UTS UNIVERSITY OF TECHNOLOGY SYDNEY

Intelligent Hybrid Approaches for Mobile Robots Path Planning

by Shiwei Lin

Thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy

under the supervision of Dr. Jianguo Wang and Dr. Xiaoying Kong

University of Technology Sydney
Faculty of Engineering and Information Technology

February 2023

Certificate of Original Authorship

I, *Shiwei Lin*, declare that this thesis is submitted in fulfilment of the requirements for the award of *Doctor of Philosophy*, in the School of Civil and Environmental Engineering, Faculty of Engineering and IT, at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by the Australian Government Research Training Program.

Signature: Production Note: Signature removed prior to publication.

Date: May 8, 2023

Dedication

For my family

Acknowledgements

I would like to thank my supervisor, Dr Jianguo Wang, for supporting my PhD research. He is an incredibly responsible supervisor and pays much attention to the student's research projects. He also provides helpful suggestions and ideas and offers professional academic support. Also, I would like to thank my co-supervisor, Dr Xiaoying Kong, for guiding me in the academic area. She provided patience and encouragement and allowed me to pursue a further degree when I was a bachelor. My supervisory panel plays the most important role in my research. Their guidance helps me to know how to become an academic researcher.

Additionally, I would like to thank my research collaborator, Mr Ang Liu, for conducting the experiments and discussing the papers during meetings; and for Mr Yunlong Han, who helped me practice presentations for my assignments and provided positive support as a friend. Also, I would like to thank my friend, Ms Ling Li and Mr Jiaxin Chen, for providing mental support during my candidate period.

My sincere thanks also to my family, especially my parents Mrs Yujuan Xu and Mr Zhencai Lin, who always trusted in me and supported me in my study and life over the past years. I appreciate that Mr Xiaojie Chen, Mr Zaizai, Ms Xiaohei and Ms Xiaoqiu were there for me when I was down. I would also like to thank my deceased grandpa, Dr Wenchuan Xu, who accompanied me during my childhood and taught me how to become a warm person.

Shiwei Lin

May 8, 2023

Sydney, Australia

List of Publications

Related to the Thesis:

1. S. Lin, A. Liu, J. Wang, and X. Kong, "A Review of Path-Planning Approaches for Multiple Mobile Robots," *Machines*, vol. 10, no. 9, p. 773, 2022. [Online]. Available: https://www.mdpi.com/2075-1702/10/9/773.

2. S. Lin, X. Kong, and L. Liu, "Development of an intelligent UAV path planning approach to minimize the costs in flight distance, time, altitude, and obstacle collision," 2019: IEEE, pp. 238-243, DOI: 10.1109/ISCIT.2019.8905119.

3. S. Lin, X. Kong, J. Wang, A. Liu, G. Fang, and Y. Han, "Development of a UAV Path Planning Approach for Multi-building Inspection with Minimal Cost," in *Parallel and Distributed Computing, Applications and Technologies*, Cham, Y. Zhang, Y. Xu, and H. Tian, Eds., 2021 2021: Springer International Publishing, pp. 82-93.

4. S. Lin, X. Kong, J. Wang, and A. Liu, "Helix-HPSO approach for UAV path planning in a multi-building environment," *Journal of Reliable Intelligent Environments*, 2022/11/24 2022, DOI: 10.1007/s40860-022-00196-z.

5. S. Lin, A. Liu, J. Wang, and X. Kong, "An intelligence-based hybrid PSO-SA for mobile robot path planning in warehouse," *Journal of Computational Science*, vol. 67, p. 101938, 2023/03/01/ 2023, DOI: https://doi.org/10.1016/j.jocs.2022.101938.

6. S. Lin, A. Liu, X. Kong, and J. Wang, "Development of Swarm Intelligence Leader-Vicsek-Model for Multi-AGV Path Planning," in *2021 20th International Symposium on Communications and Information Technologies (ISCIT)*,

2021-10-19 2021: IEEE, pp. 49-54, DOI: 10.1109/ISCIT52804.2021.9590578.

7. S. Lin, A. Liu, J. Wang, and X. Kong, "A Dual-Layer Weight-Leader-Vicsek-Model for Multi-AGV Path Planning in Warehouse," *Simulation Modelling Practice and Theory, submitted*, 2022.

8. S. Lin, A. Liu, J. Wang, and X. Kong, "An Fault-tolerant Cultural-PSO for Multi-AGV Path Planning," *Applied Soft Computing Journal, submitted*, 2022.

Others:

9. A. Liu, S. Lin, X. Kong, J. Wang, G. Fang, and Y. Han, "Development of Low-Cost Indoor Positioning Using Mobile-UWB-Anchor-Configuration Approach," in *Parallel Architectures, Algorithms and Programming*, Singapore, L. Ning, V. Chau, and F. Lau, Eds., 2021// 2021: Springer Singapore, pp. 34-46.

10. A. Liu, S. Lin, J. Wang, and X. Kong, "A Method for Non-line of Sight Identification and Delay Correction for UWB Indoor Positioning," in *The 17th IEEE Conference on Industrial Electronics and Applications (ICIEA 2022)*, Chengdu, China, 2022: IEEE.

11. A. Liu, S. Lin, J. Wang, and X. Kong, "A Succinct Method for Non-Line-of-Sight Mitigation for Ultra-Wideband Indoor Positioning System," *Sensors*, vol. 22, no. 21, 2022, DOI: 10.3390/s22218247.

Table of Contents

Certificate of Original Authorship i

Acknowledgements iii

List of Publications iv

List of Figures x

List of Tables xii

Abstract xiv

1 Introduction 1

1.1 Research Objectives and Scope 1

1.2 Research Overview and Organization 3

1.2.1 Chapter 2: Literature Review 5

1.2.2 Chapter 3: UAV-based Path Planning 5

1.2.3 Chapter 4: Particle Swarm Optimization (PSO)-Simulated Annealing (SA) Approach 6

1.2.4 Chapter 5: Dual Layer Weight-Leader-Vicsek-Model (DWLVM) 7

1.2.5 Chapter 6: Hybridization of PSO and Cultural Algorithm . . 7

1.2.6 Chapter 7: Conclusion 8

1.3 Contributions and Publications 8

1.3.1 Main Contributions and Publications 8

1.3.2 Additional Research Contributions 9

2 Literature Review 11

2.1 Introduction 11
2.2 Path Planning Algorithms 13
2.2.1 Path Planning 13
2.2.2 Mobile Robot Navigation 14
2.2.3 Discussion of Path Planning Algorithms 22
2.3 Multi-robot Path Planning 22
2.3.1 Classical Approaches 23
2.3.2 Heuristic Algorithms 26
2.3.3 Bio-Inspired Techniques 28
2.3.4 Artificial Intelligence 36
2.3.5 Others 39
2.3.6 Discussion of Multi-robot Path Planning Classification 41
2.4 Decision-making 42
2.4.1 Centralized 42
2.4.2 Decentralized 44
2.4.3 Discussion of Decision-making Strategies 46
2.5 Discussion and Conclusions 47
2.5.1 Path Planning Algorithms 47
2.5.2 Multi-robot Path Planning 49
2.5.3 Decision-making 56
2.5.4 Challenge 60

3 The Mathematical-based Path Planning algorithms for UAV 61
3.1 Quintic Hermite Interpolation 61
3.1.1 Introduction 61
3.1.2 Description of The Algorithm 62
3.1.3 Implementation of The Algorithm 64
3.1.4 Simulation Results 69
3.1.5 Conclusion of Path Planning for The Terrain Case 69
3.2 Helix-HPSO Approach 71
3.2.1 Introduction 71
3.2.2 Related Work 72
3.2.3 Problem Formulation 76

3.2.4 Helix-HPSO Approach 77
3.2.5 Computational Experiment 85
3.2.6 Conclusion of Path Planning for The Multi-building Case . . . 91

4 An Intelligence-based Hybrid PSO-SA for Mobile Robot Path Planning in Warehouse 94

4.1 Introduction 94
4.2 Hybrid PSO-SA 98
4.2.1 Preliminary Knowledge 98
4.2.2 Hybrid PSO-SA 101
4.3 Experiment 104
4.3.1 Simulation with Test Functions 104
4.3.2 Path Planning 106
4.4 Conclusion 111

5 A Dual-Layer Weight-Leader-Vicsek-Model for Path Planning of Multiple Automatic Guided Vehicles in Warehouse 115

5.1 Introduction 115
5.2 Problem Statement 117
5.3 Related Work 118
5.4 WLVM with Virtual Leaders and Weight 122
5.4.1 Preliminary Knowledge: Vicsek Model 122
5.4.2 Algorithm Description 123
5.4.3 Comparison 131
5.5 Computational Experiments 132
5.5.1 Experiment Settings 132
5.5.2 Results 134
5.5.3 Comparison with Another Algorithm 136
5.6 Conclusion and discussion 138
5.6.1 Discussion 138
5.6.2 Conclusion 139

6 An Fault-tolerant Cultural-PSO for Multi-AGV Path Planning 141

6.1 Introduction 141

6.2 Problem Formulation . . . 142
6.3 Related Work . . . 144
6.4 Preliminary Knowledge . . . 146
6.4.1 Cultural Algorithm (CA) . . . 146
6.4.2 Particle Swarm Optimization (PSO) . . . 148
6.4.3 Metropolis Rule . . . 149
6.5 Cultural-PSO (C-PSO) with Metropolis Rule . . . 150
6.5.1 Overview . . . 150
6.5.2 C-PSO . . . 150
6.5.3 Fault Tolerance . . . 153
6.5.4 Task Allocation . . . 154
6.5.5 Collision Avoidance . . . 156
6.6 Computation Experiments . . . 157
6.6.1 Performance Measurements . . . 157
6.6.2 Experiment of Path Planning . . . 158
6.7 Conclusion . . . 162

7 Conclusions and Future Work **165**
7.1 Single Mobile Robot Path Planning . . . 165
7.2 Multi-robot Path Planning . . . 166
7.3 Future Work . . . 167

References **169**

List of Figures

1.1 Research flow diagram 4
1.2 Thesis contributions 4

2.1 Classification of multi-robot path planning approaches. 23
2.2 Illustration of the APF algorithm. 24
2.3 Demonstration of the RRT algorithm. 25
2.4 Simple example of the A* algorithm. 26
2.5 Search algorithms. 29
2.6 Flowchart of the PSO algorithm. 30
2.7 Flowchart of the GA algorithm. 31
2.8 Changes of the ACO algorithm with different timeslots. 33
2.9 Diagram of a three-layer neural network. 38
2.10 Structure of decision-making framework 42

3.1 The generated paths 70
3.2 The Helix-HPSO 78
3.3 Convergence curve of HPSO 87
3.4 The Fujian Tulou 88
3.5 The flight path for each building 88
3.6 The Helix-HPSO path 89
3.7 The exterior inspection paths 90
3.8 The generated paths by HPSO and PSO 92

4.1 The flowchart of PSO algorithm 100
4.2 The flowchart of SA algorithm 101
4.3 The flowchart of path planning by Hybrid PSO-SA 105

4.4 Convergence curve for PSO-SA . . . 109
4.5 The scenario of a warehouse . . . 109
4.6 The paths for PSO and PSO-SA . . . 110
4.7 The convergence curve of PSO-SA path planning . . . 111
4.8 The scenario of the test side . . . 112
4.9 Robot with UWB . . . 112
4.10 The paths for UWB . . . 113

5.1 Path planning in the map . . . 118
5.2 WLVM process . . . 123
5.3 Principle of Weight-Leader-Vicsek-Model . . . 125
5.4 Dual layer of Weight-Leader-Vicsek-Model . . . 128
5.5 Groups with a different direction . . . 128
5.6 The positions after setting segment delay as 2 segments . . . 129
5.7 The AGV's new position after Collision avoidance . . . 130
5.8 The AGV's new position after Deadlock avoidance . . . 130
5.9 Comparison of Vicsek Model and Weight-Leader-Vicsek-Model . . . 132
5.10 The path for virtual leaders . . . 133
5.11 AGV positions during the path planning process and the grey areas indicate the packing areas . . . 135
5.12 UWB Positions following the generated path . . . 137
5.13 AGV positions generated by the RRT* algorithm . . . 138

6.1 The scenario of problem formulation . . . 143
6.2 The principle of CA . . . 147
6.3 Dual-layer of the C-PSO algorithm . . . 151
6.4 The scenario of fault tolerance . . . 153
6.5 The flow of task allocation . . . 154
6.6 Flowcharts of collision resolution . . . 157
6.7 Convergence curve of C-PSO . . . 159
6.8 AGVs' locations of different timeslot. The paths of AGV 1, AGV 2, AGV 3, AGV 4 and AGV 5 are marked by blue, orange, purple, green and yellow, respectively. . . . 161

List of Tables

2.1 Comparison of navigation methodologies 47
2.2 Comparison of multi-robot path planning algorithms 51
2.3 Comparison of decision-making approaches 57

3.1 Simulated paths for the robot 70
3.2 Test functions 85
3.3 Performance of test functions 86
3.4 Details of the Helix-HPSO path 89
3.5 Costs of the exterior paths 91
3.6 Comparison of PSO and HPSO paths 91

4.1 Test functions 106
4.2 Mean iteration times and fitness value 107
4.3 Runtime of each algorithm 108
4.4 Performance measurements of the simulated paths 111
4.5 Simulated paths 113

5.1 AGV navigation data in the same group 124
5.2 Start-destination matrix 128
5.3 Group settings 134
5.4 Performance measurements 136
5.5 Runtimes for the RRT* and WLVM algorithm 137

6.1 A example of AGV routing table 155
6.2 A example of Task table 156
6.3 A example of Staging table 156

6.4 Test functions 158
6.5 Iterations, runtime, and fitness value 163
6.6 Task list in the simulation 164
6.7 AGVs' initial location 164
6.8 Path costs of robots for each task 164
6.9 Schedule of tasks 164

Abstract

The practical applications of mobile robots are widely implemented in various areas, such as education, industry, environment, and civil applications. The requirements of robots' navigation are one of the primary considerations for autonomous operation. Path planning is essential for the successful application of mobile robots. Considering all available information, it aims to generate the robot's optimal path from the start to the target location. Depending on the operation scenarios, various factors are counted in the cost function for path planning.

