Performance Guarantee for Autonomous Robotic Missions using Resource Management: The PANORAMA Approach

Philippe Lambert, Karen Godary-Dejean, Lionel Lapierre, Lotfi Jaïem, Didier Crestani

Journal of Intelligent & Robotic Systems (2024) 110:52
https://doi.org/10.1007/s10846-024-02058-7

REGULAR PAPER

Performance Guarantee for Autonomous Robotic Missions using Resource Management: The PANORAMA Approach

Philippe Lambert[1] · Karen Godary-Dejean[1] · Lionel Lapierre[1] · Lotfi Jaiem[2] · Didier Crestani[1]

Received: 9 August 2022 / Accepted: 11 January 2024

Abstract
This paper proposes the PANORAMA approach, which is designed to dynamically and autonomously manage the allocation of a robot's hardware and software resources during fully autonomous mission. This behavioral autonomy approach guarantees the satisfaction of the mission performance constraints. This article clarifies the concept of performance for autonomous robotic missions and details the different phases of the PANORAMA approach. Finally, it focuses on an experimental implementation on a patrolling mission example.

Keywords Autonomous mobile robotics · Robotic mission management · Performance guarantee · Resource allocation

1 Introduction

In [8], the authors claim that a cyber-physical vehicle system must achieve its "locomotion and task-level objectives while meeting safety and efficiency constraints", depending on the robot and the environmental uncertainties. They highlight some challenges:

- *Fault tolerance:* when considering autonomous missions, it is essential to deal with unforeseen events and address the robustness, reliability and dependability properties. Studies on this topics in mobile robotics focusing on fault detection [9–11], fault diagnosis [12, 13] or fault/error recovery [14–16]. Fault tolerance approaches in robotics involve also fault tolerant control approaches [17–19]. Dependability management concerns also the task level [20, 21]. Few studies present a global dependability approach from fault to mission level [3, 22].
- *Energy management:* this is a critical point for autonomous robots to guarantee the success of an autonomous mission. Neglected for a long time, this issue has recently gained increased attention from researchers. These last years many papers concern the power supply modelling [23, 24] and battery recharge. A review of power solutions for mobile robots is proposed in [25]. The issue of battery charge scheduling is addressed in [26, 27]. Many works concerns UAV (Unmanned Aerial Vehicle) and proposed solutions to increase the flying time using in-flight battery switching [28], staging energy sources [29] and autonomous charging [30]. The energy efficiency of the robots is mainly studied considering the path planning issue linking path length and energy consumption. The energy cost is studied in [31, 32] for an Ackermann platform or skid steer rovers [33, 34]. The impact of the soil type and relief is also considered for path planning in [34–37]. Most studies focus on locomotion energy. Few of them take into account explicitly the energy consumed by the sensors or by the algorithmic suite in used [38, 39].
- *Resource management:* robotic system has a set of hardware (actuators, sensors, processors) and software (algorithms) resources. The complex underlying issue is that during full autonomous mission the robot's control architecture must be able to efficiently manage these resources according to the mission objectives, the surrounding environment, and their energy impacts.

In the literature the term *Resource Allocation / Management* has different meanings depending on the application domain. In computer science it mainly corresponds to the development of algorithms managing the processor(s) load

✉ Philippe Lambert
philippe.lambert1@umontpellier.fr

Karen Godary-Dejean
firstname.name@umontpellier.fr

[1] LIRMM Univ Montpellier, CNRS, 161 rue Ada, Montpellier 34095, France

[2] IADYS, Roquefort-la-Bédoule 13830, France

Published online: 22 March 2024

to execute many tasks regarding some optimization criteria such as computing performance or energy consumption for embedded systems applications or cloud data centers [40–42]. In Industrial Engineering it can concern the allocation policy of human resources to business processes to optimize performance objectives (cost, duration, human usage, etc..) [43, 44] or the problem of the machine allocation policy in job-shops environment to reduce the processing time [45] or carbon emission [46]. In robotics it mainly deals with Multi Robot (heterogeneous or not) Task Allocation (MRTA). Here the issue is twofold to reach some performance objectives like time to reach the mission goal, minimizing the total traveled distance or the consumed energy [47]:

- On the one hand, it deals with *Task Planning - Task Decomposition* which answers the question *"What has to be done"*.
- On the second hand, an answer to the question *"Who does the task ?"* must be found.

In this paper we consider a unique robot having redundant hardware and software functionalities. Surprisingly, to the best of our knowledge, there have been very few studies that tackle the challenge of dynamically and autonomously selecting a robot's hardware and software resources throughout a mission while simultaneously achieving specific mission performance goals. It seems also clear that the selection of a resource and its corresponding configuration has an impact on the robot performance. So the resources tuning is not neutral and can have a real impact on the mission performance. Some questions still remain open to guaranty that a robot successfully fulfill a mission while satisfying different performance objectives.

The PANORAMA approach developed in this paper strives to consider the resource allocation, energy management and fault tolerance challenges. The main question addressed is as follows: **Considering a robotic mission plan and some performance goals, as well as a set of available robot's hardware and software resources, what resources should be allocated throughout the mission to achieve these goals?** The proposed resource allocation mechanism effectively verifies that these goals will be met throughout the mission, regardless of the observed behavioral drift or of some unforeseen events.

This paper is organized as follows. Firstly, the performance concept is presented and detailed for the robotics domain. The next part outlines an experimental patrolling mission in an indoor dynamic environment. This mission illustrates the different aspects of the PANORAMA approach, which in turn is explained and described from Sections 4 to 7. Experimental results are presented that demonstrate the adaptability and efficiency of the proposed methodology. Before concluding the PANORAMA implementation process is summarized. Finally, the conclusion points out some limitations of our approach, but also proposes several potential tracks to extend this work and some potential applications.

2 Performance: A Key Concept Towards Autonomy

2.1 Autonomy and Performance

Autonomy can be viewed as the capacity of a system to decide and act without any assistance [5, 48]. It clearly concerns decision-making (decision autonomy) and action (behavioral autonomy). Decision autonomy relates to *"Decide what to do"*. The user is supposed to prescribe the high-level goals and constraints of the initial mission. These goals are decomposed into sub-goals and finally into a set of sequential and/or concurrent robotic tasks that must be carried out. Behavioral autonomy relates to *"How to perform what has been decided ?"*. That involves making decision when choices must be made among several alternatives to carry out a given task, depending on the available resources and on the mission context.

Decision autonomy involves three main elements: a goal to be achieved, a finite set of possible alternatives and a criterion to decide which alternative to choose.We think that a multi-criteria viewpoint is required for robotic decisions, and that performance is a relevant concept for decision autonomy (Fig. 1).

2.2 Performance: A Multi-Form Concept

Performance is a concept that is widely used in the robotics domain, but it is not clearly defined. It is a generic word in which everyone places the right meaning according to the studied work and context. We look at enterprise and business

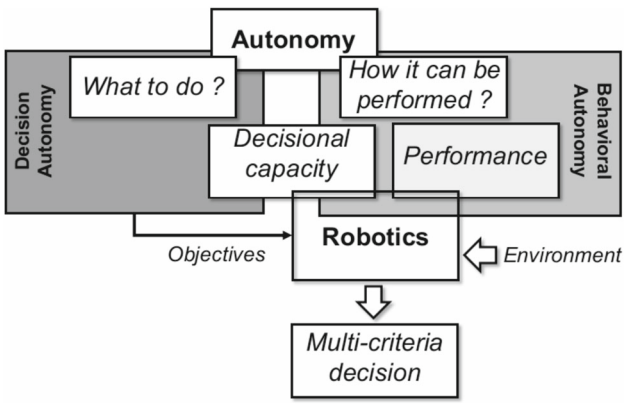

Fig. 1 Autonomy for robotics

management fields where this concept has been largely used, as a means to come up with a performance definition. The following definition extrapolated from [49] could be effectively applied to robotics.
The performance:

- Relates to an objective.
- Is a multi-dimensional concept (if multiple goals are considered).
- Is the result of actions.
- Depends on the involved resources.

It is important to distinguish between measurement and evaluation when considering performance. Measurement involves the determination of a value, while an evaluation gives an interpretation (higher, good, etc.) of the measurement with regards to a reference. The **performance indicator** is crucial to estimate the performance of a system from a given viewpoint relatively to a given goal. It corresponds to a measurement of the efficiency of a system. In a system, **performance inducers** are parameters that influence the performance indicators. If they are controllable, they can be tuned to influence and manage the system performance.

3 Performance and Robotics

In robotics, performance is an important, yet poorly defined, issue as shown by the creation in 2009 of the Technical Committee on Performance Evaluation and Benchmarking of Robotic and Automation Systems (TC-PEBRAS) [50]. Papers presented at major robotics conferences and workshops (ICRA, IROS...) regularly consider the concept of performance. Robotics competitions have proposed to measure task performance in benchmark environments (RobotCup@Home, Robocup Rescue, World Robot Summit, DARPA Challenge, etc.) and many papers in the robotics literature propose benchmarking approaches like [51] for planetary rover mission, [52] for object perception, [53] for SLAM, [54] for planning approaches or [55] for reinforcement learning. This highlights the fact that there is a real need to quantify robot performance via accepted metric and evaluation methodologies to ensure the reliability of robotic systems.

3.1 Performance and Mobile Robotics

The concept of performance is common in industrial robotics where standardization instances like [56] or [57] define many performance criteria particularly for safety standards [58]. But industrial robotics has specific environmental features (as static environment, accurate localization or unlimited energy) which are not all relevant for mobile robotics.

Most papers on performance in mobile robotics focus on the evaluation of the efficiency of a single robotic task (vision [59], obstacle avoidance [60], localization using SLAM [61] or not [62], etc.), or using specific task oriented performance criteria as for the coverage path planning issue [63, 64]. In [65], even though the authors identify several performance metrics (safety-oriented, trajectory and mission metrics), they also highlight that performance metrics are usually neglected or limited to few ones (mission duration [66, 67] or path length/energy [33]).

This demonstrates that due to the variability in mobile robot tasks, numerous performance criteria are proposed for mobile robotic missions, but in a scattered way. There is a lack of established performance criteria and classifications for mobile robotics, especially at the mission level. Moreover, the energy aspects are often (and surprisingly) neglected even though they are essential for real autonomous robotic missions.

Also, the performance evaluation timeline with regards to specific missions has not been clearly identified. Do the performance goals have to be continuously fulfilled over the mission or only at the end? The performance analysis is mainly done *a posteriori*, while we consider that it is essential to have predictive and accurate performance estimators to be able to predict the mission feasibility before launch, as well as monitoring the performance behavior during the mission. Guaranteeing that a robot will be able to complete its mission would be a pivotal issue for the future development of autonomous systems. Of course a total guaranty in all cases is an utopia, however it is important to state if the considered mission can be launched with a good chance of success. It is also important to be able to decide, during the mission, if despite some adverse events, the mission can yet succeed.

However, performance guarantee at the mission level has seldom been considered for mobile robotics. An interesting study addressing this question was carried out in [68] for critical robotic operations through formal methods. Before the mission, the authors conduct a formal analysis of the robotic mission regarding the performance properties (safety, liveness, stochastic mission efficiency, etc.), considering system's models, the mission scenario and an uncertain environment as inputs of a verification module. More recent papers consider this issue from a more limited perspective, as seen in [69] for the construction of receding search-paths, multi-agent trajectory planning [70], and energy-efficient path planning with guarantees on completeness [32].

As in these last works, we think that the robot's resources, the mission and performance goals description, as well as the environmental context, are the three main bases of the performance evaluation for autonomy. To our knowledge, despite the importance of performance at the mission level, no solutions were proposed to aim to guarantee the success

of a mission while meeting the performance goals during a real mission. The PANORAMA approach is meant to bridge this gap.

3.2 Performance and Resource Allocation in Mobile Robotics

In mobile robotics, the issue of resource allocation mainly concerns the problem of task allocation of cooperative and coordinated multi-robots teams supporting homogeneous or heterogeneous capabilities. Many surveys can be found in the literature [47, 71] for the Multi-Robot Task Allocation. The objective is then to take advantage of the robot's capabilities to optimize the use of the available resources according to one or several mission criteria (objectives) like minimizing the exploration time or maximizing the wireless coverage. Multi-robots allocation provides an efficient way to deal with self-reconfiguration, fault-tolerance and robustness.

The resource allocation is an optimal decision problem which is known to be NP-hard. So, it is impossible to find quick and scalable algorithms that answer the optimality issue. Three dimensions must be considered: the inputs of the problem, the coordination methodology and the task allocation strategy.

Inputs

A resource allocation problem considers different inputs:

- Task modeling: relationships between tasks and robots.
- Constraints supported by the tasks (partial order, time window, coupling or incompatibility, environmental, robots, concurrency).
- Utility functions used to estimate the task/robot(s) efficiency (cost, fitness, reward, etc.).
- Optimization objectives defined to determine the best resources allocation.

For the problem tackled in this paper the following links can be identified:

- Task modeling: relationships between tasks and robot's hardware and software (algorithms) resources.
- Constraints supported by the tasks: mission plan and environment are supposed to be known. However, the environment, the functioning conditions of the robot's resources, the resources compatibility, and the performance viewpoints impose many constraints that must be considered and verified.
- Utility functions used to estimate the task/robot(s) efficiency. Evidently the relationships between the selection and the configuration of the robot's resources and their impact on the robot's performance must be modeled.
- Optimization objectives defined to determine the best robot's resource allocation. As we will see, depending on the considered performance class that corresponds to the definition of continuous or end-time optimization objectives integrating the Mission view-point.

Coordination

In multi-robot problems, the coordination methodology is an important issue because the robots must share information and take common decisions. It is basically based on data sharing and networks issues, with static/dynamic and centralized/decentralized problems. In our context we are interested in embedded resource allocation on a unique robot, which make this subject not extremely relevant. It is clear that the coordination methodology is centralized since the robot's control architecture has to manage the embedded resource allocation. All the mission and robot's information are known and shared to take allocation decisions. The decision process can be considered both statically and dynamically. Statically because the resources allocation is initially considered offline before engaging a mission. Dynamically because the current resources selection can be modified online in case of resources dysfunctions or performance drift. These resources switching must be realized efficiently to be compatible with real-time constraints.

