Article

A* Based Routing and Scheduling Modules for Multiple AGVs in an Industrial Scenario

Joana Santos [1], Paulo M. Rebelo [2,*], Luis F. Rocha [2], Pedro Costa [1,2] and Germano Veiga [1,2]

1 Faculty of Engineering, University of Porto (FEUP), 4200-465 Porto, Portugal; jooanaraquel@gmail.com (J.S.); pedrogc@fe.up.pt (P.C.); germanoveiga@fe.up.pt (G.V.)
2 Institute for Systems and Computer Engineering, Technology and Science (INESC TEC), 4200-465 Porto, Portugal; luis.f.rocha@inesctec.pt
* Correspondence: paulo.m.rebelo@inesctec.pt

Abstract: A multi-AGV based logistic system is typically associated with two fundamental problems, critical for its overall performance: the AGV's route planning for collision and deadlock avoidance; and the task scheduling to determine which vehicle should transport which load. Several heuristic functions can be used according to the application. This paper proposes a time-based algorithm to dynamically control a fleet of Autonomous Guided Vehicles (AGVs) in an automatic warehouse scenario. Our approach includes a routing algorithm based on the A* heuristic search (TEA*—Time Enhanced A*) to generate free-collisions paths and a scheduling module to improve the results of the routing algorithm. These modules work cooperatively to provide an efficient task execution time considering as basis the routing algorithm information. Simulation experiments are presented using a typical industrial layout for 10 and 20 AGVs. Moreover, a comparison with an alternative approach from the state-of-the-art is also presented.

Keywords: multi-robot coordination; automated guided vehicles; routing; scheduling; motion planning; simulation; robotics

check for updates

Citation: Santos, J.; Rebelo, P.M.; Rocha, L.F. ; Costa, P.; Veiga, G. A* Based Routing and Scheduling Modules for Multiple AGVs in an Industrial Scenario. *Robotics* **2021**, *10*, 72. https://doi.org/10.3390/robotics10020072

Academic Editor: Xinjun Liu

Received: 12 April 2021
Accepted: 13 May 2021
Published: 19 May 2021

1. Introduction

In recent years, and due to the mandatory need to continuously adapt the production flow, industrial companies are increasingly adopting fully automated internal logistic systems, namely based on AGVs, instead of manual or inflexible mechanical solutions (e.g., forklifts, conveyors, and others).

Considering the Industry 4.0 initiative, both AGVs, as well as mobile manipulators, are seen as strategic tools in the Factories of the Future. In a very competitive industrial environment, these can contribute to increase productivity and reduce the costs associated with the internal logistic system, ensuring an efficient material flow. Likewise, their introduction also allows human operators to be reallocated to more complex and ergonomic tasks, with increasing value to the final product. These set of characteristic makes AGV's appealing for a wide range of industrial applications, such as goods transportation, end-of-line automation chain, warehouse and distribution. However, and despite their versatility, there is the need to deploy advanced multi-robot coordination algorithms in order to ensure the AGV's continuous operation, guaranteeing the minimum tasks execution time and the smoothness of the vehicle's movements.

The AGV's fleet coordination problem has received wide attention from both the research and industrial fields. Typically, two main systems comprise any multi-AGV application: free-collision routing system [1,2] and scheduling system, that encompass both task scheduling and dispatching [3]. The vehicle routing system is responsible for computing trajectories that minimize the total distance traveled by AGVs considering different constraints such as each vehicle's carrying capacity and the plant layout where vehicles can circulate, while ensuring free-collision routes. Some approaches are based on time windows as proposed by the authors in [4–8]. Here, the AGV route is constructed considering that one point can only be visited one time for only one vehicle at a given time interval. The feasibility of each route is evaluated, checking windows overlapping. In its turn, the scheduling system is associated with the task scheduling and their attribution to individual AGVs. This scheduling and dispatching should take into account some decision criteria, namely the task deadlines and traffic status, among others.

Bearing these ideas in mind, this paper proposes an integrated approach for both AGV route planning and task scheduling and dispatching. More in detail, the proposed routing algorithm called Time Enhanced A* (TEA*) is an extension of our previous work addressed in [9,10], that is, in this paper, further integrated with a scheduling module in order to minimize the tasks execution time. The main feature of TEA* is the addition of a temporal component to the known A* algorithm that generates routes efficiently, considering that each robot knows other robots' positions during the time. Furthermore, TEA* is an on-line approach allowing its integration in dynamic environments.

The major contributions of our paper are to propose a Multi-Robot Coordination System which includes a time-based algorithm to generate free-collision routes based on the A* algorithm and a scheduling module that use the routing algorithm information to minimize a cost function, dependent on the following parameters:

1. Average execution time of all tasks;
2. Number of stoppages;
3. Execution time for the last vehicle;

This paper is organized as follows. In Section 2 the state-of-the-art of path planning algorithms and multi-robot systems are presented. Section 3 describes the proposed multi-robot routing algorithm. Section 4 describes the industrial case scenario, followed by the comparison between the proposed approach with an alternative state-of-the-art [11]. Sections 5 and 6 present the results achieved using Tabu-Search Method. Finally, some conclusions and the contribution of this paper are presented in Section 7.

2. Related Work

In the last decade, the multi-robot coordination problem has been a target of many scientific studies. To solve it, several authors have proposed the use of meta-heuristic approaches to address both the problem of AGV routing and task scheduling. The authors in [12–16], propose using a Genetic Algorithm to find the optimal or sub-optimal solution, which satisfies the routing system goals, including the minimization of the tasks completion time, minimal distance, among others. Likewise, ref. [17] proposes an integrated solution that comprises a routing module based on genetic algorithms and a scheduling system based on bid auctions. Despite the potential of the proposed methods, normally these approaches treat each robot as an individual agent (without physical constraints) [17,18], simplifying the problem at hand. Furthermore, the results do not include industrial case scenarios. Alternatively, particle swarm strategies can also be applied to multi-AGV systems as in [19,20]. Ref. [21] proposes a Particle Swarm Optimization (PSO) based algorithm, called Fractional Order Robotic Darwinian Particle Swarm Optimization (FORDPSO), integrated with a fuzzy system to optimize the driving of multi-robots in unknown environments. However, these methodologies are not yet sufficiently tested in real industrial environments.

Considering solely the use of a bid auction strategy, the authors in [3,22] propose a solution where each robot constructs 'bids' for each task and a central module receives the bids and assigns tasks considering the fitness function's maximization. Similarly, ref. [23] proposed different combinatorial bidding strategies, comparing its performance with single-item auctions.

Recognize the use of analytical methods, the authors in [24] define a robot mission through Linear Temporal Logic formulas (LTL). An LTL approach considers that the truth of a declaration can be changed during the time. This work focus on minimizing the cost function, which is the maximum time between candidate solutions of an optimizing proposition. In [25] the authors use a model based on Integer Linear Programming (ILP) to find paths that minimize the time until the last robot reaches its goal or minimizes the total traveled distance.

