Advances in Science and Technology Research Journal

Advances in Science and Technology Research Journal, 2025, 19(1), 406–418
https://doi.org/10.12913/22998624/195453
ISSN 2299-8624, License CC-BY 4.0

Received: 2024.09.13
Accepted: 2024.11.19
Published: 2024.12.01

Cooperative and individual planning strategies for autonomous robots in warehouse parcel transportation

Tomasz Grzejszczak[1*], Witold Nocoń[1]

[1] Faculty of Automatic Control, Electronics and Computer Science, Silesian University of Technology, Akademicka 16, 44-100 Gliwice, Poland
* Corresponding author's e-mail: tomasz.grzejszczak@polsl.pl

ABSTRACT

The paper presents a simulation investigation into the individual time of parcel transportation in a multi-agent robot operated warehouse for two different artificial intelligence planning strategies (independent and cooperative), two different warehouse setups and different number of robots operating at the same time. The robots are assumed to operate in a non-communicating manner, and their only physical interactions are the collision detection and avoidance maneuvers. It is additionally assumed, that every robot is aware of its location, is able to compute the shortest path to the required destination and has access to a global orders list from which orders can be reserved and removed. The warehouse setup and robots movement is simulated using a cellular automata paradigm, and the simulation is implemented in Python. It has been shown that the cooperation of robots passing packages to each other significantly improves the overall work time needed to complete the task and reduces the number of empty runs compared to an individual approach.. The presented simulation results suggest it is possible to design a multi-robot delivery system in such a way, that robots do not communicate with each other and that no central planning or global optimization center is needed. The novel cooperative planning algorithm brings a contribution in engineering solutions, mainly for robot development in automated warehouses.

Keywords: cooperative planning, multi-agent, autonomous robots, decentralized system, PDDL.

INTRODUCTION

The world trends in shopping are shifting towards individual online orders, especially as a result of the pandemic. This requires further development and automation of warehouses. The automated warehouses with Automated Guided Vehicles are working in real life, however classic approach in a delivery order is inseparable, from shelf to drop zone. There are optimization methods described in literature, where the distances are minimized by better warehouse space management or by joining orders. This article describes the new approach that is based on dividing orders into smaller parts and pass the parcels between multiple robots that cooperates with each other. Cooperative planning can bring further development and optimization, decreasing the operation time within warehouse movement.

Overview

The problem of automated warehouses and their optimization is frequently researched. The optimization process can be divided into 6 subcategories: Mixed-shelves storage; batching, zoning, and sorting; dynamic order processing; AGV-assisted picking; shelf-moving robots; advanced picking workstations [1]. The common aim of those approaches is to minimize the time to get to parcel and to bring it to drop zone while minimizing the number of people involved in the process. Among those approaches, one is worth noticing. In batching, zoning, and sorting, a further reduction of the picking effort is enabled, if the warehouse is partitioned into disjoint zones. Order pickers only pick the part of an order that is stored in their assigned zone. Parallel zoning enables parallel processing of orders, thus faster

order processing [1]. Going one step further, instead of dividing warehouse into zones, packages can be left in midway for another agent to pick them up. This approach requires cooperation between agents.

On the other hand, a classical pick and place approach does not require cooperation and can be performed individually by each agent. Thus the contribution of this paper is a cooperative algorithm that is compared with an individual algorithm.

There is a relevant research stream that is a variant of the multi-agent pickup and delivery problem (MAPD) [2, 3], which is an extension of the broadly studied multi-agent path finding (MAPF) or a version of decentralized MADF [4, 5]. While the decentralization is similarly explained and justified in those approaches, the cooperation is more understood as negotiation when selecting orders. None of the solutions found analyze the possibility of abandoning the order during execution so that it can be taken up by another agent. All found solutions are of the pickup-and-delivery type.

The cooperative algorithm described in this paper is based on the Bucket Brigade approach, where the main idea is to feed the buckets to another person in a row in order to fight the fire more efficiently. This approach is implemented mostly in multi agent scheduling [6, 7] and is also used in other research fields for energy consumption optimization [8]. Moreover the developed algorithms are based on theory from cellular automata [9] for movement implementation, topological model for costs calculation in world map representation and STRIPS or PDDL [10, 11] for implementation of planning algorithms.

The main robotic unit or an agent of automated warehouse is a AVG (automated guided vehicle). In the research of AGVs, the main focus is on overcoming uncertainty and fault-tolerance [12] or on optimal trajectory planning [13]. As new practical developments in sensors and robot control technology were rapidly adopted, advanced AGV systems gradually emerged. These systems eventually created a new class of (driverless) vehicles called autonomous mobile robots (AMRs) [14].

Planning in artificial intelligence is the way of decision-making actions performed by robots to achieve a specific goal. Further division in planning methods are into mathematical methods (exact and heuristics), simulation studies, meta-heuristic techniques and deep learning based approaches [15]. On the other hand, there is a division into online (dynamic) [16, 17] and offline planning [10, 11, 18]. Since the warehouse model assumes autonomous robots, the decisions should be made in real time, thus offline planning is not applicable. Having this assumption, scheduling is also not applicable in this solution.

The definition of a multi agent system requires agents to be autonomous and work in decentralized manner [19]. Moreover, in a multi-agent environment, an agent's behavior can be defined as cooperative if the common utility is conscious and willingly increased compared to the referential utility [20] or cooperatively executing the same algorithm [21].

Contrary to cooperation, there is a wide research on conflicts handling in competitive (usually solved with game theory [11, 12]) or planning in swarm [22]. On the other hand, cooperative algorithms can be found in the traveling salesman problem, as the extension with multiple salesmen [18, 23]. However this approach is focused on route minimization and does not provide for passing parcels to each other.

Contribution

The planning algorithm is the important part of a robots' artificial intelligence system. The approach used in the presented paper is based on the assumption, that robots are making their own autonomous decisions. There is no central planning unit and robots cannot communicate with each other. Such hard assumption is in fact formulated based on the desire to strive for robots' autonomy. The only one-to-one interaction is the robot collision avoidance. Additionally, robots have access to a global orders list, with states cargo units with their location and intended destination. Such approach is employed in order to test a greatly scalable configuration of robots that do not require any reconfiguration of the system when robots are added or removed.

