Discrete Event Dynamic Systems (2023) 33:425–453
https://doi.org/10.1007/s10626-023-00383-x

Multi-level control for multiple mobile robot systems

Elzbieta Roszkowska[1] · **Piotr Makowski-Czerski**[2] · **Lukasz Janiec**[1]

Received: 22 March 2023 / Accepted: 10 August 2023 / Published online: 18 October 2023

Abstract
This paper contributes with a multi-level, hierarchical control system for a fleet of mobile robots sharing a common 2D motion space. The system consists of three levels, with the top level being a supervisor based on a discrete representation of the Multiple Mobile Robot System (MMRS), in which robot motion processes are seen as sequences of stages. The supervisor controls centrally the changes of their stages by robots, ensuring their collision-, and deadlock-free concurrent movement. The intermediate control level supervises locally the execution of robot motion on individual stages in a manner consistent with the decisions of the top level. The lowest level, robot control, is responsible for motion execution as determined by the local supervisor. We capitalize on some earlier results concerning the supervisory control of MMRS and propose a common framework for three supervisory control models. Then we propose relevant solutions for the local supervisors, in particular, a DES-based robot-motion-mode control and application of the Artificial Potential Field model for ensuring collision-free motion of two robots sharing a space sector. Next we assume simple robot control and subject the system to simulation experiments aimed at comparing the impact of the different solutions on the performance of MMRS.

Keywords Multiple mobile robot system · Hybrid control · DES-based control · Artificial potential field

This article belongs to the Topical Collection: *Special Issue on Applications 2022*
Guest Editors: Jan Komenda, Tomas Masopust, and Spyros Reveliotis

✉ Elzbieta Roszkowska
elzbieta.roszkowska@pwr.edu.pl

Piotr Makowski-Czerski
piomcz@gmail.com

Lukasz Janiec
lukasz.janiec@pwr.edu.pl

[1] Department of Cybernetics and Robotics, Wroclaw University of Science and Technology, Janiszewskiego 11/17, Wroclaw 50-372, Poland

[2] Graduate of Embedded Robotics Program, Wroclaw University of Science and Technology, 2022, Wroclaw, Poland

1 Introduction

The use of a mobile robot team in place of one robot substantially increases the performance of many robotic applications, including those related to area searching, search and rescue, interplanetary exploration, extraction of minerals, agriculture, forestry, or transport. A key issue in the design of such systems is to coordinate the movement of a number of robots operating in the same workspace. Regardless of their tasks, the robots must be able to effectively share a common area in order to prevent the mutual disruption of traffic and effectively pursue their missions.

In the above context, a widely considered problem in robotics has become multiple-robot motion planning. Planning approaches can be categorized in various ways, with two categories - centralized and decoupled - that can be distinguished as opposite ends of the spectrum of solutions. A centralized approach typically constructs a path in a composite configuration space, which is formed by the Cartesian product of the configuration spaces of the individual robots, e.g. Barraquand and Latombe (1991). A decoupled approach typically generates paths for each robot independently, and then considers the interactions between the robots, e.g. Buckley (1989). The approaches that lie in the middle of the spectrum, e.g. La Valle and Hutchinson (1998), constrain the robot motions before considering interactions between robots and assume that they travel on independent networks of paths called road-maps.

In classical robotics, these problems are understood as kinodynamic motion planning (Donald et al. 1993), that takes into account the robot kinematics, dynamics, and collision avoidance requirements, and seeks for each robot a trajectory specifying a continuous sequence of positions, orientations, and speeds while avoiding contact with any stationary and moving obstacles (other robots). The methodology applied usually combines techniques from optimal control and mathematical programming. For example, the approach proposed in Peng and Akella (2004) consists of identifying collision segments along each robot's path, and then optimizing the robots velocities along the collision and collision-free segments. First, for each path segment, for each robot, the minimum and maximum possible traversal times that satisfy the dynamics constraints are computed by solving the corresponding two-point boundary value problems. The collision avoidance constraints for pairs of robots can then be combined to formulate two mixed integer linear programming (MILP) models that provide schedules that are lower and upper bounds on the optimum; the upper bound schedule is designed to be a continuous velocity schedule.

An interesting concept of multiple mobile robot motion planning that applies constraint-based techniques is also considered in Pecora et al. (2012); Andreasson et al. (2015). These papers employ the notion of trajectory envelope, which is a set of spatial and temporal constraints on the position of a vehicle's reference point. An envelope defines a set of alternative execution patterns that ensure conflict-free robot motion. The approach consists of multiple constraint solvers, which progressively refine trajectory envelopes according to mission requirements.

Many works assume that robot paths are already given, and focus solely on robot velocity planning in a way to optimize some system criteria. For example, paper (Soltero et al. 2011) deals with the problem of persistent monitoring in multi-robot systems, seeking a policy to calculate the speed of each robot that maximizes the stability margin of a function characterizing changes in the environment introduced by the robots while ensuring conflict-free robot motion. The proposed approach to collision and deadlock-avoidance is based on the assumption of temporary stops of robots during their travel and is correct. However, the stopping policy is based on a condition that is sufficient, but not necessary for no-collisions, which makes the proposed solution excessively conservative.

Decoupled approaches to motion planning of multiple mobile robots are typically reactive, that is, aim to make robots adapt themselves in real time to changes in the environment (Ferrera et al. 2013, 2017). Robots are often viewed as agents with simple dynamic and kinematic models. An interesting example of such an approach is considered in Pallottino et al. (2007). The paper proposes a novel policy for steering multiple vehicles that move with a constant speed between assigned start and goal configurations, ensuring collision avoidance. This policy is based on a concept of reserved regions, over which active agents claim exclusive ownership, and traffic rules that ensure maintaining the interiors of reserved regions disjoint. The policy is decentralized, as it requires that each agent decides about its own motion by applying traffic rules based on the locally available information, and scalable, in the sense that the complexity of the decision algorithms do not increase with the number of agents in the scenario. A reactive method of motion control is also employed in Ferrera et al. (2013), where decentralized robot coordination algorithm takes the form of a state machine. Tomlin et al. (2007) and Lygeros et al. (1998) contribute with hybrid control solutions for aircraft and highway systems, respectively. The air traffic model is inspired by linear hybrid automata, with the difference that it allows for a nonlinear continuous dynamic model within each discrete state and a general discrete transition relation. The methodology of the highway model synthesis joins techniques from game theory and optimal control. Another common reactive method makes use of potential fields (Khatib 1986). In this approach, a robot moves through space as if it is being acted upon by a set of forces. Attractive forces pull the robot toward a goal destination, while repulsive forces push the robot away from obstacles and/or other robots. At each point in the configuration space, the robot moves along the vector representing the combined forces acting on that point in the configuration space.

The main contribution of the above discussed works are methodologies ensuring collision-free control of robot movement. The resulting algorithms have, however, two major drawbacks: a) due to high computational complexity, they are not scalable, and b) they do not give formal guarantees of the correct operation of the system, e.g., in terms of ensuring the completion of the robot missions. The ineffectiveness of these models in providing the above properties is mainly due to the assumed representation of the robot system, whose operation is abstracted in continuous time.

A new trend in the robotic motion planning literature is to use formal methods, like model checking, reactive synthesis and supervisory control theory, to automatically design controllers that drive a mobile robot to accomplish some high level missions in a guaranteed manner (Lin 2014; Farrell et al. 2018; Kress-Gazit et al. 2018). In robotics, formal synthesis provides a framework for specifying complex robot tasks in a mathematically precise language and automatically transforming these specifications into correct-by-construction robot controllers. As noted in Kress-Gazit et al. (2018), the process of automating high-level behavior of robots is the major focus of related communities, most notably the artificial intelligence (AI) community planning intelligent robot behavior and the discrete event systems (DES) community. In AI, the planning problem is typically represented as a set of actions, each with preconditions and postconditions. Then, planning algorithms search for a sequence of actions that will lead the system from the initial state to the goal state. In discrete event systems, the system (plant) is a transition system with states and transitions. The main problem addressed by the discrete event systems community is finding a supervisory controller that chooses which controlled transitions to take so that the system achieves a required high-level behavior.

This work falls in the area of interest of the DES community as characterized above. Based on some concepts from our earlier works (Roszkowska 2005; Reveliotis and Roszkowska 2011; Roszkowska and Reveliotis 2013), the paper contributes with a multi-level, hierarchical control system, where the logic of robot coordination is developed using the DES formalism.

The proposed models ensure the correct space sharing by a group of robots through imposing certain constraints on their motion, and can be applied to mobile robots accomplishing any arbitrarily assumed missions in both centralized and distributed supervisory control architectures. To respect the requirements of a so-defined DES-based supervisory-control scheme, the robots need to modify their individual motion control profiles, based on the CTS (Continuous Time System) abstraction. Consequently, the control of a team of multiple mobile robots, requires a hybrid control system, incorporating both DES and CTS control layers.

One of the basic issues when modeling a multiple mobile robot system (MMRS) in the form of a DES is the concept of its discrete representation. In the works cited above, two main approaches can be distinguished: i) division of robot motion paths into sub-paths or *sectors*, and ii) division of the robot motion area into sub-areas or *cells*. Then, a simple solution of the problem of collision-free robot movement consists of controlling the robot transitions between sectors/cells and, in the case of a potential conflict, refraining the robot from entering the next sector/cell until the conflict disappears. In approach (i), a conflict occurs if the minimal distance between two sectors respectively occupied by two robots does not provide a desired degree of separation between them. In approach (ii), a conflict is a situation where more than one robot occupies the same cell. More complex solutions allow exceeding these constraints, which potentially leads to more efficient MMRS performance, but require additional robot coordination levels. Introduction of the constraints ensuring collision-free robot motion has a side effect that is the deadlock phenomenon, whose handling requires imposing further restriction on the supervisory control. To achieve realization of the required logic, the robots must be able to modify their motion and be equipped with a set of CTS robot control algorithms that induce adequate motion modes.

In the following sections we present a three-level control system, including a central supervisor, local controllers of robot motion mode, and low-level robot control. Capitalizing on earlier results we introduce a common framework for three supervisory control models and propose relevant solutions for the local supervisors. The different models are subjected to simulation experiments aimed at comparing their impact on the MMRS performance.