Generally, two different approaches define the architecture of any multi-robot system: a Centralized [26] or Distributed [27] methodology. The authors in [28,29] use a centralized architecture where one of the robots is the leader, and the others are the followers. Here, the major challenges are related to ensuring communication robustness and the algorithm flexibility to change the leadership. Other works use distributed architecture like in [27,30], where each robot calculates its path independently using, for example, a D* Algorithm [31] and then, the path is broadcast for all robots, making that every robot knows all path information.

Regarding only task scheduling system, it typically aims for minimizing an objective function which includes system characteristics, such as the number of vehicles, the order in which the vehicles execute their missions, etc. In [32], Kelen et al. compares two scheduling methods that determine the ideal number of vehicles for a given industrial scenario: the Shortest Job First and Tabu Search. Additionally, a routing method based on an enhanced Dijkstra algorithm, was used to manage the AGV's path. The number of stoppages and the time that each vehicle waits for the mission assignment was not measured.

In [20], Yu Zhang et al. proposes simple heuristics at the high-level layer, referred to as the 'coalition' level, that creates an abstraction layer relatively to specific details in robots' specifications. Simple heuristics can be 'MinProcTime', that gives priority to the missions with shorter processing times, and another one can be the 'MinStepSum', similar to the 'MinProcTime', but determines the best solution incrementally when the ordering of the assignment are not pre-determined. However, in the simulation experiments presented were not considered an industrial scenario with real-world scheduling problems.

In the past half-decade, new approaches based on Artificial Intelligence (AI) are emerging. AI addressed in [33,34], is the science that seeks to study and understand the phenomenon of intelligence and, at the same time, a branch of engineering, as it seeks to build instruments to support human intelligence. In practice, an AI system besides storing and manipulate data can also acquire, represent, and manipulate knowledge. This manipulation concerns the ability to deduce or infer new knowledge from existing knowledge and use representation and manipulation methods to solve complex problems. This area of engineering is vast and has been the subject of huge investment from both business and research institutes. More specifically in Robotics, there is already a recent line of work on Multi-Agent Path-Finding (MAPF) [35–38]. On AGVs, the MAPF problem is to find the best paths, for a fixed number of agents, from their current locations to the final task position, where all agents have a free-collision path, as it is described in [39–41]. Another use of AI in Robotics, is the combination of non-AI algorithms (time-window based or greedy) for the AGV coordination with AI for the prediction of future tasks [42].

Despite the many scientific studies carried out, task scheduling solutions often resort to small heuristics (First-in-First-Out, Shortest-Distance), often decoupled from the trajectory planning system. In turn, trajectories are predefined in offline mode and designed to prevent, as far as possible, the occurrence of deadlocks. This type of solution often leads to the logistics system being oversized concerning the operation's real needs.

3. Proposed Routing Algorithm

To overcome the challenges related with the multi-AGV coordination problem, presented in the previous sections, in this section a new methodology for AGV's route planning is proposed. This novel methodology is called TEA* Algorithm [9,10], in which the paths are recalculated continuously, making it an online method. According to the vehicles' movements and the environment changes, TEA* updates the paths of each AGV in order to avoid collisions and to guarantee the continuous operation of the logistic system. In fact, as in the majority of the industrial logistic systems, it is expectable that changes on initial task information and/or unpredictable events, such as delays in the transportation system as a result of obstacles presence [21], can occur. In these scenarios, the adoption of online path planning algorithms, capable of dealing with such events, becomes mandatory.

As referred before, TEA* is based on the traditional A*, where a third dimension was added, the time. The input map has three dimensions: vertex's coordinates (x and y) and a representation of the time, as shown in Figure 1. The time is represented with temporal layers given by $k \in [0, k_{Max}]$ (k_{max} denotes the maximum number of layers). Each temporal graph is a set of free and occupied/obstacles vertexes.

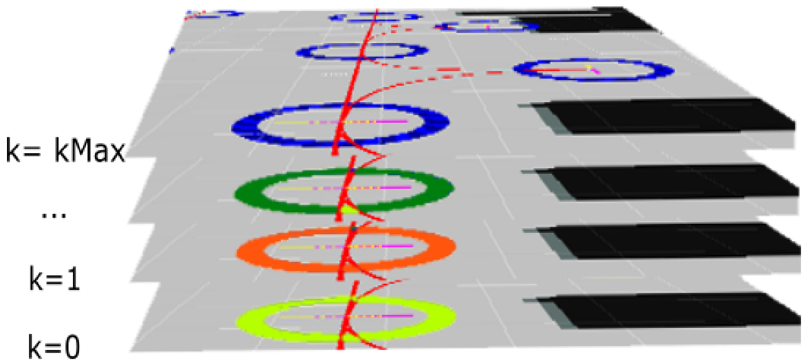

Figure 1. Representation of the input map focusing the vertexes with the same position of the AGV (over the time) denoted with different colors.

As far as computational complexity is concerned, many factors can influence the system (RAM, CPU, others), however, through [43] where the test conditions are the same for all algorithms, it is possible to validate that A* is the most suitable algorithm for robot fleet management in different environments. Therefore, taking into account the case study carried out in [43], and the results obtained in [9], it can be concluded that TEA* is a practical, versatile and quite optimized algorithm in terms of path planning algorithms.

3.1. *The TEA* Method*

In multi-AGV systems, time is a crucial component for a better prediction of the vehicles' positions. Besides the constantly recalculated paths, TEA* determines the route for each AGV during the temporal layers. This grants the identification of upcoming collisions, allowing them to be avoided with considerable anticipation. The path information for each AGV is converted to a busy vertex on the following robot's map, allowing collision avoidance.

Consider a graph G with a set of vertexes $V = \{0, 1, ..., NUM\_VERTEXES\}$ and edges $E = \{0, 1, ...NUM\_EDGES\}$ (links between the vertexes), with a representation of the time $[\,0, 1, ..., k_{max}\,]$ (as can be seen in Figure 1). Each AGV can only starts and stops in vertexes and each vertex can only be occupied by only one vehicle at the time.

During the path search for a single AGV the neighboring vertexes are evaluated using a similar approach as A* algorithm [44]. Moreover, each edge in each temporal layer, has a cost function value, denoted as $f(j,k)$, given by the sum of two terms (see Equation (1)).

$$f(j,k) = \alpha g(j,k) + \beta h(j,k), \qquad k \in [0, k_{max}], j \in [0, NUM\_VERTEXES] \quad (1)$$

Considering the path between j_0 and j_f, the first term $\alpha g(j,k)$, represents the distance between the current vertex j to the initial vertex j_0, in the k temporal layer. The second term, denoted as $\beta h(j,k)$, is a heuristic value that calculates the distance to the final vertex j_f. The terms α and β assign different weights to the distance and the heuristic function.

