An Improved Fault-tolerant Cultural-PSO with Probability for Multi-AGV Path Planning

Shiwei Lin[a,*], Ang Liu[b], Jianguo Wang[c], Xiaoying Kong[d]

[a]*Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, NSW, Australia*
[b]*Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, NSW, Australia*
[c]*Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, NSW, Australia*
[d]*School of IT and Engineering, Melbourne Institute of Technology, Sydney Campus, Sydney, NSW, Australia*

Abstract

This paper presents a hybrid evolutionary algorithm, cultural-particle swarm optimization (C-PSO), which is inspired by the cultural algorithm and the particle swarm optimization algorithm. It is aimed to balance the performance of exploration and exploitation and avoid trapping in the local optima. It introduces a probabilistic approach to update the inertia weight based on the improved metropolis rule. Generating the optimal path without collisions is challenging to ensure vehicles operate safely in real-time implementation. The contributions of C-PSO are to solve the path planning problem of multiple vehicles in modern industrial warehouses, achieving task allocation, fault tolerance and collision avoidance by a dual-layer framework. It was compared with the other algorithms, including PSO, PSO-GA, CA, HS, ABC, HPSGWO, TS and MA, by CEC benchmark functions and statistical tests to demonstrate its great performance with fewer iterations and runtime and the best solutions. It is validated through computational experiments, which involve 15 AGVs and 20 tasks for demonstration.

Keywords: optimization, multiple robots path planning, PSO, CA, fault tolerance

1. Introduction

AGVs have played a significant role in modern manufacturing systems due to safety and efficiency, providing low transportation and operation costs. Designing an AGV system needs to consider routing, scheduling, and layout (Xia et al., 2020). The transport control can be treated as a single objective or multi-optimization problem, considering the time required, total movement costs, vehicle travel times and expected waiting times etc. (Zhao et al., 2021). Multi-AGV systems have been more common in warehouses than the single-AGV counterparts, achieving efficiency and robustness of operations (Zając and Małopolski, 2021).

For the multi-AGV system, path planning is the primary consideration for robot navigation. Path planning methods can be classified as classical approaches, heuristic algorithms, bio-inspired algorithms and AI models (Lin et al., 2022). Most researchers utilize bio-inspired algorithms to provide optimization results to generate the near-optimal path because the algorithms provide fast convergence and effective solutions. The widely used bio-inspired algorithms include Particle Swarm Optimization (PSO) (Eberhart and Kennedy, 1995), Genetic Algorithm (GA) (Goldberg and Holland, 1988), and Simulated Annealing (SA) (Steinbrunn et al., 1997).

Ensuring that AGVs can operate safely and meet the tasks' requirements during online operation is challenging. Efficient multi-AGV path planning needs to manage the collisions and total consumed time and costs. Also, lacking consideration of fault tolerance has been the gap in path planning in the manufacturing system (Lin et al., 2022). This research aims to provide solutions for multi-AGV path planning, considering task allocation and fault tolerance for better practice with the developed Cultural-Particle Swarm Optimization (C-PSO). PSO is chosen in this research because of its simplicity and flexibility in solving optimization problems.

The proposed bio-inspired C-PSO for multi-AGV path planning balances the exploration and exploitation

*Corresponding author
Email addresses: Shiwei.Lin-1@student.uts.edu.au (Shiwei Lin), Ang.Liu@student.uts.edu.au (Ang Liu), Jianguo.Wang@uts.edu.au (Jianguo Wang), xkong@mit.edu.au (Xiaoying Kong)

for searching the solutions, and it avoids the algorithm trapped in local optima to avoid the drawbacks of meta-heuristic approaches. It implements probabilistic to adjust the search abilities. The dual-layer framework is also designed as the controller of the multi-AGV system to provide task allocation and fault tolerance. Its main contributions are as below:

- A novel hybrid bio-inspired technique is proposed based on improving the particle swarm optimization algorithm with the inspiration of the cultural algorithm and simulated annealing approach.
- The proposed algorithm implements a probabilistic approach to adjust the inertia weight to balance the local and global search abilities.
- It considers task allocation and fault tolerance during the path planning for the multi-AGV system, which achieves practical operation.

This paper describes a multi-AGV path planning based on the hybrid meta-heuristic algorithm to enhance search abilities and performance. It is organized as follows. Section 2 reviews the related work on multi-AGV path planning. Section 3 introduces the cultural algorithm, particle swarm optimization algorithm and metropolis rule. Section 4 proposes C-PSO with a dual-layer framework, and the experiment results are in Section 5. It is concluded in Section 6.

2. Related work

Numerous algorithms can be used to solve multi-robot path planning, including heuristic and bio-inspired algorithms and AI-based approaches. Deadlock rises during the circular wait, with no pre-emption, hold and wait, or mutual exclusion. To manage deadlock situations, deadlock detection and resolution and deadlock avoidance are required (Zając and Małopolski, 2021). Deadlock situations can be classified as pursuing collision, cross-collision, loop deadlock, and heading-on deadlock (Zhao et al., 2021). The deadlock resolution methods consist of zone control, time windows, and Petri-net according to traffic control strategies (Zhao et al., 2021).

The time window is one of the most popular deadlock resolution methods. The multi-AGV system in intelligent warehousing uses the time window for task assignment, and the A* algorithm finds the path in (Zhang et al., 2018a). A dynamic routing method is applied for multi-AGVs for supervisory control, and it evaluates paths by appropriate time windows and window overlapping tests (Smolic-Rocak et al., 2010). A conflict graph is introduced to solve the task assignment problem to deal with deadlocks (Sabattini et al., 2017). These approaches' limitations include not considering the entire task queue, fault detection or teamworking (Zhang et al., 2018a; Smolic-Rocak et al., 2010). A refined traffic model for the conflict graph is required in future work (Sabattini et al., 2017).

Correlation research has been converted into the travelling salesman, an NP-hard problem for combinational optimization, and the Tabu search algorithm is proposed based on neighbourhoods (Xing et al., 2020). Its shortcoming is assumptions of goods in the design process. The hybrid algorithm of Dijkstra and Floyd-warshall is presented with time windows in (Solichudin et al., 2020). Conflict resolution and path extension are executed in an alternate iterative conflict-based search algorithm, and the algorithm can be used on a topology map and a raster map with reduced time costs (Tao et al., 2021). But the approach does not consider task allocation. Grid blocking degree is proposed for multi-AGV path planning to minimize the handling completion time with a conflict-free path (Yu et al., 2021). The multi-AGV path planning and scheduling would consider the charging process in the next step.

An improved A* path planning algorithm based on coded marks and a grid-shaped network is proposed to guarantee the location and execution of multi-AGV (Lian and Xie, 2019). Probabilistic time constraints and queuing mechanisms are combined with an A* algorithm based on a dynamic random network in a warehouse multi-AGV system (Lian et al., 2020). A parallel algorithm is proposed by improving the A* algorithm with a penalty item, and it is composed of task assignment, path planning and navigation (Yu et al., 2022). The time-enhanced A* algorithm provides path planning in real time with temporal estimation and a supervision system (Matos et al., 2021). The adaptive iteration update and time consumption are the limitations of the algorithms when the conditions changes. The optimization of system design and intelligent algorithms are regarded as a further improvement.

Moreover, bio-inspired algorithms play important roles in solving path planning optimization problems. Three-exchange crossover heuristic operators are implemented in an improved GA with double-path constraints to minimize the distance for multi-AGV path planning (Han et al., 2017). Ant colony algorithm (ACO) is enhanced with prioritized planning for path coordination and optimization, considering the remaining battery charge and the fitness value (Zhang et al., 2018b). The GA is improved for multi-AGV scheduling planning

based on the operation of power evaluation and change of mutation operators, considering power consumption (Zhang and Li, 2020). Spinning drawing frames improve the GA for multi-AGV path planning and maneuvering scheduling decisions (Farooq et al., 2021). A PSO algorithm is used for AGV scheduling with a path time window (Sun et al., 2020).

A hybrid approach for multi-robot path planning in continuous environments is proposed with an enhanced GA algorithm and an innovative Artificial Potential Field (APF) (Nazarahari et al., 2019). It generates and improves the initial path by enhanced GA with five genetic operators in continuous space, capable of multi-objective optimization. Another multi-objective optimization algorithm is introduced with the PSO algorithm for multi-robot systems in unknown environments (Thabit and Mohades, 2019), which achieves smoothness, safety and shortness. For sharing knowledge, it considers the experiences and the current sensor information to decide.

The improved PSO algorithm is presented with two evolutionary operators to compute a safe path in a complex and known environment (Das and Jena, 2020). For each robot, it enhances the balance between diversification and intensification to minimize the path length. Another improved PSO is hybrid with a differentially perturbed velocity algorithm to minimize the arrival time and path length, which adjust the velocity with a differential operator (Das et al., 2016).

A self-adaptive PSO is improved to address the multi-robot path planning issue (Tang et al., 2020). The balance between local and global search abilities is treated as the stagnation problem, and the paper solves the problem with evolutionary game theory, updating the main control parameters. For multi-target dynamic path planning, the artificial bee colony (ABC) algorithm is modified for multi-robot navigation in an unknown dynamic environment (Faridi et al., 2018). The proposed evolutionary scheme smoothens the feasible path with evolutionary programming and a neighbourhood search planner.

Two-staged scheduling is introduced to handle collision for multi-AGV systems. In (Xu, 2017), the offline scheduling stage uses the GA for optimal path planning in the static environment. It solves opposite, node, and pursuit conflicts for AGV during the online scheduling stage. The other study of two-staged scheduling based on a GA is (Liu et al., 2015), and the GA is processed with constraints to get a stable path in the online stage. (Mu et al., 2020) presents a two-stage algorithm for multi-AGV path planning, the path of each AGV is generated by the A* algorithm with directional search and implements a time window to check conflicts and uses a conflict-based search algorithm to redesign the path.