Two approaches of robot planning in the domain of decentralized autonomous robots are presented. In the first approach each robot takes a cargo and delivers it directly to the final destination. Such strategy is referred to as the independent strategy in this paper. In the cooperative strategy, robots are trying to pass cargo to other robots and also pick up cargo left by other robots. The physical construction of warehouse limits the movement

of robots, so that collisions and traffic jams occur and need to be handled. Additionally, due to the nature of the physical model [24] on which the presented simulation is based, movement in a straight line is significantly faster than turning.

The model of a warehouse on which the simulation is based and two planning algorithms are presented in chapter 3, together with the details of both strategies. After the set of experiments (chapter 4), the paper is summarized with conclusion about the advantages of cooperative planning algorithm.

PROBLEM FORMULATION

Presented investigations are based on a tabletop model described in [24]. The physical model of AMR (Fig. 1) is equipped with a set of encoders for robot and gripper positioning, engines for forward and turning movement, and a processing unit for orders management according to selected planning algorithm. Robots are moving on railway tracks and each robot is capable of moving forward and backward. Change of direction is only possible when the robots is standing on the turntable. Every robot is equipped with a manipulator for picking up and dropping of cargo from both sides (left and right). Each robot is independently programmed using an onboard computer.

The simulation constraints were measured form the real model [24]. In particular, the turntable design results in slow change of direction with respect to fast forward driving. This is due to precise 90 degrees rotation positioning. This constraint is the basis for designing the cooperating algorithm, as it is supposedly advantageous for the robot to pass cargo from one table to the next table for the next robot to pick it up, rather than travel to the destination and spending time changing direction with each individual parcel.

Those constraints result in a problem of global optimization, but to maintain scalability of the design, robots are intended to plan their actions locally, without any global commands issued by the optimization center.

The aim of the research is to investigate the total task competition time and execution time of each robot in multi robot environment depending on number of robots, chosen algorithm (cooperative / independent) and depending on warehouse configuration. The measured quantities are described in detail in Chapter 5.

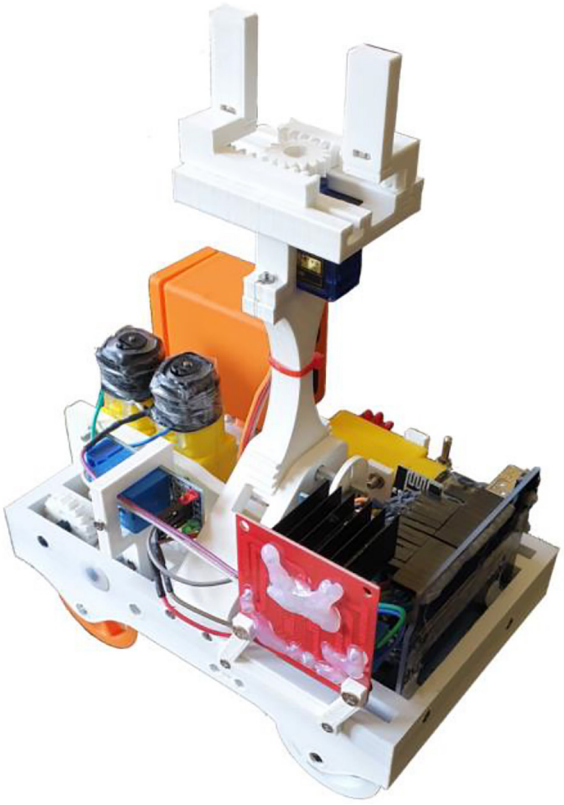


Figure 1. Physical model that inspired research [24]

SYSTEM MODELING AND CONTROL

World formulation and warehouse model

The simulation is defined with use of formal notation tuple $< P, O, W >$ similar to PDDL or STRIPS notation, containing set of predicates P, operators O and the world W that is a warehouse model. In the simulation world, there is a warehouse $W[X_W, Y_W]$ of set width X_W and height Y_W. The warehouse contains a number of robots R, tracks T, storage shelves S, cargos C and destination D, so $W = \{R, T, S, C, D\}$. The warehouse is visualized in Figure 2.

There are N_C cargos in the cargos set, where each cargo has its position in the warehouse, so

$$C = \{C_1(x_{C1}, y_{C1}), C_2(x_{C2}, y_{C2}), \ldots, C_{NC}(x_{CNC}, y_{CNC}) \quad (1)$$

and $x_C \in X_W$, $y_C \in Y_W$. Similar definitions are formulated for T, S, D and R.

There are also additional parameters. Each cargo has a defined desired destination, each robot has a planning algorithm and each track has an occupancy state. This set of parameters are

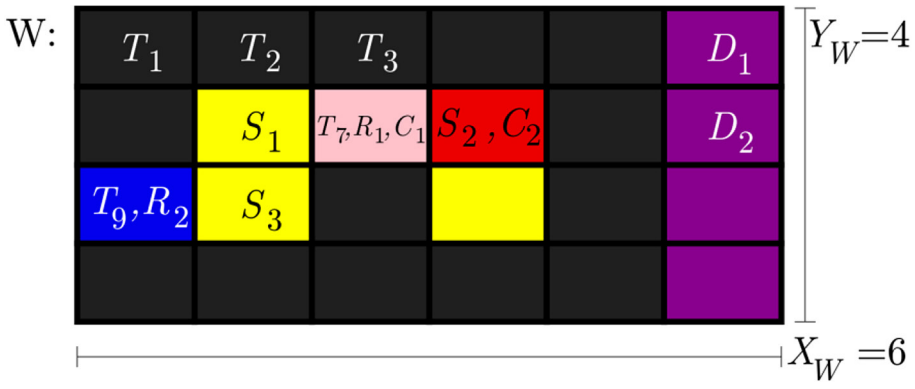


Figure 2. Exemplary warehouse model $W = \{R, T, S, C, D\}$

defined as world predicates P. The list of predicates is (A and B are predicate's parameters):

- *TO* (C_A, D_B) - Cargo's A desired position is B
- *EMPTY* (T_A) - There is no robot on track A, so a robot that would like to go there can do so
- *ON_SHELF* (C_A, S_B) - Cargo A is on shelf B
- *CARRYING* (R_A, C_B) - Robot A is carrying cargo B
- *ALGORITHM* (R_A, B) - Robot's A planning algorithm is defined as B

Topological model

For the purpose of planning of the shortest paths by each robot, a graph representation of the world is needed. Therefore the cellular automata world is translated into a weighted graph (Fig. 3). The size of the graph and number of shelves loops are adjustable and dependent upon warehouse W size (X_W, Y_W). Nodes T and P are represented separately for better graph readability.