Meeting the flexibility, robustness, and efficiency requirements for real-time mobile robot path planning implementation is challenging. A review of multi-robot path planning is published to compare the path planning approaches and decision-making strategies, listing the challenges. This thesis aims to tackle major challenges by developing intelligent hybrid approaches, including 1) trapping in local optimal, 2) slow convergence of path generation, and 3) robots' fault tolerance. It also provides the path planning algorithms for single-robot and multi-robot systems in three-dimensional and two-dimensional space.

For single mobile robot path planning, the bio-inspired approaches have gained more attention recently with high robustness and flexibility. In contrast, it is highly possible to trap in a local optimal. The proposed Harmony-particle swarm optimization algorithm significantly reduces the iterations during planning to solve the aerial path planning problem in a multi-building environment. Also, a hybrid approach of particle swarm optimization and simulated annealing is proposed for single-vehicle path planning in the industrial warehouse scenario. It updates the personal best value to jump out of the locally optimal. Compared with other evolutionary approaches, it shows excellent performance.

Moreover, fast convergence is a significant challenge for multiple robots' path planning. A dual-layer Weight-Leader-Vicsek-Model is proposed that generates the path for the virtual leaders first for each group of robots, and then all the robots will move by following their leaders. This dual-layer approach can achieve fast convergence, generating vehicle paths in one calculation step. Fault tolerance is also an essential issue for the real-time implementation of path planning, but it is lacking in previous studies. The Cultural-Particle Swarm Optimization algorithm is proposed to offer a backup plan in case of system failures. It updates the inertial weight to enhance the search abilities, balancing the global and local search abilities. The experiments and validated results are presented for each proposed approach.

Chapter 1

Introduction

1.1 Research Objectives and Scope

The trend of executing mobile robots has been raised dramatically in the past several years to reduce human repetitive or dangerous work. Robots can carry cameras and sensors or other light equipment to achieve specific missions with the enhancement of technology and economy. The various robot applications include surveillance [1], search and rescue [2], intelligent manufacture [3], agriculture [4], surface inspection [5], tracking [6], and more. The most popular mobile robots in the application consist of Unmanned Aerial Vehicle (UAV) and Automated Guided Vehicle (AGV).

Path planning of navigation is essential for implementing the autonomous operations of robots. Path planning refers to generating a safe and optimal path from the start to the target location. The generated path uses the problem objectives to be chosen to achieve the expected target [7]. The connected robot is intended to provide less computation-constrained and more cost-efficient than the automated robot. However, it has signal dead zones to reduce the operation area and efficiency [8]. The trend of automated and programmable robots is intended in this age. Robots can make simple decisions but cannot carry out sophisticated tasks due to enormous computational resources, extensive memory and many needed AI-based algorithms.

Path planning algorithms can be divided into global and local path planning based

on whether all environment information is accessible [9]. Global path planning has all information before starting, while environmental information is unknown for local path planning. The popular algorithms of path planning consist of classical and metaheuristic algorithms. The classical algorithms include a Rapidly Random Tree (RRT), Probabilistic Road Map (PRM), Dijkstra's Algorithm (DA), and Potential Field (PF). A review paper [7] considers Dijkstra's Algorithm as the benchmark solution.

From a survey [9], PSO (particle swarm optimization), GA (genetic algorithm), ACO (ant colony optimization) and APF (artificial potential field) are the most used techniques for robot path planning. Also, [7] compares several bio-inspired algorithms, including Particle Swarm Optimization (PSO), Differential Evolution (DE), Genetic Algorithm (GA), and Cuckoo Search Algorithm (CSA), and the results show the PSO performs better among these approaches in unknown environments. PSO is improved as a fuzzy enhanced multi-objective optimization algorithm for robot path planning in the static and dynamic environment [10].

Moreover, the new algorithms of mobile path planning are AI-based approaches. Machine learning model-based approaches are the primary stream of AI-based approaches, which allows the time-efficiency of model scorers while it cannot ensure solution correctness [11]. Deep reinforcement learning includes reinforcement learning and deep learning for the mobile robot path planning to deal with decision-making and perception problems [12]. Deep reinforcement learning is firstly modified by iterative training from the data set to obtain the parameters of network weight and bias, then applying the trained network to identify and classify the dataset. AI-based path-planning approaches can be classified as supervised, unsupervised, and reinforcement learning. Supervised and unsupervised learning requires much prior knowledge, while reinforcement learning can use feedback [12].

The motivation of this research project is to achieve robustness, flexibility, and efficiency for path planning. This thesis aims to solve the critical research question of path planning for mobile robots and concentrates on optimising path planning for mobile robots, including UAVs and AGVs. It also provides solutions for single-robot and multiple-robot systems. The theories implemented in this research include the

mathematical-based and bio-inspired-based approaches.

The specific research objectives are:

1. Plan an optimal path for a single or multiple robot system with minimal cost;
2. Solve the problems of local optima in path planning and slow convergence of evolutionary algorithms;
3. Provide the framework for robots' fault tolerance in multiple robot path planning.

Chapter 2 reviews the related literature for the single-robot and multi-robot systems to provide an overview of the state-of-the-art solutions to the path planning problem. The study shows that the bio-inspired optimization framework performs better than the conventional path planning algorithms, and reliable numerous training data is not required for the unknown environment. Therefore, the bio-inspired optimization algorithm is chosen as the key component for enhanced path planning implementation in different mobile robot applications, as introduced in Chapters 3, 4 and 6. Chapter 5 uses another bio-inspired model to describe the multi-robot behaviour during path planning to reduce the computation complexity and load.

1.2 Research Overview and Organization

This research aims to provide systematic optimal solutions for mobile robot path planning. Figure 1.1 describes this thesis's structure with the relationship between the chapters, including single-robot and multi-robot path planning approaches. Path planning algorithms and decision-making strategies were reviewed, and path planning algorithms were classified as classical approaches, heuristic algorithms, bio-inspired and AI-based approaches. From the literature, bio-inspired approaches are popular in this research area.

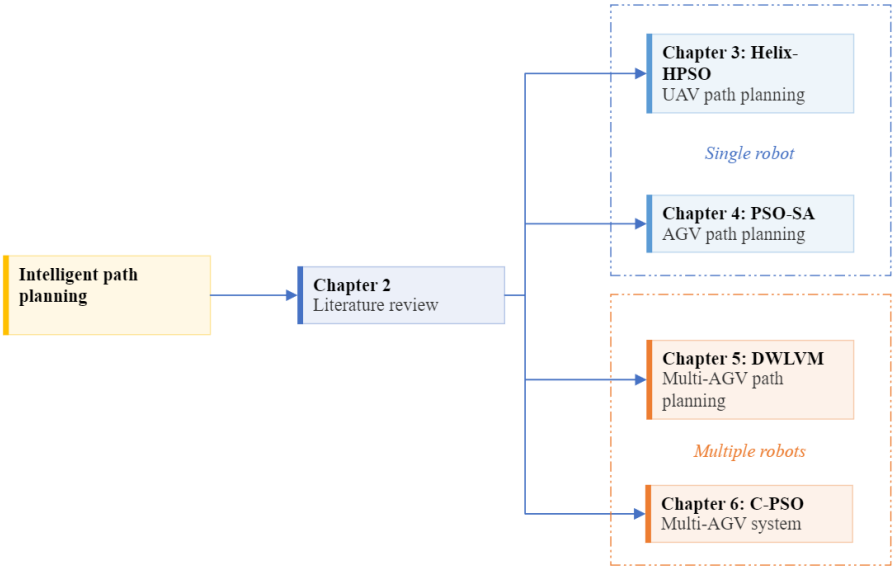


Figure 1.1: Research flow diagram

Figure 1.2 indicates each proposed approach based on the inspiration of natural behaviour. For a single robot system, UAV path planning focuses on mathematical solutions for the terrain based and using the optimization algorithms for the multi-environment building. For the single AGV system, the hybrid intelligent approach based on PSO and SA is introduced.

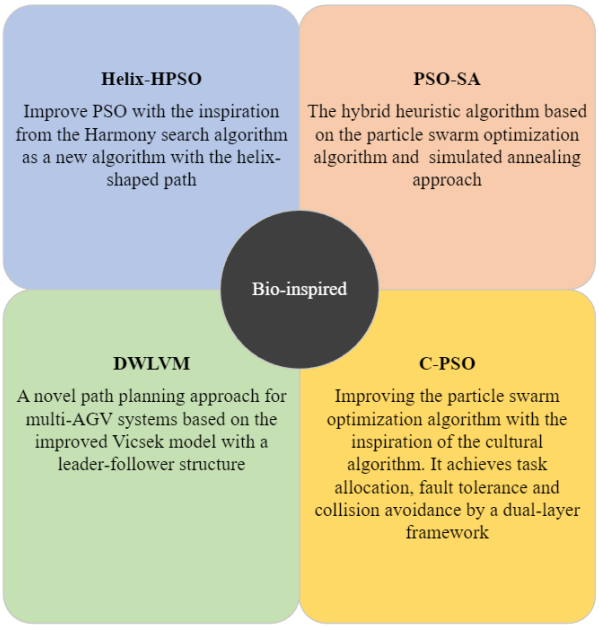


Figure 1.2: Thesis contributions

The dual layer in leader-follower formation for a multi-robot system enhances the biological approach. Also, the improved PSO methodology is inspired by CA and

SA algorithms; it probabilistically updates the inertia weight to balance global and local searches. It provides fault tolerance, path re-generation, collision avoidance and task allocation for online implementation. The contents of each chapter are summarized as follows.

1.2.1 Chapter 2: Literature Review

Numerous path-planning studies have been conducted in past decades due to the challenges of obtaining optimal solutions. This chapter reviews single-robot and multi-robot path-planning approaches and decision-making strategies and presents the path-planning algorithms for various types of robots, including UAVs and AGVs. The path-planning approaches have been classified as classical approaches, heuristic algorithms, bio-inspired techniques, and artificial intelligence approaches. Bio-inspired techniques are the most employed approaches, and artificial intelligence approaches have gained more attention recently. The decision-making strategies mainly consist of centralized and decentralized approaches. The trend of the decision-making system is to move towards a decentralized planner. Finally, the new challenge in multi-robot path planning is proposed as fault tolerance, which is important for real-time operations.

1.2.2 Chapter 3: UAV-based Path Planning

This chapter presents two mathematical based path planning algorithms for different case studies:

1. For the terrain environment: a UAV path planning approach is proposed to consider flight cost functions. The UAV flight paths are generated using Quintic Hermite interpolation. These paths are constant in speed, and the algorithm generates the paths by iteration to ensure the path segments are smooth. A Waypoint-Matrix is to store the points of the path. The paths aim to reach the defined destination by passing the waypoints and avoiding obstacles. This approach proposes the flight cost functions to evaluate the paths, path length, flight time, altitude, and collision.

2. For the multi-building environment: Regular inspection of historic buildings is

essential, while path planning of the building inspection is challenging because it requires comprehensive coverage at a low cost. Most of the previous research does not consider the multiple buildings' environment. In this section, a three-dimensional path planning approach is proposed to provide the inspection for multiple buildings. The proposed Helix-HPSO approach generates the helix-shaped path for each building and uses HPSO for path planning between buildings. The computational experiment validates the proposed approach. The helix-shaped path costs less than the traditional back-and-forth path for building inspection. HPSO is compared with other bio-inspired algorithms for optimization problems and PSO for path planning.

1.2.3 Chapter 4: Particle Swarm Optimization (PSO)-Simulated Annealing (SA) Approach

Mobile robots play crucial roles in industry and commerce, and automatic guided vehicles (AGV) are one of the primary parts of smart manufacturing and intelligent logistics. Path planning is the core task for the AGV system, and it generates the path from origin to destination. The motivation of the study is to improve the scalability, flexibility, adaptability, and performance of the robot path planning systems. This chapter proposes the hybrid PSO-SA algorithm for the optimization of AGV path planning. Compared with other heuristic algorithms by benchmark functions, including HS, FA, ABC and GA, the proposed algorithm shows excellent performance in dealing with optimization problems. It reduces the possibility of getting trapped in one local optimum and enhances the efficiency to get the best global solution with faster convergence and less time consumption. It is evaluated with multiple cost functions and path planning with simulations and experiments. The objective of the proposed algorithm is to minimize the path length and produce a smooth path without collision. The proposed PSO-SA algorithm is compared with PSO in the path planning application, and the mean runtime and iteration times are usually significantly lower than PSO.

1.2.4 Chapter 5: Dual Layer Weight-Leader-Vicsek-Model (DWLVM)

Multiple automatic guided vehicles are widely involved in industrial intelligence. Path planning is crucial for their successful application of them. However, achieving robust and efficient path planning of multiple automatic guided vehicles for real-time implementation is challenging. In this chapter, a two-layer strategy for multi-vehicle path planning is proposed. All the vehicles are grouped by the start-destination matrix, and a dynamic virtual leader is generated for each group. In the first layer, the hybrid A* algorithm is employed for the path planning of the virtual leaders. The second layer is named leader-follower; the proposed Weight-Leader-Vicsek-Model is applied to navigate the vehicles following their virtual leaders. The proposed method can reduce computational load and achieve real-time navigation by quickly updating the grouped vehicles' status. Collision and deadlock avoidance is also conducted in this model. Vehicles in different groups are treated as dynamic obstacles. MATLAB validated the method by simulations to verify its path-planning functionality and conducted the experiment with a localization sensor.

1.2.5 Chapter 6: Hybridization of PSO and Cultural Algorithm

Automatic guided vehicles are widely implemented in modern industrial warehouses. Path planning of multiple vehicles is challenging to generate the optimal path without collisions, and lacking fault-tolerant consideration is hard for real-time implementation. This chapter presents a hybrid evolutionary algorithm Cultural-PSO (C-PSO), inspired by the cultural and particle swarm optimization algorithms. It is aimed to balance the global and local search abilities and avoid trapping in the local optima. The proposed algorithm updates the inertia weight with the probability calculated by the Metropolis rule. More importantly, it provides the fault tolerance functionality for the multi-AGV system to fill the gap. The proposed Cultural-PSO approach achieves task allocation, fault tolerance and collision avoidance by a dual-layer framework. It is validated through computational and path-planning experiments compared with other bio-inspired algorithms.

1.2.6 Chapter 7: Conclusion

This chapter summarizes the results and implications of this work and provides recommended directions for the continuation of this work in the future.

