A Reconfiguration Method for Muti-Robot Monitoring Patrols

Sara Hsaini, Rabah Ammour, Leonardo Brenner, My El Hassan Charaf, Isabel Demongodin, Dimitri Lefebvre

A reconfiguration method for muti-robot monitoring patrols

Sara Hsaini[1*], Rabah Ammour[1], Leonardo Brenner[1], My El Hassan Charaf[2], Isabel Demongodin[1] and Dimitri Lefebvre[3]

[1]Aix Marseille University, CNRS, LIS, Marseille, France.
[2]LaRi, Ibn Tofail University, Morocco.
[3]GREAH, University le Havre Normandie, France.

*Corresponding author(s). E-mail(s): sara.hsaini@univ-amu.fr

Abstract

This paper addresses the problem of multi-robot task allocation and trajectory planning in industrial environments. The objective is to optimize the overall cost of robot surveillance patrols in a dynamic high-risk environment. In this context, a hybrid beam search based approach is proposed to plan the patrol trajectories iteratively to accommodate environmental changes under some functional and operational constraints. Moreover, a real-time based system is introduced for remotely monitoring dynamic surveillance missions with automated mobile agents. Finally, a case study is detailed to show the efficiency of our approach in the case of the industrial port area of Fos-sur-Mer city in France.

Keywords: Automated guided vehicles, monitoring patrol, optimization, tasks allocation, trajectory planning

1 Introduction

As of recent years, multi-robot systems have become a promising research area for their advantages in handling complex problems in different sectors. These systems comprise a set of cooperative mobile agents that collaborate to accomplish specific tasks [1]. Accordingly, a multi-robot task allocation (MRTA) has been defined as the assignment of different tasks to the appropriate robots in

order to achieve the overall objectives of the system [2]. Typically, these methods are widely used in industrial areas in particular to handle problems, such as routing patrols surveillance. In this case, tasks are spatially distributed and the problem also includes trajectory planning aspects. There may be also some other issues related to the number of robots, the number of tasks within the system, and the time required to execute each task.

Generally, MRTA problems can be classified according to three aspects as quoted in [3]. An illustrative classification is introduced in Figure 1. The first axis represents the number of tasks that can be executed by a robot. The second one represents the number of robots that can perform a task. The third axis represents the assignment time types: instantaneous or time-extended. In the case of an instantaneous assignment, the mission information (concerning robots and environment) is limited and it is not possible to make an assessment for future assignments. However, the time-extended assignment means that future tasks are known, and an assignment can be calculated for these tasks. Besides, there are two approaches to address MRTA problems: a static approach where the environment parameters are known in advance, and a dynamic approach where new tasks may occur during the execution [4]. Thus, a task re-allocation operation is necessary to accommodate these changes.

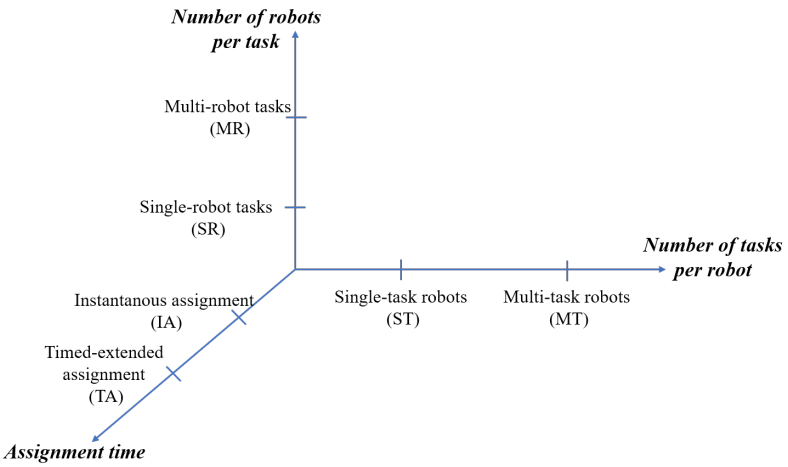


Fig. 1: MTRA problems classification.

In this paper, we deal with the multi-robot routing problem (MRR) which is a specific class of multi-robot task allocation problems. The MRR problem aims to find the optimal assignment of targets to a group of heterogeneous mobile robots and determine the trajectories of robots to visit all their assigned targets along routes that optimize an objective function [5]. Coordination in multi-robot systems represents a challenging issue, especially in the industrial

area where robotic fleets must be able to cooperate in order to accomplish their mission correctly. Therefore, software systems can provide an effective solution to remotely manage the robotic fleets by sharing information between robots and thus ensuring their cooperation, especially in dynamic environments where new events can occur during the robots' mission.

In this paper, we present a real-time-based system to remotely resolve the MRR problem in a changing environment. In this setting, the paper has three main contributions.

The first contribution is to extend our preliminary works [6–8] that were initially developed in a static context and proposes a method for handling MRR in real-time scenarios using dynamic settings. The main objective is to deal with dynamic events by reassigning tasks to robots iteratively over time according to changes during the mission. As such, we aim to ensure task reassignments whenever new events occur such as the appearance of arising tasks, the deletion of tasks or the failure of a robot.

The second contribution is to refine the cost function in order to take into account the measurement costs. Thus, to calculate the optimal solution, we consider not only the travel cost between sites as described in [6–8] but also the cost required to perform each monitoring measurement.

Meanwhile, the reassignment program and the robots require real-time coordination in order to handle various events that occur during the mission. The third contribution is to detail software programs to support real-time reassignment and coordination between the robots. For this purpose, we iterate an heuristic method based on Hybrid filtred beam search and Dijkstra algorithms for the reassignment of tasks to heterogeneous mobile robots. In this case, the monitoring patrols are timely reconfigured according to any change that may occur in the environment.

As a final step, a MRR case study for monitoring patrols, in the port area in Fos-sur-Mer city in France, is considered to elucidate our approach in this regard.

The rest of this paper is organized as follows: Section 2 gives some basic concepts related to MRR and dynamic task allocation problems, and the problem statement is discussed. We also present some related works. The improved cost function for MRR and the reconfiguration method are described in Sections 3 and 4, respectively. In sections 5 and 6, the real-time based system and simulation results are presented. Finally, Section 7 gives some conclusions and identifies future works.

2 Background

2.1 Multi Robot Routing

MRR is a particular class of multi-robot task allocation that aims to efficiently route a set of mobile robots to perform a set of tasks while optimizing an objective function. In this case, the robots must avoid obstacles such as

walls and other robots on their way to their assigned task locations.

Definition: MRR can be specified formally by [9]

- A set of robots $\mathcal{R} = \{r_1, r_2, \ldots, r_R\}$.
- A set of tasks $\mathcal{M} = \{m_1, m_2, \ldots, m_M\}$.
- A set of locations $\mathcal{P} = \{p_1, p_2, \ldots, p_P\}$.
- A travel cost $C(i, i')$ between locations p_i and $p_{i'}$.

In MRR, the cost function $C(i, i')$ refers to the travel time or distance to move from location p_i to location $p_{i'}$. The cost function is then equal to infinity in the case where there is no path between i and i'. The objective function to be optimized is the sum of cost functions over all locations to be visited by robots in order to perform the assigned tasks.

2.2 Dynamic task allocation

The mission specifications may change over time by adding or removing tasks or robots. Thus, robots must be able to adapt their behavior to these changes using a dynamic approach. Therefore, the aim of dynamic task allocation in multi-robot systems is to assign tasks to robots iteratively over time according to changes occurring in the environment and to ensure task reassignments whenever new events occur. We can distinguish three types of task allocation approaches as follows [10]:

- **Optimization-based approaches:** The aim is to find an optimal solution to the task allocation problem in order to maximize or minimize a specific criterion. The obtained solution is subjected to a set of constraints;
- **Market-based approaches:** An auctioneer robot announces the details of the tasks and solicits bids from other robots. Then, it assigns each task to the robot offering the lowest bid;
- **Behavior-based approaches:** Based on information gathered from sensors readings, actions, and mission status, as well as internal parameters, the team members determine which robot will perform a particular task, without explicit discussion among the robot team members.

In this paper, we focus in how optimization-based approaches can be used to dynamically assign tasks to robots.