Allocation Strategy

From the inputs and according to the coordination methodology, several strategies classes are proposed in robotics to solve the resource allocation problem while minimizing the cost function. The scalability of the chosen algorithmic approach is characterized by its computational complexity. *Multiple salesman* problems, market-based task allocation, behavior-based allocation or clustered task allocation strategies have been investigated. Most of these works propose only simulation results. Multiple separate objectives are rarely considered concurrently. Objectives are rather incorporated into a single utility value, which results in the loss of some information.

From this analysis we can conclude that, despite a correct formulation and identification of the dimensions of the problem of resource allocation, the usual robotic literature approaches seem to not tackle this issue with the correct angle to address embedded hardware and software resource selection adopting a global mission performance view-point.

The correct angle to address the embedded resource allocation seems more to be the control architecture view-point. From the early Sens-Plan-Act paradigm, and the architecture based on purely reactive behaviors proposed by Brooks

[72], the hybrid three levels layered architecture is largely adopted in robotics applications [73]. The higher level corresponds to a deliberate/decision layer handling high-level task planning, reasoning and interaction with human operators. The bottom layer corresponds the functional layer that interfaces the robot architecture with its material/physical components (actuators, sensors). It is classically composed of a set of basic robotic actions (motion, localization, etc..) modules. The middle layer interfaces the decisional one with the functional one. This execution layer receives the sequence of actions to be executed from the decisional layer and must select, synchronise, configure and manage dynamically the low level functions to perform these actions. Unfortunately, the process of resource allocation in the robotic control architecture is poorly detailed in the literature. Of course the material/software resource must be able to implement the proposed action, it must be available, shareable or not, and must have some functioning constraints. But when several resources can be envisaged how to select the best ones? Which relevant criteria can be used to decide between them?

To answer to these questions this paper proposes to consider the satisfaction of Mission performance objectives as the relevant resource selection criteria. So, in the sequel we consider that the resource allocation problem can be modeled as a Multi-Objective Optimization (MOO) problem as follow [74]:

$min\ \mathbf{F}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), .., f_k(\mathbf{x})]^T$

subject to $g_j(\mathbf{x}) \leq 0\ for\ j = 1, 2, ..., m$ and $h_l(\mathbf{x}) = 0\ for\ l = 1, 2, ..., e$

Where:

- k is the number of objective functions
- $\mathbf{x} \in E^n$ is a vector of decision variables. The decision variables can be continuous or discrete. They represent the independent variables we have control over.
- n is the number of decision variables creating a Decision/Design space D.
- $f_i(x)$ is an objective (criteria, payoff, cost, value) function. Objective functions are dependent variables that can be influenced through the decision variables.
- $\mathbf{F}(\mathbf{x})$ is a vector of objective functions. It is a mapping function between the Design space and the Objective space O.
- m is the number of inequality constraints.
- e is the number of equality constraints.

Now, the problem is to efficiently find the best solution minimizing a set of objectives. Ideally an optimal design $\hat{x}$ corresponds to an objective vector $\mathbf{F}^*$ minimizing each of the objective functions. Unfortunately this utopian solution can rarely be reached and the concept of Pareto front designating non-dominated solutions within the objective space is used. Multi-objective optimization often forces compromising between conflicting objectives. All the solutions belonging to the Pareto front can be considered as optimal but they correspond to a different compromise. Generally, to facilitate the selection of the best solution, weights are assigned to the different objective according to its importance.

In [75] the authors characterize the process of MOO definition and solving according to 3 main dimensions.

The first dimension concerns the problem statement. That involves the selection of the decision/design variables. Which ones, which ranges, are they discrete or continuous? That also concerns the selection of the objective functions. The constraints identification is also an important issue. Which ones, what type (equality, inequality) and ranges, can they be violated? The answer to all theses questions can have a great impact on the degree of freedom of the problem, on the dimensions of the decision space O and consequently on the complexity of the solution.

The second dimension focuses on how objective functions/constraints are calculated. Depending on the problem space, high-fidelity models can have an important impact on the computational time when a great amount of designs must be considered.

The last dimension concerns the selection and set-up of the optimization algorithm. Many algorithms have been proposed to address the MOO problem and find optimal solutions [74, 75] using stochastic or sweeping methods across the Design space. These classical methods that may have long convergence time and that can fail to find optimal solutions can be integrated in meta-heuristic algorithms to facilitate the searching process.

In this paper we detail the process of hardware and software resource allocation implemented in the execution layer of a robot's control architecture. It must select suitable resources before and during the mission to ensure its success with regard to several mission performance objectives. First, the robot's resources are presented as well as the patrolling mission. Then, the proposed PANORAMA approach is detailed and illustrated based on this experimental context. According to MOO principle we identify the decision variables, their ranges, the constraints imposed, the objective functions and their estimation mechanisms. We also present how a lexicographic-like algorithm can be used to find an acceptable initial resource allocation plan from an initial tasks breaking down of the mission. During the mission, online performance monitoring allows to increase the robustness of the system with regards to resources failure or to online performance drift. Before the conclusion we present the process to implement the PANORAMA approach on any given mission. To conclude, we highlight the main strengths and limitations of the proposed approach.

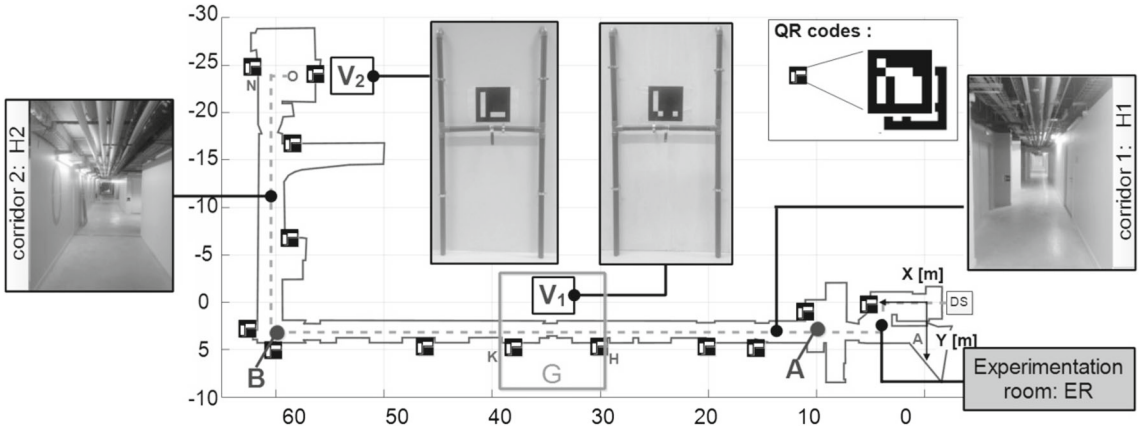

Fig. 2 The experimental environment of the patrolling mission

4 Patrolling Mission: Experimental Context

We consider a patrolling mission performed by a Pioneer P3DX mobile robot to determine the state (open or close) of two valves (V1 and V2) present in our laboratory corridors (Fig. 2).

4.1 The Robot

We use a Pioneer 3DX integrating 16 sonars and 10 bumpers. It weights $25kg$, including an embedded laptop and all the sensors.

Two top to tail lasers are used for obstacle avoidance, centering motion and robot localization. A Kinect camera and geo-referenced markers may also be used for localization. The robot communicates with an embedded laptop supporting a real-time control architecture implementing the PANORAMA approach. The robot and the laptop had their own batteries, which were respectively monitored by an embedded watt-meter. Controlled switchboards were used to independently switch on or off the Kinect and each laser.

Forward motion (MOV robotic task) could be carried out using path following or reactive centering techniques. It was combined with the Safe Maneuvering Zone (SMZ) obstacle avoidance method [76] which considers the robot's safety radius and kinematic behavior. The localization task (LOC) was implemented using either dead-reckoning navigation (odometers) or grid-based localization requiring the two lasers or QR-Code Navigation (QRCN) which combined the Kinect camera and odometers. Depending on the algorithms and sensors used, the following control methods are thus available (detailed in Table 1): 7 different moving control laws (MOV tasks), 3 localization methods (LOC tasks), a turn (TUR task) movement and 2 types of image analysis for valve state detection (DET task). The different control laws for a robotic task are called Task Implementation Options (TIOs).

4.2 The Patrolling Mission

The experimental environment is the ground floor of our laboratory, whose map is given in Fig. 2. On the right side, the experimentation room (ER) covers about $10m$ long and includes a docking station (DS). Then between the points A and B, there is a straight hall (H1) of approximately $50m$ long, ranging from $1.7m$ to $2.2m$ width. Within H1, there is a glazed area G, impeding the use of lasers. After B, another hall (H2) is about $30m$ long and more than $2.2m$ width. Two valves V1 and V2 are present in the middle of H1 and the end of H2. Many markers (QR codes) were spread across these halls to facilitate the robot localization.

The patrolling mission aims to control the state (open/close) of V1 and V2: from its docking station, the robot must go to inspect V1 and V2 before going back to its base. The total round-trip of the mission is about $187m$ long.

4.3 Experimental Constraints

Besides the performance constraints, robotic mission must deal with physical and functional constraints defining a set of limitations. For example, they correspond to the robot hardware (velocity, size, weight, sensors, actuators, etc.) and software (CPU frequency, number of cores, memory size, etc.) limitations. The environment also largely influences the robotic mission according to its dynamic, properties (opacity, transparency, color, etc.), nature (ground, water, air, flat surface, slope, etc.), complexity, presence of humans, etc. We call all these elements *experimental constraints*. These constraints are localized depending on their impact area. For example, the physical maximal velocity of the robot does not

Table 1 Allocation of robotic tasks and hardware components: definition of the TIOs (✗ means it is required)

Rob. task	SOFTWARE Information	HARDWARE Sonar	Laser	Kinect	Actuators	TIO
MOV	Path-Following with obstacle avoidance	✗			✗	M1
			✗		✗	M2
		✗	✗		✗	M3
			✗✗		✗	M4
		✗	✗✗		✗	M5
	Reactive centering with obstacle avoidance	✗	✗✗		✗	M6
			✗✗		✗	M7
LOC	Odometry					L1
	QR-Code (QRCN)			✗		L2
	Grid-Based Localization		✗✗			L3
TUR	Rotation - Valve Tracking				✗	T1
DET	Detection of the valve orientation			✗		D1
				✗	✗	D2

change all along the mission. While in specific areas with glass doors, sonar sensors must be used to detect it.

More precisely in our mission, the experimental constraints can be summarized as:

- **Maximal velocity :** the physical characteristics of the robot limits its velocity to $0.76m/s$.
- **Environmental constraints:** first, in glazed areas only moving methods using sonars must be considered ; second, the difference of widths of the two halls H1 and H2 influence the obstacle avoidance capacity.
- **Localization precision:** in areas where the image analysis of the valve must be done, QR-code navigation methods must be used, requiring the use of the robot's Kinect to ensure accurate location.
- **Software and hardware limitation:** using the camera imposes a maximal velocity of the robot ($v < 0.45m/s$) to ensure frame stability.

5 PANORAMA: Overview

An autonomous robotic system must be able to objectively decide if it is able to achieve the performance set by the user for the proposed mission. Moreover, during the mission, it must be able to maintain, at best, the quality of service while being exposed to unforeseen events or failures.

The Performance and AutoNOmy using Resource Allocation MAnagement (PANORAMA) method addresses the following question: **How a mission could be carried out while meeting a set of performance goals?**

Implemented in the execution layer of the robot's control architecture, PANORAMA addresses behavioral autonomy, answering the question "How to conduct the planned robotic tasks?".[1] PANORAMA allows to decide offline if the autonomous robotic system can perform the proposed mission and meet the user's performance goals. Moreover, during the mission, PANORAMA verifies that the robotic system meets the performance goals as long as the mission is feasible.

In our works, the following general hypotheses are assumed:

- The initial available amount of energy of the robotic system is known.
- The software and hardware capacities of the robotic system are compatible with the robotic mission.
- The environment map and the initial robot path are known for the mission.

PANORAMA reduces the design space solution sets before selecting one Fig. 3. It can be divided into two main parts of two phases (noted from P0 to P4), which are outlined in detail in the next sections. The first part concerns the problem statement and the mission mapping for resource

[1] And not the decision autonomy which addresses "What task to do?".

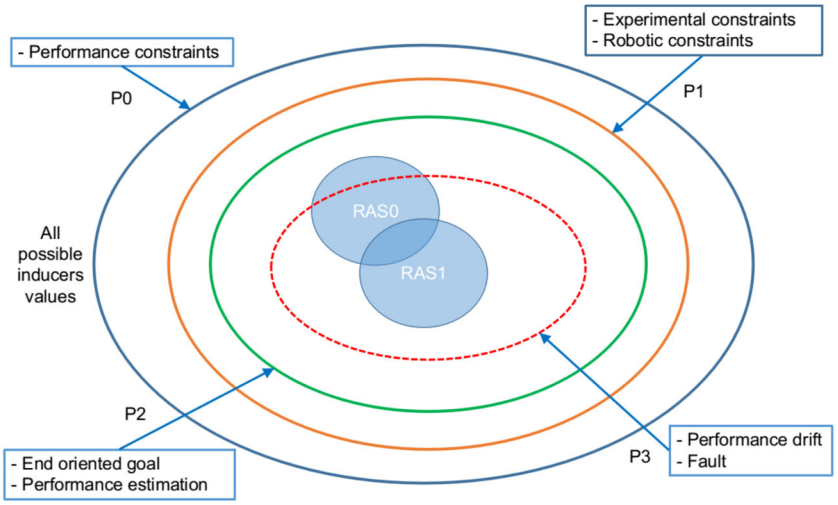

Fig. 3 PANORAMA narrowing availability space for solution

allocation. It integrates the phase P0 which defines and identifies the performance elements: axes, indicators, inducers and goals. It provides the performance constraints that must be verified during the mission. The next offline P1 phase enables the user to build a Mission Scenario MS by mapping the experimental and performance constraints on the initial mission description, i.e. on the different tasks the robot must execute and the different ways these tasks could be implemented.