1.3 Contributions and Publications

1.3.1 Main Contributions and Publications

Chapter 2 provides a comprehensive literature review relevant to the overall aim. It includes the review of path planning algorithms for single and multiple robots, and it introduces the decision-making strategies for multi-robot systems. The review of the multi-robot path planning is published as follows.

- S. Lin, A. Liu, J. Wang, and X. Kong, "A Review of Path-Planning Approaches for Multiple Mobile Robots," *Machines*, vol. 10, no. 9, p. 773, 2022. [Online]. Available: https://www.mdpi.com/2075-1702/10/9/773.

Chapter 3 proposes the path planning algorithms for a single UAV, developed under the cases of terrain and multiple buildings. The terrain case uses mathematical solutions, and the multiple-building case combines the improved bio-inspired and helix approaches. The related publications are listed as follows.

- S. Lin, X. Kong, and L. Liu, "Development of an intelligent UAV path planning approach to minimize the costs in flight distance, time, altitude, and obstacle collision," 2019: IEEE, pp. 238-243, DOI: 10.1109/ISCIT.2019.8905119.
- S. Lin, X. Kong, J. Wang, A. Liu, G. Fang, and Y. Han, "Development of a UAV Path Planning Approach for Multi-building Inspection with Minimal Cost," in *Parallel and Distributed Computing, Applications and Technologies*, Cham, Y. Zhang, Y. Xu, and H. Tian, Eds., 2021 2021: Springer International Publishing, pp. 82-93.
- S. Lin, X. Kong, J. Wang, and A. Liu, "Helix-HPSO approach for UAV path planning in a multi-building environment," *Journal of Reliable Intelligent Environments*, 2022/11/24 2022, DOI: 10.1007/s40860-022-00196-z.

Chapter 4 proposes a hybrid approach combining Particle Swarm Optimization (PSO) and Simulated Annealing (SA) to overcome the drawback of bio-inspired approaches and provide the functionality of single robot path planning, with the contribution as follows.

- S. Lin, A. Liu, J. Wang, and X. Kong, "An intelligence-based hybrid PSO-SA for mobile robot path planning in warehouse," *Journal of Computational Science*, vol. 67, p. 101938, 2023/03/01/ 2023, DOI: https://doi.org/10.1016/j.jocs.2022.101938.

Chapter 5 aims to provide fast convergence with the proposed Dual layer Weight-Leader-Vicsek-Model (DWLVM) to achieve multi-robot path planning with fast convergence and calculation, reducing the computation load. It resulted in the following publications.

- S. Lin, A. Liu, X. Kong, and J. Wang, "Development of Swarm Intelligence Leader-Vicsek-Model for Multi-AGV Path Planning," in *2021 20th International Symposium on Communications and Information Technologies (ISCIT)*, 2021-10-19 2021: IEEE, pp. 49-54, DOI: 10.1109/ISCIT52804.2021.9590578.
- S. Lin, A. Liu, J. Wang, and X. Kong, "A Dual-Layer Weight-Leader-Vicsek-Model for Multi-AGV Path Planning in Warehouse," *Simulation Modelling Practice and Theory, submitted*, 2022.

Chapter 6 proposes a hybrid approach as Cultural-PSO (C-PSO) that improves the hybrid optimization approach by the inspiration of the Cultural algorithm to balance the search global and local abilities, providing task allocation and fault tolerance.

- S. Lin, A. Liu, J. Wang, and X. Kong, "An Fault-tolerant Cultural-PSO for Multi-AGV Path Planning," *Applied Soft Computing Journal, submitted*, 2022.

1.3.2 Additional Research Contributions

The additional research focuses on the contributions of a localization sensor as Ultra Wide Band (UWB). The collaborative experiments were conducted with precise indoor positioning UWB for the following publications, working as the team member and co-author. It offers the corrections of localization bias in indoor positioning and

reduces the costs during the experiment setup.

- A. Liu, S. Lin, X. Kong, J. Wang, G. Fang, and Y. Han, "Development of Low-Cost Indoor Positioning Using Mobile-UWB-Anchor-Configuration Approach," in *Parallel Architectures, Algorithms and Programming*, Singapore, L. Ning, V. Chau, and F. Lau, Eds., 2021// 2021: Springer Singapore, pp. 34-46.

- A. Liu, S. Lin, J. Wang, and X. Kong, "A Method for Non-line of Sight Identification and Delay Correction for UWB Indoor Positioning," in *The 17th IEEE Conference on Industrial Electronics and Applications (ICIEA 2022)*, Chengdu, China, 2022: IEEE.

- A. Liu, S. Lin, J. Wang, and X. Kong, "A Succinct Method for Non-Line-of-Sight Mitigation for Ultra-Wideband Indoor Positioning System," *Sensors*, vol. 22, no. 21, 2022, DOI: 10.3390/s22218247.

Chapter 2

Literature Review

2.1 Introduction

Robot applications have been widely implemented in various areas, including industry [13], agriculture [4], surveillance [14], search and rescue [15], environmental monitoring [16], and traffic control [17]. A robot is referred to as an artificial intelligence system that integrates microelectronics, communication, computer science, and optics [18]. Due to the development of robotics technology, mobile robots have been utilized in different environments, such as Unmanned Aerial Vehicles (UAVs) for aerospace, Automated Guided Vehicles (AGVs) for production, Unmanned Surface Vessels (USVs) for water space, and Autonomous Underwater Vehicles (AUVs) for underwater.

To perform tasks, employing a set of vehicles cooperatively and simultaneously has gained interest due to the increased demand. Multiple robots can execute tasks in parallel and cover larger areas. The system continues working even with the failure of one robot [19], and it has the advantages of robustness, flexibility, scalability, and spatial distribution [20]. Each robot has its coordinates and independent behaviour for a multi-robot system, and it can model the cooperative behaviour of real-life situations [21]. For reliable operation of the robot, the robotics system must address the path/motion planning problem. Path planning aims to find a collision-free path from the source to the target destination.

Path planning is an NP-hard problem in optimization, and it involves multiple objectives, resulting in its solution being polynomial verified [22]. The robots are aimed to accomplish the tasks in the post-design stage with higher reliability, higher speed, and lower energy consumption [23]. Task allocation, obstacle avoidance, collision-free execution, and time windows are considered [24]. Multi-robot path planning has high computational complexity, which results in a lack of complete algorithms that offer solution optimality with computational efficiency [25].

Substantial optimality criteria have been considered in path planning, such as the rendezvous and operation time, path length, velocity smoothness, safety margin, and heading profiles for generating optimal paths [26]. During missions, the robot systems have limitations, such as limited communication with the centre or other robots, stringent nonholonomic mission constraints, and limited mission length because of weight, size, and fuel constraints [27]. The planned path must be a smooth curve due to nonholonomic motion constraints and support kinematic constraints with geometric continuity. Furthermore, the path's continuity is significant for collaborative transport [28].

Path-planning approaches can be divided into offline and real-time implementation. Offline generation of a multi-robot path cannot exploit the cooperative abilities, as there is little or no interaction between robots, leading to the multi-robot system not ensuring that the robots are moving along a predefined path or formation [29]. Real-time systems have been proposed to overcome the problems created by offline path generation, and these can maximize the efficiency of algorithms. The chart of offline/real-time implementation included in the literature review is exhibited in the discussion section.

Decision-making strategies can be classified as centralized and decentralized approaches. The centralized system has the central decision-maker, and thus the degree of cooperation is higher than in the decentralized approach. All robots are treated as one entity in the high-dimensional configuration space [30]. A central planner assigns tasks and plans schedules for each robot, and the robots start operation after completion of the planning [31]. The algorithms used in the centralized structure are without limitation because the centralized system has better global

support for robots.

However, the decentralized approach is more widely employed in real-time implementation. Decentralized methods are typical for vehicle autonomy and distributed computation [32]. These have the robots communicate and interact with each other and have higher degrees of flexibility, robustness, and scalability, thereby supporting dynamic changes. The robots execute computations and produce suboptimal solutions [31]. The decentralized approach includes task planning and motion planning, and it reduces computational complexity with limited shared information [33].

Many surveys have been conducted for the mobile robot path planning strategies [34]–[36]; however, these papers only focus on single robot navigation without cooperative planning. This review's motivation is to introduce the state-of-art path-planning algorithms of multi-robot systems and provide an analysis of multi-robot decision-making strategies, considering real-time performance. This chapter investigates 2D or ground path planning and the 3D environment.

The recent literature is reviewed and classified the path-planning approaches based on the main principles. The chapter is organized as follows. Section 2.2 discusses the path planning algorithms. Section 2.3 discusses multi-robot path planning approaches. Section 2.4 discusses the decision-making strategies.

2.2 Path Planning Algorithms

2.2.1 Path Planning

Path planning has been the most crucial consideration of mobile robot navigation, which plans the path from the start to the target for mobile robots [37]. Obstacle avoidance functions must be developed to operate AGVs when considering dynamical limitations and dynamical safety [38], [39]. Motion planning regards safe collision-free and shortest distance as constraints when generating a path and converting the path into control variables such as velocity and acceleration [40].

Meta-heuristic-based methods, graph-search-based methods, mathematical optimisation-based methods, and virtual potential field and navigation-based methods are the

four categories for navigation algorithms [38]. The classic graph search algorithm has been widely implemented for AGV path planning, such as the A* and Dijkstra algorithm [40], [41].

Besides, the A* algorithm is used to establish an optimal path on the cost map for motion planning, and the AGV equips with a SLAMTEC laser scanner for detecting the dynamic obstacles' position [40]. Different algorithms can be combined to enhance performance and adaptability and overcome some drawbacks of each algorithm. For example, the Dijkstra algorithm is for initial static path planning, and the virtual potential function algorithm is for dynamic path planning [42].

Also, sampling-based methods are significant for single AGV path planning, such as the Voronoi graph and RRT methods [43]. More algorithms generate the optimal path in a known map; artificial potential field methods and the probabilistic road map (PRM) are also proposed for global and local control with real-time operation and feasibility [40].

Besides, the mathematical optimisation-based approach consists of an open-loop strategy and a closed-loop strategy [38]. A nonlinear predictive control (MPC) algorithm has been proposed for large-size AGVs with onboard LIDAR sensors and a 14-DoF vehicle dynamics model [38]. The cost function of the nonlinear MPC algorithm is used to evaluate the heading angle, the distance, and the control effort [38].

Deep reinforcement learning can process high-dimensional environment data, such as images, and it has intelligent decision-making ability and powerful perception ability [37]. A neural network structure and Dueling DDQN-PER has been implemented as AGV path planning for multi-modal sensing environmental information, such as global positioning system (GPS), cameras, and speed sensors [37].

2.2.2 Mobile Robot Navigation

AGV Navigation

Mobile robots are applied to the industry with the development of intelligent logistics, industrial intelligence, and intelligent factories [37], [43]. AGVs have many

commercial and industrial applications, resulting in increasing significance to improve transportation efficiency with fewer transportation costs [38], [43], [44]. AGV can improve safety and decrease labour costs in a production environment for the high demands in production [45], [46]. AGV for industry comprises towing vehicles, unit load vehicles, pallet trucks, and forklifts [47].

AGVs are utilised in flexible production lines of modern manufacturing and integrated into automated intelligent control systems [46]–[48]. They can operate at high speeds to transfer the products in chaotic situations in warehouses. AGVs become essential parts of the servo handling system, logistics, and AGVs become an essential part of the servo handling system, logistics, warehousing system, and storage industry [46], [49].

AGVs are employed in varying areas, such as distribution, material handling in manufacturing, transportation, and transhipment [3]. Detecting objects in the path and eliminating the problems automatically with sensors' help can increase the adaptability and intelligence of AGVs [46].

Sensors for AGVs cover close-range infrared sensors, a global positioning system, and long-ranged high-frequency radar [39]. The navigation system determines the sensors equipped for navigation, and the primary sensors for AGVs are the magnetic guide sensor and the laser scanner [47].

Magnetic sensors receive path information from the magnetic stripe quickly and accurately [49]. Location can be obtained by calculating of mathematical equation for a fixed path with a known start position by rotational counters, inclinometers, and electronic compasses or by an outside system source technology [39].

Additionally, the basic principles for AGV navigation can be divided into a fixed path and free navigation. The critical technologies based on the principles consist of the inertial guide, laser guide, vision-based guide, magnetic spot guidance, barcode guidance, and wired guide [39], [46], [47], [50].

A directed network considers aisle intersections, delivery, and pick-up locations as nodes in a guide-path layout based on the flow topology [51]. Free-ranging allows the AGV to deviate significantly from guide-paths and program the preferred tracks

by software [51].

Single navigation method usually has disadvantages. Inertial navigation system (INS) has increasing navigation error and slowly varying drift [48], [52]. Simultaneous localisation and mapping (SLAM) navigation cannot stitch the map in real-time [43], [52]. The real-time performance is poor for visual navigation and is constrained by weather factors [52]. Global Navigation Satellite System (GNSS) has continuous and real-time precise positioning data, but a slow data update rate and positioning accuracy would be affected if the AGV is in a container [52], [53].

The costs would increase when the working area expands, more communication base stations are required for Ultra Wide Band (UWB) wireless positioning and navigation, and the coverage of reference nodes determines its performance [48], [52]. The typical navigation methods involve visual image navigation, INS, GNSS, lidar SLAM navigation, antenna radar-transponder navigation, and UWB wireless navigation [45], [52].

Indoor localisation methods use infrared, ultrasonic, radio, and magnetic sensors [54]. Benchmarking low-cost and medium-cost inertial measurement units (IMUs) are mounted on an AGV and apply SLAM in indoor production sites [45]. Moreover, artificial landmarks are recognised visually to help AGV obtain rich guide information to increase its intelligence and attain easy reconfiguration or high flexibility, fusing the position information from the camera, and the laser [51].

Web cameras and marker recognition are explored for an efficient, low-cost, guiding AGV system [54]. Magnet spot guidance is achieved by encoders, counters, and hall-effect sensors through a dead reckoning with marker recognition [54]. For developing low-cost guidance systems, RFID and vision sensors are arranged with a topological map describing the flow-path network [51].