Each vertex, in each temporal layer, has different values of $g(j,k)$ and $h(j,k)$, according with the Equation (2). Here, $g(j,k)$ is given by the sum of the distance between the current vertex j and the initial vertex j_0, being the edge distance between j and its adjacent vertex $j+1$ denoted as $dis(j, j+1, k)$.

$$
\begin{aligned}
g(j,k) &= dis(j, j_0, k) + dis(j, j+1, k) \\
h(j,k) &= dis(j, j_f, k)
\end{aligned}
\tag{2}
$$

The main differences between TEA* and the known A* algorithm are mainly concerned with the addition of the time component and can be defined as follows:

Definition 1. *The neighbor vertexes belong to the next temporal layer. The neighbor vertexes of a vertex j (v^j_{adj}) are given by the set of all adjacent vertexes in the next time component ($k+1$). The number of temporal layers depends on the required iterations to achieve the final point of the mission and the map dimensions. Note that the larger the map, more time layers are required.*

Definition 2. *The neighbor vertexes include the vertex containing the AGV's current position. The set of neighbor vertexes includes not only the adjacent vertexes but also the vertex corresponding to the position in analysis. This property allows a vehicle to maintain its position between consecutive time instants if any neighbor vertex is free. In this case, $dis(j, j, k+1)$ assumes a constant value that corresponds to the cost of keeping its position.*

The Algorithm 1 describes the TEA* approach for a single AGV with the following parameters:

- val^k_j: Value of vertex j in the time layer k (Free—0 or Occupied/Obstacle—1).
- $pos^k_{l,j}$: AGV l occupies the vertex j in the k time layer.
- $O = \{o^k_j, ..\}$: Open list contains the vertex j in the k time instant. Each item contains the respective cost value, $o^k_j.cost$.
- j_0: initial vertex.
- j_f: final vertex.
- v^j_{adj}: adjacent vertex of vertex j.
- $p_{j,k} = (i, \tau)$: The vertex i in the time layer τ is the parent vertex of vertex j in the instant k.
- h^k_j: Heuristic Value for vertex j in k temporal layer.
- g^k_j: Distance Value for vertex j in k temporal layer.
- $dis(j_1, j_2)$: Distance value of the edge (j_1, j_2).

3.2. Smoothing Trajectories

For TEA* to be applied, the shop floor layout requires to be modeled as a set of vertexes and edges (links between the vertexes). Each AGV travels on these graph paths, from one node to another, through a pre-defined set of edges. Each edge is represented as a cubic Bézier curve (as proposed in article [45]), given by Equation (3).

$$
\begin{aligned}
x(\lambda) &= a_x\lambda^3 + b_x\lambda^2 + c_x\lambda + x_0 \\
y(\lambda) &= a_y\lambda^3 + b_y\lambda^2 + c_y\lambda + y_0
\end{aligned}
\tag{3}
$$

Here, λ denotes an integer value between 0 and 1 according with the AGV's position in the curve, (x_0, y_0) is the initial point of the curve, and a_x, b_x, c_x, defines the spline's curvature. Figure 2 represents a portion of the map which contains Bézier curves and straight lines.

Algorithm 1: TEA* ALGORITHM

```
$O \leftarrow o^0_{vi}$;
while $OpenList.size() \neq 0$ do
    $j = min_O\{o^k_j.cost\}$;
    if $j == j_f$ then
        return
    for $v^j_{adj}$ adjacent vertexes of $j$ do
        if $val^{k+1}_{v^j_{adj}} == 0$ then
            Only the non-visited vertexes have heuristic zero;
            if $h^{k+1}_{v^j_{adj}} == 0$ then
                $CalculateHeuristic(h^{k+1}_{v^j_{adj}})$;
                $p_{j,k} = (v^j_{adj}, k+1)$;
                $CalculateCost(o^{k+1}_{v^j_{adj}})$;
                $O \leftarrow o^{k+1}_{v^j_{adj}}$;
            else if $g^{k+1}_{v^j_{adj}} > g^k_{v^j_{adj}} + dis(j, v^j_{adj})$ then
                $UpdateCost(o^{k+1}_{v^j_{adj}})$;
                $O \leftarrow o^{k+1}_{v^j_{adj}}$
return $0$;
```

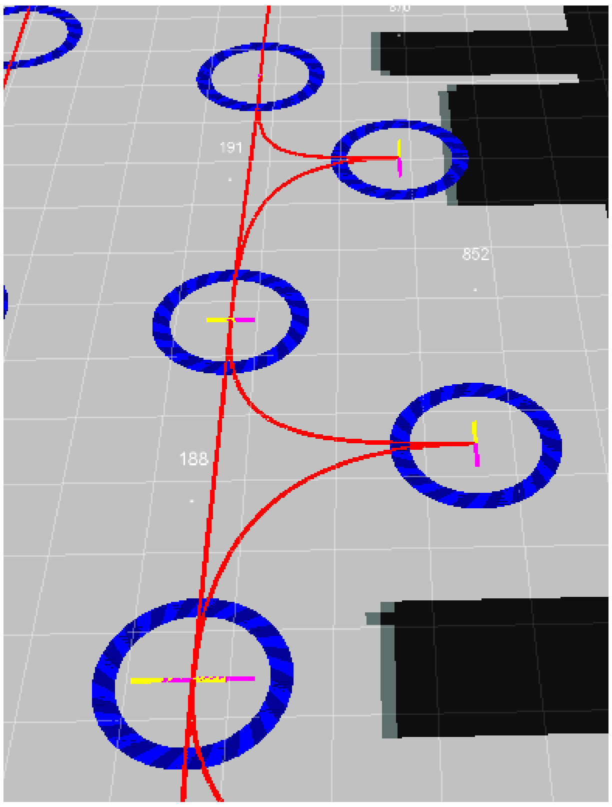

Figure 2. Cubic Bézier Curves Example.

4. Industrial Layout and Comparison Scenario Description

In this section, the TEA*, based on Robot Operating System (ROS), is compared with a state of the art alternative, namely the coordination algorithm presented in [46]. This algorithm relies on coordination diagrams for planning the coordinated motion of a fleet of AGVs. One of the contributions of [46] is the definition of a heuristic function that estimates the number of times a vehicle starts and stops during its path execution.

For the comparison of the two algorithms, a set of experiments were conducted using the same layout, the same number of vehicles and the same missions' list used by the authors of [11]. Each mission is defined by four tasks (*Sn*- Starting positions; *Pn*- Pick-up Positions; *Dn*- Drop-off Stations; *Rn*- Rest Positions). The layout dimensions are 80 $\times$ 110 m and 10 AGVs were used to generate the results of [11].

Figure 3 represents the input map of TEA* Algorithm. The graph was built using a graph editor, which was created with the Robot Operating System Visualization (RVIZ) platform and the Interactive Markers tool.