2 Assumptions and requirements

We consider a Multiple Mobile Robot System (MMRS) viewed as a group of autonomous mobile robots sharing a 2D space. Each robot performs a mission that requires it to travel along a specific path. The path of each robot is planned independently, without taking into account any positional constraints introduced by the paths of other robots. The robots operate asynchronously and are able to control their motion with path-following algorithms that allow each of them to correctly perform its mission when alone on the stage. When sharing the motion space, the robots must refine their motion strategies in order to avoid collisions, through modification of their paths, velocity profiles, or both.

The objective of the MMRS control is to ensure that the operation of the system is *correct* and *efficient*, which generates the following problems:

1. How to modify dynamically the initially assumed motion control of the robots so that:
   (a) at each moment of their motion, the areas occupied by the robots are disjoint,
   (b) in a finite time interval, all the robots will have accomplished their missions.
2. How to induce efficient MMRS behavior within the admissible (i.e. observing requirements (1.a) and (1.b)) robot concurrent operation.

As can be noticed, satisfaction of requirement (1.a) implies collision-free motion of the robots, whereas requirement (1.b) ensures that the modification of the initially assumed robot motion strategies will guarantee their convergence to the destination points and occurrence of no such side effects as deadlocks or starvation. Requirement (2) implies the need of a flexible model of MMRS control, that leaves room for the optimization of the system efficiency, and of tools to carry it out. The notion *efficiency* refers to real-valued measures of individual robots performance or their combination into single scalar criteria to be optimized. For example, a typical individual criterion for a robot accomplishing its travel mission is the minimal travel time, and a system performance criterion is the minimal time taken by the last robot to reach its goal.

To achieve realization of these postulates, we propose a hierarchical, hybrid control based on the concept of dividing the robot motion processes into stages and consisting of three control levels:

- stage transition control
- stage pass control
- robot motion control

The stages of robot travel processes arise from the partition of robot paths into sectors; the presence of a robot in a particular sector determines the current stage of its motion process. The two upper control levels are supervisors that provide the logic of the admissible progress of each particular robot on its path. The stage transition control decides if a robot can transfer from one sector to the next one in a way that ensures no collisions or deadlocks among the robots. The stage pass control defines a set of robot motion modes (e.g., constant velocity, acceleration, deceleration, standstill) and the logic of their application. They take the form of a state machine that enables each robot to conform to the decisions set at the stage transition control level. Finally, the motion control is a set of CTS robot control algorithms that induce the motion of a robot in the modes defined by the mode control.

3 Stage transition control

In this section, we discuss DES models of MMRS to be applied for the supervisory control of the robot motion processes. First, we introduce an uncontrolled MMRS, a DES model of the system of asynchronous robot motion processes, and next, we modify it in order to ensure the required behavior of the system. Moreover, as mentioned in Section 1, two ways of the discrete representation of MMRS can be considered: direct partition of robot paths into sectors, and indirect partition of robot paths into sectors, resulting from the division of the motion space into cells. Both of these approaches will be considered below.

3.1 Uncontrolled model of MMRS

For the purposes of the supervisory control development, we will view MMRS as a set of mobile robots $A = \{A_i : i = 1 \ldots n\}$ sharing a planar work space WS. Each robot A_i is represented by a disk with radius a_i, and a curve p_i describing the path followed by A_i. Being a curve on a plane, each path p_i is a continuous map from a one-dimensional space $\mathcal{R}$ to $\mathcal{R}^2$. Thus, in the XY coordinate system, p_i is represented with two functions: $x_i(l)$ and $y_i(l)$, and the starting and ending points, (x_i^s, y_i^s) and (x_i^e, y_i^e), respectively. At the supervisory control level of MMRS, instead of the detailed knowledge of the robots' and paths' geometry, the control model will use a map of MMRS topology, given in the form of a certain relation between robot stages distinguished as follows.

Let $P = \{p_i^k : i = 1 \ldots n, k = 1 \ldots m_i\}$ be the set of path sectors obtained through an arbitrary partition of each path p_i into sub-paths p_i^k such that for each $k > 1$, the end-point of p_i^{k-1} is the start-point of p_i^k. Then $Z = Z_1 \cup Z_2 \cup \ldots \cup Z_n$, where $Z_i = \{z_i^k : k = 1 \ldots m_i\}$, is the set of *robot stages*. The stages represent in a discrete way the states of the robot motion processes. In particular, robot A_i traveling along the k-th sector of its path is at stage z_i^k.

The dynamics of the so characterized MMRS will be viewed as an event-driven system, described in terms of a deterministic automaton (Cassandras and Lafortune 2010).

Definition 1 A deterministic automaton is a 6-tuple $G = (S, E, \Gamma, f, s_0, S_M)$, where

- S is the set of *states*.
- E is the set of *events*. The occurrence of an event causes a state transition in G.
- $\Gamma : S \to 2^E$ is the *feasible event function*. Event $e \in E$ can occur in state $s \in S$ iff $e \in \Gamma(s)$.
- $f : S \times E \to S$ is the *transition function*. f is a partial function defined for pairs (s, e) such that $e \in \Gamma(s)$, i.e., event e is feasible in state s. $s' = f(s, e)$ is the new state resulting from the occurrence of event e in state s.
- $s_0 \in S$ is the *initial state*.
- $S_M \subseteq S$ is the *set of marked states*.

In the discrete picture of MMRS, the state of each robot A_i is distinguished with the accuracy of a stage number, that is, state s_i of robot A_i is given by the number k of its current stage z_i^k. The state of the system of n robots will be represented by the vector of the robot stage numbers $(s_1, s_2, \ldots, s_n)$. The event set E of MMRS will be constituted by the events e_i, $i = 1 \ldots n$, associated with stage transitions of each robot A_i. In the initial (final) state, each robot A_i is at stage 1 (stage m_i, resp.). The uncontrolled MMRS is defined as follows.

Definition 2 Given the set of robot stages $Z = \{z_i^k : i = 1 \ldots n, k = 1 \ldots m_i\}$, the uncontrolled model of MMRS, $U\text{-}MMRS(Z)$ is a deterministic automaton $G = (S, E, \Gamma, f, s_0, S_M)$ such that:

- $S = \{s = (s_1, s_2, ..., s_n) \in \mathcal{N}_1 \times \mathcal{N}_2 \times \ldots \times \mathcal{N}_n\}$, where $\mathcal{N}_i = \{1, \ldots, m_i\}$
- $E = \{e_i : i = 1..n\}$
- $\Gamma(s_1, s_2, ..., s_n) = \{e_i : s_i < m_i\}$
- $f((s_1, s_2, ..., s_n), e_i) = (s_1, \ldots, s_{i-1}, s_i + 1, s_{i+1}, \ldots, s_n)$
- $s_0 = (1, 1, \ldots, 1)$,
- $S_M = \{s_m\}$, where $s_m = (m_1, m_2, \ldots, m_n)$

The above defined model represents asynchronous robot movement. Thus, any event e_i can occur in any state s providing that robot A_i has not reached its final stage yet. The occurrence of event e_i affects only the state of robot A_i, that is, the value of the i-th component of s changes from s_i to $s_i + 1$.

3.2 Collision avoidance in *ξ-MMRS*

We will use the following information about the robot paths' geometry in order to ensure collision-free robot movement.

Definition 3 Given the set of robot path sectors P and the values a_i, $i = 1 \ldots n$ of the robot disks' radii, the *stage conflict relation* is the subset $\xi \subseteq Z \times Z$ such that $\forall i, j = 1..n$, $\forall k = 1..m_i$, $\forall l = 1..m_j$: $(z_i^k, z_j^l) \in \xi$ iff $i \neq j$ and $d_{min}(p_i^k, p_j^l) < a_i + a_j$; where $d_{min}(p_i^k, p_j^l)$ is the minimal distance between path sectors p_i^k and p_j^l.

The concept of the conflict relation is illustrated in Fig. 1 that depicts two paths, p_1 and p_2, of two robots, A_1 and A_2, respectively. Path p_1 consists of six sectors, p_1^1 - p_1^6, and path p_2 consists of seven sectors, p_2^1 - p_2^7. Thus, the stage set $Z = \{z_1^k : k = 1..6 \cup \{z_2^k : k = 1..7\}$. The stage conflict relation ξ includes the sectors that cross each other: $\{(z_1^2, z_2^6), (z_1^3, z_2^5), (z_1^5, z_2^2)\}$, their symmetric pairs: $\{(z_2^6, z_1^2), (z_2^5, z_1^3), (z_2^2, z_1^5)\}$, and the pairs of stages corresponding to the sectors that do not intersect, but the shortest distance between them is less than the sum of the robots radii: $\{(z_1^3, z_2^4), (z_2^4, z_1^3)\}$.

The following property lets us define a strategy for robot traffic coordination that ensures no collisions among the robots.

Property 1 The sub-spaces occupied by any pair of robots $A_i, A_j \in A$ are always disjoint if their respective conflicting stages never overlap each other in time. That is, for each conflicting pair $(z_i^k, z_j^l) \in \xi$, the entry of robot A_i into path sector p_i^k prevents the entry of robot A_j into path sector p_j^l as long as robot A_i does not leave p_i^k.

Proof It stems directly from the definition of the conflict-relation. If conflicting stages never overlap in time then the distance between any two robots will always exceed the sum of their disk radii. Hence the sub-spaces occupied by any two robots will be disjoint. □

The set of stages Z and the stage conflict relation ξ give the parameters of a discrete representation of MMRS. Note that the new abstraction does not involve robots' paths, that is, any direct information on the system geometry. Instead, it consists of a set Z of abstract elements and a well-defined relation ξ on Z. The "well-defined" requirement means that the consideration is limited to such a class of relations that arise from geometrical systems, and no initial or final stage of any robot is in conflict with any other stage in Z. More precisely, we will require that: i) for each ξ, there exists a set of planar paths that can be divided into sectors related in the way given by ξ, and ii) for each $i = 1..n$, neither stage z_i^1 nor $z_i^{m_i}$ is in conflict with any stage in Z,

In order to avoid collisions, the robots must observe the mutual exclusion of conflict stages. Thus, robot A_i must not change stage from z_i^k to z_i^{k+1} if any robot A_j is currently on stage z_j^l

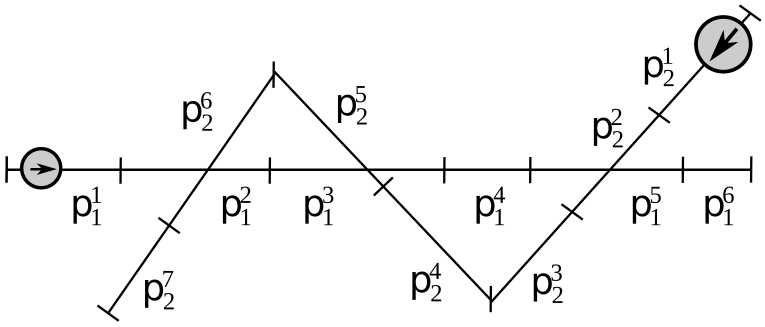


Fig. 1 Example paths of two mobile robots, partitioned arbitrarily into sectors. Robot stages z_1^k and z_2^l conflict iff $d_{min}(p_1^k, p_2^l) < a_1 + a_2$

and stages z_i^{k+1} and z_j^l are in conflict. Otherwise the event associated with the state transition is feasible. Implementation of this principle in the uncontrolled model $U\text{-}MMRS$ requires a relevant restriction of function $\Gamma(s)$, and yields a collision-free control model, $\xi\text{-}MMRS$.