Also, the RFID sensor obtains the site position information for AGVs, making AGVs execute corresponding instructions and identify the positioning block [49]. LiDAR, RFID site labels, and ultrasonic sensors are adopted to a navigation system that is controlled by the PID algorithm [49]. LiDAR and ultrasonic sensors ensure the safe operation of AGVs [49]. Ultrasonic sensors, light sensors, and cameras are combined for the fixed path with a programmable robotics kit [46].

Furthermore, distributed filtering algorithms estimate the AGV's state vector based on the measurements coming from onboard sensors through a multi-stage fusion procedure [52], [55]. Failure-prone sensors provide the measurement which can estimate the AGV's Cartesian coordinates, such as wheel encoders [55]. A fault detection and isolation algorithm through the statistical processing of residuals offers an indication of the condition of sensors, and possible failures [55].

AGV navigation includes AGV's attitude and position estimation and correction, obstacle detection and avoidance, path tracking and control of drive devices, and attitude estimation consisting of relative and absolute ways [56]. The navigation combines the analysis with magnetic nails, encoders, acceleration sensors, inertial guidance, gyros, ultrasonic sensor, and an infrared sensor for multi-sensor fusion through Kalman filtering [56]. Also, the Kalman filtering algorithm is applied to the Dead Reckoning (DR) signal and GPS signal to compensate for each other and gain high-precision positioning data [53].

An interactive multiple model algorithm decreases the impact of positioning in non-line-of-sight and line-of-sight states when integrating INS and UWB, setting up two Kalman filters in the states by different distance error characteristics [48]. Extended Kalman filter and optimal weighted voting fusion and employed for an improved integrated navigation method based on GNSS, single-axis rotating INS, and kinematics [52]. An enhanced distributed nonlinear Kalman filtering has been proposed as the derivative-free extended information filter with getting the data from GPS, IMU, laser, gyrocompasses, and vision sensors [55].

UAV Navigation

Besides the 2D navigation, 3D navigation for UAVs is also reviewed. Creating a collision-free and safe path is critical for UAV navigation [57]. The challenges of creating paths include awareness of the surroundings and navigational ability [58]. More precisely, to apply the navigation algorithms, the environment or terrain information is required to be transferred by methods.

The perception of the environment is achieved by sensors, such as laser scanners, LiDARs, cameras, and sonars [58]. Defining obstacles can be achieved by p-norms,

Dubins curves, polygonal shapes and sampling-based planning [58]. The discretisation of airspace by network and edges' costs can be used for UAV path planning [59].

Determining the optimal path is mainly based on the discretisation of the environment data [60]. 3D path planning can be transferred to the constraint optimal control problem, considering dynamic and kinematic equations of UAV flight [61]. The flight time, the technical properties of UAVs, the path length, and the path's straightness are taken into account to determine the optimal path [59]. The path must be smooth to follow UAVs' kinematic and dynamic properties [62].

Some graph search algorithms and optimisation methods are applied for UAV navigation. The improved sparse A* algorithm is proposed for UAVs in the marine environment [63]. A Sparse bidirectional A* search algorithm is developed and has the robustness and better path planning performance than the Sparse A* algorithm [64].

The continuous optimisation method is proposed as the fast-marching solution, and it can produce smoothed and continuous trajectory [60]. Mixed integer linear programs, the Dijkstra algorithm, D* algorithm, Theta* algorithm, and some graph search algorithms are also solutions [65]. A* algorithm performs better than Bellman Ford's, Dijkstra, and Floyd-Warshall's algorithms because of heuristic search and optimal path guarantee [66].

Additionally, there are some frameworks and developed algorithms. Bio-inspired optimisation framework can be applied to UAV path planning, such as ant colony optimisation [60]. For real-time path planning, particle swarm optimisation and genetic algorithms can be applied [67]. The anytime algorithm improves the particle swarm optimisation algorithm to generate feasible paths in an obstacle-rich environment [68].

Splines, Bezier curves, polynomials, and semi-analytical methods are for quadrotor UAVs [69]. The adaptive vortex search algorithm is used for path planning [2]. Speed-up robust features algorithm is used in visual navigation [70]. The spline-RRT* algorithm is based on the RRT* algorithm and is the asymptotically optimal path planner for following the terrain [71].

Some algorithms consider the shortest path the best, but the best path is not always the shortest one but with minimal costs. Therefore, cost functions are developed to consider the complex characteristics and evaluate the generated paths. If a navigation algorithm can reach the near-optimal point quickly, the number of needed flight tasks will reduce [72]. The definition of the best path has been changed to the path with minimal costs of travelled distance, fuel consumption, and average altitude [67].

There are many cost functions that focus on the algorithms. The cost function of the collaborative search path planning algorithm concerns the distances, the search area boundaries, the number of discrete points, and the turn angle [73]. For the sparse bidirectional A* search algorithm, the cost function is related to the length of the specific edge, the threat cost, and the A* search cost [64].

Routing protocols imply metrics to evaluate the path, and a metric can be related to time used to communicate with the neighbours, path bandwidth, the distance between neighbours, and sensor capabilities [66]. The cost of the spline-RRT* algorithm is reduced if the number of vertices increases [71].

Moreover, the cost functions consider the properties of UAVs and paths. The energy of cruise steady, relative position change, airborne equipment, charging path, and mission energy are involved in a UAV energy model as the cost function [74]. The vertical climb and descent of the path segment are considered with the length of the path segment, and the climb phase spends more fuel while the descent phase spends less fuel [68].

The sum of the deviation of the corners and the height deviations are summed up for the cost function of the adaptive vortex search algorithm [2]. The developed cost function penalises the cost of longer paths, more power, more fuel, collision with the terrain, danger zones, higher average altitude, and unsmoothed paths [67].

More factors should be considered for generating the optimal path. For those UAVs that have missions, finding a path to achieve the required mission is the goal for UAV navigation. The critical points of minimising the cost can be linked to minimising the path length, flight time, fuel consumption, or risk of destruction [75].

For multiple UAVs, the path should be flyable and avoid threats so that the costs can be threat cost and fuel consumption based on UAV speed and path length [76]. The survivability of the UAV is measured by a single cost function related to the threat cost [76]. Path regulation, terrain threats, un-flyable zones and altitude constraints are defined as environmental factors to be calculated in the cost function [60].

Furthermore, there are many articles that support UAV navigation under GPS outages or in GPS-denied environments. The network-based method can be used for path planning for UAVs with obstacle avoidance with long-term GPS outages and uses landmark-based navigation [62]. The software framework and the instantaneous task specification using constraints methodology are developed to combine with shared control, object tracking and obstacle avoidance. It can be applied in uncertain dynamic environments [58].

Motion capture system/INS integrated navigation system is proposed for indoor UAV navigation with Kalman filter [77]. The RGB-D SLAM algorithm is proposed to improve the SLAM algorithm for indoor navigation [78]. The navigation problem is transferred to a Partially-observable Markov decision process for UAV uncertainty-based navigation [79].

Vision sensors are popular because the limited payload capability of small UAVs motivates to use the of vision sensors, which are lightweight, cheaper, and more flexible [80]. The guidance strategy with a forward-facing camera is proposed for UAV navigation, which considers computed optical flow [80].

Speed-up robust features algorithm is used in visual navigation [70]. The new self-adaptive methodology for UAV autonomous navigation is proposed based on computer vision [81]. This approach can change configuration based on the different environments, and a Bayesian network and a Multiplayer Perceptron were trained to classify the images.

The novel vision-aided navigation architecture is proposed for accurate UAV localization by aiding the inertial navigation system (INS) and referring to geographic information system (GIS) data [82]. For orchard-like environments, vision and altitude sensors are utilised in the flight system for vision-based UAV navigation [83]. Sensor-based odometry can be generated from the RGB camera for UAV navigation

to local the UAV, then plan the path by desired waypoints [84].

Synthetic Aperture Rader can aid UAV navigation, and the attitude and the position can be inferred by inspecting both phase and amplitude of Synthetic Aperture Rader images [85]. The object-oriented landmark recognition system is proposed for UAV navigation, and the keys to swift adaption are clear object-oriented coding on the landmarks, their properties, and constraints [86].

An optimal flow-based algorithm is designed as an artificial bee colony-based block-matching algorithm combined with an extended Kalman filter as a low-cost solution [87]. The aerodynamic model/INS/GPS, failure-tolerant navigation with Federated Kalman filter, is proposed for multirotor UAVs [88]. The primary navigation sensors are IMU and GPS for the current UAV system [89].

The new particle filter (PF) model is designed for real-time residual estimation, and the novel fuzzy inference system-based anomaly decision algorithm is used for prediction and precision. The Wireless Local Positioning System (WLPS) is a recent technology for measuring the position, and a Weighted Measurement Fusion Kalmen Filter is proposed, and each measurement of the process fusion process is weighted according to the signal travelling distance [90].

Additionally, the algorithms for radar-odometry with ultralight radar sensor has been provided with IMU data for sensor fusion for mini- and micro-UAV [91]. For UAV navigation in indoor corridor environments, the monocular vision-aided approach is proposed with vanishing points, scale-invariant vital points, and a Kalman filter [92]. Finding motion elements can be based on a UAV navigation approach with the optical flow to provide an additional way of navigation [93].

There are some other approaches for UAV navigation. A chaotic UWB-MIMO waveform design can be implemented to identify and avoid potential collision threats with a Dirichlet-process-mixture-model-based Bayesian clustering approach and a change-point detection algorithm [94]. GPS and IMU can collaborate with the Air Data system to raise the accuracy of UAV navigation, and the modified two-step adaptive Kalman filter is involved in solving navigation parameters [95].

2.2.3 Discussion of Path Planning Algorithms

The optimal path is considered the shortest path, so many graph search algorithms were applied as solutions, such as the A* algorithm, RRT searching, Dijkstra algorithm, and Bellman Ford's algorithm. Many papers propose improvements to the A* algorithm. Optimization frameworks also act as the solutions, such as genetic algorithms, ant colony optimization, and particle swarm optimization. Robots can be operated in several environments, including air and indoor, even in GPS-denied environments. There are many path-planning solutions, and most of the solutions implement the Kalman filter in the literature.

However, the shortest path is not the optimal path in many cases, so the costs of the paths are considered, and cost functions are developed to evaluate paths. Cost functions are proposed with specific algorithms or methods. Cost functions are usually based on the length of the path, the consumption of power and fuel, the threat and forbid zones, the collisions, and the properties of robots. From these cost functions, it can be concluded that the path's costs should not only consider the length of the path, and the best path should be the path with minimal costs.

2.3 Multi-robot Path Planning

Figure 2.1 presents the classification of multi-robot path-planning algorithms, and it is divided into four categories: classical approaches, heuristic algorithms, artificial intelligence (AI)-based and bio-inspired approaches. The subcategories are linked to the primary categories and only display the significant subcategories. The classical approaches include the Artificial potential field, sampling-based, and graph-based approaches. The heuristic algorithms mainly consist of A* and D* search algorithms. The AI-based approaches are the most common algorithms for multi-robot systems, and the bio-inspired approaches take most of the attention. Metaheuristic has been applied to most of the research, and the famous algorithms are PSO and GA. From [9], GA and PSO are the most commonly used approaches.

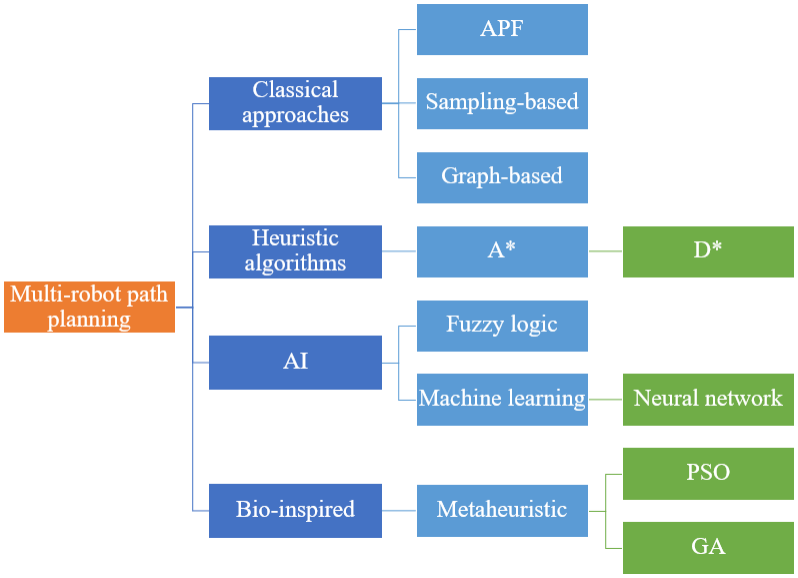


Figure 2.1: Classification of multi-robot path planning approaches.

2.3.1 Classical Approaches

Artificial Potential Field (APF)

The APF uses its control force for path planning, and the control force sums up the attractive and the repulsive potential field. The illustration of APF is shown in Figure 2.2; the blue force indicates the attractive field and the yellow force represents the repulsive field. The APF establishes path-planning optimization and dynamic particle models, and the additional control force updates the APF for multi-robot formation in a realistic and known environment [96]. Another APF-based approach is presented for a multi-robot system in a warehouse.

It uses the priority strategy and solves the drawbacks of traffic jams, local minima, collisions, and non-reachable targets [97]. An innovative APF algorithm is proposed to find all possible paths under a discrete girded environment. It implements a time-efficient deterministic scheme to obtain the initial path and then uses enhanced GA to improve it [98]. A potential field-based controller in [99] supports robots to follow the designed path, avoid collision with nearby robots, and distribute the robots stochastically across different paths in topologically distinct classes.

An improved APF is proposed to overcome the traditional APF's shortcomings, including target unreachable and a local minimum in [100] for real-time performance

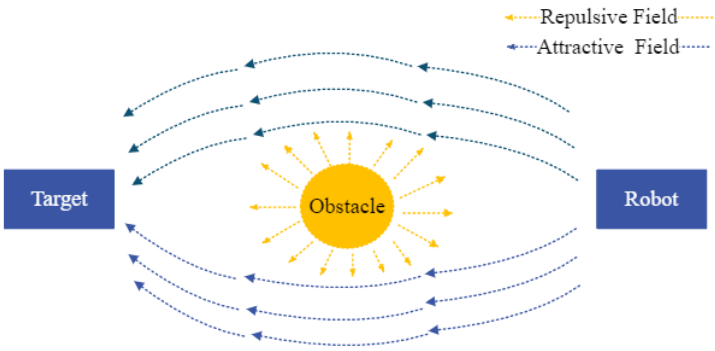


Figure 2.2: Illustration of the APF algorithm.

with dynamic obstacles for realizing local path planning. A collision avoidance strategy and risk assessment are proposed based on the improved APF and the fuzzy inference system for multi-robot path planning under a completely unknown environment [101].