Tracks connection is straightforward to represent in graph form. Two main challenges are the turntable and the procedure of cargo picking / dropping. The turntable is designed in such way, that passing straight ahead is fast, however turning 90° is time consuming because it requires additional actions and proper positioning. This is why the turntable track was split into 4 nodes with small cost of going straight g_s and high cost of turning g_t. The package handling is designed in such way, that the task of package picking / dropping requires navigation to node adherent do track node. The cost is a sum of positioning time g_p and manipulator operation time g_m. In order to create an easier, non-directional graph, the edge value is set to average of those two time measures as $(g_p + g_m)/2$.

The edge weights of graph g were experimentally measured from physical model's operation time [24] and are set to: 90° turn g_t = 22; cargo handling = 9 (where positioning g_p = 5 and manipulation g_m = 4); move forward to next track g_f = 1; turntable straight ahead g_s = 1. Weights are measured in unit cycles and are proportional to task execution time.

Operations

Robots are capable of performing the following primary operations O:

- *MOVE*(R_A, T_B, T_C) – Move robot A form track B to track C.
- *PICK_CARGO*(R_A, C_B) – Pickup cargo B by robot A

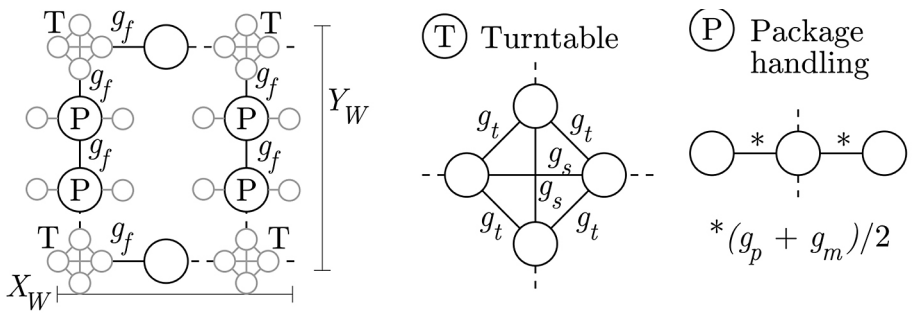


Figure 3. Warehouse graph model of a single shelf loop

- *DROP_CARGO(R_A, D_B)* – Drop cargo that is transported by robot *A* to a destination *D*
- *TRY_DROP_RIGHT (R_A)* – Drop cargo to the right of robot *A* if the shelve on the right of robot position is empty, that is for any *C*, ¬*ON SHELF(C, S(x_{RA} + 1, y_{RA}))* (symbol ¬ means negation).

Primary operations are grouped into tasks. A generation of a task requires the robot to plan a sequence of primary operations. The following planning tasks are available:

- *GOTO_PICK(R_A, C_B)* – Find a plan for picking up cargo *B* by robot *A*
- *GOTO_DROP (R_A, D_B)* – Find a plan for delivering a cargo to delivery *B* by robot *A*
- *RETURN_TO_ORIGIN (R_A)* – Find a plan to return to a starting position
- *DROP_RIGHT (R_A, C_B)* – Find the place to drop a cargo *B* on the available shelf
- *WITHDRAWL_MOVE(R_A)* – In case of deadlock, perform an evasive maneuver having a random number of steps, to make sure the deadlock with other robot is resolved

Planning algorithms

At this point of model development, each robot can be programmed to use one of two planning strategies: an individual (Table 1) or cooperative (Table 2) strategy. Both algorithms are highlighted with use of pseudocode. The most important part of each algorithm is the planning part with path finding algorithm with resulting path *P*. Its aim is to find a set of coordinates of tracks

$$P = [T(x_0, y_0), T(x_1, y_1), T(x_2, y_2), \ldots, T(x_i, y_i)] \quad (2)$$

that connects the source with destination, depending on how the source $T(x_0, y_0)$ and destination $T(x_i, y_i)$ is defined.

Both algorithms share two collision avoidance parts with checking if track *T* indicated by next element of *P* is occupied. In case that a robot tries to move to occupied track, there is a counter set to random number of cycles that immobilizes robot. If the move is not possible after this time, the robot stops executing the current plan and tries to find another path. The random number of cycles should prevent deadlocks. If two robots try to head to the opposite directions and decide to wait the same amount of time, they will both chose different paths and a collision may occur in a different place. In case of random time, one robot will likely head in another direction while the robot that waits longer will be able to proceed. This is also the reason why the waiting time for robots with cargo are higher (5-10) than for robots without cargo (1-5), because delivering is more important than finding cargo. Additionally, withdrawal maneuvers are incorporated, for cases when robots are unable to resolve a conflict of movement. In such a case, each robots drives away from its destination by a random number of steps, and then plans the task at hand again.

