medoids of clusters are minimal. In contrast to means, medoids must be observations, which is useful when means cannot be defined or are not representative of observations. The rest of the algorithm is similar to K-means, initializing medoids with a certain pattern and making iterations of assignment and update.

- Hierarchical [23]: This technique iteratively builds the clusters by combining similar observations (agglomerative) or separating dissimilar ones (divisive). For this purpose, two parameters are defined: metric and linkage criterion. The metric is required to compute the distances between observations and has impact on the shapes of clusters, whereas the linkage criterion is used to compute the distances between the sets of observations as a function of the distances between the pairs of observations.

The same occurs with the criteria for evaluating the quality of clusters [175]. In this work, three different criteria are considered:

- Calinski-Harabasz (CH) [28]: As shown in Equation 4.1, it is defined as the division between the overall between-cluster and within-cluster variances (SS_B and SS_A), multiplied to a factor that depends on the numbers of observations and clusters (N and K). The higher value of CH points out the higher quality of clusters.
- Davies-Bouldin (DB) [42]: This criterion is shown by Equation 4.2, where K is the number of clusters, $\bar{d}_i$ the average distance from each point in cluster i to its centroid, $\bar{d}_j$ the average distance from each point in cluster j to its centroid and $d_{i,j}$ the distance between the centroids of both clusters. In contrast to CH, the lower value of DB reveals the higher quality of clusters.
- Silhouette (S) [197]: As shown in Equation 4.3, the silhouette of a point is computed through the average distances of this point to the ones in the same cluster (a_i) and the ones in other clusters (b_i). In this case, the higher value of the sum of all the points indicates the higher quality of clusters.

$$CH = \frac{N - K}{K - 1} * \frac{SS_B}{SS_W} \quad (4.1)$$

$$DB = \frac{1}{K} * \sum_{i=1}^K \max_{j \neq i} \frac{\bar{d}_i + \bar{d}_j}{d_{i,j}} \quad (4.2)$$

$$S = \sum_{i=1}^N \frac{b_i - a_i}{\max(a_i, b_i)} \quad (4.3)$$

All these clustering techniques and evaluation criteria are implemented in *MATLAB R2017a*, which simplifies the preparation of data, integration of clustering and prediction algorithms and analysis of results.

4.1.2 Prediction

A prediction is a statement of a future event based on a previous knowledge. This definition can be formulated as shown in Equation 4.4, which estimates the value of variable V at time T_n through their values at previous times $T_1, T_2, \dots, T_{n-1}$.

$$V(T_n) = F(V(T_1), V(T_2), \dots, V(T_{n-1})) \quad (4.4)$$

The function F can be a generator of random numbers that follow a certain probability distribution. This probability distribution provides the probabilities of occurrence of the possible outcomes in an experiment. There are two types of probability distributions: discrete, which present a discrete set of possible outcomes and are represented by probability mass functions, and continuous, which can take values in continuous ranges and are represented by probability density functions. Therefore, the adequate probability distribution depends on the types and values of the possible outcomes. A list with the most common probability distributions obtained from [22] is reproduced below:

- Uniform distribution ($U(x|a, b)$): It is a continuous probability distribution that assigns equal probability to the values of x between a and b .
- Normal or Gaussian distribution ($N(x|\mu, \sigma^2)$): It is a continuous probability distribution that has two parameters: a mean μ that defines its location and a variance σ^2 that defines its scale. It is widely used because many variables in nature and society are distributed normally or can be approximated to a normal distribution.
- Student's t distribution ($St(x|\mu, \lambda, \nu)$): It is a continuous probability distribution that has three parameters: the mean μ , precision λ and degrees of freedom ν .
- Binomial distribution ($Bin(x|n, p)$): It is a discrete probability distribution that estimates the number of successes in a set of experiments with two possible results in function of the number of experiments n and the probability of success p .
- Bernoulli distribution ($Bern(x|p)$): It is a special case of binomial distribution with a single experiment ($n = 1$), where the probability of success is p and the probability of failure is $1 - p$.

4.2 Methodology

This section presents the methodology developed to model and predict the behavior of operators. As shown in Figure 4.1, the methodology takes into account four issues that are addressed in the following subsections: profiles (Section 4.2.1), classification (Section 4.2.2) and prediction (Section 4.2.3).

FIGURE 4.1: The proposed methodology for modeling operators and predicting preferences.

4.2.1 Definition of profiles

The profiles of operators can be defined as sets of variables related to their work $P = \{V_1, V_2, \dots, V_D\}$. These variables can be related to operator strategy (e.g. selection of robots, task allocation, method for commanding...), style (e.g. most used commands, robot paths, most requested information...) and performance (e.g. achieved objectives, mission time, consumed resources...).

The proposal of this work is to compute the profile for time T_n through the data collected from T_1 to T_{n-1} . For this purpose, three profiles are computed: mean, linear and quadratic. As shown in Equation 4.5, the mean profile consists of the average values of variables before T_n . The linear and quadratic profiles were developed to give more relevance to the recent strategies against the older ones. The first one is shown in Equation 4.6, whereas the second one is shown in Equation 4.7.

$$V_n^A = \frac{1}{n-1} * \sum_{i=1}^{n-1} V_i \quad (4.5)$$

$$V_n^B = \frac{1}{\sum_{i=1}^{n-1} i} * \sum_{i=1}^{n-1} i * V_i \quad (4.6)$$

$$V_n^C = \frac{1}{\sum_{i=1}^{n-1} i^2} * \sum_{i=1}^{n-1} i^2 * V_i \quad (4.7)$$

Given an individual operator that is controlling a multi-robot mission,

a profile can contain the previous actions of this operator, the previous actions of a cluster of similar operators or the previous actions of the whole population of operators. In the first case, the data is more adapted to the particular operator, but its amount can be not enough to make correct predictions. In the latter case, all the information is available to make predictions, but it can be too general to do it successfully with a particular operator. The second case is a compromise among them that can be adequate for having enough information and precision.

4.2.2 Classification of operators

As explained in Section 4.2.1, profiles can be built and predictions can be done by using three levels of information: individual operator, cluster of similar operators or population of operators. The second case requires finding groups of similar operators according to their behaviors to improve the accuracy of predictions.

The proposal of this work is to apply clustering algorithms to individual operators in order to obtain clusters of similar operators. For this purpose, the clustering techniques reported in Section 4.1.1 (K-means, SOM, PAM and Hierarchical) are applied to obtain the clusters and the three criteria reported in that section (CH, DB and S) are used to evaluate them. These clustering algorithms are used with different number of clusters in order to find the best cluster distribution.

4.2.3 Prediction of preferences

The proposal of this work is to predict the future actions of operators through the ones performed in the past. As mentioned in Section 4.1.2, this work defines probability distributions by using the previous data and generates random numbers that follow these distributions.

The profiles generated in this work consist of discrete variables with two possible values (0 or 1), so the Bernoulli distribution is used to predict future values from previous ones. However, future works would work with continuous variables or discrete variables with more possible values and, therefore, they would require the use of other probability distributions such as Gaussian or Binomial ones.

4.3 Experiments

Serious games are created for other purposes than entertainment, such as training and learning [134], evaluating systems or processes [232] and collecting behavioral data [237]. These games have been used in robotics, among other things, for studying multi-UAV mission planning problem [182].

In this work, a game for mobile devices is proposed to study the operators of multi-robot missions. Why a videogame instead of a simulator? Because of three reasons. The first one is that a game can be designed to reproduce multi-robot missions, reducing the complexity and avoiding unnecessary details (e.g. the dynamics and kinematics of robots). The second one is that a game for mobile devices can facilitate the dissemination of study and, therefore, increase the amount of available data. Finally, the

FIGURE 4.2: Captures of USARSIM: (A) Main menu, (B) Selection of strategy, (C) Control of drone and (D) Control of rover.

third one is that a competitive game can motivate the players to play more times and pay more attention to each one.

The developed game was called USARSIM because it reproduces an urban search and rescue mission. It was designed looking for a compromise between a videogame, to be played by general public, and a robotic simulator, to make conclusions about multi-robot missions. For this reason, it combines some elements of robot missions (e.g. the definition of plans, control of aerial and ground robots and use of cameras) and other ones that are common in videogames (e.g. the touch, prizes and punishments, scores and mazes). Figure 4.2 shows some screenshots of USARSIM.

The player of USARSIM has to command two robots (drone and rover) to search victims in a maze. For this purpose, multiple strategies can be defined before the game: one of the robots is manually controlled by the player, whereas the other one is autonomous and follows a certain pattern. The drone can be configured to explore with back and forth, spiral or random paths, whereas the rover can follow the left or right walls of maze or directly perform a random path. During the game, the player watches the camera of selected robot, the map explored until this moment and a miniature of the camera of other robot.

The game has four levels with different features: level 1 uses normal drone and rover, level 2 uses a light rover that is faster, level 3 uses a heavy rover that can break the walls, and level 4 a "magic" rover that not only is faster but also can break the walls. The design of these levels searches to reproduce the variability of real USAR missions.

4.4 Application

A total of 259 levels in 72 games played by 21 players were collected during one month. This data was split into training and test sets (90% and 10% respectively), in order to develop and validate the algorithms. Tactical, technical and performance data were stored in logs by a server. The analysis of this data is split into four steps: definition of profiles (Section 4.4.1), classification of operators (Section 4.4.2) and prediction of preferences (Section 4.4.3).

4.4.1 Definition of profiles

The profiles were defined through ten variables related to the tactics, techniques and performances of operators. These variables have ranges between 0 and 1 to facilitate the later study. Some variables were split into two (e.g. the strategy selected for drone was divided into "Drone Random vs Strategy" and "Drone Spiral vs Scan"), whereas some other were normalized (e.g. the number of games of each player was divided by the maximum of all the players). The ten variables of profiles with their meanings are listed below:

- Number of games: It shows the number of games of a player divided by the maximum number of games in all the players.
- Auto: Rover/Drone: It points out robot that is acting autonomously, from 0 (always the rover) to 1 (always the drone).
- Drone Random/Strategy: It indicates if the drone is moving randomly (0), following a strategy (1) or a balance between them.
- Drone Spiral/Scan: It indicates if the drone is following a spiral path (0), a back and forth path (1) or a balance between them.
- Rover Random/Strategy: It indicates if the rover is moving randomly (0), following a strategy (1) or a balance between them.
- Rover Left/Right: It indicates if the drone is following left walls (0), right walls (1) or a balance between them.
- Strategy changes: It shows the number of strategy changes of a player divided by the maximum number of strategy changes in all the players.
- Number of touches: It represents the number of touches on screen of a player divided by the maximum number of touches on screen in all the players.
- Wall crashes: It shows the number of crashes to the walls of a player divided by the maximum number of crashes to the walls in all the players.
- Score: It represents the time to complete the mission of a player divided by the worse time to complete the mission in all the players.

TABLE 4.1: Standard deviations for the variables of the profiles of players (operators) and levels (missions).

Profile	Players	Levels
Number of games	0.2888	0.2547
Auto: Rover/Drone	0.2470	0.1316
Drone: Random/Strategy	0.2592	0.0396
Drone: Spiral/Scan	0.3520	0.1002
Rover: Random/Strategy	0.3206	0.1055
Rover: Left/Right	0.3816	0.0772
Strategy Changes	0.1161	0.0869
Number of touches	0.2332	0.1729
Wall crashes	0.2463	0.4716
Score	0.2835	0.0894
Global	0.2815	0.1946

Figure 4.3 shows the average profiles computed for the four top players and the four levels of the game. These graphics reveal that there are differences between the different players and levels in multiple variables of profiles. For instance, the most relevant differences between players belong to game strategy (configuration of drone and rover), whereas the most important differences between levels are related to playing style (touches and wall crashes).

The variability of profiles was studied to determine if the differences in their variables can be explained by operators, missions or both. Table 4.1 shows the standard deviations of the variables of players and levels. As shown in data, players present more variability than levels in all the variables except wall crashes. This exception can be explained because there are two levels where the walls can be broken and other two where this is not possible. The global standard deviation, obtained from the mean of the trace of covariance matrix [168], also shows players have more variability than levels. Therefore, the study of operators through the generated profiles makes sense.

Additionally, the evolution of profiles in function of time was studied to search the best information to make predictions. As shown in Figure 4.4, the values of variables oscillate throughout games, so this information is not useful to make predictions. For this reason, three profiles are computed taking into account not only the last but also the previous data: mean, linear and quadratic. Each one of these profiles makes the evolution smoother and gives more relevance to recent values than the previous one. Figure 4.5 depicts the evolution of mean profiles for four players during the games. It can be seen that the players need a set of games to stabilize their strategies, so the errors of first predictions are bigger than the errors of the following ones.

4.4.2 Classification of operators

As previously explained, the players are classified by applying four clustering algorithms (K-means, SOM, PAM and Hierarchical) with different number of clusters (from 2 to 9). These clusters are evaluated by means of

(A)

(B)

FIGURE 4.3: Profiles of (A) operators and (B) missions.

FIGURE 4.4: Evolution value, mean, linear and quadratic strategy profiles.

FIGURE 4.5: Evolution of the mean of strategy variables of four players.

three basic metrics (CH, DB and S) and a global one that is obtained from them through Equations 4.8 and 4.9. The first one obtains the global metric G of cluster C_k through a weighted sum of the $n_v = 3$ metrics for cluster C_k . The second one defines the weight w_v for each variable V_v by multiplying the average values of the rest of variables V_i in all the clusters C_j . In this manner, the metrics are normalized and their differences are compensated.

$$G(C_k) = \sum_{v=1}^{n_v} w_v(C_k) * V_v(C_k) \quad (4.8)$$

$$w_v(C_k) = \begin{cases} -\prod_{i \neq v} \frac{1}{n_c} * \sum_{j=1}^{n_c} V_i(C_j), & \text{if } V_v = DB \\ \prod_{i \neq v} \frac{1}{n_c} * \sum_{j=1}^{n_c} V_i(C_j), & \text{otherwise} \end{cases} \quad (4.9)$$

Tables 4.2, 4.3 and 4.4 collect the results of operator classification and cluster evaluation. The best score is obtained by using the PAM algorithm for all the profiles. This algorithm is followed by SOM (two times second and once third), K-means (two times third and once second) and Hierarchical (last in all the profiles). Regarding the number of clusters, the best scores depend on the specific algorithm and dataset, but are obtained by considering from 6 to 9 clusters. The clusters obtained by using the PAM algorithm with 9 clusters are used in the rest of the work.

4.4.3 Prediction of preferences

As previously explained, the predictions were performed by using three types of profiles (individual operators, cluster of similar operators and population of operators) computed by means of three methods (mean, linear and quadratic). The predictions were performed with training set and validated with test set and each of them was repeated a hundred times to produce significant results.

Table 4.5 shows the results of the predictions with different configurations. As shown, the best result is obtained by using clusters of operators and quadratic profiles. The predictions with clusters clearly dominate the ones with individuals and population, whereas the predictions with linear profiles slightly beat the ones with mean and quadratic profiles. The best result is 65.82% of success (26.85% better than random prediction), whereas the average result is 61.40% of success (18.33% better than random prediction).

Table 4.6 shows these results dividing the players in infrequent (those that played less than or equal to 10 games) and frequent ones (those that played more than 10 games). This table shows that using clusters of similar operators is the best option to predict the preferences of infrequent operators, reaching a 66.39% of success versus the 57.29% of individuals and 60.59% of population. Nevertheless, the results are different with frequent operators and the differences are less significant: 62.56% with individuals, 61.11% with clusters and 60.36% with population.

The results demonstrate the predictions based on previous data increase the probability of success. Specifically, the clusters of operators are useful to deal with the lack of information of specific operators, whereas linear profiles are robust because they give more relevance to recent data.

However, these results also reveal some problems and challenges. A first one is there are some factors that are not related to operators and have

influence on the profiles. The most relevant of these factors is the mission. Although the analysis shows the variability is less in levels than in operators, the levels can introduce disturbances in some variables of the profiles of operators. A second one are the changes in tactics, techniques and performances due to the learning of missions. Sometimes, the first games act as a disturbance on the preferences in the rest of games and the importance of this effect highly depends on the specific player.

4.5 Contributions

This chapter explores the modeling of operators to predict their preferences and allow the adaptation of interfaces to them. In order to achieve this objective, it proposes a systematic methodology (Section 4.2, performs a set of experiments with a multi-robot game (Section 4.3) and applies the methodology to the data obtained from the experiments (Section 4.4).

The main contributions of this chapter are summarized below:

- It proposes an original methodology to classify the operators of multi-robot missions and predict their preferences. This methodology is based on clustering algorithms and evaluation metrics, as well as generation of random numbers following probability distributions.
- It develops a serious game based on the control of multi-robot missions to obtain data of operators. This game allows to collect massive data of people without the common limitations of time and resources of multi-robot missions.
- It applies the methodology to the data obtained from the game, demonstrating the predictions based on operator clusters and linear profiles provide the best rate of success, improving in more than 25% the results of random predictions.

TABLE 4.2: Comparison of clustering techniques with mean profiles.

Technique	N_C	CH	DB	S	G
K-means	2	5.03	1.62	0.32	0.53
	3	4.62	1.40	0.26	0.49
	4	4.26	1.27	0.34	1.10
	5	5.18	1.15	0.22	0.95
	6	6.95	0.84	0.43	3.58
	7	7.02	0.78	0.43	3.71
	8	6.89	0.76	0.44	3.77
	9	5.36	0.69	0.45	3.36
SOM	2	5.55	1.56	0.34	-0.73
	3	4.77	1.10	0.38	0.66
	4	6.23	0.84	0.48	2.63
	5	6.54	0.80	0.48	2.93
	6	7.44	0.67	0.57	4.21
	7	6.16	0.71	0.44	2.79
	8	8.53	0.58	0.56	4.91
	9	8.41	0.63	0.55	4.62
PAM	2	4.59	1.68	0.27	-2.08
	3	5.90	1.09	0.41	1.39
	4	6.58	0.96	0.44	2.30
	5	6.62	0.79	0.50	3.23
	6	7.01	0.81	0.49	3.32
	7	9.01	0.66	0.60	5.34
	8	8.53	0.58	0.56	5.20
	9	7.85	0.52	0.61	5.41
Hierarchical	2	4.40	0.43	0.65	2.11
	3	4.05	0.48	0.53	1.43
	4	3.68	0.53	0.43	0.84
	5	6.31	0.70	0.54	1.44
	6	7.44	0.67	0.57	1.98
	7	6.29	0.59	0.58	1.94
	8	7.79	0.56	0.62	2.63
	9	7.85	0.52	0.61	2.78

TABLE 4.3: Comparison of clustering techniques with linear profiles.

Technique	N_C	CH	DB	S	G
K-means	2	3.67	1.77	0.18	-0.45
	3	5.57	1.34	0.33	1.66
	4	4.39	1.04	0.38	1.93
	5	5.28	0.82	0.44	2.93
	6	6.56	1.00	0.48	3.31
	7	4.84	0.85	0.30	1.97
	8	4.56	0.97	0.23	1.32
	9	2.87	1.02	0.25	0.76
SOM	2	4.72	1.63	0.26	-0.67
	3	4.90	1.19	0.34	1.10
	4	4.14	1.26	0.30	0.29
	5	6.25	1.00	0.44	2.78
	6	6.86	0.78	0.52	4.13
	7	7.67	0.79	0.56	4.69
	8	6.64	0.77	0.46	3.70
	9	7.12	0.62	0.61	5.17
PAM	2	3.01	2.16	0.17	-3.26
	3	4.47	1.30	0.31	0.65
	4	5.64	1.04	0.39	2.42
	5	6.16	1.08	0.43	2.86
	6	6.56	1.00	0.48	3.56
	7	7.67	0.79	0.56	5.19
	8	7.36	0.68	0.59	5.53
	9	7.47	0.55	0.61	6.10
Hierarchical	2	4.05	0.45	0.62	1.88
	3	3.04	0.59	0.36	0.54
	4	3.32	0.55	0.39	0.81
	5	3.14	0.56	0.37	0.68
	6	4.86	0.69	0.49	1.21
	7	6.18	0.67	0.54	1.78
	8	6.62	0.66	0.58	2.04
	9	5.86	0.60	0.60	2.02

TABLE 4.4: Comparison of clustering techniques with quadratic profiles.