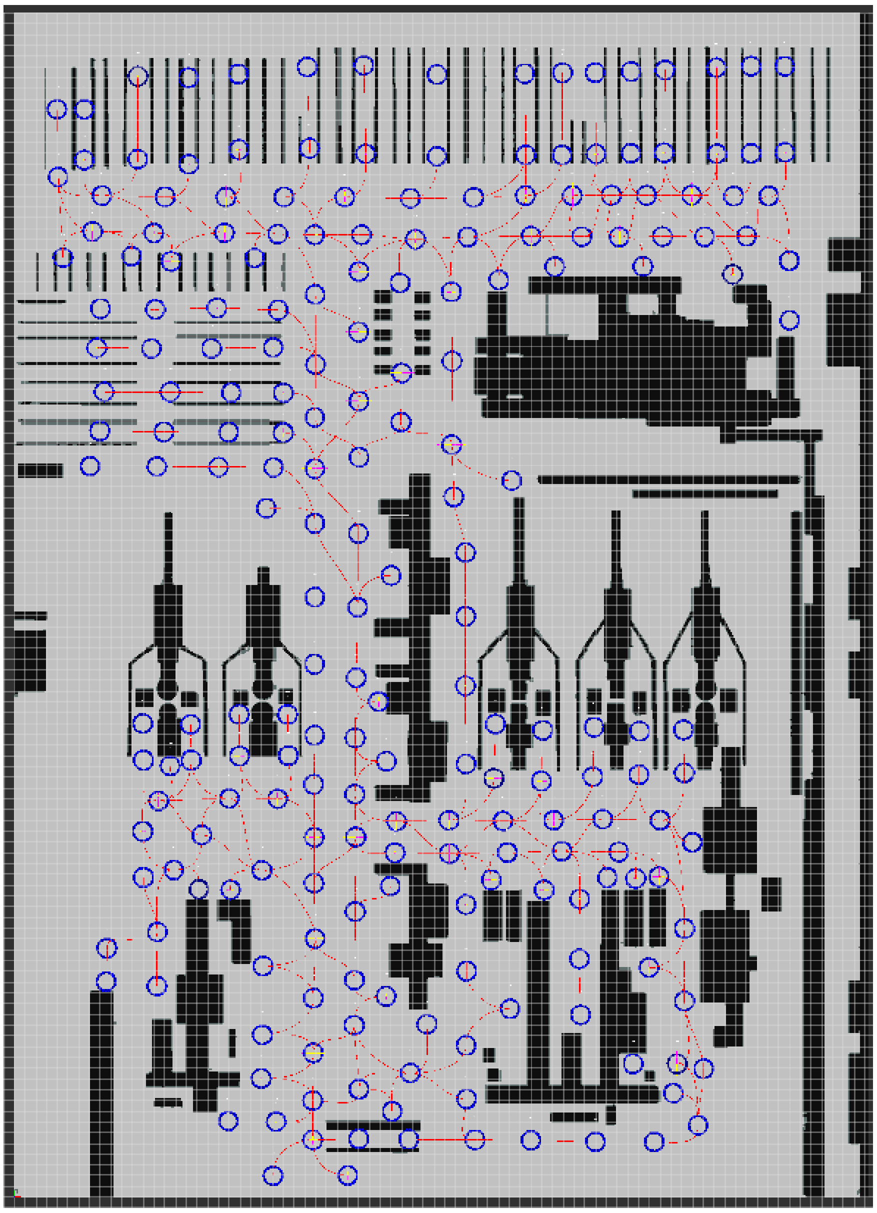

Figure 3. Layout Snapshot built in RVIZ.

5. Routing Algorithms Comparison

Considering the industrial scenario previously presented, Table 1 illustrates the mission execution time for each vehicle with the TEA* Algorithm, using 10 AGVs and 30 tasks. For each AGV, the time of advancement (T_{adv}) and the stopping time (T_{stop}) are reported. In Figure 4 is illustrated the final solution found for each vehicle. Two black circles represent possible collision points in different robot paths, but the third dimension of TEA* allows AGVs to share the same trajectory by passing at different times at the common points, i.e., avoiding possible collisions.

Comparing the results achieved with the TEA* (Table 1), with the results presented by the authors in [11] (Table 2), it is possible to conclude that TEA* is advantageous mainly considering the T_{adv} of the last vehicles (AGVs 6, 7, 8, 9, 10). Conversely, the T_{adv} of the AGVs 3, 4 and 5 surpasses the value of the same robots in Table 2. However, in these cases the $T_{average}$ and the T_{max} are lower in TEA*. Therefore, this algorithm presents itself as a better solution for multi-robot path planning.

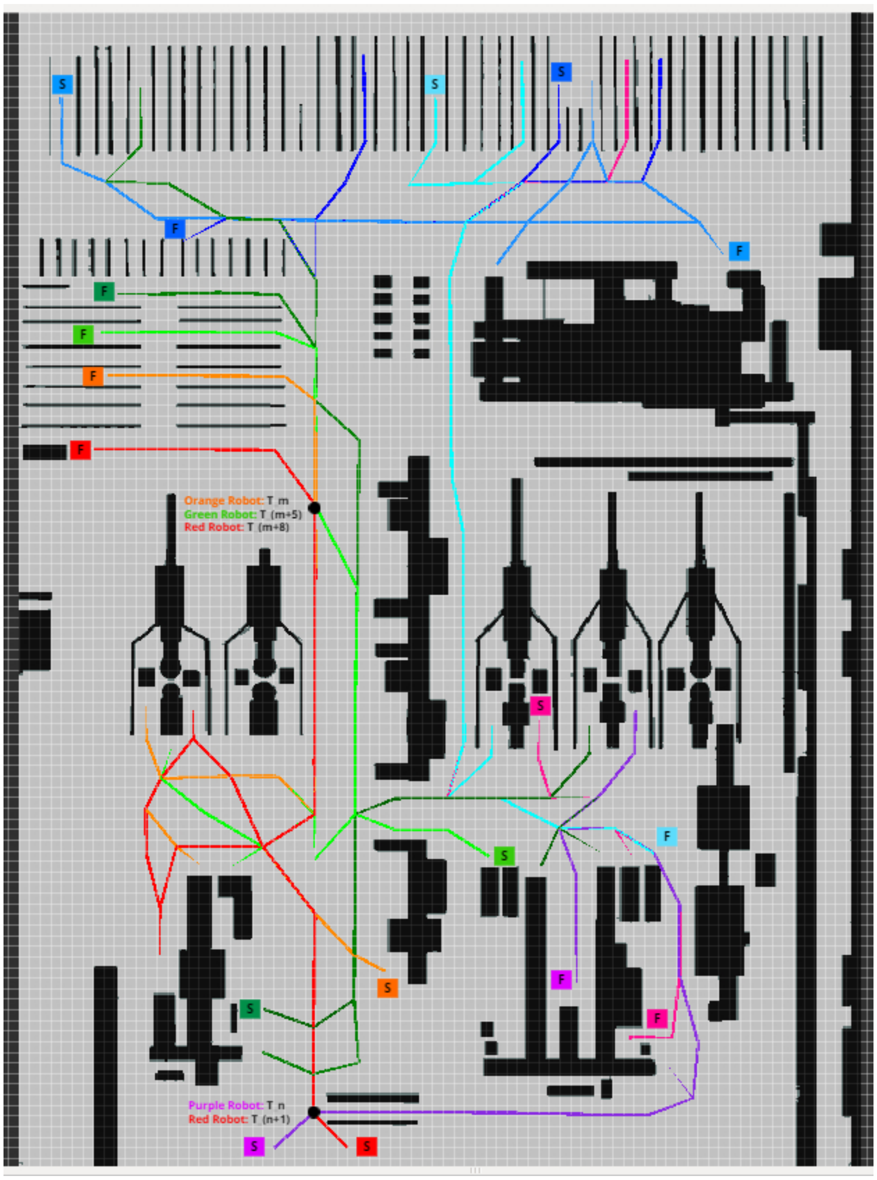

Figure 4. Routing Algorithm Results—10 AGVs.