The main difference between algorithms is that a robot with individual strategy finds cargo

Table 1. Algorithm 1 individual algorithm

	Require: $R_n(x_{Rn}, y_{Rn})$, $C_m(x_{Cm}, y_{Cm})$, $D_m(x_{Dm}, y_{Dm})$ **Ensure**: $(x_{Cm}, y_{Cm}) = (x_{Dm}, y_{Dm})$
1:	For given *n* find *m* that indicates C_m closest to R_n
2:	*GOTO_PICK(R_n, C_m)* : Find shortest path P_1 from (x_{Rn}, y_{Rn}) to $(x_{Cm} + 1, y_{Cm})$
3:	**for** each element *i* of P_1 **do**
4:	**if** $T_{P1i} \neq$ empty **then**
5:	wait (random(1–5) cycles)
6:	**Go to**: 1
7:	**else**
8:	*MOVE(R_n, $T(x_{Rn}, y_{Rn})$, $T(P_1)$)*
9:	**end if**
10:	**end for**
11:	*PICK_CARGO(R_n, C_m)*
12:	*GOTO_DROP(R_n, D_m)* : Find shortest path P_2 from (x_{Rn}, yRn) to $(x_{Dm} - 1, y_{Dm})$
13:	**for** each element *i* of P_2 do
14:	**if** $T_{P2i} \neq$ empty **then**
15:	wait (random(5-10) cycles)
16:	Go to: 12
17:	**else**
18:	*MOVE(Rn, $T(x_{Rn}, y_{Rn})$, $T(P_{2i})$)*
19:	**end if**
20:	**end for**
21:	*DROP CARGO(R_n, D_m)*

Table 2. Algorithm 2 cooperative algorithm

Require: $R_n(x_{Rn}, y_{Rn})$, $C_m(x_{Cm}, y_{Cm})$, $D_m(x_{Dm}, y_{Dm})$
Ensure: $(x_{Cm}, y_{Cm}) = (x_{Dm}, y_{Dm})$

```
Find alley x_a with max( Σ ON SHELF (C_*, S_(xa−1,*)) – Σ ON_SHELF (C_*, S_(xa+1,*)))
Find m that indicates first Cm with x_Cm = x_a − 1
GOTO_PICK(R_n, C_m) : Find shortest path P_1 from (x_Rn, y_Rn) to (x_Cm + 1, y_Cm)
for each element i of P_1 do
   if TP_1i ≠ empty then
      wait (random(1–5) cycles)
      Go to: 1
   else
      MOVE(R_n, T(x_Rn, y_Rn), T(P_1i))
      end if
end for
PICK_CARGO(Rn, Cm)
TRY_DROP_RIGHT (R_n) : Find y_e with ¬ON_SHELF (C(x_a +1, y_e)), S(x_a + 1, y_e))
if TRY_DROP_RIGHT (R_n) failed then
   repeat with x_a = x_a + 2
end if
GOTO_DROP (R_n, D_m): Find shortest path P_2 from (x_Rn, y_Rn) to (x_a, y_e)
for each element i of P_2 do
   if T_P2i ≠ empty then
      wait (random(5-10) cycles)
      Go to: 17
   else
      MOVE(R_n, T(x_Rn, y_Rn), T (P_2i))
   end if
end for
DROP CARGO(R_n, (x_a, y_e))
```

and delivers it to the ultimate destination, while a robot with cooperative strategy heads to one alley and shift cargo from left to right, similar to bucket brigade strategy described before. This is introduced in Algorithm 2 as a function of finding the coordinate x_a of alley with maximum occupied shelves on left and empty shelves on right and a function of finding closest empty shelve coordinate y_e on the right of the robot. In the case that there is no empty shelves on the right, robot tries to switch alleys to the right. In case of every shelve occupied, it moves to the most right alley with a drop zone, where drop zone is always empty.

EXPERIMENTAL SETUP

Simulation of the presented setup has been implemented in Python, with a graphical user interface (GUI) designed in Tkinter library. The warehouse is modelled as a world $W[X_W, Y_W]$ implemented as an array of cells. Each cell can hold any object derived from the *Cell()* class. Those objects represent robots (blue), robots with cargo (pink), tracks (black), storage shelves (yellow), cargos (red) and destinations (purple). All objects implement a virtual method *run()*, the execution of which calculates the action of a cell depending on neighboring cells and its current state. Each simulation step executes *run()* methods of all the cells. Cargo objects may only occupy a cell previously occupied by a storage shelve or destination, while picking up of a cargo changes the visualization to a shelve in place of the picked up cargo. When cargo is picked up, the robot remembers the reference to the picked up cargo object and removes this object from the world. If cargo is dropped on to a shelve, it's reference is copied to the world in place of a shelve. If cargo is dropped to a destination, the destination adds this reference to its own list of all cargo dropped to this destination.

Robot logic is implemented in its *run()* method. When planning of action is needed, a series of tasks is placed in Task queue. Once a new task is taken out of this queue, a task planning routine is invoked, and places a series of primary operations in Moves queue. Therefore, each time, the *run()* method is executed, one move is taken from the Moves queue and executed. Additionally, moves have different execution times in real world. Therefore, an additional counter is used to implement moves taking more than one simulation iteration.

An example of the initial state of the warehouse is presented on Figure 4 and is of size 11 × 7 that results in 6 alleys and 5 tables (5 + 6 = 11) and each table with 5 places to put package (5 + 2 = 7). Robots are parked at the bottom-left side, destinations are in the rightmost alley (marked purple)

Figure 4. Initial set up of the warehouse containing cargo for delivery to destinations

and all shelves are occupied by cargo addressed to particular destinations (numbers on red).

Once simulation is started, robots execute their planning algorithm and start moving. Figure 5 presents a state in which robot 3 has picked up cargo addressed to destination 3, robot 2 picked cargo addressed to address 1 and robot 1 has already dropped its cargo and is driving to pick up another.

When all cargo is transferred to its destinations, robots return to their parking positions, all cargo are in their destinations and all shelves are empty (Fig. 6)

EXPERIMENTAL RESULTS

The presented strategies have been tested using multiple simulation runs for two different warehouse configurations presented in Figure 7. Configuration 1 has 5 columns of tables with 10 positions in each column (resulting in W = (11, 12)), while configuration 2 has 10 columns of tables with 5 positions in each column (resulting in W = (21, 7)). In both configurations, drop off destinations are located on the right end of the warehouse (12 in setup 1 and 6 in setup 2). In

Figure 5. Initial set up of the warehouse containing cargo for delivery to destinations

