Flexible Scheduling System for AGVs Using Auction-Based Allocation and TSP Planning

Pedro Maia[1, 2, 3], Ana Moura[4], and José Santos[1, 2, 3]

[1]Department of Mechanical Engineering, University of Aveiro, Portugal
[2]TEMA - Centre for Mechanical Technology and Automation, DEM, University of Aveiro, Portugal
[3]LASI - Intelligent Systems Associate Laboratory, Guimarães, Portugal
[4]DEGEIT/GOVCOPP, University of Aveiro, Portugal

***Abstract*— This work presents a flexible scheduling system for Automated Guided Vehicles (AGVs) that seamlessly adapts to single and multi-vehicle environments with varying load capacities. By integrating an auction-based task allocation mechanism with Traveling Salesman Problem (TSP) route optimization, the proposed system efficiently distributes tasks while minimizing overall travel distance and energy consumption. The auction model enables AGVs to bid based on computed costs derived from solving a pickup-and-delivery TSP variant that incorporates capacity constraints, thus ensuring optimal resource utilization and balanced workload distribution. Computational experiments demonstrate significant improvements in task completion times and route efficiency in dynamic scenarios, highlighting the system's potential for scalable deployment in modern industrial automation settings. Future work will address enhancements in conflict-free routing and dynamic task allocation to further align the model with real-world applications.**

***Index Terms*— AGV, Scheduling, Auction-Based, TSP, Dynamic Task Allocation, Transport Optimization**

I. INTRODUCTION

Effective scheduling in Automated Guided Vehicles (AGV)-based logistics is crucial for improving operational efficiency, minimizing delays, reducing operational costs and ensuring seamless operation. Integrating AGV scheduling with production planning can significantly improve material handling efficiency and support flexible manufacturing systems [1]. Multi-objective optimization further refines scheduling by balancing makespan, energy consumption, and AGV utilization [2]. Overall, implementing robust and effective scheduling frameworks is important to maximize the benefits of AGV systems in various logistics applications, reducing bottlenecks and supporting scalable automation in modern logistics.

Dynamic task allocation and route optimization are relevant in various domains, including multi-robot systems. It faces challenges like real-time adaptability and balancing computational efficiency with optimal resource utilization. Dynamic task allocation must consider real-time capabilities and needs to maintain balance and optimize system performance. Traditional offline task allocation methods may not be enough, highlighting the need for online and dynamic strategies adaptable to changing operator requirements and ensuring both productivity and operator well-being. Addressing these challenges requires the development of advanced algorithms capable of real-time adaptation, efficient resource utilization, and seamless integration of human factors to enhance overall system performance [3]. Edge computing frameworks attempt to address these issues by offloading tasks dynamically, but they must manage latency and scalability trade-offs [4], [5].

In contrast, multi-AGV systems enhance operational efficiency but introduce complexities in task allocation and coordination. The need for a scalable system that works for both single and multiple AGVs is critical in modern industrial and logistics environments, where flexibility and efficiency are of extreme importance. Traditional centralized control systems, while effective for small fleets, often struggle with scalability as computational complexity grows exponentially with the number of vehicles, limiting their applicability to systems with more than 10–15 AGVs [4], [6]. Moreover, hybrid task allocation methods, which address single-AGV and multi-AGV tasks, are essential for environments where diverse material sizes require different handling strategies [7]. Such systems must also ensure robustness, avoiding deadlocks and collisions while optimizing throughput and energy consumption [4], [8].

This article proposes a scheduling system able to adapt to different kinds of AGV environments. This is accomplished by using an auction model to distribute tasks, while a Traveling Salesman Problem (TSP) solver is used to define the cost and the route that should be taken considering the constraints necessary.

This method is very reliable and versatile, allowing for adaptation to various systems and to dynamic problems. It is also efficient by guaranteeing that the workload is balanced and the AGVs route can include the maximum number of tasks with the minimal of detours.

This paper includes the following sections: Related works in section II, briefly describing traditional AGV approaches; Section III describes in detail the system design and the considerations taken into account; Section IV describes the implementation and the simulations done; Section V discusses the results obtained and some comparisons; Lastly,

pedromaia@ua.pt

section VI talks about the conclusions and points from where this work can be upgraded.

II. RELATED WORK

The literature review for this work had in mind the objective to refine the knowledge about systems and approaches that were already tested by others. The main conclusions are presented in this section.