The mission scenario can be seen as a series of mission slices, called activities, with invariant constraints. The second part concerns the resources allocation process. For each of these activities, the offline P2 phase locally estimates the possible performance values (depending on the hardware and software resources used). Then a decision-making algorithm builds an initial Resource Allocation Solution RAS_0 which enables the mission to be performed while globally satisfying all of the performance constraints. Finally, during the mission, the online P3 phase involves monitoring the real mission performances with regard to the expected and planned performances. If, due to unforeseen events, the models drift or some resources fail, the mission success can no longer be ensured using the current RAS_i, a new RAS_{i+1} that still meets the performance goals is then calculated in real time according to the remaining available resources, if it is possible.

The first part of the PANORAMA approach corresponds to the problem statement dimension of a MOO problem where the decision variables are identified, their range set, the problem constraints enumeration and the objective functions fixed. That allows to define the problem maximum decision space. The second part of PANORAMA correspond to the selection and set-up of the MOO algorithm solution.

Now we will detail the process of the first part of the PANORAMA approach.

6 PANORAMA: Problem Statement and Mission Mapping for Resource Allocation

6.1 PANORAMA P0-Preliminary Performance Definition

The bases of the performance concept are established in this preliminary phase. The user defines the performance axes of interest and then the corresponding performance indicators and inducers are identified. The user also sets the reference value (goal) that each indicator must comply with. That defines the reference constraints of the robotic system for the target mission.

6.1.1 Performance Axes: A Framework for Robotics

For autonomous robotic missions, we propose to represent performance through axes of two types. The *major* axes must necessarily be considered regardless of the robotic mission. The *minor* axes are related to user-oriented performance. We consider that four major axes should always be considered for autonomous mobile robotic missions:

Safety. It is essential to guarantee the physical integrity of Humans, the robot and the environment throughout the mission.
Energy. This is a key point for autonomous missions.
Localization. The robot must be able to localize itself with acceptable accuracy during the mission.
Stability. This concerns the mechanical and control aspects in order to ensure convergence of control and fusion algorithms.

Minor axes concern user-oriented robotic tasks. Depending on the expected service, many performance axes can be

envisaged, such as map accuracy for exploration missions or more generally mission duration.

6.1.2 Performance Inducer and Indicator Identification

For each performance axis, the performance indicators and their corresponding inducers must be identified.

Performance Inducers

Performance inducers are specific elements of the system. Generally in robotics we find:

- Characteristics of the mission environment are important performance inducers. The environment can be static or dynamic, and its physical properties (nature, absorption coefficient, etc.) can be known or not.
- Material resources, such as sensor or actuator capacities, affect the mission performance (accuracy, range, energy consumption, etc.).
- Software resources are also critical for the mission success according to their efficiency, frequency, CPU load, etc. The software integrates the control architecture, which is a key point with regard to mission execution and performance satisfaction. It allows real-time selection of the resources according to the mission plan and its execution. The control frequency impacts the robot's energy consumption, reactivity and control stability.
- Input data: the available input data can influence the performance. For example depending on the chosen path to follow in a motion task, the duration of the mission will be different. Another example could be the input map of the robot environment that can influence the localization performance.
- Finally, energy sources are critical performance inducers for autonomous missions. They can be controlled offline or can be sometimes regenerated during the mission.

Measurement of Performance Indicators

For the performance indicators that can be measured, several classes can be defined depending on the measurement means and the monitoring times. Considering a performance axis $axis$ and a performance indicator ind on this axis, the value of this performance (i.e. its measurement) at time t is denoted $Perf_{axis}^{ind}(t)$. This measurement could be i) either *observed* at a given time from a specific component (for example a dedicated sensor); ii)*estimated* from a model of the performance behavior. This estimation can be updated during the mission execution according to the known past. For example in Fig. 4, the solid line represents the series of the observed values of an indicator. The dotted black line represents the value of the performance indicator estimated from the initial time (initial estimated performance). This figure also shows that estimated performance generally do not completely match with reality (observed performance).

6.1.3 Performance Constraints

A performance constraint implies checking the results of the comparison of an indicator value with a goal. If a performance axis depends on several performance indicators, then all of the performance constraints must be jointly satisfied to obtain an acceptable performance for the considered axis.

Some of the performance constraints must be satisfied throughout the mission. They are called **core-oriented**. Their associated goals can be defined as constant (same goal all along the mission) or variable (the constraint respect is

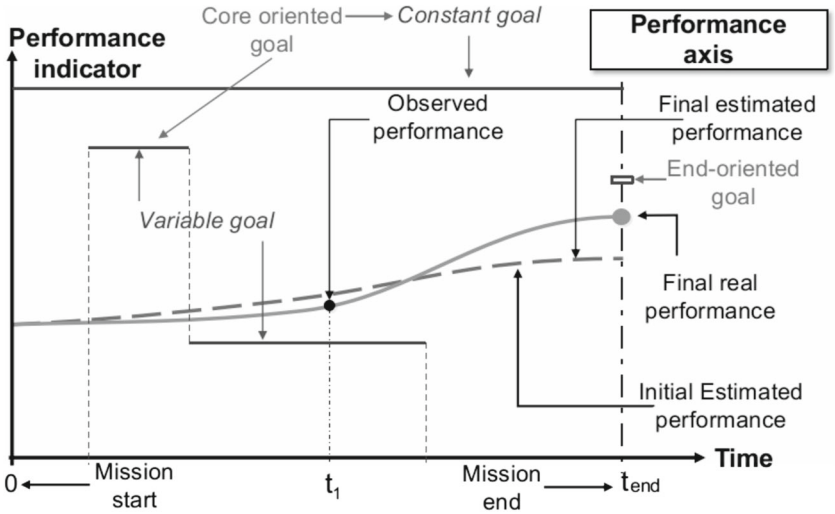

Fig. 4 Measurement of performance indicators

Table 2 Example of P0 phase: definition of the performance indicators and inducers

Axis	Perf. indicators	Perf. inducers
Safety	Harmlessness	Velocity
		Sensor choices and settings (resource allocation)
	Obstacle avoidance capacity	Control algorithm setting (resource allocation)
		Robot dynamics (a priori set)
Energy	Laptop Consumption	Velocity
		Planned path (a priori set)
	Robot Consumption (RC)	Sensor choices and settings (resource allocation)
Duration	Mission Duration (MD)	Velocity
		Planned path (a priori set)
		Control algorithms (resource allocation)

permanently necessary, but with goal values that are variable throughout the mission, see Fig. 4).

Formally, considering a performance axis $axis$ and one of its performance indicators ind, we can define a core-oriented performance constraint as a partial order relation linking the performance value $Perf_{axis}^{ind}(t)$ with regards to its associated goal $Goal_{axis}^{ind}(t)$:

$$Perf_{axis}^{ind}(t) \;\; op \;\; Goal_{axis}^{ind}(t) \; \forall t \in [0, t_{end}], \quad \text{with } op \in \{<, \leq, =, \geq, >\} \tag{1}$$

Conversely, for some performance axes, the defined goals must only be satisfied at the end of the mission, defining an **end-oriented** performance goal. For example, for the energy axis, the overall robot consumption at the end of the mission must be less than a predefined value (the amount of energy initially available in the batteries). Formally, for an end-oriented performance of a given performance indicator ind and axis $axis$, we denote $FPerf_{axis}^{ind}(MS)$ the Final Performance value of the mission at the end ($t = t_{end}$) of the Mission Scenario (MS), and $FGoal_{axis}^{ind}(MS)$ the Final performance Goal that must be respected. Thus an end-oriented performance constraint is defined as:

$$FPerf_{axis}^{ind}(MS) \;\; op \;\; FGoal_{axis}^{ind}(MS) \quad \text{with } op \in \{<, \leq, =, \geq, >\} \tag{2}$$

6.1.4 Example of P0

For simplification, in our example we do not tackle all of the previously presented performance axes. We only focus on two major axes: Safety and Energy, and on the minor axis: Duration. The safety performance is core-oriented as the mission must always be safe, i.e. the safety constraints must always be verified. On the contrary, the energy and duration performances are end-oriented: the robot and the embedded laptop have a limited amount of energy, and the user wants to limit the total duration of the mission.

Table 2 summarizes the indicators and the main inducers of the studied performance axes. In fact, **in our system, we act on only two inducers: the robot's velocity and the resource allocation**. The robot's velocity is an unavoidable inducer, impacting all of the performance indicators. Resource allocation is also an essential inducer, managing the control algorithms (allowing to manage the motion, the obstacle avoidance or the localization) and the sensors used (influencing the localization precision, the consumed energy, the acquisition frequency of data, etc..). The other inducers of the Table 2 are fixed a priori for a given mission and a given robot.

The next step is to establish the performance constraints for each axis. For our mission we define the following performance constraints (resumed Table 3):

Safety axis:

- Harmlessness constraint: in case of unpredictable dynamic obstacles, the obstacle avoidance cannot be guaranteed. Thus, we state that, in presence of moving obstacles, the safety of the robot and its environment must be guaranteed by insuring that the energy dissipated during an impact (denoted e) is less than $4J$[2], i.e. if $Perf_{safety}^{harmless}(t) = e < 4J \; \forall t$. As $e = \frac{1}{2}mv^2$ with m being the mass of the robot and v its velocity, and as the mass of our robot is $m = 25kg$, then this constraint becomes:[3] $v < 0.56m/s$.
- The Obstacle Avoidance (OA) constraint is to never bump into an obstacle. We can expressed it as: $Perf_{safety}^{OA}(t) = min(Dist(o)) > 0m \; \forall o, \forall t$, with $Dist(o)$ the distance to an obstacle o. According to the SMZ algorithm [76], the capacity to guarantee this constraint is a function which depends on the safety radius value, on the environment of the robot, on the availability and the sampling frequency

[2] ISO 10218: norm that limits maximum kinetic energy tolerated for an impact. [77]

[3] To simplify we note v instead of $v(t)$.

Table 3 Example of P0 phase: definition of the performance constraints

Axis	Indicator	Type	Constraint
Safety	Harmlessness	core-oriented	$e < 4J \Rightarrow v < 0.56m/s$
	Obstacle avoidance	core-oriented	$Dist(o) > 0$
Energy	Robot consumption	end-oriented	$FPerf^{RC}_{Energy}(MS) \leq 2.4Wh$
Duration	Mission duration	end-oriented	$FPerf^{MD}_{Duration}(MS) \leq 600s$

of the sensors, on the algorithms used to detect/avoid an obstacle, and on the robot's dynamics and velocity.

Energy axis: the total energy consumed during the mission by both the laptop and the robot must be below a known available amount. We focus here on the consumption of the robot (RC): $FPerf^{RC}_{Energy}(MS) \leq 2.4Wh$.

Duration axis: the total mission duration (MD) must be below a given value. For example: $FPerf^{MD}_{Duration}(MS) \leq 600s$.

6.2 PANORAMA P1 - Preliminary Phase

This section describes the PANORAMA P1 phase and illustrates it on our experimental mission. It designs the mission scenario, i.e. the initial mission is broken down into several activities with their associated sets of possible resources and inducer values, while respecting the core-oriented and experimental constraints.

6.2.1 Mission Decomposition

We consider that a **mission** corresponds to the sequential execution of n_{obj} objectives. An **objective** is a high-level robotic action (going to a point, finding an object, analyzing the data, etc.) that is executed via one or several robotic tasks.

The **mission scenario** is built from the initial mission description by projecting the experimental and performance mission constraints on the initial mission description. We suppose that an objective, as well as the constraints, can be spatially defined in a specific area of the mission map. A constraint can thus be mapped on the mission description to generate a decomposition of the mission objectives in spatial areas called **activities**. So a mission will be broken down into a sequence of n_{act}[4] activities A_k, each activity being associated with constant constraints.

As seen before, the evaluation of the performance constraints depends on the type of the performance. As end-oriented performance constraints must only be satisfied at the end of the mission, they do not influence the mission decomposition in activities, which is mainly based on the core-oriented performance constraints and on the experimental constraints.

Figure 5 highlights the decomposition concept. First, the mission is described through n_{obj} objectives. Then, to consider the specific constraint C_i, the objectives O_{II} and O_{III} are both divided into two activities. Activities A_{II} and A_{III} are different because they do not respect the same constraints, while activities A_{III} and A_{IV} are different because they do not belong to the same objective, even if they have the same constraints.

6.2.2 Mapping of Experimental and Core-Oriented Constraints

Previous constraint mapping leads to the identification of the activities, but it will also be used to reduce the set of the possible performance inducer values.

The first inducer to consider is *resource allocation*: the sensors and algorithms needed to execute a robotic task. The resource allocation inducer is an important element to consider in robotics (even if it is sometimes ignored).

Each task can be implemented using a set of different **Task Implementation Options**. A TIO corresponds to a set of hardware and software resources that can be used to execute the corresponding robotic task. For example, the localization task could be based only on odometers, only on camera image analysis, or both, which defines 3 TIOs for this task. But the mapping of the core-oriented and experimental constraints on the activities[5] could reduce the implementation possibilities of the tasks, thus reducing their set of possible TIOs. For example, in Fig. 5, the task T_i of the objective O_{II} can theoretically be executed using one of the possible TIOs. But the constraint C_i removes all TIOs except TIO_1 and TIO_3 for the activity A_{III}.

To perform a robotic task, a unique TIO must be selected at a given time. Selecting a unique TIO for each robotic task involved in a specific activity allows to define an **Implementation Alternative (IA)**. Many possible IAs could exist for an activity, each one representing a different combination

[4] $n_{act} \geq n_{obj}$

[5] Let us remind that each objective corresponds to a set of robotic tasks that must be executed in parallel. Of course, each activity of an objective inherits its robotic tasks.