Technique	N_C	CH	DB	S	G
K-means	2	4.79	1.65	0.28	0.44
	3	4.55	1.32	0.31	1.03
	4	4.69	1.16	0.33	1.52
	5	2.08	1.13	0.11	-0.60
	6	4.64	1.06	0.29	1.45
	7	6.33	0.82	0.51	3.60
	8	5.72	0.72	0.57	3.83
	9	5.14	0.68	0.49	3.28
SOM	2	3.53	0.48	0.56	2.82
	3	5.29	1.33	0.29	0.28
	4	5.22	1.07	0.39	1.31
	5	4.93	0.99	0.39	1.39
	6	3.51	0.93	0.36	0.83
	7	6.33	0.82	0.51	2.87
	8	6.78	0.65	0.58	3.76
	9	6.25	0.72	0.48	2.97
PAM	2	4.79	1.65	0.28	-0.24
	3	4.93	1.35	0.26	0.42
	4	5.00	1.12	0.34	1.38
	5	5.79	0.96	0.43	2.54
	6	5.64	0.89	0.44	2.71
	7	5.56	0.74	0.49	3.26
	8	5.81	0.65	0.54	3.82
	9	5.39	0.59	0.61	4.16
Hierarchical	2	3.53	0.48	0.56	1.59
	3	4.13	0.80	0.44	1.08
	4	3.63	0.74	0.37	0.85
	5	2.99	0.76	0.18	0.23
	6	2.44	0.66	0.24	0.34
	7	2.29	0.68	0.27	0.35
	8	2.46	0.75	0.36	0.51
	9	5.07	0.61	0.61	1.93

TABLE 4.5: Performance of predictions with different profiles (data in rows and methods in columns). Note: Performance with totally random prediction was 51.89%.

Performance	Mean	Linear	Quadratic	Average
Individuals	61.06%	60.23%	59.37%	60.22%
Clusters	63.82%	65.82%	62.29%	63.97%
Population	60.03%	60.70%	59.31%	60.01%
Average	61.64%	62.25%	60.32%	61.40%

TABLE 4.6: Performance of predictions for operators according to their number of games: Individual (I), Clusters (C), Population (P), Mean (M), Linear (L) and Quadratic (Q).

Players	$Games \leq 10$	$Games > 10$
I-M	57.56%	66.16%
I-L	57.60%	61.12%
I-Q	56.71%	60.40%
C-M	66.16%	61.01%
C-L	66.49%	62.88%
C-Q	66.55%	59.45%
P-M	60.90%	60.61%
P-L	60.63%	60.45%
P-Q	60.03%	60.03%

Chapter 5

Adaptive interface

I've seen things you people wouldn't believe. Attack ships on fire off the shoulder of Orion. I watched c-beams glitter in the dark near the Tannhäuser Gate. All those moments will be lost in time, like tears in rain. Time to die.

Roy Batty (Blade Runner)

As the Chapters 3 and 4 respectively addressed the modeling of missions and operators, this chapter uses these models to search the relevant information in the mission data. In this way, the interfaces can perform some functions of operators, such as data processing and information discovery, reducing their workload during the missions.

The chapter collects and expands the content of this publication:

- Roldán, J. J., Peña-Tapia, E., Martín-Barrio, A., Olivares-Méndez, M. A., Del Cerro, J., & Barrientos, A. (2017). Multi-robot interfaces and operator situational awareness: Study of the impact of immersion and prediction. *Sensors*, 17(8), 1720 [192].

The rest of the chapter is organized as follows: Section 5.1 analyzes the resources available for data processing and information discovery in multi-robot missions, Section 5.2 describes a work in which the interface is able to select the most relevant information of a multi-UAV mission, Section 5.3 addresses another work in which the interface performs the evaluation functions of the operator, and Section 5.4 summarizes the main contributions of this chapter.

5.1 Resources

As previously mentioned, the aim of this chapter is to adapt the information shown by interface to mission context and operator preferences. In order to achieve this goal, there are several potential techniques that vary from expert systems to machine learning.

Expert systems collect a series of rules generated by human experts and codified as a set of conditions with associated operations [99]. In the context of multi-robot missions, these systems are able to collect the operator criteria, which can be represented as a set of mission situations with associated decisions and actions. However, these systems presents some limitations: they have to be trained before the missions, the rules must be generalizable

to all operators and cannot represent multiple different responses to one potential situation, and it is hard to cover all the potential cases. For these reasons, this technique can be useful to model operation procedures, but may not be suitable to represent operator behaviors.

Machine learning allows systems to learn without being explicitly programmed [22]. In the context of multi-robot missions, these techniques may allow interfaces to learn the criteria of operators during missions and, therefore, to discover the relevant information of mission and provide it to operators.

There are two main categories of machine learning: supervised and unsupervised learning. In supervised learning, the system receives some example signals with desired responses and automatically maps the inputs to the outputs. In unsupervised learning, the system only receives the inputs and has to infer the outputs. Additionally, reinforcement learning allows to train the system by executing actions and receiving feedback from the environment in form of reward and punishment.

In the context studied by this thesis, supervised methods are more suitable than unsupervised ones, since they have to learn from operators. The main difficulty is to perform an adequate training involving human operators, since there is a dilemma between its cost and quality: if the input dataset is small, the model may suffer overfitting or underfitting, whereas if the input dataset is large, the training may consume excessive time and resources. In this application, reinforcement learning does not require the direct evaluation of operators, but may take advantage of other interactions of them.

The main models of machine learning are decision trees, artificial neural networks (including conventional and deep ones), inductive logic programming, support vector machines, clustering methods, bayesian networks and evolutionary algorithms. In this work, neural networks are integrated in interfaces to learn from operators the capability to evaluate missions (see Section 5.3).

5.2 Discovering mission information

This first work aims at developing an adaptive interface for multi-UAV missions. As illustrated in Figure 5.1, this interface performs some operator functions as the analysis of mission and discovery of information. For this purpose, it uses the models of mission and operator respectively described in Chapters 3 and 4.

The interface should be integrated in the SAVIER multi-UAV Demonstrator provided by Airbus Defence & Space. As shown in Figure 5.2, this demonstrator consists of the Unmanned Aerial Vehicles, which are powered by instances of ArduPlane¹, and the Ground Control Station, which is formed by the QGroundControl-3.0². This GCS includes the components developed in the context of SAVIER project: specifically, mission planning, task planning, mission monitoring (the contribution of this thesis) and payload monitoring.

¹<http://ardupilot.org/plane/>

²<http://qgroundcontrol.com/>

FIGURE 5.1: Discovery of information from data by using mission and operator models.

FIGURE 5.2: Diagram of SAVIER multi-UAV Demonstrator.

The mission monitoring window is shown in Figure 5.3, together with the main window in the mission planning tag. This window consists of three panels: a left one with the list of UAVs involved in mission, a centered one with the information of most relevant UAV, and a right one with the lists of alerts and events.

The list of UAVs shows the aerial robots that take part in the mission ordered by their priority in the current situation. This priority is computed through Equation 5.1, where v_i are the variables of mission and w_i are their assigned weights. The considered variables in the first tests are the level of battery, activation of sensors and actuators, current task, distance to route, distance to mission area and distance to other UAVs, but other relevant variables can be added in the future. The weights are used to normalize these variables and control their influence in the interface according to the experience of operators in previous missions.

$$Priority = \sum_{i=1}^n w_i * v_i \quad (5.1)$$

This panel shows the current and next task of each UAV, as well as its priority in the context of mission. The current task is obtained through the Petri nets generated from previous missions and the real-time data provided by the current one (as described in Chapter 3). The next task is chosen among the possible ones in Petri net through the decision trees generated from previous missions and the real-time data provided by current one (again, as addressed in Chapter 3).

Two types of variables are used to model missions and apply models: primary and secondary variables. On the one hand, the primary variables are directly provided by the UAVs and include the base mode, custom mode, latitude, longitude, altitude, relative altitude, roll, pitch, heading, ground speed, air speed, climb rate, battery current, battery consumed, battery percent, battery temperature, battery voltage, wind direction, wind speed, wind X, wind Y, wind Z, last WP latitude, last WP longitude, next WP latitude and next WP longitude. On the other hand, the secondary variables are obtained by operating with the primary ones and include relative latitude (from min to max planned latitudes), relative longitude (from min to max planned longitudes), relative altitude (from min to max planned altitudes), speed in X, Y, Z, Roll, Pitch and Heading, acceleration in X, Y, Z, Roll, Pitch and Heading, indicator of UAV in mission area, UAV in task area, UAV in loiter and UAV in route, distance to airport, GCS, contingency and other vehicles, estimated return time and battery, current and next tasks, mission, flight and task times, remaining mission, flight and task times, covered flight and task distances, remaining flight and task distances and priority.

The panel of most relevant UAV is divided into four panels. The first one collects the information of this UAV: current and next state, last and next WP, distance, estimated time of arrival (ETA), battery, absolute and relative altitude, air and ground speeds, climb rate and indicator of UAV in route. The second one shows a tree that points out the reason why this UAV is the most relevant between three categories (state, role and environment) and six reasons (battery, sensors and actuators, task, route, area and other UAVs). The third one contains a map centered in the most relevant UAV

(A)

(B)

FIGURE 5.3: Captures of the interface: (a) Mission planning tag, (b) Mission monitoring window.

FIGURE 5.4: Contribution of prediction to interfaces.

that shows the locations of its WPs and the other UAVs. Finally, the fourth one shows the attitude and compass indicators of the most relevant UAV.

Finally, the lists of alerts and events respectively collect the alarms and warnings of UAVs during missions (low battery, out of route, waiting command...) and the recent and relevant mission events (state changes, accomplished tasks...). These alerts and events are sorted by their times and relevances.

This interface was tested in several demonstrations of the SAVIER project in the facilities of Airbus Defence & Space. During the mentioned demonstrations, several people involved in the development and application of GCSs for UAVs provided their comments and suggestions, which were considered in the development of interface.

5.3 Transferring operator functions

This second work aims at transferring functions from operator to interface in order to reduce workload. As shown in Figure 5.4, the operators that use conventional interfaces must receive data, discover information, make decisions, generate commands and send them to robots. A potential solution to relieve this workload is to transfer functions from operator to interface. Specifically, the idea behind this work is assigning to interface the discovery of information instead of to operator. In this study, the missions described in Chapter 3 were used to develop and validate this solution.

An interesting related work can be found in [210], which tries to provide robots with human capabilities, such as perceiving, understanding and avoiding risks during missions. For this purpose, the author studies human perception, cognition and reaction against risks and develops a framework to apply these concepts to aerial robots.

The amount of data generated by a multi-robot mission depends on the number and complexity of robots and tasks. As already mentioned, this raw data includes robot states (position, orientation, speed, battery...)

and payloads (measurements of sensors, images of cameras, state of actuators...). When the amount of data is too large, operators may find themselves unable to process these data in order to extract the most relevant information. Therefore, they might be unable to get information about which task is more critical, which robot requires more attention, which situation presents more risks...

The proposed solution is to automatically determine what information is important from the raw data of mission. For this purpose, all instances of data generated by the mission are considered input variables, and the problem is simplified by selecting three output variables: the task, relevance and risk of each robot at any moment of the mission. Current and next tasks are determined by using Petri nets and decision trees respectively, following the methods described in Chapter 3. Relevance and risk are determined through a procedure with four steps: evaluation of human operator, preparation of datasets, neural network training and subsequent validation. The definitions of these variables are explained below:

- **Relevance.** This variable measures the importance of the robot in a certain situation of the mission. In this work, it is considered as a percentage, which varies from 0% (i.e. the robot is not involved in the mission) to 100% (i.e. it is the unique one that is performing the mission). The sum of the relevances of all the robots that take part in the mission must be 100%.
- **Risk.** This variable measures the potential danger that the robot can suffer in a certain situation of the mission. In this work, it is considered as a percentage, which varies from 0% (i.e. the robot is completely safe) to 100% (i.e. it has suffered an accident). In this case, the risk of one robot is independent of the risks of the rest of the fleet.

The first step of this work consists of an operator evaluating the relevance and risk of the robots during missions: watching mission videos and noting their values over time. This evaluation was performed with sixteen missions: eight for training and eight for validation. Some of the operator criteria for the evaluation of relevance and risk are shown in Figure 5.5. On the one hand, the relevance of a robot is related to the task that it is performing, together with every possible anomaly or problem. On the other hand, the risk of a robot is influenced by factors such as its proximity to obstacles or other robots. However, it must be remarked that this evaluation is not deterministic, since the operator can provide different outputs for similar inputs.

The second step of the procedure involves the preparation of dataset. The result is a series of spreadsheet with an average of 590 readings whose columns are the mission variables and whose rows are their values at different timestamps. In this case, mission data includes 101 variables that can be classified as primary variables (60 variables collected by telemetry, such as the position, orientation, battery...), secondary variables (37 variables that are obtained by operating with them, such as the task, distance between robots...), and target variables (relevance and risk of all the robots). These latter variables were subjected to a correlation test: showing absolute inverse correlation between relevances and low correlations between the rest

FIGURE 5.5: Approximation of evaluation criteria of operator for robot relevance and risk.

of variables (0.594 between risks, ± 0.178 and ± 0.395 between relevances and risks).

In third place, the neural networks are trained using the operator's evaluation of risk and relevance. For this purpose, the eight training datasets above mentioned were used, and a back propagation algorithm was implemented in *RapidMiner Studio 7.5*. A total of sixteen NNs have been generated: four NNs (1, 2, 3 and 4 hidden layers) for the four variables (robot 1 and robot 2 relevance and risk).

Finally, the fourth step of the procedure involves the validation of NNs with the other eight datasets. Figure 5.6 shows the mean and standard deviation of the NN variable prediction errors. It can be appreciated that relevances have greater errors than risks, probably due to the nature of these variables: the first ones oscillate from 0 to 100% with an average of 50%, whereas the second ones are usually under 20% and only presents some peaks. NNs with four hidden layers have been chosen because they obtain the best results with three of four variables: both risks and the relevance of robot 2. Figure 5.7 shows the real and predicted values for the four variables, as well as the absolute error for these variables. As shown, the NNs predict successfully the trends of the variables (i.e., rises and falls), but they are often unable to reach the peaks, explaining the errors above mentioned.

The comparison among the errors of NNs with training and validation datasets reveals a phenomenon of overfitting. Specifically, the errors with training dataset are five times lower than with validation one. However, the interfaces do not use directly the values of relevances and risks, but thresholds for these variables. For instance, when the relevance of an UAV is higher than the other UAV, the interfaces select this UAV, or when the risks of an UAV are over 50%, the interfaces show an alert. In this case, the total error of NNs including both false positives and false negatives is 7.64% (11.55% in UAV 1 relevance, 5.1% in UAV 1 risk, 12.28% in UAV 2 relevance and 1.6% in UAV 2 risk). This result is considered as adequate for the correct work of interfaces. The neural networks are integrated in the interfaces described in Chapter 6.

FIGURE 5.6: Prediction errors of neural networks.

(A)

(B)

FIGURE 5.7: Evaluation of operator vs. prediction of neural networks. (a) Direct comparison; (b) Absolute error and mean value.

5.4 Contributions

This chapter explores how to manage the information of multi-robot missions to reduce the workload of operators. In order to achieve this objective, two works are developed: one related to the discovery of relevant information from mission data (Section 5.2) and another one related to the transference of functions from operator to interface (Section 5.3).

The main contributions of this chapter are summarized below:

- It proposes a method to determine the most important robot at any moment of mission based on its state, role and environment.
- It applies neural networks to reproduce the evaluations of operators, predicting with success the relevances and risks of robots during the mission.

Chapter 6

Immersive interfaces

*What is real? How do you define "real"?
If you're talking about what you can feel,
what you can smell, what you can taste
and see, then "real" is simply electrical
signals interpreted by your brain.*

Morpheus (The Matrix)

This chapter addresses the objective of developing immersive interfaces to introduce operators in scenarios and improve their situational awareness. These interfaces must cover the two activities that operators perform during multi-robot missions: monitoring and commanding.

The chapter collects and expands the content of the following publications:

- Roldán, J. J., Peña-Tapia, E., Martín-Barrio, A., Olivares-Méndez, M. A., Del Cerro, J., & Barrientos, A. (2017). Multi-robot interfaces and operator situational awareness: Study of the impact of immersion and prediction. *Sensors*, 17(8), 1720 [192].
- Peña Tapia, E., Roldán Gómez, J. J., Garzón Oviedo, M. A., Martín Barrio, A., & Barrientos Cruz, A. (2017). Interfaz de Control para un Robot Manipulador mediante Realidad Virtual [169].
- Roldán, J. J., Peña-Tapia, E., Garzón-Ramos, D., de León, J., Garzón, M., del Cerro, J., & Barrientos, A. (2018). Multi-Robot Systems, Virtual Reality and ROS: Developing a new generation of operator interfaces. *Robot Operating System (ROS) - The Complete Reference Volume 3* [193].

The rest of the chapter is organized as follows: Section 6.1 presents the resources available to develop immersive interfaces, Section 6.2 describes the multi-robot missions used to validate the interfaces, Section 6.3 reports the development of an immersive interface for monitoring and commanding multi-robot missions, Section 6.4 addresses the experiments performed to validate the interface, Section 6.5 analyzes the results of these experiments, and Section 6.6 summarizes the main contributions of this chapter.

6.1 Resources

As reported in chapter 2, there are three main technologies available for immersive interfaces: virtual reality (VR), augmented reality (AR) and mixed

reality (MR). Therefore, the first question to be asked before developing an immersive interface is "VR, AR or MR: when should we apply each technology?".

- Virtual reality: This technology is the best option when the scenario is remote (e.g. spacial operations), the mission is dangerous (e.g. intervention in disaster areas), the scenario is large (e.g. open fields) or the operator has to control multiple agents (e.g. multi-robot systems). In those contexts, viewing the real scenarios and interacting with their elements does not add value to operations; whereas reproducing remote, dangerous or large scenarios and freely move there makes it easier to perform these tasks.
- Augmented reality: This technology is the best option when the operator is close to mission (e.g. indoor scenarios), the mission is complex (e.g. industrial applications), the scenario is small (e.g. rooms and corridors) or the operator has to interact with robot (e.g. social robotics). In these contexts, in contrast to the previous ones, viewing the real scenario with virtual elements is more useful than watching a virtual reproduction of it.
- Mixed reality: This technology is useful in the same scenarios than AR and, specially, when human-robot collaboration is required. This is because it provides the operators and robots with the possibility to interact with the real and virtual objects of the scenario.

In this thesis, virtual reality has been chosen among these technologies for the development of immersive interfaces, taking into account both the proposed guidelines and the considered scenarios. Once decided to work with VR, the second question is "Which resources of hardware and software can be used to develop the interface?"