2.3 Problem statement

To assign tasks to the appropriate robots and determine the trajectory of each robot in order to visit all task locations and ensure optimal results is commonly considered as a complex problem. This often involves heterogeneous robots equipped with different types of sensors that have to perform various tasks with varying requirements and constraints.

The problem becomes more complex in a dynamic environment, such as an industrial area. In this case, reassignment is to be performed iteratively over

time as new events may arise at different locations during the mission. The solution must take into account new events to be handled, the existing tasks, as well as the position of all robots before the execution of any reassignment. Additionally, the MMR problem is classified as NP-hard since no optimal solution can be found using a polynomial-time algorithm. Accordingly, heuristic algorithms are considered to be an effective means for obtaining an acceptable solution in this case.

The multi-robot routing problem addressed in this paper is classified as ST-MR-TA (see Figure 1), that is, one task can be performed by several robots, but each robot can only manage to accomplish one task at a time. In this context, various constraints are considered, namely: (i) the energy autonomy of each robot, (ii) the time required to accomplish each task, (iii) the maximum number of sensors each robot can carry, and (iv) the time between two consecutive reassignments.

Finally, the successful execution of a mission in a real-world context requires coordination between the assignment process and the robots in order to plan and control the performance of tasks in real time. Thus, software can serve as a centralized planner that remotely manages the execution of the whole mission. Our operational objective is to provide a real-time processing of dynamic environment data. The mobile robots will then use the task assignment results as well as the optimal trajectories to accomplish their missions.

2.4 Related works

Due to its complex structure and hazardous products, monitoring an industrial area remains a difficult and risky activity, resulting in significant damage if any details are overlooked [11, 12].

In this context, the use of multi-robot systems (MRS) for inspecting such areas is increasing as these systems are more flexible, less expensive, and capable of performing a variety of tasks [13].

Many approaches have been presented in the literature to address the multi-robot patrolling and task allocation problems in complex domains such as surveillance industrial areas.To provide a clearer understanding of these approaches, we have categorized them into two distinct categories : heuristic-based approaches and dynamic MRR approaches.

2.4.1 Heuristic-based approaches

Heuristic approaches have been proven to be effective methods for dealing with multi robots routing problems. A study conducted by [14] suggests a centralized heuristic approach to handle inspection problems in large industrial areas. The proposed approach combines the A* with genetic algorithms to solve task allocation and path planning problems by minimizing travel distance and avoiding collision problems [15]. Likewise, the authors in [16] propose SAGL, a new centralized algorithm for solving complex multi-robot

routing problems with a minimum makespan by applying linear programming duality for the Steiner Forest Problem. Another work [17] addresses a new multi-robot routing problem known as Multi-Robot Scheduling Problem with Blocking and Enabling Constraints (MRSBE), where robots must visit all their assigned task locations by minimizing overall makespan and respecting two kinds of constraints: enabling and collision constraints. To deal with this kind of MRR problem, the authors propose a new heuristic local search approach based on Value-Biased Stochastic Sampling and Depth First Search algorithms. The authors in [18], introduce another heuristic based approach to resolve the Generalized Team Orienteering Problem (GTOP), which is a combinatorial optimization problem for routing heterogeneous robots in exploration tasks. As a solution, they propose to use Self Organizing Map, an algorithm that perfectly meets the scalability challenge of the MRR problem.

In addition, the MRR problem has also been addressed using other combinatorial optimization methods, including metaheuristic [19, 20], linear programming [21, 22] and dynamic programming [23]. In this context, the study [24] demonstrates that the heuristic methods outperform other methods, such as deterministic annealing algorithm or stochastic—simulated annealing, in terms of execution time to solve either smaller or bigger MRR problems.

2.4.2 Dynamic approaches

Dynamic task allocation has also received a great deal of attention recently and numerous studies have been conducted to address this issue. For example, the authors in [25] present different dynamic approaches to solve the multi-robot task allocation problem in the exploration and destruction domain, namely: the auction-based approach, the vacancy chain approach, and the deep Q-learning approach based on strategy-level selection. As well, they demonstrate the effectiveness of the proposed approaches to solve the MRTA problems through a simulation system based on Robot Operating System and Gazebo.

In [26], the authors present SCoBA, a hierarchical dynamic approach for solving MRTA problem. While SCoBA uses the multi-agent conflict resolution algorithm to approach the MRTA problem under two different constraints: task uncertainty and time constraints, the authors in [27] have been inspired by predatory behavior in social animals to propose a distributed dynamic allocation approach for non-holonic robots based upon a competitive mechanism. Furthermore, the authors in [28] present a fleet management system based on a dynamic approach to schedule and control the performance of a logistic mission by a set of mobile robots in hospital environment. Another work [29] suggests to deploy an assignment algorithm to manage a modern megastore dynamic environment. The aim is to minimize the total length of paths used by all the robots as well as the risks related to the human-robot interaction. In addition, the authors in [30] propose a cloud-based management system to remotely monitor and control the mission of a fleet of unmanned surface vehicles (USV).

Finally, this paper can be considered as a continuity of [6–8] where we propose a hybrid filtred beam search static approach to solve the multi-robot routing problem related to the monitoring patrols in an industrial area. Hence, we extend the approach by suggesting a real-time system to manage the execution of a set of surveillance tasks by a fleet of heterogeneous robots in a dynamic industrial environment. The system aims to reconfigure the execution of each mission according to any change that may occur in the environment.

3 An improved cost function for MRR

This section focuses on an improved cost function that takes into account not only the travelling cost of the robots but also the measurement cost of the sensors.

3.1 Problem description

We consider a two-dimensionnal environment represented by a rectangular mesh of size $N_X \times N_Y$ cells. The cells are identified by their addresses a_i, $i = 1, \ldots, N$ where $N = N_X \times N_Y$. Each cell (x, y) defines a spatial area where an agent can stay. The cells are assumed to be as large as necessary such that each cell can be visited by several agents at the same time. The environment may include several types of obstacles or one-way paths. Measurement tasks should be performed in some specific cells of the environment, named sites. The following notations are introduced. $\mathcal{P}$ stands for the set of sites p_i where tasks should be performed and $P =\mid \mathcal{P} \mid$ is the number of such sites. $\mathcal{R}$ stands for the set of robots r_j and $R =\mid \mathcal{R} \mid$. $\mathcal{M}$ stands for the set of measurement tasks m_k and $M =\mid \mathcal{M} \mid$. In addition, we refer to the set of measurement tasks in site p_i as to $\mathcal{M}(i)$. The set of tasks in different sites can be defined as follows:

$$Locations = (L(k,i)) \in \{0,1\}^{M \times P} \tag{1}$$

where each site p_i, $i = 1, \ldots, P$, corresponds to a cell where one or more measurements are required and m_k, $k = 1, \ldots, M$, are the different types of measurement tasks. The table *Locations* of size $M \times P$ specifies which measurements should be taken in the sites in $\mathcal{P}$, i.e.,

$$L(k,i) = \begin{cases} 1, \text{ if a measurement of type } m_k \text{ should be performed in site } p_i; \\ 0, \text{otherwise.} \end{cases}$$

The table *Locations* also includes the site p_1 where the agents start and end their patrol (there is no measurement to be taken at this position, thus $L(k,1) = 0$, $k = 1, ..., M$).

In this work, we consider that (i) each deplacement between two given sites p_i and $p_{i'}$ as an average cost $C_d(i, i')$ that is the energy required to move from p_i to $p_{i'}$ ($C_d(i, i')$ being equal to infinity when there is no possibility to reach $p_{i'}$ from p_i); (ii) each measurement m_k has a specific cost $C_m(k, i')$ which denotes the average amount of energy required to perform it. In addition, (iii) each agent r_j has two particular operating costs: a deplacement cost $C_d(j)$ that depends on the agent itself, in particular if the agent is an Automated Guided Vehicles (AGVs) or Unmanned Automated Vehicles (UAVs), and a measurement cost $C_m(j)$ that depends on both the agent and the sensors that it carries on. If no specific operating costs are considered then $C_d(j) = 1$ and $C_m(j) = 1$.