APF is applied in the approximate cost function in [102], and integral reinforcement learning is developed for the minimum time-energy strategy in an unknown environment, converting the finite horizon problem with constraints to an infinite horizon optimal control problem. APF is introduced for the reward functions and integrates Deep Deterministic Policy Gradient and Model Predictive Control to address uncertain scenes [103].

Sampling-based

The rapidly exploring random tree (RRT) searches high-dimensional and nonconvex space by using a space-filling tree randomly, and the tree is built incrementally from samples to grow towards unreached areas. The sampling-based approach's outline is demonstrated in Figure 2.3, and the generated path is highlighted in green. For a multi-robot centralized approach, multi-robot path-planning RRT performs better in optimizing the solution and exploring search space in an urban environment than push and rotate, push and swap, and the Bibox algorithm [103]. The discrete-RRT extends the celebrated RRT algorithm in the discrete graph with a speedy exploration of the high-dimensional space of implicit roadmaps [104].

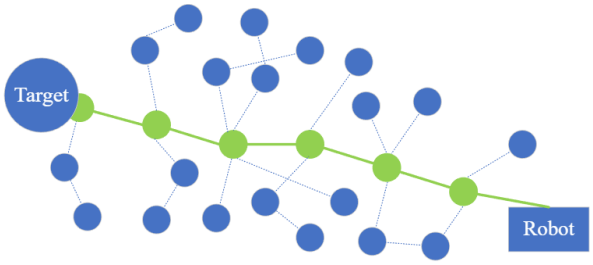


Figure 2.3: Demonstration of the RRT algorithm.

Other Classical Approaches

Tabu search keeps searching for the solutions in the neighbourhood and records the solutions in the Tabu list. The classic Tabu search is integrated with particle swarm optimization (PSO) to enhance optimization ability in [105], and it is aimed at the decision-making of routing and scheduling. It is based on the PSO and Tabu search algorithm with a "minimum ring" for obtaining the dynamic path planning for adapting the online requirements for a dynamic environment.

A polygon area decomposition strategy is applied to explore a target area with located waypoints. It analyzes the effect of the partition of the area, and the number of robots [106]. Planar graphs are used to solve optimal multi-robot path planning problems with computational complexity and establish the intractability of the problems on the graphs to reduce the sharing of paths in opposite directions [107]. The grid pattern map decomposition is developed for coverage path planning and employing multiple UAVs for collecting the images and creating a response map to obtain helpful information [108].

For remote sensing and area coverage with multi-robot, graph-based task modelling is proposed with mixed-integer linear programming to route the multiple robots [109]. A mixed-integer linear programming model is presented based on the hexagonal grid-based decomposition method [110]. It can be applied for multi-UAV coverage path planning in rescue and emergency operations. The biconnected graph, user input, and small critical benchmark are controlled by a path planner as presented in [111] to solve the multi-AGV path planning problems of AGV planetary exploration, automatic packages, and robotics mining.

A multi-robot informative path-planning approach transforms the continuous region

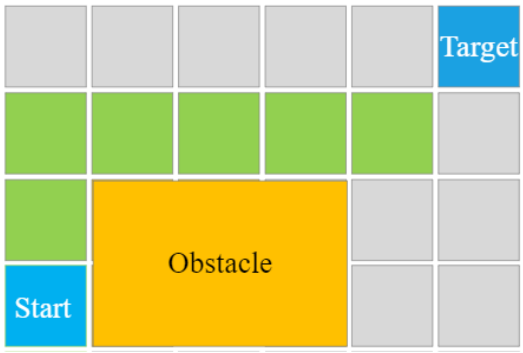


Figure 2.4: Simple example of the A* algorithm.

into Voronoi components, and the robots are allocated free regions [112]. The multi-robot navigation strategy with path priority is presented in [113]; a generalized Voronoi diagram divides the map according to the robot's path-priority order and finds the path-priority order for each robot.

For the cited papers, the classical approaches consist of APF and sampling-based algorithms. The classical algorithms usually involve a predefined graph, requiring high computational space. The trend of implementing the classical algorithms is combined with other state-of-art algorithms. The heuristic algorithms are proposed for complete and fast path planning.

2.3.2 Heuristic Algorithms

A* search

The A* search algorithm is one of the most common heuristic algorithms in path planning. Figure 2.4 shows the simple example of the gird-based A* algorithm, and the path is highlighted in green. It uses the heuristic cost to determine the optimal path on the map. The relaxed-A* is used to provide an optimal initial path and fast computation, and Bezier-splines are used for continuous path planning to optimize and control the curvature of the path and restrict the acceleration and velocity [28].

A two-level adaptive variable neighbourhood search algorithm is designed to be integrated with the A* search algorithm for the coupled mission planning framework. It models the path planning problem and the integrated sensor allocation to minimize travel costs and maximize the task profit [114]. For the multi-AGV routing problem,

the improved A* algorithm plans the global path and uses a dynamic RRT algorithm to obtain a passable local path with kinematic constraints, avoiding collisions in the grid map [43].

Additionally, [115] utilized the A* algorithm for the predicted path and generated a flyable path by cubic B-spline in real-time for guidance with triple-stage prediction. With the computational efficiency of cluster algorithms and A*, the proposed planning strategy supports online implementation. An optimal multi-robot path-planning approach is proposed with EA* algorithm with assignment techniques and fault-detection algorithm for the unknown environment based on the circle partitioning concept in [116]. A proposed navigation system integrates a modified A* algorithm, auction algorithm, and insertion heuristics to calculate the paths for multiple responders. It supports connection with a geo-database, information collection, path generation in dynamic environments, and spatiotemporal data analysis [117].

The D* algorithm is employed for multi-robot symbiotic navigation in a knowledge-sharing mechanism with sensors [19]. It allows robots to inform other robots about environmental changes, such as new static obstacles and path blockage, and it can be extended for real-time mobile applications. Additionally, D* Lite is applied with artificial untraversable vertex to avoid deadlocks and collisions for real-time robot applications, and D* Lite has fast re-planning abilities [20].

A cloud approach is developed with D* Lite and multi-criteria decision marking to offer powerful processing capabilities and shift computation load to the cloud from robots in the multi-robot system with a high level of autonomy [118]. An integrated framework is proposed based on D* Lite, A*, and uniform cost search, and it is used for multi-robot dynamic path-planning algorithms with concurrent and real-time movement [119].

Other Heuristic Algorithms

A constructive heuristic approach is presented to perceive multiple regions of interest. It aims to find the robot's path with minimal cost and cover target regions with heterogeneous multi-robot settings [17]. Conflict-Based Search is proposed for multi-agent path planning problems in the train routing problem for schedul-

ing multiple vehicles and setting paths in [120]. For multi-robot transportation, a primal-dual-based heuristic is designed to solve the path planning problem as the multiple heterogeneous asymmetric Hamiltonian path problem, solving in a short time [121]. The linear temporal logic formula is applied to solve the multi-robot path planning by satisfying a high-level mission specification with Dijkstra's algorithm in [122].

A modified Dijkstra's algorithm is introduced for robot global path planning without intersections, using a quasi-Newton interior point solver to smooth local paths in tight spaces [123]. Moreover, cognitive adaptive optimization is developed with transformed optimization criteria for adaptively offering the accurate approximation of paths in the proposed real-time reactive system; it takes into account the unknown operation area and nonlinear characteristics of sensors [29].

The Grid Blocking Degree (GBD) is integrated with priority rules for multi-AGV path planning, and it can generate a conflict-free path for AGV to handle tasks and update the path based on real-time traffic congestion to overcome the problems caused by most multi-AGV path planning offline scheduling [124]. Heuristic algorithms, minimization techniques, and linear sum assignment are used in [125] for multi-UAV coverage path and task planning with RGB and thermal cameras. [126] designed the extended Angular-Rate-Constrained-Theta* for a multi-agent path-planning approach to maintaining the formation in a leader-follower formation.

Figure 2.5 displays the overview of the mentioned heuristic algorithms. The heuristic algorithms are widely used in path planning, and the heuristic cost functions are developed to evaluate the paths. The algorithms can provide the complete path in a grid-like map. However, for the requirement of flexibility and robustness, bio-inspired algorithms are proposed.

2.3.3 Bio-Inspired Techniques

Particle Swarm Optimization (PSO)

PSO is one of the most common metaheuristic algorithms in multi-robot path planning problems and formation. The flowchart of PSO is shown in Figure 2.6. It is a stochastic optimization algorithm based on the social behaviour of animals,

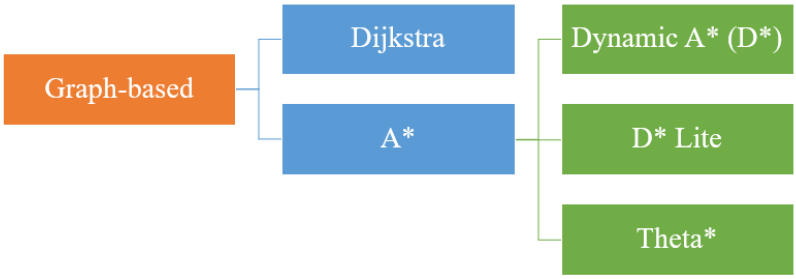


Figure 2.5: Search algorithms.

and it obtains global and local search abilities by maintaining a balance between exploitation and exploration [127]. [128] presents an interval multi-objective PSO using an ingenious interval update law for updating the global best position and the crowding distance of risk degree interval for the particle's local best position. PSO is employed for multiple vehicle path planning to minimize the mission time, and the path planning problem is formulated as a multi-constrained optimization problem [129], while the approach has low scalability and execution ability.

An improved PSO is developed with differentially perturbed velocity, focusing on minimizing the maximum path length and arrival time with a multi-objective optimization problem [130]. The time stamp segmentation model handles the coordination cost. Improved PSO is combined with modified symbiotic organisms searching for multi-UAV path planning, using a B-spline curve to smooth the path in [131]. For a non-stationary environment, improved PSO and invasive weed optimization are hybrids for planning a path for each robot in the multi-robot system, balancing diversification and intensification, and avoiding local minima [132].

PSO is adapted for a leader-follower strategy in multi-UAV path planning with obstacle avoidance [127]. A distributed cooperative PSO is proposed for obtaining a safe and flyable path for a multi-UAV system, and it is combined with an elite keeping strategy and the Pythagorean hodograph curve to satisfy the kinematic constraints [133]. The enhanced PSO is improved by greedy strategy and democratic rule in human society inspired by sine and cosine algorithms. The projected algorithm can generate a deadlock-free path with preserving a balance between intensification and diversification [134].

For the multi-robot path planning issue, a coevolution-based PSO is proposed to ad-

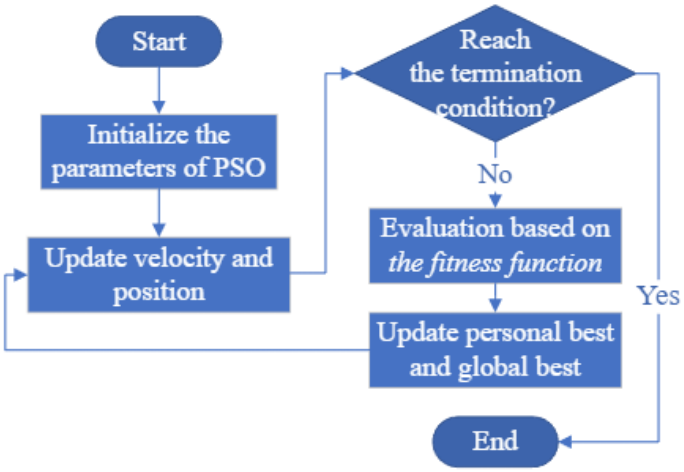


Figure 2.6: Flowchart of the PSO algorithm.

just the local and goal search abilities and solve the stagnation problem of PSO with evolutionary game theory [135]. An improved gravitational search algorithm is integrated with the improved PSO for a new methodology for multi-robot path planning in the clutter environment, and it updates the particle positions and gravitational search algorithm acceleration with PSO velocity simultaneously [136].

A hybrid algorithm of democratic robotics PSO and improved Q-learning is proposed to balance exploitation and exploration, and it is fast and available for a real-time environment. However, it cannot guarantee the completeness of the path, and it is hard to achieve robot cooperation [137]. PSO-based and a B-Spline data frame solver engine is developed for uninterrupted collision-free path planning. It is robust to deal with current disturbances and irregular operations and provides quick obstacle avoidance for real-time implementation [26].

A wireless sensor network is presented for locating obstacles and robots in a dynamic environment. It combines a jumping mechanism PSO algorithm and a safety gap obstacle avoidance algorithm for multi-robot path planning [18]. The jumping mechanism PSO estimates the inertia weight based on fitness value and updates the particles. The safety gap obstacle avoidance algorithm focuses on robots struck when avoiding obstacles. [138] designed the hybrid GA and PSO with fuzzy logic controller for multi-AGV conflict-free path planning with rail-mounted gantry and quay cranes; however, it is inapplicable to real-time scheduling.

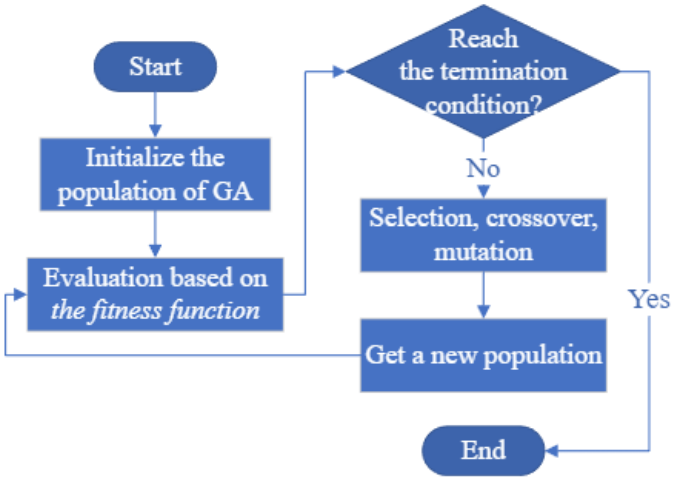


Figure 2.7: Flowchart of the GA algorithm.