Definition 4 For each well defined pair (Z_w, D_w), $\xi\text{-}MMRS(Z, \xi)$ is a deterministic automaton $G = (S, E, \Gamma, f, s_0, S_M)$ such that:

- the state set S, the event set E, the transition function f, the initial state s_0, and the marked-state set S_M are defined in the same way as in Definition 2 for $U\text{-}MMRS(Z)$
- the feasible event function is given by:

$$\Gamma(s_1, s_2, ..., s_n) = \{e_i : s_i < m_i \ \wedge \ \forall j = 1..n, \ (z_i^{s_i+1}, z_j^{s_j}) \notin \xi\}$$

Function $\Gamma(s)$ ensures that each event e_i, associated with stage transition by robot A_i, is feasible in state s iff robot A_i has not reached yet its final stage and its next stage, $z_i^{s_i+1}$, is not in conflict with the current stage, s_j, of any robot A_j. The automaton model of MMRS gives the logic of the relative robots' progress on their paths, which, due to that no states with conflict stages are reachable, allows the robots to avoid colliding with one another.

3.3 Collision avoidance in *D-MMRS*

A discrete model of MMRS operation can also be obtained through partitioning of the motion space WS into a set W of squares with a side length of at least the diameter of the robot disk, called *cells*. A graphical illustration of such a space division is given in Fig. 2, where the cell borders are depicted by solid horizontal and vertical lines. As can be noticed, depending on its location in W, the disk of a robot will overlap from one to four cells at a time. The subset

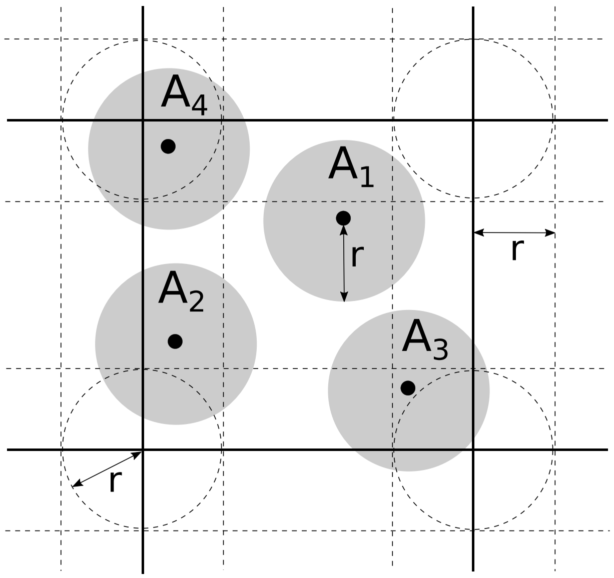


Fig. 2 Robots' work space partitioned into cells (solid lines), and their further partitioning into areas of a fixed number of overlapped cells (dashed lines). Depending on the position of the center of the robot disk, it overlaps 1 cell (A_1), 2 cells (A_2), 3 cells (A_3), or 4 cells (A_4)

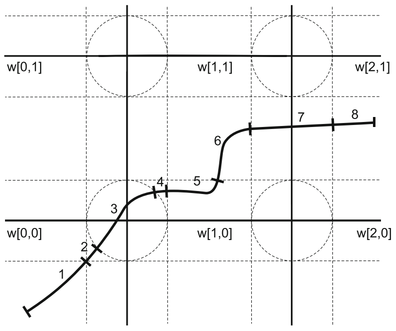
of the cells overlapped or *occupied* by robot A_i can be effectively determined by the relative position of the center point of A_i with respect to another partitioning of the motion plane, induced by the geometry of the cells and the robot disk, and depicted by the dashed lines in the figure. A robot with its disk center in the inner square of a cell (case A_1) overlaps one cell, a robot with its disk center in any of the adjacent rectangles belonging to two neighboring cells (case A_2) overlaps these two cells, a robot with its disk center in a corner square of the cell, but not in the circle inscribed in this square (case A_3), overlaps three cells, and a robot with its disk center in a corner circle of a cell (case A_4) overlaps four cells.

As follows from the above discussion, the path of any robot can be divided into maximal sectors characterized by that at each position of the robot disk center within a sector, the robot overlaps the same subset of cells. The beginnings and ends of such sectors are determined by the intersection points of the path with the induced grid, depicted with dashed lines. An example of such a path partitioning is given in Fig. 3. The path is divided into eight sectors that define eight respective stages of the robot travel. The table below the figure lists the subsets of cells overlapped by the disk of the robot at each particular stage.

Note that in ξ-$MMRS$, the set of robot stages Z arises from any arbitrary partition of the paths into sectors. In contrast, in the model considered now, the set of stages depends on the work space partition into cells and the geometry of the paths. Clearly, if no cell is occupied by more than one robot at a time, the sub-spaces overlapped by the robot disks are disjoint and no collision among the robots occurs. To determine the model that implements this principle, we will treat the set of cells W as system resources, and define the *resource requirement function*

$$D : Z \to 2^W \tag{1}$$

For each $z_i^k \in Z$, $D(z_i^k) \subset W$ indicates the subset of cells occupied by robot A_i at its k-th stage, $k = 1 \ldots m_i$. Similar to the specification of ξ-$MMRS$, we will require that the pair (Z, D) be well defined, that is, i) the values of function D arise from the geometry of the



Stage	Overlapped cells
1	$\{w[0,0]\}$
2	$\{w[0,0], w[0,1], w[1,0]\}$
3	$\{w[0,0], w[0,1], w[1,0], w[1,1]\}$
4	$\{w[0,1], w[1,0], w[1,1]\}$
5	$\{w[1,0], w[1,1]\}$
6	$\{w[1,1]\}$
7	$\{w[1,1], w[2,1]\}$
8	$\{w[2,1]\}$

Fig. 3 Robot path divided into maximal stages with constant subsets of cells overlapped by the disk of the robot traveling along this path

system, and ii) for each robot $A_i \in A$, the resource sets required for its initial and final stage are 'private' singletons, i.e., A_i is entirely located in a cell that is not required for any other stage in Z. The parameters Z and D allow us to modify function $\Gamma(s)$ in the uncontrolled $U\text{-}MMRS(Z)$ so that the robot movement is collision-free.

Definition 5 For each well defined pair (Z, D), $D\text{-}MMRS(Z, D)$ is a deterministic automaton $G = (S, E, \Gamma, f, s_0, S_M)$ such that:

- the state set S, the event set E, the transition function f, the initial state s_0, and the marked-state set S_M are defined in the same way as in Definition 2 for $MMRS(Z)$
- the feasible event function is given by:

$$\Gamma(s_1, s_2, ..., s_n) = \{e_i : s_i < m_i \;\wedge\; \forall j = 1..n,\, j \neq i,\; R_j^{s_j} \cap R_i^{s_i+1} = \emptyset\}$$

As defined above, event e_i is feasible in state $s = (s_1, s_2, ..., s_n)$ iff for each robot $A_j \neq A_i$, the set of resources used in the current state s, $R_j^{s_j}$, and the set of resources required by robot A_i at its next stage z_{i+1}, $R_i^{s_i+1}$, are disjoint. This ensures that in each state reachable from s_0, the subsets of resources used by any two robots are disjoint, i.e., no cell is overlapped by more than one robot at a time, hence no collisions occur.

4 Deadlock avoidance in MMRS

The MMRS models discussed in the previous sections, $\xi\text{-}MMRS$ and $D\text{-}MMRS$, satisfy requirement (1.a) imposed on the control system, that is, ensure collision-free traffic of the robots. Fulfillment of condition (1.b) - deadlock-free robot movement - requires a further restriction of the possibility of event occurrence given by function $\Gamma(s)$. The approach to the development of a relevant deadlock-avoidance algorithm depends on the computational complexity of the *coaccessibility problem* for a given class of models. The problem consist in determining whether or not a given reachable state s is *coaccessible*, i.e., whether or not the final state s_m is reachable from s. As similar problems have been studied in the context of Resource Allocation Systems (RAS) taxonomy (Reveliotis 2005), in the next section, we will consider the MMRS models in this framework.

4.1 RAS classification of MMRS models

The RAS classification applies to systems of discrete processes that share a a set of resources in a non-preemptive manner. Each process consists of a set of stages, and each resource is characterized by its capacity or the number of its units. Subclasses of RAS are distinguished by two criteria: the required execution order of the process stages, and the resource requirements of the process stages. Specifically, the LIN-CON-RAS class consists of the systems having a linearly ordered set of stages and resource requirements defined by the function

$$D : Z \to R_{MS} \tag{2}$$

where R_{MS} is the set of all finite multisets

$$b = \Sigma_{i=1}^{|R|} b_i R_i, \quad b_i \in \{0, 1, \ldots\} \tag{3}$$

defined for a resource set $R = \{R_1, R_2, \ldots R_{|R|}\}$.

The class of RASs corresponding to $D\text{-}MMRS$ is a sub-class of LIN-CON-RAS called FREE-RANGE-RAS (Reveliotis and Roszkowska 2011). Both these classes deal with linear processes, yet in FREE-RANGE-RAS, each resource has only one unit, and the resource requirement function is more specific. Namely, in LIN-CON-RAS, the values of of function D are any multisets of R, whereas in FREE-RANGE-RAS, for each $z_i^k \in Z$, $D(z_i^k)$ is a multiset with at most one copy of each element of R, i.e., it is a set $D(z_i^k) = R_i^k \subset R$, and consists of only 1 to 4 resources, depending on the geometry of the path sector followed by robot A_i at its k-th stage.