The tasks are performed by mobile agents which are initially positioned in the site p_1 (depot). Indeed, each agent r_j carries a specific subset of sensors that perform some measurements. Two agents may be equipped with identical or different sensors. It is assumed that each sensor performs a given measurement. The table *Robots* defines the set of sensors carried on by each robot r_j and in the next we refer to the set of measurement tasks that are effectively taken by the robot r_j in a site p_i as to $\mathcal{M}(j, i)$.

$$Robots = (R(k, j)) \in \{0, 1\}^{M \times R} \tag{2}$$

with :

$$R(k, j) = \begin{cases} 1, \text{if the robot } r_j \text{ has the sensor to handle measurement } m_k; \\ 0, \text{ otherwise.} \end{cases}$$

The purpose of this paper is to determine the optimal assignment of measurements to robots as well as the trajectory for each robot with the aim to minimize the total mission cost.

3.2 Mathematical modeling

To describe in a formal way the MRR problem addressed in this paper, we use binary integer programming in which all decision variables have to be equal to 0 or 1 and formulate the problem from an optimization perspective. The objective function (3) to be minimized, is the total mission cost, which can be defined as the energy required to perform all tasks of the mission (in a certain sense this cost is also an image of the total cumulative time needed by the agents). Due to autonomy limitation, there exist situations where several robots equipped with the same sensors are required to perform all the measurement tasks. To deal with such issue, let us introduce a partition within the fleet of robots: $\mathcal{R} = \mathcal{R}_1 \cup \ldots \cup \mathcal{R}_H$, where each subset $\mathcal{R}_h$, $h = 1, ..., H$, corresponds to a specific type of robots (all the robots within $\mathcal{R}_h$ being similar). In the case where solutions with different numbers of robots and the same value of the cost function are found, solutions with a small number of robots are preferred. The objective function can be expressed as follows

$$Cost = \sum_{\mathcal{R}_h \subseteq \mathcal{R}} \sum_{r_j \in \mathcal{R}_h} \sum_{p_i, p_{i'} \in \mathcal{P}} X^j_{i,i'} \times \left(C^j_{i,i'} + \sum_{m_k \in \mathcal{M}} Y^j_{k,i'} \times C^j_{i'}(k) \right) \quad (3)$$

- $C^j_{i,i'} = C_d(j) \times C_d(i,i')$ denotes the cost required to move from site p_i to site $p_{i'}$ for the robot r_j. It equals the product of the deplacement operating cost $C_d(j)$ of the robot r_j and $C_d(i,i')$ the average cost to move from site p_i to site $p_{i'}$;
- $C^j_{i'}(k) = C_m(j) \times C_m(k,i')$ refers to the cost required to perform the measurement m_k in site $p_{i'}$ for the robot r_j. It equals the product of the operating measurement cost $C_m(j)$ of the robot r_j and $C_m(k,i')$ the cost to perform the measurement m_k in $p_{i'}$;

- $X^j_{i,i'} = \begin{cases} 1, \text{if the robot } r_j \text{ moves from site } p_i \text{ to } p_{i'}; \\ 0, \text{otherwise;} \end{cases}$
- $Y^j_{k,i'} = \begin{cases} 1, \text{if the robot } r_j \text{ performs measurement } m_k \text{ in } p_{i'}; \\ 0, \text{otherwise.} \end{cases}$

This objective function has to be minimized under the following constraints.

- In each site $p_{i'}$, the task m_k must be performed by one and only one robot, i.e.,

$$\sum_{r_j \in \mathcal{R}} Y^j_{k,i'} \left(\sum_{p_i \in \mathcal{P}} X^j_{i,i'} \right) = L(k,i'), \qquad \forall m_k \in \mathcal{M}, p_{i'} \in \mathcal{P}. \quad (4)$$

- The robot r_j cannot be assigned to perform the task m_k unless it is equipped with the required sensor, i.e.,

$$Y^j_{k,i'} \leq R(k,j), \qquad \forall m_k \in \mathcal{M}, r_j \in \mathcal{R}, p_{i'} \in \mathcal{P}. \quad (5)$$

- The trajectory σ_{r_j} of the robot of type r_j includes the site p_1 (the depot);

$$\sum_{p_{i'} \in \mathcal{P}} X^j_{i',1} > 0, \qquad \forall r_j \in \mathcal{R}. \quad (6)$$

- Each robot r_j exits a given site p_i as many times as it enters p_i,

$$\sum_{p_{i'} \in \mathcal{P}} X^j_{i,i'} = \sum_{p_{i'} \in \mathcal{P}} X^j_{i',i}, \qquad \forall r_j \in \mathcal{R}, p_i \in \mathcal{P}. \quad (7)$$

- Each robot r_j should visit the sites where it has one or more measurement tasks to perform in a given order, defined by the variables U^j_i, $p_i \in \mathcal{P}(r_j)$ where $\mathcal{P}(r_j)$ represents the subset of sites visited by the robot $r_j \in \mathcal{R}$.

The constraint that eliminates subcircuits is given according to the Miller-Tucker-Zemlin formulation:

$$U_i^j + X_{i,i'}^j \leq U_{i'}^j + (\mid \mathcal{S} \mid -1) \times (1 - X_{i,i'}^j), \\ \forall p_i \in \mathcal{P}(r_j),\ \forall p_{i'} \in \mathcal{P}(r_j)/\{p_1\},\ \forall r_j \in \mathcal{R} \tag{8}$$

Consequently, according to equation (6) to (8), each robot r_j moves along a circuit that includes the depot.

- For each robot r_j, the cost required to perform the assigned tasks must not exceed the maximal energy of the robot $Emax^j$, i.e.,

$$\sum_{p_i, p_{i'} \in \mathcal{P}} X_{i,i'}^j \times \left(C_{i,i'}^j + \sum_{m_k \in \mathcal{M}} Y_{k,i'}^j \times C_{i'}^j(k) \right) \leq Emax^j, \qquad \forall r_j \in \mathcal{R}. \tag{9}$$

3.3 Optimization based on HFBS

Since the MRR problem discussed in this paper is a non-linear problem, we propose a heuristic approach to solve this problem. The proposed approach is divided into two steps. The first one calculates an initial assignment solution, while the second performs periodically a reallocation solution by taking into account the new events that have occured during the last period Δ_t'. We use two different algorithms to calculate the assignment solution: Dijkstra and Hybrid Filtered Beam Search (HFBS).

- Dijkstra algorithm is one of the most common routines for finding the path of smallest cost between two given nodes in directed or undirected graphs [31] and has been applied in various domains, in particular for transportation problems. In this work, we use Dijkstra to compute the trajectories of lowest cost $C_d(i, i')$ between any pair of sites $(p_i, p_{i'}), p_i, p_{i'} \in \mathcal{P}$.
- Hybrid Filtered Beam Search is a class of graph search algorithms in weighted graphs that is based on a heuristic function that evaluates the cost to reach a given final node (or a set of final nodes) that satisfies some specific termination conditions TC from a given initial node. This variant ensures diversification in the population of candidates by limiting the number of successors generated by the same parent. In this paper, we use HFBS to assign measurements to robots based on the low cost trajectories obtained by the Dijkstra algorithm. The HFBS uses both a global filter with parameter β_g, which limits the number of nodes at each level of the search tree and a local filter that keeps only the β_l best successors for each expanded node. Each node S of the graph will represent a candidate solution for the considered problem. This candidate solution is either task-complete(TC) i.e., (i) all tasks have been performed; (ii) all agents have returned to the site p_1. In such case, we write $TC(S) = 1$; otherwise S is non task-complete and we write $TC(S) = 0$.

Based on [8], a candidate solution S is defined by the sequences $Agent(S, r_j)$ of pairs formed by the successive sites s_i^j, $i = 1, ..., h$, to be visited by the agents of each type r_j and by the measurement tasks $\mathcal{M}(j, s_i^j)$ taken by the agents r_j at each visited site s_i^j.

$$S = \{Agent(S, r_j) = (s_1^j, \mathcal{M}(j, s_1^j))...(s_h^j, \mathcal{M}(j, s_h^j)), j = 1, \ldots, R\}. \quad (10)$$

$Agent(S, r_j)$ is composed by one or more trajectories. Typically, a trajectory starts and ends with the site p_1, but in the case of a reassignment solution, a trajectory can start with the actual positions of robots and end with the site p_1. Due to energy constraints, several trajectories may be required to conduct the patrol of the agents r_j. In fact, when an agent does not have enough energy to collect all measurements in the different sites, it returns to the site p_1 (to recharge its battery) and another agent is used to continue the patrol.