Genetic Algorithm (GA)

GA is widely utilized for solving optimization problems as an adaptive search technique, and it is based on a genetic reproduction mechanism, and natural selection [139]. The flowchart of GA is indicated in Figure 2.7. [140] uses GA and reinforcement learning techniques for multi-UAV path planning, considers the number of vehicles and a response time, and a heuristic allocation algorithm for ground vehicles. GA solves the Multiple Traveling Sales Person problems with the stop criterion and the cost function of Euclidean distance, and Dubins curves achieve geometric continuity while the proposed algorithm cannot avoid the inter-robot collision or support online implementation [27].

A 3D sensing model and a cube-based environment model are involved in describing a complex environment, and non-dominated sorting GA is modified to improve the convergence speed for the Pareto solution by building a voyage cost map by the R-Dijkstra algorithm in [141] as an omnidirectional perception model for multi-robot path planning. [142] applies the sensors in the area to obtain a minimal cost and solves the travelling salesman, and GA is adapted for persistent cooperative coverage.

Efficient genetic operators are developed to generate valid solutions on a closed metric graph in a reasonable time and are designed for multi-objective GA for multi-agent systems [143]. GA assigns the regions to each robot, sets the visiting orders,

and uses simultaneous localization and mapping to create the global map in [144] for coverage path planning. [145] presents GA to optimize the integration of motion patterns that represent the priority of the neighbour cell and divide the target environment into cell areas, and then use a double-layer strategy to guarantee complete coverage.

A domain knowledge-based operator is proposed to improve GA by obtaining the elite set of chromosomes, and the proposed algorithm can support robots with multiple targets [146]. For intelligent production systems, the improved GA is aimed at complicated multi-AGV path planning and maneuvering scheduling decisions with time-dependent and time-independent variables. It first addresses AGV resource allocation and transportation tasks and then solves the transportation scheduling problem [147].

An improved GA was presented with three-exchange crossover heuristic operators than the traditional two-exchange operators, which consider double-path constraints for multi-AGV path planning [148]. [139] proposed a boundary node method with a GA for finding the shortest collision-free path for a 2D multi-robot system and using a path enhancement method to reduce the initial path length. Due to the short computational time, it can be used for real-time navigation, while it can only be implemented in a known environment without dynamic obstacles.

A high degree of GA is employed for optimal path planning under a static environment at offline scheduling, and online scheduling is aimed to solve conflicts between AGVs for the two-stage multi-AGV system [149]. The evolution algorithm is used for planning a real-time path for multi-robot cooperative path planning with a unique chromosome coding method, redefining mutation and crossover operator in [150].

Ant Colony Optimization (ACO)

Ants will move along the paths and avoid obstacles, marking available paths with pheromones, and the ACO treats the path with higher pheromones as the optimal path. The principle of ACO is demonstrated in Figure 2.8, and the path with a higher pheromone is defined as the optimal path marked by green. For collision-free routing and job-shop scheduling problems, an improved ant colony algorithm is

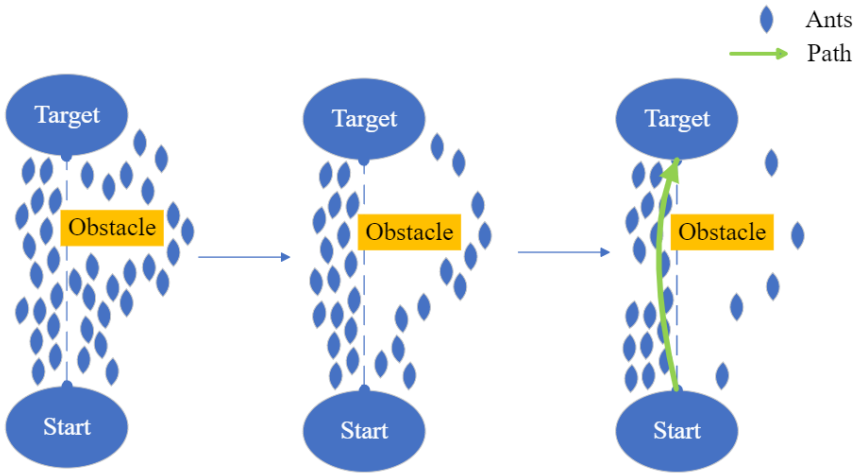


Figure 2.8: Changes of the ACO algorithm with different timeslots.

enhanced by multi-objective programming for a multi-AGV system [151].

For multi-UGVs, a continuous ACO-based path planner focuses on coordination and path planning. It is integrated with an adaptive waypoint-repair method and a probability-based random-walk strategy to balance exploration and exploitation and improve the algorithm's performance, resolving the coordination with a velocity-shifting optimization algorithm [152].

K-degree smoothing and the improved ACO are integrated as a coordinated path planning strategy for the multi-UAV control and precise coordination strategy in [153]. Voronoi models the environment by considering various threats, and the improved ACO's pheromone update method and heuristic information are redefined for path planning, then using a k-degree smoothing method for the path smoothing problem. For precision agriculture and agricultural processes, ACO, Bellman-Held-Karp, Christofides, and Nearest Neighbor based on K-means clustering are used for the optimization path of multi-UAV [154].

Pigeon-Inspired Optimization (PIO)

Pigeon navigation tools inspired PIO, and it uses two operators for evaluating the solutions. Social-class PIO is proposed to improve the performances and convergence capabilities of standard PIO with inspiring by the inherent social-class character of pigeons [155], and it is combined with time stamp segmentation for multi-UAV path planning. [156] analyzing and comparing the changing trend of fitness value of local

and global optimum positions to improve the PIO algorithm as Cauchy mutant PIO method, and the plateau topography and wind field, control constraints of UAVs are modelled for cooperative strategy and better robustness.

Grey Wolf Optimizer (GWO)

GWO is inspired by the hunting behaviour and leadership of grey wolves, and it obtains the solutions by searching, encircling, and attacking prey. An improved grey wolf optimizer is employed for the multi-constraint objective optimization model for multi-UAV collaboration under the confrontation environment. It considers fuel consumption, space, and time [157]. The improvements of the grey wolf optimizer are individual position updating, population initialization, and decay factor updating. An improved hybrid grey wolf optimizer is proposed with a whale optimizer algorithm in a leader-follower formation and fuses a dynamic window approach to avoid dynamic obstacles [158].

The leader-follower formation controls the followers to track their virtual robots based on the leader's position and considers the maximum angular and linear speed of robots. [159] proposed a hybrid discrete GWO to overcome the weakness of traditional GWO, and it updates the grey wolf position vector to gain solution diversity with faster convergence in discrete domains for multi-UAV path planning, using greedy algorithms and the integer coding to convert between discrete problem space and the grey wolf space.

Other Bio-Inspired Techniques

The fruit fly optimization approach usually solves the nonlinear optimization problem. The multiple swarm fruit optimization algorithm is presented for the coordinated path planning for multi-UAVs, and it improves the global convergence speed and reduces the possibilities of local optimum [160].

An improved gravitational search algorithm is proposed for multi-robot path planning under the dynamic environment based on a cognitive factor, social, and memory information of PSO, and deciding the population for the next generation based on greedy strategy [161].

The simulated annealing is integrated with the Dijkstra algorithm for calculating the optimal path based on the Boolean formula and the global map for a high-level specification for multi-robot path planning [24]. The hybrid algorithm of sine-cosine and kidney-inspired is developed for multi-robot in a complex environment. It selects the optimal positions for each robot to avoid conflicts with teammates and dynamic obstacles [162].

The hybridization of invasive weed optimization and firefly algorithm is employed to adjust the movement property of the firefly algorithm and spatial dispersion property of invasive weed optimization for exploration and exploitation [163]. The Differential Evolution algorithm tunes the differential weight, population size, generation number, and crossover for multi-UAV path planning in [164]. It defines the minimum generation's weighting required between the computational and the path cost.

Physarum is a bio-inspired method for path planning, and it can take a quick response to external change. [23] proposed a Physarum-based algorithm for multi-AGV for model-based mission planning in dynamic environments with an adaptive surrogate modelling method. A novel swarm intelligence algorithm is developed as an Anas platyrhynchos optimizer for multi-UAV cooperative path planning.

The Anas platyrhynchos optimizer simulates the swarm's moving process and warning behaviour [165]. It proposes low-communication cooperation and heterogeneous strategies for online path planning based on differential evolution-based path planners [166]. It summarizes local measurements with the sparse variation Gaussian process, sharing information even in a weak communication environment. [167] developed a multi-task multi-robot framework for challenging industrial problems. It proposes Large Neighbor Search as a new coupled method to make task assignment choices by actual delivery costs.

The artificial immune network algorithm is improved with the position tracking control method for providing the abilities of diversity and self-recognition for multi-robot formation path planning with leader robots, and it overcomes the shortcomings of immature convergence and local minima [168]. Differential evolution algorithm is improved in [169] for calculating collision-free optimal path with multiple dynamic

obstacle constraints in a 2D map. An efficient artificial bee colony algorithm is proposed for online path planning, selecting the appropriate objective function for collision avoidance, target, and obstacles [170].

Bio-inspired techniques mainly include PSO, GA, ACO, PIO, and GWO. They are inspired by animals' natural behaviours and employ particles for path generation. As for computational efficiency and powerful implementation, they are popular in multi-robot path planning. AI-based approaches are proposed due to the development of intelligent systems and the requirements of adapting to changing environments.

2.3.4 Artificial Intelligence

Fuzzy Logic

Fuzzy logic uses the principle of "degree of truth" for computing the solutions. It can be applied for controlling the robot without the mathematical model but cannot predict the stochastic uncertainty in advance. As a result, a probabilistic neuro-fuzzy model is proposed with two fuzzy level controllers and an adaptive neuro-fuzzy inference system for multi-robot path planning and eliminating the stochastic uncertainties with leader-follower coordination [171]. The fuzzy C-means or the K-means methods filter and sort the camera location points, then use A* as a path optimization process for the multi-UAV travelling salesman problem in [16].

For collision avoidance and autonomous mobile robot navigation, Fuzzy-wind-driven optimization and a singleton type-1 fuzzy logic system controller are hybrids in the unknown environment [172]. The wind-driven optimization algorithm optimizes the function parameters for the fuzzy controller, and the controller controls the motion velocity of the robot by sensory data interpretation. [173] proposed a reverse auction-based method and a fuzzy-based optimum path planning for multi-robot task allocation with the lowest path cost.

Machine Learning

Machine learning simulates the learning behaviour to obtain the solutions. It is used for path planning, embracing mobile computing, hyperspectral sensing, and rapid telecommunication for the rapid agent-based robust system [174]. Kernel

smooth techniques, reinforcement learning, and the neural network are integrated for greedy actions for multi-agent path planning in an unknown environment [21] to overcome the shortcomings of traditional reinforcement learning, such as high time consumption, slow learning speed, and disabilities of learning in an unknown environment.

The self-organizing neural network has self-learning abilities and competitive characteristics for the multi-robot system's path planning and task assignment. [175] combined it with Glasius Bio-inspired neural network for obstacle avoidance and speed jump while the environment changes have not been considered in this approach. The biological-inspired self-organizing map is combined with a velocity synthesis algorithm for multi-robot path planning and task assignment. The self-organizing neural network supports a set of robots to reach multiple target locations and avoid obstacles autonomously for each robot with updating weights of the winner by the neurodynamic model [176].

Convolution Neural networks analyze image information to find the exact situation in the environment, and Deep q learning achieves robot navigation in a noble multi-robot path-planning algorithm [177]. This algorithm learns the mutual influence of robots to compensate for the drawback of conventional path-planning algorithms. In an unknown environment, a bio-inspired neural network is developed with the negotiation method, and each neuron has a one-to-one correspondence with the position of the grid map [178]. A biologically inspired neural network map is presented for task assignment and path planning, and it is used to calculate the activity values of robots in the maps of each target and select the winner with the highest activity value, then perform path planning [179]. The simple neural network diagram is exhibited in the following Figure 2.9.

Moreover, a multi-agent path-planning algorithm based on deep reinforcement learning is proposed, providing high efficiency [180]. Another multi-agent reinforcement learning is developed in [181], and it constructs a node network and establishes an integer programming model to extract the shortest path. The improved Q-learning plans the collision-free path for a single robot in a static environment and then uses the algorithm to achieve collision-free motion among robots based on prior knowl-

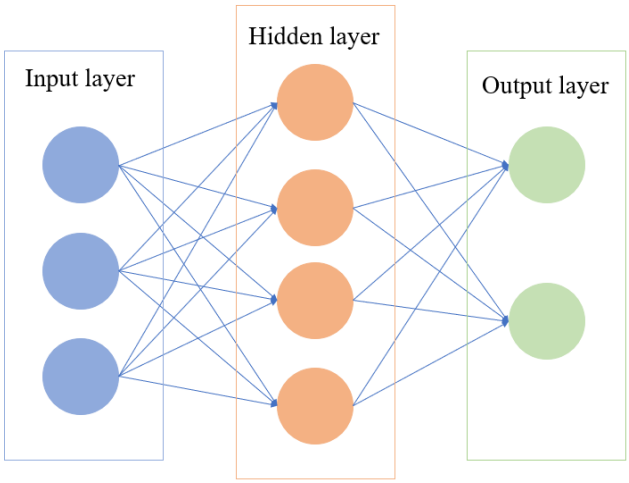


Figure 2.9: Diagram of a three-layer neural network.

edge [182]. The reinforcement learning framework is applied to optimize the quality of service and path planning, describe the users' requirements, and consider geometric distance and risk by reinforcement learning reward matrix with a sigmoid-like function [183].

An attention neural network is used for generating the multimachine collaborative path planning as attention reinforcement learning, and it can meet high real-time requirements [184]. A deep Q-network is implemented with a Q-learning algorithm in a deep reinforcement learning algorithm for a productive neural network to handle multi-robot path planning with faster convergence [185]. The meta-reinforcement learning is designed based on transfer learning [186], and it improves proximal policy optimization by covariance matrix adaptation evolutionary strategies to avoid static and dynamic obstacles.