For supporting deadlock and time-efficient collision resolution, the spare zone-based hierarchical motion coordination algorithm is introduced by adjusting the AGVs' path in a decentralized manner (Zhao et al., 2021). (Yang et al., 2022) proposes a multi-AGV scheduling system with extended Kalman filtering, global vision, and an oriented bounding box to estimate the heading angles and coordinates of AGVs. A cloud robotics architecture is presented in (Cardarelli et al., 2017) for supporting local path planning and a flexible and cooperative route assignment with knowledge extension. The PSO algorithm is improved to reduce congestion and provide efficiency for multi-AGV path optimization (Cao and Zhu, 2021).

However, there are some shortcomings in the previous papers. Some papers lack considerations of dynamic environment (Das and Jena, 2020; Zhang and Li, 2020; Sun et al., 2020), multi-robot cooperation (Das et al., 2016; Sun et al., 2020; Zhang et al., 2018b). Practical implementations are treated as future research, such as (Das et al., 2016; Han et al., 2017; Zhang et al., 2018b; Sun et al., 2020; Nazarahari et al., 2019; Tang et al., 2020; Liu et al., 2015), etc. Unknown area exploration is also the future work for multi-robot systems (Faridi et al., 2018). Limited adaptability is also the key challenge for multi-robot path planning in most papers. Real-time and algorithm performance should also be improved (Mu et al., 2020; Yang et al., 2022). Collision avoidance decision-making strategy should collaborate with path planning algorithms (Cao and Zhu, 2021).

Deep reinforcement learning can process high-dimensional environment data, such as images, and it has intelligent decision-making ability and powerful perception ability (Guo et al., 2020; Liao et al., 2020). A neural network structure and Dueling DDQN-PER has been implemented as AGV path planning for multi-modal sensing environmental information, such as GPS, cameras, and speed sensors (Guo et al., 2020). The multi-agent reinforcement learning technique is regarded as future work. For large-scale space, a reinforcement learning algorithm combines with a deep q-network in a complex environment (Liao et al., 2020). The cooperation of multiple AGVs is limited, and dynamic environments are challenging for optimal path planning. Multi-agent reinforcement learning is proposed as a deep deterministic policy gradient for anti-conflict multi-AGV path planning in (Hu et al., 2021), considering conflict situations as an integer programming model. Conflict prevention that considers the yard

exchange area is the next step.

Additionally, machine learning and bio-inspired combination have recently garnered interest in some research areas. A Convolutional Neural Network (CNN) is employed in a novel chaotic Firefly algorithm to improve the exploration performance (Bacanin et al., 2021b). It uses a chaotic local search strategy and explicit exploration mechanism on global optimization, strengthening CNN accuracy. The future study will combine it with other machine learning approaches to solve NP-hard problems. Feature selection is to select the relevant features for pattern classification in machine learning, and Malakar et al. (2020) develops a hierarchical feature selection model based on the GA algorithm for global and local features. The local search technique would be integrated with the global optimization strategy for improvement.

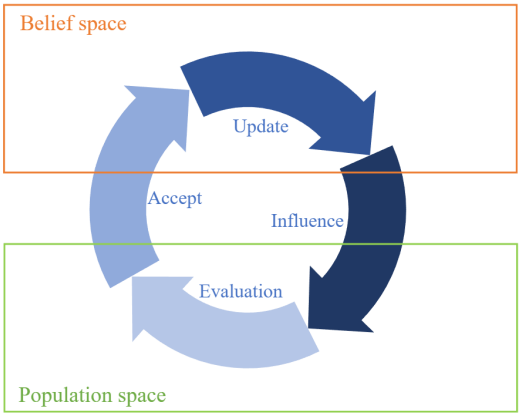
Another nature-inspired combination algorithm is proposed with Artificial Neural Networks and Artificial Bee Colony (ABC) algorithm to overcome the drawbacks of slow convergence and local optima, and it optimizes the hidden units and connection weights of the neural network (Bacanin et al., 2021a). The drawbacks of the algorithm are an additional effort of setting control parameters and more calculations in each iteration. Tian et al. (2021) enhances PSO with a jumping mechanism for accurate and real-time implementation of multi-robot path planning. It uses an AI algorithm to plan the path with new learning samples, and it locates obstacles and robots with wireless sensor networks. The shortcomings include its inability to deal with many robots and slow convergence.

The literature shows that graph-based, bio-inspired, and AI-based search approaches are widely used. A*, GA, PSO and deep learning are the popular algorithms in the cited literature. Some papers use a two-stage framework for a multi-AGV system for scheduling and path planning. However, from (Lin et al., 2022) and the cited papers, fault tolerance is barely considered during multi-AGV real-time path planning. This paper proposes a fault-tolerant multi-AGV path planning algorithm based on the hybridization of cultural algorithm and particle swarm optimization.

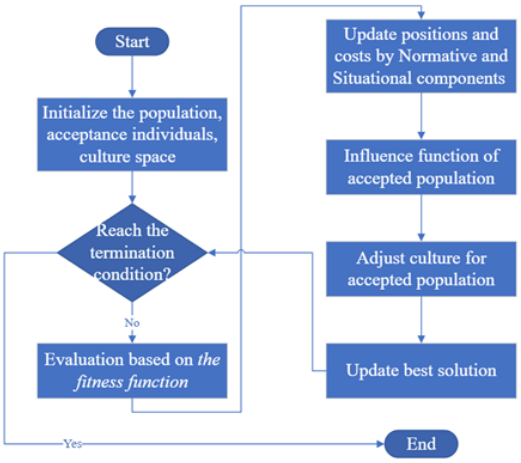
3. Preliminary knowledge

3.1. Cultural Algorithm (CA)

A cultural algorithm (CA) is proposed by Reynolds and ChanJin (1996) as an evolutionary algorithm by gaining solutions through normative and situational knowledge. The principle of CA is illustrated in Figure



(a) The principle of CA



(b) The flowchart of CA

Figure 1: The principle of CA

1. It consists of belief and population space and uses evolutionary knowledge to determine the solutions. CA uses the influence function to get the new population as Eq. (1) (Reynolds and ChanJin, 1996).

$$x_{i,j}^{t+1} = \begin{cases} x_{i,j}^{t} + |(u_j^t - l_j^t) \cdot rand|, \text{if } x_{i,j}^{t} < l_j^t \\ x_{i,j}^{t} - |(u_j^t - l_j^t) \cdot rand|, \text{if } x_{i,j}^{t} > l_j^t \end{cases} \quad (1)$$

Where $x_{i,j}^{t}$ and $x_{i,j}^{t+1}$ are the positions, t and $t+1$ are the iteration numbers, i indicates the individual, and j means the jth value of i. u_j^t is the maximum position limit of the jth value of individual i, while l_j^t is the minimum limit. $rand$ is a random number between 0 and 1.

The accept function usually selects n individuals with better solutions from the population. The update function implements Eqs. (2) – (5) to update the positions and fitness values (Reynolds and ChanJin, 1996).

$$l_j^{t+1} = \begin{cases} x_{i,j}^t, \text{if } x_{i,j}^t \le l_j^t \text{ or } f_{obj}(x_i^t) < L_j^t \\ l_j^t, \text{otherwise} \end{cases} \quad (2)$$

$$L_j^{t+1} = \begin{cases} f_{obj}(x_i^t), \text{if } x_{i,j}^t \le l_j^t \text{ or } f_{obj}(x_i^t) < L_j^t \\ L_j^t, \text{otherwise} \end{cases} \quad (3)$$

Where f_{obj} is the objective function to evaluate the fitness value. L_j^{t+1} presents the fitness value of the lower bound.

$$u_j^{t+1} = \begin{cases} x_{i',j}^t, \text{if } x_{i',j}^t \ge u_j^t \text{ or } f_{obj}(x_{i'}^t) < U_j^t \\ u_j^t, \text{otherwise} \end{cases} \quad (4)$$

$$U_j^{t+1} = \begin{cases} f_{obj}(x_{i'}^t), \text{if } x_{i',j}^t \ge u_j^t \text{ or } f_{obj}(x_{i'}^t) < U_j^t \\ L_j^t, \text{otherwise} \end{cases} \quad (5)$$

Where individual i' impacts the upper bound. $x_{i',j}^t$ indicates the upper position. U_j^{t+1} is the maximum fitness value in the $t+1$ iteration for *jth* value.

CA is inspired by bicultural evolution theory, which implements horizontal and vertical information for individuals within the population space (Jalili, 2022). It inherits the information from parent solutions and learns from accumulated cultural information from previous generations. But its framework is complex and it requires high computational cost. It is time-consuming during operation.

3.2. Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) is widely implemented in optimization (Eberhart and Kennedy, 1995). The particles explore the searching space to get the solutions for the optimization problems. The equations of velocity and position of the particles are updated as Eqs. (6) - (7) (Eberhart and Kennedy, 1995). The personal best of a particle and the global best is updated as Eqs. (8) - (9) (Eberhart and Kennedy, 1995).

$$v_i^{t+1} = \omega v_i^t + c_1 r_1 (pbest_i^t - x_i^t) + c_2 r_2 (gbest^t - x_i^t) \quad (6)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (7)$$

Where ω is the inertia weight, and v_i^{t+1} denotes the velocity of the particle in iteration $t+1$. c_1 and c_2 are cognitive and social parameters. r_1 and r_2 are random number. $pbest_i^t$ is the personal best for particle i in iteration t. $gbest$ stands for the global best value. x_i^{t+1} represents the position of particle i in iteration $t+1$.

$$pbest_i^{t+1} = \begin{cases} pbest_i^t, \text{if } f_{obj}(x_i^{t+1}) \ge f_{obj}(pbest_i^t) \\ x_i^{t+1}, \text{otherwise} \end{cases} \quad (8)$$

$$f_{obj}(gbest^t) = \min(f_{obj}(x_1^t)), f_{obj}(x_2^t)), ..., f_{obj}(x_i^t))) \quad (9)$$

PSO is one of the most commonly used and powerful algorithms for optimization problems in several domains with easy implementation. It has fast convergence when generating solutions. But it has the drawbacks of exploitation and risky random positions (Senel et al., 2019). It may suffer from the local optima if the start position is far from the global optima.