For reasons explained later, we will also consider another subclass of LIN-CON-RAS, namely FREE-RANGE*-2-RAS (Roszkowska and Reveliotis 2013). This class differs from FREE-RANGE-RAS in two aspects:

- the capacity of each resource is equal to 2, i.e., each cell can be occupied by up to 2 robots at a time, and
- the path of each robot avoids the areas where the robot's disk overlaps more than two cells at a time. That is, each robot traverses a sequence of cells and its motion process consists of, alternately, a stage when the robot's disk fits entirely in the cell it is traversing, and a stage when the robot transits to the next cell occupying two cells - the previous and the next one.

The automaton modeling this type of MMRS, denoted by $D\text{-}2\text{-}MMRS$, differs from $D\text{-}MMRS$ by its feasible event function $\Gamma(s)$.

Definition 6 For each well defined pair (Z, D), $D\text{-}2\text{-}MMRS(Z, D)$ is a deterministic automaton $G = (S, E, \Gamma, f, s_0, S_M)$ such that:

- the state set S, the event set E, the transition function f, the initial state s_0, and the marked-state set S_M are defined in the same way as in Definition 2 for $MMRS(Z)$
- the feasible event function is given by:

$$\Gamma(s_1, s_2, ..., s_n) = \{e_i : s_i < m_i \ \wedge \ \Sigma_{j=1}^n \mid (R_i^{s_i+1} \setminus R_i^{s_i}) \cap R_j^{s_j} \mid \leq 1\}$$

Taking into account that in the considered model, the resource requirement R_i^k is a set of either one cell or two adjacent cells, we can distinguish two cases of the event feasibility: i) $R_i^{s_i+1} \setminus R_i^{s_i} = \{w\}$, $w \in W$, and ii) $R_i^{s_i} \setminus R_i^{s_i+1} = \{w'\}$, $w' \in W$. Case (i) concerns the event e_i that represents the start of a transition of robot A_i to its next cell w', which is feasible providing that this cell is occupied by no more than one other robot, hence can be allocated to A_i. Case (ii) concerns the event e_i that represents the end of a transition of robot A_i to cell w, which is always feasible as $w \in R_i^{s_i}$, i.e., cell w has already been allocated to A_i, and no more resources to do this are required. On the contrary, the occurrence of e_i results in the release of the previous cell w' by robot A_i.

Note that $D\text{-}2\text{-}MMRS$ ensures the collision avoidance providing that no collision occurs among any two robots sharing a single cell. Thus, another control level is needed to coordinate the robot motion within a cell.

The model $MMRS(Z, \xi)$, using the conflict relation for avoiding collisions, does not belong to the RAS taxonomy, but can be equivalently transformed to a class of RAS. To do this, with each two conflicting stages we associate a distinct resource to be used on the mutually exclusive basis. Consequently, the resource set is defined by

$$R = \{R_{ij}^{kl} = R_{ji}^{lk} : (z_i^k, z_j^l) \in \xi\} \tag{4}$$

Since each stage $z_i^k \in Z$ can be in conflict with a number of other stages $z_j^l \in Z$, $j \neq i$, the motion process of each robot A_i, $i = 1..n$, will be viewed as a sequence of stages z_i^k, $k = 1..m_i$, each of which requires a subset of resources $R_i^k \subset R$, where:

$$R_i^k = D(z_i^k) = \{R_{ij}^{kl} : (z_i^k, z_j^l) \in \xi\} \tag{5}$$

Note that for stages z_i^k that are not in conflict with any other stage, the resource set $R_i^k = \emptyset$. The principle of collision avoidance is in this case the same as in $D\text{-}MMRS$, i.e., the mutually exclusive use of resources. Thus, in both cases, we can use the same automaton model given in Definition 7. However, one should keep in mind that in each case, $D\text{-}MMRS(Z, D)$ is defined for its specific class of parameters (Z, D), arising from its specific way of determining the stages Z and their resource requirements D.

4.2 Deadlock avoidance algorithms

The optimal, i.e., the least restrictive approach to avoid deadlocks is to forbid the occurrence of only those events that get a negative solution of the following decision problem: *Given a coaccessible state s and a feasible event $e_i \in \Gamma(s)$, is state $s' = f(s, e_i)$ coaccessible* Alas, as proved in Reveliotis and Roszkowska (2010), the coaccessibility problem is NP-hard in FREE-RANGE-RAS, which makes it practically impossible to develop an optimal deadlock avoidance algorithm for $D\text{-}MMRS(Z, D)$ systems in the case when the set of stages Z and the resource requirement function D are determined based on the work space partition into cells. For $D\text{-}MMRS$ systems obtained from $\xi\text{-}MMRS$, although no formal proof of the computational complexity has been demonstrated, the construction of a polynomial algorithm to solve the coaccessibility problem does not seem easily achievable. Therefore for both classes of $D\text{-}MMRS$, we will use the same policy, a compromise solution that is not optimal but allows calculation of the decisions in polynomial time.

To provide the correct operation of MMRS, in each current state s, we will screen the events in $\Gamma(s)$. That is, we substitute the feasible-event function $\Gamma(s)$ with a more restrictive *admissible-event function* $\Gamma^*(s)$ that ensures the reachability of the final state s_m from each state s reachable under the screening policy from the initial state s_0.

The admissibility test assumes distinguishing between two types of stages: *private* and *shared*. In $D\text{-}MMRS$, obtained from $\xi\text{-}MMRS$, where the resource set R represents the conflicts among stages, stage $z_i^k \in Z$ is private if the required resource set $R_i^k = \emptyset$. In $D\text{-}MMRS$, where the resource set R is the set of cells, stage $z_i^k \in Z$ is private if for each $z_j^l \in Z, i \neq j, R_i^k \cap R_j^l = \emptyset$, i.e., no robot $A_j \neq A_i$ ever occupies any cell that belongs to the resource requirements of z_i^k. Otherwise stage z_i^k is shared. We will consider event $e_i \in \Gamma(s^*)$ as admissible if the next state $s = f(s^*, e_i)$ is found *safe* by the following algorithm.

State safety test

Input:

Examined state $s = [s_1, \ldots, s_n]$

For all robots A_i and their all stages z_i^k:

– resource requirement $R_i^k \subset R$,

– stage type $Type(i, k) \in \{private, shared\}$

1. Set the content of the working robot set $A^* = A$
2. For each $A_i \in A$ find the nearest private stage, i.e., the smallest value $b_i \geq s_i$ such that $Type(i, b_i) = private$.

3. In A^*, find robot A_i such that no other robot in A^* is allocated any resource required by any stage between the current stage of A_i, $z_i^{s_i}$, and its nearest private stage $z_i^{b_i}$.
   IF robot A_i does not exist THEN EXIT {state s is not safe }
   ELSE $A^* = A^* \setminus \{A_i\}$.
4. IF $A^* \neq \emptyset$ THEN GO TO 3 ELSE EXIT {state s is safe}

Below, we prove that the restriction of event occurrence to those that are admissible ensures the reachability of the final state i.e., deadlock avoidance.

Theorem 1 *For $MMRS(Z, D)$ as in Definition 7, let $\Gamma(s)$ be restricted to only such events e_i that the potential next state $s' = f(s, e_i)$ is found safe by the state-safety test. Then the final state s_m is reachable from each state reachable from the initial one.*

Proof The algorithm finds state s' safe iff there exists an order of robots such that, one by one, they can move to their nearest private stages. Clearly, from such a state, the robots can, one by one, reach their final stages. □

In contrast to the above discussed models, the coaccesibility problem has an optimal, scalable solution for systems in FREE-RANGE*-2-RAS class (Roszkowska and Reveliotis 2013), described here by automata D-2-$MMRS$. Note that in this model, the events representing the end of the transition of robot A_i from one cell to the next one are associated with the release of the previous cell and require no additional resource. Thus, their occurrence cannot render the state unsafe. Consequently, we can merge each pair of events -representing the start and the end of the transition of robot A_i from one cell to the next one - and consider such a transition as one event e_i. Then, the set of stages Z only comprises the stages z_i^k associated with the presence of a robot in a cell, and their resource requirements are single cells defined by $D : Z \to W$. In this case, a well-defined pair (Z, D) requires that for each robot A_i, $i = 1 \dots n$, stages z_i^1 and $z_i^{m_i}$ are private, and for each $k = 2..m_i$, the resource requirements of stages z_i^{k-1} and z_i^k are vertically or horizontally adjacent cells $w = D(z_i^{k-1}) \in W$ and $w' = D(z_i^k) \in W$, respectively.

The automaton modeling this type of MMRS, denoted by D-2-$MMRS^*$, differs from D-2-$MMRS$ by the feasible event function $\Gamma(s)$ and is defined as follows.

Definition 7 For each well defined pair (Z, D), D-2-$MMRS^*$ is a deterministic automaton $G = (S, E, \Gamma, f, s_0, S_M)$ such that:

- the state set S, the event set E, the transition function f, the initial state s_0, and the marked-state set S_M are defined in the same way as in Definition 2 for $MMRS(Z)$
- the feasible event function is given by:

$$\Gamma(s_1, s_2, ..., s_n) = \{e_i : s_i < m_i \ \wedge \ \Sigma_{j=1}^n \mid (R_i^{s_i+1} \cap R_j^{s_j} \mid \leq 1\}$$

The feasibility and admissibility of the event occurrence in D-2-$MMRS^*$ can be considered in terms of the feasibility and admissibility of the allocation to the robots their next required cells. The algorithm for state-safety testing is based on the analysis of the *cell allocation graph* that is defined below and illustrated with an example depicted in Fig. 4.

Definition 8 A cell allocation graph G is a directed graph that depicts the state of the cell allocation and the nearest further needs of the robots. The vertices of G represent the cells, and the arcs of G represent the active robots. There exists arc $A_i = (w_k, w_l)$ in G iff robot A_i occupies cell w_k and the next stage of its route is cell w_l.