The HFBS uses an evaluation function f which represents the global cost of the node S. The function f is computed at each node S of the graph by $f(S) = g(S) + h(S)$ where $g(S)$ is the actual cost from initial node S_0 to S and $h(S)$ is an estimation of the cost from S to the nearest node that satisfies TC. To converge, the heuristic function $h(S)$ should under-estimate the actual cost. Equation (11) defines the function $g(S)$ used in this paper.

$$g(S) = \sum_{r_j \in \mathcal{R}} \left(\sum_{i=1...h-1} \left(C_{i,i+1}^j + \sum_{m_k \in \mathcal{M}(j, s_{i+1}^j)} C_{i+1}^j(k) \right) \right) \quad (11)$$

where:

- $C_{i,i+1}^j$ is the travel cost from s_i^j to s_{i+1}^j by the robot r_j;
- $C_{i+1}^j(k)$ is the cost to perform the measurement $m_k \in \mathcal{M}(j, s_{i+1}^j)$ by the robot r_j.

To calculate the estimation $h(S)$ let us first define $L(S)$ as the set of sites that still need a visit at S, $p(S)$ as the set of agent current positions at S , $L_1(S) = L(S) \cup \{p_1\}, L_p(S) = L(S) \cup \{p(S)\}$ and $M(S)$ as the set of measurements that still need to be carried out at S. $h(S)$ is formulated as follows:

$$h(S) = \max \left\{ \underbrace{(C^*(p(S), L_1(S)) + \sum_{p \in L(S)} C^*(p, L_1(S)))}_{h_1(S)}, \underbrace{\sum_{p' \in L_p(S)} C^*(L_p(S), p')}_{h_2(S)} \right\} + \underbrace{\sum_{r_j \in R} C^*(M(S), j)}_{h_3(S)} \quad (12)$$

where $C^*(p(S), L_1(S))$ is the minimal non-null cost from the current positions of the agents to the sites in $L_1(S)$, $C^*(p, L_1(S))$ is the minimal non-null

cost from the current position p of one of the agents to the sites in $L_1(S)$, $C^*(L_p(S), p')$ is the minimal non-null cost from the sites in the set $L_p(S)$ to the particular site p' and $C^*(M(S), j)$ is the minimal non-null cost to perform each measurement in $M(S)$ by the different agents in S.

4 Reconfiguration method

As stated above, we consider a dynamic environment where new changes can occur during the mission execution. Specifically, we focus on three categories of unexpected events: the occurrence of new tasks, the cancellation of tasks not currently being performed, and the failure of a robot.

In order to cope with dynamic events, e.g., robot failures or new task requirements, we apply a periodic reassignment approach which solves a static MRR problem corresponding to the current environment at some particular time stamps while optimizing the objective function defined in (1). Figure 2 provides the chart of the proposed reconfiguration method.

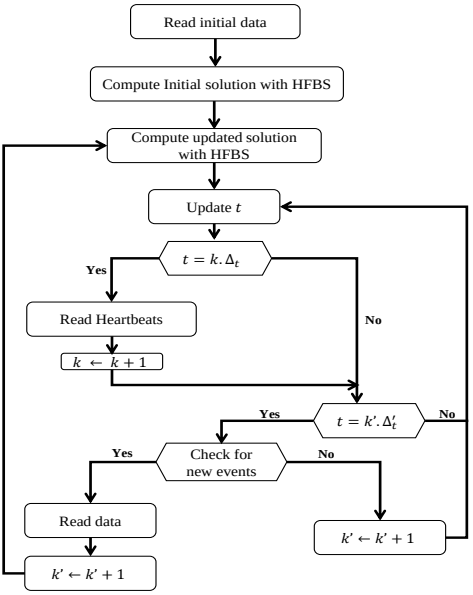


Fig. 2: HFBS based reconfiguration approach.

More precisely, robot heartbeats are received after each period of time Δ_t to confirm that the robot is still healthy. Otherwise, if a robot fails to send a heartbeat for a period of time, it will be considered to have failed and a fault event will be recorded in the database. The information about new tasks that are required by the operator during the mission (or tasks that are canceled before they are completed) is also stored in the database.

Afterwards, the system checks for new events in the database every period of time Δ_t' (we assume $\Delta_t < \Delta_t'$). If new events are found, the allocation algorithm is restarted with the new data and the updated solution is sent to the robots. When a robot fails, the algorithm will consider a new robot equipped with the same sensors and located in the depot to continue the mission.

Moreover, the positions of all the robots in the MRR problem are continuously updated during the execution of the mission. As a result, at every reassignment, each robot must start its trajectory from its actual position and returns to the depot point at the end of the mission, contrary to the initial assignment in which all robots start and end their trajectories at the same point (Figure 3).

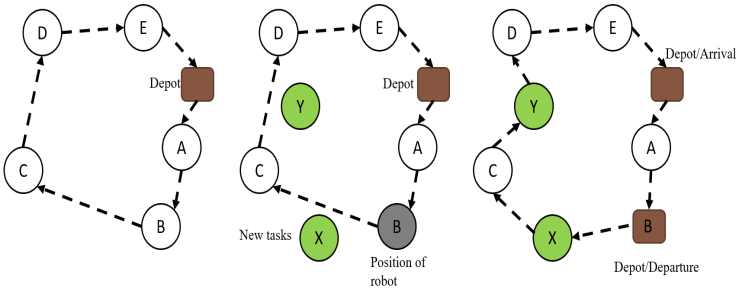


Fig. 3: Dynamic MRR approach.

To sum up, the approach re-executes, at each time $k.\Delta_t'$, an algorithm to solve the MRR problem by considering recent information such as the updated list of tasks and the current positions of robots. The time interval $k.\Delta_t'$ between two consecutive assignments is calculated in order to take into account new events in the dynamic surveillance mission as soon as possible while minimizing the cumulative time of the whole mission.

It is worth noting that the proposed cost function aims also to minimize the number of agents used to execute the whole mission. Thus, it is preferred to integrate all robots that are already devoted to executing different tasks in the reassignment rather than allocated a new robot to tackle the new event. Depending how the elementary costs $C_d(j), C_d(i,i')$ and $C_m(j), C_m(k,i')$ are defined, this method minimizes the energy consumption to execute the mission or the cumulative time to perform all the tasks (but it does not intend to minimize the duration of the mission). The algorithm below describes the reassignment process adopted in this paper. The algorithm input is a fixed time Δ_t' which represents the amount of time to be respected between two consecutive reassignments, and produces a set of updated trajectories for all the robots. In this context, the function *check_new_event*() checks if an unexpected event has occurred in the environment (line 2). If this function has found a new event

(line 3), then the algorithm reads all information about it, i.e., the type of event and site with the function *get_events*() (line 4). Indeed, the function *updates*() aims to update the information about the mission according to the new events, i.e., the number of robots used, the starting cell of each robot, that is not necessarily a site with one or more measurement to be taken, and the list of sites to be visited and measurements to be performed (line 5). Finally, the algorithm runs the reassignment using the function *reassignment*() to compute the new trajectories σ based on the updated information of the environment (line 6).

Algorithm 1 Reassignment

Input: Δ'_t interval time of reassignment
Output: σ the set of updated trajectories
foreach Δ'_t **do**
  $test =$ **check_new_event();**
  if $test == 1$ **then**
    $Events =$ **get_events();**
    updates(robots, initial_points, tasks_locations,events);
    $\sigma =$ **reassignment(robots, initial_points, tasks_locations);**
  end
end

5 Real time implementation

5.1 Architecture

This section describes our proposed real-time system for managing multi-robot routing. The aim is to remotely allocate tasks to robots in a dynamic environment and determine in real time the optimal route to be explored by each robot in order to perform all the assigned tasks.