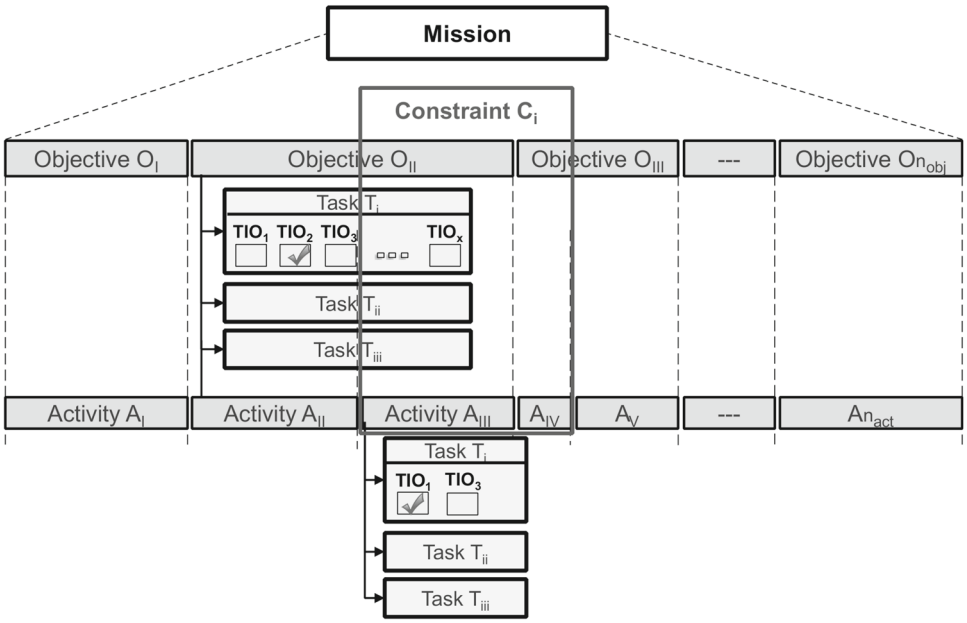

Fig. 5 Description of the mission: constraint mapping and activity identification

of TIOs. For example, Fig. 6 presents an activity A implementing three robotic tasks (T_i, T_{ii}, T_{iii}). The number of TIOs for each of these tasks is respectively 4, 2 and 3. So the number of all the possible implementation alternatives is $n^A_{IA} = 4*2*3 = 24$. **Thus the resource allocation inducer corresponds to the IAs of an activity**

The second essential performance inducer in robotics is the robot's *velocity*, which influences most of the performances. In our experimental mission, velocity is the main inducer of all of the performance axes. Each activity must be associated to a set of possible velocity values. Initially, the velocity value must be lower than the maximal value imposed by the physical characteristics of the robot (experimental constraints). But the robot velocity must also respect the core-oriented performance constraints. For example, safety axis constraints may reduce the maximum velocity value due to the reaction time needed to detect and react to unexpected obstacles.

6.2.3 Example of P1

Mission Decomposition

For the patrolling mission described Section 4.2, Table 4 describes the sequence of objectives that must be sequentially executed and their corresponding robotic tasks.[6] Their spatial location across the mission is defined along a longitudinal axis over the distance travelled by the robot. The robot is supposed to follow a straight path in the middle of the corridors. Initially the robot is located at $x = 0m$, while at the end of the mission, the robot will return to the docking station at $x = 187m$. The operator | denotes a concurrent execution of the robotic tasks.

For the mapping of the constraints, we first consider the experimental ones given Section 4.3. We also have to consider the core-oriented performance constraints, which concern in our case the safety axis (harmlessness and obstacle avoidance) as described Section 6.1.3 and Table 2. We have shown that the harmlessness depends on the robot's velocity, and the obstacle avoidance capacity depends on the width of the corridor (for the safety radius), and on the used sensors, hence they greatly depend on the environment. This is expressed in grey in the Table 5 as follows:

- The environment elements linked with the experimental constraints are given in the lines Env.: the halls H1 and H2, the valves and the glazed areas.
- The line C_{EXP} projects these constraints on the resource allocation inducer: ALL means that all methods are useable, LAS means that only methods using the sonars must be considered (glazed area), and QR means that QR-code and Kinect are needed to ensure accurate localization before the valve monitoring.

Thus, combining the 9 predefined objectives of the initial mission with the mapping of the experimental and core-oriented performance constraints, we generate 17 activities for the patrolling mission. The final mission scenario is presented Table 5, where the 5^{th} line represents the name of the activities and the two next ones their spatial location (x_{start} and x_{end}) throughout the mission round-trip path.

[6] The possible robotic tasks are described in Section 4.1 and Table 1.

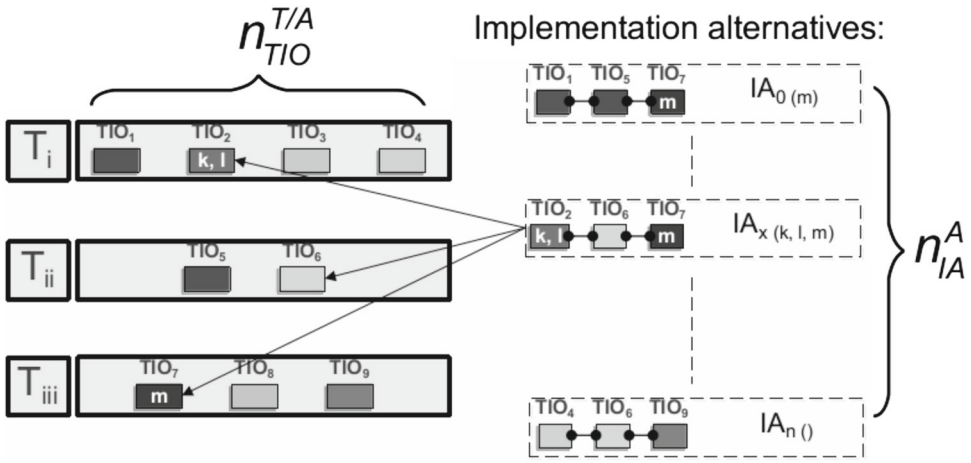

Fig. 6 Implementation alternatives for an activity with 3 robotic tasks

Table 4 Example of P1 phase: description of the patrolling mission

Objective	Location (m) (x_{start} - x_{end})	Objective description	Robotic tasks
O_I	0 - 37	Travel to V1	MOV | LOC
O_{II}	37	Turn to V1	TUR | LOC
O_{III}	37	Analyse V1 state	DET
O_{IV}	37	Turn back	TUR | LOC
O_V	37 - 93.5	Travel to V2	MOV | LOC
O_{VI}	93.5	Turn to V2	TUR | LOC
O_{VII}	93.5	Analyse V2 state	DET
O_{VIII}	93.5	Turn back	TUR | LOC
O_{IX}	93.5 - 187	Travel back to Docking Station	MOV | LOC

Table 5 The initial patrolling mission scenario MS

Objective	**O_I**			**O_{II}**	**O_{III}**	**O_{IV}**	**O_V**				**O_{VI}**	**O_{VII}**	**O_{VIII}**	**O_{IX}**			
Robotic Tasks	MOV | LOC			TUR | LOC	DET	TUR | LOC	MOV | LOC				TUR | LOC	DET	TUR | LOC	MOV | LOC			
Env.	**H1**										**H2**			**H1**			
		Glass		**Valve**			**Glass**				**Valve**					**Glass**	
C_{EXP}	**ALL**	**US**	**QR/US**				**US**	**ALL**		**QR**				**ALL**		**US**	**ALL**
Activity	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9	A_{10}	A_{11}	A_{12}	A_{13}	A_{14}	A_{15}	A_{16}	A_{17}
x_{start} (m)	0	31	34	37	37	37	37	41	63.5	91.5	93.5	93.5	93.5	93.5	123.5	146	156
x_{end} (m)	31	34	37	37	37	37	41	63.5	91.5	93.5	93.5	93.5	93.5	123.5	146	156	187
n_{IA}^{Oi}	21			3	2	3	21				3	2	3	21			
$n_{IA}^{A_j}$	21	12	4	3	2	3	12	21	21	4	3	2	3	21	21	12	21
$v_{max}^{A_j}$ (m/s)	0.41	0.33	0.33	-	-	-	0.33	0.41	0.56	0.41	-	-	-	0.56	0.41	0.33	0.41

Mapping of the Constraints

We have to define the set of values for the pertinent inducers of our system: the Implementation Alternatives and the velocity of the robot. Information on the possible values of these two inducers are given in the last three lines of Table 5:

- $n_{IA}^{O_i}$ and $n_{IA}^{A_j}$ respectively represent the maximal number of IAs that are initially available for the objective (depending on the objective's tasks), then the number of possible IAs for the activity considering the constraints. For example, for O_I, we have seen in Table 1 there are 7 different TIOs for the moving task, and 3 for the location task, thus generating 21 possible IAs. This number decreases with the constraint mapping: A2 must only use sonar for the moving task, leading to only 12 possible IAs, and A3 must also use the Kinect for localization, thus reducing the choice to only 4 possible IAs.
- Then the last line gives the maximum velocity value, set as follows:
  - Throughout the mission the velocity cannot exceed $0.76m/s$ because of the robot's physical limitations, but also cannot exceed $0.56m/s$ because of the harmlessness safety constraint.
  - The safety radius for the obstacle avoidance constraint can be higher in H2 than in H1, thus $v_{max}^{A_j}$ in H2 remains equal to $0.56m/s$ (e.g. in A9), whereas $v_{max}^{A_j} = 0.41m/s$ in H1 (e.g. in A_8).
  - Within the glazed area, the obstacle avoidance method must use sonar information, which have lower sampling frequency than the lasers. Thus $v_{max}^{A_j} = 0.33m/s$ when sonars must be used (e.g. in A_2).
  - If the QRCN method is used for location, $v_{max}^{A_j} = 0.41m/s$ (e.g. in A_{10}) to ensure the stability of the Kinect frame imaging.

The initial mission scenario being defined the resource allocation process can start.

7 PANORAMA: Resources Allocation Process

At the end of the P1 phase, we have defined the performance axes, and the performance indicators and inducers that are relevant for the targeted system. We also define the constraints related to this system in a given context and mission. Phase P1 also allows to reduce the set of possible values of the performance inducers while respecting the experimental and core-oriented constraints. Now, it is necessary to select the most adequate inducer value respecting the end-oriented performance constraints. As one off the main inducer are the IAs which correspond to the hardware and software resources of the activities, the next two phases of PANORAMA mainly deal with the resource allocation problem based on the estimation of end-oriented mission performance.

7.1 End-Oriented Performance Estimation

To estimate the end-oriented performance constraints, i.e. at the mission level, it is necessary to estimate the values of the indicators performance based on the decomposition of the mission into activities.

7.1.1 Estimation of the Mission Performance

As defined in Section 6.1.3, to evaluate the end-oriented performance constraints, we need to estimate the final performance value $FPerf_{axis}^{ind}(MS)$ of the mission MS. As MS is a sequence of activities, we decompose the computation of the performance mission adding the local performance of each activity. Thus, with $FPerf_{axis}^{ind}(A_j)$ the estimation of the performance value at the end of the activity period A_j, we have:

$$FPerf_{axis}^{ind}(MS) = \sum_{j=1}^{n_{act}} FPerf_{axis}^{ind}(A_j) \tag{3}$$

7.1.2 Estimation of an Activity Performance

We supposed that it exists a model that provides an estimation of the performance indicator value from the inducers values. Thus, all the inducers must be set to a specific value, chosen in the set of all the possible values remaining after the mapping of the core-oriented and experimental constraints. But many possible values are still possible, even more if we consider the combination of all the possible values of all the performance inducers. Hence, it is necessary to establish some strategies allowing to estimate the activity performance. In PANORAMA we use strategies inspired by both empirical and well-admitted logic.

The first main inducer of most of the performance axes is the robot's velocity, which is a continuous parameter, leading to an infinite number of possible values. Thus, we must choose a restricted number of inducer values to consider. More precisely, we select only one value for the velocity inducer, significantly reducing the number of calculated performance values.

The second important element to consider is the allocation resource through the implementation alternatives which manages the control algorithms and the sensors used. The way a task is executed has an influence in many of the end-oriented performances. Typically, the choice of the used sensors has an important impact on the energy performance. Thus we choose to consider all the possible IAs of an activity,

leading to consider several possible performance values for each activity (one per possible IA).

However, when executing a mission only one IA is selected per activity. So when we estimate the performance value of a mission, we consider that the performance value of an activity A_j to the value of its selected implementation alternative IA_k:

$$FPerf_{axis}^{ind}(A_j) = FPerf_{axis}^{ind}(IA_k) \quad (4)$$

This allows to calculate the performance values **of the performance indicators** of the mission, necessary to evaluate the end-oriented performance constraints.

7.1.3 Example of Final End-Oriented Performance Estimation

We remind that in our example, we focus on two end-oriented constraints (Table 3): the mission duration and the energy consumption of the robot energy. In our resource allocation algorithm, we first address the duration then the energy.

End-Oriented Performance Estimation

We now want to estimate the performance value of the IAs (then the activities) for the two indicators implicated in the end-oriented performance we focus on (see Table 3): the mission duration and the robot energy consumption. We will not give all the numerical values in detail here, but only present the methods we used to estimate the values of these performance indicators.

The estimation of the energy performance indicator $FPerf_{Energy}^{RC}$ of our robotic system for an IA is done using the following energy consumption model [78]:

$$FPerf_{Energy}^{RC}(IA) = \beta_0 P_{R_{Motion}}(v_{max}^{IA}) + \beta_1 P_{R_{Sonar}} + \beta_2 P_{R_{Kinect}} + k_1 \beta_3 P_{R_{Laser}} \quad (5)$$

where $k_1 \in \{0, 1, 2\}$ denotes the number of active lasers, β_i are Boolean coefficients that indicate whether the corresponding component is used or not, and v_{max}^{IA} represents the selected maximal velocity of the IA. $P_{R_{Motion}}()$ which corresponds to the instantaneous motion power consumption has been determined using an experimental model definition [79].

Concerning the duration performance indicator of an IA, it is easy to compute its planned duration when the robot's planned path (considered as a straight line) is known and its velocity is set. Activities (and thus IAs) dedicated to the valve observation are supposed to have a constant and, known, duration.

We now present how the velocity parameter is selected.

Velocity Parameter Selection

To calculate the performance values, we have to set the velocity value for a given IA. For a given activity A_j (and thus for all of its IAs), the velocity must be chosen in the interval $[0, v_{max}^{A_j}]$. A priori, using the maximal velocity is a good solution for duration. Also, in some previous studies dealing with the energy model of our robot [78, 79], we have shown that, to travel a given distance, the optimal velocity value for the energy axis is $v_{opt} = 0.76m/s$. Moreover, the consumed energy decreases in a monotonic way with the velocity until v_{opt} is reached. So, while v_{opt} is not reached during the mission, moving as fast as possible minimizes the consumed energy.