Le-Ahn & De Koster [9] provide a foundational review of AGV systems, categorizing traditional scheduling approaches into scheduling (online and offline) and dispatching. If all tasks are known before the planning period, the scheduling problem can be solved offline and the complete vehicle routes can be optimized and constructed before vehicles carry them out. However, in practice, exact information about tasks is usually known at a very late instant. This makes offline scheduling hardly possible. Therefore, online scheduling or dispatching systems are needed to control vehicles. Online scheduling consists of adapting the scheduling based on the change of information [9].

The dispatching system can be seen as a scheduling system with a zero planning horizon . Dispatching rules are simple and can be easily adapted for AGV management systems. There are two main kinds of dispatching systems: decentralized and centralized. Decentralized control systems dispatch vehicles based on local information only. The main advantage of the decentralized control system is its simplicity, but its efficiency is low. In centralized dispatching systems, a central controller keeps track of all movements regarding internal transport [9].

On the task allocation side of things, Lee [10] explores the task allocation for multi-robot systems, particularly an auction-based approach. In the auction algorithms, an auctioneer agent evaluates the received bids from the bidder agents. The auctioneer then allocates the task to the bidder that has the best bid [10].

When it comes to the application of the TSP in AGV environments, De Ryck et al. [6] proposes an approach to optimize resource management in AGV systems using a TSP-based model. The authors formulate the AGV routing and charging problem as a modified TSP, where charging stations can be optimally inserted into the task sequence to minimize total travel time. The study demonstrates how this approach improves efficiency in decentralized AGV fleets [6]

Even though approaches that use auction-based task allocation and/or Pickup and Delivery Problems (PDP) or TSP related path planning have been developed before, they always represent use-cases where the system is well defined as multi-load or multi-vehicle. This study pretends to create an approach that can be applied in any case of real-world AGV systems.

III. SYSTEM DESIGN AND ARCHITECTURE

This section gives an overview in the development process and the key decisions taken during the stages of building the proposed framework. It also starts by explaining the implications that the real-world systems have in thought process of this work.

An AGV system can be of various types, based on the genre of AGV that can be categorized into three main types: forklift, underride, and tugger. Forklifts and underrides are usually vehicles that can only carry one load at a time (unit-load). Tuggers, on the other hand, can move more than one load at a time but usually have constraints related to the placement of the loads. Usually, you cannot access the first load without removing the others, so it needs to be loaded following the Last-In-First-Out (LIFO) order. The methods that were chosen to use in this work are the ones that can be applied in the case of tugger AGVs because the unit-load AGVs are a special case of the latter, where the capacity is constrained to one load per vehicle.

Concerning route planning, the AGVs usually work in a principle that follows the PDP where AGVs need to pick up a task in a determined place and deliver it to another location without exceeding the vehicle capacity. It must not be mistaken for the Capacitated Vehicle Routing Problem (CVRP), where the vehicles leave the depot with the load already allocated. This is only applicable to AGVs that work in distribution centers (like taking goods from a warehouse to another place).

Lastly, task allocation can be centralized or decentralized depending on the complexity of the system (number of vehicles and/or size of the installation). Usually, the centralized approach works well in the majority of the cases because it allows for a near-optimal global solution, at the expense of more computational resources.

A. Auction-Based Task Allocation

Auction-based task allocation is a dispatching method for assigning tasks to AGVs in dynamic environments. Inspired by economic auction principles, this approach treats tasks as items to be “sold” and AGVs as “bidders” competing based on cost efficiency.

This method ensures efficient resource utilization, scalability, and adaptability in real-time operations. AGVs continuously recompute optimal paths while respecting constraints like battery life, load capacity, and task sequencing. By taking into account combinatorial optimization (e.g., TSP with pickup and delivery (TSP-PD)), the system minimizes total travel costs while maintaining flexibility in dynamic logistics scenarios.

Auction mechanisms are particularly effective in multi-AGV systems, as they balance workload distribution making them robust for warehouse automation, manufacturing, and other material-handling applications.

The auction-based task allocation mechanism developed uses a bidding system where AGVs compete to win tasks based on cost efficiency by evaluating the cost function associated with incorporating a new task into their existing route. When tasks are pending, each eligible AGV (with sufficient battery and capacity) tries to predict the cost of adding the task to its current route by solving a TSP with its respective constraints (explained in further detail in the next

section). The cost function primarily represents the travel distance required to complete all assigned tasks, including the new one. The task is then awarded to the AGV with the lowest bid, ensuring optimal resource utilization.