The average time ($T_{average}$) to complete the missions list with TEA* is 116.7 s and in the case of [11] this time is approximately 121.78 s. The last vehicle in [11] takes 144.2 s to finish its tasks, while in TEA*, the last vehicle performs its tasks in 135.8 s.

To highlight the capability of TEA* Algorithm to optimize the routes even with considerable workload in the system, Table 3 presents the same industrial scenario but now with 20 AGVs and 60 tasks. Note that the difference in the average time between 10 and 20 AGVs are approximately 11 s. The fact that a vehicle considers the current and future positions of each vehicle as obstacles during the discrete-time, gives it the possibility to wait for some instants for the traveling of previous AGV, instead of calculating a longer deviation.

Table 1. TEA* Results—10 AGVs.

Vehicle	**1**	**2**	**3**	**4**	**5**
$T_{adv}(s)$	123.5	110.3	135.8	134.2	91.6
$T_{stop}(s)$	0	0	6	0	3
Vehicle	**6**	**7**	**8**	**9**	**10**
$T_{adv}(s)$	129.0	103.4	105.4	108.6	125.6
$T_{stop}(s)$	0	0	6	0	0
$\boldsymbol{T_{average}(s)}$	**116.7**				
$\boldsymbol{T_{max}(s)}$	**135.8**				
$\boldsymbol{T_{stop}(s)}$	**15**				

Table 2. TRAFCON Results—10 AGVs, adapted from [11].

Vehicle	**1**	**2**	**3**	**4**	**5**
$T_{adv}(s)$	144.2	118.6	115.2	127	73.4
$T_{stop}(s)$	26.8	0	1.6	0	17
Vehicle	**6**	**7**	**8**	**9**	**10**
$T_{adv}(s)$	151	123	107	121.6	136.8
$T_{stop}(s)$	0	0	1.4	0	0
$\boldsymbol{T_{average}(s)}$	**121.8**				
$\boldsymbol{T_{max}(s)}$	**151**				
$\boldsymbol{T_{stop}(s)}$	**46.8**				

For the sake of completeness, it is important to refer that the TEA* Algorithm has been designed to be integrated into dynamic industrial environments, thus allowing direct scaling concerning the number of robots to be used. The structure of the algorithm itself is already prepared for different exchanges of industrial scenarios.

Table 3. TEA* Results—20 AGVs.

Vehicle	**1**	**2**	**3**	**4**	**5**
$T_{adv}(s)$	123.5	110.3	135.8	134.2	91.6
$T_{stop}(s)$	0	0	6	0	3
Vehicle	**6**	**7**	**8**	**9**	**10**
$T_{adv}(s)$	129.0	103.4	105.4	108.6	125.6
$T_{stop}(s)$	0	0	6	0	0
Vehicle	**11**	**12**	**13**	**14**	**15**
$T_{adv}(s)$	144.3	166.6	107.7	129.6	121.5
$T_{stop}(s)$	9	3	0	12	0
Vehicle	**16**	**17**	**18**	**19**	**20**
$T_{adv}(s)$	161.4	139.5	179.3	81.5	154.7
$T_{stop}(s)$	3	0	3	3	3
$\boldsymbol{T_{average}(s)}$	**127.7**				
$\boldsymbol{T_{max}(s)}$	**179.3**				
$\boldsymbol{T_{stop}(s)}$	**51**				

6. Task Scheduling Algorithm

As referred earlier, the problem of AGV coordination is not only closely related with the route planning of the AGVs, but also with the task scheduling. The performance of the routing algorithm can be improved. The AGV execution order affects the calculation of routes since the path positions over time for a given vehicle are obstacles for the following AGVs. If the execution order changes, the paths and respective task execution times for each vehicle are different.

6.1. Tabu Search Method

The Tabu Search Method is a 'meta-heuristic' adaptive method of local search in continuous exploration within a search space, moving from one solution to another, the Tabu Moves, diversifying the solutions found in this process of the search for an improved solution [47]. The best permissible movement is the one with the highest evaluation in the vicinity of the current solution regarding target function value and taboo restrictions. Thus, the 'meta-heuristic' Tabu Search is an iterative search algorithm characterized by dynamic memory and consisting of two parts: initialization and search.

Starting from an initial randomly generated solution or using a heuristic, the Tabu Search will evaluate a set of different mutations (neighborhood exploration) of the current solution in each iteration. The best mutation will be accepted, and the changes made saved in a Tabu List adopted to store the most used changes, which are classified as prohibited in later iterations. This strategy is necessary to avoid a return to solutions already checked previously.

Therefore, in this method, in each iteration, the evaluation function consists of validating a certain quantity of new solutions, where the best solution, based on the objective function, is accepted, even if its cost is higher than the cost of the current solution. Thus, the algorithm chooses the new solution that produces an improvement or the least deterioration in the cost function (an attempt to evade minimal locations). The Tabu Search algorithm runs until a stop criteria is reached.

Figure 5 presents the main blocks of the Tabu Search method.