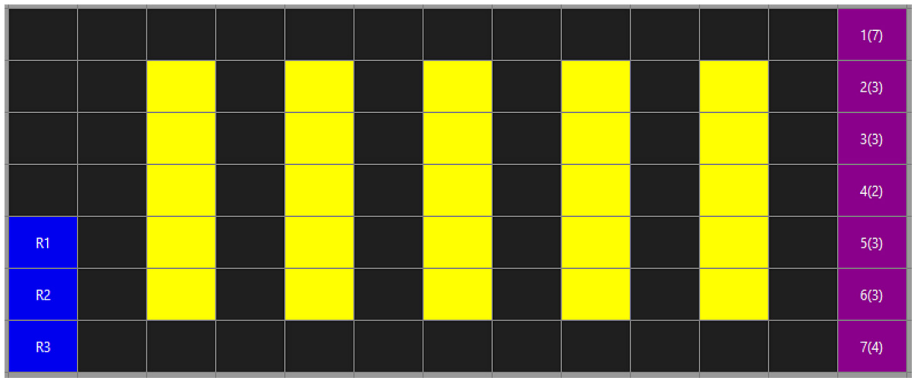
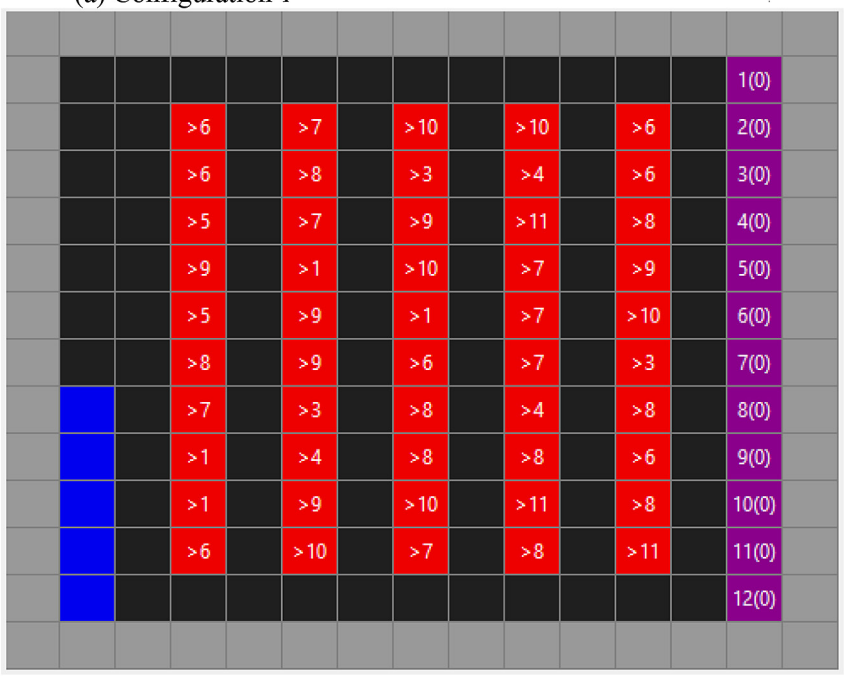


Figure 6. Final state of the simulation with all shelves empty, and all cargo in destinations

(a) Configuration 1



(b) Configuration 2

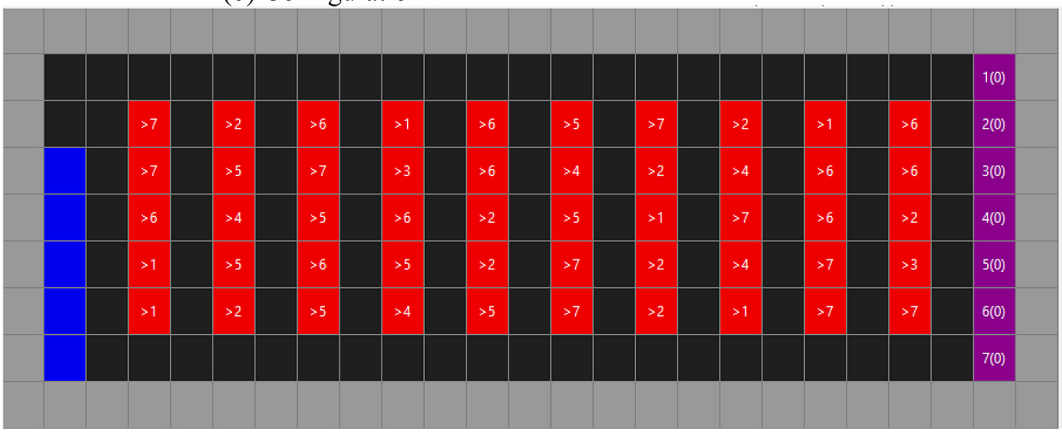


Figure 7. Initial set up of the warehouse containing cargo for delivery to destinations(a) Configuration 1: warehouse of size 5 × 10 (W = (11, 12)), (b) Configuration 2: warehouse of size 10 × 5 (W = (21, 7))

each simulation run, all tables are filled with cargo having a random destination. Due to this introduced randomness, each simulation is repeated 10 times for different setups and for different number of robots. A minimum of 1 robot and a maximum of 5 robots working at the same time is assumed

Results are investigated based on the following quantities. First, total time T_{total} needed to distribute all cargo is computed and is represented by the number of simulation steps needed to

complete the whole distribution task. Secondly, average robot execution time T_R is quantified as the average number of moves (cycles) performed in the warehouse by each robot and a total robot execution time of all robots T_{Rtotal} is quantified the same way but for all robots combined. Thirdly, energy wastage factor $\%E_U$ is defined as factor of empty runs, that is the percentage of step performed without cargo on board.

In the presented results, there is a separate plot for each warehouse setup: configuration 1 (5 × 10) and configuration 2 (10 × 5). In each plot the robots' algorithms are compared. Each warehouse configuration was solved for different number of robots in warehouse: $R_N = \{1,..., 5\}$. Each experiment was repeated multiple times, thus the plots present average numbers. Each obtained test result is presented for each configuration and an average function that shows the trend.