The steps in this algorithm are the following:

1) When a task becomes available, each eligible AGV computes the optimal route to service its current tasks plus the new task by solving the TSP-PD.
2) The solver minimizes the total path cost (distance) while ensuring the constraints are validated.
3) If a feasible route exists, the AGV's bid is the total cost of the optimized route returned by the TSP-PD solver.
4) The system collects all valid bids and assigns the task to the AGV with the lowest cost bid.
5) The winning AGV updates its route to include the new task, while other AGVs retain their original plans.

The key features of this method are the distance-based optimization and the usage of the constraints as penalties. Also, this method allows for the dynamic recalculation of paths, allowing for the least rerouting possible.

B. TSP-Based Route Optimization

Oftentimes, route planning based on traditional rules like first-come-first-served or greedy approaches leads to inefficient routes, especially when handling multiple pickup-and-delivery tasks with capacity constraints, which can be the case for many AGV applications. For that, there was the need to adopt a method that could successfully help in finding the best route.

Usually, common vehicle routing problems (VRP) require a fleet of vehicles and well-known demands of each delivery point. Some of these problem variations, like the vehicle routing problem with pickup and delivery (VRPPD), are only capable of providing an optimal solution when all the tasks are available before the vehicles start their movement, and this does not correspond to the dynamic environment where AGVs are usually deployed. One method that does well with dynamic recalculations is the TSP method, which guarantees that a vehicle visits each node in a graph starting and ending at a determined node, usually the depot. However, this method does not support fleets of various vehicles but that issue is tackled by the auction-based system (talked about in the previous section) that guarantees that each agent will calculate their route independently.

By formulating the problem as a mixed-integer linear program (MILP), the system ensures that tasks are completed following an optimal order while respecting battery limits, load capacities, and other delivery constraints. The resulting route minimizes travel distance, reduces idle time, and improves system performance in dynamic warehouse environments.

The mathematical formulation is the one that follows: Let $G = (N, A)$ be a directed graph, where N is the set of nodes and $A \subseteq N \times N$ is the set of arcs connecting them. Each arc $(i, j) \in A$ is associated with a cost c_{ij}. The binary decision variable $x_{ij} \in \{0, 1\}$ indicates whether arc (i, j) is included in the route. The binary variable $y_{ij} \in \{0, 1\}$ represents whether node i is visited before node j in the total ordering of the tour. In addition, q_i denotes the load carried by the vehicle upon arrival at node i, δ_{ij} represents the load variation when moving from node i to node j (positive for pickups and negative for deliveries), and C is the vehicle's maximum capacity. For the task-related variables, T is the set of all tasks, where each task involves a pickup node p and a corresponding dropoff node d.

Objective:

$$\min \sum_{(i,j)\in A} c_{ij} x_{ij} \tag{1}$$

Assignment Constraints:

$$\sum_{\substack{j\in N \\ j\neq i}} x_{ij} = 1 \quad \forall i \in N \qquad \sum_{\substack{j\in N \\ j\neq i}} x_{ji} = 1 \quad \forall i \in N \tag{2}$$

Total Order Constraints:

$$y_{ij} + y_{ji} = 1 \quad \forall i, j \in N \setminus \{0\},\ i \neq j \tag{3}$$

Transitivity Constraints:

$$y_{ij} + y_{jk} - y_{ik} \leq 1 \quad \forall i, j, k \in N \setminus \{0\},\ \text{distinct} \tag{4}$$

Crossover Constraints:

$$x_{ij} \leq y_{ij} \quad \forall (i, j) \in A,\ i, j \neq 0,\ i \neq j \tag{5}$$

$$x_{ik} + y_{ij} + y_{jk} \leq 2 \quad \forall i, k \in N,\ i, k \neq 0,\ i \neq k, \tag{6}$$

$$j \in N \setminus \{0, i, k\} \tag{7}$$

Precedence Constraints:

$$y_{p,d} = 1 \quad \forall p, d \in T,\ p \neq d \tag{8}$$

Load Initialization:

$$q_0 = 0 \tag{9}$$

Flow Conservation / Load Constraints:

$$q_j \geq q_i + \delta_{ij} - (1 - x_{ij}) \cdot C \quad \forall (i, j) \in A,\ i \neq j \tag{10}$$

$$q_j \leq q_i + \delta_{ij} + (1 - x_{ij}) \cdot C \quad \forall (i, j) \in A,\ i \neq j \tag{11}$$