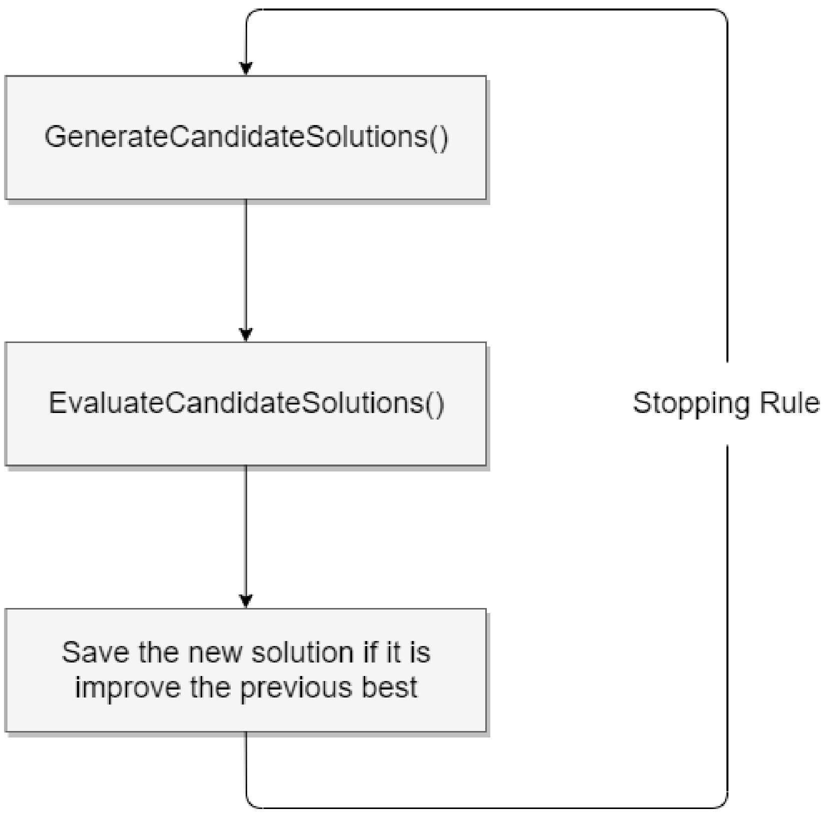
GenerateCandidateSolutions()
EvaluateCandidateSolutions()
Save the new solution if it is improve the previous best
Stopping Rule

Figure 5. Tabu Search Diagram.

In the AGV scheduling problem presented, the final objective is the allocation of the n sub-tasks by the j available AGVs, in order to minimize the total time for the overall task completion. The main goal is to determine/distribute the best sequence of attendance of the sub-tasks by the AGVs to minimize the total time of execution.

For the problem presented, the initial solution (sequence of sub-tasks assigned to each AGV) is generated using the closest neighboring heuristic, as presented in Algorithm 2. In other words, each AGV (and taking into account the last sub-task performed, which influences its position on the map) is assigned to the next sub-task with lower cost (shorter travel time/distance).

This process is executed cyclically until all sub-tasks have been assigned to one, and only one, AGV.

Algorithm 2: CLOSEST NEIGHBOUR ALGORITHM—PSEUDO CODE

```
1 while $AllSubTasksUnallocated$ do
2     for $j = 1$ to $NumAGVs$ do
3         $NewAGV(j)SubTask \leftarrow NeighbourNext <$
            $SubTasksUnallocated, PreviousSubTaskAGV(j) >;$
4         $ListSubTasksAGV(j) \leftarrow< NewSubTaskAGV(j) >;$
5         $PreviousAGV(j)SubTask \leftarrow NewAGV(j)SubTask;$
6         $AllUnassignedSubTasks \leftarrow delete < NewSubTaskAGV(j) >;$
```

The Algorithm 3, describes the implementation of the Tabu Search used in the proposed approach.

6.2. TEA* Algorithm with Tabu Search Method—Results

The Tabu Search Method was implemented to find better vehicle configuration and to schedule the order in which vehicles execute their tasks. A configuration comprises the order in which vehicles should be processed by the TEA* Algorithm.

Algorithm 3: 'META-HEURISTIC' TABU SEARCH—PSEUDO CODE

```
1  $s \leftarrow s0;$
2  $BestSolution \leftarrow s;$
3  for $k = 1$ to $TabuSearchMaxIteration$ do
4      $CandidateList \leftarrow null;$
5      for $sCandidate$ in $sNeighborhood$ do
6          if not $containsTabuElements < Candidate, tabuList >$ then
7              $candidateList \leftarrow candidateList + sCandidate;$
8      $sCandidate \leftarrow LocateBestCandidate < candidateList >;$
9      $tabuList \leftarrow$ addFeatureDifferences $< sCandidate, sBest >;$
10     $s \leftarrow sCandidate;$
11     if $fitness < s >$ < $fitness < sBest >$ then
12         $sBest \leftarrow s;$
13     UpdateTabuList $< tabuList >$;
14 return $sBest$
```

The AGV execution order affects the calculation of routes since the path positions over time for a given vehicle are obstacles for the following AGVs. If the execution order changes, each vehicle's paths and respective task execution times will be different.

The optimization goal is the minimization of the three following parameters:

1. Time of the last vehicle, denoted as T_{max} in seconds;
2. Average Time of the missions execution, denoted as $T_{average}$ in seconds;
3. Number of Stoppages, denoted as n_{stop};

That were aggregated in the following cost function (Equation (4)):

$$c = \gamma \times T_{last} + \psi \times T_{average} + \tau \times n_{stop} \tag{4}$$

Here, γ, ψ and τ are components that are weighting parameters. In the simulation experiments the following values were used, respectively 0.1, 0.7, 0.2. These were manually defined considering an iterative and experimental way, with the main goal of minimizing the cost function c.

The configuration that leads to the lowest cost function is chosen as the better solution. In our approach is not required to achieve the optimal solution, a near-optimal configuration that leads to an efficient TEA* execution is enough.

To obtain the initial configuration of the Tabu Search Method, a heuristic function was defined. It consists of the path computation for each vehicle without consider the other vehicles as obstacles (optimal solution) and ordering it by decreasing the order of execution task times. The objective is to process firstly the longer paths minimizing the number of stoppages. The candidate solutions are generated, changing two by two the vehicle execution order from the current solution.

Table 4 presents the results for the TEA* Algorithm using the AGV configuration solution found by the Tabu Search method. The average time for completing all tasks is lower, but the more significant improvement was the waiting time. In Table 1 using as initial configuration = $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ AGVs were stopped 15 s. Using the Tabu Search solution = $\{6, 2, 4, 3, 1, 10, 9, 7, 8, 5\}$, total waiting time was 6 s.

Considering 20 AGVs, besides the stoppage time to be higher, the task completion time is lower than TEA* results without scheduling module. In Table 3 the total time to complete all tasks was 179.3 s and in Table 5 this time was 159.1 s.

Table 4. TEA* Results with the Tabu Search Configuration—10 AGVs.

Vehicle	**6**	**2**	**4**	**3**	**1**
$T_{adv}(s)$ $T_{stop}(s)$	129.0 0	110.3 0	133.7 0	122.6 0	125.4 0
Vehicle	**10**	**9**	**7**	**8**	**5**
$T_{adv}(s)$ $T_{stop}(s)$	124.2 0	108.6 0	103.4 0	105.4 3	91.3 3
$\boldsymbol{T_{average}}$	**115.4**				
$\boldsymbol{T_{max}(s)}$	**133.7**				
$\boldsymbol{T_{stop}(s)}$	**6**				

Table 5. TEA* Results with the Tabu Search Configuration—20 AGVs.