For configuration 1 the cooperative strategy surpasses the independent strategy in the total time needed for the completion of the distribution

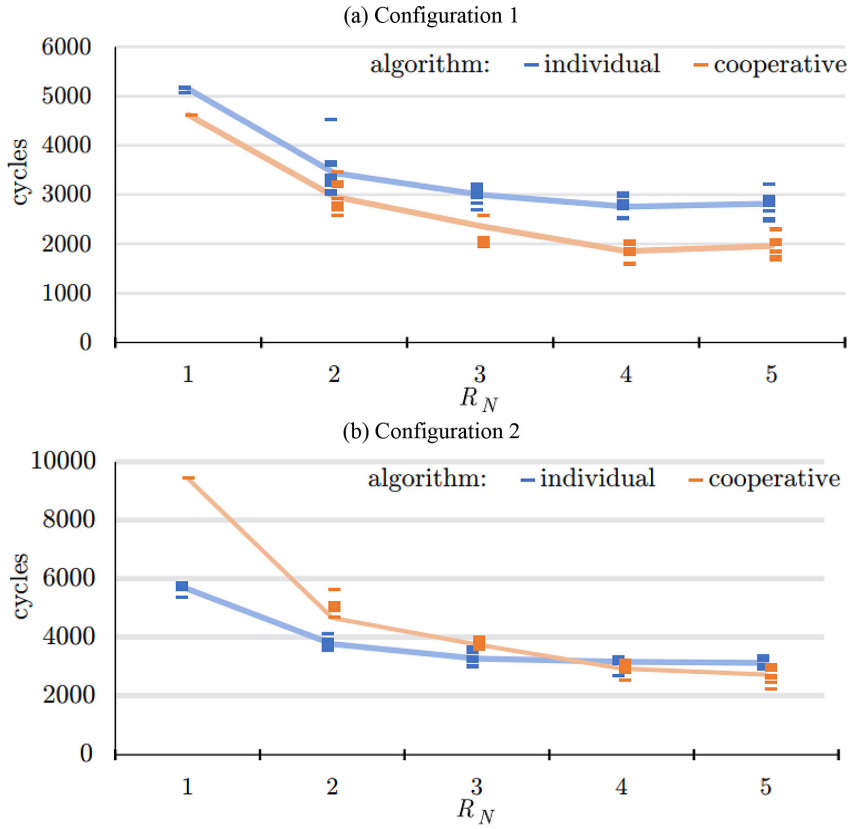


Figure 8. Total time T_{total} to complete the whole distribution task (a) Configuration 1, (b) Configuration 2

task (Fig. 8). In the optimal case of 4 robots in Configuration 1, the cooperative strategy resulted in a 30% decrease of the time needed for the completion of the whole task. However, the greater the number of robots, the smaller the gain from addition of another robot. This congestion effect increases faster with the number of robots in Configuration 2, where for 5 robots T_{total} time is smaller if robots work in the independent mode. Additionally, because in Configuration 1 only 5 corridors are available for the robots working in a cooperative mode, a greater number of robots was not tested, as it would result in a permanent inactivity of the "sixth" robot. Additionally, small number of robots in Configuration 1 results in shorter total time for the independent strategy. Leaving cargo on shelves when few or no robots are available to pick cargo from the other side, requires the available robot to perform unnecessary drop and pick operations.

The average robot operation time signifies an optimum number of working robots in some cases (Fig. 9), especially in case of the independent strategy. Configuration 1 also seems to be unfavorable for the independent strategy, as the average robot cycles usage increases rapidly with the number of robots.

Naturally, the total robots execution time T_{Rtotal} increases when more robots are working on the task. However, for the independent strategy, T_{Rtotal} increases more rapidly with respect to the cooperative strategy (Fig. 10). For Configuration 2, the cooperative strategy needs at least 4 robots to surpass the independent strategy with respect to total execution time T_{Rtotal}. All experimental data presented in Figures 8–10 are presented in Table 3 for better comparison.

When the energy wastage defined by the number of moves performed without cargo is concerned, in Configuration 2, the cooperative strategy significantly outperforms the independent strategy, and in the worst case (setup 1 for 3–5 robots), the cycles wastage is the same for both strategies (Fig. 11). Those results also suggest, that in the cooperative strategy robots are naturally spending less time travelling for cargo and more time actually delivering cargo to either tables being closer to the final drop off destination or to the final destinations itself. In general the presented results show, that when only few robots are working (as compared to the size of the