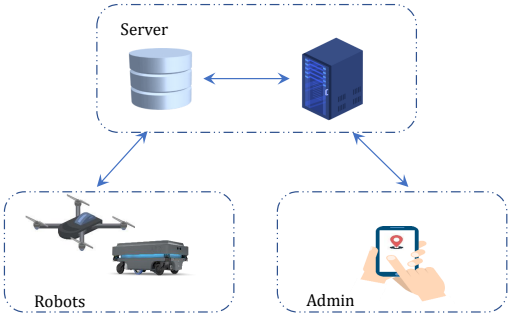


Fig. 4: Real Time-Based System for MRR.

As shown in Figure 4, the architecture comprises the following components.

- **The server** hosts a database that keeps records of the robots' data, information about the environment, and the results of dynamic task allocation and routing. The server also contains an internal management system to calculate the task allocation and routing for each robot based on two different algorithms: beam search and Dijkstra;
- **The Admin application** provides the ability to save in the initial database details about the environment as well as any new events that are created during the mission.
- **The robots** perform different tasks during the mission. We consider in this paper several types of robots being Automated Guided Vehicles (AGVs), Unmanned Automated Vehicles (UAVs) equipped with various types of sensors;

5.2 System process

The proposed system aims to allocate each task to the most appropriated robot and to determine the optimal routes of each robot to accomplish its assigned tasks at the lowest cost. The workflow of our system is reported in Figure 5:

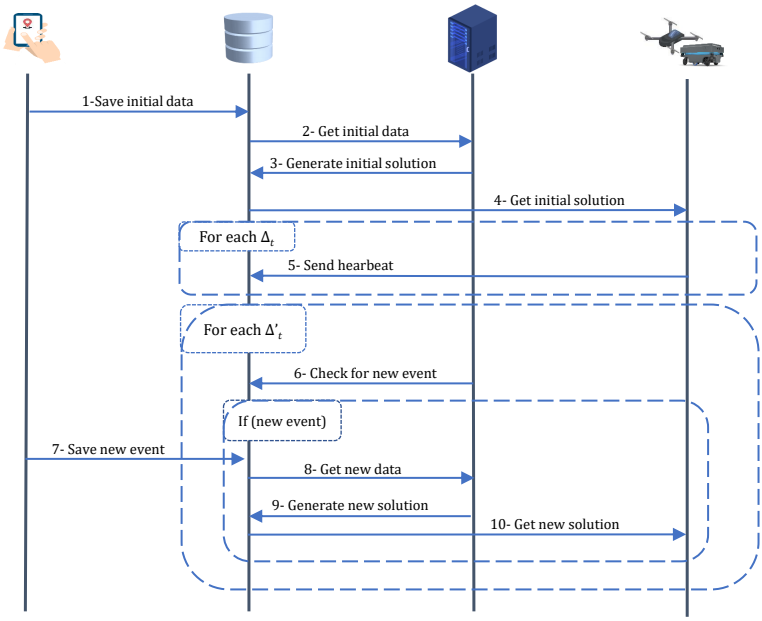


Fig. 5: Real Time-Based System for MRR.

- at the beginning, the administrator uploads the initial information, such as task types, task locations, depot locations, and task costs, in the database through the mobile application;
- the system reads initial data from the database and generates the initial assignment to solve the MRR problem;
- the robots get the routing paths as well as the task allocation results and start the mission;
- periodically, each Δ_t time units (TUs) each robot sends a signal (heartbeat) to the server, indicating that it is still alive and performing its assigned tasks;
- after having performed each task, the robot sends a confirmation to the system;
- the administrator uploads each new task in database through the mobile application;
- the computing system checks periodically, each Δ'_t TUs, for new events, i.e., if there is new information added by the administrator in the database or if a robot has failed:
  - when the system finds new events during the mission, it runs again the MRR program with the updated dataset;
  - the robots receive the updated routes with the tasks allocated and continue the mission according to the new results.

6 Case Study

The number of dangerous damages associated with hazardous industrial areas has received increased attention in recent years. Thus, monitoring activities have become increasingly relevant to prevent catastrophic events. In this case study, we demonstrate the configuration of the surveillance patrol for the industrial port area of Fos-sur-mer in France in order to validate the proposed dynamic multirobot routing approach (Figure 6).

Fig. 6: Industrial port area of Fos-sur-Mer.

The industrial port area comprises a number of buildings and installations that generate high-risk activities. The environment has been represented by a grid of dimension 80×48 and divided into 3840 cells of addresses $a_1, \ldots, a_{3840}$. Patrols of unmanned aerial vehicles are considered. Black cells represent buildings or obstacles where robots cannot stay whereas white cells represent open areas where mobile robots can stay and take measurements. To avoid any collision problem, we consider that the size of each cell is 10m $\times$ 10m.

The resolution of a multi-robot routing problem in such a large and complex envi- ronment is very challenging and expensive. We assume in this case study that such a complex industrial environment can be divided into several sub-areas based on criteria such as capacity, size, and geographical characteristics. As a result, the MRR problem can be addressed separately for each environment sub-area, which simplifies its solution. In this context, we aim in this experiment to demonstrate the effectiveness of our approach to resolving the MRR problem by focusing on a sub-area of the industrial environment (see the limited area (dashed line) in Figure 7). Validating our approach in this subarea will enable us to generalize its application to all other subareas of the environment as well.

In this study, three specific risks will be addressed. For simplicity, we assume that each risk can be directly addressed with a set of appropriate sensors and we consider the measurements A, B and C as follows:

- the risk of fire or explosion of a flammable product following a leak or spill (measurement A);
- the risk of a toxic emission of a dangerous product due to a leak or a spill (measurement B);
- the risk of water pollution due to a leak or spill (measurement C).

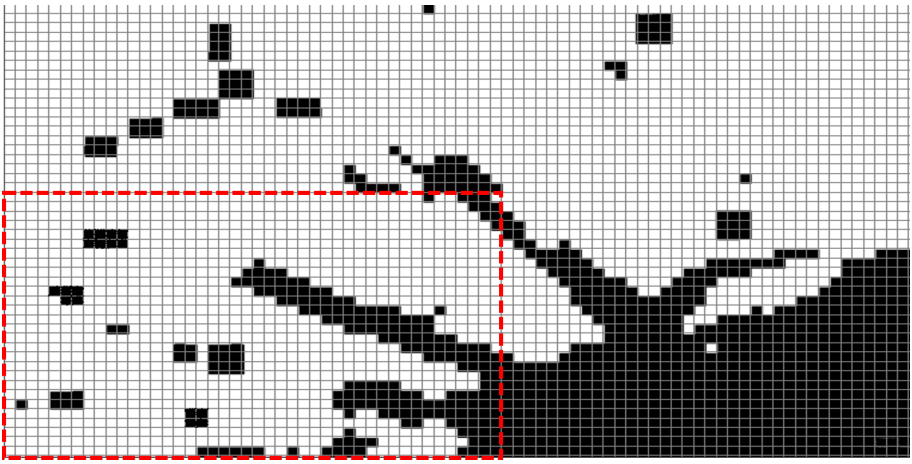


Fig. 7: The environment with 3840 cells for the Industrial port area of Fos-sur-Mer.

Table 1: Initial environnement data.

Site	Address	Position(x, y)	Measurements/cost() in TUs
p_1	a_{641}(depot)	(11,8)	0
p_2	a_{885}	(5,12)	$A/C_{885}(A) = 4$
p_3	a_{966}	(6,13)	$B/C_{966}(B) = 3$
p_4	a_{1950}	(30,25)	$A/C_{1950}(A) = 2$ $B/C_{1950}(B) = 4$ $C/C_{1950}(C) = 5$
p_5	a_{523}	(43,7)	$A/C_{523}(A) = 5$ $C/C_{523}(C) = 2$
p_6	a_{753}	(33,10)	$C/C_{753}(C) = 7$
p_7	a_{1290}	(10,17)	$B/C_{1290}(B) = 2$ $C/C_{1290}(C) = 2$
p_8	a_{343}	(23,5)	$A/C_{343}(A) = 7$ $C/C_{343}(C) = 9$
p_9	a_{348}	(28,5)	$A/C_{348}(A) = 4$ $C/C_{348}(C) = 4$
p_{10}	a_{2380}	(60,20)	$A/C_{2380}(A) = 6$
p_{11}	a_{774}	(10,30)	$A/C_{774}(A) = 4$
p_{12}	a_{873}	(73,11)	$B/C_{873}(B) = 5$
p_{13}	a_{2330}	(54,10)	$A/C_{2330}(A) = 6$ $C/C_{2330}(C) = 6$

Initially, the robots will visit 10 sites to take surveillance measurements during the monitoring mission. In Table 1, the geographical coordinates of the task locations, the different measurements that need to be performed at each site, and the basic costs associated with their execution are described. Such costs correspond to the time required to move the robots and complete the measurements. A measurement of a given type A, B or C can have a different

cost at each site based on the severity of the risk. But, for simplicity, we assume that the measurement costs do not depend on the robot that performs it.