The objective function (1) minimizes the total travel cost across all arcs. Assignment constraints (2) ensure that each node is visited exactly once, with one incoming and one outgoing arc. Total order (3) and transitivity constraints (4) define a valid visitation order among non-depot nodes to prevent sub-tours. Crossover constraints (5)-(7) ensure consistency between arc usage and visitation order, and prevent path conflicts. Precedence constraints (8) enforce that each pickup precedes its corresponding delivery. The load initialization (9) and flow conservation constraints (10) and (11) maintain vehicle capacity along the route by updating the load based on pickups and deliveries, ensuring the feasibility of the tour in terms of carried load.

However, the formulation of the TSP brings some issues in real-world systems. For example, if there are different tasks with the same pick-up and/or delivery nodes the TSP doesn't handle it that well because it requires each node to be visited

exactly once. To tackle this issue it was added virtual nodes. These are duplicates of the original nodes, with the same distances to other places. The only difference is that they have a really small distance between them and the original node, so the TSP can treat them as separate stops.

Another issue is that in the real world, not every location is directly connected to every other one, but the TSP needs that to avoid subtours. To avoid problems, two helper functions were developed. One is to calculate and store the shortest distance between all nodes even if they are not connected. The other takes the route given by the TSP solver and expands it into a path possible to follow. For example, imagine four points (A, B, C, D) lined up. The first function builds a distance table between them. If the TSP solver says the best order is A–D–B–C, the second function expands the route, like A–b–c–D–c–B–C, where the lowercase letters are the actual steps between the main points.

C. Integration and Scalability

The system transitions from single-AGV to multi-AGV scenarios by utilizing the task assignment mechanism that coordinates multiple AGVs while adhering to constraints like battery life, capacity, and LIFO task handling. In single-AGV scenarios, there is no need to use the task allocation system because tasks can be assigned sequentially and the AGV follows a computed path to pick up and deliver tasks. For multi-AGV scenarios, the system uses the developed auction-based approach.

The system should employ several coordination strategies to avoid collisions and deadlocks in multi-AGV scenarios. One aproach that could be implemented is the increase of cost in edges of the graph where some AGV already is entitled to take. That way path planning will try to ensure that AGVs will avoid that congested places while following the best paths, reducing the likelihood of any conflit.

For real-time collision avoidance, AGVs should monitor their progress along edges and recalculate paths if conflicts are detected (e.g. if an AGV is stranded or an edge is blocked). The emergency depot return protocol could ensure that low-battery or stranded AGVs retreat to the depot, freeing up network resources. Furthermore, the precomputed shortest paths and bidirectional edge checks could prevent two AGVs from occupying the same edge simultaneously.

These strategies collectively ensure smooth coordination, minimizing delays and maintaining system efficiency in dynamic multi-AGV environments.

D. Computational Complexity

The overall complexity of the developed algorithm is directly related to the TSP problem that is a NP-hard problem and has an exponential solve time.

IV. IMPLEMENTATION AND EXPERIMENTAL SETUP

The development environment for this AGV simulation is based on Python, utilizing several key libraries to implement the logic and visualization. The simulation relies on matplotlib for creating animated visualizations of the AGV movements and task assignments, and PuLP for solving the TSP with pickup and delivery constraints. The network of nodes and edges is modeled using dictionaries and custom classes (AGV, Task, PathNetwork, and AGVEnvironment), using breadth-first search (BFS) for pathfinding. The code is designed to run in a standard Python environment. The visualization provides real-time updates of AGV statuses, task progress, and network topology, making it suitable for testing and demonstrating AGV routing algorithms.

Key assumptions of the system include: AGVs move at a constant speed (1 unit/step), consume battery linearly (1% per move step), and recharge only at the depot (at a rate of 2% per step). The network is fully connected, ensuring paths exist between all nodes. Tasks have fixed pickup and drop-off locations and are spawned probabilistically with a 10% chance per step, and AGVs must respect their load capacity constraints. Emergency protocols trigger if battery levels drop critically low, forcing AGVs to return to the depot. The simulation assumes perfect communication and no delays in task assignment or path recalculation. The simulation ends after completing 50 tasks and a fixed random seed (111) was used to ensure deterministic behavior. Experiments were conducted on a system with an Intel i5-10210U CPU, 8GB RAM, and Windows 11, using Python 3.13.2 and PuLP 3.0.2. Various simulation runs were executed with these settings to validate the algorithm's deterministic performance under constrained AGV capacities and task delivery logic.