The development of VR systems has been fast in the last years, and new systems with improved features appear frequently. The main VR headsets today available in the market are collected by Table 6.1. Currently, headsets can be classified into two categories: mobile headsets, which depend on the use of a phone for visualization, and tethered, which require a connection to a computer to work. However, new untethered headsets that do not require additional devices to work will appear in the following months. Meanwhile, the mobile ones can be more flexible and affordable, but only the tethered ones meet the requirements for being applied in the context of robotics, such as high frequency, resolution and field of view. Nowadays, the *HTC Vive*¹ and *Oculus Rift*² headsets are the most suitable options for the development of applications, since they are compatible with *Unity* and *Unreal Engine 4* game engines.

The interfaces explained in this chapter have been developed for the *HTC Vive* virtual reality headset. This system was originally conceived for gaming, but it is currently used for multiple purposes in industry, services and research. The system shown in Figure 6.1 consists of a head-mounted display (HMD), two controllers and a tracking system. The HMD integrates two screens with resolution of 1080x1200 pixels and frequency of 90 Hz, and

¹HTC Vive: <https://www.vive.com/>

²Oculus Rift: <https://www.oculus.com/>

TABLE 6.1: Study of available VR headsets.

Name	Type	Hardware Required
Sony PlayStation VR	Tethered	PlayStation 4
HTC Vive	Tethered	PC
Oculus Rift	Tethered	PC
Google Daydream View	Mobile	Daydream compatible phone
Samsung Gear VR	Mobile	Latest Samsung Galaxy models
Homido VR	Mobile	Android and iOS phones
FreeFly VR	Mobile	Android and iOS phones
Google Cardboard	Mobile	Android and iOS phones

FIGURE 6.1: HTC Vive HMD, controllers and base stations.

an audio jack that allows the integration of sounds to improve the immersion. The controllers allow to interact with the elements of virtual scenario, providing multiple types of buttons and various degrees of vibration. Finally, the tracking system consists of two base stations that are placed in two opposite vertices of the work area and provide the localization of HMD and controllers with sub-millimeter accuracy and low latency [160].

The most common method of programming applications for *HTC Vive* is based on the use of *Unity* and *SteamVR*. *Unity* is a game engine that provides a complete set of resources, such as realistic scenarios, models of objects, kinematic and dynamic models, characters with artificial intelligence, etc. *SteamVR* is a plugin that provides developers with some resources related to virtual reality and allows them to compile and deploy their applications. A current limitation of this plugin is that it only works under Windows-based operating systems, so an additional computer or a virtual machine is necessary for its integration in ROS-based environments.

The design of immersive interfaces highly depends on the fleet, operators, mission and scenario. An interface for a single robot should be centered in this robot and its payload; for instance, adopting a first person view and giving priority to the images of its cameras. On the other hand, an interface for multiple robots should combine the visualization of the whole scenario with a closer view of each robot involved in the mission. In a similar way, if there is a single operator to control the mission, the interface should provide all the information required to perform it, whereas if there are multiple operators with different roles, the interface should integrate these roles and provide the adequate information according to them: this means that robot operators need to know the close environment of the robots, payload operators require to know the state and readings of the sensors, and mission commander needs to acquire a general view of mission. Furthermore,

TABLE 6.2: Main considerations for developing immersive interfaces.

Issue	Objectives	Addressed
Scenarios	Manual design and load	Yes
	Offline reconstruction	No
	Obstacle detection and mapping	No
	Online reconstruction	No
Monitoring	Representation and movement	Yes
	Information of state	Yes
	Integration of sensors	Yes
	Integration of cameras	Yes
Commanding	Teleoperation	No
	WP commands	Yes
	Task commands	No
Robots	Mobile	Yes
	Manipulators	Yes
	Multi-robot systems	Yes

the type of mission has impact on the information displayed by the interface. For instance, an interface for surveillance must show clearly the found targets, whereas another one for manipulation must give accurate information about the effector and target. Finally, the scenario also impacts on the interface: small scenarios can be reproduced in 1:1 scale and traversed by the operator, whereas large ones need to be reduced in the virtual reality and may require the use of resources like teleporting.

Apart from these issues, a set of considerations for the development of immersive interfaces are summarized in Table 6.2. The virtual scenarios can be built manually by using *Unity* and its assets, or reconstructed by means of 3D cameras or LIDAR sensors. In the latter case, this reconstruction can be performed before the mission, online or a combination of both (e.g. acquiring the scenario before the mission and detecting and mapping obstacles as it is performed). However, real-time reconstruction and immersion is more a goal than a reality, since the computational cost of the process of reconstructing the environment, improving the 3D map, creating a mesh and importing it in virtual reality is very high [158]. Furthermore, the interfaces must show diverse types of information: from the representation of the robots and their movements in the scenario, to the integration of their state, the readings of their sensors and the images of their cameras. For this purpose, multiple elements can be added to robots to represent the information that is relevant at any moment. When the fleets are large and the amount of information is huge, these elements can be shown or hidden at operator request. Additionally, the interfaces can integrate different methods for commanding robots: from teleoperation, which is useful with single robots, to task commands, which are practical in multi-robot systems, through waypoint (WP) commands, which are intuitive and require an intermediate effort. Finally, the integration of different types of robots and fleets will require developing different resources for their supervision and control.

As mentioned above, *Unity* provides a complete set of assets developed by its community, which can be downloaded and applied in the projects.

These assets facilitate the work for building the scenarios, managing the information and commands and developing human-robot interaction. Among the *Unity* assets, the *SteamVR* plugin is specially useful, since it contains the basic interactions required by virtual reality, such as visualizing, picking, placing and teleporting. The main resources used in the development of the interfaces reported in this chapter are listed below:

- Teleport: The operator can change the location in the scenario by pointing with the controller and pressing the trigger. This is actually useful when the scenario is large and the task require accuracy.
- Selection of robots/goals: The operator can use the controller to shoot to a robot, selecting it to see information and enter commands, or a position in the scenario, to send it as a goal for the robot.
- Robot panel: A panel over a robot can show the relevant information of this robot: e.g. the current task, level of battery, images of camera...
- Robot mark: Some elements can be added to the interface in order to guide the attention of the operator to some robots.
- Sound: The sounds of robots and other relevant elements (e.g. water and fire) help the operators to get proper orientations and find them in virtual scenario.
- Vibration: Controller vibration can be used to draw the attention of operators to the alerts, as well as to guide them in the robot commanding task.

6.2 Missions

The same set of multi-robot missions of Chapter 3 was used to evaluate the developed interfaces. As shown in Figure 6.2, these missions were carried out at the laboratory of the Interdisciplinary Centre for Security, Reliability and Trust (SnT) of the University of Luxembourg in April 2016 [188]. However, as also shown, the interface tests took place at the laboratory of Robotics and Cybernetics Group (RobCib) of the Technical University of Madrid (UPM) in April 2017. The missions were reproduced successfully with the aid of stored telemetry and commands, together with video recordings of every session.

The aim of missions was to control two aerial robots that perform tasks of detecting-extinguishing fires, and finding-following intruders. The scenario was set up in a 5.35 m × 6.70 m × 5.00 m room with a robot base, a water well and a fire with variable locations. The UAVs had to perform the following tasks to accomplish the mission:

- 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.

FIGURE 6.2: Multi-robot missions performed in Luxembourg in 2016 and virtually reproduced in Madrid in 2017.

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

Two *Parrot AR.Drone 2.0* UAVs [119] and a *KUKA Youbot* [21] UGV were employed to perform the missions. The aerial robots were operated through an interface; whereas the ground robot was considered as a target for detection and tracking and, therefore, was out of control for the UAV operator.

A motion capture system *Optitrack* was used to get the accurate position and orientation of aerial and ground robots [44]. This system was able to capture and track the robots around the scenario with a series of infrared cameras together with a set of reflective markers attached to the robots.

All hardware and software components of the missions, including the robots, motion capture system and operator interface, were integrated by means of the Robot Operating System (ROS) open-source framework [172].

The aerial robots rendered a full telemetry record that consisted 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. Additionally, the motion capture system provided accurate readings of the positions and orientations of the aerial and ground robots. Finally, the user interface allowed the operator to send commands with the robot tasks referenced in real time. All this information was stored for future use in ROS bag files, as well as XLSX spreadsheets.

6.3 Interfaces

As previously mentioned, the objective of this chapter is to develop an immersive interface for monitoring and controlling missions with multiple robots. This interface must meet a set of requirements:

- Integration of multiple heterogeneous robots: e.g. UGVs, UAVs...
- Integration of complex robots: e.g. mobile manipulator.
- Realistic reproduction of real scenarios.
- Immersion of operators to enhance situational awareness.
- Modular architecture to adapt to different scenarios, robots...
- Integration with Robot Operating System (ROS).

6.3.1 Robots

In order to meet the requirements listed above, a diverse set of robots was implemented in the interface. Their integration was performed through 3D models, displays of information and methods of commanding. The ground, aerial and manipulator robots integrated in the interface are described in the following sections.

Aerial robots

The aerial robots were the *Parrot AR.Drone 2.0* used in the missions of Section 6.2. They were integrated in game objects that consist of the 3D models, animation scripts and sound sources. These robots were controlled by means of the *ardrone_autonomy* package in some of the tests and simulated in *Gazebo* by using the *rotors_simulator* package in the rest of them.

The information of UAVs is mainly collected by three messages published in three topics. The *ardrone_autonomy/Navdata* message provides the drone state (taking-off, flying, hovering, landing...), battery, orientation, velocity, acceleration, motor voltage, pressure, temperature and altitude. The *geometry_msgs/Pose* message provides the location of drone generated by *Gazebo* in the simulations and the *Optitrack* motion capture system in the experiments. Finally, the *sensor_msgs/Image* message provides the image of the chosen camera of drone: the front or the bottom one.

Other messages and topics are applied to command the UAVs. The *std_msgs/Empty* messages are used to send commands of take-off, land and reset, whereas the *geometry_msgs/Twist* messages are used to send speed commands. However, an interface was developed to control the drone by means of waypoint commands in *geometry_msgs/Pose* messages.

Ground robots

Two types of ground robots were implemented in the interface: the *KUKA Youbot* and *Robotnik Summit XL/HL*. They were integrated as game objects with both 3D models and scripts that control their movements. These robots are respectively controlled by the *youbot_oodl* and *summit_xl_common* and simulated by the *youbot_oodl* and *summit_xl_sim* packages.

Although these robots generate a considerable amount of data, only a part of this data is useful for the interface. In this way, the *geometry_msgs/Pose* messages are used to transmit the position and orientation of robots. This information is collected by the *Optitrack* motion capture system in indoor experiments and obtained by sensor fusion (odometry, IMU and GPS) or localization algorithms (SLAM and AMCL) in outdoor experiments. Additionally for the *Robotnik Summit XL/HL*, the level of battery is sent through *std_msgs/Float32* message and the images of cameras by means of *sensor_msgs/Image* messages. The rest of information is not transmitted to the interface to not overload communications, but can be easily transmitted if it is required.

Additionally, two types of messages must be used to allow two types of commanding. On the one hand, the *geometry_msgs/Twist* messages can be used for a lower level commanding by using the controllers of HMD headset. On the other hand, the *geometry_msgs/Pose* messages can be used for a higher level commanding through waypoints.

Manipulator robots

The robot used for this application was a *Kinova Jaco*³ with six degrees of freedom (shoulder, arm, forearm, wrist 1, wrist 2 and hand) and one effector with three fingers. This robot is able to manage a payload of 1.6 kilograms and reach distances of 900 millimeters.

In this case, the interface requires the real-time values of the angles of the joints, in order to represent the state of manipulator robot in any time of the mission. This information is collected by the *sensor_msgs/JointState* message that is published by the controller node of manipulator robot.

Two types of messages are necessary to command this manipulator robot: *geometry_msgs/Pose* to send goals with position and orientation to the effector, and *std_msgs/Float32* to open and close the gripper for picking and placing objects.

6.3.2 Scenarios

As the main idea of immersive interfaces is to introduce operators in scenarios, these scenarios must be reproduced with as many details as possible. As previously discussed, the scenarios can be modeled before the missions, but some elements must be added during them.

In this work, three scenarios were created to validate three interfaces. The first one reproduces the scenario of the experiments of Chapter 3 and was used to develop and test a multi-robot monitoring interface. The second one reproduces the environment of laboratory and was used to develop

³<http://www.kinovarobotics.com/>

FIGURE 6.3: Non-predictive virtual reality interface.

a commanding interface for a manipulator robot. Finally, the third one reproduces a critical infrastructure and was used to develop a commanding and monitoring interface for aerial and ground robots.

These monitoring and commanding interfaces are respectively described in Section 6.3.3 and Section 6.3.4.

6.3.3 Monitoring

Two virtual reality interfaces were developed for monitoring the missions described in Section 6.2: one non-predictive and another predictive.

The non-predictive virtual reality interface is based on the scenario of experiments and represents at full scale all its elements. Figure 6.3 shows a screen capture of the interface with its static and dynamics elements. The static elements include not only the floor, glass walls and water well, but also an observation platform that offers the user an additional viewpoint. The dynamic elements are the two UAVs, the UGVs and the fires, together with the HTC Vive controllers. The UAVs are able to fly according to the mission they are performing, with realistic rotor movement and sound that indicates their proximity.

The robots are equipped with translucent screens in which the battery state and name of the task in performance can be checked easily. These screens are animated to always face the headset camera, so that the information in them is accessible all the time. The UGV and fire remain hidden during the missions until they are detected.

Although the operators can walk around their position in the virtual scene, the teleporting is the main way to navigate in the scenario. The users have to press either of the Vive controller triggers and drive the arc to the desired destination. In order to facilitate access to the observation platform,

FIGURE 6.4: Teleport to floor (left) and observation platform (right).

there are two teleporting points in the wall where it is attached, as can be seen in Figure 6.4.

The predictive virtual reality interface keeps the same elements from the non-predictive version, and adds a predictive component with two new elements that symbolize risk and relevance. The information of risk and relevance was obtained in Chapter 5 and the hints of interface are shown in Figure 6.5. There is a spotlight that follows the most relevant drone during the mission, giving hints to the operator about where to look. Additionally, when a risk factor above the threshold is detected, a smoke cloud surrounds the endangered UAV.

6.3.4 Commanding

Two virtual reality interfaces were created to command two types of robots: one for a manipulator robot and another for a fleet of aerial and ground robots.

Manipulator robot

The first of these interfaces was developed to command a manipulator robot. This type of robot was selected due to its complexity in terms of Degrees of Freedom (DoF) compared to mobile robots. Figure 6.6 shows the architecture developed to connect the real and virtual worlds.

As mentioned above, the robot used for this application was a *Kinova Jaco*². The control was distributed between the robot, which performed low-level functions such as joint control, and a computer, which performed high-level tasks such as motion planning and obstacle avoidance, both connected via USB or Ethernet. The computer received the effector goals from the interface, computed the trajectory and sent joint values to the robot.

FIGURE 6.5: Predictive virtual reality interface.

FIGURE 6.6: Architecture to connect robot and VR worlds.

FIGURE 6.7: Commanding of the manipulator.

The motion planning was performed by means of *MoveIt!* [220], which is a software for mobile manipulation that provides kinematic functionalities such as collision checking. This tool integrates the *Open Motion Planning Library (OMPL)* [221], which implements multiple sampling-based algorithms, such as RRT [122], PRM and EST.

The virtual reality interface was developed under the *Unity* environment and *SteamVR* asset for *HTC Vive* headset. The model of the robot was introduced in *Unity* split into eight parts: base, shoulder, arm, forearm, wrist 1, wrist 2, hand and fingers. Each part was implemented through two objects: one for assembling it with the previous one (considering the proper position and orientation), and another one to allow the rotation of its joint. The result was a virtual robot whose movements are identical to those of the real robot.

The robot commanding was performed through a virtual sphere with three axes. The sphere represents the position that the effector must reach, whereas the three axes defines the orientation that it must acquire. The process is shown in Figure 6.7: when the sphere is in the effector, the robot remains at rest. The operator can pick the sphere, move it within the scene and place it in another location. When the operator leaves the sphere, the goal is sent to the robot. Additionally, the operator can teleport within the scene to reach the best perspective and select the goals more comfortably.

In this case, the communication between robot and VR worlds was performed by means of the *Unity* asset *RosBridgeLib* and the ROS package *RosBridge*. Two ROS messages were used: *sensor_msgs/JointState* for sending joint angles from the robot to the interface, and *geometry_msgs/Pose* for sending goals from the interface to the robot.

The results of the tests with operators indicate that the interface enhances situational awareness and provides an intuitive commanding method. The operators emphasize the advantages of choosing the point of view to monitor the robot, as well as commanding the robot by moving the goal of its effector. Although some of the operators are not familiar with the control of this kind of robots, they are able to move it to the desired locations efficiently and safely.

FIGURE 6.8: Multi-robot complete interface: operator enjoying the immersion in the mission (left), 2D view (top-right) and headset view (down-right).

Mobile robots

The second of these interfaces was developed for commanding multiple ground and aerial robots. This application integrated both ground and aerial robots to increase the difficulty of monitoring and commanding. The ground robots were *Robotnik Summit XL* ⁴, simulated in *Gazebo* by using their corresponding package ⁵. Further information about this platform and its start up under ROS can be found in references [83] and [84]. The aerial robots were *Parrot Bebop 2*, also simulated in *Gazebo* by using the *rotor_simulator* package ⁶.

The simulations had two particularities: on the one hand, the robots were simulated in multiple *Gazebo* simulators distributed in multiple computers, in order to reproduce the integration of real robots in the architecture; and on the other hand, the robots appeared and disappeared during the mission, in order to test the robustness of the architecture and the performance of the interface.

Again, the virtual reality interface is based on the reproduction of the scenario. Figure 6.8 tries to illustrate the immersion of the operator in this scenario, as well as the view of the robots and their environment.

This interface includes a module that continuously checks the robots that are exchanging messages. If a new robot sends a message to the interface, the robot is identified and spawned in the virtual scene. The identification is performed through an ID parameter: from 0 to 100 for ground robots and from 101 to 200 for aerial robots.

⁴<http://wiki.ros.org/Robots/SummitXL>

⁵http://wiki.ros.org/summit_xl_sim

⁶http://wiki.ros.org/rotors_simulator

FIGURE 6.9: Multi-robot commanding interface: teleporting closer to robots (top-left), selecting ground robot (top-right), selecting aerial one (down-left) and generating a waypoint for it (down-right).

In addition to this, the interface provides the operator with a set of resources to ease robot control, as shown in Figure 6.9. As in the other interfaces, the operator can teleport throughout the scenario by pressing the controller's trigger and pointing to the desired destination. In this case, the operator can use the controller's touchpad to select a robot and, then, send this robot a goal. The selected robot is surrounded by a translucent sphere, to point out its selection and avoid commanding mistakes. Finally, a pop-up panel is added to the selected robot to show the images of its camera.

The communications between robot and VR worlds were addressed by following the architecture shown in Figure 6.10. Nevertheless, there were two differences: first, there was only a VR interface, and second, all the robots were simulated by *Gazebo*.

The *nav_msgs/Odometry* messages were used not only for the odometries, which were generated by the robots and sent to the interface, but also for the commands, which were generated by the interface and sent to the correspondent robots. The reason behind using this message is its flexibility: it can integrate both waypoint and speed commands, as well as collect all the information of the robot state. Specifically, *frame_id* was used to name the robot that has to send or receive the message, the *pose* message to exchange the position of the robot or the goal command, and the *twist* message served to exchange the speed of the robot or the speed command. Additionally, the *std_msgs/Header* message was used to send the selected robot and the *sensor_msgs/CompressedImage* one was used to receive the images of its camera. A good idea for future works is to define custom ROS messages that can integrate more information, such as the level of battery and payload information.