3.3. Metropolis rule

The Metropolis rule is proposed in Simulated Annealing (SA), and the probabilistic of the Metropolis rule is as Eq. (10) (Steinbrunn et al., 1997). SA reduces "temperature" to get a lower energy state for reaching the solutions to the optimization problem.

$$\rho = \begin{cases} 1, \text{if } f_{obj}(x_i^{t+1}) \le f_{obj}(x_i^t) \\ e^{\frac{f_{obj}(x_i^{t+1}) - f_{obj}(x_i^t)}{T}}, \text{otherwise} \end{cases} \quad (10)$$

Where $f_{obj}(x_i^{t+1})$ is the fitness value of the new state x_i in the iteration $t+1$, and T is the temperature in the SA.

4. Cultural-PSO (C-PSO)

4.1. Overview

The proposed C-PSO consists of centralized and decentralized layers, as shown in Figure 2. The centralized layer processes map generation and task allocation for the near-optimal path. Task allocation evaluates the path cost and task cost for each AGV. Assign the available AGV with minimal cost, and set it as unavailable in the current stage. The decentralized layer achieves information sharing, path re-planning, collision avoidance and fault tolerance within a time window. For the decentralized layer, AGVs follow the defined path and communicate with each other for positions and statuses in real-time. If the collision occurs or there is an AGV turned down, path re-planning and fault tolerance are achieved.

4.2. C-PSO

Inspired by CA and the proposed PSO-SA (Lin et al., 2023), Cultural-PSO (C-PSO) is proposed to overcome the drawbacks of PSO and CA. It combines the characteristics of CA to update the inertia weight for PSO to

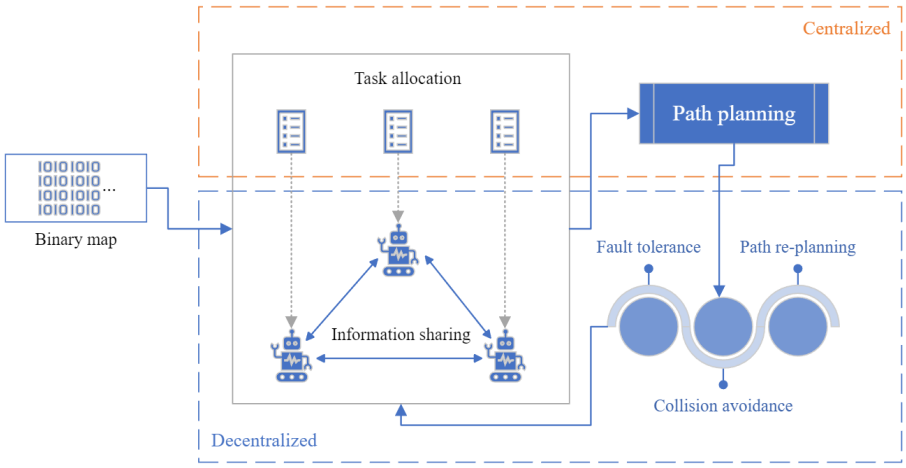


Figure 2: Dual-layer of the C-PSO algorithm

provide fast convergence speed and a great balance between exploration and exploitation. The inertia weight has fluence for the global and local search abilities. When inertia weight is larger, PSO emphasizes diversification, enabling global space exploration. While it is smaller, PSO focuses on intensification, performing local searching. C-PSO uses the probability from PSO-SA to update the inertia weight and combines PSO and CA as Algorithm 1.

The probability is calculated by Eqs. (11) - (12) (Lin et al., 2023), and compared with a generated random number. If the average cost of the personal best solutions is higher than the current particle cost, and the probability is larger than or equal to the generated random number, the inertia weight changes as Eq. (13). If the average cost exceeds the current particle cost, the swarm should search for the global space to get a more accurate solution and avoid trapping on the local optima.

$$\delta = \frac{f_{obj}(i_t) - f_{best}(i_t)}{f_{best}(i_t)} \tag{11}$$

$$\rho = e^{-\frac{\delta}{T}} \tag{12}$$

Where ρ is the calculated probability, and δ is calculated by Eq. (11). $f_{obj}(i_t)$ is the cost of the particle i in the current iteration t, and $f_{best}(i_t)$ is the personal best value of the particle i. T is the temperature.

$$\omega = \begin{cases} \omega + |(f_{obj}(i_t) - f_{best}(i_t))|, \text{if } f_{obj}(i_t) < f_{obj}(avg_t) \\ \omega_{min}, \text{if } f_{obj}(i_t) \geq f_{obj}(avg_t) \end{cases} \tag{13}$$

Where ω is the inertia weight for PSO. ω is the current value for the inertia weight, and ω_{min} is the minimum

Algorithm 1: C-PSO algorithm

```
Initialization
for iteration = 1 : iteration_max do
    for particle = 1 : particle_max do
        Update Velocity
        Update Position
        Evaluation
        if pcost_i^t < pbest_i^t then
            Update Personal Best
            if pbest_i^t < gbest^t then
                Update Global Best
            end
        else
            delta ← (pcost_i^t − pbest_i^t)/pbest_i^t
            p ← exp(−delta/T)
            if p > rand then
                if pcost_i^t < avgcost then
                    w ← w + abs(pcost_i^t − pbest_i^t)
                else
                    w ← wmin
                end
            end
        end
    end
    Update Global Best Cost
    w ← w ∗ wdamp
    T ← alpha ∗ T
end
```

value of the inertia weight. $f_{obj}(avg_t)$ is the average cost of all personal best in iteration t. i_t is the particle i. When ω is larger, the global search ability is enhanced.

4.3. Path planning

The path planning problem of the multi-AGV system consists of multiple starts and destinations for the involved AGVs and the two-dimensional environment. A binary map indicates the two-dimensional environment: zero is allowed to move and is marked by white, while one stands for the wall or obstacle marked by black. Figure 3 indicates the warehouse scenario, and each AGV has one start and target in each task. The generated path cannot be overlapped with the obstacle or have collisions with other AGVs during operation. The objective function of path planning is to minimize the path length and avoid collision as Eqs. (14) – (17). The best solution is treated as the globally near-optimal path with minimal costs. If a collision occurs, the result is not accepted.

$$f^t_{length} = \sum_{i=1}^{n} \sqrt{(x^t_i - x^t_{i+1})^2 + (y^t_i - y^t_{i+1})^2} \tag{14}$$

Where each position r^t_i is represented by (x^t_i, y^t_i). The AGV is indicated by i, and the iteration is indicated by t. The total number of AGVs is n. The current AGV is i, and the next AGV is $i + 1$.

The violation cost of collision is evaluated by Eqs. (15) and (17).

$$d^t_i = \sqrt{(x^t_i - x_c)^2 + (y^t_i - y_c)^2} \tag{15}$$

$$c^t_i = \sum_{c=1}^{z} r_c - d^t_i, \quad if \quad d^t_i < r_c \tag{16}$$

$$f^t_{collision} = \sum_{i=1}^{n} c^t_i \tag{17}$$

Where (x_c, y_c) is the central coordination of an obstacle, and r_c is the radius of the obstacle. d^t_i is the distance from the AGV to the central coordination of the obstacle. The total number of obstacles is z, and c^t_i is the collision cost for the AGV i, which sums up the total collision cost of the current AGV. Then summing all collision costs of all AGVs as the collision cost.

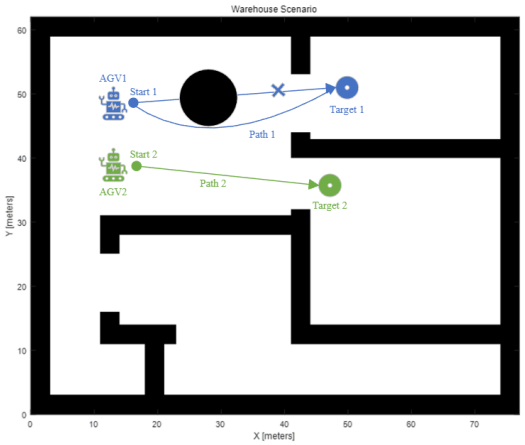


Figure 3: The scenario of problem formulation

4.4. Task allocation

Task allocation is essential for the multiple robot system, as demonstrated in Figure 4. It considers the task priority, the objective value of the path and the time of completing the previous task as (18) – (19).

$$f_{path} = w_1 * f_{length} + w_2 * f_{collision} \tag{18}$$

Where f_{path} is to minimise the path length and avoid collisions. The definitions of f_{length} and $f_{collision}$ refer to Section 4.3. The weight factors are w_1 and w_2 of each cost, and their sum is 1 and set as 0.5 and 0.5, respectively, because the objectives of path planning regarding minimizing the path length and avoiding collisions play the same important roles.

$$f_{task} = w_3 * f_{path} + w_4 * f_{taskPriority} + w_5 * f_{timeOfPreviousTask} \tag{19}$$

Where the task objective value is $f_{taskCost}$, it calculates the path objective value f_{path} and time of the previous task $f_{timeOfPreviousTask}$, considering the task priority $f_{taskPriority}$. The time of the previous task includes the time of following the previous path and packing the goods. The sum of w_3, w_4 and w_5 is 1. The objective value of the path is the most significant factor that affects task completion, so w_3 is set as 0.8. The task priority and the time of the previous task are considered less important factors in the objective function, so w_4 and w_5 are set as 0.1 and 0.1.