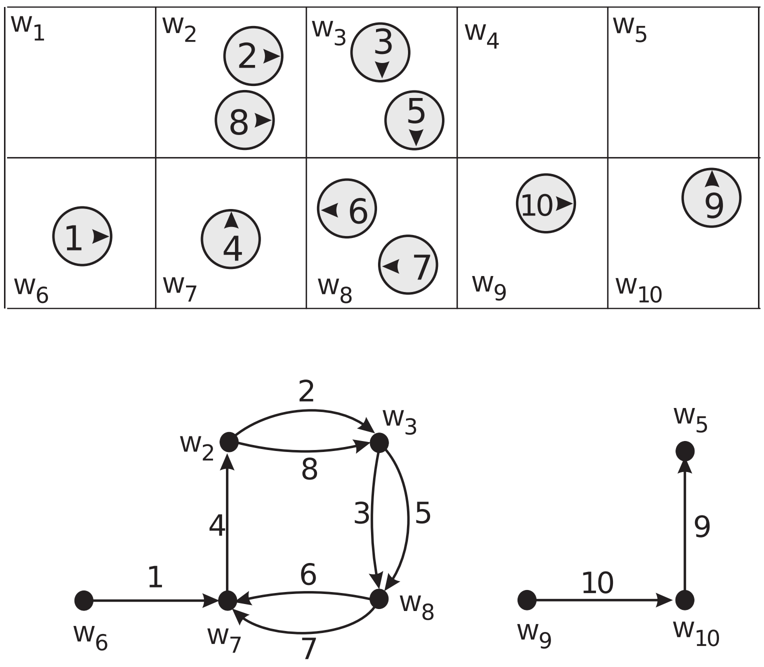


Fig. 4 An example resource allocation graph (lower part of the figure) corresponding to a group of robots executing their missions. The upper part of the figure depicts the distribution of the robots in the motion area as well as the directions towards their next cells on the route

Next we employ this concept for the feasible and admissible cell allocation to the robots.

Definition 9 Given a graph G and an arc $A_i = (w_k, w_l)$, allocation of cell w_l to robot A_i is *feasible* iff cell w_l has less than two outgoing arcs (w_l is allocated to less than two robots). Allocation of cell w_l to robot A_i is *admissible* iff any of the following conditions holds:

1. w_l is the last cell on the route of robot A_i, or
2. in graph G, w_l has no outgoing arcs (cell w_l is not allocated to any robot), or
3. in graph G', obtained from G by allocating cell w_l to robot A_i and deallocating cell w_k, there exists a path from vertex w_l to some vertex w_p that has no more than one outgoing arc (i.e., to cell w_p that is allocated to at most one robot)

If the allocation of the next required cell w_l to robot A_i is admissible in state s then the event e_i representing the transition of the robot from its current cell w_k to w_l will also be considered as admissible. This results in the following theorem.

Theorem 2 *For D-2-$MMRS^*$, let $\Gamma(s)$ be restricted to only such events e_i that are admissible in state s. Then the final state s_m is reachable from each state reachable from the initial one.*

Proof D-2-$MMRS^*$ is an automaton model of FREE-RANGE*-2-RAS systems considered in Roszkowska and Reveliotis (2013), where their liveness under the admissibility conditions is proved. This implies reachability of the final state (hence, also deadlock-free robot motion) also in this case. □

4.3 Architecture of the central MMRS supervisor

Figure 5 shows the architecture of the central supervisor implementing the established logic. It is assumed that the robots move autonomously within path sectors as well as they can cross a sector border if it is associated with event e_i causing a release of a resource. A transition to the next sector, associated with event e_i, that requires an additional resource, must be authorized by the supervisor. Issuing such a permission to robot A_i is a controllable event $next_i$. In addition, there are two observable events ap_i, and rp_i related to the achievement of certain points on the path by robot A_i, and the transmission of this information to the supervisor. Event ap_i reports the approach of robot A_i to the next sector and a request for permission to enter it. Event rp_i announces the transition of robot A_i's to the sector that does not require permission and the release of no more needed resources.

The supervisor maintains the state of MMRS, which is changed by event occurrence. Event $next_i$ can occur in state s if a prior event ap_i occurred and event e_i is admissible in s. If the supervisor decides to activate it, a message $next_i$ is sent to robot A_i and the state of the supervisor is updated to $s' = f(s, e_i)$. Receipt of a message of type rp announces a sector change by the robot that sent it, and is directly followed by a relevant state update. Receipt of a message of type ap causes placement of the robot that sent it in the queue of robots awaiting permission to enter next sector. The queue is sorted according to the adopted priority rule. If event e_i, where A_i is the the first element of the queue, is admissible in the current state s, a message of type $next$ is sent to robot A_i. Next, this robot is removed from the list of waiting robots and the state of the system is updated. Otherwise, the above process is repeated for the next robot in the queue.

5 Stage pass control

Implementation of the established logic raises the problem of how to prevent a robot from entering a new sector when it is has not received permission from the supervisor to do this. Moreover, whereas all the discussed models ensure the correct transition of robots between path sectors, in D-2-$MMRS$, collisions can occur among two robots that are allowed in a single cell. Both of these problems concern the behavior of the robots when passing through a given sector of the path, and are studied in this section.

5.1 Robot motion mode control

As mentioned in the previous section, the supervisor communicates with robots using messages, whose sending and receiving can be viewed as events. The supervisor generates only one event, *next*, and the robots inform the supervisor about reaching two characteristic points on their path sectors: ap - request to enter the next path sector, and rp - information about transition to a new sector, whose entering does not require permission. Additionally, each robot can sense the critical points cp, lying at the braking distance from its path sectors' ends.. Figure 6 depicts an automaton modeling the robot motion-mode control during its travel in a sector and transition to the next one. We distinguish four motion modes: *next sector*, *towards cp*, *deceleration*, *acceleration*, and *standstill*,

In the initial or the final state, and while waiting for the permission to enter the next sector, the robot remains in the *standstill* state. Occurrence of event $next$ allows the robot to start moving to the next path sector and results in the transition to state *acceleration*. The robot

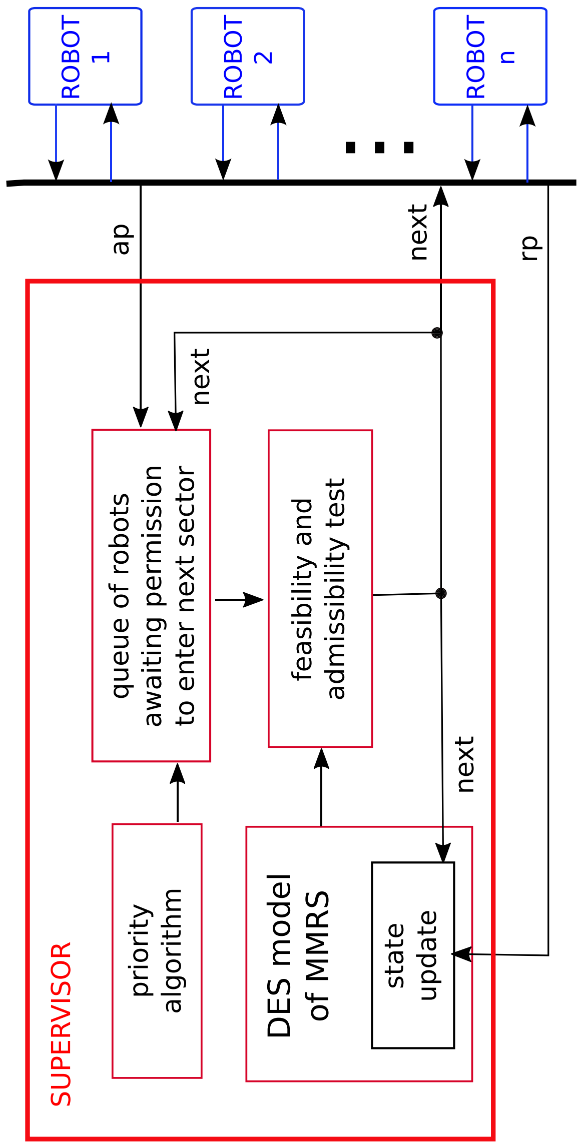


Fig. 5 Architecture of the central MMRS supervisor

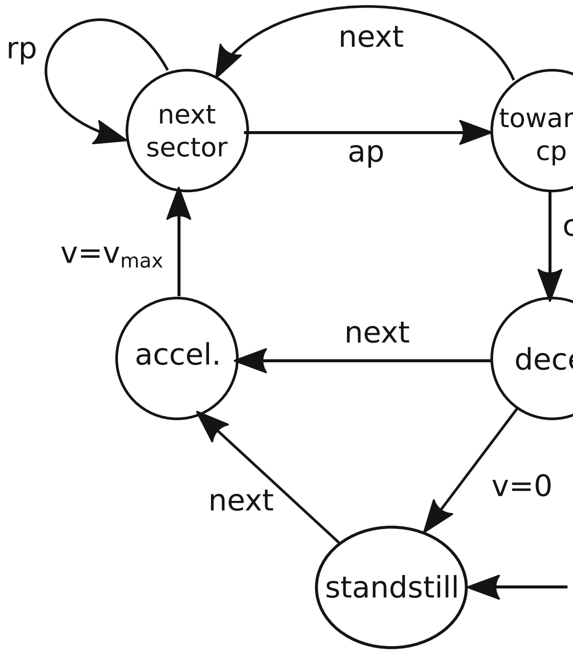


Fig. 6 Robot motion-mode control

remains in this state until the velocity $v = v_{max}$, and continues its motion with constant velocity in the next path sector. The occurrence of event rp signals that the robot has entered a consecutive sector (a sector that does not require supervisor's authorization). Occurrence of event ap signals that the robot is approaching a consecutive sector and asks permission to enter it. If the event $next$ occurs before the critical point cp is reached, the robot proceeds to the next sector. Otherwise, at point cp, the robot changes its motion mode to *deceleration* and starts awaiting signal *next*, which can occur after the robot has come to the standstill or still when decelerating. In both cases, having received the *next* signal, the robot starts accelerating and heads to the next sector.