The results of the first tests with operators reveal that the interface improves the knowledge about missions and allows a close control of robots. The operators remark the teleporting resource and intuitive commanding as advantages, but the size of scenario and the remoteness of robots are proven drawbacks.

(A)

(B)

FIGURE 6.10: (A) The architecture that connects simulated and real robots to conventional and virtual reality interfaces. (B) The section of the architecture dedicated to VR interfaces in further detail.

FIGURE 6.11: Conventional interface without the predictive component.

6.4 Experiments

A series of experiments were carried out to measure the impact of immersion and prediction in multi-robot mission monitoring. For this purpose, two conventional interfaces have been developed in a equivalent way to the two virtual reality ones: one non-predictive and another predictive.

The non-predictive conventional interface is based on the one used to perform the experiments and described in Chapter 3. This interface is shown in Figure 6.11 and consists of the following panels: map, commanding, configuration, robot and payload. As the experiments involved mission monitoring but not robot commanding, some panels and some options were not used, such as the related to commands and payloads.

The map panel shows a scheme of the scenario with its main elements (center, water ...), the location of both UAVs, and the location of fire and UGV when they are detected. Under the map there is a button for each UAV that the user should click to select it. The robot panel shows the full information of the selected UAV, such as the level of battery, altitude, horizontal and vertical speed and task that is performing. Finally, the configuration panel was used to connect the interface to the telemetry, define the mode and perform the stops and starts.

The predictive conventional interface is similar to the previous one, but it includes predictions to support operators. As explained before, the predictive elements are the relevance and risk of UAVs during the mission. The relevance of UAVs was integrated by automatically selecting the most relevant UAV, instead of asking the operator to select it. The risk of UAVs was integrated by means of an indicator in the robot panel, as well as an alert sign in the map when it exceeds the 25% (as shown in Figure 6.12).

During the experiments, the operators used the four interfaces - non-predictive conventional interface (CI), predictive conventional interface (PCI), non-predictive virtual reality interface (VRI) and predictive virtual reality interface (PVRI) - to monitor multi-robot missions, as shown in Figure 6.13.

A total of 24 subjects of different age, genre and expertise were involved in the experiments. Specifically, the participants' age ranged among 21 and 34 years old, there were 9 women and 15 men and their expertise was classified according to their study level: BSc students (16), MSc students (2),

FIGURE 6.12: Conventional interface with predictive component.

FIGURE 6.13: Testing virtual reality interfaces: An operator working hard in real world (left) and what the operator is doing in virtual world (right).

TABLE 6.3: Design of experiments.

Subject	Interface	Mission
O1	VRI and CI	M1 and M2
O2	VRI and PCI	M3 and M4
O3	PVRI and CI	M5 and M6
O4	PVRI and PCI	M7 and M8
O5	CI and VRI	M8 and M7
O6	CI and PVRI	M6 and M5
O7	PCI and VRI	M4 and M3
O8	PCI and PVRI	M2 and M1
O9	VRI and CI	M8 and M7
O10	VRI and PCI	M6 and M5
O11	PVRI and CI	M4 and M3
O12	PVRI and PCI	M2 and M1
O13	CI and VRI	M1 and M2
O14	CI and PVRI	M3 and M4
O15	PCI and VRI	M5 and M6
O16	PCI and PVRI	M7 and M8
O17	VRI and CI	M5 and M6
O18	VRI and PCI	M7 and M8
O19	PVRI and CI	M8 and M7
O20	PVRI and PCI	M6 and M5
O21	CI and VRI	M4 and M3
O22	CI and PVRI	M2 and M1
O23	PCI and VRI	M1 and M2
O24	PCI and PVRI	M3 and M4

PhD students (5) and PhD (1). The participants were also asked about their experience with videogames and robot missions in a scale among 0 (never) and 5 (work/hobby).

Each participant used two different interfaces to monitor two different missions. The assignation and order of interfaces and missions were planned to compensate the influence of a learning curve on results, as well as to avoid the dependencies between interfaces and missions. In order to explain the design of experiments, the Table 6.3 shows the interfaces and missions assigned to each participant. This design of experiments provides with 24 samples to compare CIs and VRIs, and 6 samples to compare the pairs CI vs VRI, CI vs PVRI, PCI vs VRI and PCI vs PVRI. In the first case, this figure allows to make conclusions with a statistical confidence level of 95% and a statistical power of 30%. In the second one, the number of samples could be not enough to reach significant conclusions, but the cost of increasing significantly the number of participants or the number of interfaces per participant was not affordable.

Interfaces were evaluated according to the operator workload and situational awareness. We used the NASA-TLX and SAGAT questionnaires in Spanish to obtain values for these variables.

The structure of one of the 24 experiments is detailed below:

1. Explanation of missions:

- The objective of the experiment is to watch multi-robot missions, collect information and answer a series of questions.
- The goals of the missions are to detect and extinguish fires, and to find and track potential intruders.
- The mission elements are two drones (one red and another blue), a ground robot, a fire and a water well.
- The drones perform the following tasks: begin (take-off), surveillance (cover the area to detect fire or intruder), reconnaissance (visit the points to check detections), tracking (follow an intruder), capture (load the water), release (download on fire) and finish (land).
- It is important to know where the drones are, what tasks they are performing, their battery level, etc.

2. Explanation of interfaces:

- Conventional interface (CI): The map, the elements (UAVs, fire and UGV), the manual selection of UAV and the information (battery and task).
- Predictive conventional interface (PCI): The map, the elements (UAVs, fire and UGV), the predictive components (spotlight and alert), the autonomous selection of UAV and the information (battery and task).
- Virtual reality interface (VRI): The environment (scenario and platform), the teleport mechanism, the elements (UAVs, fire and UGV) and the information (battery and task).
- Predictive virtual reality interface (PVRI): The environment (scenario and platform), the teleport mechanism, the elements (UAVs, fire and UGV), the predictive components (spotlight and smoke) and the information (battery and task).

3. Annotation of user information: Age, genre and expertise.

4. NASA-TLX (weighing): The user puts into order six variables (mental, physical and temporal demands, effort, performance and frustration) according to their estimated influence on workload (as seen in Figure 6.14).

5. Test of interface #1:

- Start: The user starts to monitor the multi-robot mission.
- Stop #1: We notify the user and, after ten seconds, stop the interface.
- SAGAT (first part): The user answers some questions about the past, current and future locations and states of UAVs. The questionnaire is explained in further detail below.
- Resume: The user starts again to monitor the multi-robot mission.
- SAGAT (second part): The user answers some questions about the past, current and future locations and states of UAVs. The questionnaire is explained in further detail below.

FIGURE 6.15: Situation awareness global assessment technique (SAGAT): English translation of the questionnaire of the experiments.

TABLE 6.4: Summary of the results of the experiments.

Interface	Workload (NASA-TLX)	Situational Awareness (SAGAT)	Evaluation (+/-)
CI	853	14.91	0/7
PCI	915	14.33	0/7
VRI	638	16.25	5/5
PVRI	740	16.46	10/5

TABLE 6.5: T-test with pairs of interfaces in terms of NASA-TLX scores.

	VRI	PVRI
CI	CI > VRI Significant ($p = 0.0180$)	CI > PVRI Non-significant ($p = 0.3716$)
PCI	PCI > VRI Significant ($p = 0.0237$)	PCI > PVRI Non-significant ($p = 0.1008$)

that virtual reality interfaces tend to reduce operator workload, whereas the predictive components tend to increase it. Nevertheless, these results could be influenced by the procedure and not be statistically significant.

In order to check the statistical significance, the results have been split according to the workload variables (mental, physical and temporal demand, effort, performance, and frustration) and the interfaces (on one hand, CI vs. PCI vs. VRI vs. PVRI, and, on the other hand, CI and PCI vs. VRI and PVRI).

Figure 6.16 shows box and whisker diagrams of the workload and its variables for the four interfaces. The one-way analysis of variance (ANOVA) of the complete dataset shows there are no significant differences between the workload of the four interfaces ($F = 2.26$, $p = 0.1061$). Similar studies for the variables of workload show there are significant differences with $\alpha = 0.05$ in performance ($F = 2.59$, $p = 0.065$) and frustration ($F = 3.37$, $p = 0.0266$), while these differences do not apply to the remaining variables.

Figure 6.16 also shows the same diagrams for the interfaces now grouped in two blocks: conventional and virtual reality. The one-way ANOVA of this dataset shows there are significant difference with $\alpha = 0.05$ between the workload of both groups of interfaces ($F = 5.58$, $p = 0.0225$). Similar studies for workload variables keep these significant differences in effort ($F = 4.54$, $p = 0.0384$), performance ($F = 7.56$, $p = 0.0085$) and frustration ($F = 8.47$, $p = 0.0055$). Once again, these differences do not apply to the other three variables.

Finally, the t-test for each pair of interfaces provides the results of Table 6.5. It can be appreciated that, considering $\alpha = 0.05$, VRI is significantly better than CI and PCI in terms of workload. The rest of the differences between the pairs of interfaces cannot be considered as significant.

To sum up, we can assert that workload is significantly lower in virtual reality interfaces compared to conventional ones, both predictive and non-predictive. The effects of prediction on workload seem to be negative, but are nevertheless non significant. These unexpected results were probably due to the operators' need of training to interpret properly the predictive cues of the interfaces.

As far as situational awareness is concerned, Table 6.4 shows that the interfaces can be arranged in the following order: PVRI, VRI, CI and PCI. This order points out the virtual reality interfaces tend to increase the situational awareness of operators, whereas the effects of the predictive components depend on the interface. However, the statistical significance of these results must be checked.

Figure 6.17 shows box and whisker diagrams for situational awareness

(A)

(B)

FIGURE 6.16: Box and whiskers diagrams for the workload and its variables. (A) Plot with the four interfaces: Conventional (CI), Predictive Conventional (PCI), Virtual Reality (VRI) and Predictive Virtual Reality (PVRI). (B) Plot with two groups of interfaces: Conventional (CIs) and Virtual Reality (VRIs).

FIGURE 6.17: Box and whisker diagrams for the situational awareness with four interfaces (left) and two groups of interfaces (right).

TABLE 6.6: T-test with pairs of interfaces in terms of NASA-TLX scores.

	VRI	PVRI
CI	CI < VRI	CI < PVRI
	Non-significant ($p = 0.2584$)	Non-significant ($p = 0.3461$)
PCI	PCI < VRI	PCI < PVRI
	Non-significant ($p = 0.1011$)	Non-significant ($p = 0.0978$)

of the four interfaces (CI, PCI, VRI and PVRI) and the two groups of interfaces (CIs and VRIs). The one-way ANOVA with $\alpha = 0.05$ of the four interfaces does not provide significant results ($F = 1.49$, $p = 0.2296$). However, the same analysis with the two groups of interfaces detects a significant difference ($F = 4.37$, $p = 0.042$).

Finally, the t-test of each pair of interfaces provides the results of Table 6.6. In this case, the differences between pairs of interfaces are not statistically relevant. However, the differences of VRI and PVRI over PCI are closer to $\alpha = 0.05$ than the rest of them.

To sum up, we can state that virtual reality significantly improves the situational awareness of operators, since their score in SAGAT questionnaire is significantly higher than the score of conventional interfaces. In this case, the effects on prediction on situational awareness depend on the interface, probably due to the fact that prediction implementation with virtual reality is easier to understand than the implementation in a conventional interface. Nevertheless, these effects once again are not significant.

Finally, the questionnaires had a field of observations where users could write comments and suggestions. A total of 39 reviews about the interfaces have been collected: 15 were positive and 24 were negative. The results can be seen in Table 6.4 and are described below.

The conventional interface had 7 negative reviews, mainly related to the complexity of understanding the information of the mission (3) and the

need to select the UAVs to get their full information (2).

The predictive conventional interface also received 7 negative reviews, in this case about the amount of information (3) and the complexity to understand some variables (3).

The virtual reality interface received 5 positive and 5 negative reviews. The positive comments pointed out that the interface is easy to understand (3) and fun to use (2), whereas the negative ones reported different problems about the observation platform (3), including perspective difficulties and physical discomfort.

The predictive virtual reality interface received 10 positive and 5 negative reviews. In this case, the positive comments stated the interface is easy to understand (3) and comfortable (3), they also praised the usefulness of teleporting (1) and prediction (1). On the other hand, the negative ones reported difficulties to read some variables (3) and the mentioned problems with the observation platform (2).

By way of curiosity, the situational awareness (SA) and workload (W) scores do not show correlation with the experience with videogames (V) and robot missions (RM). Specifically, these are the correlation coefficients: V-SA (0.0102), RM-SA (-0.2048), V-W (0.0855) and RM-W (0.0675).

6.6 Contributions

This chapter addresses the development of immersive interfaces to improve the situational awareness of operators in multi-robot missions. In order to achieve this objective, it describes the virtual reality interfaces for ground, aerial and manipulator robots (Section 6.3) and performs a series of experiments to compare them against conventional ones (Section 6.5).

The main contributions of this chapter are listed below:

- It proposes one of the first immersive interfaces in the literature based on virtual reality for multi-robot missions.
- It studies the resources available to develop this kind of interfaces and creates the components required for monitoring and commanding.
- It demonstrates the virtual reality interfaces improve the situational awareness of operators without increasing their workload.

Chapter 7

Experiments and results

Look up at the stars and not down at your feet. Try to make sense of what you see, and wonder about what makes the universe exist. Be curious.

Stephen Hawking

This chapter describes the experiments that were performed in order to validate the contributions of the thesis related to mission modeling (Chapter 3), operator modeling (Chapter 4), adaptive interface (Chapter 5) and immersive interface (Chapter 6).

The aim of these experiments was to compare an adaptive & immersive interface that integrates several developments of the thesis against a state-of-art one. For this purpose, a set of operators used both interfaces to conduct two missions (surveillance and intervention) in two scenarios (outside and inside a chemical plant) with three types of robots (ground, aerial and manipulator). The results provided by these experiments in terms of performance, situational awareness and workload allow to estimate the potentialities and limitations of adaptive & immersive interfaces in the future of multi-robot missions.

The chapter collects and expands the content of the following publication:

- Roldán, J. J., Peña-Tapia, E., Garcia-Aunon, P., del Cerro, J., & Barrientos, A. Bringing adaptive & immersive interfaces to real-world multi-robot scenarios: application to surveillance and intervention in critical infrastructures (review).

The remainder of the chapter is organized as follows: Section 7.1 describes the considered scenarios (outdoor and indoor), Section 7.2 presents the interfaces to be compared (conventional vs adaptive & immersive), Section 7.3 explains the methodology of experiments and Section 7.4 collects their main results and conclusions.

7.1 Scenarios

A set of multi-robot missions has been developed to test the proposed immersive & adaptive interface against a state-of-art one. The design of these missions took into account a set of requirements in order to obtain results as relevant as possible:

- **Scenarios:** The scenarios shall be as realistic as possible, even inspired by real-world robotics applications, and sufficiently varied, including both outdoor and indoor locations. Therefore, they should be the result of a compromise between knowledge and uncertainty: i.e. some elements of the scenarios can be known prior to missions, whereas some others should be discovered during them.
- **Robots:** A fleet with multiple robots shall be used to perform missions. The robots must be heterogeneous to extend the validity of results: e.g. Unmanned Ground Vehicles (UGVs), Unmanned Aerial Vehicles (UAVs), manipulators... Some robots must be used simultaneously in certain tasks to touch operator's workload, whereas other ones can be used separately in different tasks to determine operator's performance.
- **Operators:** The operators shall perform the experiments one by one. They must be familiarized with robots to be able to develop missions successfully, but they must not have experience in this kind of missions to avoid biases in the evaluation of interfaces.
- **Other:** The experiments with the interfaces must be relevant and reproducible. For this purpose, common hardware and software resources must be used and integrated with Robot Operating System (ROS) [172].

In order to meet these requirements, the following scenarios have been defined:

An accident has happened in a chemical plant. It has been reported there is a leakage inside the building and there may be other damages outside it. The area is dangerous for people because the spilled products are toxic. A team of robots has been deployed at the entrance of the plant. The objective is to collect information and, if possible, to solve the failure. The team is formed by two UAVs and a UGV equipped with a manipulator. The mission can be divided into two phases:

- *Phase 1: The robots recognize the exterior of plant to evaluate the damages caused by accident. The UAVs can be used to quickly locate critical elements, whereas the UGV can be utilized to accurately inspect them. This phase ends when the UGV arrives to the entrance of the building.*
- *Phase 2: The ground robot equipped with the manipulator is used to solve the failure. The UGV is driven through the building towards a control panel. Once there, the manipulator is used to close a valve, stopping the leakage and finishing the mission.*

As shown in Figure 7.1, the scenario has been reproduced in the facilities of the Technical School of Industrial Engineering (ETSII) of the Technical University of Madrid (UPM). The two phases of mission correspond to the two scenarios: the outdoor one is explained in Section 7.1.1, whereas the indoor one is described in Section 7.1.2.

7.1.1 Outdoor scenario

This scenario is located in a courtyard of the university and reproduces the enclosure of the chemical plant. It is an irregular area of $1,275 \text{ m}^2$ between various buildings that contains parkings and vegetation.

FIGURE 7.1: Scenarios considered for the experiments.
Maps ©2018 Google and IGN España.

FIGURE 7.2: Mission in outdoor scenario.

As mentioned above, a ground robot and two aerial robots are used in this scenario. These robots are deployed in the begin point and must go to the end point marked in Figure 7.2. On the way, they must search and inspect a series of chemical containers, checking if they are adequately closed or present leakages.

The objective of this phase is to check if the interfaces are appropriate to monitor and control multiple robots in an outdoor, irregular and partially unknown scenario. This scenario is expected to be a challenge for operators in terms of workload, due to the amount of information to analyze and the demand of making fast and correct decisions.

The ground robot used is a *Robotnik Summit HL* [83], which has a size of 722 x 613 x 416 mm, weight of 65 kg and load capacity of 65 kg. It has an autonomy between 10 hours (continuous motion) and 40 hours (standard laboratory use), four wheels with four motors that allow it to rotate in place, and a maximum speed of 3 m/s. It is controlled by an embedded computer with Linux and Robot Operating System (ROS).

The specific unit used in this experiments is equipped with the following sensors: a Hokuyo UTM-30LX laser scanner with a field of view of 270°

FIGURE 7.3: Maps generated by SLAM algorithm.

and a range of $[0.1m, 30m]$, a Kinect 2.0 with a range of $[0.5m, 4.5m]$ and a pan-tilt-zoom camera that can be used for inspection.

In this scenario, the *Robotnik Summit HL* has to go from starting to end point and inspect the detected chemical containers. There were two options to perform this navigation: the use of Global Navigation Satellite System (GNSS) to determine its absolute position and the use of laser scanner to locate it in a previously known map. The first option was discarded because this system presents noise between the buildings of the city center. Therefore, the second option was chosen because it is suitable for the experiments and can be assumed for a real scenario. In fact, this technique only requires a previous map of the facilities and it is relatively robust to the changes in layout.

Following the architecture shown in Figure 7.4a, the robot was tele-operated to build a map of the environment. The Simultaneous Location And Mapping (SLAM) algorithm implemented in the *gmapping* package of *slam_gmapping* stack was used for this purpose (more information can be found in [81] and [80]). This package uses the range measurements of laser scanner to build a map and, at the same time, locate the robot in this map. Figure 7.3a shows the map generated by this process, which reveals the contours of buildings and some objects like cars or containers. However, other elements in the scenario are placed after building the map to reproduce the uncertainty of a real-world scenario.