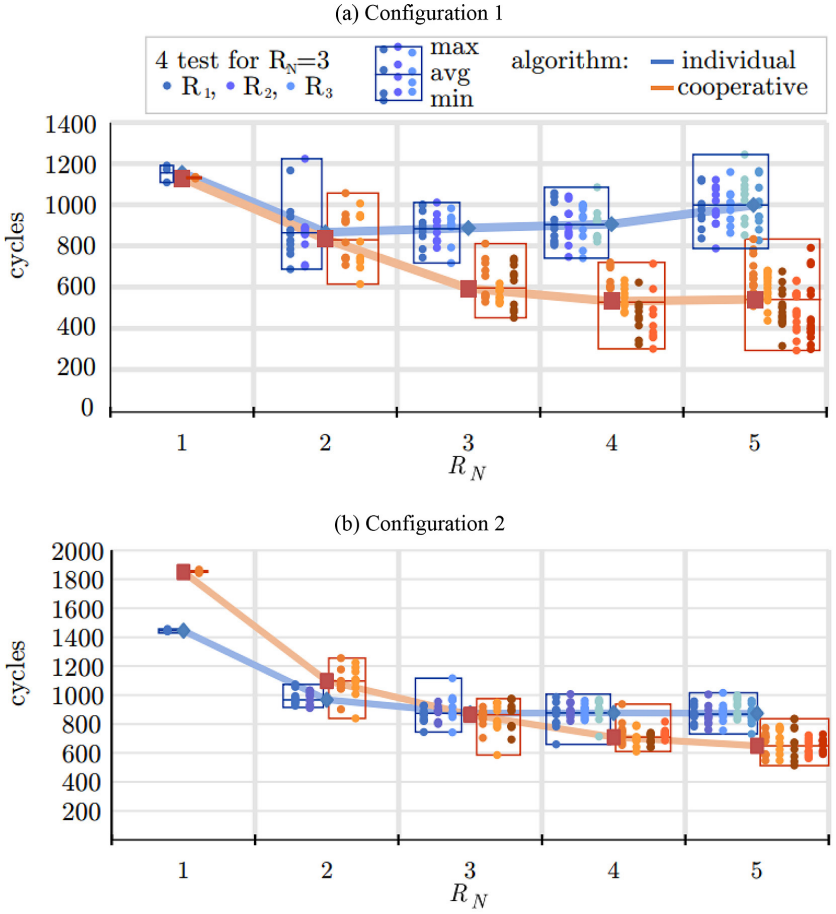


Figure 9. Average robot operation time T_R by individual robots in the warehouse

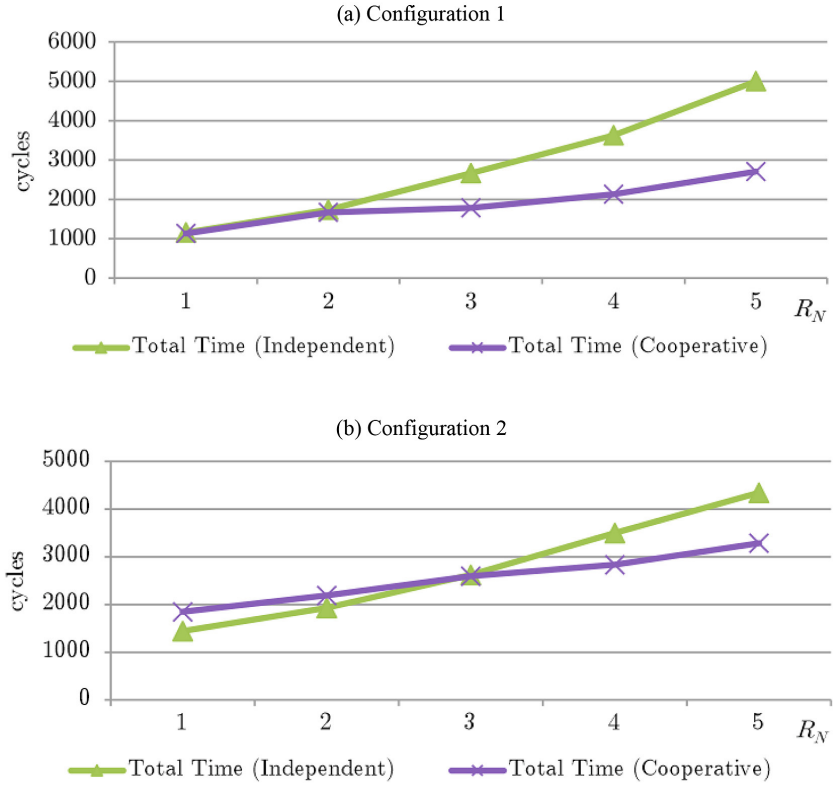


Figure 10. Total execution time T_{Rtotal} for all robots

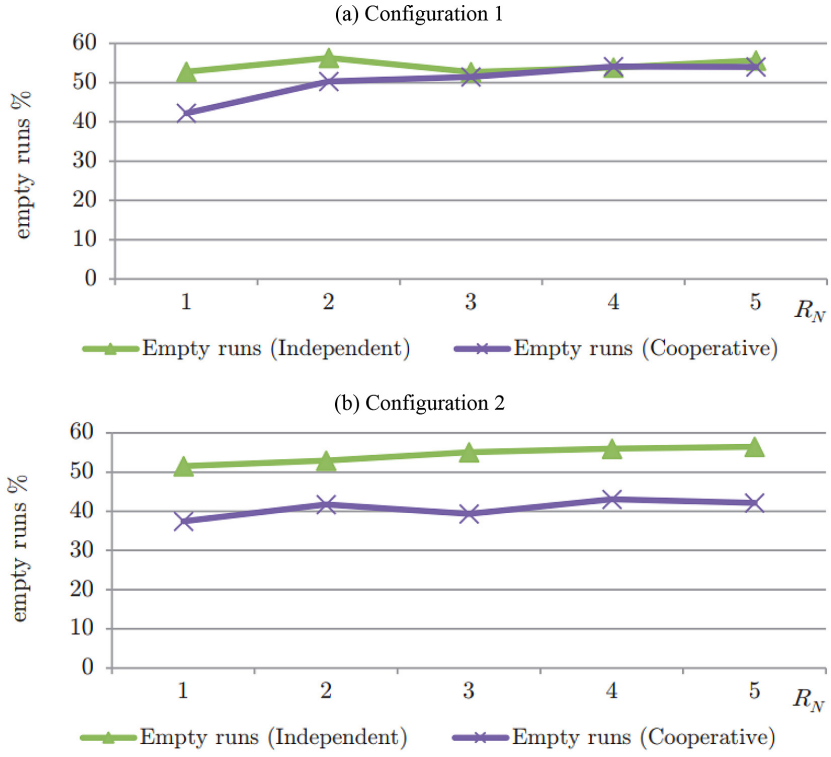


Figure 11. Empty runs $\%E_U$ for all robots

Table 3. Summary of experimental data presented in the Figures 7–9

Parameter		(a) Configuration 1					(b) Configuration 2				
	R_N	1	2	3	4	5	1	2	3	4	5
T_{total} (Fig. 8)	Ind	5145	3441	3000	2756	2820	5657	3776	3285	3172	3134
	Coop	4621	2961	2364	1853	1957	9436	4670	3764	2944	2736
T_R (Fig. 9)	Ind	1156	867	887	906	1198	1443	966	873	875	1041
	Coop	1130	834	593	531	671	1848	1097	864	708	787
T_{Rtotal} (Fig. 10)	Ind	1156	1734	2661	3624	5990	1443	1932	2619	3500	5205
	Coop	1130	1668	1779	2124	3355	1848	2194	2592	2832	3935

warehouse), it is better to use the individual strategy, as using the cooperative strategy requires more iterations to complete all tasks. For a medium number of robots working in the warehouse, the total cost is similar, but if the increasing number of robots causes congestion and collisions, it is better to use the cooperative strategy

CONCLUSIONS

The paper examined a possibility of non-communicating robots cooperation in a warehouse parcel transportation system. Individual strategies, in which robots deliver parcels directly to the designated destinations are always possible. However, such systems need additional optimization to select the number of robots that matches the size of the warehouse and prevents congestion. On the other hand it is possible to devise cooperative strategies, in which robots cooperate without directly communicating with one another. Obviously, such strategies must incorporate additional knowledge about the warehouse setup. In the presented example it was assumed that a global orders list allows for corridor reservation, parcels are always transported in one general direction, and that shelves are approachable from both sides, enabling parcels to be dropped by one robot and picked up by another robot on the other side of the shelve. When such assumptions are made, non-communicating robots may have higher efficiency by working in a cooperative fashion.