The map and simulation interface can be seen in the image 1, where the length of each edge can be deducted based on the axis. Dashed lines between triangles indicate tasks that are not currently assigned to any vehicle.

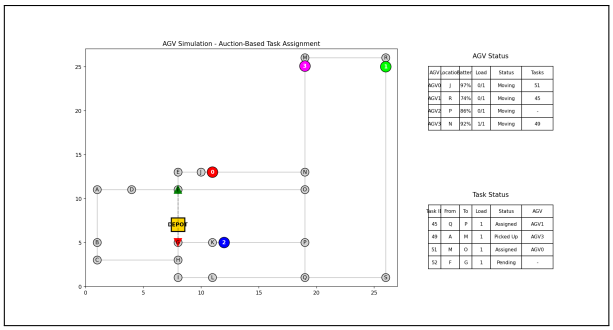

Fig. 1: Simulation environment of the developed system using the matplotlib module, working with 4 AGVs

Some of the tests were executed in different scenarios to test the influence some changes in the environment would have on the system. The scenarios tested were about systems with a single-AGV, multi-AGV and, dynamic task arrivals. The single AGV scenario tested the influence of the variation in the capacity of a single AGV to check how it could modify the system. The multi-AGV scenario was made using single-load AGVs, varying the number of agents in the system. Lastly, dynamic task arrivals was based on changing the number of tasks that would be generated simultaneously, to

check if it had any impact in the overall performance of the system.

The performance criteria used to measure the performance of the system was the task completion time (interval between the task generation and its completion, in simulation steps), total traveled distance (distance made by all the AGVs, in simulation units), computational efficiency (time taken by the TSP solver to plan a feasible path, in seconds) and the AGV utilization (time moving versus total time of the simulation, in percentage).

V. COMPUTATIONAL EXPERIMENTS

The results obtained in the computational experiments are presented in the table I and table II.

TABLE I: Performance for a singe AGV with N load capacity

N	Task completion time [steps]		Distance Traveled	TSP solver time [s]		AGV Utilization
	Average	Max		Total	Average	
2	109,3	214	1097	191,3	3,61	98,3
3	94,4	223	991	115,04	2,21	98,5
4	96,4	204	1021	102,75	2,01	98,6
6	99,6	212	1019	35,25	0,71	98,3

TABLE II: Performance for N AGVs with unitary load capacity

N	Task completion time [steps]		Distance Traveled	TSP solver time [s]		AGV Utilization
	Average	Max		Total	Average	
2	76,4	157	1510	3,69	0,06	98,7
3	59,1	211	1864	3,54	0,05	94,6
4	43,4	173	1926	3,53	0,05	92,6
6	32,3	89	2091	6,11	0,04	72,6
10	31,1	89	2095	13,68	0,04	43,6
20	31,1	89	2095	34,65	0,04	21,8

The conclusions that can be drawn from these results show that systems with multi-AGV are both more efficient in the time taken to complete the tasks and in the computation resources needed to calculate the best path. However, they come at the expense of a bigger financial investment by buying more vehicles. Also, it can be seen that increasing the number os AGVs (to 10 or 20) only reduces the utilization percentage, meaning that the system got to a convergence point and it must also be said that AGV collisions or conflicts were not taken into account and that could penalize systems where there are too many AGVs, reducing the effectiveness of the whole system. Besides that, the fact that there were more AGVs, the usage of the TSP solver was more requested because of the bidding system (the average time remained similar, but it increased the total time). On the side of the multi-load single-AGV, the main advantage taken from the developed system was the traveled distance, since the TSP solver always tried to optimize the route so that it could take the maximum amount of cargo with the least amount of detours. The great increase in the computational resources needed to find the optimal solution to the routing problem comes specifically from the load constraint because it would lead to the need to abort the best path multiple times to ensure that the maximum capacity was never violated. That is the reason behind the fact that the value for the total time spent in calculations is decreasing with the increase of the load capacity. It can be seen in the graph from image 2 the comparison between methods in greater detail.