For assigning tasks for AGVs, the proposed approach has a routing table to record the status of AGVs as shown in Table 1. The status of occupied and available is indicated by 0 or 1. For the occupied status, "0"

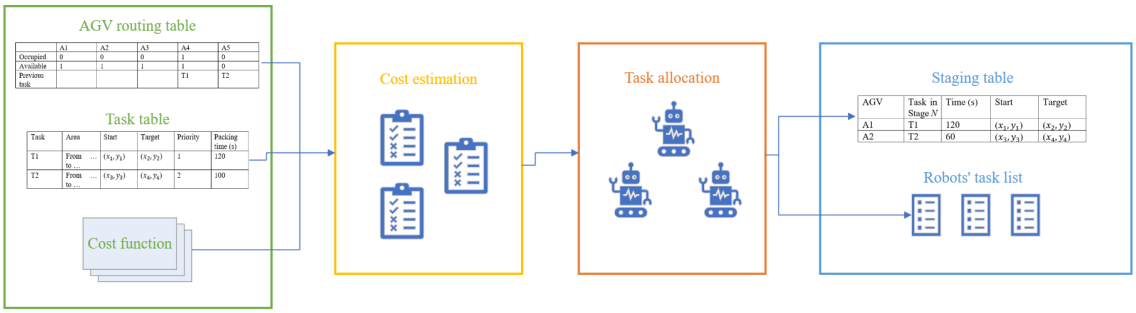


Figure 4: The flow of task allocation

Table 1: A example of AGV routing table

	A1	A2	A3	...	An
Occupied	0	0	0	...	1
Available	1	1	1	...	1
Previous task		T1		...	T2

Table 2: A example of Task table

Task	Area	Start	Target	Priority	Packing time (s)
T1	From ... to ...	(x_1, y_1)	(x_2, y_2)	1	$PackingTime_1$
T2	From ... to ...	(x_3, y_3)	(x_4, y_4)	2	$PackingTime_2$
...	From ... to ...	...	...	...	...
Tn	From ... to ...	(x_n, y_n)	(x_n, y_n)	2	$PackingTime_n$

means it can be assigned tasks; and "1" stands for operating tasks. For the available status, "0" means the robot shuts down, and "1" means the robot can operate tasks.

Table 2 is the task table, and it lists the task number and priority, starts and target locations, and the packing time for each task. The task table is for the centralized task allocation and the individual robot's record. The task is urgent if the priority is low.

The steps of task allocation include the following:

1. Calculate the task cost for available AGVs
2. Start assigning the tasks based on the urgency
3. Assign the task to the AGV with minimal cost based on the task costs
4. Set the robot as occupied in the current stage and add the task to the robot's task list. If a robot is occupied, the cost of the next task starts by calculating the time of the current task for the robot
5. Start the next stage of the task allocation for the remaining tasks until all tasks are assigned

Table 3 indicates the staging table. It records the task for each robot in the current stage. Time calculates the time to follow the path and the packing time for the current task. Start and target record the locations of the source and destination.

Table 3: A example of Staging table

AGV	Task in Stage N	Time (s)	Start	Target
A1	T1	$Time_1$	(x_1, y_1)	(x_2, y_2)
A2	T2	$Time_2$	(x_3, y_3)	(x_4, y_4)
...	...	...	...	...
An	Tn	$Time_n$	(x_n, y_n)	(x_{n+1}, y_{n+1})

4.5. Collision avoidance

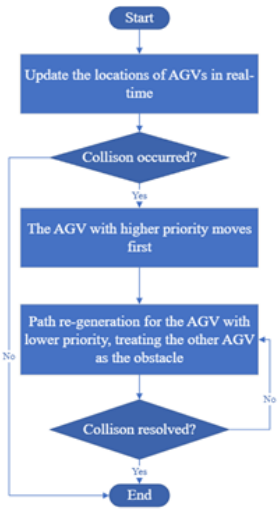
Collision avoidance should be achieved during the path planning and the online stage to ensure the AGVs operate safely. The flow charts for avoiding the other AGVs and the dynamic obstacles are displayed in Figure 5.

If an AGV occurs in a collision or deadlock with another AGV, AGVs get the priority information to determine which AGV is required to change the generated path. The priority of an AGV is higher when it is closer to or arrives at the destination, operates tasks or is occupied, or has a more urgent task. For the AGV with lower priority, the path is re-generated by the C-PSO and treated as the other AGV as the obstacle during planning until the collision or deadlock is resolved.

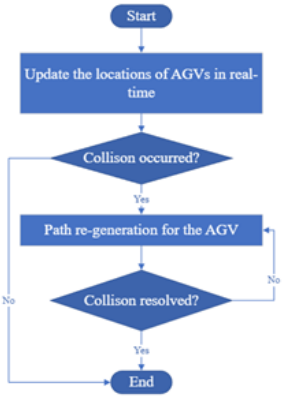
When an AGV detects dynamic obstacles by the equipped sensors, the system determines whether it would affect the AGV's operation. If the collision is highly possible, the AGV must re-generate the path by the C-PSO. It treats the dynamic obstacles as static obstacles in each iteration during operation until the resolution is accomplished.

4.6. Fault tolerance

Fault tolerance plays an important role in the real-time implementation of the multi-AGV system. The proposed approach has the benefit of handling the fault

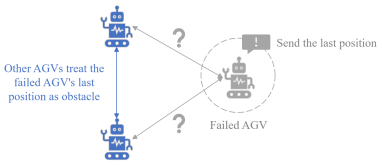


(a) Collision resolution for AGVs

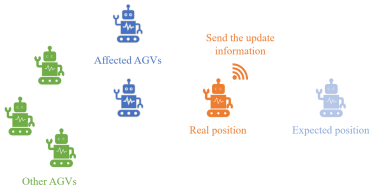


(b) Collision resolution for dynamic obstacles

Figure 5: Flowcharts of collision resolution



(a) When one AGV fails



(b) When one AGV is not operated as expected

Figure 6: The scenario of fault tolerance

independently without affecting the other AGVs. Also, it is achieved in the decentralized layer, which would not lead to changes in the centralized layer. The AGVs update the information in real-time to ensure their operation performance; when the signal of an AGV is lost, or the AGV is slower/faster than expected operation, the system is notified. The expected operations of AGVs are to follow the defined path safely and perform the assigned tasks with regular communication with robots.

The situations and solutions for fault tolerance are as Figure 6 and follows.

1. When one AGV fails
   - The failed AGV sent the last information to the system and shut down the failed AGV
   - Other AGVs treat the failed AGV's last position as the static obstacle
   - If an AGV is available, send the AGV to finish its tasks
   - If there is no available AGV after the closest AGV finishes its task, add the tasks to its task table
2. When one AGV is not operated as expected
   - Update the locations in real-time
   - Path re-planning for the affected AGV, such as AGV in the neighbourhood

5. Computational experiments

5.1. Comparative analysis

5.1.1. Performance measurements

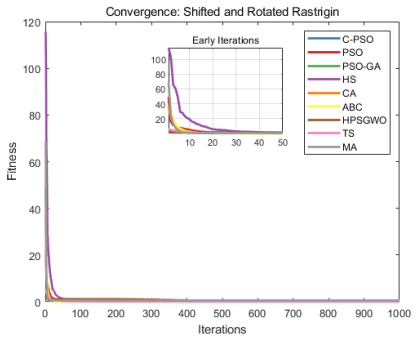
The proposed C-PSO has been compared with other bio-inspired algorithms, including PSO, PSO-GA

(Garg, 2016), CA, Harmony Search (HS) (Lee and Geem, 2005), Artificial Bee Colony (ABC) (Karaboga and Basturk, 2007), a hybrid PSO-GWO (HPSGWO) (Senel et al., 2019), Transit Search Optimization (TS) (Mirrashid and Naderpour, 2022), and Mayfly Algorithm (MA) (Zervoudakis and Tsafarakis, 2020). The benchmark functions are from CEC 2019, with rotated and shifted, which are listed in Table 4. The parameters of the compared methods are set as Table 5. Each algorithm runs 20 times.

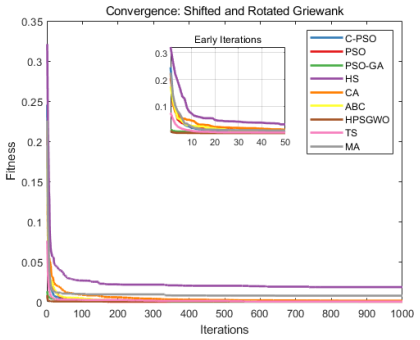
Table 6 compares the fitness value for each benchmark function. It lists the mean fitness value computed by the algorithms and the correct digits that match the best solutions. In this case, the maximum number of correct digits is 5; if there is no correct digit, the solutions would not be considered to compare runtime and iteration. The consumed iteration and runtime are recorded in Table 7 when getting the best solution. The best value is highlighted in bold.

C-PSO usually can generate the best solution with fewer iterations and runtime with the best fitness value. Figure 7 displays the convergence curve of each algorithm in each benchmark function. C-PSO gets the best fitness value for most optimization function f_1, f_2, and f_4, and the difference between the PSO solution of f_3 is insignificant. C-PSO, PSO and PSO-GA have the best performance for correct digits at 4.5, but C-PSO has the least runtime and iteration among these methods. Even HS and MA cost less iteration or runtime in some functions, but they cannot get the best fitness value; the accuracy is quite low, as they only achieve 2 and 2.75 in mean correct digits when the maximum number of iterations is 1000. But MA has the best performance in f_4.

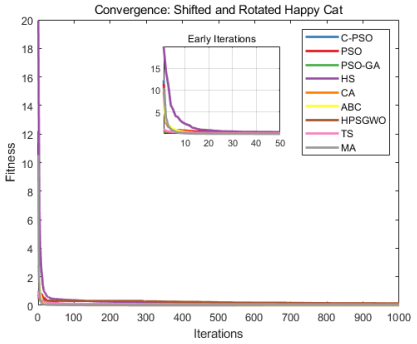
For statistical tests, Friedman ranks are assigned based on the performance of the algorithms (Derrac et al., 2011). Table 8 indicates the ranks. The performance ranks consider the best solutions, runtimes and iterations, comparing the solution optimality first, then runtime and iterations. C-PSO reaches the second least mean iteration and the least runtime for generating the results, but it achieves the best performance with consideration of solution optimality. It is a steady algorithm to perform the searching ability in most optimization problems and is useful for online path planning due to its computational speed. C-PSO performs best according to the Friedman ranks with 1.375. From the Friedman test, the p-value is 0.0044. Because the p-value is less than 0.05, a Post-Hoc Analysis is required. Table 9 provides the test results with the binary indicators. If the indicator is 1, the pair of algorithms is statistically significant, while 0 is not.