Multi-agent reinforcement learning is improved by an iterative single-head attention mechanism for multi-UAV path planning, and it calculates robot interactions for each UAV's control decision-making [187]. Fuzzy reinforcement learning is proposed for the continuous-time path-planning algorithm, combining a modified Wolf-PH and fuzzy Q-iteration algorithm for cooperative tasks [188].

2.3.5 Others

The algorithms based on mathematical principles or other unclassified systems are listed in this session. These algorithms are not typically classified into classical, heuristic, bio-inspired, and AI-based approaches.

A multi-robot path planning system is developed with Polynomial-Time for solutions with optimality constant-factor [25], and it provides efficient implementations and adapted routing subroutines. A multi-robot path-planning algorithm for industrial robots is presented based on the first low polynomial-time algorithm on grids [189]. An innovative method based on Fast Marching Square is proposed in [190] for simple priority-based speed control, the planning phase, and conflict resolution in 3D urban environments. The fast Marching Square algorithm is also used in a triangular deformable leader-follower formation for multi-UAV coverage path planning [191].

[192] combined polynomial time with Push and spin algorithm for multi-robot path-planning algorithm and enhances the performance of choosing the best path. A first low-polynomial running time algorithm is proposed for multi-robot path planning in grid-like environments and solves average overall problem instances by constant factors make-span optimal solutions [193]. For optimal multi-robot coverage path planning, spanning tree coverage is proposed, and it divides the surface into many equal areas for each robot to guarantee minimum coverage path, complete coverage, and a non-backtracking solution [194].

For multi-UAV coverage path planning, a metric Cartesian system is proposed, and it transforms the coordinates into Cartesian and splits the field to assign to each robot, then forms the path with minimizing the time [4]. Probability Hypothesis Density representation is used to optimize the number of observed objects in multi-agent informed path planning, and it can represent unseen objects [195]. An iterative min–max improvement algorithm is designed to make span-minimized multi-agent path planning to solve the constrained optimization problem using a local search approach in discrete space [196].

The new route-based optimization model is presented for multi-UAV coverage path planning with column generation, and it can generate feasible paths and trace energy required for mission phases [197]. A multi-agent collaborative path-planning algo-

rithm is provided in [198] to guarantee complete area coverage and exploration and use a staying alive policy to consider battery charge level limitation in the indoor environment.

Integer linear programming models the path planning problem for three objectives with the task due times, including minimizing total unit penalties, tardiness, and maximum lateness [99]. Integer linear programming solves the multi-robot association path planning problem for optimizing the path and robots' access points associations in industrial scenarios [199]. For finding the optimal path for robots to perform tasks, the optimal problem is transformed into integer linear programming with the Petri net model in [200]. One-way multi-robot path planning is proposed for the warehouse-like environment, and it is based on Integer programming to reduce the robots' configuration costs [201]. A mixed-integer linear programming formulation is designed for multi-robot discrete path planning, and it extends the single-robot decision model to multi-robot settings with anticipated feedback data [15]. It supports real-time action based on modelling extension.

For multi-agent navigation, the reciprocal velocity obstacles (RVO) model is used for collision detection and prevention and uses an agent-based high-level path planner [202]. A cooperative cloud robotics architecture is developed as a cooperative data fusion system to gather data from various sensing sources and renew the global view to extend the field of view for each AGV in the industrial environment and uses flexible global and local path planning to avoid unexpected obstacles, and congestion zones [13].

The hybrid approach is presented in [6] based on the improved Interfered Fluid Dynamical System and the Lyapunov Guidance Vector Field for multi-UAV cooperative path planning. It introduces a vertical component for target tracking and uses the improved Interfered Fluid Dynamical System to resolve local minimum problems and avoid obstacles. Cooperative sensing and path planning for multi-vehicle is transformed as a partially observable decision-making problem, and it uses Markov decision processes as a decision policy and deploys a multi-vehicle communication framework [203].

2.3.6 Discussion of Multi-robot Path Planning Classification

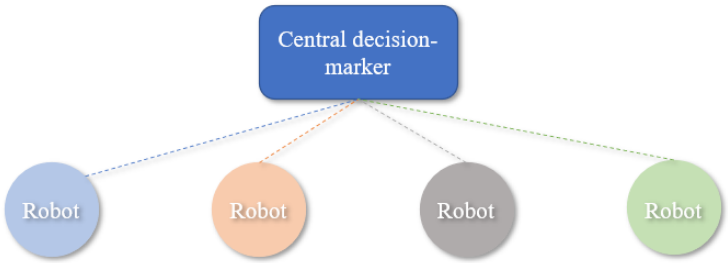
The classical approaches include APF and sampling-based algorithms, such as RRT. The classical techniques usually require more computational time and space, especially for the sampling-based methods. Furthermore, the classical techniques cannot ensure completeness or capability, and it requires a predefined graph and is difficult for them to re-plan the path during implementation.

A* and dynamic A* (D*) algorithms are standard applications for heuristic algorithms. The heuristic algorithms primarily consist of the graph search algorithm, and they are easy to apply for path planning problems and evaluate the path by the developed cost function. The heuristic algorithms can successfully provide the globally optimal path with lower required runtime and space than the classical approaches in a graph.

The bio-inspired approaches have been widely researched in recent years as the primary algorithms used in multi-robot path planning, especially metaheuristic algorithms. This chapter discusses PSO, GA, ACO, PIO, and GWO. They are inspired by nature, such as the social behaviour of animals. The bio-inspired approaches use various particles to generate the optimal solution for the defined problem.

The AI-based approaches based on fuzzy logic or machine learning have gained more attention recently. They have fast computation abilities, and the models are usually adapted for online path planning. The AI-based strategies learn from the previous data to train the models. The neural network is the primary application of machine learning for multi-robot path planning, which consists of multiple layers for learning. The detailed analysis refers to Section 2.5.2.

Path planning is part of the multi-robot system's consideration, and the structure of the multi-robot system can be classified as centralized or decentralized based on the planner. The multi-robot system is centralized if the system has supervisory control or a central planner. For robots making their decisions, the system is decentralized. The details of the decision-making of the multi-robot system refer to Section 2.4.



(a) Structure of a centralized framework

(b) Structure of a decentralized framework

Figure 2.10: Structure of decision-making framework

2.4 Decision-making

A Multi-robot system can be a centralized or decentralized structure. A centralized system is controlled by the central decision-maker, while a decentralized multi-robot system has no supervisory control. Figure 2.10 exhibits a centralized framework that has more vital fault-tolerant ability while poor global ability and a decentralized framework in which robots use the neighbours' information.

2.4.1 Centralized

A centralized framework for an industrial robot is proposed in [204], which combines GA and A* algorithms for 2D multi-robot path planning. GA is utilized for task allocation, and the A* algorithm is for path planning, and this approach addresses collision avoidance. A two-stage centralized framework solves multi-agent pickup and delivery problems, and it achieves path and action planning with orientation under non-uniform environments by heuristic algorithms, detecting and resolving conflicts by a synchronized block of information [205].

A practical centralized framework is developed based on an integer linear programming model, and it operates time expansion in the discrete roadmap to obtain the space-time model with dived and conquer heuristic and reachability analysis [30].

In grid graphs, a centralized and decoupled algorithm is proposed for multi-robot path planning in automated and on-demand warehouse-like settings, and it explores optimal sub-problem solutions and path diversification databases for resolving local path conflicts [206]. It uses a decoupling-based planner with two heuristic attack phases and goal configuration adjustments.

[207] uses a centralized controller for multi-target multi-sensor tracking for environmental data acquisition for path planning and feedback control for sending the path to the system. The optimal bid valuation is proposed with the Dijkstra algorithm to find the shortest path, and the proposed centralized model supports an alternative sampling-based method to reduce the computation time with achieving optimality [31].

A self-organizing map is used for data collection tasks and active perception for online multi-robot path planning, and it jointly picks and allocates nodes and finds sequences of sensing positions [208]. A mixed-integer programming formulation is adapted for a discrete centralized multi-agent path planning problem, and a two-phase fuzzy programming technique gains the Pareto optimal solution in [209].

The centralized simultaneous inform and connect (SIC) strategy is applied for multi-objective path planning by GA, and it uses SIC to optimize search, communicate and find the best path, and monitor tasks with quality of service [210]. A developed synthesized A* algorithm is used for path planning through a centralized meta-planner based on Bag of Tasks, and it runs on distributed computing platforms to avoid dynamic obstacles [211].

A wireless network is proposed for commutation among the robots in APF links, and it uses the Software Defined Network technique to update the network architecture and employ the topology and APF to establish a network control model [212]. A centralized architecture has a high degree of coordination, while dynamic and real-time actions are weak [213]. The decentralized structure is proposed to overcome the drawback of the centralized structure, thereby, providing a higher level of flexibility.

2.4.2 Decentralized

Task assignment for multi-robot is essential during path planning. The decentralized heuristic path-planning algorithm is proposed as Space utilization optimization for multi-robot structures, and it reduces computation time and the number of conflicts to gain the solution for one-shot and life-long problems [214]. An offline time-independent approach is developed with deadlock-based search and conflict-based search to assign the path to each robot when agents cannot share information [215].

The distributed multi-UAV system utilizes an insertion-based waypoint for path planning and its reconfiguration in [216]. The roadmap algorithm receives near-optimal paths in a decentralized coordination strategy to maximize connectivity and redundancy, while the global path planning utilizes shared information for the proposed two-layer control architecture [217].

The coordinated locomotion of a multi-robot system is divided into sub-problems, such as homogenous prioritized multi-robot path planning and task planning, and it uses prioritized reinforcement learning for these problems [33]. For a swarm of UAVs, PSO is adapted as a planner for distributed full coverage path planning in a dynamic and stochastic environment, minimizing the cost function and maximizing the fitness function [14].

The enhanced A* algorithm referred to as the MAPP algorithm, is delivered in [218] as the decentralized planner for task assignment and cooperative path planning for multi-UAV in urban environments. Free-ranging motion scheme is implemented in autonomous multi-AGV path planning and motion coordination. It considers non-holonomic vehicle constraints for path planning and reliable detection and resolution of conflicts for motion coordination based on a priority scheme [219].

A sampling-based motion planning paradigm is developed for decentralized multi-robot belief space planning in an unknown environment for high-dimensional state spaces in [32], and it calculates the utility of each path based on incremental smoothing of efficient inference and insights from the factor graph. A fully completed distributed algorithm is developed for considering plan restructuring, individual path planning, and priority decision-making for a distributed multi-agent system in [220].

Graph search algorithm and APF are mixed for multi-robot delivery service in different environments, and it uses a strongly connected digraph to simplify the path planning problem and use APF to prove flexibly [221]. A cluster-based decentralized task assignment is proposed for real-time missions [115]. It generates a path, assigns tasks for each robot in the initial planning stage, and adds the popup tasks to the task list to be considered in the next planning stage. A novel smooth hypocycloidal path is developed for multi-robot motion planning with local communication, and it maintains safe clearances with obstacles [222].

A multi-agent distributed framework formulates the path planning problem as a centralized linear program and then uses a framework for each agent while only communicating with its neighbours as the distributed algorithms [223]. The proposed model in [224] integrates decision-making policies and local communication for multi-robot navigation in constrained workspaces, and it uses a convolutional neural network to extract features from observations with a graph neural network to achieve robot communication.

A localized path planning and a task allocation module are combined into a decentralized task and path planning framework, and it models each task as a mixed observed Markov Decision Process or Markov Decision Process, using the max-sum algorithm for task allocation and the localized forward dynamic programming scheme for conflict resolution [225]. A Graph Neural Network is utilized to combine with a key-query-like mechanism to evaluate the relative importance of messages and learn communication policies in a decentralized multi-robot system [226].

The path planning problem is formulated as a decentralized partially observable Markov decision process in [227], and the multi-agent reinforcement learning approach is proposed for multi-robot path planning to harvest data from distributed end devices. It can support non-communicating, cooperative, and homogenous UAVs, and the control policy can be used for challenging urban environments without prior knowledge.

A genetic programming approach is proposed in a decentralized framework, and the robots conduct the learning program to determine the following action in real-time until they reach their respective destinations [228]. A decentralized multi-

robot altruistic coordination is improved for cooperative path planning and resolves deadlock situations [229]. APF is adapted in a proposed decentralized space-based potential field algorithm for a group of robots to explore an area quickly and connect with the team by dispersion strategy, using a monotonic coverage factor for a map exchange protocol, avoiding minima and realistic sensor bounds [230].

Another study [231] proposed APF with the notion of priority, the neighbourhood system, and the non-minimum speed algorithm to resolve the intersection of robots and minimum local problems for the multi-robot system. The multi-agent Rapidly Exploring Pseudo-random Tree is developed for real-time multi-robot motion planning and control based on the classical Probabilistic Road Map (PRM) algorithm. It extends PRM as a deterministic planner with probabilistic completeness, simplicity, and fast convergence [232].

2.4.3 Discussion of Decision-making Strategies

The centralized framework has higher control abilities for robots, and the actions are directly sent from the centre controller to the robots, making decisions for each robot. It provides better support and task assignment scheduling, and the algorithms applied in the centralized framework have no restrictions. The cited papers use the classical approaches, the heuristic algorithms, bio-inspired, and AI-based techniques for the centralized framework, in particular, the heuristic algorithms.

However, centralized frameworks are weak in dynamic applications. The decentralized structure is proposed to overcome the drawbacks of the centralized frameworks, and it makes robots communicate with others and share information. The algorithms used in the decentralized structure involve heuristic algorithms, optimization metaheuristic algorithms, neural networks, APF, sampling-based approaches, and AI-based algorithms. In this thesis, the combination of centralized and decentralized frameworks as a dual-layer model is proposed to provide flexibility for the multi-robot system. It enhances the robot's communication with others and the central controller for decision-making. For more discussion of decision-making strategies, refer to Section 2.5.3.

2.5 Discussion and Conclusions

2.5.1 Path Planning Algorithms

Robot path planning is essential because it provides the optimal and safe path for autonomous navigation. The environment, the navigation ability, the obstacles, and the properties of robots and paths are considered during path planning. Therefore, it is a complicated research problem. The environment data is discretized into meaningful data for processing path planning to generate the paths. Graph search algorithms and improved bio-inspired frameworks can be applied for path planning because graph search algorithms guarantee path optimality, and bio-inspired frameworks provide fast calculation. The cost functions are developed with these algorithms to evaluate the path with minimal costs to determine the optimal path. Table 2.1 compares navigation methodologies.