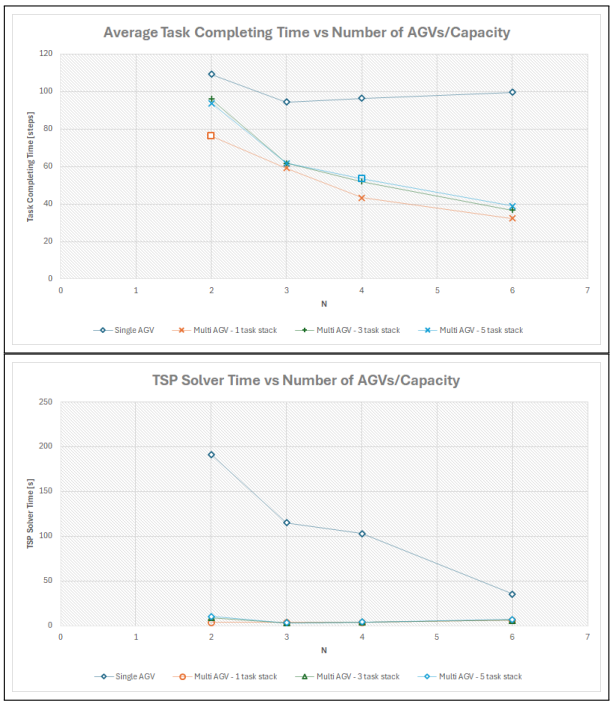

Fig. 2: Simulation results related to task completion times between single-AGVs and multi-AGVs systems with different task stacks

The auction-based task allocation proved to be effective in distributing the task load between the AGVs in multi-AGV systems. However, it must be noted that its influence only became truly influential at the beginning of the simulation, or when there was more than one AGV idle, waiting for a new task. Even though it did not use its full potential in the simulation environment developed, in practical applications in manufacturing systems, it is not strange to see various tasks appearing at the same time when there was a long idle time for the AGVs, which can make it a great approach to create efficient task allocation between multiple vehicles.

The TSP solver proved to be effective in providing feasible efficient routes, but it also proved to struggle when it needed to compute more tasks simultaneously (six tasks was the sweet-spot number of tasks from which the computation time taken was reasonable). That problem appears because the number of nodes, and thus the number of possible combinations, increased with every new task added to the calculation. An approach using near-optimal algorithms, like heuristics, could be developed to improve the computational efficiency of the proposed solution.

Other issues appeared during the development of this work that, because of the shortage of time, were not tackled like conflict-free routing, the LIFO constraints, and the dynamic task allocation.

TABLE III: Performance of N AGVs with unitary load with task generating in stacks of n tasks

N	n	Task completion time	Distance Traveled	TSP Solver Time		AGV Util.
				Total	Average	
2	1	76,4	1510	3,69	0,06	98,7
	3	96	1752	9,1	0,16	97,9
	5	93,9	1685	10,4	0,17	97,6
3	1	59,1	1864	3,54	0,05	94,6
	3	61,8	1701	3,26	0,05	97,4
	5	61,8	1701	3,26	0,05	97,4
4	1	43,4	1926	3,53	0,05	92,6
	3	52	1884	4	0,05	93,2
	5	53,5	1903	4,03	0,05	91,2
6	1	32,3	2091	6,11	0,04	72,6
	3	36,7	1876	6,22	0,05	82,1
	5	39	1944	6,82	0,05	80,4
10	1	31,1	2095	13,68	0,04	43,6
	3	31	2008	12,6	0,04	57,2
	5	29,8	1991	13,86	0,04	60,4
20	1	31,1	2095	34,65	0,04	21,8
	3	31	2008	34,75	0,04	28,6
	5	29,8	1991	35,49	0,04	30,2

The conflict-free routing and collision prediction and avoidance, if implemented, could show different results in the multi-AGV systems since re-planing and waiting strategies oftentimes lead to suboptimal or inefficient status. Also, it would increase the level of similarity to real-world systems. An approach like the one proposed by Li et al. [11], where the conflicts are predicted by time of arrival and avoided with the rerouting could be implemented.

About the LIFO constraints, there were made efforts to implement them but ultimately it was not working because the solver was not taking them into account when solving the problem, leading to wrong orders of pickup and delivery. Some more effort could be done in this regard to try to find the correct formulation for the MILP problem, but the shortage of time made that not possible.

Lastly, it was not developed the dynamic task allocation, because it is a difficult behavior to implement with a lot of edge cases and nuances, and it was not the focus of this work. An approach like the one suggested by Hu et. al [12] where a Hybrid Discrete State Transition Algorithm (HDSTA) with elite solution sets and Tabu Search could be investigated and implemented.