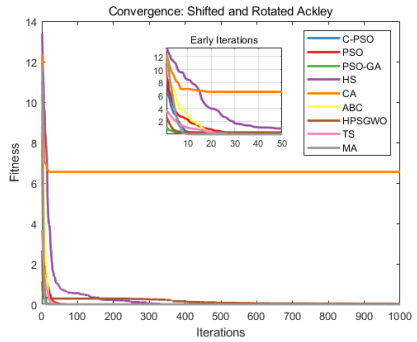
(a) $F_1(x)$: Shifted and Rotated Rastrigin's Function



(b) $F_2(x)$: Shifted and Rotated Griewank's Function



(c) $F_3(x)$: Shifted and Rotated Happy Cat Function



(d) $F_4(x)$: Shifted and Rotated Ackley Function

Figure 7: Convergence curve of each algorithm

Table 4: Benchmark functions

Number	Function	Equation	Search range
$F_1(x)$	Shifted and Rotated Rastrigin's Function	$f_1(x) = \sum_{i=1}^{n} \left(x_i^2 - 10\cos(2\pi x_i) + 10\right)$, $F_1(x) = f_1(M(x - o_1)) + F_1^*$	[-100,100]
$F_2(x)$	Shifted and Rotated Griewank's Function	$f_2(x) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$, $F_2(x) = f_2(M(x - o_2)) + F_2^*$	[-100,100]
$F_3(x)$	Shifted and Rotated Happy Cat Function	$f_3(x) = \left\lvert\sum_{i=1}^{n} x_i^2 - n\right\rvert^{\frac{1}{4}} + \left(0.5\sum_{i=1}^{n} x_i^2 + \sum_{i=1}^{n} x_i\right)/n + 0.5$, $F_3(x) = f_3(M(x - o_3)) + F_3^*$	[-100,100]
$F_4(x)$	Shifted and Rotated Ackley Function	$f_4(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$, $F_4(x) = f_4(M(x - o_4)) + F_4^*$	[-100,100]

Table 5: Parameter settings

Algorithm	Parameter
C-PSO	$Iteration = 1000,\ size_{population} = 100,\ T_0 = 0.025,\ alpha = 0.99,\ w = 1,\ wdamp = 0.99,\ w_{max} = 0.9,\ w_{min} = 0.4,\ c_1 = 1.5,\ c_2 = 1.5$
PSO	$Iteration = 1000,\ size_{population} = 100,\ w = 0.8,\ c_1 = 1.5,\ c_2 = 1.5$
PSO-GA	$Iteration_{PSO} = 1000,\ Iteration_{GA} = 500,\ w = 0.8,\ c_1 = 1.5,\ c_2 = 1.5,\ rate_{crossover} = 0.8,\ rate_{mutation} = 0.2$
CA	$Iteration = 1000,\ size_{population} = 100,\ rate_{accept} = 0.35$
HS	$Iteration = 1000,\ size_{harmony\ memory} = 100,\ size_{new} = 20,\ rate_{consideration} = 0.9,\ rate_{adjustment} = 0.1$
ABC	$Iteration = 1000,\ size_{population} = 100$
HPSGWO	$Iteration = 1000,\ size_{population} = 100,\ prob = 0.3,\ w = 0.8,\ c_1 = 1.5,\ c_2 = 1.5$
TS	$Iteration = 1000,\ size_{population} = 100,\ ns = 5,\ SN = 20$
MA	$Iteration = 1000,\ size_{population} = 100,\ size_{female} = 50,\ size_{male} = 50,\ w = 0.8,\ rate_{mutation} = 0.01,\ size_{offspring} = 50$

Table 6: Mean fitness value of each algorithm

Algorithms		$F_1(x)$	$F_2(x)$	$F_3(x)$	$F_4(x)$	Mean
C-PSO	Mean	**0.0000**	**0.0000**	0.0052	**0.0000**	
	Correct digits	**5**	**5**	**3**	**5**	**4.5**
PSO	Mean	**0.0000**	**0.0000**	**0.0041**	**0.0000**	
	Correct digits	**5**	**5**	**3**	**5**	**4.5**
PSO-GA	Mean	**0.0000**	**0.0000**	0.0063	**0.0000**	
	Correct digits	**5**	**5**	**3**	**5**	**4.5**
HS	Mean	0.3482	0.0182	0.1276	0.0002	
	Correct digits	1	2	1	4	2
CA	Mean	0.0034	0.0013	0.0134	6.5473	
	Correct digits	3	3	2	0	2
ABC	Mean	**0.0000**	0.0002	0.0202	**0.0000**	
	Correct digits	**5**	4	2	**5**	4
HPSGWO	Mean	0.0497	0.0001	0.1395	0.0302	
	Correct digits	2	4	1	2	2.25
TS	Mean	**0.0000**	0.0004	0.0643	**0.0000**	
	Correct digits	**5**	4	2	**5**	4
MA	Mean	0.5472	0.0076	0.0237	**0.0000**	
	Correct digits	1	3	2	**5**	2.75

Table 7: Mean iteration and runtime of each algorithm

Algorithms		$F_1(x)$	$F_2(x)$	$F_3(x)$	$F_4(x)$	Mean
C-PSO	Iteration	140.65	190.9	**581**	340.15	313.175
	Runtime	0.3502	**0.4579**	1.4533	0.8578	**0.7798**
PSO	Iteration	283.55	323.85	829.45	545.1	495.4875
	Runtime	0.5899	0.6971	1.8964	1.1700	1.08835
PSO-GA	Iteration	260	285.45	840.55	498.25	471.0625
	Runtime	4.2646	4.6532	14.3131	8.1364	7.841825
HS	Iteration	917.65	912.9	794.05	876.25	875.2125
	Runtime	0.7475	0.7560	**0.7326**	0.8365	0.76815
CA	Iteration	367.3	550.8	999.95	-	639.35
	Runtime	0.5743	0.9524	1.7415	-	1.0894
ABC	Iteration	552.55	493.85	587.65	591.4	556.3625
	Runtime	1.8947	1.7734	2.1286	2.1093	1.9765
HPSGWO	Iteration	734.4	623	730.75	714.6	700.6875
	Runtime	1.3643	1.2730	1.8791	1.9373	1.613425
TS	Iteration	254.75	315.55	632.4	619.2	455.475
	Runtime	1.0000	1.2364	2.5687	2.4135	1.80465
MA	Iteration	**27.4**	**162.5**	997.25	**40.65**	**306.95**
	Runtime	**0.1211**	0.7185	4.8653	**0.1762**	1.470275

Table 8: Friedman ranks

Algorithms	$F_1(x)$	$F_2(x)$	$F_3(x)$	$F_4(x)$	Mean
CPSO	**1**	**1**	**1.5**	2	**1.375**
PSO	2	2	**1.5**	3	2.125
PSO-GA	3	3	3	6	3.75
HS	8.5	9	8	7	8.125
CA	6	7.5	4.5	9	6.75
ABC	5	6	4.5	4	4.875
HPSGWO	7	4.5	9	8	7.125
TS	4	4.5	6	5	4.875
MA	8.5	7.5	7	**1**	6

Table 9: Post-hoc analysis

	C-PSO	PSO	PSO-GA	HS	CA	ABC	HPSGWO	TS	MA
CPSO	0	0	0	1	0	0	1	0	0
PSO	0	0	0	1	0	0	0	0	0
PSO-GA	0	0	0	0	0	0	0	0	0
HS	1	1	0	0	0	0	0	0	0
CA	0	0	0	0	0	0	0	0	0
ABC	0	0	0	0	0	0	0	0	0
HPSGWO	1	0	0	0	0	0	0	0	0
TS	0	0	0	0	0	0	0	0	0
MA	0	0	0	0	0	0	0	0	0

Table 10: Start and target locations

AGV	Path	Start	Target
1	1	(10,40)	(60,50)
2	2	(15,50)	(60,58)
3	3	(10,35)	(30,5)
4	4	(30,50)	(5,10)

Table 11: Performance of path planning

		Time	Cost	Iteration
Path 1	CPSO	**0.1057**	**25.4951**	**1**
	PSO	0.1081	**25.4951**	**1**
Path 2	CPSO	**7.5919**	**22.9765**	**74**
	PSO	47.9826	22.9776	466
Path 3	CPSO	**19.7208**	**19.2997**	**190**
	PSO	48.1500	19.3000	452
Path 4	CPSO	**12.3759**	24.4785	**125**
	PSO	36.4600	**24.4767**	351

5.1.2. Path planning

A multi-AGV path planning scenario is in Table 10, listing the start and target locations. From the previous Section 5.1.1, C-PSO and PSO are the top algorithms for optimization problems, so C-PSO and PSO are used to compare the path planning performance. The parameters are set as Table 5, but the population size is 150 for the path planning problem. The number of maximum iterations is 500. Table 11 compares the solutions, costs, iterations and time. C-PSO used 30% mean runtime and 31% mean iterations to get the best solution than PSO when generating four paths. Figure 8 demonstrates the paths generated by C-PSO and PSO.

5.2. Experiment of path planning

5.2.1. Experiment settings

The AGVs are initially located in the loading area. The pillars and walls are annotated with black, and the AGVs are marked with different colours. Table 12 lists the tasks for the multi-robot system.The start and target locations are randomly distributed between Areas G, A, B, C, and D, where Area G is the loading zone. The task priorities are allocated from 1 to 5, and packing time is randomly from 100 to 150 seconds. The locations of the AGVs are shown in Table 13. The computation experiment involves 20 tasks and 15 AGVs.