An additional advantage of the presented cooperative strategy is that it selects an optimal number of cooperating robots. This selection is naturally performed based on the global orders list reservation mechanism. When all corridors are reserved, an additional robot returns to its parking location.

Future work should investigate whether it is possible to enhance the logic of a robot in such a way, that changes in strategies are made only on the basis of observations the robot can make (frequency of collisions for example). Such a robot could switch between the individual and cooperative strategies automatically, adapting to the current conditions in the warehouse.

Acknowledgements

This research was funded by the Silesian University of Technology (SUT) through the subsidy for maintaining and developing the research potential grant in 2024: 02/060/BK_24/0059 and Rector's habilitation grant 02/060/RGH22/0042.

REFERENCES

1. Makusee M., Glock C. H., and Grosse E. H. Order picker routing in warehouses: A systematic literature review. International Journal of Production Economics 2020; 224: 107564.
2. Contini A. and Farinelli A. Coordination approaches for multi-item pickup and delivery in logistic scenarios, Robotics and Autonomous Systems, 2021; 146: 103871.
3. Gansterer M., Küçüktepe M., and Hartl R. F. The multi-vehicle profitable pickup and delivery problem. Or Spectrum 2017; 39: 303–319.
4. Skrynnik A., Andreychuk A., Nesterova M., Yakovlev K., and Panov A. Learn to follow: Lifelong multi-agent pathfinding with decentralized replanning, in PRL Workshop Series {\textendash} Bridging the Gap Between AI Planning and Reinforcement Learning, 2023.
5. Chen L., Wang Y., Mo Y., Miao Z., Wang H., Feng M., and Wang S. Multiagent path finding using deep reinforcement learning coupled with hot supervision contrastive loss, IEEE Transactions on Industrial Electronics, 2023; 70(7): 7032–7040.
6. Hong S. A performance evaluation of bucket brigade order picking systems: Analytical and simulation approaches, Computers & Industrial Engineering, 2019; 135: 120–131.
7. Mangla N., Singh M., and Rana S. K. Resource scheduling in cloud environmet: A survey. Advances in Science and Technology. Research Journal 2016; 10(30): 38–50.
8. Tian G., Zhang C., Fathollahi-Fard A. M., Li Z., Zhang C., and Jiang Z. An enhanced social engineering optimizer for solving an energy-efficient disassembly line balancing problem based on bucket brigades and cloud theory, IEEE Transactions on Industrial Informatics, 2022.
9. Lima, O. D. A and MB G. A cellular automata ant memory model of foraging in a swarm of robots, Applied Mathematical Modelling, 2017; 47(21): 551–572.
10. Brandys W., Ga luszka A., Jedrasiak K., Radziszewska J., and Daniec K. Hierarchical game approach to solve conflicts in multiagent systems, in Advanced Technologies in Practical Applications for National Security, 2018; 135–145, Springer.
11. Gałuszka A. and Świerniak A. Non-cooperative game approach to ´ multi-robot planning, 2005.
12. Witczak M., Majdzik P., Stetter R., and Lipiec B. Multiple AGV fault-tolerant within an agile manufacturing warehouse, IFACPapersOnLine, 2019; 52(13): 1914–1919.
13. Digani V., Hsieh M. A., Sabattini L., and Secchi C. Coordination of multiple AGVs: a quadratic optimization method, Autonomous Robots, 2019; 43: 539–555.
14. Pizoń J., Wójcik Ł., Gola A., Kański Ł., and Nielsen I. E. Autonomous mobile robots in automotive remanufacturing: A case study for intra-logistics support. Advances in Science and Technology. Research Journal, 2024; 213–230.
15. Fazlollahtabar H. and Saidi-Mehrabad M. Methodologies to optimize automated guided vehicle scheduling and routing problems: a review study, Journal of Intelligent & Robotic Systems, 2015; 77(3): 525– 545.
16. Sahin C., Demirtas M., Erol R., Baykaso˘glu A., and Kaplano˘glu V. A multi-agent based approach to dynamic scheduling with flexible processing capabilities, Journal of Intelligent Manufacturing, 2017; 28(8), 1827–1845.
17. Tanabe R., Yasuda S., Nakajima K., and Uemura W. A Finding Place Method Considering distance between Robots. 2023 IEEE 12th Global Conference on Consumer Electronics (GCCE). IEEE, 2023.
18. Iwańkowicz R. R., and Sekulski Z. Solving the problem of vehicle routing by evolutionary algorithm. Advances in Science and Technology. Research Journal 2016; 10: 29.
19. Dorri A., Kanhere S. S., and Jurdak R. Multi-agent systems: A survey. IEEE Access 6, 2018; 28573–28593.
20. Duering M. and Pascheka P. Cooperative decentralized decision making for conflict resolution among autonomous agents, in 2014 IEEE International Symposium

on Innovations in Intelligent Systems and Applications (INISTA) Proceedings IEEE, 2014; 154–161.

21. S¸enba¸slar B., H¨onig W., and Ayanian N. Rlss: real-time, decentralized, cooperative, networkless multi-robot trajectory planning using linear spatial separations, Autonomous Robots, 2023; 1–26.
22. Chawla V., Chanda A., and Angra S. Scheduling of multi load agvs in fms by modified memetic particle swarm optimization algorithm, Journal of Project Management, 2018; 3(1): 39–54.
23. Kitjacharoenchai P., Ventresca M., Moshref-Javadi M., Lee S., Tanchoco J. M., and Brunese P. A. Multiple traveling salesman problem with drones: Mathematical model and heuristic approach, Computers & Industrial Engineering, 2019; 129: 14–30.
24. Grzejszczak T., Krzyzanowski W., and Gałuszka A. Warehouse model for interaction planning of mobile robots, in Modelling and Simulation 2020 – The European Simulation and Modelling Conference, ESM, 2020; 229–233.