All of this issues and improvements, if taken into consideration, could lead to a more realistic environment representation and also lead to better demonstrate the advantages of the proposed approach.

VI. CONCLUSION AND FUTURE WORK

The key findings that this work provided concern the comparison between different types of AGV systems, namely multi-load or multi-AGV. Besides that, it provided a framework with great real-world deployment potential by being simple implementation and good for scalable systems. The auction-based task allocation method was useful in distributing the work by finding the vehicle that would be the least affected by a change in path. The TSP-based route planning did what was expected and found feasible routes that avoid returning to the depot each time a task is completed, allowing for an increase in efficiency.

This developed approach had always in mind the practical application in manufacturing AGV systems, with all their differences and possible variations. That means that this approach can be implemented in almost any AGV system, and some of it can be suppressed in some special cases (in example, the auction-based task allocation its not necessary in single-AGV systems).

Potential extensions of this proposed work include the ones mentioned in the previous section and also the potential to include better strategies (e.g. the usage of machine leaning to develop better strategies in the auction strategies, heuristics for computational heavy TSP calculations). The handling of unforeseen disruptions in real-time would also be a relevant point to develop further.

ACKNOWLEDGMENT

The present study was developed in the scope of the Project AM2R " [C644866475-00000012– project n. 15], financed by PRR– Recovery and Resilience Plan under the Next Generation EU from the European Union, and was also supported by the project UID 00481 Centre for Mechanical Technology and Automation (TEMA)

REFERENCES

[1] N. Singh, P. V. Sarngadharan, and P. K. Pal, "Agv scheduling for automated material distribution: A case study," *Journal of Intelligent Manufacturing*, vol. 22, pp. 219–228, 4 2011.

[2] Y. Liu, S. Ji, Z. Su, and D. Guo, "Multi-objective agv scheduling in an automatic sorting system of an unmanned (intelligent) warehouse by using two adaptive genetic algorithms and a multi-adaptive genetic algorithm," *PLoS ONE*, vol. 14, 12 2019.

[3] M. Calzavara, M. Faccio, I. Granata, and A. Trevisani, "Achieving productivity and operator well-being: a dynamic task allocation strategy for collaborative assembly systems in industry 5.0," *International Journal of Advanced Manufacturing Technology*, vol. 134, pp. 3201–3216, 10 2024.

[4] I. Draganjac, T. Petrović, D. Miklić, Z. Kovačić, and J. Oršulić, "Highly-scalable traffic management of autonomous industrial transportation systems," *Robotics and Computer-Integrated Manufacturing*, vol. 63, 6 2020.

[5] B. Suganya, R. Gopi, A. R. Kumar, and G. Singh, "Dynamic task offloading edge-aware optimization framework for enhanced uav operations on edge computing platform," *Scientific Reports*, vol. 14, 12 2024.

[6] M. D. Ryck, M. Versteyhe, and F. Debrouwere, "Automated guided vehicle systems, state-of-the-art control algorithms and techniques," 1 2020.

[7] Y. Hu, X. Wu, J. Zhai, P. Lou, X. Qian, and H. Xiao, "Hybrid task allocation of an agv system for task groups of an assembly line," *Applied Sciences (Switzerland)*, vol. 12, 11 2022.

[8] W. Lu, S. Guo, T. Song, and Y. Li, "Analysis of multi-agvs management system and key issues: A review," *Computer Modeling in Engineering & Sciences*, vol. 131, pp. 1197–1227, 2022.

[9] T. Le-Anh and M. D. Koster, "A review of design and control of automated guided vehicle systems," *European Journal of Operational Research*, vol. 171, pp. 1–23, 5 2006.

[10] D. H. Lee, "Resource-based task allocation for multi-robot systems," *Robotics and Autonomous Systems*, vol. 103, pp. 151–161, 5 2018.

[11] C. Li, L. Zhang, and L. Zhang, "A route and speed optimization model to find conflict-free routes for automated guided vehicles in large warehouses based on quick response code technology," *Advanced Engineering Informatics*, vol. 52, 4 2022.

[12] E. Hu, J. He, and S. Shen, "A dynamic integrated scheduling method based on hierarchical planning for heterogeneous agv fleets in warehouses," *Frontiers in Neurorobotics*, vol. 16, 1 2023.