The assumptions of the simulation are listed below:

- AGVs have the map, and the speed is 0.3 m/s.

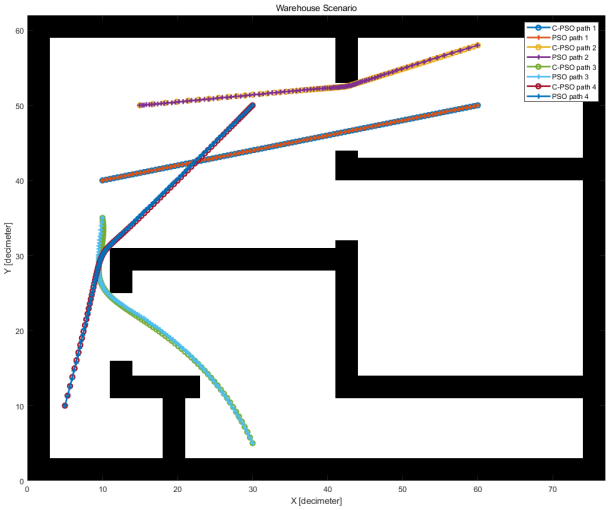


Figure 8: Simulation of path planning

Table 12: Task list in the simulation

Task	Start	Target	Start location	Target location	Priority	Packing time (s)
T1	G	C	Loading zone	(21,20)	4	133
T2	G	A	Loading zone	(55,50)	1	111
T3	B	G	(49,35)	Loading zone	5	121
T4	G	D	Loading zone	(5,9)	5	122
T5	G	A	Loading zone	(58,49)	3	130
T6	A	C	(52,50)	(13,19)	4	113
T7	G	A	Loading zone	(60,48)	2	121
T8	B	G	(52,37)	Loading zone	2	140
T9	G	D	Loading zone	(10,8)	4	124
T10	B	A	(42,36)	(62,48)	3	146
T11	B	A	(48,39)	(58,52)	3	131
T12	B	G	(49,39)	Loading zone	4	109
T13	C	A	(13,23)	(48,48)	2	150
T14	C	G	(20,20)	Loading zone	2	131
T15	D	G	(5,7)	Loading zone	5	140
T16	B	A	(54,35)	(45,49)	3	126
T17	D	C	(9,9)	(25,20)	4	120
T18	D	G	(6,7)	Loading zone	2	147
T19	D	B	(7,9)	(58,37)	4	129
T20	A	B	(48,50)	(45,35)	2	136

Table 13: AGVs' initial location

AGV	Area	Position
AGV1	Loading zone (G)	(19,45)
AGV2	Loading zone (G)	(15,43)
AGV3	Loading zone (G)	(19,41)
AGV4	Loading zone (G)	(12,42)
AGV5	Loading zone (G)	(15,49)
AGV6	Loading zone (G)	(16,47)
AGV7	Loading zone (G)	(10,43)
AGV8	Loading zone (G)	(16,44)
AGV9	Loading zone (G)	(13,47)
AGV10	Loading zone (G)	(10,45)
AGV11	Loading zone (G)	(14,50)
AGV12	Loading zone (G)	(11,50)
AGV13	Loading zone (G)	(11,42)
AGV14	Loading zone (G)	(12,40)
AGV15	Loading zone (G)	(11,44)

- AGVs have onboard sensors to detect other AGVs and obstacles
- AGVs can communicate with the other AGVs for position and direction information
- The proposed algorithm is implemented in the AGVs' board
- AGVs can transform the good automatically
- For testing the fault tolerance performance, AGV 1 is broken; AGV 2 will shut down after 5s; AGV 4 will move slower with 1s' delay after 10s.

5.2.2. Results

For assigning the tasks, the system arranges the most urgent task in the first based on the task priority; if the task priority is the same, the system compares the task costs to determine the order of operation. Tasks were assigned based on task priority and cost considerations, adding to the staging table. Task 2 has the highest priority so it would be the first performed task. Table 14 records the task costs of each robot as the Task table. The system searched the Task table to determine the employed robot based on the minimal costs. AGV 3 is assigned to Task 2.

If the task priority is the same, the costs would be the first consideration factor. Tasks 7, 8, 13, 14, 18 and 20 have the second priority. Because AGV 3 completed Task 2, it was on the target location of Task 2; then, the Task table was updated. In the following stages, once every robot completes a task, the Task table will update based on the new location and completion time. Task 20 has the minimal cost among six tasks, so it is assigned to AGV 3 in Stage 2 planning. Then Task 7 to AGV 6, Task 14 to AGV 14, Task 18 to AGV 13, Task 13 to AGV 4 and Task 8 to AGV 8 in Stage 1 planning.

The third urgent tasks are Tasks 5, 10, 11 and 16. AGV 5 for Task 5 has minimal cost among these tasks, so it is assigned in AGV 5 Stage 1 planning. Then Task 10 is considered, and AGV 2 is assigned, but it was broken after 5s, so a new AGV must be allocated to resume its task, and AGV 2's last location is treated as the obstacle. AGV 5 is occupied, so AGV 9 performs the fault tolerance. Because AGVs 5 and 9 are occupied, they cannot be used for Tasks 11 and 16. Task 16 has a lower cost than 11 for AGV 11, so AGV 11 is assigned to Task 16, and AGV 15 is for Task 11.

Tasks 1, 6, 9, 12, 17, and 19 have the fourth priority. Task 1 with AGV 7 is minimal in these tasks, so AGV 7 is allocated. Therefore, Tasks 9 and 17 must choose AGVs 10 and 12, respectively. Because AGVs 7, 10, and 12 are first chosen, their operations are in Stage 1. Then AGV 11 is for Task 12, and AGV 6 is for Task 6 in Stage 2. AGVs 7, 10, 11, and 12 are occupied, so AGV 13 is selected for Task 19 with fewer costs in Stage 2. Then the system decided that perform Task 4 with AGV 11, Task 3 with AGV 3, and Task 15 with AGV 10. AGVs 3 and 11 are in Stage 3, and AGV 10 is in Stage 2. The stages of the tasks are as follows, and the consumed time consists of the time following the path and operating the task. Figure 9 shows the entire AGVs' path and AGVs' locations highlighted by circles in different timeslots.

1. Stage 1:
   - Task 1: AGV 7, 142s
   - Task 2: AGV 3, 123s
   - Task 5: AGV 5, 144s
   - Task 7: AGV 6, 135s
   - Task 8: AGV 8, 165s
   - Task 9: AGV 10, 136s
   - Task 10: AGV 2 → AGV 9, 165s → 170s
   - Task 11: AGV 15, 154s
   - Task 13: AGV 4, 172s → 173s
   - Task 14: AGV 14, 150s
   - Task 16: AGV 11, 149s
   - Task 17: AGV 12, 140s
   - Task 18: AGV 13, 169s
2. Stage 2:

- Task 6: AGV 6, 134s
- Task 12: AGV 11, 127s
- Task 15: AGV 10, 156s
- Task 19: AGV 13, 162s
- Task 20: AGV 3, 145s

3. Stage 3:
   - Task 3: AGV 3, 134s
   - Task 4: AGV 11, 136s

From the comparative analysis, the proposed algorithm's Friedman rank is 1.375 when comparing the solution optimality, runtime, and iterations for shifted and rotated benchmark functions. C-PSO has rapid convergence in f_1, f_2 and f_4, which represents a great ability to solve the problems of multi-modal and huge local optima. It demonstrated that the proposed algorithm could eliminate the effect of huge local optima.

C-PSO, PSO and PSO-GA have the best ranks of mean correct digits, but the proposed C-PSO uses the least computational runtime. Also, for path planning, C-PSO saves 70% computational runtime and 69% iterations to get the solutions that have less costs. Its convergence is faster than PSO, which is important for time-critical implementations for decision-making.

The computational experiment is conducted based on the scenario from a real warehouse, including 20 tasks and 15 AGVs. It aims to provide a general scenario for an industrial or logistics warehouse. The proposed multi-AGV approach considers the task priority first, then the task costs. It employs an AGV routing table, a Task table and a Staging table for determining the task assignments. The system will assign an AGV to complete the most urgent tasks.

When assigning an AGV, the system decides based on the low to high task costs, and the routing table will be checked for AGV status and whether it is available. If an AGV is assigned, the next task will be performed in the next stage. If the path passes through more obstacles, the iterations would be raised. The approach also provides fault-tolerance ability to ensure the system's adaptivity and functioning. It considers the available AGV first; if there is no available AGV, then the AGV is closely located to the failed AGV after completing the current task.

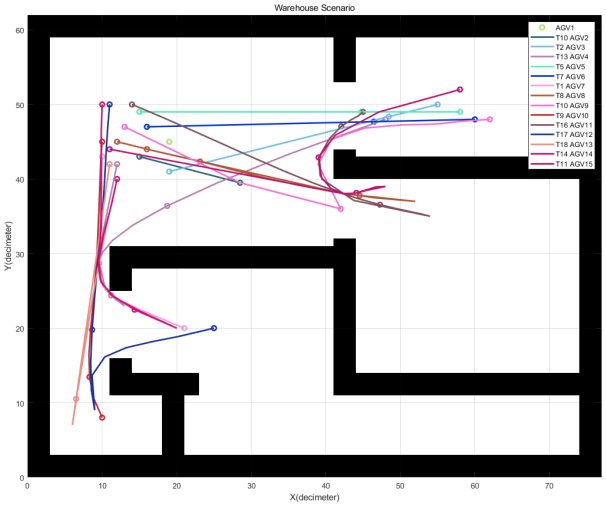
6. Conclusion

This paper presents the Cultural-PSO algorithm for improving the PSO algorithm. It is aimed to provide a solution to the multi-AGV path planning problem. The proposed C-PSO uses adaptive inertia weight to balance the global and local search abilities to overcome the drawback of the evolutionary algorithm. It updates the inertia weight with the probability gained by the improved Metropolis rule. It enhances the search abilities for the hybrid evolutionary algorithm without being trapped in the local optima. When the average fitness value of the swarm is larger than the current particle, it is possible to set the inertia weight larger to search for more space. It can also avoid local optima without slowing down the convergence speed. The proposed C-PSO has the best Friedman rank as 1.375, and PSO, PSO-GA, ABC, TS, MA, CA, HPSGWO, and HS has 2.125, 3.75, 4.875, 4.875, 6, 6.75, 7.125, and 8.125, respectively. In multi-AGV path planning, C-PSO only costs around 30% mean iterations and runtime to get the best solution than PSO.