Furthermore, the safety constraints reduce the local maximal velocity in the activities. In particular, the harmlessness constraint imposes $v < 0.56m/s < v_{opt}$. So in our context, it is justified, from the energy viewpoint, to go as fast as possible. Thus we estimate the performance indicators by setting the robot's velocity at the maximal velocity value allowed for each IA.

At the end of this phase, we now have estimated, for a mission, all the values of the performance inducers of all IAs of activities. From the Eq. 3 we are able to estimate the final end-oriented performances of every possible mission scenario. We can now use these information as a basis for the offline resource allocation problem.

7.2 PANORAMA P2 - Offline Resource Allocation Phase

Let a Resource Allocation Solution (RAS) be a plan which proposes a resource allocation, i.e. the selection of an IA for each activity of the mission, while satisfying all the performance constraints, included the end-oriented ones. The P2 phase aims to identify RAS_0, which corresponds to the initial RAS designed before the the mission launch.

7.2.1 Complexity of the Resource Allocation Problem

More than only choosing an allocation solution, we are interested in an appropriate local selection of AIs, while globally verifying the end-oriented performance constraints. This corresponds to the formulation of a knapsack class problem, which is known to be NP-hard. The decision complexity of our problem relates to the total number of combinations of the possible IAs for each activity, called Number of Global Alternatives (NGA). It corresponds to the overall space that

should be explored to find the optimal solution, and is equal to the product of the numbers of all the different possible IA combinations:

$$NGA = \prod_{k=1}^{n_{act}} n_{IA}^{A_j} \tag{6}$$

NGA significantly increases with the number of tasks per activity and of TIOs per task. For example in our case-study mission, we have

$$NGA \approx 7.7 * 10^{14}$$

The problem now is: How to efficiently solve this NP-hard problem?

7.2.2 Resource Allocation Algorithm: Principles

The efficiency of PANORAMA greatly depends on the efficiency of the Resource Allocation (RA) Algorithm . We dealt with this complexity by adapting the algorithm proposed in [80] for human resource allocation to enterprise process. This algorithm seeks effective solutions to the knapsack problem. When several end-oriented performance constraints must be considered, the problem becomes a multiple knapsack problem and the iterative version of the RA algorithm is used.

The efficiency of the chosen RA algorithm is based on three main assumptions:

- For a given activity and performance indicator, the IAs can be sorted from best to worst with regard to their local performance estimation.
- Each activity performances evaluation is independent from each other.
- The composition law used for the estimation of the final performance of the mission from each local performance value preserves the local ordering relation. Thus, when the selection, for an activity, of a performance that is lower (resp. higher) than the current selected choice will lead to a lower (resp. higher) final performance of the mission.

The algorithm executes the 3 following steps, for each indicator implicated into each end-oriented constraint: sorting of the IAs, selection of the highest IAs combination satisfying the performance constraint, and the reduction of the set of possible IAs.

The first step is **the sorting of the IAs for each activity (see Fig. 7) depending on their performance value** $FPerf_{axis}^{ind}(AI_j)$. They are sorted depending on the considered performance constraints, from the best to the worst values. For example, for the duration constraint where the final performance must be inferior to a goal, the smallest IA's performance value is considered to be at the bottom and the greatest one at the top.

The second step is **the selection of the highest IAs combination satisfying the considered performance constraint**. To find a solution, the RA algorithm uses a binary search algorithm for each activity so as to find locally the better pivotal point selection of IA, leading to the best global solution. The algorithm for a local IA selection is as follows (Fig. 7):

1. The initial selection is composed of the set of the highest IA for each activity.
2. Then the final performance value of the mission is computed with the local performance values of these selected IAs (see Section 7.1).
3. If the end-oriented performance constraint is not verified, the selected IA of the last activity is modified, while searching by dichotomy for the highest value that could verify the constraint.
4. If the constraint is still not verified even if the selected IA is the lowest one, then the process described previously in 3 is repeated for the previous activity, and so on.

The chosen RA algorithm, like some others MOO algorithms, can be used sequentially for different performance objectives sorted in order of importance. Then each objective is treated one after the other so that each sub-problem does

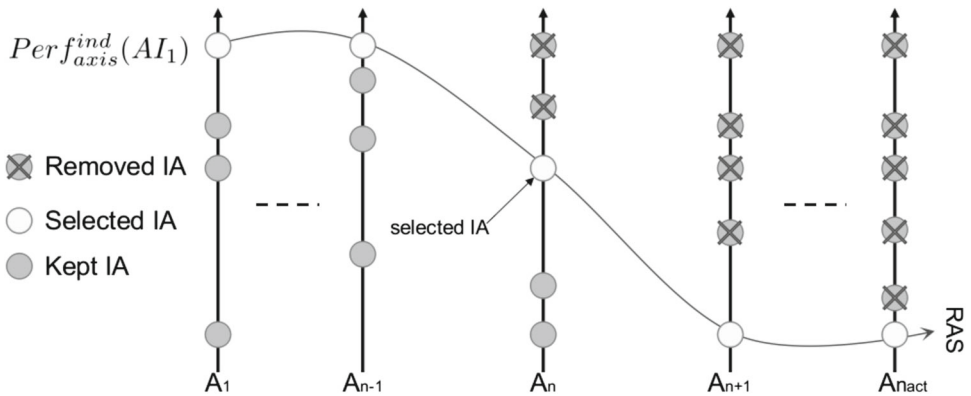

Fig. 7 Sorting and selection of the IAs for one end-oriented performance constraint: Resource Allocation algorithm principle

Table 6 RAS_0 for the patrolling mission

Objective	O_I			O_{II}	O_{III}	O_{IV}	O_V				O_{VI}	O_{VII}	O_{VIII}	O_{IX}			
Robotic Tasks	MOV | LOC			TUR | LOC	DET	TUR | LOC	MOV | LOC				TUR | LOC	DET	TUR | LOC	MOV | LOC			
Env.	H1							H2							H1		
		Glass		Valve			Glass				Valve					Glass	
C_{EXP}	ALL	US	QR/US				US	ALL		QR				ALL		US	ALL
Activity	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9	A_{10}	A_{11}	A_{12}	A_{13}	A_{14}	A_{15}	A_{16}	A_{17}
v^{A_j} (m/s)	0.41	0.33	0.33	-	-	-	0.33	0.41	0.41	0.41	-	-	-	0.41	0.41	0.33	0.41
RAS0	M5 | L2	M5 | L2	M5 | L2	T1 | L2	D2	T1 | L2	M5 | L2	M5 | L2	M5 | L2	M5 | L2	T1 | L2	D2	T1 | L2	M5 | L2	M5 | L2	M5 | L2	M3 | L1

not degrade any of the previous solutions. This way of treatment of MOO problem is called lexicographic approach for optimal searching. In our approach, the final solution is not optimal, but reachable and the principle remains the same. So we can say that we implement a lexicographic-like approach to solve the MOO problem.

When a solution has been found for one end-oriented performance constraint, the selected IAs are the highest possible ones for all the activities while satisfying this constraint. Indeed, the objective of this selection is to maximize the state space of valid solutions to optimize the chance of finding a solution for the rest of the end-oriented constraints.

We then execute the third step of our algorithm: **the reduction of the set of the possible IAs**, considering only the subset still acceptable. In Fig. 7, this subset corresponds to the IAs that are under the dotted line. Then we address the next performance constraints,[7] looping to the step 1, until there is no further end-oriented constraint to respect, or if there is no resource allocation solution.

Thus, the resource allocation algorithm can select, for all mission activities, the IAs that must be used to satisfy all of the performance constraints of the mission. This algorithm does not give the optimal solution for our resource allocation problem, but it has the essential advantage of being low in complexity (logarithmic) meaning that it can find a good resource allocation in real time (necessary for the phase P3).

The selected initial solution is named RAS_0. That is the end of the offline process and the mission can start using this pre-defined plan. If there is no solution, then we consider that the mission cannot be performed regarding the performance constraints.

[7] Remark: this shows that the order in which the constraints are processed may affect the resulting solution set.

7.2.3 Example of Resource Allocation Solution

Firstly, it's important to remind that the core-oriented constraints (here the safety) remain satisfied for all IAs, guarantying this performance viewpoint all along the mission. Then, the first end-oriented performance considered is the Duration. The application of RA algorithm allows to remove IAs that cannot satisfy the duration constraint. The second end-oriented performance considered is the Energy. Once more RA algorithm allows to remove IAs leading to excessive consumption.

Now the problem is: **Among all the remaining possible IAs, how to select the exact IA sequence RAS_0 to do the mission?**

In our example, the verification of the duration constraint does not reduce the IAs possibilities. Thus this performance has no influence in the IAs choices. For the Energy performance, the classical strategy would be to select the least energy consuming IAs. Unfortunately this choice would lead to the selection of odometry for all localization tasks, to prevent using more consuming sensors like the lasers. But it is well known that due to odometry drift, using this solution all along a mission is unreasonable. To overcome this limitation, we will suppose that using the more consuming sensors allows to use better localization algorithms. So we choose to always select the most energy consuming IA still satisfying globally the end-oriented energy performance constraint.

In the following we do not give all the details of the algorithm steps, but the resulting RAS_0 defined by our algorithm is given in Table 6. In the last RAS_0 line, each cell represents the selected implementation alternative of the associated activity. These IAs are described with one TIO for each robotic task required for this activity, according to the TIO numbering given in Table 1. Concerning the

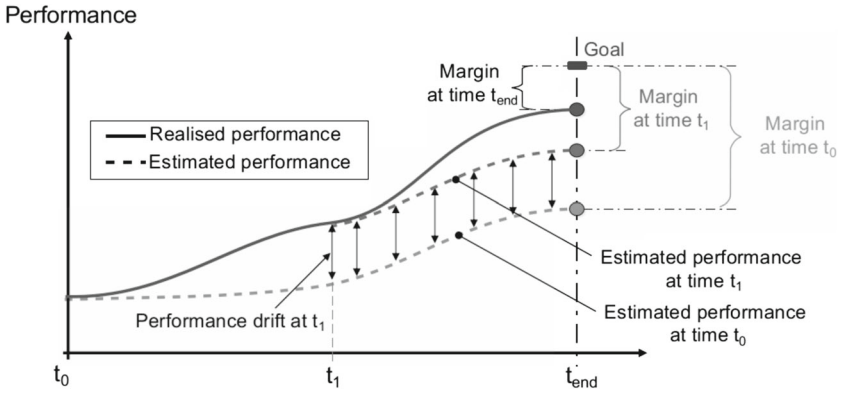

Fig. 8 The performance drift and margin concept

robot consumption constraint, for all the activities from A1 to A16 the "best" IAs has been selected, i.e. the most energy-consuming ones. For example, for the moving task of activity A1, we select the TIO M5 which uses all the sensors, and the TIO L2 for the localization task, which uses the high energy-consuming Kinect. However, for A_{17}, the more energy consuming IA M5|L2 could not be used as it violates the final energy constraint, M3|L1 is used instead.

7.3 PANORAMA P3 - Online Performance Management Phase

The P2 phase provides a solution RAS_0 to execute the mission respecting the performance constraints. But this initial resource allocation solution could be questioned for two major reasons: 1) either the real mission performances do not correspond to the planned ones, leading to violation of the end-oriented performance constraints; 2) or hardware/software failures induce a resource reallocation. To deal with these unanticipated problems, a new RAS_{i+1} must be calculated during the mission if RAS_i cannot be used.

Thus the P3 phase of PANORAMA deals with dynamic resource allocation management during the mission.

7.3.1 Performance Margins

The margin concept is widely used in control theory but rarely to manage mission execution. Hereafter, the robustness of PANORAMA with regard to model drifts or unforeseen events mainly depends on the efficiency of the resource allocation algorithm, on the accuracy of the models, and on the performance margin management. For an end-oriented constraint, a performance margin can be thought of as a performance "stock" used to overcome the drift between the performance estimated for the planned mission and the actual performance during the execution of that mission. So the performance robustness increases with the growth of the performance margin. The best IAs in terms of end-oriented performances are the ones which maximise these margins. In the following, we explain the concept of performance drift and margin, and re-define end-oriented constraints to be managed in real-time.

Performance Drift In the planned mission, the environment is considered as static, the models are assumed to be perfect, and the planned path is supposed to be perfectly exactly followed, which is not the case in reality [78]. This performance drift could be caused for example by the imprecision of the model used to estimate the initial performance value, or induced by the difference between the real path and the planned one. So during a mission, the performance drift must be monitored by comparing the instantaneous performance measurement to the expected value.

Figure 8 illustrates such a drift: first, at t_0, the performance value is estimated from a model, given the lowest dotted curve. But the unforeseen event encountered during the mission lead to the real performance values represented by the continuous curve. At t_1, an observed value of the performance is available, which reveals a performance drift. Thus a new estimation could be calculated for the performance, leading to the higher dotted curve.

Performance margin

At a given time t, the performance margin corresponds to the difference between the final performance goal and the final performance value of the system estimated at this time. Figure 8 shows an example of performance margins. The first margin, denoted $M^{ind}_{axis}(t_0)$, is the difference between the final performance value[8] estimated at t_0 (bottom grey circle) and the performance goal of the mission. In t_1 the detection of the performance drift leads to a new estimated final performance value and then a lower margin $M^{ind}_{axis}(t_1)$.

Given an end-oriented performance constraint, we have defined in Section 6.1.3 that $FPerf^{ind}_{axis}(MS)$ is the final

[8] for the performance axis $axis$ and the performance indicator ind

performance value observed at the end of the mission (i.e. at t_{end}). But at any time t of the mission, it can be estimated from the current performance value measured at t combined with the value estimated with a performance model from t. This allows to project the final performance value in the future from the current state. Thus we introduce another parameter to this function: the instant from which the value has been estimated. We thus define $FPerf^{ind}_{axis}(MS)(t)$ as the final performance value of the mission MS estimated from the time t.

And the performance margin is formally defined as:

$$M^{ind}_{axis}(t) = FGoal^{ind}_{axis}(MS) - FPerf^{ind}_{axis}(MS)(t) \quad (7)$$

In Fig. 8, we have 3 performance margins: the final one, calculated from the observed measurement value $FPerf^{ind}_{axis}(MS)(t_{end})$, and two margins estimated during the mission at time t_0 and t_1. We see that these 3 values are different, due to the drift of the real performance values reducing the margin.