5.2 Cell traverse control in *D*-2-*MMRS*

With the cell transition control discussed in the previous section, the problem of correct robot motion in D-2-$MMRS$ reduces to enabling collision-free movement of two robots within a cell. Achieving this goal requires an online policy to modify the paths of the robots, that takes into account the specifics of their missions. In this work, two such solutions are considered. In the case of *no-task cells*, i.e., when there are no particular locations that the robots are supposed to visit, we assume the establishment of virtual roundabouts to ensure the correct traffic. If, however, the robots must be able to reach some arbitrarily designated points in the motion area, i.e., in the case of *task cells*, the traffic on roundabouts is no more a satisfactory solution, and we focus on a reactive collision avoidance method, based on the Artificial Potential Field (APF) abstraction. These two solutions can also be combined if needed, as e.g. in the *puck collection system*. Then the robots traversing concurrently a cell with no pucks travel on roundabout, whereas the motion of robots in cells containing pucks is coordinated using the APF concept.

The role of the cell-traverse control is to supervise the entry of each robot into a new cell and its travel in the cell in the case when the cell is being shared with another robot and their paths are *conflicting*. The latter property means that the paths of the robots either cross in this cell or do not cross but go in the opposite direction and the minimal distance between them is shorter than the sum of the robots radii. Otherwise, that is if the robot is the only user of a cell or the robots' paths are not conflicting, no supervision of their motion is necessary. The supervisor communicates both, with the cell transition controller and the robot controllers, and takes action as soon as some robot A_i gets allocated a new cell w_k. The control takes the following steps.

General scheme of the cell-traverse control

1. If cell w_k is only allocated to robot A_i, or it is also allocated to another robot A_j but their paths in w_k are not conflicting then prompt A_i to enter w_k and continue its initial travel plan. Otherwise go to (2).
2. Since (1) is not true, there exists another robot, say A_j, that was allocated cell w_k earlier than A_i. If A_j has already been allocated its next cell, say w_l, then wait until it has finished the transit to w_l and go to (1). Otherwise go to (3).
3. If w_k is a no-task cell then apply roundabout-control. Otherwise go to (4).
4. Apply APF-control.

The control procedures called in steps (3) and (4) of this algorithm are discussed below.

Roundabout-control

The RC procedure is not called unless some robot A_i is permitted to enter a new no-task cell w_k that is already occupied by another robot A_j. The control takes the form of a supervisor that induces a change of the robots' motion modes. The robots are commanded to enter one by one the roundabout, head towards their cell exits, and either directly leave the circle and drive to the next cell or wait at the roundabout until such a transition is enabled by the cell-transition controller. If the way to the exit of the robot supposed to leave the cell first is blocked by the other robot then the latter is ordered to make another circle on the roundabout and thus let the former reach the exit. Moreover, it is assumed that the robots are equipped with proximity sensors, which allows them to keep the distance and avoid collisions between each other. Alternatively, the collision free motion can be enforced centrally based on a RAS model. In this case the roundabout is partitioned into sectors that represent resources, and the supervisor controls their mutually exclusive use. The left part of Fig. 7 shows exemplary robot paths on the roundabout. The actual shape of the path entry on the roundabout will depend on the kinematics of the robots.

APF-control

The APF-control procedure is called each time when some robot A_i is permitted to enter a new task-cell w_k that is already occupied by another robot A_j. In this case, one or both robots need to visit some specific locations in the cell and then head towards the entry to their next cells. The AFP concept deals with the potential field constituted by attractive fields and repulsive fields that induce attractive and repulsive forces, respectively. The robot is represented by a particle that is subjected to these forces, which results in its movement. In

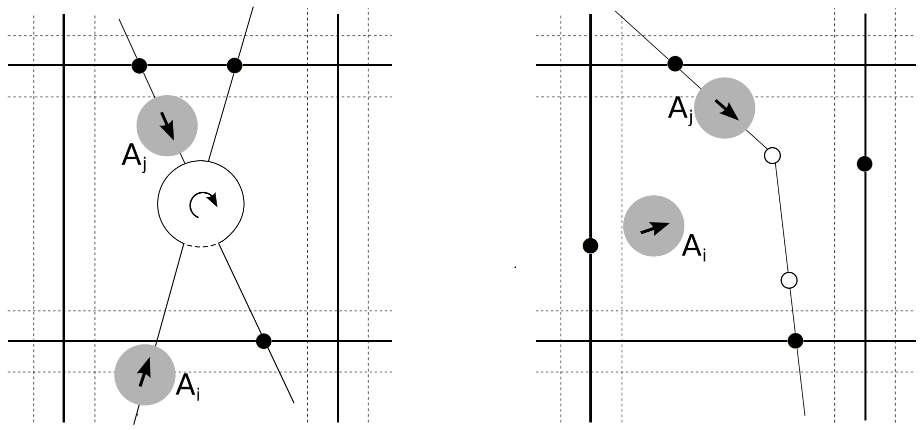


Fig. 7 Modification of the robot paths enforced by the cell-traverse control. Left figure shows the motion on roundabout. Right figure illustrates the APF-control assumption that robot A_j follows its path, while the path of A_i is determined in real-time by the combination of the attractive APF force of the goal and the repulsive APF forces of the borders and the other robot

planning the robot path, the attractive force is associated with the robot target, whereas the repulsive forces, with the obstacles. When using this abstraction for the coordination of two robots motion in a cell, we make the following assumptions.

- Robot A_i is subjected to the APF forces with the force interaction point placed in the center of the robot's disk.
- The *borders*, depicted by the dashed lines spaced from the sides of the cell by the radius of the disc (see Fig. 7) are considered as obstacles, which ensures that when traversing the cell, the robot is entirely contained in it. The repulsive force is only activated for the borders of w_k, that is of the cell currently traversed by robot A_i. During its transit to the next cell, the forces of both cells are active, and once A_i leaves entirely w_k, the repulsive forces associated with the borders of w_k get deactivated.
- The locations that robot A_i has to visit in cell w_k are considered its targets, and modeled as attractive potential fields that become activated in the order of their occurrence on the initial robot path. That is, if there is no puck to pick in cell w_k, the attractive force associated with the exit from w_k is immediately activated. Otherwise the attractive force of the first puck to collect is activated and kept active until the puck gets collected. The same is repeated for all the subsequent pucks and finally for the exit. Then the exit force is deactivated and the robot either directly transits to the next cell or stops and waits for the permission to continue its motion.
- The robots are considered as moving obstacle, generating a repulsive potential field, which prevents collisions between them.
- Robot A_j is not subjected to the APF forces (hence its path is not modified) unless at a standstill, while awaiting permission to enter the next cell, blocks the exit of robot A_i. Then the repulsive field of A_i is increased and the APF control for robot A_j is initiated, which makes it move aside.

When implementing this concept, the basic problem is the construction of smooth repulsive and attractive APF functions over the extent of the configuration space. Their combination should provide high values when the robot is near to an obstacle, lower values when it is further away, the lowest value at the desired goal location, and its increase when moving to configurations that are further away. Then the gradient of such a function, defining the resultant APF force, can be used to guide the robot to the desired configuration. The control algorithm consists in the repeated calculation of its value for the subsequent positions of the robot. The force vector indicates the direction of the robot motion at the current control step, while the velocity function can be defined arbitrarily. For example, one can choose to move at a constant speed until close enough to the target and then to scale the speed depending on the distance to the target. Based on Matoui et al. (2015), we chose the following functions to define the potential fields.

- Attractive Potential Field:

$$U_{att}(X) = \frac{1}{2}k\rho^2(X, X_g) \tag{6}$$

  where :
  $X = (x, y)$ – position of the robot
  $X_g = (x_g, y_g)$ – goal of the robot
  $\rho(X, X_g) = ||X_g - X||$ – distance from robot to goal
  K – positive scaling vector
  The function $U_{attra}(X)$ is positive or null and attains the minimum in goal point. Attractive force: $F_{attra}(X)$ is the negative gradient of the attractive potential field function:
- Attractive force:

$$F_{att}(X) = -\nabla[U_{att}(X)] = -k\rho(X, X_g) \tag{7}$$

- Repulsive Potential Field:

$$U_{repu}(X) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{\rho(X,X_o)} - \frac{1}{\rho_o}\right) \text{ if } (X, X_o) \leq \rho_o \\ 0 \text{ if } (X, X_o) > \rho_o \end{cases} \tag{8}$$

  where:
  $\rho(X, X_o)$ – the minimum distance between the robot the obstacles
  ρ_o – distance influence imposed by obstacles
  $\eta > 0$ – positive scaling factor
- Repulsive force:

$$F_{rep}(X) = \nabla[U_{rep}(X)] = \begin{cases} \eta\left(\frac{1}{\rho(X,X_o)} - \frac{1}{\rho_o}\right)\frac{1}{\rho^2(X,X_o)} \text{ if } (X, X_o) \leq \rho_o \\ 0 \text{ if } (X, X_o) > \rho_o \end{cases} \tag{9}$$

The values of the scaling factors K and η, as well as of the distance influence ρ_o need to be set experimentally. The force acting on robot A_i at a given position X is equal to the attractive force generated by the nearest goal and the repulsive forces generated by the cell borders and of the other robot, and can be calculated using the above defined functions.

$$F(X) = F_{rep}^{border}(X) + F_{rep}^{other-robot} + F_{att}^{goal} \tag{10}$$

Using the direction of the force vector at position X, the next position of the robot, X', is calculated at a unit distance from X, which corresponds to the assumption of a constant speed of A_i.

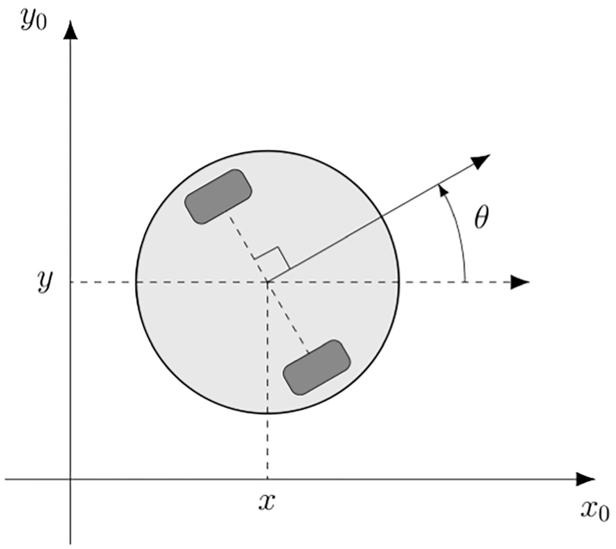


Fig. 8 State variables of the differential drive robot

6 Robot control

The low level robot control model reflects the specifics of the robot construction. In particular, we assume the implementation of the control concept presented in this work for differential drive robots, an example of which are Turtlebots. These robots have two separately driven wheels placed on either side of its body, and an additional, free of control wheel, used for their stabilization.