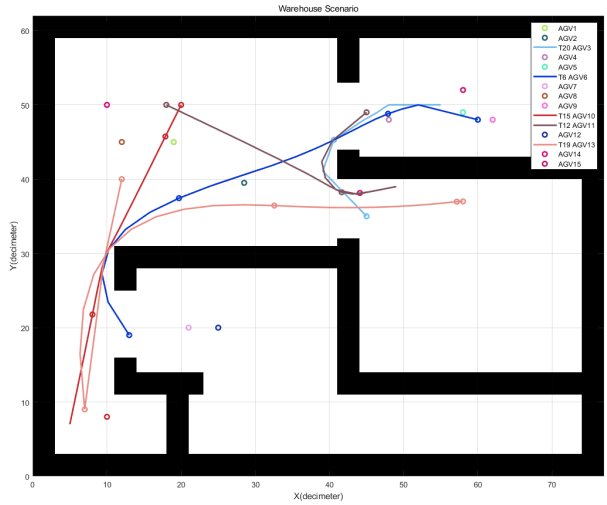
Additionally, it fills the gap of fault tolerance for the multi-robot system. It provides task allocation and fault tolerance considerations with centralized and decentralized layers. The centralized layer is utilized to process initial task allocation and path planning. The decentralized layer is aimed at performing real-time actions according to the changes in the AGV groups and environment. An AGV is intended to communicate with other AGVs to transfer information about positions and angles in the decentralized layer. If collisions or deadlocks occur, the proposed approach regenerates the path. When an AGV is not operated as expected, the proposed system addresses the fault.

The future work is regarding the practical implementation of the proposed approach. The control of the physical system needs to consider the dynamics and real dimensions of the AGVs, such as (Husain et al., 2022). The dynamics of a robot involve the forces, and it enables the predictions of motion. For theoretical limitations, the assumptions of completeness of path generation would be analysed. The sensor fusion algorithm would be developed for detecting the surroundings and environment.

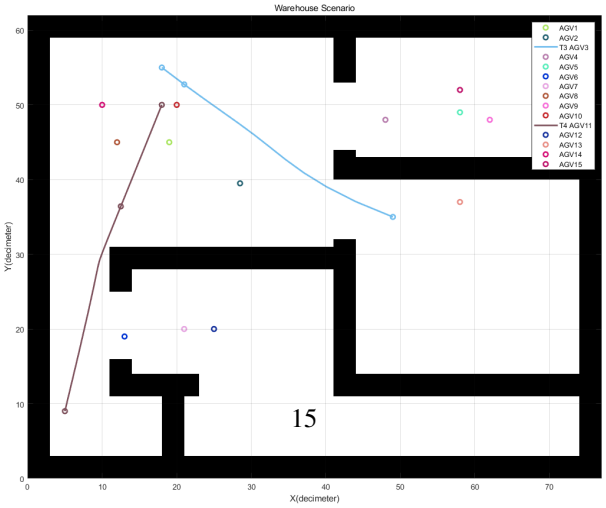
The equipment installed in the robots should also be considered in practical applications. The potential sensors may include distance sensors, vision-based sensors, localization sensors, and inertial measurement units. Distance and vision-based sensors are for obstacle avoidance and robot perception, and localization sensors and inertial measurement units are to estimate the positions and angles. The fusion model should focus on the non-linear system, such as Extend Kalman Filter or a combination with a neural network.



(a) AGVs' paths in Stage 1. Each AGV performs its task and is marked by different colours. But AGV 2 shuts down, then AGV 9 promotes fault-tolerance to perform the reminding task of AGV 2. The circles indicate the positions of each AGV in different timeslots when $t = 0$, $t = 10$, $t = 20$ and $t = 30$.



(b) AGVs' paths in Stage 2. The circles indicate the positions of each AGV in different timeslots when $t = 130$, $t = 140$, $t = 150$, $t = 160$, $t = 170$, $t = 180$, $t = 190$, $t = 200$ and $t = 210$.



(c) AGVs' paths in Stage 3. The circles indicate the positions of each AGV in different timeslots when $t = 270$, $t = 280$, and $t = 290$.

Figure 9: AGVs' paths and locations

Table 14: Path costs of robots for each task

	AGV1	AGV2	AGV3	AGV4	AGV5	AGV6	AGV7	AGV8	AGV9	AGV10	AGV11	AGV12	AGV13	AGV14	AGV15
T1	-	-	38.0908	43.9684	46.6473	45.5123	**12.3582**	29.7382	50.6975	13.1650	42.6659	15.1662	28.2145	30.2714	47.2653
T2	-	16.3431	**14.9432**	17.5951	16.1050	15.7461	18.3165	15.8835	16.9428	18.2108	16.5000	17.6000	17.8886	17.6590	17.7629
T3	-	-	**31.4154**	42.6019	43.6325	49.1261	49.2477	47.1465	47.7080	45.0201	41.7942	46.0558	35.2033	47.1295	45.0588
T4	-	-	40.0075	58.0920	48.5954	42.9250	40.4011	31.6697	52.6373	43.5986	**30.4573**	47.3837	49.0299	32.1215	49.1746
T5	-	17.5666	41.3987	50.8105	**17.4000**	48.2236	19.5494	23.0868	18.2178	19.4666	17.8046	19.0042	23.4361	24.0052	19.1061
T6	-	-	48.9306	63.6397	40.9536	**40.9190**	41.1522	56.7450	45.1997	41.0391	41.8489	**40.5205**	57.5129	56.0205	42.7205
T7	-	18.3108	16.8552	19.5498	18.2045	**17.8046**	20.2998	17.8726	19.0042	20.2360	18.6174	19.8163	19.9466	19.6802	19.8652
T8	-	31.5424	**29.8389**	32.6671	32.2117	31.5745	33.5158	**31.2326**	32.6989	33.6620	32.7614	33.8479	33.0638	32.5868	33.1905
T9	-	-	40.1065	45.9917	48.7028	47.5517	**14.4059**	31.7768	52.7367	**15.2055**	44.6942	17.2231	30.2086	32.3049	47.6895
T10	-	**24.3994**	38.1010	37.9517	**25.2290**	38.8454	26.3450	42.2707	**25.6488**	26.5390	25.7643	26.8482	42.2485	42.3137	26.0486
T11	-	**27.4002**	30.4540	40.5152	**28.0450**	41.4371	29.3482	45.2552	**28.5320**	29.5334	**28.5838**	29.7046	45.3225	45.1566	**29.0463**
T12	-	-	41.0697	40.9181	42.0462	41.8982	**29.7467**	45.6393	46.2054	**29.8981**	**37.6582**	**30.1415**	45.7211	45.5944	43.3869
T13	-	29.5490	29.7098	**28.8387**	31.8360	31.2090	29.2270	30.0808	30.9032	29.9488	32.2074	32.0101	**28.7789**	**28.0508**	29.5946
T14	-	26.5965	26.7408	25.8738	28.8774	28.2662	26.1812	27.1202	27.9286	26.9872	29.1754	29.0014	25.8337	**25.1030**	26.6072
T15	-	-	57.2417	54.9070	63.6528	39.8034	41.8766	50.8229	67.7894	**34.5067**	49.6113	43.2097	64.1902	51.2830	65.3329
T16	-	**27.2157**	29.3909	40.3846	**27.8986**	41.8230	29.1794	45.0605	**28.3827**	29.3397	**28.4228**	29.5318	45.1383	45.0067	28.8636
T17	-	-	47.1432	52.4485	53.5816	53.4032	**23.2363**	40.6173	57.7660	**24.0403**	49.1263	**26.0634**	39.0482	41.1399	54.9626
T18	-	28.5430	28.6635	27.8260	30.8245	30.1800	28.1097	29.0626	29.8656	28.9050	31.1206	30.9370	**27.7632**	**27.0418**	28.5556
T19	-	-	66.0766	71.3523	72.6367	72.3224	**42.2582**	59.6444	76.6046	**43.0553**	68.0209	**45.0832**	**58.0782**	60.1492	73.7693
T20	-	22.5712	**15.3000**	23.8288	22.2836	21.9337	24.5333	22.1006	23.1289	24.4086	22.6775	23.8775	24.2195	24.0227	24.0709

References

Bacanin, N., Bezdan, T., Venkatachalam, K., Zivkovic, M., Strumberger, I., Abouhawwash, M., and Ahmed, A. B. (2021a). Artificial neural networks hidden unit and weight connection optimization by quasi-refection-based learning artificial bee colony algorithm. *IEEE access*, 9:169135–169155.

Bacanin, N., Stoean, R., Zivkovic, M., Petrovic, A., Rashid, T. A., and Bezdan, T. (2021b). Performance of a novel chaotic firefly algorithm with enhanced exploration for tackling global optimization problems: Application for dropout regularization. *Mathematics (Basel)*, 9(21):2705.

Cao, X. and Zhu, M. (2021). Research on global optimization method for multiple agv collision avoidance in hybrid path. *Optimal Control Applications and Methods*, 42(4):1064–1080.

Cardarelli, E., Digani, V., Sabattini, L., Secchi, C., and Fantuzzi, C. (2017). Cooperative cloud robotics architecture for the coordination of multi-agv systems in industrial warehouses. *Mechatronics*, 45:1–13.