Real-Time End-Oriented Constraints Verification

A mission is successful if the performance margins of all of its end-oriented performance constraints remain positive until the end of the mission, i.e. if:

$$M^{ind}_{axis}(t) \geq 0 \quad \forall axis, \forall ind, \forall t \quad (8)$$

Due to the run-time performance drift, the PANORAMA approach must be able to dynamically customize the software and hardware resource allocation to ensure the mission success.

7.3.2 Dynamic Resource Allocation Solution

During the mission, one performance margin can become negative. This means that, in the future, the associated (end-oriented) performance constraint can no longer be verified with the current Resource Allocation Solution RAS_i. In addition, the mission cannot continue if a hardware or software resource fails and if that resource belongs to the currently executed IA or to one of the scheduled IAs in the current RAS_i. In these cases, a new RAS_{i+1} must be determined. This allows to anticipate future violations of constraints, to manage the availability of resources or to adapt the mission.

RAS replanning uses the same process as in P2, but a Reduced Mission Scenario (RMS) is considered instead of the initial MS, starting from the mission's current state. If some resources are no longer available, the new RAS_{i+1} will be calculated by eliminating IAs which use these failed resources. If no new solution can be found, the mission will be aborted because the performance constraints can no longer be satisfied.

7.3.3 Example of P3

The performance margins are managed differently depending on the strategies used for the considered performance constraint:

- **Duration:** since the robot moves as fast as possible (while satisfying the safety constraint), the duration margin is initially maximized but as a result there are not many ways to improve it in case of violation.
- **Energy:** on the contrary, the local selection, for each activity, of the most consuming IA compatible with all constraints leads to a limited energy margin, but with many solutions to improve it in case of violation.

In the following experimental example (Fig. 9) we consider the previous patrolling mission, with the deliberate addition of many obstacles to be avoided, and of an error detection on one laser sensor (X=118m).

Figure 9 shows the experimental curves of the performance margins for the mission duration (duration margin DM, at the top) and the robot energy (robot energy margin REM, at the bottom). In this figure, DM regularly decreases due to obstacle avoidance, but the margin remains positive.

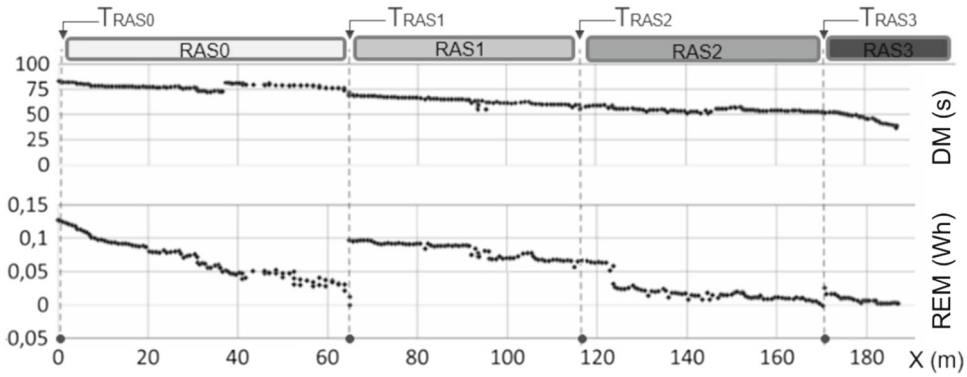

Fig. 9 Performance margins of the patrolling mission

Table 7 The different RAS_i used in the experimental patrolling mission

Objective	O_I			O_{II}	O_{III}	O_{IV}	O_V				O_{VI}	O_{VII}	O_{VIII}	O_{IX}			
Robotic Tasks	MOV | LOC			TUR | LOC	DET	TUR | LOC	MOV | LOC				TUR | LOC	DET	TUR | LOC	MOV | LOC			
Env.	H1							H2						H1			
		Glass		Valve			Glass				Valve					Glass	
C_{EXP}	ALL	US	QR/US				US	ALL		QR				ALL		US	ALL
Activity	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9	A_{10}	A_{11}	A_{12}	A_{13}	A_{14}	A_{15}	A_{16}	A_{17}
v^{A_j} (m/s)	0.41	0.33	0.33	-	-	-	0.33	0.41	0.41	0.41	-	-	-	0.41	0.41	0.33	0.41
RAS0	**M5 | L2**	**M5 | L2**	**M5 | L2**	**T1 | L2**	**D2**	**T1 | L2**	**M5 | L2**	**M5 | L2**	M5 | L2	M5 | L2	T1 | L2	D2	T1 | L2	M5 | L2	M5 | L2	M5 | L2	M3 | L1
RAS1								**M5 | L2**	**M5 | L2**	**M5 | L2**	**T1 | L2**	**D2**	**T1 | L2**	**M5 | L2**	M5 | L2	M1 | L2	M1 | L1
RAS2														**M3 | L2**	**M3 | L2**	**M1 | L2**	**M3 | L2**
RAS3																	**M1 | L2**

For REM, we identify four singular points denoted $TRAS_i$, directly linked to the detection of a particular event during the mission which leads to a new RAS calculation. Table 7 summarizes these different RAS used during this patrolling mission. The control schemes actually executed during the mission are in bold. We remind that the patrolling mission is described in Fig. 2, the TIOs in Table 1, and the robotic tasks in Table 4.

- $TRAS_0$: at the beginning of the mission, the initial resource allocation solution RAS_0 satisfying all of the performance constraints has been found: the mission starts.
- $TRAS_1$: at the location $x = 64m$, during the execution of A_8, the robot's energy margin becomes negative due to odometry drift, obstacle avoidance and consumption model errors. A new resource allocation solution RAS_1 is calculated to return to a positive energy margin. It is obtained by modifying the IA allocation for activities A_{16} and A_{17} by using only sonar (less consuming) for obstacle avoidance (TIOs M_5 and M_3 are replaced by M_1). Switching off lasers in A_{16} and A_{17} reduces the overall energy consumption and so increase the energy margin.
- $TRAS_2$: during the execution of A_{14} (at $x = 118m$), a fault is simulated on one laser used in the current activity (as well as in future ones). So a new RAS_2 is calculated that eliminates this hardware resource. All activities using two lasers are switched to use a single one: TIO M_5 is switched to M_3 in activities A_{14} and A_{15}. In return, the energy saved by using a single laser allows M_3 to be rescheduled for A_{17}.
- $TRAS_3$: dring the last activity, at $x = 170m$, the energy margin again becomes negative, for the same reasons as in $TRAS_1$, and the same strategy is used (sonar vs laser) to save energy to finish the mission.

So finally, despite several unforeseen events, the mission has been successfully executed while satisfying the predefined performance constraints thanks to resource and constraint autonomous management.

We present now how to implement the PANORAMA methodology for a mission with a given robotic system.

8 PANORAMA: The Implementation Process

To implement the PANORAMA process, the user must verify that certain working hypotheses are satisfied:

- The mission environment is known, a map is available to perform localisation.
- The robot's moving and sensing capacities are compatible with the mission.
- The robot's available robotics algorithms are adapted to the mission.
- The robot's planning path is known.

If these hypothesis are satisfied the PANORAMA implementation process can start. Figure 10 summarizes the implementation process followed by the user.

Before being able to use PANORAMA, an important task requiring many interconnected steps is necessary. It can be decomposed into three steps related to the description of the robotic aspects of the mission (in yellow in Fig. 10), the

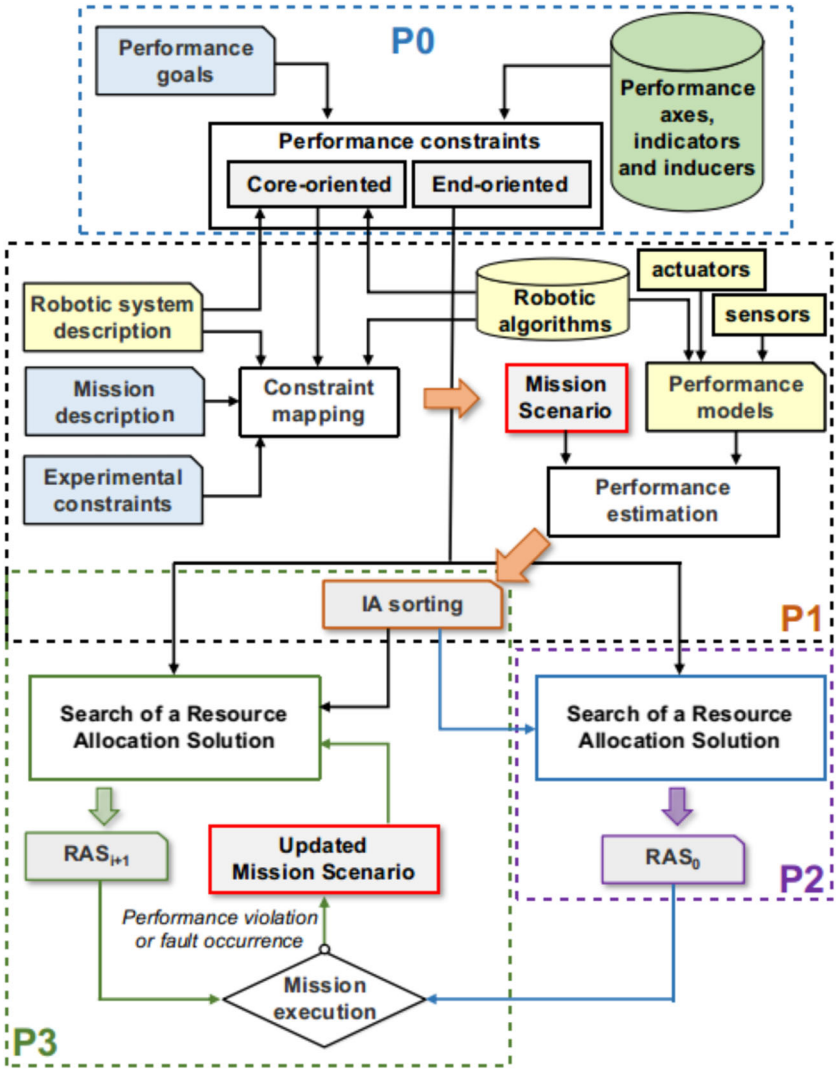

Fig. 10 The PANORAMA implementation process

performance axes (in green in Fig. 10), and the mission and its experimental constraints (in blue in Fig. 10).

The description of the robotic aspects of the mission involves several steps. Firstly, the characteristics of the robotic system must be defined. This encompasses various aspects such as the robot's dimensions, available sensors and actuators, maximum linear and angular velocity, acceleration, payload capacity, and more. It also involves gathering important information about the robotic control architecture, including factors like the frequency of the basic control loop and whether an embedded laptop with or without its own battery is used. The work related to actuators and sensors modelling can be particularly time-consuming but is crucial for optimizing energy and security performance. Concerning actuators, it is essential to experimentally identify the relationship between the robot's velocity and power consumption [79]. This addresses the propulsion aspect of energy consumption. As for sensors, obtaining various pieces of information, including operating frequencies, power consumption, usage conditions, accuracy, and more, is necessary. Additionally, accurately estimating the energy level of the battery (rather than relying solely on vendor product data) is crucial information for the mission.

Secondly, the robotic algorithms can be identified by establishing their connections with the robotic tasks they can be involved in, as well as the actuators and sensors that are either necessarily or potentially used. Some robotic algorithms may also have operational constraints, such as maintaining a maximum robot velocity to reduce image blurring in the case of a Kinect camera, for example.

Thirdly, if an embedded laptop is used with its own battery, it is essential to identify the power consumption of various laptop connections, including those associated with low-level robot control and sensors, as well as the power consumption of different robotic algorithms [79].

Finally, all the performance estimation models must be identified. These models will enable the estimation of activity performance during the PANORAMA process for each selected combination of robotic algorithms and chosen inducer configuration values.

The different performance axes must also be described:

- The performance axes are defined in terms of indicators, inducers, and performance class (core or end-oriented). Cause-effect diagrams can be useful tools for identifying performance indicators and inducers. The identification of relevant inducers is crucial. Some can be controlled before the mission begins, such as the chosen travel path or the energy level of the robot's battery. Others can be controlled during the mission, such as the robot's velocity or sensor frequencies, for instance.
- Each performance indicator must have a sorting function.
- For each end-oriented goal, the global performance composition law must preserve the local performance partial order.

While the description of the robotic aspects of the mission and of the performance axes are time-consuming, they offer the advantage of potential reuse in future missions involving the same robot, robotics algorithms, and sensors.

The final initialisation step is dedicated to the mission description. It can be decomposed in two parts. Firstly, the user must describe the mission plan within the environment. This description must specify which robotic tasks are involved in the mission and their spatial constraints. Concurrent implementation of several tasks is possible. For each task, the user must also select the various robotic algorithms that could potentially be used. Additionally, the planned mission path must be specified, as this is essential information for estimating the performance of the Duration and Energy performance indicators. Secondly, the user must overlay the mission description with the experimental constraints that need to be verified and specify their spatial constraints. These constraints may include areas in the environment that impact sensor functionality or the limitations under which the user mandates the selection of specific robotic algorithms.

This ends the initialisation of the PANORAMA methodology which can be now really launched (Fig. 10) following the 4 phases previously detailed in the paper.

- The Preliminary phase P0 is dedicated to defining mission performance. In this phase, the user selects the performance axes they want to consider and the performance goals they aim to achieve. Subsequently, the core-oriented and end-oriented constraints are derived from the performance description database.
- The Preliminary phase P1 is dedicated to integrating all the constraints and estimating the performance of each Implementation Alternative. Mapping the constraints is a challenging task due to their diverse origins. This mapping, in conjunction with the mission plan, helps identify the final activities with constant constraints and determines the functioning range of the inducers within these activities. Once the mission scenario generates the performance models, it becomes possible to compute the performance level of each Implementation Alternative for each performance axis. Finally, for each identified activity and performance axis, the available Implementation Alternatives are locally sorted.
- Phase P2 involves the offline search for the initial Resource Allocation Solution. It consists of sequentially applying the resource allocation algorithm based on the chosen order of performance importance.
- Phase P3 focuses on online performance management in the event of a final performance goal violation or the detection of a faulty robotic task. It follows the same logic as Phase P2 to generate a new Resource Allocation Solution based on the current position of the robot within the mission scenario.