The state of the robot is represented with three variables: position (x, y) of the central point of the robot disk, and its orientation θ (see Fig. 8). There are two control variables: linear velocity v and angular velocity ω of the robot platform. The kinematic model of the robot is described by the following differential equation:

$$\dot{q} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \cos\theta \\ \sin\theta \\ 0 \end{pmatrix} v + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \omega \tag{11}$$

Then, the angular velocity of the robot is given by:

$$\dot{\theta} = \omega = \frac{v_r - v_l}{D} \tag{12}$$

where D is the distance between the centers of the wheels, and v_l, v_r are the linear velocities of the left and right wheels, respectively. The velocity of the robot is the average of the individual wheel velocities:

$$v = \frac{v_l + v_r}{2}, \quad \text{where} \quad v_l = v - \frac{D}{2}\omega, \quad v_r = v + \frac{D}{2}\omega \tag{13}$$

The low-level robot controller is responsible for the motion execution. It contains a local planner, which calculates the required wheel velocities given a path to be followed and the current robot position with respect to this path. In the case of differential drive robots, the angular and forward velocities can be calculated separately. The angular velocity is calculated based on the spline approximation to smoothly follow the path and to reduce any drift error if it appears. The forward velocity of the robot depends on the motion modes required to perform its mission and conform to the decisions of the higher level controllers. For the ‘accelerate’ and ‘decelerate’ modes it forms a trapezoidal velocity profile constrained with maximum values of velocity, regular acceleration and deceleration. In the case of the ’stop’ requirement, an emergency break is activated to immediately stop the robot.

7 Simulation experiments

The aim of the simulation experiments was a comparison of the system behavior under the established three supervisory control models. The research comprised three MMRS simulation instances, each of which aimed at the visualization of the concurrent travel of a group of robots along a specific set of paths as well as at the measurement of parameters such as time and distance traveled by the robots. The considered assignments for the robot fleet are delivery tasks specified by a set of locations to visit. The motion area was assumed to be a square with side length of 30 m and each cell with dimension $2m \times 2m$. Animations of the studied simulation instances are accessible at https://drive.google.com/drive/folders/1no1sHDCunRuXtzjt167SKy21hqW0g-ec?usp=sharing.

Experiment 1

The experiment included three instances, corresponding to the simulation of the robot movement along the same cyclic paths, but under the supervision based on the three discussed models, ξ-$MMRS$, D-$MMRS$, and D-2-$MMRS$. Figure 9 presents the considered motion area and its tessellation employed in D-$MMRS$ and D-2-$MMRS$. Next to the layout of

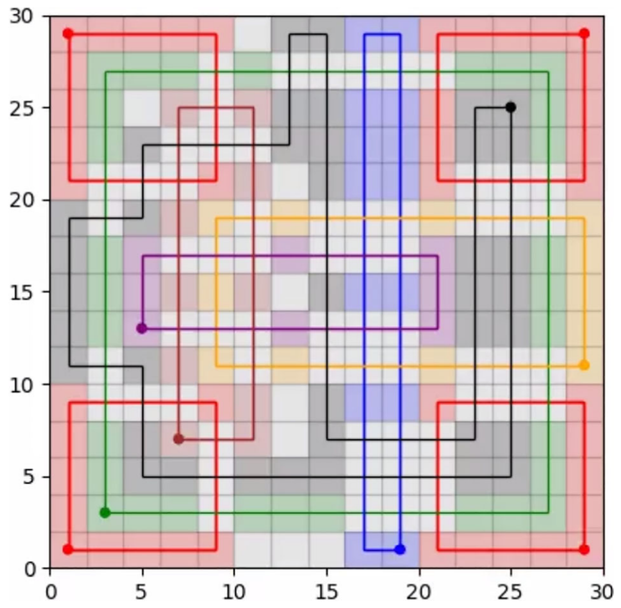


Fig. 9 Simulation instance 1 - 10 robots moving along cyclic paths. Motion space partitioned into cells with depicted robots’ paths and their private sectors (lighter shade of the path color)

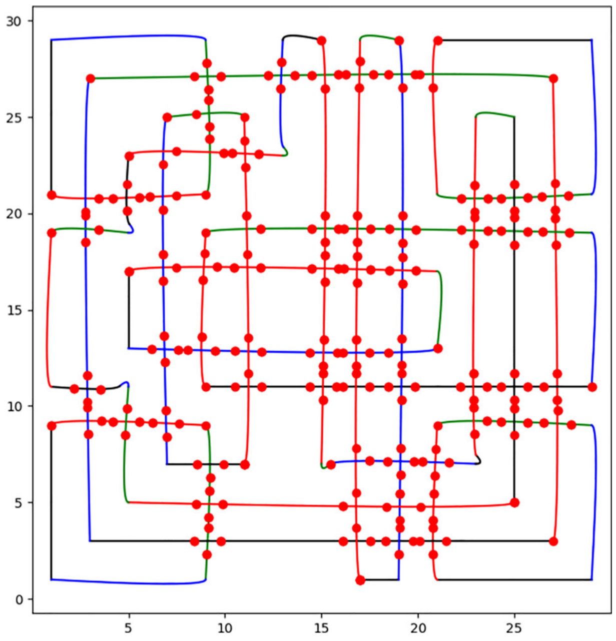


Fig. 10 Simulation instance 1 - 10 robots moving along cyclic paths. Approximation of the paths with Bezier curves and their analytically established partition into sectors with the minimum length of the conflicting stages

the robot paths, the figure depicts their private sectors (colored with a lighter shade of the path's color) and their shared sectors (uncolored) used in the state-safety algorithm. The ξ-$MMRS$ was established through path division based on the criterion of minimal length of the conflicting sectors. To enable analytical optimization, the paths of robots were presented in the mathematical form. Figure 10 depicts Bezier splines approximating the considered paths together with their partition into sectors. The simulation assumed the red robots to pass along their paths three times, the yellow, purple, brown and blue robots - twice, and the green and black robots - once.

Table 1 presents the makespan the makespan C_{max} (the completion time of the task completed at the latest) values obtained using the the three supervisory control models. In fact, the makespans values are very similar. The shortest time was achieved for D-2-$MMRS$, but the difference between the other cases is less than 2%. Moreover, the approximation of the paths with Bezier splines changed slightly their lengths.

Experiment 2

Table 1 Experiment 1: simulation results

model	C_{max} [s]	Total length [m]
D-$MMRS$	234	999.8
D-2-$MMRS$	230	1004.4
ξ-$MMRS$	235	1016.8

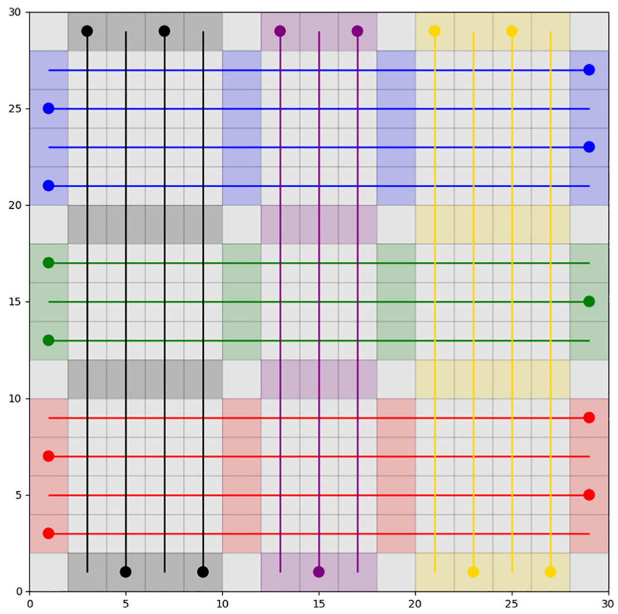


Fig. 11 Simulation instance 2 - the paths of 22 robots

The second scenario dealt with a system of an increased number of robots and their potential conflicts, and only concerned comparison of $D\text{-}MMRS$ and $D\text{-}2\text{-}MMRS$ supervisory control models. Figure 11 shows the paths of the robots and their initial positions. The paths of each robot had the same length, and each robots followed its path four times.

Table 2 presents the averaged results for 5 runs of each robot under the supervision of $D\text{-}MMRS$ and $D\text{-}2\text{-}MMRS$ models. The following two basic observations can be made.

- The average distance traveled in system $D\text{-}2\text{-}MMRS$ is about 0.84% longer than in $D\text{-}MMRS$.
- Robots' average task completion time is 10.87% lower in $D\text{-}2\text{-}MMRS$ compared to $D\text{-}MMRS$.

The obtained results are consistent with the expectations. The decrease of time in the case of $D\text{-}2\text{-}MMRS$ is due to both a better usage of the space (two robots in a cell) and a less restrictive state-safety test.

Experiment 3

A possible approach to minimize the makespan is to provide more robots to the system and divide between them long, perhaps cyclic, tasks. Figure 12 shows the path of five robots that are to be cyclically tracked a fixed number of times. The red, green, yellow, and cyan robots are supposed to make 5 cycles, and the navy-blue robot, traveling on the longest path, four cycles. The realization of the last task was examined with its assignment to one robot, and its division among two, three, and four robots., both for $D\text{-}MMRS$ and $D\text{-}2\text{-}MMRS$ control systems. The obtained results are presented in Table 3. One can observe that introducing one

Table 2 Experiment 2: simulation results

	D-$MMRS$		D-2-$MMRS$	
Robot ID	Distance [m]	Time [s]	Distance [m]	Time [s]
R1	111.09	311.258	111.28	292.628
R2	111.11	352.73	112.22	307.03
R3	111.08	319.52	111.31	298.28
R4	111.12	342.10	111.69	300.11
R5	111.10	350.84	112.48	312.91
R6	111.08	352.80	112.70	312.49
R7	111.07	328.99	111.98	298.90
R8	111.09	330.96	111.64	299.45
R9	111.082	306.57	111.50	300.17
R10	111.09	333.02	111.45	299.18
R11	111.09	307.53	111.24	290.39
R12	111.07	384.93	111.71	294.61
R13	111.14	342.62	111.10	295.35
R14	111.09	369.43	112.25	302.40
R15	111.08	351.62	111.85	300.56
R16	111.08	332.82	114.07	324.15
R17	111.10	311.58	114.29	315.37
R18	111.07	315.94	113.18	317.79
R19	111.10	361.31	111.50	296.34
R20	111.11	350.94	112.01	303.79
R21	111.08	354.28	111.52	295.90
R22	111.08	350.16	111.44	292.79
Average	111.09	339.18	112.02	302.30

more robot reduces the makespan by nearly half. Each additional robot does not have such a significant impact on shortening this time. For $D - 2 - MMRS$ model, an acceleration of 3-4% relative to the D-$MMRS$ can be noticed.