During the missions, the Augmented Monte Carlo Localization (AMCL) algorithm implemented in the *amcl* package of ROS *navigation* stack was used to localize the robot in the map. Additionally, the *move_base* package of the same stack was used to generate the speed commands required to arrive to the pose goals sent by the operators. The ROS *navigation* stack manages both global and local costmaps that allows to simultaneously manage global and local plans, which is key to navigate avoiding obstacles in a partially unknown environment. Further information of this stack can be found in publications [142] and [243]. The ROS architecture used for navigating with the map can be seen in Figure 7.4b.

On the other hand, the aerial robots are simulated because of the logistic and regulation problems of flying in the city center. In this scenario, there are two drones that have to cover the scenario and search the chemical

FIGURE 7.4: Simplified ROS architectures of the ground robot. Squares represent nodes and ovals represent topics.

containers. Therefore, the aerial robots take advantage of their velocity to find the targets, whereas the ground robot uses their payload to get more information about them.

In order to do this, a ROS node named *drone_simulator* has been developed. This node receives the target positions for the drones and generates both their trajectories and detections.

The dynamics of drones is shown by equations 7.1, 7.2 and 7.3. The first one computes the velocity $\mathbf{v}(t_i)$ as the product of a module $v(t_i)$ and a unit vector $\mathbf{u}(t_i)$. The second one calculates the module of velocity at a certain time $v(t_i)$ through the commanded velocity at this time $v_c(t_i)$ and the actual velocity at the previous time $v(t_{i-1})$, in order to avoid sharp changes in velocity and stop at the goal positions. Finally, the third one computes the unit vector that represents the direction $\mathbf{u}(t_i)$ as the difference between goal $\mathbf{g}(t_i)$ and current $\mathbf{p}(t_i)$ positions divided by the distance d .

$$\mathbf{v}(t_i) = v(t_i)\mathbf{u}(t_i) \quad (7.1)$$

$$v(t_i) = (1 - e^{-\frac{(t_i - t_{i-1})}{\tau}})(1 - e^{-\frac{d}{4}})v_c(t_i) + e^{-\frac{(t_i - t_{i-1})}{\tau}}v(t_{i-1}) \quad (7.2)$$

$$\mathbf{u}(t_i) = \frac{1}{d}(\mathbf{g}(t_i) - \mathbf{p}(t_i)) \quad (7.3)$$

7.1.2 Indoor scenario

This scenario is located in a research laboratory of the university and reproduces a control room of the chemical plant. As shown in Figure 7.6, it is a

FIGURE 7.5: ROS architecture to manage simulated drones. Squares represent nodes and ovals represent topics.

FIGURE 7.6: Mission in indoor scenario.

5 x 5 x 3 m room with a control panel in one of its corners. As also shown in this figure, the control panel consists of two tubes that join into a unique one and three valves to control the three flow rates.

As mentioned above, a ground robot with a manipulator has arrived to this room and it is at the begin point. The ground robot has to move throughout the room avoiding the debris to arrive the control panel. Then, the manipulator has to close one of the valves to stop the leakage and finish the mission.

The objective of this phase is to test if the interfaces are appropriate to command a complex robot (mobile manipulator robot) performing a critical task (close a valve). This scenario tests the accuracy of operators in both perception and action, since they have to place the mobile manipulator close to the panel and its hand over the handle of valve.

In this scenario, it was used not only the *Robotnik Summit HL* mobile robot, but also the *KinovaJaco²* manipulator joined to it [84]. This manipulator has six articulations (shoulder, arm, forearm, wrist 1, wrist 2 and hand) and an effector with three fingers. It has a total weight of 5.5 kg, can reach a maximum distance of 984 mm and can manipulate a maximum payload of 2.4 kg.

Similarly to the other scenario, the SLAM algorithm was used to build a map (shown in Figure 7.3b) and the AMCL one to localize the robot in

the map. In contrast to this scenario, the robot was controlled by means of a joystick instead of commanded by sending goals. This fact is justified because the locations of the ground robot must be accurate to facilitate the actions of the manipulator one.

In the case of manipulator, the desired positions and orientations of gripper are sent to the controller node, which call the services of *MoveIt!* to plan and execute the trajectories.

7.2 Interfaces

As mentioned above, the experiments compare an interface with the developments of thesis is with another based on the state-of-art. These interfaces are described in the following sections: the conventional one in Section 7.2.1 and the adaptive & immersive one in Section 7.2.2.

7.2.1 Conventional interface

RViz is the most common 3D visualizer for Robot Operating System (ROS). It provides a set of displays and controls that allows the monitoring and commanding of a wide set of robots. In fact, this tool has been used with mobile robots [233, 224, 87], drones [209, 24], manipulators [76, 246], mobile manipulators [36, 79] and humanoid robots [13, 92, 178, 225].

RViz was chosen to develop the conventional interface due to its simplicity and modularity. Two configurations were created: one for the outdoor scenario and another for the indoor scenario. Each configuration was designed taking into account the required tools for each specific mission.

The outdoor interface (shown in Figure 7.7a) integrates the model of ground robot, three maps (one of the static scenario for localization, other of the permitted areas for planning and other with dynamics obstacles for navigation), the long and short term trajectories, the locations of virtual drones and the detections of boxes. The commanding of robots in scenario is performed by means of navigation goals. The operator just has to select the robot to command and click the destination in the map.

The indoor interface (shown in Figures 7.7b and 7.7c) integrates elements for controlling both the ground robot and its manipulator. For the ground robot, it includes the model with the measurements of laser scanner and images of axis camera, as well as two maps dedicated to the static scenario and dynamic obstacles to support the teleoperation. For the manipulator robot, it includes the complete model with its joints and effector. In this interface, the operator controls the ground robot by using a joystick, since the scenario is smaller and the control must be closer, and the manipulator one by means of a sphere with three axes.

7.2.2 Adaptive & immersive interface

The adaptive & immersive interface was developed by using *Unity* game engine and *SteamVR* plugin. In the same way that the previous one, this interface has two configurations: one for outdoor and another for indoor scenario.

A capture of the interface configured for outdoor scenario is shown in Figure 7.8. Some of the elements of the interface can be seen in this image:

(A) Outdoor configuration.

(B) Indoor configuration for ground robot.

(C) Indoor configuration for manipulator robot.

FIGURE 7.7: Captures of conventional interface.

FIGURE 7.8: Capture of adaptive & immersive interface in outdoor scenario.

the recreation of scenario, the models of robots, a cube with the images of camera, the arc of teleport...

The main monitoring resources of this configuration are explained below:

- **Scenario:** The outdoor scenario is recreated in 3D as it was before the accident. This scenario represents the actual location of buildings, pavements and gardens, but some elements may have changed, such as the location of cars and containers.
- **Robots:** The models of UGV and UAVs are represented with their actual positions and orientations in the scenario.
- **Camera:** The images of camera are shown in the faces of a cube, which can always be found over the UGV. In this manner, the operator can combine the perspective of virtual scenario with the images of real environment.
- **Hints:** The interface can highlight some robots and their potential destinations during the missions, in order to help the operator to efficiently use the resources and explore the map.

And the commanding resources used in this configuration are described below:

- **Touchpad:** It is only used to command the robots in scenario. The operator has to press this button, point a ground or aerial robot and release it to select the robot. Then, he has to press the button, point the destination and release it to send the goal.
- **Trigger:** It is used to teleport the operator in the scene and control the transmission of images. For the first purpose, the operator has to press the button, point the destination and release it to teleport to this location. For the second one, he has to put the controller over the cube and press this button to activate or deactivate it.

FIGURE 7.9: Captures of adaptive & immersive interface in indoor scenario.

Figure 7.9 shows a sequence of captures of the interface configured for indoor scenario. These images show some of the elements of this configuration, such as the model of robot, the control panel, a cube with the images of camera, the commanding method for manipulator...

The main monitoring resources of this configuration are explained below:

- **Scenario:** The indoor scenario is modeled in the same way that the outdoor one. In this case, it represents the structure of room and the location of panel, but does not take into account the potential obstacles produced by the accident.
- **Robots:** The model of mobile manipulator is represented with its actual position and orientation in the scenario.
- **Camera:** The images of camera are shown in the faces of a cube, which can always be found over the UGV. In this manner, the operator can combine the perspective of virtual scenario with the images of real environment.
- **Hints:** The interface can follow the development of mission and support the operator in the control of robot to the panel or the location of gripper on the valve.

And the commanding resources applied in this configuration are described below:

- **Touchpad:** It is used to control the velocity of ground robot. The linear velocity can be modified by pressing this button in forward or backward direction, whereas the angular velocity by pressing it in left or right direction. The magnitude varies from 0% (do not touch the button) to 100% (touch the contour of button).

TABLE 7.1: Design of experiments.

Operator	A	B	C	D
1	Out-CI	Out-A&II	In-CI	In-A&II
2	Out-CI	Out-A&II	In-A&II	In-CI
3	Out-A&II	Out-CI	In-A&II	In-CI
4	Out-A&II	Out-CI	In-CI	In-A&II
5	In-CI	In-A&II	Out-CI	Out-A&II
6	In-CI	In-A&II	Out-A&II	Out-CI
7	In-A&II	In-CI	Out-A&II	Out-CI
8	In-A&II	In-CI	Out-CI	Out-A&II

- **Trigger:** It is used to teleport the operator, control the manipulator robot and manage the transmission of images. For the first purpose, the operator has to press the button, point the destination and release it to teleport to this location. For the second one, she has to put the controller over the gripper, press the button to take it, move to the desired destination and release the button to send it. For the third one, she has to put the controller over the screen and press this button to activate or deactivate it.
- **Gripper:** It can be used to open and close the hand of manipulator robot.

7.3 Experiments

Eight operators with different ages (from 21 to 32 years old), genre (5 men and 3 women) and studies (4 BSc and 4 PhD students) took part in the experiments. Each of them performed four trials: one for each combination of scenario and interface: outdoor with conventional interface (Out-CI), outdoor with adaptive & immersive interface (Out-A&II), indoor with conventional interface (In-CI) and outdoor with adaptive & immersive interface (In-A&II). The scenarios and interfaces were ordered as shown in Table 7.1, in order to compensate the influence of learning on the results.

The following procedure was followed in each of the trials:

1. Explanation of mission:

- **Outdoor:** *In this mission you have to drive the ground robot from the begin point at the entrance of the plant to the end point at the entrance of the building. Meanwhile, you can use the aerial robots to search the boxes along the scenario and the ground one to check if they are opened or closed.*
- **Indoor:** *In this mission you have to drive the ground robot through the room from the begin point to the control panel. There, you have to place the manipulator's gripper over the valve's handle in order to stop the leakage.*

2. Explanation of interface:

- Conventional: Description of map, commanding methods (joy-pad for ground robot and sphere for manipulator in indoor scenario, and navigation goals for ground and aerial robots in outdoor scenario), potential detections, images of cameras...
 - Adaptive & immersive: Description of environment, commanding methods (speed commands for ground robot and goal commands for manipulator in indoor scenario, and navigation goals for all the robots in outdoor scenario), images of cameras, hints...
3. Annotation of user information: Age, genre, studies, experience with robots and experience with videogames.
 4. NASA-TLX questionnaire (weighing): Evaluation of six variables (mental demand, physical demand, temporal demand, effort, performance and frustration) according to their potential influence on workload.
 5. Performance of mission.
 6. Measurement of performance:
 - Outdoor: Annotation of time, detected and identified boxes and errors.
 - Indoor: Annotation of time, relative location of ground robot to panel, relative location of manipulator's gripper to valve and errors.
 7. NASA-TLX questionnaire (scoring): Evaluation of the mission performed by using the interface taking into account six variables (mental demand, physical demand, temporal demand, effort, performance and frustration).
 8. SAGAT questionnaire:
 - Outdoor: Draw in a map the paths of robots, as well as the location and state of the boxes.
 - Indoor: Draw in a map the path of ground robot, the state of panel and the location of gripper.

As mentioned above, NASA-TLX and SAGAT tests were used to estimate the workload and situational awareness of operators respectively. Figure 7.10 shows the questionnaires that the operators had to fill after each of the missions. As shown, NASA-TLX allows to estimate the workload of operators using each interface, whereas SAGAT tests their knowledge about the mission, robots and targets.

Some situations of missions are shown in Figure 7.11. The search for dangerous objects outside the plant is shown in Figures 7.11a and 7.11b, whereas the intervention inside the building is shown in Figures 7.11c and 7.11d.

The outdoor missions were evaluated taking into account the performance, situational awareness and workload as explained below:

- Performance: This score takes into account both the mission time and operator actions and is comprised among 0 (the worst mission) and 100 (the perfect one).

FIGURE 7.11: Workload and situational awareness tests.

- Time: A 50% of score is assigned according to the mission time, which varies from 0 points (600 seconds) to 100 points (0 seconds).
 - Actions: The other 50% of score is obtained by going from begin to end point (10%) and detecting and identifying all the boxes (40%).
 - Errors: Any type of collision of the robot against its environment is penalized by increasing 10 points the total score of performance.
- Situational Awareness: This score is computed from the answers to the SAGAT questionnaire and includes the knowledge of operator about the trajectories of robots (50%) and the location and state of boxes (50%).
 - Workload: This score is calculated from the answers to the NASA-TLX questionnaire and computes the workload through the weighting and evaluation of the previously mentioned variables according to the operator's criteria.

In the same manner, the indoor missions were evaluated taking into account the performance, situational awareness and workload as explained below:

- Performance: This score takes into account both the mission time and operator actions and is comprised among 0 (the worst mission) and 100 (the perfect one).

- Time: The time to drive the robot from begin point to control panel counts a 20% of score, whereas the time to place the manipulator over the valve sums another 20% of score. Both variables are normalized in the way that 0 seconds suppose 0 points and more than 240 seconds imply 100 points.
 - Location: The error in the relative location of ground robot to control panel counts the 20% of score. The optimal location considers the robot is aligned to the center of panel and distanced 200 mm from it (100 points), whereas the worse one considers a deviation of more than 400 mm (0 points). Additionally, the angle formed by ground robot and control panel counts the 10% of score. An angle of 0° scores 100 points, whereas an angle of less than -45° or more than 45° scores 0 points. Finally, the error in the relative location of the manipulator's hand to the valve's handle counts a 30% of score. In this case, the optimal location considers a null distance between these elements (100 points), whereas the worst location considers a deviation of 400 mm (0 points).
 - Errors: Any type of collision of the robot against its environment is penalized by decreasing 10 points the total score of performance.
- Situational Awareness: This score is computed from the answers to the SAGAT questionnaire and integrates the knowledge of operator about the trajectory of ground robot (33%), the location of the manipulator (33%) and the state of valves (33%).
 - Workload: This score is calculated from the answers to the NASA-TLX questionnaire and computes the workload through the weighting and evaluation of the previously mentioned variables according to the operator's criteria.

Finally, the implementation of experiments required the use of two computers: one with *Ubuntu 16.04* to launch the robots and execute the RViz interface, and another with *Windows 8* to execute the VR interface. The communications between robots and computers were performed through a wireless network generated by one router in indoor scenario and three routers in outdoor scenario.

7.4 Results

The main results of experiments are collected in Table 7.2. The workload, situational awareness and performance scores are normalized from 0 to 100. In the case of workload, the best score is the lowest one, whereas in the rest of variables, the best score is the highest one. As shown by this table, the adaptive & immersive interface (A&II) has better results than the conventional one (CI) in terms of workload, situational awareness and performance in both scenarios. The figures in bold indicates that these results are statistically relevant according to the further study. These results are analyzed with more detail in the following sections.

TABLE 7.2: Summary of results.

Scenario	Interface	Workload (NASA-TLX)	S.A. (SAGAT)	Performance
Out	CI	45.04	77.91	60.53
Out	A&II	36.58	89.79	74.16
In	CI	54.33	62.90	51.25
In	A&II	43.33	63.74	58.63

7.4.1 Workload

The results of workload in indoor scenario are collected by Figure 7.12. As shown, the CI provides better results of mental and physical demands, but the A&II has better results in temporal demand, effort, performance, frustration and total workload. A t-test with $\alpha = 0.05$ reveals these results are not statistically significant in this scenario, being the ones related to temporal demand ($p=0.0732$) and performance ($p=0.0719$) which are closest to be relevant.

The results of workload in indoor scenario are shown in Figure 7.13. As can be seen, the A&II provides better results of mental demand, temporal demand, effort, performance, frustration and total workload, whereas the CI only has better results in physical demand. According to a t-test with $\alpha = 0.05$, the results related to mental demand ($p=0.0142$), temporal demand ($p=0.0093$), effort ($p=0.0047$), frustration ($p=0.0153$) and total workload ($p=0.0015$) are significant in this scenario.

7.4.2 Situational awareness

The results of situational awareness in outdoor scenario are shown in Figure 7.14. In this case, the average result of A&II is clearly better than CI (89.78 vs 77.91) and a t-test with $\alpha = 0.05$ demonstrates this difference is significant ($p=0.0228$).

The results of situational awareness in indoor scenario are shown in Figure 7.15. Although the result of A&II is also slightly better than CI in this scenario (63.74 vs 62.90), the t-test reveal this difference is not significant with $\alpha = 0.05$.

7.4.3 Performance

The detailed results of performance in indoor scenario are collected by Table 7.3 and Figure 7.16. As shown, A&II is better than CI, since it has less mission time, more completion and detection rates and less errors. According to a t-test with $\alpha = 0.05$, these differences are significant in mission time and total score.

The detailed results of performance in indoor scenario are collected by Table 7.4 and Figure 7.17. As can be seen, the CI is better in terms of time and deviation in the control of ground robot, whereas the A&II is better in terms of time and deviation in the control of manipulator robot, produces less errors commanding both robots and has the best global score. However, the t-test with $\alpha = 0.05$ reveals these differences are only significant in mission times and global score.

FIGURE 7.12: Box and whiskers diagrams for workload and its related variables in outdoor scenario.

FIGURE 7.13: Box and whiskers diagrams for workload and its related variables in indoor scenario.

FIGURE 7.14: Box and whiskers diagrams for situational awareness in outdoor scenario.

FIGURE 7.15: Box and whiskers diagrams for situational awareness in indoor scenario.

FIGURE 7.16: Box and whiskers diagrams for performance and its related variables in outdoor scenario.

FIGURE 7.17: Box and whiskers diagrams for performance and its related variables in indoor scenario.

TABLE 7.3: Results of performance for outdoor scenario.

Interface	CI	A&II
Time	388 secs	307 secs
Completion	87.50%	100%
Detections	96.25%	100%
Errors	0.5	0.14
Total	60.53	74.16

TABLE 7.4: Results of performance for indoor scenario.
GR means Ground Robot, whereas MR means Manipulator Robot.

Interface	CI	A&II
Time (GR)	70 secs	115 secs
Deviation (GR)	175 mm	193 mm
Errors (GR)	0.125	0
Time (MR)	171 secs	80 secs
Deviation (MR)	173 mm	157 mm
Errors (MR)	0.25	0
Total	51.24	58.63

7.5 Contributions

This chapter validates the contributions of thesis related to mission and operator modeling and adaptive and immersive interfaces. For this purpose, two scenarios outside and inside a chemical plant are recreated (Section 7.1), a state-of-art and an adaptive & immersive interface are developed (Section 7.2), a set of experiments with eight operators are performed (Section 7.3) and their results in terms of workload, situational awareness and performance are analyzed (Section 7.4).

The main contributions of this chapter are summarized below:

- As far as we know, this is the first time that an adaptive & immersive interface is validated in the control and monitoring of multi-robot missions in a realistic scenario.
- The results of experiments show this kind of interface is promising, since it not only can improve workload and situational awareness, but also has impact on performance.