In order to perform measurements, the agents of type r_1 are equipped with sensors for measurements A and B, while the agents of type r_2 are equipped with sensors for measurements A and C (Table 2).

Table 2: Types of robots

	Sensor - A	Sensor - B	Sensor - C	Operational cost	Autonomy (TUs)
r_1	✓	✓		2	250
r_2	✓		✓	1	200

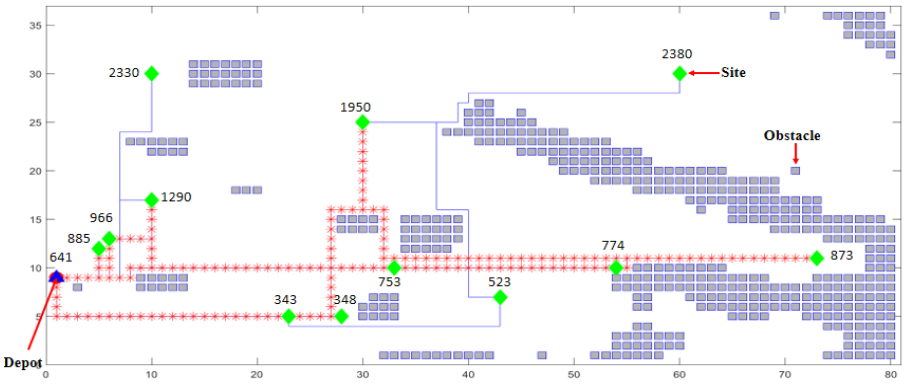
Based on the initial assignment, the system generates trajectories for robots of type r_1 (Figure 8):

$Agent(TC, r_1) = (p_1, \emptyset)\ (p_7, B)\ (p_3, B)\ (p_1, \emptyset)\ (p_2, A)\ (p_1, \emptyset)\ (p_8, A)\ (p_4, \{A, B\})\ (p_{12}, B)\ (p_1, \emptyset)$,

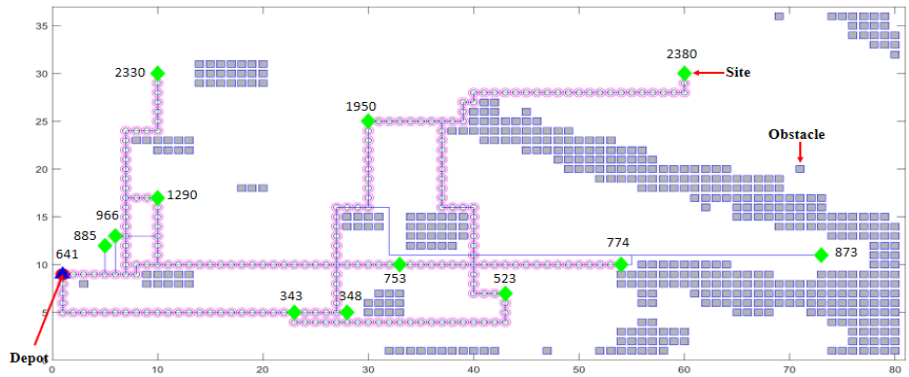
(observe that $s_1^1 = p_1, s_2^1 = p_7$ and so on), and for robots of type r_2:

$Agent(TC, r_2) = (p_1, \emptyset)\ (p_{13}, \{A, C\})\ (p_7, C)\ (p_1, \emptyset)\ (p_8, C)\ (p_5, \{A, C\})\ (p_{10}, A)\ (p_4, C)\ (p_9, \{A, C\})\ (p_1, \emptyset)\ (p_{11}, A)\ (p_6, C)\ (p_1, \emptyset)$,

TC being the node where all terminal conditions are fulfilled. The optimal solution is obtained by considering $\beta_l = 45$ and $\beta_g = 60$ with a total patrol cost $f^* = 888$ TUs. As part of this initial solution, the robots will start and end their missions at the same site a_{431}. Observe that, for autonomy reasons, the robot of type r_1 returns to the depot point to recharge its battery during the mission.



(a) Trajectory σ_{r1} of the robot r_1



(b) Trajectory σ_{r2} of the robot r_2

Fig. 8: Simulation results for the initial assignment

As the mission progresses, new events occur and an appropriate reassignment solution is needed. In order to avoid too frequent repetitions of the reconfiguration with respect to the progress of surveillance operations and to ensure the successful execution of the dynamic mission, we consider the reassignment time $\Delta'_t = 12$. Thus, new events are checked every 12 TUs. In our context, two reassignment scenarios are considered.

- Scenario 1: new tasks occur as time is running;
- Scenario 2: one of the robots experiences a breakdown.

For the first scenario, consider that two tasks with different measurements have appeared during the mission at time $t = 78$ TUs. Table 3 presents the position of the new tasks and the related measurements.

To reassign tasks to robots, the system updates the set $\mathcal{P}$ by removing the tasks that have already been performed and adding the new tasks. Let us first refer to p_1^1 and p_1^2 as the cells where the robots r_1 and r_2 stay when a

Table 3: New tasks

Site	Address	Position(x, y)	Measurements/cost() in TUs
p_{14}	a_{1135}	(15,15)	$B/C_{1135}(B) = 3$
p_{15}	a_{750}	(30,10)	$A/C_{750}(A) = 4$ $B/C_{750}(B) = 2$

.

reconfiguration occurs. The obtained patrol after reassignment is as follows:

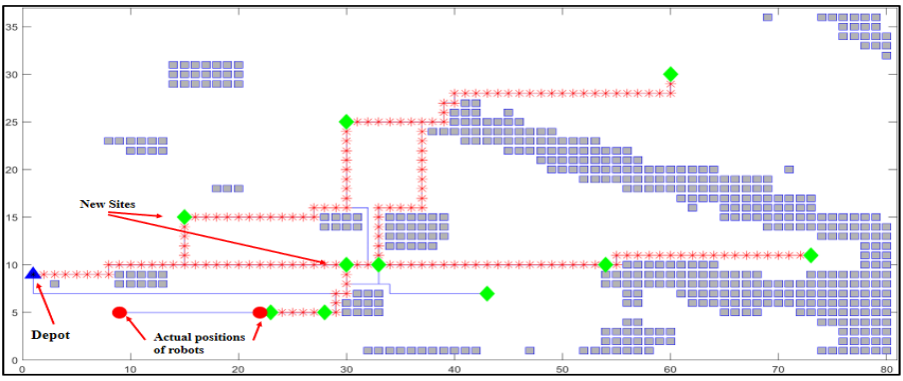
$Agent'(TC, r_1) = (p_1^1, \emptyset)\ (p_9, (A))\ (p_8, A)\ (p_{15}, B)\ (p_{14}, B)\ (p_4, (A, B))\ (p_{10}, A)$ $(p_1, \emptyset)\ (p_{12}, B)\ (p_1, \emptyset)$,

and

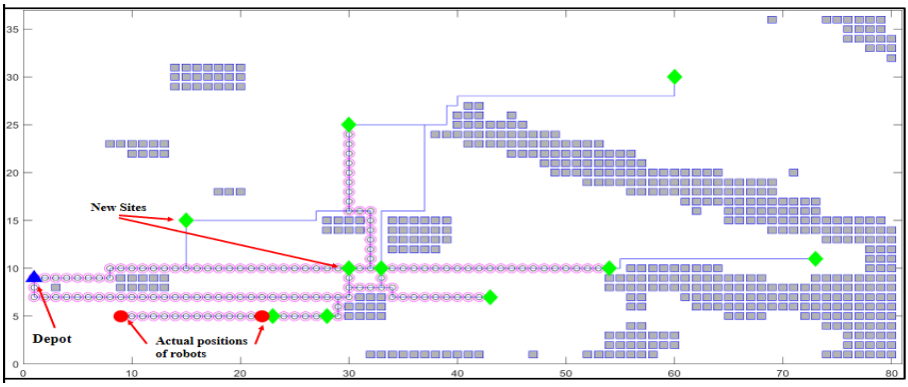
$Agent'(TC, r_2) = (p_1^2, \emptyset)\ (p_9, C)\ (p_{15}, A)\ (p_4, C)\ (p_6, C)\ (p_5, C)\ (p_1, \emptyset)$ $(p_{11}, A)\ (p_1, \emptyset)$.