Das, P. K., Behera, H. S., Das, S., Tripathy, H. K., Panigrahi, B. K., and Pradhan, S. K. (2016). A hybrid improved pso-dv algorithm for multi-robot path planning in a clutter environment. *Neurocomputing (Amsterdam)*, 207:735–753.

Das, P. K. and Jena, P. K. (2020). Multi-robot path planning using improved particle swarm optimization algorithm through novel evolutionary operators. *Applied soft computing*, 92:106312.

Derrac, J., García, S., Molina, D., and Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and evolutionary computation*, 1(1):3–18.

Eberhart, R. and Kennedy, J. (1995). A new optimizer using particle swarm theory. In *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pages 39–43. IEEE.

Faridi, A. Q., Sharma, S., Shukla, A., Tiwari, R., and Dhar, J. (2018). Multi-robot multi-target dynamic path planning using artificial bee colony and evolutionary programming in unknown environment. *Intelligent service robotics*, 11(2):171–186.

Farooq, B., Bao, J., Raza, H., Sun, Y., and Ma, Q. (2021). Flow-shop path planning for multi-automated guided vehicles in intelligent textile spinning cyber-physical production systems dynamic environment. *Journal of manufacturing systems*, 59:98–116.

Garg, H. (2016). A hybrid pso-ga algorithm for constrained optimization problems. *Applied mathematics and computation*, 274:292–305.

Goldberg, D. E. and Holland, J. H. (1988). Genetic algorithms and machine learning. *Machine learning*, 3(2):95–99.

Guo, X., Ren, Z., Wu, Z., Lai, J., Zeng, D., and Xie, S. (2020). A deep reinforcement learning based approach for agvs path planning. In *2020 Chinese Automation Congress (CAC)*, pages 6833–6838.

Han, Z., Wang, D., Liu, F., and Zhao, Z. (2017). Multi-agv path planning with double-path constraints by using an improved genetic algorithm. *PloS one*, 12(7):e0181747–e0181747. Competing Interests: The authors have declared that no competing interests exist.

Hu, H., Yang, X., Xiao, S., and Wang, F. (2021). Anti-conflict agv path planning in automated container terminals based on multi-agent reinforcement learning. *International journal of production research*, ahead-of-print(ahead-of-print):1–16.

Husain, S. S., Kadhim, M. Q., Al-Obaidi, A. S. M., Hasan, A. F., Humaidi, A. J., and Al Husaeni, D. N. (2022). Design of robust control for vehicle steer-by-wire system. *Indonesian Journal of Science and Technology*, 8(2):20.

Jalili, S. (2022). *Cultural Algorithms (CAs)*, pages 29–57. Springer Nature Singapore, Singapore.

Karaboga, D. and Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. *Journal of global optimization*, 39(3):459–471.

Lee, K. S. and Geem, Z. W. (2005). A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. *Computer methods in applied mechanics and engineering*, 194(36):3902–3933.

Lian, Y. and Xie, W. (2019). Improved a multi-agv path planning algorithm based on grid-shaped network. In *2019 Chinese Control Conference (CCC)*, pages 2088–2092. Technical Committee on Control Theory, Chinese Association of Automation.

Lian, Y., Xie, W., and Zhang, L. (2020). A probabilistic time-constrained based heuristic path planning algorithm in warehouse multi-agv systems. In *the 21st IFAC World Congress*, volume 53, pages 2538–2543.

Liao, X., Wang, Y., Xuan, Y., and Wu, D. (2020). Agv path planning model based on reinforcement learning. In *2020 Chinese Automation Congress (CAC)*, pages 6722–6726. IEEE.

Lin, S., Liu, A., Wang, J., and Kong, X. (2022). A review of path-planning approaches for multiple mobile robots. *Machines*, 10(9):773.

Lin, S., Liu, A., Wang, J., and Kong, X. (2023). An intelligence-based hybrid pso-sa for mobile robot path planning in warehouse.

Journal of Computational Science, 67:101938.

Liu, J., Wang, Z., Xu, Q., and Huang, Q. (2015). Path scheduling for multi-agv system based on two-staged traffic scheduling scheme and genetic algorithm. *Journal of Computational Methods in Sciences and Engineering*, 15(2):163–169.

Malakar, S., Ghosh, M., Bhowmik, S., Sarkar, R., and Nasipuri, M. (2020). A ga based hierarchical feature selection approach for handwritten word recognition. *Neural computing & applications*, 32(7):2533–2552.

Matos, D., Costa, P., Lima, J., and Costa, P. (2021). Multi agv coordination tolerant to communication failures. *Robotics (Basel)*, 10(2):55.

Mirrashid, M. and Naderpour, H. (2022). Transit search: An optimization algorithm based on exoplanet exploration. *Results in control and optimization*, 7:100127.

Mu, T., Zhu, J., Li, X., and Li, J. (2020). *Research on Two-Stage Path Planning Algorithms for Storage Multi-AGV*, book section Chapter 35, pages 418–430. Communications in Computer and Information Science. Springer.

Nazarahari, M., Khanmirza, E., and Doostie, S. (2019). Multi-objective multi-robot path planning in continuous environment using an enhanced genetic algorithm. *Expert systems with applications*, 115:106–120.

Reynolds, R. G. and ChanJin, C. (1996). A self-adaptive approach to representation shifts in cultural algorithms. pages 94–99. IEEE.

Sabattini, L., Digani, V., Secchi, C., and Fantuzzi, C. (2017). Optimized simultaneous conflict-free task assignment and path planning for multi-agv systems. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1083–1088. IEEE.

Senel, F. A., Gokce, F., Yuksel, A. S., and Yigit, T. (2019). A novel hybrid pso–gwo algorithm for optimization problems. *Engineering with computers*, 35(4):1359–1373.

Smolic-Rocak, N., Bogdan, S., Kovacic, Z., and Petrovic, T. (2010). Time windows based dynamic routing in multi-agv systems. *IEEE Transactions on Automation Science and Engineering*, 7(1):151–155.

Solichudin, S., Triwiyatno, A., and Riyadi, M. A. (2020). Conflict-free dynamic route multi-agv using dijkstra floyd-warshall hybrid algorithm with time windows. *International Journal of Electrical and Computer Engineering (IJECE)*, 10(4).

Steinbrunn, M., Moerkotte, G., and Kemper, A. (1997). Heuristic and randomized optimization for the join ordering problem. *The VLDB journal*, 6(3):191–208.

Sun, S., Li, J., Zhu, J., Chen, H., Xu, Q., Liu, J., Cui, H., Li, G., and Hu, W. (2020). Optimization of waste smoke recovery scheduling strategy based on multi agv. *IOP conference series. Materials Science and Engineering*, 719(1):12082.

Tang, B., Xiang, K., Pang, M., and Zhanxia, Z. (2020). Multi-robot path planning using an improved self-adaptive particle swarm optimization. *International Journal of Advanced Robotic Systems*, 17(5).

Tao, L., Zhang, S., Chen, S., and Zheng, N. (2021). Multi-agv pathfinding for automatic warehouse applications.

Thabit, S. and Mohades, A. (2019). Multi-robot path planning based on multi-objective particle swarm optimization. *IEEE access*, 7:2138–2147.

Tian, S., Li, Y., Kang, Y., and Xia, J. (2021). Multi-robot path planning in wireless sensor networks based on jump mechanism pso and safety gap obstacle avoidance. *Future Generation Computer Systems*, 118:37–47.

Xia, P., Xu, A., and Zhang, Y. (2020). *A Multi-AGV Optimal Scheduling Algorithm Based on Particle Swarm Optimization*, book section Chapter 47, pages 527–538. Communications in Computer and Information Science. Springer.

Xing, L., Liu, Y., Li, H., Wu, C.-C., Lin, W.-C., and Chen, X. (2020). A novel tabu search algorithm for multi-agv routing problem. *Mathematics*, 8(2).

Xu, W. (2017). Path planning for multi-agv systems based on two-stage scheduling. *International Journal of Performability Engineering*.

Yang, Q., Lian, Y., Liu, Y., Xie, W., and Yang, Y. (2022). Multi-agv tracking system based on global vision and apriltag in smart warehouse. *Journal of Intelligent & Robotic Systems*, 104(3).

Yu, D., Hu, X., Liang, K., and Ying, J. (2022). A parallel algorithm for multi-agv systems. *Journal of ambient intelligence and humanized computing*, 13(4):2309–2323.

Yu, N. N., Li, T. K., Wang, B. L., Yuan, S. P., and Wang, Y. (2021). Reliability oriented multi-agvs online scheduling and path planning problem of automated sorting warehouse system. *IOP conference series. Materials Science and Engineering*, 1043(2):22035.

Zając, J. and Małopolski, W. (2021). Structural on-line control policy for collision and deadlock resolution in multi-agv systems. *Journal of Manufacturing Systems*, 60:80–92.

Zervoudakis, K. and Tsafarakis, S. (2020). A mayfly optimization algorithm. *Computers & industrial engineering*, 145:106559.

Zhang, W., Peng, Y., Wei, W., and Kou, L. (2018a). Real-time conflict-free task assignment and path planning of multi-agv system in intelligent warehousing. In *2018 37th Chinese Control Conference (CCC)*, volume 2018, pages 5311–5316. Technical Committee on Control Theory, Chinese Association of Automation.

Zhang, Y. and Li, M. (2020). A multi-agv scheduling planning method based on improved ga. *Journal of physics. Conference series*, 1550(2):22014.

Zhang, Y., Wang, F., Fu, F., and Su, Z. (2018b). Multi-agv path planning for indoor factory by using prioritized planning and improved ant algorithm. *Journal of Engineering and Technological Sciences*, 50(4):534–547.

Zhao, Y., Liu, X., Wu, S., and Wang, G. (2021). Spare zone based hierarchical motion coordination for multi-agv systems. *Simulation Modelling Practice and Theory*, 109.