Chapter 8

Conclusions

*We know very little, and yet it is
astonishing that we know so much, and
still more astonishing that so little
knowledge can give us so much power.*

Bertrand Russell

This chapter summarizes the main conclusions of the thesis (Section 8.1), describes its main contributions to the state of art (Section 8.2), enumerates its main achievements in terms of publications (Section 8.3), describes the areas where the developments have been applied (Section 8.4) and explains the future works that it has opened (Section 8.5).

8.1 Conclusions

The first result of the thesis is a better understanding about multi-robot missions: the requirements of missions, classes of fleets, challenges for operators, types of interfaces, issues of human-system interaction and related disciplines. As shown in Figure 8.1, the thesis has totally or partially covered most of the topics identified in the state of art.

Entering in the developments of thesis, it has been concluded that process mining is a suitable tool to automatically model multi-robot missions. Among all the possible resources, the Petri nets have been validated to follow mission state and decision trees to predict its evolution. The study of the discovery algorithms available in process mining reveals inductive miner as the best one in the context of robotics.

Regarding the modeling of operators, it has been demonstrated the relevance of classifying the operators before predicting their preferences. In fact, the use of data from clusters of operators, instead of from the single operators or the whole population, allows to make better predictions in all the studied cases.

Additionally, the study of available techniques reveals neural networks as the most promising to transfer functions from operators to interfaces in adaptive interfaces. In fact, the experiments show the performance of these models to reproduce the evaluations of missions performed by operators.

Finally, it has been concluded that immersive interfaces based on virtual reality are suitable for the monitoring and commanding of multi-robot missions. According to the experiments, this kind of interfaces improve the situational awareness of operators without increasing their workload.

FIGURE 8.1: Cloud of topics related to multi-robot missions. The shaded area represents the ones covered by this thesis.

The experiments have demonstrated that an adaptive & immersive interface with the developments of the thesis can improve the workload, situational awareness and performance of operators in multi-robot missions.

8.2 Contributions

The main contributions of the thesis to the state of art are listed below:

- As far as we know, this thesis is the first work that applies process mining to real robot missions, demonstrating that these techniques are suitable to model, analyze and improved this kind of missions.
- It proposes a new methodology to classify the operators and predict their preferences and validates it with a relevant amount of data provided by a serious game.
- It applies neural networks to model the evaluation of missions and successfully transfer this task from operators to interfaces.
- To our knowledge, this thesis is the first work to develop a complete immersive interface based on virtual reality and validate it in the monitoring and commanding of real robot missions.

8.3 Achievements

To this day, the thesis has produced the 6 journal papers that are listed below. All of these papers were published in journals included in Journal Citation Reports: five of them as Q1 - Sensors (3), Journal of Field Robotics (1) and IEEE Intelligent Systems (1) - and the other one as Q2 - Autonomous Robots - in their respective categories. The three first papers (1-3 in the list) performed a study of multi-robot missions that motivated this work and also applied some of its developments to areas such as security and agriculture. The three last papers (4-6 in the list) directly present the developments of the thesis and validate them with experiments.

1. Roldán, J. J., Joossen, G., Sanz, D., del Cerro, J., & Barrientos, A. (2015). Mini-UAV based sensory system for measuring environmental variables in greenhouses. *Sensors*, 15(2), 3334-3350 [191].
2. 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 [69].
3. Roldán, J. J., Garcia-Aunon, P., Garzón, M., de León, J., del Cerro, J., & Barrientos, A. (2016). Heterogeneous multi-robot system for mapping environmental variables of greenhouses. *Sensors*, 16(7), 1018 [190].
4. Roldán, J. J., Olivares-Méndez, M. A., del Cerro, J., & Barrientos, A. (2017). Analyzing and improving multi-robot missions by using process mining. *Autonomous Robots*, 1-19 [188].

5. Roldán, J. J., del Cerro, J., & Barrientos, A. (2017). Using process mining to model multi-UAV missions through the experience. *IEEE Intelligent Systems*, 32(4), 40-47 [186].
6. Roldán, J. J., Peña-Tapia, E., Martín-Barrio, A., Olivares-Méndez, M. A., Del Cerro, J., & Barrientos, A. (2017). Multi-robot interfaces and operator situational awareness: Study of the impact of immersion and prediction. *Sensors*, 17(8), 1720 [192].

Additionally, 13 conference papers have been published during the thesis: 6 of them in international conferences (among them the IEEE International Conference on Systems, Man and Cybernetics 2016, IEEE Mediterranean Conference on Control and Automation 2015 and ROBOT Iberian Conference 2015 and 2017) and the rest in national conferences. Some of these conference papers present developments of thesis (the case of 1, 2, 4, 5, 6, 8 and 10 of the list), whereas the rest present applications in other areas (the case of 3, 7, 9, 11, 12 and 13 of the list).

1. J.J. Roldán, J. del Cerro and A. Barrientos. Minería de procesos aplicada a las misiones robóticas. XXXVI Jornadas Nacionales de Automática. Bilbao, September 2-4th 2015 [185].
2. J.J. Roldan; J. del Cerro and A. Barrientos. A proposal of methodology for multi-UAV mission modeling, 23th Mediterranean Conference on Control and Automation (MED), pp.1-7, June 16-19th 2015 [184].
3. A. Ruiz-Larrea, J.J. Roldán, M. Garzón, J. del Cerro and A. Barrientos. A UGV approach to measure the ground properties of greenhouses. ROBOT2015: Second Iberian Robotics Conference. 2015 [204].
4. J.J. Roldán, B. Lansac, J. del Cerro and A. Barrientos. A proposal of multi-UAV mission coordination and control architecture. ROBOT2015: Second Iberian Robotics Conference. 2015 [187].
5. J.J. Roldán, J. del Cerro and A. Barrientos. Multiple robots, single operator: considerations about information and commanding. RoboCity16: Open Conference on Future Trends in Robotics, Madrid, Spain, May 26-27th 2016 [195].
6. F. Díez, J.J. Roldán and A. Barrientos. Modelado y Análisis de Misiones Multi-Robot mediante Minería de Procesos. XXXVII Jornadas Nacionales de Automática. Madrid, September 7-9th 2016 [48].
7. M. Taboada, J.J. Roldán and A. Barrientos. Interfaz Inteligente y Adaptativa para Personas con Discapacidad. XXXVII Jornadas Nacionales de Automática. Madrid, September 7-9th 2016 [222].
8. J.J. Roldán, P. Garcia-Aunon, J. del Cerro and A. Barrientos. Determining mission evolution through UAV telemetry by using decision trees. 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 000188-000193, 2016 [189].
9. J.J. Roldán, M. Garzón, J. de León, D. Garzón-Ramos, A. Martín-Barrio, S. Terrile, P. García-Aunon, J. del Cerro, C. Rossi and A. Barrientos. Proyecto PRIC: Protección Robotizada de Infraestructuras Críticas. Jornadas Nacionales de Robótica. Valencia, June 8-9th 2017 [196].

10. E. Peña-Tapia, J.J. Roldán, M. Garzón, A. Martín-Barrio and A. Barrientos. Interfaz de Control para un Robot Manipulador mediante Realidad Virtual. XXXVIII Jornadas Nacionales de Automática. Gijón, September 6-8th 2017 [169].
11. K. Garapati, J.J. Roldán, M. Garzón, J. del Cerro and A. Barrientos. A Game of Drones: Game Theoretic Approaches for Multi-Robot Task Allocation in Security Missions. Robot 2017: Third Iberian Robotics Conference. Seville, November 22-24th 2017. Springer Cham, p. 855-866, 2017 [68].
12. F. Navarro, J. Fdez, M. Garzón, J.J. Roldán and A. Barrientos. Integrating 3D reconstruction and virtual reality: a new approach for immersive teleoperation. Robot 2017: Third Iberian Robotics Conference. Seville, November 22-24th 2017. Springer Cham, 2017 [158].
13. E. Crespo, J.J. Roldán, A. Martín-Barrio, J. del Cerro, A. Barrientos. Virtual reality and process mining applied to operator training in complex assembly tasks. IV Seminario de Innovación Docente en Automática (CEA-IFAC). León, January 10-12, 2018 [37].

Finally, 3 book chapters have been published including contents of the thesis. The first one served as motivation for the work, the second one applied some concepts of thesis to agriculture and the third one directly contains developments of the thesis.

1. M. Garzón, J. Valente, J. J. Roldán, D. Garzón-Ramos, J. de León, A. Barrientos and J. del Cerro. Using ROS in Multi-robot Systems: Experiences and Lessons Learned from Real-World Field Tests. Robot Operating System (ROS), Volume 2, pp. 449-483. Springer International Publishing, 2017 [71].
2. J.J. Roldán, J. del Cerro, D. Garzón-Ramos, P. Garcia-Aunon, M. Garzón, J. de León and A. Barrientos. Robots in Agriculture: State of Art and Practical Experiences. Service Robots. In Tech, 2018 [194].
3. J.J. Roldán, E. Peña-Tapia, D. Garzón-Ramos, J. de León, M. Garzón, J. del Cerro and A. Barrientos. Multi-Robot Systems, Virtual Reality and ROS: Developing a new generation of operator interfaces. Robot Operating System (ROS), Volume 3, pp. TBD. Springer International Publishing, 2018 [193].

8.4 Applications

Finally, as shown in Figure 8.2, the developments of thesis have been applied to multiple scenarios:

- The mission modeling and monitoring techniques were applied in the context of agriculture to control heterogeneous teams of robots in the environmental monitoring of greenhouses.
- An adaptive interface was developed to supervise multi-UAV missions of surveillance, reconnaissance and tracking in the context of the SAVIER project of Airbus Defence & Space.

- An adaptive interface based on process mining techniques (Petri nets and decision trees) was developed to assist disabled people in daily life.
- All the developments of thesis related to mission and operator modeling and adaptive and immersive interfaces have been validated by means of security and search and rescue missions.
- A combination of process mining and virtual reality is being used to develop a training system for industrial operators in assembly tasks.
- An adaptive and immersive interface is being developed in the context of SwarmCity Project, which aims at applying a flying swarm to monitor the variables of a city: traffic, pedestrians, climate, pollution...

8.5 Future works

The thesis have opened the following works to be performed in the future:

- The generalization of the use of neural networks and the potential application of deep learning to transfer functions from operator to interface: e.g. evaluating the mission, asking for information, making decisions, generating commands...
- The application of adaptive and immersive interfaces not only to the control and monitoring of robots during the mission, but also to the processing of the collected information in the scenario.

(A)

(B)

(C)

(D)

FIGURE 8.2: Applications of thesis developments to greenhouse agriculture, search and rescue and city monitoring.

Bibliography

- [1] W. Van der Aalst. "Process mining: Discovery, conformance and enhancement of business processes". In: Springer-Verlag, Berlin, 2011.
- [2] Wil Van der Aalst, Arya Adriansyah, and Boudewijn van Dongen. "Replaying history on process models for conformance checking and performance analysis". In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 2.2 (2012), pp. 182–192.
- [3] Wil Van der Aalst, Ton Weijters, and Laura Maruster. "Workflow mining: Discovering process models from event logs". In: *IEEE Transactions on Knowledge and Data Engineering* 16.9 (2004), pp. 1128–1142.
- [4] Wil MP Van der Aalst. "The application of Petri nets to workflow management". In: *Journal of circuits, systems, and computers* 8.01 (1998), pp. 21–66.
- [5] Wil MP Van der Aalst and Minseok Song. "Mining Social Networks: Uncovering interaction patterns in business processes". In: *International Conference on Business Process Management*. Springer. 2004, pp. 244–260.
- [6] Wil MP van der Aalst et al. "Business process mining: An industrial application". In: *Information Systems* 32.5 (2007), pp. 713–732.
- [7] Wil MP van der Aalst et al. "Process mining: A two-step approach using transition systems and regions". In: *BPM Center Report BPM-06-30, BPMcenter.org* 6 (2006).
- [8] Ahmad Abbadi and Václav Přenosil. "Safe path planning using cell decomposition approximation". In: *Distance Learning, Simulation and Communication* 8 (2015).
- [9] Bart Adams and Frank Suykens. "Astute: Increased Situational Awareness through proactive decision support and adaptive map-centric user interfaces". In: *Intelligence and Security Informatics Conference (EISIC), 2013 European*. IEEE. 2013, pp. 289–293.
- [10] NATO Standardization Agency. "STANAG 4586". In: *Standard Interface of the Unmanned Control System (UCS) for NATO UAV interoperability* (2012).
- [11] James F Allen. "Maintaining knowledge about temporal intervals". In: *Communications of the ACM* 26.11 (1983), pp. 832–843.
- [12] Karl P Arnold. "The UAV Ground Control Station: Types, Components, Safety, Redundancy, and Future Applications". In: *International Journal of Unmanned Systems Engineering*. 4.1 (2016), p. 37.
- [13] João Avelino et al. "Experiments with Vizzy as a Coach for Elderly Exercise". In: (2018).

- [14] Hasan Ayaz et al. "Optical brain monitoring for operator training and mental workload assessment". In: *Neuroimage* 59.1 (2012), pp. 36–47.
- [15] Kenneth D Bailey. *Typologies and taxonomies: an introduction to classification techniques*. Vol. 102. Sage, 1994.
- [16] Rodrigo Coelho Barros et al. "A survey of evolutionary algorithms for decision-tree induction". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42.3 (2012), pp. 291–312.
- [17] Jenay Beer, Arthur D Fisk, and Wendy A Rogers. "Toward a framework for levels of robot autonomy in human-robot interaction". In: *Journal of Human-Robot Interaction* 3.2 (2014), p. 74.
- [18] Calin Belta et al. "Symbolic planning and control of robot motion [grand challenges of robotics]". In: *IEEE Robotics & Automation Magazine* 14.1 (2007), pp. 61–70.
- [19] Amine M Bensaid et al. "Partially supervised clustering for image segmentation". In: *Pattern recognition* 29.5 (1996), pp. 859–871.
- [20] Joël Billieux et al. "Problematic involvement in online games: A cluster analytic approach". In: *Computers in Human Behavior* 43 (2015), pp. 242–250.
- [21] Rainer Bischoff, Ulrich Huggenberger, and Erwin Prassler. "Kuka youbot-a mobile manipulator for research and education". In: *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE. 2011, pp. 1–4.
- [22] C Bishop. "Pattern Recognition and Machine Learning (Information Science and Statistics), 1st edn. 2006. corr. 2nd printing edn". In: *Springer, New York* (2007).
- [23] Roger K Blashfield. "Mixture model tests of cluster analysis: Accuracy of four agglomerative hierarchical methods." In: *Psychological Bulletin* 83.3 (1976), p. 377.
- [24] K Boudjit and Chevif Larbes. "Detection and implementation autonomous target tracking with a Quadrotor AR. Drone". In: *Informatics in Control, Automation and Robotics (ICINCO), 2015 12th International Conference on*. Vol. 2. IEEE. 2015, pp. 223–230.
- [25] Arne Brutschy et al. "Self-organized task allocation to sequentially interdependent tasks in swarm robotics". In: *Autonomous agents and multi-agent systems* 28.1 (2014), pp. 101–125.
- [26] Joos CAM Buijs, Boudewijn F Van Dongen, and Wil MP van der Aalst. "On the role of fitness, precision, generalization and simplicity in process discovery". In: *OTM Confederated International Conferences "On the Move to Meaningful Internet Systems"*. Springer. 2012, pp. 305–322.
- [27] Yifan Cai and Simon X Yang. "A survey on multi-robot systems". In: *World Automation Congress (WAC), 2012*. IEEE. 2012, pp. 1–6.
- [28] Tadeusz Caliński and Jerzy Harabasz. "A dendrite method for cluster analysis". In: *Communications in Statistics-theory and Methods* 3.1 (1974), pp. 1–27.

- [29] Luciano Cantelli et al. "UAV/UGV cooperation for surveying operations in humanitarian demining". In: *2013 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE. 2013, pp. 1–6.
- [30] Hyunjin Chang and Taeseok Jin. "Command fusion based fuzzy controller design for moving obstacle avoidance of mobile robot". In: *Future information communication technology and applications*. Springer, 2013, pp. 905–913.
- [31] Jessie YC Chen and Michael J Barnes. "Human-agent teaming for multirobot control: A review of human factors issues". In: *IEEE Transactions on Human-Machine Systems* 44.1 (2014), pp. 13–29.
- [32] Jessie YC Chen, Michael J Barnes, and Michelle Harper-Sciarini. "Supervisory control of multiple robots: Human-performance issues and user-interface design". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 41.4 (2011), pp. 435–454.
- [33] Jessie YC Chen, Ellen C Haas, and Michael J Barnes. "Human performance issues and user interface design for teleoperated robots". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 37.6 (2007), pp. 1231–1245.
- [34] Gianfranco Ciardo, Reinhard German, and Christoph Lindemann. "A characterization of the stochastic process underlying a stochastic Petri net". In: *IEEE Transactions on Software Engineering* 20.7 (1994), pp. 506–515.
- [35] Christopher M Clark. "Probabilistic road map sampling strategies for multi-robot motion planning". In: *Robotics and Autonomous Systems* 53.3 (2005), pp. 244–264.
- [36] Sven Cremer et al. "Investigation of human-robot interface performance in household environments". In: *Sensors for Next-Generation Robotics III*. Vol. 9859. International Society for Optics and Photonics. 2016, p. 985904.
- [37] Elena Crespo Afonso et al. "Virtual reality and process mining applied to operator training in complex assembly tasks". In: (2018).
- [38] Rongxin Cui, Ji Guo, and Bo Gao. "Game theory-based negotiation for multiple robots task allocation". In: *Robotica* 31.06 (2013), pp. 923–934.
- [39] Mary L Cummings and Paul J Mitchell. "Predicting controller capacity in supervisory control of multiple UAVs". In: *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 38.2 (2008), pp. 451–460.
- [40] Mary L Cummings et al. *Automation architecture for single operator, multiple UAV command and control*. Tech. rep. DTIC Document, 2007.
- [41] Mary L Cummings et al. "Boredom and distraction in multiple unmanned vehicle supervisory control". In: *Interacting with Computers* 25.1 (2013), pp. 34–47.
- [42] David L Davies and Donald W Bouldin. "A cluster separation measure". In: *IEEE transactions on pattern analysis and machine intelligence* 2 (1979), pp. 224–227.

- [43] Geert De Cubber et al. "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)*. IEEE. 2013, pp. 1–4.
- [44] Jan Dentler et al. "A real-time model predictive position control with collision avoidance for commercial low-cost quadrotors". In: *Control Applications (CCA), 2016 IEEE Conference on*. IEEE. 2016, pp. 519–525.
- [45] S Hamid Dezfoulan, Dan Wu, and Imran Shafiq Ahmad. "A generalized neural network approach to mobile robot navigation and obstacle avoidance". In: *Intelligent Autonomous Systems 12*. Springer, 2013, pp. 25–42.
- [46] Bin Di, Rui Zhou, and Haibin Duan. "Potential field based receding horizon motion planning for centrality-aware multiple UAV cooperative surveillance". In: *Aerospace Science and Technology* 46 (2015), pp. 386–397.
- [47] Davide Di Ruscio, Ivano Malavolta, and Patrizio Pelliccione. "A family of domain-specific languages for specifying civilian missions of multi-robot systems". In: *First Workshop on Model-Driven Robot Software Engineering-MORSE*. 2014.
- [48] Félix Díez Arias, Juan Jesús Roldán Gómez, and Antonio Barrientos Cruz. "Modelado y Análisis de Misiones Multi-Robot mediante Minería de Procesos". In: (2016).
- [49] Remco M Dijkman, Marlon Dumas, and Chun Ouyang. "Semantics and analysis of business process models in BPMN". In: *Information and Software technology* 50.12 (2008), pp. 1281–1294.
- [50] Stephen R Dixon, Christopher D Wickens, and Dervon Chang. "Mission control of multiple unmanned aerial vehicles: A workload analysis". In: *Human Factors: The Journal of the Human Factors and Ergonomics Society* 47.3 (2005), pp. 479–487.
- [51] Birsén Donmez, Carl Nehme, and Mary L Cummings. "Modeling workload impact in multiple unmanned vehicle supervisory control". In: *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 40.6 (2010), pp. 1180–1190.
- [52] Jill L Drury, Laurel Riek, and Nathan Rackliffe. "A decomposition of UAV-related situation awareness". In: *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*. ACM. 2006, pp. 88–94.
- [53] Jill L Drury, Jean Scholtz, and Holly A Yanco. "Awareness in human-robot interactions". In: *Systems, Man and Cybernetics, 2003. IEEE International Conference on*. Vol. 1. IEEE. 2003, pp. 912–918.
- [54] Jill L Drury et al. *Comparing situation awareness for two unmanned aerial vehicle human interface approaches*. Tech. rep. DTIC Document, 2006.
- [55] Gregory Dudek et al. "A taxonomy for multi-agent robotics". In: *Autonomous Robots* 3.4 (1996), pp. 375–397.