As presented in Figure 9, the robot trajectories $Agent'(TC, r_1)$ and $Agent'(TC, r_2)$ start from the actual robot positions which are the cells where they are located at the time of reassignment. In this example, r_1 stays at $p_1^2 = a_{342}$ (in the path between the sites p_1 and p_8) and r_2 stays at $p_1^2 = a_{329}$ (in the path between the sites p_1 and p_8) when the reconfiguration occurs. The reconfiguration computation is performed at $t = 84$ TUs.

In this case, the total patrol cost after reassignment increases to $f^* = 1091$ TUs (obtained for $\beta_l = 45$ and $\beta_g = 60$), which reflects the cumulative time for executing the initial solution before reassignment ($f_1^* = 143$ TUs) and the cumulative time for executing the solution after reassignment ($f_2^* = 948$ TUs). Observe that another situation can occur when the robot is located on a site and has not yet completed its measurements at the time of reassignment. In this case, the new trajectory of the robot will start from this site and after the robot has completed its tasks.



(a) Trajectory σ'_{r_1} of the robot r_1



(b) Trajectory σ'_{r_2} of the robot r_2

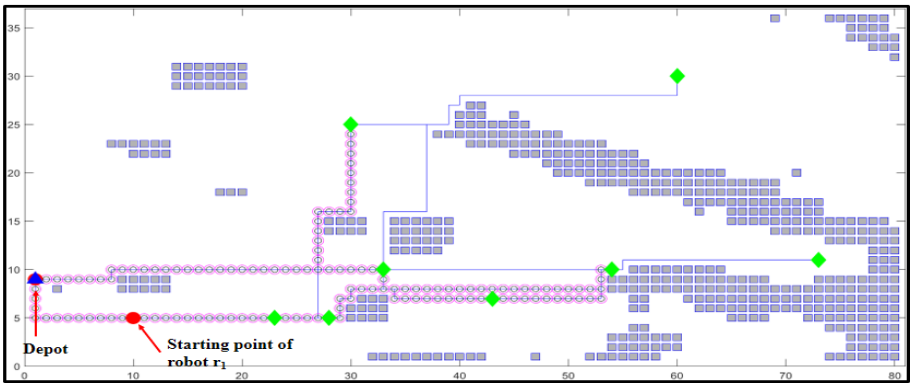
Fig. 9: Reassignment for the first scenario

For the second scenario, we assume that the robot r_2 failed at time $t = 30$ TUs, thus, the system computes a reassignment for another robot of type r_2 assumed to be ready for use at the depot. The reassignment takes into account all the remaining tasks in the mission and not only the tasks of the robot that fails down. Consequently, the updating may also affects the trajectory of the robot of type r_1. Let us refer to p_1^1 and p_1^2 as the cell for where the robots continue the mission after reconfiguration. In the simulation we can observe that the robot r_1 starts its new trajectory from its current position p_1^1=a_{1084} at time $t = 60$ TUs and the new robot of type r_2 starts its trajectory at the depot p_1^2=a_{641} (Figure 10). As a result of the reassignment, the following patrol has been obtained with a cost of $f^* = 1031$ TUs with $f_1^* = 100$ TUs and $f_2^* = 931$ TUs (obtained for $\beta_l = 45$ and $\beta_g = 60$):

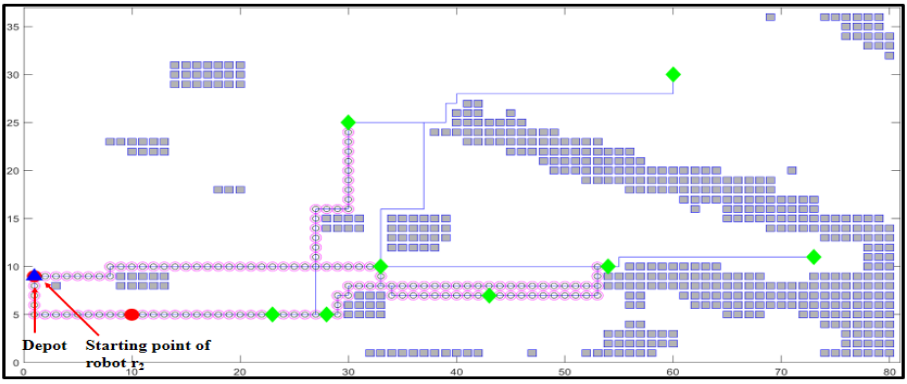
$Agent"(TC, r_1) = (p_1^1, \emptyset)\ (p_8, A)\ (p_9, A)\ (p_4, \{A, B\})\ (p_{10}, A)\ (p_1, \emptyset)\ (p_{12}, B)\ (p_1, \emptyset)$,

and

$Agent"(TC, r_2) = (p_1^2, \emptyset)\ (p_8, C)\ (p_{11}, C)\ (p_5, \{A, C\})\ (p_6, C)\ (p_1, \emptyset)\ (p_4, C)\ (p_1, \emptyset)\ (p_9, C)\ (p_1, \emptyset)$.



(a) Trajectory σ''_{r_1} of the robot r_1



(b) Trajectory σ''_{r_2} of the robot r_2

Fig. 10: Reassignment for the second scenario

As expected we can note that reassignment increases the cumulative time in both scenarios by comparison to the initial solution (of cumulative time of $f^* = 888$ TUs). The main reasons are that in the first scenario additional tasks are considered whereas in the second scenario an additional agent is involved. In spite of this increase, reassignment remains the most cost-effective solution for resolving MRR problems in dynamic environments with respect to the considered changes.

7 Conclusion

This paper has presented a real time system to remotely and automatically manage monitoring patrols in high risk industrial area. The system uses a heuristic approach based on HFBS algorithm to deal with dynamic multi-robot routing problems by executing a reassignment program periodically to take into account all changes in the industrial environment during the surveillance patrol. The proposed method was illustrated using different simulation scenarios by considering the industrial port area of Fos-sur-mer in France as a case study.

In future works, we discuss more deeply the choice of the cell size and the impacts of such choice in the performance of the approach. It would also be interesting to enhance the presented approach by combining distributed methods and some learning approaches so as to reinforce smart communication and coordination between robots, in which each robot can be considered as a smart decision-making unit. For validation purpose, we aim to apply our approach in a real context to test the efficiency of our system in hazardous industrial areas. Additionally, we can improve our approach for performing other critical missions, such as rescue missions, where agents must carry not only sensors, but also resources that need to be transported and distributed from origin to destination.

Conflict of interests

- The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Funding

- The authors thank the National Research Agency, France, Region Normandie, Region Hauts de France for their financial support with project ANR-21-SIOM-0009 - APPRENTIS 2021 – 2023 “Formal APPRoach and artificial intElligeNce for the moniToring and Intervention optimization of mobile agents in industrial Sites”.

References

[1] Y. Rizk, M. Awad, and E. W. Tunstel, “Cooperative heterogeneous multi-robot systems: A survey”, ACM Computing Surveys, vol. 52, no. 2. Association for Computing Machinery, May 01, 2019. doi: 10.1145/3303848.

[2] E. Schneider, E.I. Sklar, S. Parsons, “Mechanism Selection for Multi-Robot Task Allocation”, in: Gao, Y., Fallah, S., Jin, Y., Lekakou, C. (eds) Towards Autonomous Robotic Systems. TAROS 2017. Lecture Notes in Computer Science, vol 10454, pp. 421-435. Springer, Cham, 2017.