For the current implementation of this work, all initialization steps were performed manually. However, using the generated data structures, all PANORAMA phases have been integrated into the robot control architecture to manage the robot's mission performance. In the future, it is conceivable to develop mission scenario and performance description editors. Furthermore, once the robotic platform is fully described, it would be possible to offer off-the-shelf components for the robot's description, robotic task implementation based on selected sensors and actuators, and performance computation.

9 Conclusion

This paper addresses the problem of hardware and software resources allocation during autonomous robotic mission. It can be viewed as a Multi Objectives Optimisation problem and can be considered as a multiple NP-hard Knapsack problem. We propose and detail a new approach, named PANORAMA implemented in the execution layer of the robot's control architecture. It enhances the robot behavioral autonomy while addressing the performance satisfaction issue for autonomous robotics. Before the autonomous mission launch it allows to ensure that some performance goals can be achieved. During a mission, the robot can autonomously and dynamically choose which hardware and software resources must be used locally to meet these performance goals.

We highlighted the lack of consensus concerning the performance for robotic missions and we propose a definition for this concept. The PANORAMA approach is divided into four

main phases. The initial phase P0 is dedicated to the definition of the performance elements: characterization of the performance axes and identification of the performance inducers, indicators and their estimation models. Then from an initial description of the robotic system and the mission, phase P1 consists in mapping some of different constraints (environment, robotic platform, safety, etc..) on this initial mission to decompose the mission into specific sections, denoted activities, with constant constraints. Phase P2 concerns off-line calculation of an initial resource hardware and software allocation solution satisfying the performance goals. Finally, the last phase, using online monitoring of performance margins, verifies if these constraints remain satisfied throughout the mission despite the unavoidable performance drifts and some unforeseen events or hardware or software failure. In case of violation of a performance constraint, the robot can adapt its current and future resource allocation, or detect that there are no longer any solutions respecting the performance constraints. The main aspects of the PANORAMA approach are illustrated through an experimental patrolling mission.

The proposed methodology addresses the hardware and software resource allocation issue at the mission level which is often neglected in robotics. It provides an efficient answer to the embedded real-time resources allocation problem in robotic control architectures. It considers many core and end-oriented performance viewpoints. Among them, the energy axe, which is seldom considered in detail at the mission level, is thoroughly managed. Moreover, it enhances the robustness of the mission by finding adapted recovery solutions in case of hardware or software resource failures.

However, this approach has some limitations with regard to long and complex missions:

- The formalization of the performance and experimental constraints is difficult. It depends on substantial information that are hard to extract.
- The proliferation of constraints generates many activities, thus increasing the combinatorial complexity of the state space to consider for resource allocation.
- The energy model is a core feature of PANORAMA since it is a major criterion for resource allocation selection. Therefore, accurate energy consumption model will have to be developed to manage different robotic systems.
- The localization performance is assumed sufficient in our study. So from the energy viewpoint, the proposed strategy to ensure a good localization is to locally use the most energy consuming sensors while verifying that globally, at the mission level, the allocated energy reserve is not totally consumed. This strategy is not entirely satisfactory.

We are currently developing a strategy with a deeper analysis of localization axis performance. Being sure of the performance of the chosen localization method enables us to consider a resource allocation solution minimizing the energy consumption and then maximising the energy margin. We are also interested in comparing our approach with other multi-constraint optimization methods, as for example constraint programming methods. Finally the proposed methodology could be extended to more complex missions. For example, PANORAMA could be helpful to enhance the robustness of autonomous exploration mission in an unknown environment. Our approach can not be used for the exploration itself as the map is not initially known, but it could guaranty that the way back of the robot is always possible. We plan to use this method for exploration missions of harsh environments like karstic networks [81]. It could also be interesting to investigate full autonomous systems that combine decision and behavioral autonomy in other contexts. Harsh environments, e.g. extraterrestrial or underwater environments, are clearly application domains where behavioral autonomy, resource management, fault tolerance and energy management are crucial.

GLOSSARY

Performance indicator (ind): A measurement of the efficiency of a system.

Performance inducer: Element that can be tuned to manage the performance. It influences the performance indicator values.

Performance axis $(axis)$: Point of view according to which the performance of the system is estimated.

Performance constraint: Partial order relation linking a performance to a goal.

Core-oriented performance constraint: The performance constraint must be satisfied throughout the mission.

End-oriented performance constraint: The performance constraint must only be verified at the end of the mission.

Mission (M): Sequential execution of Objectives.

Objective (O): High level robotic action performed using one or more robotic parallel tasks.

Activity (A): Part of a mission realized under constant constraints.

Task Implementation Option (TIO): Chosen set of hardware and software resources implementing a robotic task.

Implementation Alternative (IA): Selection of a single TIO for each robotic task involved in the related activity.

Mission Scenario (MS): Sequence of the activities of a mission.

Resources Allocation Solution (RAS): Projected plan for the allocation of resources for each mission activity.

Performance Margin (M_{axis}^{ind}): Difference between the final estimated or actual performance value and the performance goal.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s10846-024-02058-7.

Acknowledgements We acknowledge the University of Montpellier who funds two PhD grants supporting this research work

Author Contributions All authors contributed to the study conception and design. This study is a part of the results obtained during the PhD Thesis of L. Jaiem and P. Lambert. D. Crestani and L. Lapierre were the PhD Supervisor and Co-Supervisor of this research work. K. Godary-Dejean was also involved in this research study. The first draft of the manuscript was written by P. Lambert and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding This work was supported by two thesis grants from University of Montpellier (France).

Code or data availability The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Declarations

Ethics approval Not applicable

Consent to participate Not applicable

Consent for publication Not applicable

Conflicts of interest/Competing interests All the authors declare they have no financial interests.

References

1. Steinbauer, G.: A Survey about Faults of Robots Used in RoboCup. In: Chen, X., Stone, P., Sucar, L.E., van der Zant, T. (eds.) RoboCup 2012: Robot Soccer World Cup XVI, pp. 344–355. Springer, Berlin, Heidelberg (2013)
2. Doran, M., Sterritt, R., Wilkie, G.: Autonomic architecture for fault handling in mobile robots. Innovations Syst. Softw. Eng. **16**, 263–288 (2020). https://doi.org/10.1007/s11334-020-00361-8
3. Hereau, A., Godary-Dejean, K., Guiochet, J., Crestani, D.: A fault tolerant control architecture based on fault trees for an underwater robot executing transect missions. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 2127–2133 (2021). https://doi.org/10.1109/ICRA48506.2021.9561735
4. Lazarova-Molnar, S., Mohamed, N.: In: Risco Martín, J.L.,Mittal, S., Ören, T. (eds.) Reliability Analysis of Cyber Physical Systems, pp. 385–405. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-51909-4_15
5. Truszkowski, W., Hallock, H., Rouff, C., Karlin, J., Rash, J., Hinchey, M., Sterritt, R.: Autonomous and Autonomic Systems: with Applications to NASA Intelligent Spacecraft Operations and Exploration Systems. Springer, Berlin, Heidelberg (2009)
6. Wong, T., Wagner, M., Treude, C.: Self-adaptive systems: A systematic literature review across categories and domains. arXiv:2101.00125 (2021)
7. Brooks, S., Roy, R.: An overview of self-engineering systems. J. Eng. Des. **32**(8), 397–447 (2021). https://doi.org/10.1080/09544828.2021.1914323
8. Bradley, J.M., Atkins, E.M.: Optimization and control of cyber-physical vehicle systems. Sensors. **15**(9), 23020–23049 (2015)
9. Keipour, A., Mousaei, M., Scherer, S.: Alfa: A dataset for uav fault and anomaly detection. The International Journal of Robotics Research. **40**, 515–520 (2021). https://doi.org/10.1177/0278364920966642
10. Cabahug, J., Eslamiat, H.: Failure detection in quadcopteruavs using k-means clustering. Sensors. **22**(16), 387–406 (2022). https://doi.org/10.3390/s22166037
11. Ferdowsi, H., Cai, J., Jagannathan, S.: Actuator and sensor fault detection and failure prediction for systems with multi dimensional nonlinear partial differential equations. Int. J. Control Autom. Syst. **20**, 789–802 (2022). https://doi.org/10.1007/s12555-019-0622-3
12. Abci, B., El Badaoui El Najjar, M., Cocquempot, V.e.a.: An informational approach for sensor and actuator fault diagnosis for autonomous mobile robots. Journal of Intelligent & Robotic Systems. **99**, 387–406 (2020). https://doi.org/10.1007/s10846-019-01099-7
13. Schnell, T., Bott, K., Puck, L., Buettner, T., Roennau, A., Dillmann, R.: Robigan: A bidirectional wasserstein gan approach for online robot fault diagnosis via internal anomaly detection. In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4332–4337 (2022). https://doi.org/10.1109/IROS47612.2022.9982240
14. Chatzilygeroudis, K., Vassiliades, V., Mouret, J.-B.: Reset free Trial-and-Error Learning for Robot Damage Recovery. Robot. Auton. Syst. **100**, 236–250 (2018). https://doi.org/10.1016/j.robot.2017.11.010
15. Wu, R., Kortik, S., Santos, C.H.: Automated behavior tree error recovery framework for robotic systems. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 6898–6904 (2021). https://doi.org/10.1109/ICRA48506.2021.9561002
16. Liao, Y., Yeaser, A., Yang, B., Tung, J., Hashemi, E.: Unsupervised fault detection and recovery for intelligent robotic rollators. Robot. Auton. Syst. **146**, 103876 (2021). https://doi.org/10.1016/j.robot.2021.103876
17. Lv, T., Zhou, J., Wang, Y., Gong, W., Zhang, M.: Sliding mode based fault tolerant control for autonomous underwater vehicle. Ocean Eng. **216**, 107855 (2020). https://doi.org/10.1016/j.oceaneng.2020.107855
18. Sohege, Y., Quinones-Grueiro, M., Provan, G.: A novel hybrid approach for fault-tolerant control of uavs based on robust rein-

forcement learning. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 10719–10725 (2021). https://doi.org/10.1109/ICRA48506.2021.9562097
19. Nava, G., Pucci, D.: Failure detection and fault tolerant control of a jet-powered flying humanoid robot. In: 2023 IEEE International Conference on Robotics and Automation (ICRA), pp. 12737-12743 (2023). https://doi.org/10.1109/ICRA48891.2023.10160615
20. Diehl, M., Ramirez-Amaro, K.: A causal-based approach to explain, predict and prevent failures in robotic tasks. Robot. Auton. Syst. **162**, 104376 (2023). https://doi.org/10.1016/j.robot.2023.104376
21. Xu, K., Chen, R., Zhao, S., Li, Z., Yu, H., Chen, C., Wang, Y., Xiong, R.: Failure-aware policy learning for self-assessable robotics tasks. In: 2023 IEEE International Conference on Robotics and Automation (ICRA), pp. 9544–9550 (2023). https://doi.org/10.1109/ICRA48891.2023.10160889
22. Guiochet, J., Machin, M., Waeselynck, H.: Safety-critical advanced robots: A survey. Robot. Auton. Syst. **94**, 43–52 (2017). https://doi.org/10.1016/j.robot.2017.04.004
23. Tomasov, M., Kajanova, M., Bracinik, P., Motyka, D.: Overview of battery models for sustainable power and transport applications. Transportation Research Procedia. **40**, 548–555 (2019). https://doi.org/10.1016/j.trpro.2019.07.079. TRANSCOM 2019 13th International Scientific Conference on Sustainable, Modern and Safe Transport
24. Boukoberine, M.N., Zhou, Z., Benbouzid, M.: A critical review on unmanned aerial vehicles power supply and energy management: Solutions, strategies, and prospects. Appl. Energy **255**, 113823 (2019). https://doi.org/10.1016/j.apenergy.2019.113823
25. Farooq, M.U., Eizad, A., Bae, H.-K.: Power solutions for autonomous mobile robots: A survey. Robot. Auton. Syst. **159**, 104285 (2023). https://doi.org/10.1016/j.robot.2022.104285
26. Tomy, M., Lacerda, B., Hawes, N., Wyatt, J.L.: Battery charge scheduling in long-life autonomous mobile robots via multi-objective decision making under uncertainty. Robot. Auton. Syst. **133**, 103629 (2020). https://doi.org/10.1016/j.robot.2020.103629
27. Fouad, H., Beltrame, G.: Energy autonomy for resource-constrained multi robot missions. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 7006–7013 (2020). https://doi.org/10.1109/IROS45743.2020.9341455
28. Jain, K.P., Mueller, M.W.: Flying batteries: In-flight battery switching to increase multirotor flight time. In: 2020 IEEE International Conference on Robotics and Automation (ICRA), pp. 3510–3516 (2020). https://doi.org/10.1109/ICRA40945.2020.9197580
29. Jain, K.P., Tang, J., Sreenath, K., Mueller, M.W.: Staging energy sources to extend flight time of a multirotor uav. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1132–1139 (2020). https://doi.org/10.1109/IROS45743.2020.9341804
30. Saviolo, A., Mao, J., B, R.B.T.M., Radhakrishnan, V., Loianno, G.: Autocharge: Autonomous charging for perpetual quadrotor missions. In: 2023 IEEE International Conference on Robotics and Automation (ICRA), pp. 5400–5406 (2023). https://doi.org/10.1109/ICRA48891.2023.10161503
31. Nimmagadda, M.R., Dattawadkar, S., Muthukumar, S., Honkote, V.: Adaptive directional path planner for real-time, energy-efficient, robust navigation of mobile robots. In: 2020 IEEE International Conference on Robotics and Automation (ICRA), pp. 455–461 (2020). https://doi.org/10.1109/ICRA40945.2020.9197417
32. Zhang, H., Zhang, Y., Liu, C., Zhang, Z.: Energy efficient path planning for autonomous ground vehicles with ackermann steering. Robot. Auton. Syst. **162**, 104366 (2023). https://doi.org/10.1016/j.robot.2023.104366
33. Effati, M., Skonieczny, K., Freiman, T., Balkcom, D.J.: An equivalent time-optimal problem to find energy-optimal paths for skid-steer rovers. In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 13341–13346 (2022).https://doi.org/10.1109/IROS47612.2022.9982069
34. Fiset, J.-S., Effati, M., Skonieczny, K.: Power and energy consumption of skid-steer rovers turning on loose soil. Journal of Field Robotics 40(2), 193-214 (2023) https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.22123. https://doi.org/10.1002/rob.22123
35. D. Wallace, N., Kong, H., J. Hill, A., Sukkarieh, S.: Energy aware mission planning for wmrs on uneven terrains. IFAC-PapersOnLine 52(30), 149–154 (2019). https://doi.org/10.1016/j.ifacol.2019.12.513. 6th IFAC Conference on Sensing, Control and Automation Technologies for Agriculture AGRICONTROL 2019
36. Visca, M., Bouton, A., Powell, R., Gao, Y., Fallah, S.: Conv1d energy-aware path planner for mobile robots in unstructured environments. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 2279–2285 (2021). https://doi.org/10.1109/ICRA48506.2021.9560771
37. Quann, M., Ojeda, L., Smith, W., Rizzo, D., Castanier, M., Barton, K.: Off-road ground robot path energy cost prediction through probabilistic spatial mapping. Journal of Field Robotics. **37**(3), 421–439 (2020). https://doi.org/10.1002/rob.21927
38. Jambotkar, S., Guo, L., Jia, Y.: Adaptive optimization of autonomous vehicle computational resources for performance and energy improvement. In: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 7594–7600 (2021). https://doi.org/10.1109/IROS51168.2021.9635828
39. Liu, L., Zhong, R., Willcock, A., Fisher, N., Shi, W.: An open approach to energy-efficient autonomous mobile robots. In: 2023 IEEE International Conference on Robotics and Automation (ICRA), pp. 11569–11575 (2023). https://doi.org/10.1109/ICRA48891.2023.10161110
40. Singh, A.K., Dziurzanski, P., Mendis, H.R., Indrusiak, L.S.: A survey and comparative study of hard and soft real-time dynamic resource allocation strategies for multi-/many-core systems. ACM Comput. Surv. **50**(2), 24–12440 (2017). https://doi.org/10.1145/3057267
41. Saifullah, A., Fahmida, S., Modekurthy, V.P., Fisher, N., Guo, Z.: Cpu energy-aware parallel real-time scheduling. Leibniz international proceedings in informatics. **165** (2020). https://doi.org/10.4230/LIPIcs.ECRTS.2020.2
42. Sabyasachi, A.S., Muppala, J.K.: Cost-effective and energy-aware resource allocation in cloud data centers. Electronics. **11**(21) (2022). https://doi.org/10.3390/electronics11213639
43. Arias, M., Saavedra, R., Marques Samary, M., Munoz-Gama, J., Sepulveda, M.: Human resource allocation in business process management and process mining: A systematic mapping study. Manag. Decis. **56**, 376–405 (2018). https://doi.org/10.1108/MD-05-2017-0476
44. Chiang, H.Y., Lin, B.M.T.: A decision model for human resource allocation in project management of software development. IEEE Access. **8**, 38073–38081 (2020). https://doi.org/10.1109/ACCESS.2020.2975829
45. Renna, P.: Allocation improvement policies to reduce process time based on workload evaluation in job shop manufacturing systems. Int. J. Ind. Eng. Comput. **8**, 373–384 (2017). https://doi.org/10.5267/j.ijiec.2016.12.001
46. Su, X., Dong, W., Lu, J., Chen, C., Ji, W.: Dynamic allocation of manufacturing resources in iot job shop considering machine state transfer and carbon emission. Sustainability. **14**(23) (2022). https://doi.org/10.3390/su142316194
47. Seenu, N., Kuppan Chetty, R.M., Ramya, M.M., Mukund, N.J.: Review on state-of-the-art dynamic task allocation strategies for multiple-robot systems. Industrial Robot: the international journal of robotics research and application. **47**(6), 929–942 (2020). https://doi.org/10.1108/IR-04-2020-0073
48. Müller, M., Müller, T., Talkhestani, B.A., Marks, P., Jazdi, N., Weyrich, M.: Industrial autonomous systems: a survey on defi-