Vehicle	**18**	**16**	**4**	**12**	**6**
$T_{adv}(s)$ $T_{stop}(s)$	159.1 0	152.5 0	133.9 0	157.3 6	131.2 0
Vehicle	**11**	**10**	**1**	**15**	**17**
$T_{adv}(s)$ $T_{stop}(s)$	133.7 6	124.2 0	123.5 0	125.8 9	136.1 0
Vehicle	**3**	**14**	**2**	**9**	**13**
$T_{adv}(s)$ $T_{stop}(s)$	148.5 0	131.6 12	136.0 3	119.1 0	138.4 3
Vehicle	**20**	**7**	**8**	**5**	**19**
$T_{adv}(s)$ $T_{stop}(s)$	145.1 3	121.4 0	106.7 3	105.0 18	78.5 0
$\boldsymbol{T_{average}}$	**130.4**				
$\boldsymbol{T_{max}(s)}$	**159.1**				
$\boldsymbol{T_{stop}(s)}$	**63**				

7. Conclusions

This article proposes a multi-AGV system that comprises a routing algorithm based on the search method A*. This algorithm is suitable for multi-robot applications, avoiding collisions and deadlocks and guaranteeing any industrial scenario required safety levels. To optimize the results achieved scheduling method (Tabu Search) minimizes a fitness function defined by several parameters calculated by TEA*. The two modules work cooperatively, sharing the TEA* information.

Our work's major contributions are: (i) Presentation of a promising approach for multi-AGV applications in warehouse environment, improving the flexibility and efficiency of the complete system; (ii) Validation of an on-line Multi-Robot Coordination Algorithm comparing it with a state-of-the-art alternative.

As future work, it will be interesting to validate the TEA* Algorithm in a real environment, with a real robotic system, by comparing it to the simulation results.

Author Contributions: The contributions of the authors of this work are pointed as follows: Conceptualization: J.S., L.F.R., P.C. and G.V.; Methodology: J.S., L.F.R., P.C. and G.V.; Software J.S., P.M.R. and P.C.; Validation: J.S., P.M.R. and P.C.; Writing—Review and Editing: J.S., P.M.R., L.F.R. and P.C.; Supervision: L.F.R., P.C. and G.V. All authors have read and agreed to the published version of the manuscript.

Funding: The research leading to these results has received funding from the European Union's Horizon 2020—The EU Framework Programme for Research and Innovation 2014–2020, under grant agreement No.101006798—Project `Mari4_YARD`.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: This study didn't report any new data.

Acknowledgments: The authors of this work would like to thank the members of INESC TEC for all the support rendered to this project.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AGV	Autonomous Guided Vehicle
AI	Artificial Intelligence
FORDPSO	Fractional Order Robotic Darwinian Particle Swarm Optimization
ILP	Integer Linear Programming
LTL	Linear Temporal Logic
MAPF	Multi-Agent Path-Finding
MDPI	Multidisciplinary Digital Publishing Institute
PSO	Particle Swarm Optimization
ROS	Robot Operating System
RVIZ	Robot Operating System Visualization
TEA*	Time Enhanced A*

References

1. Kalinovcic, L.; Petrovic, T.; Bogdan, S.; Bobanac, V. Modified Banker's algorithm for scheduling in multi-AGV systems. In Proceedings of the IEEE International Conference on Automation Science and Engineering, Hong Kong, China, 20–21 August 2011. [CrossRef]
2. Cirillo, M.; Pecora, F.; Andreasson, H.; Uras, T.; Koenig, S. Integrated motion planning and coordination for industrial vehicles. In Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS), Portsmouth, NH, USA, 21–26 June 2014.
3. Erol, R.; Sahin, C.; Baykasoglu, A.; Kaplanoglu, V. A multi-agent based approach to dynamic scheduling of machines and automated guided vehicles in manufacturing systems. *Appl. Soft Comput. J.* **2012**, *12*, 1720–1732. [CrossRef]
4. Bräysy, O.; Gendreau, M. Vehicle routing problem with time windows, Part I: Route construction and local search algorithms. *Transp. Sci.* **2005**, *39*, 104–118. [CrossRef]
5. Möhring, R.H.; Köhler, E.; Gawrilow, E.; Stenzel, B. *Conflict-Free Real-Time AGV Routing*; Springer: Berlin/Heidelberg, Germany, 2005. [CrossRef]
6. Smolic-Rocak, N.; Bogdan, S.; Kovacic, Z.; Petrovic, T. Time windows based dynamic routing in multi-AGV systems. *IEEE Trans. Autom. Sci. Eng.* **2010**, *7*, 151–155. [CrossRef]
7. Ballal, P.; Giordano, V.; Lewis, F. Deadlock free dynamic resource assignment in multi-robot systems with multiple missions: A matrix-based approach. In Proceedings of the 14th Mediterranean Conference on Control and Automation (MED'06), Ancona, Italy, 28–30 June 2006. [CrossRef]
8. Guan, X.; Li, Y.; Xu, J.; Wang, C.; Wang, S. A literature review of deadlock prevention policy based on petri nets for automated manufacturing systems. *Int. J. Digit. Content Technol. Its Appl.* **2012**, *6*, 426–433. [CrossRef]
9. Santos, J.; Costa, P.; Rocha, L.F.; Moreira, A.P.; Veiga, G. Time enhanced A: Towards the development of a new approach for Multi-Robot Coordination. In Proceedings of the IEEE International Conference on Industrial Technology, Seville, Spain, 17–19 March 2015; pp. 3314–3319. [CrossRef]
10. Santos, J.; Costa, P.; Rocha, L.; Vivaldini, K.; Moreira, A.P.; Veiga, G. Validation of a time based routing algorithm using a realistic automatic warehouse scenario. In Proceedings of the Advances in Intelligent Systems and Computing, Lviv, Ukraine, 6–10 September 2016. [CrossRef]
11. Olmi, R.; Secchi, C.; Fantuzzi, C. Coordinating the motion of multiple AGVs in automatic warehouses. In Proceedings of the Workshop on Robotics and Intelligent Transportation Systems, Anchorage, Alaska, 8 May 2010.
12. Liu, C.; Kroll, A. A centralized multi-robot task allocation for industrial plant inspection by using A* and genetic algorithms. In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*; Springer: Berlin/Heidelberg, Germany, 2012. [CrossRef]