[3] B. P. Gerkey and M. J. Matarić, "A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems", International Journal of Robotics Research, 23(9), pp.939-954, 2004.

[4] A. Khamis, A. Hussein, and A. Elmogy, "Multi-robot task allocation: A review of the state-of-the-art", Studies in Computational Intelligence, vol. 604, pp. 31–51, 2015. doi: 10.1007/978-3-319-18299-5_2.

[5] J. K. Mogali, J. Kinable, S. F. Smith, and Z. B. Rubinstein, "Scheduling for multi-robot routing with blocking and enabling constraints", Journal of Scheduling, vol. 24, no. 3, pp. 291–318, Jun. 2021. doi: 10.1007/s10951-021-00684-9.

[6] M. Gam, D. Lefebvre, A. J. Telmoudi, and L. Nabli, "Configuration of surveillance patrols with Petri nets for safety issues", in 7th International Conference on Control, Decision and Information Technologies, CoDIT 2020, pp. 427–432, Jun. 2020. doi: 10.1109/CoDIT49905.2020.9263900.

[7] M. Gam, D. Lefebvre, L. Nabli, and A. J. Telmoudi, "A Petri nets based approach for the optimization of surveillance patrols", International Journal of Sensor Networks, 2021. doi/epdf/10.1504/IJSNET.2021.117486.

[8] M. Gam, A. J. Telmoudi, and D. Lefebvre, "Hybrid Filtered Beam Search Algorithm for the Optimization of Monitoring Patrols", J Intell Robot Syst 107, 26, 2023. https://doi.org/10.1007/s10846-022-01800-3.

[9] M. G. Lagoudakis, E. Markakis, D. Kempe, P. Keskinocak, A. Kleywegt, S. Koenig, C. Tovey, A. Meyerson, and Sonal Jain, "Auction-Based Multi-Robot Routing", Robotics: Science and Systems, pp. 343-350, 2005.

[10] N. Seenu, R. M. Kuppan Chetty, M. M. Ramya, and M. N. Janardhanan, "Review on state-of-the-art dynamic task allocation strategies for multiple-robot systems", Industrial Robot, vol. 47, no. 6. Emerald Group Holdings Ltd., pp. 929–942, Oct. 09, 2020. doi: 10.1108/IR-04-2020-0073.

[11] A. Dakkoune, L. Vernières-Hassimi, S. Leveneur, D. Lefebvre, and L. Estel, "Analysis of thermal runaway events in French chemical industry", J. of Loss Prevention in the Process Industries, Volume 62, 103938, 2019.

[12] R. Almadhoun, T. Taha, L. Seneviratne, J. Dias, and G. Cai, "A survey on inspecting structures using robotic systems", Int. J. of Advanced Robotic Systems, vol. 13, no. 6. SAGE Publications Inc., pp. 1–18, Nov. 2016.

[13] Espina, M.V. et al., "Multi-robot Teams for Environmental Monitoring", in: Remagnino, P., Monekosso, D.N., Jain, L.C. (eds) Innovations in Defence Support Systems – 3. Studies in Computational Intelligence, vol 336. Springer, Berlin, Heidelberg, 2011. doi: 10.1007/978-3-642-18278-5_8.

[14] Liu, C., and Kroll, A., "A Centralized Multi-Robot Task Allocation for Industrial Plant Inspection by Using A* and Genetic Algorithms", in: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds) Artificial Intelligence and Soft Computing. ICAISC 2012. Lecture Notes in Computer Science, vol 7268. Springer, Berlin, Heidelberg, 2012. doi: 10.1007/978-3-642-29350-4_56

[15] K. Jose and D. K. Pratihar, "Task allocation and collision-free path planning of centralized mult i-robots system for industrial plant inspection using heuristic methods", Rob. Auton. Syst., vol. 80, pp. 34–42, Jun. 2016.

[16] H. Xu, T. K. Satish Kumar, D. Johnke, N. Ayanian and S. Koenig, "SAGL: A New Heuristic for Multi-Robot Routing with Complex Tasks", 2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI), San Jose, CA, USA, pp. 530-535, 2016. doi: 10.1109/ICTAI.2016.0087.

[17] J. K. Mogali, J. Kinable, S. F. Smith, and Z. B. Rubinstein, "Scheduling for multi-robot routing with blocking and enabling constraints", Journal of Scheduling, vol. 24, no. 3, pp. 291–318, Jun. 2021.

[18] T. Sakamoto, S. Bonardi and T. Kubota, "A Routing Framework for Heterogeneous Multi-Robot Teams in Exploration Tasks", in IEEE Robotics and Automation Letters, vol. 5, no. 4, pp. 6662-6669, Oct. 2020. doi: 10.1109/LRA.2020.3016285.

[19] J. Mikula, "Meta-heuristics for routing problems", Bachelor's thesis, CTU in Prague, 2018.

[20] Y. Q. Han, J. Q. Li, Z. Liu, C. Liu, and J. Tian, "Metaheuristic algorithm for solving the multi-objective vehicle routing problem with time window and drones", International Journal of Advanced Robotic Systems, vol. 17, no. 2, Mar. 2020. doi: 10.1177/1729881420920031.

[21] G.R. Brodie, C.D.J. Waters, "Integer linear programming formulation for a vehicle routing problem", European Journal of Operational Research, Volume 34, Issue 3, 1988.

[22] N. Haghani, J. Li, S. Koenig, G. Kunapuli, C. Contardo, and J. Yarkony, "Integer Programming for Multi-Robot Planning: A Column Generation Approach", Jun. 2020, [Online]. Available: http://arxiv.org/abs/2006.04856

[23] J. Melvin, P. Keskinocak, S. Koenig, C. Tovey and B. Y. Ozkaya, "Multi-robot routing with rewards and disjoint time windows", IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2332-2337, 2007. doi: 10.1109/IROS.2007.4399625.

[24] J. David, T. Rögnvaldsson, B. Söderberg, and M. Ohlsson, "Deterministic annealing with Potts neurons for multi-robot routing", Intelligent Service Robotics, Jul. 2022. doi: 10.1007/s11370-022-00424-8.

[25] W. Dai, H. Lu, J. Xiao, Z. Zeng, and Z. Zheng, "Multi-Robot Dynamic Task Allocation for Exploration and Destruction", Journal of Intelligent and Robotic Systems: Theory and Applications, vol. 98, no. 2, pp. 455–479, May 2020. doi: 10.1007/s10846-019-01081-3.

[26] S. Choudhury, J. K. Gupta, M. J. Kochenderfer, D. Sadigh, and J. Bohg, "Dynamic Multi-Robot Task Allocation under Uncertainty and Temporal Constraints", Robotics:Science and Systems, 2020, [Online]. Available: https://arxiv.org/abs/2005.13109.

[27] L. Jin, S. Li, H. M. La, X. Zhang, and B. Hu, "Dynamic task allocation in multi-robot coordination for moving target tracking: A distributed approach", Automatica, vol. 100, pp. 75–81, Feb. 2019. doi: 10.1016/j.automatica.2018.11.001.

[28] G. Ortiz, B. Andres, F. Fraile, R. Poler, and Á. O. Bas, "Fleet management system for mobile robots in healthcare environments", Journal of Industrial Engineering and Management, vol. 14, no. 1, pp. 55–71, 2021. doi: 10.3926/jiem.3284.

[29] N. Zagradjanin, A. Rodic, D. Pamucar, and B. Pavkovic, "Cloud-based multi-robot path planning in complex and crowded environment using fuzzy logic and online learning", Information Technology and Control, vol. 50, no. 2, pp. 357–374, 2021. doi: 10.5755/j01.itc.50.2.28234.

[30] Z. Wang, S. Yang, X. Xiang, A. Vasilijevic, N. Miskovic, and D. Nad, "Cloud-based mission control of USV fleet: Architecture, implementation and experiments", Control Engineering Practice, vol. 106, Jan. 2021. doi: 10.1016/j.conengprac.2020.104657.

[31] R. Dindokar, N. Choudhury and Y. Simmhan, "Analysis of Subgraph-Centric Distributed Shortest Path Algorithm", 2015 IEEE International Parallel and Distributed Processing Symposium Workshop, pp. 1185-1190, 2015. doi: 10.1109/IPDPSW.2015.87.