nitions, characteristics and abilities. at-Automatisierungstechnik. **69**(1), 3–13 (2021). https://doi.org/10.1515/auto-2020-0131
49. Bourguignon, A.: Peut-on définir la performance ? Technical report (sep 1995)
50. Websites, I.: Technical commitee for Performance Evaluation & Benchmarking of Robotic and Automation Systems. https://www.ieee-ras.org/performance-evaluation] (2023)
51. Lamarre, O., Limoyo, O., Marić, F., Kelly, J.: The canadian planetary emulation terrain energy-aware rover navigation dataset. The International Journal of Robotics Research. **39**(6), 641–650 (2020). https://doi.org/10.1177/0278364920908922
52. Nazarczuk, M., Mikolajczyk, K.: Shop-vrb: A visual reasoning benchmark for object perception. In: 2020 IEEE International Conference on Robotics and Automation (ICRA), pp. 6898–6904 (2020). https://doi.org/10.1109/ICRA40945.2020.9197332
53. Rogers, J.G., Gregory, J.M., Fink, J., Stump, E.: Test your slam! the subt-tunnel dataset and metric for mapping. In: 2020 IEEE International Conference on Robotics and Automation (ICRA), pp. 955–961 (2020). https://doi.org/10.1109/ICRA40945.2020.9197156
54. Wen, J., Zhang, X., Bi, Q., Pan, Z., Feng, Y., Yuan, J., Fang, Y.: Mrpb 1.0: A unified benchmark for the evaluation of mobile robot local planning approaches. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 8238–8244 (2021). https://doi.org/10.1109/ICRA48506.2021.9561901
55. Xu, Z., Liu, B., Xiao, X., Nair, A., Stone, P.: Benchmarking reinforcement learning techniques for autonomous navigation. In: 2023 IEEE International Conference on Robotics and Automation (ICRA), pp. 9224–9230 (2023). https://doi.org/10.1109/ICRA48891.2023.10160583
56. ANSI/RIA R15.05: Industrial Robots and Robots Systems-Path related and Dynamic Performance Characteristics-Evaluation. http://standards.globalspec.com/std/307040/ansi-ria-r15-05-2 (1992)
57. ISO Certification-Manipulating Industrial Robots-Performance Criteria and Related Test Methods, (2015). http://www.iso.org/iso/fr/catalogue_detail.htm?csnumber=22244
58. R15.08-1-2020, A.: Industrial Mobile Robots-Safety Requirements-Part 1: Requirements for the Industrial Mobile Robot. https://webstore.ansi.org/standards/ria/ansiriar15082020 (2020)
59. Robinson, N., Tidd, B., Campbell, D., Kulić, D., Corke, P.: Robotic vision for human-robot interaction and collaboration: A survey and systematic review. ACM Transactions on Human-Robot Interaction. **12**(1), 1–66 (2023). https://doi.org/10.1145/3570731
60. Kastner, L., Bhuiyan, T., Le, T.A., Treis, E., Cox, J., Meinardus, B., Kmiecik, J., Carstens, R., Pichel, D., Fatloun, B., Khorsandi, N., Lambrecht, J.: Arena-bench: A benchmarking suite for obstacle avoidance approaches in highly dynamic environments. IEEE Robotics and Automation Letters. **7**(4), 9477–9484 (2022). https://doi.org/10.1109/lra.2022.3190086
61. Zarrabi, N., Fesharakifard, R., Menhaj, M.B.: Robot localization performance using different slam approaches in a homogeneous indoor environment. In: 2019 7th International Conference on Robotics and Mechatronics (ICRoM), pp. 338–344 (2019). https://doi.org/10.1109/ICRoM48714.2019.9071902
62. Wang, J., Li, C., Li, B., Pang, C., Fang, Z.: High-precision and robust localization system for mobile robots in complex and large-scale indoor scenes. Int. J. Adv. Rob. Syst. **18**(5), 17298814211047690 (2021). https://doi.org/10.1177/17298814211047690
63. Bouman, A., Ott, J., Kim, S.-K., Chen, K., Kochenderfer, M.J., Lopez, B., Agha-mohammadi, A.-a., Burdick, J.: Adaptive coverage path planning for efficient exploration of unknown environments. In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 11916–11923 (2022). https://doi.org/10.1109/IROS47612.2022.9982287
64. Li, R., Zhou, C., Dou, Q., Hu, B.: Complete coverage path planning and performance factor analysis for autonomous bulldozer. Journal of Field Robotics. **39**(7), 1012–1032 (2022). https://doi.org/10.1002/rob.22085
65. Ceballos, N.D.M., Valencia, J.A., Ospina, N.L.: Quantitative performance metrics for mobile robots navigation. In: Mobile Robots Navigation. IntechOpen (2010)
66. Azevedo, C., Lacerda, B., Hawes, N., Lima, P.: Long-run multi robot planning under uncertain action durations for persistent tasks. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4323–4328 (2020).https://doi.org/10.1109/IROS45743.2020.9340901
67. Peltzer, O., Bouman, A., Kim, S.-K., Senanayake, R., Ott, J., Delecki, H., Sobue, M., Kochenderfer, M.J., Schwager, M., Burdick, J., Aghamohammadi, A.-a.: Fig-op: Exploring large-scale unknown environments on a fixed time budget. In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 8754–8761 (2022). https://doi.org/10.1109/IROS47612.2022.9981271
68. Lyons, D.M., Arkin, R.C., Nirmal, P., Jiang, S.: Designing autonomous robot missions with performance guarantees. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2583–2590 (2012). https://doi.org/10.1109/IROS.2012.6385952
69. Biggs, B., Stilwell, D.J., McMahon, J.: Extended performance guarantees for receding horizon search with terminal cost. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 6741–6748 (2020). https://doi.org/10.1109/IROS45743.2020.9341582
70. Park, J., Kim, J., Jang, I., Kim, H.J.: Efficient multi-agent trajectory planning with feasibility guarantee using relative bernstein polynomial. In: 2020 IEEE International Conference on Robotics and Automation (ICRA), pp. 434–440 (2020).https://doi.org/10.1109/ICRA40945.2020.9197162
71. Quinton, F., Grand, C., Lesire, C.: Market approaches to the multi-robot task allocation problem: a survey. Journal of Intelligent & Robotic Systems. **107**(29), 1012–1032 (2023). https://doi.org/10.1007/s10846-022-01803-0
72. Brooks, R.: A robust layered control system for a mobile robot. IEEE Journal on Robotics and Automation. **2**(1), 14–23 (1986). https://doi.org/10.1109/JRA.1986.1087032
73. Alami, R., Chatila, R., Fleury, S., Ghallab, M., Ingrand, F.: An Architecture for Autonomy. The International Journal of Robotics Research. **17**(4), 315–337 (1998). https://doi.org/10.1177/027836499801700402
74. Marler, R.T., Arora, J.S.: Survey of multi-objective optimization methods for engineering. Struct. Multidiscip. Optim. **26**, 369–395 (2004)
75. Stewart, R.H., Palmer, T.S., DuPont, B.: A survey of multi-objective optimization methods and their applications for nuclear scientists and engineers. **138**, 103830 (2021). https://doi.org/10.1016/j.pnucene.2021.103830
76. Lapierre, L., Zapata, R.: A guaranteed obstacle avoidance guidance system. Auton. Robot. **32**(3), 177–187 (2012). https://doi.org/10.1007/s10514-011-9269-5
77. Robots and robotic devices-Safety requirements for industrial robots-Part 1: Robots. Standard, International Organization for Standardization, Geneva, CH (July 2011)
78. Jaiem, L., Druon, S., Lapierre, L., Crestani, D.: A Step Toward Mobile Robots Autonomy: Energy Estimation Models. In: Alboul, L., Damian, D., Aitken, J.M. (eds.) Towards Autonomous Robotic Systems, pp. 177–188. Springer, Cham (2016)
79. Jaiem, L., Crestani, D., Lapierre, L., Druon, S.: Energy Consumption of Control Schemes for the Pioneer 3DX Mobile Robot:

Models and Evaluation. Journal of Intelligent & Robotic Systems. **102**(1), 23 (2021). https://doi.org/10.1007/s10846-021-01374-6
80. Bennour, M., Crestani, D., Crespo, O., Prunet, F.: Computer-aided decision for human task allocation with mono-and multi-performance evaluation. Int. J. Prod. Res. **43**(21), 4559–4588 (2005). https://doi.org/10.1080/00207540500124579
81. Explore Team Applicative Projects (2024). https://explore.lirmm.fr/?page_id=709#Aleyin

Philippe Lambert received his Ph.D in Robotics from the University of Montpellier, France, in 2021. He currently works as postdoctoral researcher at LIRMM in the EXPLORE team on autonomous mission planning for mobile robots in open environnement. His areas of interest is the development of autonomous robotic missions with performance management.

Karen Godary-Dejean received her Ph.D. degree in Computer Science from INSA of Lyon, France, in 2004. She joined the LIRMM (UMR 5506) in 2005 to be an Assistant Professor at the University of Montpellier. She currently belongs to the EXPLORE team of the Robotics Department. Her research interests are related to dependability for autonomous systems, including fault tolerance for robotics control architectures and formal modeling and validation for decision making process. She is interested in critical application domains (automotive industry, aeronautics, medical systems) and currently focuses on underwater mobile robotics (AUV).

Lionel Lapierre received his Ph.D. degree in Robotics, from the University of Montpellier, France, in 1999, and his HDR in 2015. He joined the team of Professor A. Pascoal within the European project FreeSub for three years. Since 2003, he has been with the Underwater Robotics Division, Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (LIRMM), Montpellier, France. He is now leading the project on 'Robotics for Karst Exploration'. In September 2023 he joined the Robex Team of LabSticc, ENSTA Bretagne, Brest, France.

Lotfi Jaiem is a mechatronic engineer from ENIS (Sousse - Tunisia). He received his Ph.D. degree in Robotics, from the University of Montpellier, France, in 2016. From 2017 he works as robotic engineer in the IADYS company specialized in the development of robotics and IA for preserving the water.

Didier Crestani received his Ph.D. degree in computer science from the University of Montpellier, France, in 1999 and is Professor in the French University of Montpellier from 2010. He belongs to the robotics Department of the LIRMM laboratory. His area of interest is the development of fault tolerant autonomous robotic missions with guaranteed performance.