- [56] Michael B Eisen et al. "Cluster analysis and display of genome-wide expression patterns". In: *Proceedings of the National Academy of Sciences* 95.25 (1998), pp. 14863–14868.
- [57] Robert J Elliott, Lakhdar Aggoun, and John B Moore. *Hidden Markov models: estimation and control*. Vol. 29. Springer Science & Business Media, 2008.
- [58] Mica R Endsley. "Design and evaluation for situation awareness enhancement". In: *Proceedings of the human factors and ergonomics society annual meeting*. Vol. 32. 2. SAGE Publications. 1988, pp. 97–101.
- [59] Mica R Endsley. "Level of automation effects on performance, situation awareness and workload in a dynamic control task". In: *Ergonomics* 42.3 (1999), pp. 462–492.
- [60] Mica R Endsley. "Situation awareness global assessment technique (SAGAT)". In: *Aerospace and Electronics Conference, 1988. NAECON 1988., Proceedings of the IEEE 1988 National*. IEEE. 1988, pp. 789–795.
- [61] Mica R Endsley and Daniel J Garland. *Situation awareness analysis and measurement*. CRC Press, 2000.
- [62] Adil Fahad et al. "A survey of clustering algorithms for big data: Taxonomy and empirical analysis". In: *IEEE transactions on emerging topics in computing* 2.3 (2014), pp. 267–279.
- [63] Alessandro Farinelli, Luca Iocchi, and Daniele Nardi. "Multirobot systems: a classification focused on coordination". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 34.5 (2004), pp. 2015–2028.
- [64] Krzysztof Foit. "Mixed reality as a tool supporting programming of the robot". In: *Advanced Materials Research*. Vol. 1036. Trans Tech Publ. 2014, pp. 737–742.
- [65] Florian Frische and Andreas Lüdtkke. "Sa-tracer: A tool for assessment of uav swarm operator sa during mission execution". In: *Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), 2013 IEEE International Multi-Disciplinary Conference on*. IEEE. 2013, pp. 203–211.
- [66] Christian Fuchs et al. "An ecological approach to the supervisory control of UAV swarms". In: *International Journal of Micro Air Vehicles* 6.4 (2014), pp. 211–229.
- [67] Enric Galceran and Marc Carreras. "A survey on coverage path planning for robotics". In: *Robotics and Autonomous Systems* 61.12 (2013), pp. 1258–1276.
- [68] Kala Garapati et al. "A Game of Drones: Game Theoretic Approaches for Multi-robot Task Allocation in Security Missions". In: *Iberian Robotics conference*. Springer. 2017, pp. 855–866.
- [69] Mario Garzón et al. "A Multirobot System for Distributed Area Coverage and Signal Searching in Large Outdoor Scenarios". In: *Journal of Field Robotics* (2015).
- [70] Mario Garzón et al. "An aerial-ground robotic system for navigation and obstacle mapping in large outdoor areas". In: *Sensors* 13.1 (2013), pp. 1247–1267.

- [71] Mario Garzón et al. "Using ROS in Multi-robot Systems: Experiences and Lessons Learned from Real-World Field Tests". In: *Robot Operating System (ROS)*. Springer, 2017, pp. 449–483.
- [72] Avinash Gautam and Sudeept Mohan. "A review of research in multi-robot systems". In: *Industrial and Information Systems (ICIIS), 2012 7th IEEE International Conference on*. IEEE. 2012, pp. 1–5.
- [73] Nigel Gilbert. *Agent-based models*. 153. Sage, 2008.
- [74] Chad Goerzen, Zhaodan Kong, and Bernard Mettler. "A survey of motion planning algorithms from the perspective of autonomous UAV guidance". In: *Selected papers from the 2nd International Symposium on UAVs, Reno, Nevada, USA June 8–10, 2009*. Springer. 2009, pp. 65–100.
- [75] Michael A Goodrich and Alan C Schultz. "Human-robot interaction: a survey". In: *Foundations and trends in human-computer interaction* 1.3 (2007), pp. 203–275.
- [76] David Gossow et al. "Interactive markers: 3-d user interfaces for ros applications [ros topics]". In: *IEEE Robotics & Automation Magazine* 18.4 (2011), pp. 14–15.
- [77] Joachim de Greeff et al. "Human-robot teamwork in USAR environments: the TRADR project". In: *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction Extended Abstracts*. ACM. 2015, pp. 151–152.
- [78] Jason Gregory et al. "Application of multi-robot systems to disaster-relief scenarios with limited communication". In: *Field and Service Robotics*. Springer. 2016, pp. 639–653.
- [79] Phillip M Grice and Charles C Kemp. "Assistive mobile manipulation: Designing for operators with motor impairments". In: *RSS 2016 Workshop on Socially and Physically Assistive Robotics for Humanity*. 2016.
- [80] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. "Improved techniques for grid mapping with rao-blackwellized particle filters". In: *IEEE transactions on Robotics* 23.1 (2007), pp. 34–46.
- [81] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. "Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling". In: *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. IEEE. 2005, pp. 2432–2437.
- [82] Christian W Günther and Anne Rozinat. "Disco: Discover Your Processes." In: *BPM (Demos)* 940 (2012), pp. 40–44.
- [83] Roberto Guzman et al. "Robotnik—Professional service robotics applications with ROS". In: *Robot Operating System (ROS)*. Springer, 2016, pp. 253–288.
- [84] Roberto Guzmán et al. "Robotnik—Professional Service Robotics Applications with ROS (2)". In: *Robot Operating System (ROS)*. Springer, 2017, pp. 419–447.
- [85] Ellen C Haas et al. "Multimodal controls for soldier/swarm interaction". In: *RO-MAN, 2011 IEEE*. IEEE. 2011, pp. 223–228.

- [86] Pranab Haldar et al. "Cluster analysis and clinical asthma phenotypes". In: *American journal of respiratory and critical care medicine* 178.3 (2008), pp. 218–224.
- [87] Osama Hamzeh and Ashraf Elnagar. "Localization and Navigation of Autonomous Indoor Mobile Robots". In: ().
- [88] Markus Hannebauer and Jan Wendler. *Balancing reactivity and social deliberation in multi-agent systems: from RoboCup to real-world applications*. Vol. 2103. Springer Science & Business Media, 2001.
- [89] Jeffrey T Hansberger. *Development of the Next Generation of Adaptive Interfaces*. Tech. rep. DTIC Document, 2015.
- [90] Sandra G Hart. "NASA-task load index (NASA-TLX); 20 years later". In: *Proceedings of the human factors and ergonomics society annual meeting*. Vol. 50. 9. Sage Publications Sage CA: Los Angeles, CA. 2006, pp. 904–908.
- [91] Sandra G Hart and Lowell E Staveland. "Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research". In: *Advances in psychology* 52 (1988), pp. 139–183.
- [92] Stephen Hart, Paul Dinh, and Kimberly Hambuchen. "The affordance template ROS package for robot task programming". In: *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE. 2015, pp. 6227–6234.
- [93] John A Hartigan and Manchek A Wong. "Algorithm AS 136: A k-means clustering algorithm". In: *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 28.1 (1979), pp. 100–108.
- [94] Alan Hobbs and Stanley R Herwitz. "Human factors in the maintenance of unmanned aircraft". In: *Small* 15.100 (2014), 100s.
- [95] Amy Hocraffer and Chang S Nam. "A meta-analysis of human-system interfaces in unmanned aerial vehicle (UAV) swarm management". In: *Applied Ergonomics* 58 (2017), pp. 66–80.
- [96] Alexander Hong. "Human-Robot Interactions for Single Robots and Multi-Robot Teams". PhD thesis. 2016.
- [97] Ming Hou et al. "Optimizing operator-agent interaction in intelligent adaptive interface design: A conceptual framework". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 41.2 (2011), pp. 161–178.
- [98] Félix Ingrand and Malik Ghallab. "Deliberation for autonomous robots: A survey". In: *Artificial Intelligence* (2014).
- [99] Peter Jackson. "Introduction to expert systems". In: (1986).
- [100] Bob Jacobs et al. "Application of Intelligent Aiding to Enable Single Operator Multiple UAV Supervisory Control". In: *Association for the Advancement of Artificial Intelligence, Palo Alto, CA* (2010).
- [101] Nikita Jain and Vishal Srivastava. "Data mining techniques: a survey paper". In: *IJRET: International Journal of Research in Engineering and Technology* 2.11 (2013), pp. 2319–1163.
- [102] Adiyabaatar Janchiv et al. "Complete coverage path planning for multi-robots based on". In: *Control, Automation and Systems (ICCAS), 2011 11th International Conference on*. IEEE. 2011, pp. 824–827.

- [103] Mieke Jans et al. "A business process mining application for internal transaction fraud mitigation". In: *Expert Systems with Applications* 38.10 (2011), pp. 13351–13359.
- [104] Phillip Jasper, Ciara Sibley, and Joseph Coyne. "Using Heart Rate Variability to Assess Operator Mental Workload in a Command and Control Simulation of Multiple Unmanned Aerial Vehicles". In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. Vol. 60. 1. SAGE Publications Sage CA: Los Angeles, CA. 2016, pp. 1125–1129.
- [105] Xiao Jia and Max Q-H Meng. "A survey and analysis of task allocation algorithms in multi-robot systems". In: *Robotics and biomimetics (ROBIO), 2013 IEEE international conference on*. IEEE. 2013, pp. 2280–2285.
- [106] Yichuan Jiang. "A survey of task allocation and load balancing in distributed systems". In: *IEEE Transactions on Parallel and Distributed Systems* 27.2 (2016), pp. 585–599.
- [107] Yichuan Jiang. "Concurrent collective strategy diffusion of multi-agents: the spatial model and case study". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 39.4 (2009), pp. 448–458.
- [108] Athanasios Ch Kapoutsis et al. "Real-time adaptive multi-robot exploration with application to underwater map construction". In: *Autonomous Robots* (2015), pp. 1–29.
- [109] Leonard Kaufman and Peter J Rousseeuw. "Partitioning around medoids (program pam)". In: *Finding groups in data: an introduction to cluster analysis* (1990), pp. 68–125.
- [110] S Kavitha, S Veena, and R Kumaraswamy. "Development of automatic speech recognition system for voice activated Ground Control system". In: *Trends in Automation, Communications and Computing Technology (I-TACT-15), 2015 International Conference on*. Vol. 1. IEEE. 2015, pp. 1–5.
- [111] Amirreza M Khaleghi et al. "A comparative study of control architectures in UAV/UGV-based surveillance system". In: *IIE Annual Conference. Proceedings*. Institute of Industrial Engineers-Publisher. 2014, p. 3455.
- [112] Alaa Khamis, Ahmed Hussein, and Ahmed Elmogy. "Multi-robot Task Allocation: A Review of the State-of-the-Art". In: *Cooperative Robots and Sensor Networks 2015*. Springer, 2015, pp. 31–51.
- [113] Elsa A Kirchner et al. "An Intelligent Man-Machine Interface—Multi-Robot Control Adapted for Task Engagement Based on Single-Trial Detectability of P300". In: *Frontiers in Human Neuroscience* 10 (2016).
- [114] Teuvo Kohonen. "The self-organizing map". In: *Neurocomputing* 21.1 (1998), pp. 1–6.
- [115] Andreas Kolling, Steven Nunnally, and Michael Lewis. "Towards human control of robot swarms". In: *Proceedings of the seventh annual ACM/IEEE international conference on human-robot interaction*. ACM. 2012, pp. 89–96.

- [116] G Ayorkor Korsah, Anthony Stentz, and M Bernardine Dias. "A comprehensive taxonomy for multi-robot task allocation". In: *The International Journal of Robotics Research* 32.12 (2013), pp. 1495–1512.
- [117] Thomas Koshy. *Discrete mathematics with applications*. Academic Press, 2004.
- [118] Mangal Kothari, Ian Postlethwaite, and Da-Wei Gu. "Multi-UAV path planning in obstacle rich environments using rapidly-exploring random trees". In: *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*. IEEE. 2009, pp. 3069–3074.
- [119] Tomáš Krajník et al. "AR-drone as a platform for robotic research and education". In: *International Conference on Research and Education in Robotics*. Springer. 2011, pp. 172–186.
- [120] Geert-Jan M Kruijff et al. "Experience in system design for human-robot teaming in urban search and rescue". In: *Field and Service Robotics*. Springer. 2014, pp. 111–125.
- [121] Ivana Kruijff-Korbayová et al. "TRADR project: Long-term human-robot teaming for robot assisted disaster response". In: *KI-Künstliche Intelligenz* 29.2 (2015), pp. 193–201.
- [122] James J Kuffner and Steven M LaValle. "RRT-connect: An efficient approach to single-query path planning". In: *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*. Vol. 2. IEEE. 2000, pp. 995–1001.
- [123] Hindra Kurniawan, Alexandr V Maslov, and Mykola Pechenizkiy. "Stress detection from speech and galvanic skin response signals". In: *Computer-Based Medical Systems (CBMS), 2013 IEEE 26th International Symposium on*. IEEE. 2013, pp. 209–214.
- [124] Rafael Guedes Lang, Ivan Nunes da Silva, and Roseli AF Romero. "Development of distributed control architecture for multi-robot systems". In: *Robotics: SBR-LARS Robotics Symposium and Robocontrol (SBR LARS Robocontrol), 2014 Joint Conference on*. IEEE. 2014, pp. 163–168.
- [125] Benoit Larochelle et al. *Establishing human situation awareness using a multi-modal operator control unit in an urban search & rescue human-robot team*. IEEE, 2011.
- [126] Corinna E Lathan and Michael Tracey. "The effects of operator spatial perception and sensory feedback on human-robot teleoperation performance". In: *Presence: Teleoperators and virtual environments* 11.4 (2002), pp. 368–377.
- [127] Jin-Shyan Lee. "A Petri net design of command filters for semiautonomous mobile sensor networks". In: *IEEE Transactions on Industrial Electronics* 55.4 (2008), pp. 1835–1841.
- [128] Sander JJ Leemans, Dirk Fahland, and Wil MP van der Aalst. "Discovering block-structured process models from event logs-a constructive approach". In: *International Conference on Applications and Theory of Petri Nets and Concurrency*. Springer. 2013, pp. 311–329.

- [129] Sander JJ Leemans, Dirk Fahland, and Wil MP van der Aalst. "Process and Deviation Exploration with Inductive Visual Miner." In: *BPM (Demos)*. 2014, p. 46.
- [130] Charles Lesire et al. "A distributed architecture for supervision of autonomous multi-robot missions". In: *Autonomous Robots* 40.7 (2016), pp. 1343–1362.
- [131] Nico Li et al. "Flying Frustum: A Spatial Interface for Enhancing Human-UAV Awareness". In: *Proceedings of the 3rd International Conference on Human-Agent Interaction*. ACM. 2015, pp. 27–31.
- [132] Michael Lindemuth et al. "Sea robot-assisted inspection". In: *IEEE robotics & automation magazine* 18.2 (2011), pp. 96–107.
- [133] Hong Liu, Weiwei Wan, and Hongbin Zha. "A dynamic subgoal path planner for unpredictable environments". In: *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE. 2010, pp. 994–1001.
- [134] Christian Sebastian Loh, Yanyan Sheng, and Dirk Ifenthaler. "Serious Games Analytics: Methodologies for Performance Measurement". In: *Assessment, and Improvement*. Springer (2015).
- [135] Niels Lohmann, Eric Verbeek, and Remco M Dijkman. "Petri Net Transformations for Business Processes-A Survey." In: *Trans. Petri Nets and Other Models of Concurrency* 2 (2009), pp. 46–63.
- [136] Susan Lomax and Sunil Vadera. "A survey of cost-sensitive decision tree induction algorithms". In: *ACM Computing Surveys (CSUR)* 45.2 (2013), p. 16.
- [137] Robert J Lysaght et al. *Operator workload: Comprehensive review and evaluation of operator workload methodologies*. Tech. rep. DTIC Document, 1989.
- [138] Thi Thoa Mac et al. "Heuristic approaches in robot path planning: A survey". In: *Robotics and Autonomous Systems* 86 (2016), pp. 13–28.
- [139] Ana Rocío Cárdenas Maita et al. "A systematic mapping study of process mining". In: *Enterprise Information Systems* (2017), pp. 1–45.
- [140] RS Mans et al. "Application of process mining in healthcare—a case study in a dutch hospital". In: *International Joint Conference on Biomedical Engineering Systems and Technologies*. Springer. 2008, pp. 425–438.
- [141] Tomás Mantecón et al. "New generation of human machine interfaces for controlling UAV through depth-based gesture recognition". In: *SPIE Defense+ Security*. International Society for Optics and Photonics. 2014, pp. 90840C–90840C.
- [142] Eitan Marder-Eppstein et al. "The office marathon: Robust navigation in an indoor office environment". In: *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE. 2010, pp. 300–307.
- [143] Alessandro Marino et al. "A decentralized architecture for multi-robot systems based on the null-space-behavioral control with application to multi-robot border patrolling". In: *Journal of Intelligent & Robotic Systems* 71.3-4 (2013), pp. 423–444.

- [144] Vicente Matellán and Daniel Borrajo. "Abc2 an agenda based multi-agent model for robots control and cooperation". In: *Journal of Intelligent & Robotic Systems* 32.1 (2001), pp. 93–114.
- [145] Jason S McCarley and Christopher D Wickens. *Human factors implications of UAVs in the national airspace*. University of Illinois at Urbana-Champaign, Aviation Human Factors Division, 2005.
- [146] Daniel McDuff, Sarah Gontarek, and Rosalind Picard. "Remote measurement of cognitive stress via heart rate variability". In: *Engineering in Medicine and Biology Society (EMBC), 2014 36th Annual International Conference of the IEEE*. IEEE. 2014, pp. 2957–2960.
- [147] Justin Menda et al. "Optical brain imaging to enhance UAV operator training, evaluation, and interface development". In: *Journal of intelligent & robotic systems* 61.1 (2011), pp. 423–443.
- [148] "Mobile robot virtual training system for robot operators". In:
- [149] Prases K Mohanty and Dayal R Parhi. "Controlling the motion of an autonomous mobile robot using various techniques: a review". In: *Journal of Advance Mechanical Engineering* 1.1 (2013), pp. 24–39.
- [150] Valiallah Mani Monajjemi et al. "Integrating multi-modal interfaces to command UAVs". In: *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction*. ACM. 2014, pp. 106–106.
- [151] Neville Moray. *Mental workload: Its theory and measurement*. Vol. 8. Springer Science & Business Media, 2013.
- [152] Michael Mortimer, Ben Horan, and Mehdi Seyedmahmoudian. "Building a Relationship between Robot Characteristics and Teleoperation User Interfaces". In: *Sensors* 17.3 (2017), p. 587.
- [153] Alejandro R Mosteo and Luis Montano. "A survey of multi-robot task allocation". In: *Instituto de Investigación en Ingeniería de Aragón, University of Zaragoza, Zaragoza, Spain, Technical Report No. AMI-009-10-TEC* (2010).
- [154] Tadao Murata. "Petri nets: Properties, analysis and applications". In: *Proceedings of the IEEE* 77.4 (1989), pp. 541–580.
- [155] Robin R Murphy. "Human-robot interaction in rescue robotics". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 34.2 (2004), pp. 138–153.
- [156] Jawad Nagi et al. "Human control of UAVs using face pose estimates and hand gestures". In: *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction*. ACM. 2014, pp. 252–253.
- [157] Chang S Nam et al. "Evaluation of human-agent user interfaces in multi-agent systems". In: *International Journal of Industrial Ergonomics* 39.1 (2009), pp. 192–201.
- [158] Francisco Navarro et al. "Integrating 3D Reconstruction and Virtual Reality: A New Approach for Immersive Teleoperation". In: *Iberian Robotics conference*. Springer. 2017, pp. 606–616.
- [159] Thomas Nestmeyer et al. "Decentralized simultaneous multi-target exploration using a connected network of multiple robots". In: *Autonomous Robots* (2016), pp. 1–23.