13. Tuncer, A.; Yildirim, M. Dynamic path planning of mobile robots with improved genetic algorithm. *Comput. Electr. Eng.* **2012**, *38*, 1564–1572. [CrossRef]
14. Kapanoglu, M.; Alikalfa, M.; Ozkan, M.; Yazıcı, A.; Parlaktuna, O. A pattern-based genetic algorithm for multi-robot coverage path planning minimizing completion time. *J. Intell. Manuf.* **2012**, *23*, 1035–1045. [CrossRef]
15. Zafar, M.N.; Mohanta, J.C. Methodology for Path Planning and Optimization of Mobile Robots: A Review. *Procedia Comput. Sci.* **2018**, *133*, 141–152. [CrossRef]
16. Kala, R. Multi-robot path planning using co-evolutionary genetic programming. *Expert Syst. Appl.* **2012**, *39*, 3817–3831. [CrossRef]
17. Jones, E.G.; Dias, M.B.; Stentz, A. Time-extended multi-robot coordination for domains with intra-path constraints. *Auton. Robot.* **2011**, *30*, 41–56. [CrossRef]
18. Fauadi, M.H.F.B.M.; Yahaya, S.H.; Murata, T. Intelligent combinatorial auctions of decentralized task assignment for AGV with multiple loading capacity. *IEEJ Trans. Electr. Electron. Eng.* **2013**, *8*, 371–379. [CrossRef]
19. Sun, D.; Kleiner, A.; Nebel, B. Behavior-based multi-robot collision avoidance. In Proceedings of the IEEE International Conference on Robotics and Automation, Hong Kong, China, 31 May–7 June 2014. [CrossRef]
20. Zhang, Y.; Gong, D.W.; Zhang, J.H. Robot path planning in uncertain environment using multi-objective particle swarm optimization. *Neurocomputing* **2013**, *103*, 172–185. [CrossRef]
21. Wang, D.; Wang, H.; Liu, L. Unknown environment exploration of multi-robot system with the FORDPSO. *Swarm Evol. Comput.* **2016**, *26*, 157–174. [CrossRef]
22. Simmons, R.; Apfelbaum, D.; Burgard, W.; Fox, D. Coordination for multi-robot exploration and mapping. In Proceedings of the Aaai/Iaai, Austin, TX, USA, 1–3 August 2000.
23. Berhault, M.; Huang, H.; Keskinocak, P.; Koenig, S.; Elmaghraby, W.; Griffin, P.; Kleywegt, A. Robot Exploration with Combinatorial Auctions. In Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Las Vegas, NV, USA, 27 October–1 November 2003. [CrossRef]
24. Ulusoy, A.; Smith, S.L.; Ding, X.C.; Belta, C.; Rus, D. Optimality and robustness in multi-robot path planning with temporal logic constraints. *Int. J. Robot. Res.* **2013**, *32*, 889–911. [CrossRef]
25. Yu, J.; Lavalle, S.M. Planning optimal paths for multiple robots on graphs. In Proceedings of the IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013. [CrossRef]
26. Chaimowicz, L.; Sugar, T.; Kumar, V.; Campos, M.F. An architecture for tightly coupled multi-robot cooperation. In Proceedings of the IEEE International Conference on Robotics and Automation, Seoul, Korea, 21–26 May 2001. [CrossRef]
27. Guo, Y.; Parker, L.E. A distributed and optimal motion planning approach for multiple mobile robots. In Proceedings of the IEEE International Conference on Robotics and Automation, Washington, DC, USA, 11–15 May 2002. [CrossRef]
28. Jose, K.; Pratihar, D.K. Task allocation and collision-free path planning of centralized multi-robots system for industrial plant inspection using heuristic methods. *Robot. Auton. Syst.* **2016**, *80*, 34–42. [CrossRef]
29. Zhang, X.; Yan, M.; Ju, Y. A tabu search based flocking algorithm of motion control for multiple mobile robots. In Proceedings of the 2012 5th International Conference on Intelligent Computation Technology and Automation (ICICTA 2012), Zhangjiajie, China, 12–14 January 2012. [CrossRef]
30. Fierro, R.; Das, A.K.; Kumar, V.; Ostrowski, J.P. Hybrid control of formations of robots. In Proceedings of the IEEE International Conference on Robotics and Automation, Seoul, Korea, 21–26 May 2001. [CrossRef]
31. Stentz, A. Optimal and efficient path planning for partially-known environments. In Proceedings of the IEEE International Conference on Robotics and Automation, San Diego, CA, USA, 8–13 May 1994. [CrossRef]
32. Vivaldini, K.; Rocha, L.F.; Martarelli, N.J.; Becker, M.; Moreira, A.P. Integrated tasks assignment and routing for the estimation of the optimal number of AGVS. *Int. J. Adv. Manuf. Technol.* **2016**, *82*, 719–736. [CrossRef]
33. Nilsson, N.J. *Principles of Artificial Intelligence*; Morgan Kaufmann: Burlington, MA, USA, 2014.
34. Mitchell, R.; Michalski, J.; Carbonell, T. *An Artificial Intelligence Approach*; Springer: Berlin/Heidelberg, Germany, 2013.
35. Atzmon, D.; Stern, R.; Felner, A.; Wagner, G.; Barták, R.; Zhou, N.F. Robust multi-agent path finding and executing. *J. Artif. Intell. Res.* **2020**. [CrossRef]
36. Li, J.; Sun, K.; Ma, H.; Felner, A.; Satish Kumar, T.K.; Koenig, S. Moving agents in formation in congested environments. In Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS), Auckland, New Zealand, 9–13 May 2020.
37. Atzmon, D.; Stern, R.; Felner, A.; Sturtevant, N.R.; Koenig, S. Probabilistic robust multi-agent path finding. In Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS), Nancy, France, 14–19 June 2020.
38. Liu, M.; Ma, H.; Li, J.; Koenig, S. Task and path planning for multi-agent pickup and delivery. In Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS), Montreal, QC, Canada, 13–17 May 2019.
39. Surynek, P. A novel approach to path planning for multiple robots in bi-connected graphs. In Proceedings of the IEEE International Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009. [CrossRef]
40. Wagner, G.; Kang, M.; Choset, H. Probabilistic path planning for multiple robots with subdimensional expansion. In Proceedings of the IEEE International Conference on Robotics and Automation, Saint Paul, MN, USA, 14–18 May 2012. [CrossRef]
41. Barták, R.; Švancara, J.; Škopková, V.; Nohejl, D. Multi-agent path finding on real robots: first experience with ozobots. In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*; Springer: Cham, Switzerland, 2018. [CrossRef]

42. Li, D.; Ouyang, B.; Wu, D.; Wang, Y. Artificial intelligence empowered multi-AGVs in manufacturing systems. *arXiv* **2019**, arXiv:1909.03373.
43. Sasi Kumar, G.; Shravan, B.; Gole, H.; Barve, P.; Ravikumar, L. *Path Planning Algorithms: A Comparative Study*; Space Transportation Systems Division: Houston, TX, USA, 2011.
44. Moreira, A.P.; Costa, P.J.; Costa, P. Real-time path planning using a modified A* algorithm. In Proceedings of the 9th Conference on Mobile Robots and Competitions, Castelo Branco, Portugal, 7 May 2009; pp. 141–146.
45. Petrinec, K.; Kovačić, Z. The application of spline functions and bézier curves to AGV path planning. In Proceedings of the IEEE International Symposium on Industrial Electronics, Dubrovnik, Croatia, 20–23 June 2005. [CrossRef]
46. Secchi, C.; Olmi, R.; Fantuzzi, C.; Casarini, M. TRAFCON—Traffic control of AGVs in automatic warehouses. In *Springer Tracts in Advanced Robotics*; Springer: Cham, Switzerland, 2014. [CrossRef]
47. Glover, F. Future paths for integer programming and links to artificial intelligence. *Comput. Oper. Res.* **1986**, *13*, 533–549. [CrossRef]