Other conclusions drown from the simulation experiments are as follows.

- Local maneuvers can make the robot paths longer, but only to a small extent in relation to the total path length.
- Two robots allowed in the cell not always help with the system speedup if simultaneous requests for resources are rare. Effect of this is significant when the instance is dense, with more agents in the motion space. The time gain is visible in systems with longer, perhaps cyclical tasks that could greatly benefit from the concurrent execution.
- Use of the potential field algorithm in the D-2-$MMRS$ requires careful adjustment of parameter values for forces between agents and resource barriers. Overly large values may cause problems in maintaining the state of resource allocations in the supervisor controller. This algorithm is computationally expensive and requires access to not always available information (positions of other robots, their velocities).

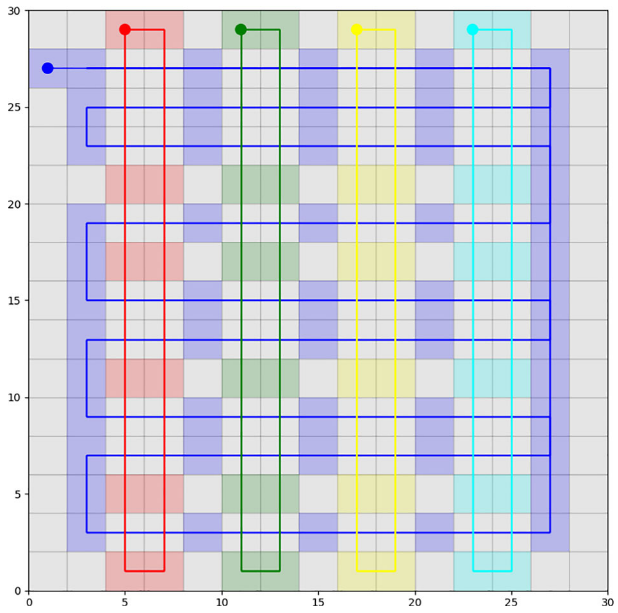


Fig. 12 Simulation instance 3: blue path is supposed to be traveled four times, and the remaining paths, five times

8 Conclusions

This paper contributes with a hybrid, three-level control system for MMRS, comprising a central supervisor that controls the robot transitions between the stages of their motion processes, local controllers that induce the robot's motion mode within a stage to suit the supervisor's decisions, and low-level robot controllers that are responsible for the required motion-mode execution. As can be noticed, the solutions proposed for the highest control level affect the solutions needed for the implementation of lower levels and, as a result, the operation of the entire system. The paper mostly focuses on the two upper levels, leaving the specific problems of robot control to be solved at the laboratory research stage. Within a common framework, we considered three supervisory control models that differ in the discretization

Table 3 Experiment 3: simulation results

model	Main task	n	C_{max} [s]	Total length [m]
$D\text{-}MMRS$	1 robot assigned	5	1960	2346.6
	2 robots assigned	6	955	2341.6
	3 robots assigned	7	636	2338.1
	4 robots assigned	8	508	2335.0
$D\text{-}2\text{-}MMRS$	1 robot assigned	5	1898	2360.4
	2 robots assigned	6	941	2362.3
	3 robots assigned	7	612	2387.8
	4 robots assigned	8	503	2403.1

of the robot continuous motion processes and/or the way of event occurrence restriction to provide collision-, and deadlock-free traffic. Next we proposed relevant solutions for the local controllers, and a general architecture of the supervisor that was implemented in a simulation system. We conducted three experiments to compare the impact of the different solutions on the performance of MMRS. The scenarios concerned transport tasks concurrently executed by a team of robots. The main observation was that type of MMRS model used does not affect the efficiency of the system as long as the layout of the robot paths is not very dense. Then some differences were observed. However, further research is needed to determine the influence of other system parameters on its performance, e.g., cell size, event prioritization policy, or local controllers' parameters. Other problems, such as, e.g., the impact of the robot positioning precision or the communication packets loss on the system performance, need to be established at the level of experiments with real robots.

Acknowledgements This research was partially supported by the National Science Centre, Poland, under the project number 2016/23/B/ST7/01441.

Declarations

Conflict of Interest The authors have no conflicts of interest to declare that are relevant to this article.

References

Andreasson H, Bouguerra A, Cirillo M, Dimitrov D, Driankov D, Karlsson L, Lilienthal A, Pecora F, Saarinen J, Sherikov A, Stoyanov T (2015) Autonomous transport vehicles: Where we are and what is missing. IEEE Robot Autom Mag 22(1):64–75

Barraquand J, Latombe JC (1991) Robot motion planning: A distributed representation approach. The Int J Robot Res 10:628–649

Buckley SJ (1989) Fast motion planning for multiple moving robots. In: Proceedings, 1989 international conference on robotics and automation, pp 322–3261

Cassandras C, Lafortune S (2010) Introduction to Discrete Event Systems. Springer, NY

Donald B, Xavier P, Canny J, Reif J (1993) Kinodynamic motion planning. Journal of ACM 40(5):1048–1066

Farrell M, Luckcuck M, Fisher M (2018) Robotics and integrated formal methods: Necessity meets opportunity. Integrated Formal Methods. Springer, Cham, pp 161–171

Ferrera E, Capitan J, Rodriguez Castano A, Marron P (2017) Decentralized safe conflict resolution for multiple robots in dense scenarios. Robot Auton Syst 91:179–193

Ferrera E, Castano A, Capitan J, Ollero A, Marron P (2013) Decentralized collision avoidance for large teams of robots. In: 2013 16th international conference on advanced robotics (ICAR), pp 1–6

Khatib O (1986) Real-time obstacle avoidance for manipulators and mobile robots. Int J of Robotics Res 8(1):90–98

Kress-Gazit H, Lahijanian M, Raman V (2018) Synthesis for robots: Guarantees and feedback for robot behavior. Annual Review of Control, Robotics, and Autonomous Systems 1(1):211–236

La Valle SM, Hutchinson SA (1998) Optimal motion planning for multiple robots having independent goals. IEEE Trans Robot Autom 14(6):912–925

Lin H (2014) Mission accomplished: An introduction to formal methods in mobile robot motion planning and control. Unmanned Systems 02(02):201–214

Lygeros J, Godbole DN, Sastry S (1998) Verified hybrid controllers for automated vehicles. IEEE Trans Autom Control 43(4):522–539

Matoui F, Boussaid B, Abdelkrim MN (2015)Local minimum solution for the potential field method in multiple robot motion planning task. In: 2015 16th international conference on sciences and techniques of automatic control and computer engineering (STA), pp 452–457

Pallottino L, Scordio VG, Bicchi A, Frazzoli E (2007) Decentralized cooperative policy for conflict resolution in multivehicle systems. IEEE Trans Robot 23:1170–1183

Pecora F, Cirillo M, Dimitrov D (2012) On mission-dependent coordination of multiple vehicles under spatial and temporal constraints. In: 2012 IEEE/RSJ international conference on intelligent robots and systems, pp 5262–5269

Peng J, Akella S (2004) Coordinating Multiple Robots with Kinodynamic Constraints along Specified Paths. Springer, Berlin, Heidelberg, pp 221–237

Reveliotis SA (2005) Real-Time Management of Resource Allocation Systems. Springer, NY

Reveliotis SA, Roszkowska E (2011) Conflict resolution in free-ranging multivehicle systems: A resource allocation paradigm. IEEE Trans Robot 27(2):283–296

Roszkowska E (2005) Provably correct closed-loop control for multiple mobile robot systems. In: Proceedings of the 2005 IEEE international conference on robotics and automation, pp 2810–2815

Reveliotis SA, Roszkowska E (2010) On the complexity of maximally permissive deadlock avoidance in multi-vehicle traffic systems. IEEE Trans Autom Control 55(7):1646–1651

Roszkowska E, Reveliotis SA (2013) A distributed protocol for motion coordination in free-range vehicular systems. Automatica 49(6):1639–1653

Soltero DE, Smith SL, Rus D (2011) Collision avoidance for persistent monitoring in multi-robot systems with intersecting trajectories. In: Proceedings of IEEE/RSJ international conference on intelligent robots and systems, pp 3645–3652

Tomlin C, Pappas GJ, Sastry S (1998) Conflict resolution for air traffic management: a study in multiagent hybrid systems. IEEE Trans Autom Control 43:509–521

Elzbieta Roszkowska holds the M.S. degree in computer science and the Ph.D. and D.Sc. (habilitation) degrees in automation and robotics. She is a professor at the Department of Cybernetics and Robotics, at Wroclaw University of Science and Technology (WUST), Poland, involved in both research and teaching. Her professional interest is in the event-based and hybrid control for complex systems with a specific reference to automation and robotics applications. She coordinated a number of research projects, the recent of which has been nominated to the award granted by Polish Center of Intelligent Development. As one of the creators of a new field of study at WUST - Embedded Robotics, prof. Roszkowska acts currently as its director. She also serves as expert in national and international bodies, including European Commission and industry. Elzbieta Roszkowska is a senior member of IEEE, worked two terms as an associate editor in the IEEE Transactions on Automation Science and Engineering, and has been a PC member of many international conferences.

Piotr Makowski-Czerski was born on August 8, 1996 in Zielona Gora, Poland. He received the Automation and Robotics Engineer degree and M.Sc degree in Embedded Robotics from the Wroclaw University of Science and Technology, Wroclaw, Poland, in 2019 and 2022 respectively. He is currently working as a robotics software engineer. His main interests include control of a fleet of autonomous mobile robots.

Łukasz Janiec holds a Master's degree in Control Engineering and Robotics from Wrocław University of Science and Technology, specializing in Robotics. He is currently a PhD Student at the Department of Cybernetics and Robotics at the same university. His research interests focus on investigating the control of multiple mobile robot groups, their coordination, collision avoidance, and deadlock prevention, as well as the use of ROS 2 in robotic simulations and experiments.