- [160] Diederick C Niehorster, Li Li, and Markus Lappe. "The Accuracy and Precision of Position and Orientation Tracking in the HTC Vive Virtual Reality System for Scientific Research". In: *i-Perception* 8.3 (2017), p. 2041669517708205.
- [161] Bashra Kadhim Oleiwi et al. "Multi Objective Optimization of Trajectory Planning of Non-holonomic Mobile Robot in Dynamic Environment Using Enhanced GA by Fuzzy Motion Control and A". In: *International Conference on Neural Networks and Artificial Intelligence*. Springer. 2014, pp. 34–49.
- [162] Wesley A Olson and Nadine B Sarter. "Automation management strategies: Pilot preferences and operational experiences". In: *The International Journal of Aviation Psychology* 10.4 (2000), pp. 327–341.
- [163] Wesley A Olson and MG Wuennenberg. "Autonomy based human-vehicle interface standards for remotely operated aircraft". In: *Digital Avionics Systems, 2001. DASC. 20th Conference*. Vol. 2. IEEE. 2001, pp. 7D3–1.
- [164] M Padmanabhan and GR Suresh. "Coalition formation and Task Allocation of multiple autonomous robots". In: *Signal Processing, Communication and Networking (ICSCN), 2015 3rd International Conference on*. IEEE. 2015, pp. 1–5.
- [165] Raja Parasuraman, Thomas B Sheridan, and Christopher D Wickens. "A model for types and levels of human interaction with automation". In: *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on* 30.3 (2000), pp. 286–297.
- [166] António Pascoal, Carlos Silvestre, and Paulo Oliveira. "Vehicle and mission control of single and multiple autonomous marine robots". In: *IEE Control Engineering Series* 69 (2006), p. 353.
- [167] Marco Pedrotti et al. "Automatic stress classification with pupil diameter analysis". In: *International Journal of Human-Computer Interaction* 30.3 (2014), pp. 220–236.
- [168] Daniel Peña. *Análisis de datos multivariantes*. Vol. 24. McGraw-Hill Madrid, 2002.
- [169] Elena Peña Tapia et al. "Interfaz de Control para un Robot Manipulador mediante Realidad Virtual". In: (2017).
- [170] Sara Perez-Carabaza et al. "A Real World Multi-UAV Evolutionary Planner for Minimum Time Target Detection". In: *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference*. ACM. 2016, pp. 981–988.
- [171] Nicolas Poggi et al. "Business process mining from e-commerce web logs". In: *Business Process Management*. Springer, 2013, pp. 65–80.
- [172] Morgan Quigley et al. "ROS: an open-source Robot Operating System". In: *ICRA workshop on open source software*. Vol. 3. 3.2. Kobe. 2009, p. 5.
- [173] Cristian Ramirez-Atencia et al. "Performance Evaluation of Multi-UAV Cooperative Mission Planning Models". In: *Computational Collective Intelligence*. Springer, 2015, pp. 203–212.

- [174] Cristian Ramirez-Atencia et al. "Solving complex multi-UAV mission planning problems using multi-objective genetic algorithms". In: *Soft Computing* (2016), pp. 1–18.
- [175] William M Rand. "Objective criteria for the evaluation of clustering methods". In: *Journal of the American Statistical association* 66.336 (1971), pp. 846–850.
- [176] Carmine Tommaso Recchiuto, Antonio Sgorbissa, and Renato Zaccaria. "Visual feedback with multiple cameras in a UAVs Human–Swarm Interface". In: *Robotics and Autonomous Systems* 80 (2016), pp. 43–54.
- [177] Peng Ren et al. "Off-line and on-line stress detection through processing of the pupil diameter signal". In: *Annals of biomedical engineering* 42.1 (2014), pp. 162–176.
- [178] Igor Rodriguez et al. "Humanizing NAO robot teleoperation using ROS". In: *Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference on*. IEEE. 2014, pp. 179–186.
- [179] Víctor Rodríguez-Fernández, Antonio Gonzalez-Pardo, and David Camacho. "Modeling the behavior of unskilled users in a multi-uav simulation environment". In: *International Conference on Intelligent Data Engineering and Automated Learning*. Springer International Publishing. 2015, pp. 441–448.
- [180] Víctor Rodríguez-Fernández, Héctor D Menéndez, and David Camacho. "A study on performance metrics and clustering methods for analyzing behavior in UAV operations". In: *Journal of Intelligent & Fuzzy Systems* 32.2 (2017), pp. 1307–1319.
- [181] Víctor Rodríguez-Fernández, Héctor D Menéndez, and David Camacho. "Automatic profile generation for uav operators using a simulation-based training environment". In: *Progress in Artificial Intelligence* 5.1 (2016), pp. 37–46.
- [182] Victor Rodriguez-Fernandez, Cristian Ramirez-Atencia, and David Camacho. "A multi-uav mission planning videogame-based framework for player analysis". In: *Evolutionary Computation (CEC), 2015 IEEE Congress on*. IEEE. 2015, pp. 1490–1497.
- [183] Lior Rokach and Oded Maimon. *Data mining with decision trees: theory and applications*. World Scientific, 2008.
- [184] Juan Jesús Roldán, Jaime del Cerro, and Antonio Barrientos. "A proposal of methodology for multi-UAV mission modeling". In: *Control and Automation (MED), 2015 23th Mediterranean Conference on*. IEEE. 2015, pp. 1–7.
- [185] Juan Jesús Roldán, Jaime del Cerro, and Antonio Barrientos. "Minería de procesos aplicada a la monitorización de las misiones robóticas". In: (2015).
- [186] Juan Jesús Roldán, Jaime del Cerro, and Antonio Barrientos. "Using process mining to model multi-UAV missions through the experience". In: *IEEE Intelligent Systems* 32.4 (2017), pp. 40–47.

- [187] Juan Jesús Roldán et al. "A proposal of multi-UAV mission coordination and control architecture". In: *Robot 2015: Second Iberian robotics conference*. Springer. 2016, pp. 597–608.
- [188] Juan Jesús Roldán et al. "Analyzing and improving multi-robot missions by using process mining". In: *Autonomous Robots* (2017), pp. 1–19.
- [189] Juan Jesús Roldán et al. "Determining mission evolution through UAV telemetry by using decision trees". In: *Systems, Man, and Cybernetics (SMC), 2016 IEEE International Conference on*. IEEE. 2016, pp. 000188–000193.
- [190] Juan Jesús Roldán et al. "Heterogeneous Multi-Robot System for Mapping Environmental Variables of Greenhouses". In: *Sensors* 16.7 (2016), p. 1018.
- [191] Juan Jesús Roldán et al. "Mini-UAV based sensory system for measuring environmental variables in greenhouses". In: *Sensors* 15.2 (2015), pp. 3334–3350.
- [192] Juan Jesús Roldán et al. "Multi-robot interfaces and operator situational awareness: Study of the impact of immersion and prediction". In: *Sensors* 17.8 (2017), p. 1720.
- [193] Juan Jesús Roldán et al. "Multi-Robot Systems, Virtual Reality and ROS: Developing a new generation of operator interfaces". In: *Robot Operating System (ROS)*. Springer, 2018, TBD.
- [194] Juan Jesús Roldán et al. "Robots in Agriculture: State of Art and Practical Experiences". In: *Service Robots*. InTech, 2018.
- [195] Juan Jesús Roldán Gómez, Jaime del Cerro Giner, and Antonio Barrientos Cruz. "Multiple Robots, Single Operator: Considerations About Information And Commanding". In: (2016).
- [196] Juan Jesús Roldán Gómez et al. "Proyecto pric: protección robotizada de infraestructuras críticas". In: (2017).
- [197] Peter J Rousseeuw. "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis". In: *Journal of computational and applied mathematics* 20 (1987), pp. 53–65.
- [198] Anne Rozinat and Wil MP van der Aalst. "Decision mining in ProM". In: *Business process management* 4102 (2006), pp. 420–425.
- [199] Susana Ruano et al. "Augmented Reality Tool for the Situational Awareness Improvement of UAV Operators". In: *Sensors* 17.2 (2017), p. 297.
- [200] Heath A. Ruff, S. Narayanan, and Mark H. Draper. "Human Interaction with Levels of Automation and Decision-aid Fidelity in the Supervisory Control of Multiple Simulated Unmanned Air Vehicles". In: *Presence: Teleoper. Virtual Environ.* 11.4 (Aug. 2002), pp. 335–351. ISSN: 1054-7460. DOI: [10.1162/105474602760204264](https://doi.org/10.1162/105474602760204264). URL: <http://dx.doi.org/10.1162/105474602760204264>.

- [201] Heath A Ruff, Sundaram Narayanan, and Mark H Draper. "Human interaction with levels of automation and decision-aid fidelity in the supervisory control of multiple simulated unmanned air vehicles". In: *Presence: Teleoperators and virtual environments* 11.4 (2002), pp. 335–351.
- [202] JJ Ruiz et al. "A simulation framework to validate the use of head-mounted displays and tablets for information exchange with the UAV safety pilot". In: *Research, Education and Development of Unmanned Aerial Systems (RED-UAS), 2015 Workshop on*. IEEE. 2015, pp. 336–341.
- [203] JJ Ruiz et al. "Immersive displays for building spatial knowledge in multi-UAV operations". In: *Unmanned Aircraft Systems (ICUAS), 2015 International Conference on*. IEEE. 2015, pp. 1043–1048.
- [204] Alberto Ruiz-Larrea et al. "A UGV approach to measure the ground properties of greenhouses". In: *Robot 2015: Second Iberian Robotics Conference*. Springer. 2016, pp. 3–13.
- [205] André A Rupp and Shauna J Sweet. *Analysis of multivariate social science data*. 2011.
- [206] Paul Salmon et al. "Situation awareness measurement: A review of applicability for C4i environments". In: *Applied ergonomics* 37.2 (2006), pp. 225–238.
- [207] Carlos Sampedro et al. "A flexible and dynamic mission planning architecture for uav swarm coordination". In: *Unmanned Aircraft Systems (ICUAS), 2016 International Conference on*. IEEE. 2016, pp. 355–363.
- [208] Julian Sanchez. "Conceptual model of human-automation interaction". In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. Vol. 53. 18. SAGE Publications. 2009, pp. 1403–1407.
- [209] Jose Luis Sanchez-Lopez et al. "A system for the design and development of vision-based multi-robot quadrotor swarms". In: *Unmanned Aircraft Systems (ICUAS), 2014 International Conference on*. IEEE. 2014, pp. 640–648.
- [210] David Sanz Muñoz. "Cognitive risk perception system for obstacle avoidance in outdoor mUAV missions". PhD thesis. Industriales, 2015.
- [211] Stefano Scheggi et al. "Cooperative human-robot haptic navigation". In: *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE. 2014, pp. 2693–2698.
- [212] Eric Schneider et al. "Auction-based task allocation for multi-robot teams in dynamic environments". In: *Conference Towards Autonomous Robotic Systems*. Springer. 2015, pp. 246–257.
- [213] Jean Scholtz. "Theory and evaluation of human robot interactions". In: *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*. IEEE. 2003, 10–pp.

- [214] Jean Scholtz et al. "Evaluation of human-robot interaction awareness in search and rescue". In: *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*. Vol. 3. IEEE. 2004, pp. 2327–2332.
- [215] Matthias Schultze-Kraft et al. "Unsupervised classification of operator workload from brain signals". In: *Journal of neural engineering* 13.3 (2016), p. 036008.
- [216] Thomas B Sheridan. *Humans and automation: System design and research issues*. John Wiley & Sons, Inc., 2002.
- [217] Thomas B Sheridan and William L Verplank. *Human and computer control of undersea teleoperators*. Tech. rep. DTIC Document, 1978.
- [218] Masaru Shimizu and Tomoichi Takahashi. "Training platform for rescue robot operation and pair operations of multi-robots". In: *Advanced Robotics* 27.5 (2013), pp. 385–391.
- [219] Brian D Simpson, Robert S Bolia, and Mark H Draper. "Spatial audio display concepts supporting situation awareness for operators of unmanned aerial vehicles". In: *Human Performance, Situation Awareness, and Automation: Current Research and Trends HPSAA II, Volumes I and II 2* (2013), p. 61.
- [220] Ioan A Sucan and Sachin Chitta. "Moveit!" In: *Online at <http://moveit.ros.org>* (2013).
- [221] Ioan A. Şucan, Mark Moll, and Lydia E. Kavraki. "The Open Motion Planning Library". In: *IEEE Robotics & Automation Magazine* 19.4 (2012), pp. 72–82. DOI: [10.1109/MRA.2012.2205651](https://doi.org/10.1109/MRA.2012.2205651). URL: <http://ompl.kavrakilab.org>.
- [222] María Taboada Pérez, Juan Jesús Roldán Gómez, and Antonio Barrientos Cruz. "Interfaz Inteligente y Adaptativa para Personas con Discapacidad". In: (2016).
- [223] Florent Teichteil-Königsbuch and Patrick Fabiani. "A multi-thread decisional architecture for real-time planning under uncertainty". In: *3rd ICAPS'07 Workshop on Planning and Plan Execution for Real-World Systems*. 2007.
- [224] Susanne Thierfelder et al. "Robbie: A message-based robot architecture for autonomous mobile systems". In: *INFORMATIK-Informatik Schafft Communities* (2011).
- [225] ANA PASCUAL TOMÁS. "Desarrollo de una aplicación basada en cámaras 3D para la generación de movimientos en un robot humanoide". PhD thesis. 2016.
- [226] Nicolas A Tsokas and Kostas J Kyriakopoulos. "Multi-robot multiple hypothesis tracking for pedestrian tracking". In: *Autonomous Robots* 32.1 (2012), pp. 63–79.
- [227] Stephen Tully, George Kantor, and Howie Choset. "Leap-frog path design for multi-robot cooperative localization". In: *Field and service robotics*. Springer. 2010, pp. 307–317.
- [228] Patrick Ulam et al. *Integrated mission specification and task allocation for robot teams-Part 2: Testing and evaluation*. Tech. rep. GEORGIA INST OF TECH ATLANTA COLL OF COMPUTING, 2006.

- [229] João Valente et al. "An air-ground wireless sensor network for crop monitoring". In: *Sensors* 11.6 (2011), pp. 6088–6108.
- [230] Wil Van Der Aalst, Arya Adriansyah, and Boudewijn Van Dongen. "Causal nets: a modeling language tailored towards process discovery". In: *International Conference on Concurrency Theory*. Springer. 2011, pp. 28–42.
- [231] Boudewijn F Van Dongen, AK Alves De Medeiros, and Lijie Wen. "Process mining: Overview and outlook of petri net discovery algorithms". In: *Transactions on Petri Nets and Other Models of Concurrency II*. Springer, 2009, pp. 225–242.
- [232] TWJ Van Ruijven. "Serious games as experiments for emergency management research: a review". In: *ISCRAM 2011: Proceedings of the 8th International Conference on Information Systems for Crisis Response and Management, Lisbon, Portugal, 8-11 May 2011*. ISCRAM. 2011.
- [233] KKDKU Weerasinghe et al. "Mapping and path planning for long distance autonomous navigation using multisensory data". In: *Electrical Engineering Conference (EECon)*. IEEE. 2016, pp. 1–6.
- [234] Anton JMM Weijters and Wil MP Van der Aalst. "Rediscovering workflow models from event-based data using little thumb". In: *Integrated Computer-Aided Engineering* 10.2 (2003), pp. 151–162.
- [235] Jan Martijn EM Van der Werf et al. "Process discovery using integer linear programming". In: *International Conference on Applications and Theory of Petri Nets*. Springer. 2008, pp. 368–387.
- [236] Choon Yue Wong and Gerald Seet. "Workload, awareness and automation in multiple-robot supervision". In: *International Journal of Advanced Robotic Systems* 14.3 (2017), p. 1729881417710463.
- [237] Richard TA Wood, Mark D Griffiths, and Virginia Eatough. "Online data collection from video game players: Methodological issues". In: *CyberPsychology & Behavior* 7.5 (2004), pp. 511–518.
- [238] Zhi Yan, Nicolas Jouandeau, and Arab Ali Cherif. "A survey and analysis of multi-robot coordination". In: *International Journal of Advanced Robotic Systems* 10.12 (2013), p. 399.
- [239] Jie Yang, Yangsheng Xu, and Chiou S Chen. "Human action learning via hidden Markov model". In: *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 27.1 (1997), pp. 34–44.
- [240] Jun Yang et al. "Task allocation for wireless sensor network using modified binary particle swarm optimization". In: *IEEE Sensors Journal* 14.3 (2014), pp. 882–892.
- [241] Liang Yang et al. "Survey of Robot 3D Path Planning Algorithms". In: *Journal of Control Science and Engineering* 2016 (2016), p. 5.
- [242] X Jessie Yang, Christopher D Wickens, and Katja Hölttä-Otto. "How users adjust trust in automation: Contrast effect and hindsight bias". In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. Vol. 60. 1. SAGE Publications Sage CA: Los Angeles, CA. 2016, pp. 196–200.

- [243] Safdar Zaman, Wolfgang Slany, and Gerald Steinbauer. "ROS-based mapping, localization and autonomous navigation using a Pioneer 3-DX robot and their relevant issues". In: *Electronics, Communications and Photonics Conference (SIECPC), 2011 Saudi International*. IEEE. 2011, pp. 1–5.
- [244] Yong Zhang, Dun-wei Gong, and Jian-hua Zhang. "Robot path planning in uncertain environment using multi-objective particle swarm optimization". In: *Neurocomputing* 103 (2013), pp. 172–185.
- [245] Wei Zheng, Yuechao Wang, and Ning Xi. "Development of an internet-based multi-robot teleoperation system". In: *Mechatronics and Automation, Proceedings of the 2006 IEEE International Conference on*. IEEE. 2006, pp. 329–333.
- [246] Longjiang Zhou et al. "A robotics simulator platform for RADOE". In: *Control, Automation and Robotics (ICCAR), 2016 2nd International Conference on*. IEEE. 2016, pp. 44–48.
- [247] Qiuming Zhu. "Hidden Markov model for dynamic obstacle avoidance of mobile robot navigation". In: *IEEE Transactions on Robotics and Automation* 7.3 (1991), pp. 390–397.