Table 2.1: Comparison of navigation methodologies

Papers	**Methodologies**	**Sensors**
[57], [63], [66]	A* search	
[63], [64]	Sparse A* search algorithm	
[59], [62]	The network-based method	
[65], [66]	Mixed integer linear programs and graph search algorithm	
[67], [68], [233]	Bio-inspired optimization framework	
[2]	The adaptive vortex search algorithm	
[70]	Speed-up robust features algorithm	
[71]	The spline-RRT* algorithm	
[87], [88]	Kalman filter	
[87]	Artificial bee colony-based block-matching algorithm	

[56]	Kalman filter	The encoder, gyro, acceleration sensor, ultrasonic sensor and infrared sensor
[40]	A*	SLAMTEC Laser Scanner
[53]	Kalman filter, Dead Reckoning	GPS
[55]	The Derivative-free Extended Information Filter (DEIF)	IMU, GPS, gyrocompasses, laser
[37]	Deep Reinforcement Learning	Cameras, GPS, speed sensors
[49]	PID	Ultrasonic, Lidar, RFID
[45]	SLAM	IMU, laser scan
[52]	Extended Kalman filter filters, optimal weighted voting fusion method	Rotating INS, GNSS
[44]	A* and Optimized Strategy	Odometry, an RGB-D sensor
[51]	Topological map	Vision, RFID
[234]	The Prioritized Coordination (PC) algorithm	
[38]	Nonlinear model predictive control formulation	LiDAR
[3]	A*, Fuzzy-based genetic algorithm	
[48]	Interactive multiple model (IMM) algorithm and Kalman filter	INS, UWB
[148]	Genetic algorithm	
[43]	The dynamic RRT algorithm	

The classical approaches are widely implemented for single robot path planning, especially the A* algorithm. Bio-inspired and mathematical-based techniques are

also employed. For gaining data from the environment, sensors are utilized. Sensor fusion is typical for overcoming the drawbacks of one sensor, and the Kalman filter is the classical solution for sensor fusion [52], [56]. INS and vision-based sensors are popular during operation because they can operate in the GPS-denied area [37], [48]. The future directions of the single robot path planning research should regard real-time planning and develop the cost functions, which can be generally applied to many algorithms or methods. The concerns of real-time path planning consider dynamic environments, path re-planning and computational efficiency.

2.5.2 Multi-robot Path Planning

From the literature, the multi-robot path-planning approaches are classified into four primary categories: classical approaches, heuristic approaches, bio-inspired techniques, and artificial intelligence-based approaches. Table 2.2 summarizes the main algorithms used in the categories, focusing on real-time implementation. The offline executions occupy 62% of the multi-robot path-planning approaches, and real-time operation reaches 38%.

The classical approach requires huge computational space to save the predefined map and generated nodes, and thus they are mainly implemented in offline strategies. In the mentioned papers, only 36.36% of the classical approaches can be employed for online performance. The hybridization of the classical approach is adapted to solve the mentioned problem and achieve real-time implementation by other algorithms with developed algorithms or functions. 72.73% of papers are improved as hybrid algorithms to overcome the drawbacks of the classical approaches.

The heuristic algorithms require less computation space than the classical approaches and can produce complete solutions. It is typical for the heuristic algorithms to be integrated with other algorithms, and the percentage of the hybrid approaches reaches 88.89%. Furthermore, 66.67% of the papers indicate they can be applied for online path planning and are achieved by computational efficiency. The power heuristic algorithms or the approaches involved in interactive robots can be used for online processing but with poor convergence performance.

Bio-inspired techniques are proposed for simple but powerful and robust solutions.

They can consider multiple constraints during path planning, even for a complex or dynamic environment. From the cited literature, PSO and GA are mainly involved in path optimization. High computational efficiency and fast convergence ensure real-time performance in dealing with dynamic obstacles, and 19.44% of metaheuristic algorithms demonstrate real-time abilities. Hybrid coevolutionary algorithms are usually proposed to overcome the drawbacks of a single evolutionary algorithm, such as trapping in local optima and uncertain scenes. The percentage of the hybrid approaches reaches 66.67%.

The AI-based approaches are developed to satisfy the dynamic environmental changes, especially with machine learning. Machine learning for multi-robot path planning mainly includes neural networks and reinforcement learning. They can usually achieve dynamic operation according to the environmental changes with the designed model or sensors, reaching 75% cited in AI-based papers. 60% of AI-based algorithms are combined with other approaches to improve learning abilities and reduce the consumed time.

Table 2.2: Comparison of multi-robot path planning algorithms

Category	**Approach**	**Paper**	**Real-Time**	**How to Achieve Real-Time Implementation**	**Experiment**	**Hybrid**
Classical	APF	[96]	N		N	N
		[97]	N		N	N
		[98]	N		N	Y
		[99]	N		Y	Y
		[100]	Y	Repulsion function	N	N
		[101]	Y	Priority-based algorithm	N	Y
		[102]	Y	APF	N	Y
	Sampling-based	[103]	N		N	Y
		[43]	N		N	Y
		[104]	N		N	N
Heuristic	A*	[28]	N		N	Y
		[114]	N		N	Y
		[115]	Y	Computational efficiency	N	Y
		[116]	Y	Robot	N	N
		[117]	Y	Computational efficiency	N	Y

	D*	[19]	Y	Sharing mechanism for robots	Y	Y
		[20]	Y	Algorithm	N	Y
		[118]	N		N	Y
		[119]	Y	Algorithm	N	Y
Bio-inspired	PSO	[127]	N		N	N
		[128]	N		N	N
		[129]	N		N	N
		[130]	N		Y	Y
		[131]	N		N	Y
		[132]	N		Y	Y
		[133]	N		N	N
		[134]	N		N	Y
		[135]	N		N	N
		[136]	N		Y	Y
		[137]	N		Y	Y
		[26]	Y	Computational efficiency	N	Y
		[18]	Y	Computational efficiency	N	Y
		[138]	N		N	Y

GA	[139]	Y	Computational efficiency	Y	Y
	[140]	N		N	Y
	[27]	N		N	Y
	[141]	N		N	Y
	[142]	N		N	Y
	[143]	N		N	Y
	[144]	N		N	Y
	[145]	N		N	Y
	[146]	N		N	N
	[147]	Y	Simplify the model	N	N
	[148]	N		N	N
	[149]	Y	Two-stage strategies	N	N
	[150]	Y	Computational efficiency	N	Y
ACO	[151]	N		N	Y
	[152]	N		N	N
	[153]	N		Y	Y
	[154]	N		N	Y
PIO	[155]	N		N	Y
	[156]	N		N	N

	GWO	[157]	N		N	N
		[158]	N		N	Y
		[159]	Y	Computational efficiency	N	Y
AI-based	Fuzzy logic	[171]	N		N	Y
		[16]	N		N	Y
		[172]	Y	Model	Y	Y
		[173]	Y	Computational efficiency	N	N
	Machine Learning	[174]	Y	Sensor	N	N
		[21]	Y	Algorithm	Y	Y
		[180]	N		N	Y
		[181]	N		N	Y
		[182]	N		N	N
		[183]	Y	Model	N	N
		[184]	Y	Model	N	N
		[185]	Y	Algorithm	N	N
		[186]	Y	Model	N	Y
		[187]	Y	Model	N	Y
		[188]	Y	Model	Y	Y
		[175]	Y	Model	N	Y

	[176]	Y	Algorithm	N	Y
	[177]	N		N	Y
	[178]	N		N	N
	[179]	Y	Algorithm	N	N

Where N stands for No, and Y stands for Yes.

2.5.3 Decision-making

Additionally, the decision-making strategies can be divided into centralised and decentralised categories. Table 2.3 compares the decision-making approaches for algorithms, real-time applications, and hybrid techniques. The percentage of real-time performance reaches 56%, and the portion of the offline techniques is 44%. The real-time implementation has a higher rate due to the cited literature on the decentralized framework.

For the centralized framework, the implemented algorithms include classical, bio-inspired, heuristic, and AI-based approaches. It is general for an algorithm to combine with other algorithms for improvement, and 72.73% of the cited centralized papers propose hybrid strategies. The heuristic techniques or the classical methods are integrated with the bio-inspired algorithms or network communications. The rate of real-time operation in the centralized framework reaches 54.55%.

The centralized framework achieves real-time implementation by an online network/system, the algorithm with fast speed, or data generation from the sensors. The decentralized framework has more real-time applications than the centralized framework. The robots gain information from the neighbours' robots to determine the next step and immediately operate the local communication system. A total of 57.14% of the decentralized approaches support online operations. The algorithms with fast convergence, simplicity, excellent robustness or little computational time and space are widely implemented in the structure. Only 23.81% of the cited decentralized papers involve the hybrid approaches.

Moreover, the hybrid structure has been developed recently to combine the advantages of centralized and decentralized approaches. It uses centralized problem formation while the robots make decisions during task operations. Robots can gain information from other robots or accomplish tasks under the distributed structure. The employed techniques have no restrictions because the hybrid method combines the benefits of centralized and decentralized methods as [206], [223], [235].

Table 2.3: Comparison of decision-making approaches

Category	**Approach**	**Paper**	**Real-Time**	**How to Achieve Real-Time Implementation**	**Experiment**	**Hybrid**
Centralized	GA and A*	[204]	N		N	Y
	Dijkstra and A*	[205]	N		N	Y
	Integer linear programming	[30]	N		N	N
	Feedback loop	[207]	Y	Multi-sensor	N	N
	Bid valuation and sampling-based approach	[31]	Y	Computational efficiency	N	Y
	Self-organizing map	[208]	Y	Computational efficiency	N	N
	Fuzzy programming	[209]	N		N	Y
	Simultaneous inform and connect	[210]	Y	Computational efficiency	N	Y
	A* and cloud computing	[211]	Y	Computational efficiency	N	Y
	Software Defined Network and APF	[212]	Y	Wireless network	N	Y

	Space Utilization Optimization	[214]	N		N	N
	Conflict based search	[215]	N		N	N
	Insertion	[216]	N		N	N
	Roadmap	[217]	N		N	Y
	Prioritized reinforcement learning	[33]	N		N	N
	PSO	[14]	N		N	N
	Free-ranging motion	[219]	N		N	N
Decentralized	A*	[220]	N		N	N
	APF	[221]	Y	Computational efficiency	N	Y
	Hypocycloid geometry	[222]	Y	Local communication	Y	N
	Linear program	[223]	Y	Computational efficiency	N	N
	Graph neural network	[224]	Y	Communications among robots	N	Y
	Graph Neural Network	[226]	Y	A key-query-like mechanism to communicate	N	Y

	Multi-agent reinforcement learning	[227]	Y	Computational efficiency	N	N
	Genetic Programming	[228]	Y	Computational efficiency	N	N
	Altruistic coordination	[229]	Y	Computational efficiency	N	N
	Potential field	[230]	Y	Robot communications	N	N
	APF	[231]	Y	Computational efficiency	N	N
	RRT and PRM	[232]	Y	Algorithms	N	Y
Hybrid	Path diversification heuristic	[206]	N		N	Y
	A*	[218]	N		N	N
	Markov Decision Process	[225]	Y	Computational efficiency	N	N

Where N stands for No, and Y stands for Yes.

2.5.4 Challenge

From the review of multi-robot path planning and decision-making strategies, the traditional challenges involved in multi-robot path planning can be considered local optima, ungranted completeness, and slow convergence. Many papers aim to solve these problems by integrating the different algorithms or using a developed controller. Nevertheless, this paper discovered a new challenge, as the multi-robot path-planning approaches have not considered fault tolerance. The proposed papers mention real-time implementation; however, most articles mainly focus on computational efficiency or model simplicity to provide faster convergence for online computation.

However, in real-time performance, updating the robots' status and backing up the robots' failures are essential. The robots can send positions or motions to the controller or the neighbours to update their status immediately rather than entirely relying on the predefined path, which can be achieved by localization or vision sensors. The multi-robot system's fault tolerance is aimed to support the system operating as expected, even if a robot fails.

For actual applications, a multi-robot system should detect the failure immediately and broadcast the information to avoid collisions with other robots or path congestion. Communications between the controller and robots and among robots are important during failure tolerance. The other robots should adjust their defined task plans or paths in real-time to achieve the tasks. The factors of communication overheads would include energy consumption, bandwidth, latency, communication network, and quality of service. It has no limitations of the system framework for fault tolerance because the centralized framework can inform all robots quickly, and the decentralized framework can send the fault signs to the neighbour robots.

Chapter 3

The Mathematical-based Path Planning algorithms for UAV

This chapter presents two mathematical-based path planning algorithms for different case studies, one is for the terrain environment based on the Quintic Hermite interpolation in Section 3.1, and the other one proposes Helix-HPSO for the multi-building environment in Section 3.2.

3.1 Quintic Hermite Interpolation

3.1.1 Introduction

Path planning and obstacle avoidance are major processes in Unmanned Aerial Vehicle (UAV) navigation. Path planning is to find a path between one position and one destination position. UAV navigation can be operated in several environments, such as indoor areas, air, and underwater. UAVs can carry cameras, sensors, urgent suppliers, or communications equipment, and they can be controlled remotely or operate autonomously [59]. UAVs can perform civilian applications, containing aerial photography, video shoots, aerial mapping, inspection, reconnaissance and surveillance, wildfire suppression, 3D modelling, agricultural services, traffic monitoring, intelligence, environmental monitoring, and search-and-rescue [58], [69], [79], [236].

To achieve the missions of UAVs, terrain following flight is involved in protecting

UAVs from collisions. The terrain following is to find a path that is close to the ground [59]. Goal assignment, team composition, path optimization, and resource allocation are contained to support multiple UAVs coordination [237]. Mostly, solving the best path planning is related to deterministic search algorithms. But there was a trend of using non-deterministic algorithms, such as the particle swarm optimization algorithm and the genetic algorithm [67].

Also, Dijkstra's algorithm, Bellman-Ford algorithm, Floyd-Warshall's algorithm, the vortex search algorithm, and A* can be used for finding the shortest path [2], [66]. A discretization of the airspace is meaningful for applying the path planning algorithms [67], [75]. From previous research, Quintic Hermite interpolation can also be used to calculate path segments with simplified calculation and implementation [59], [62]. Cost functions are utilized to evaluate the optimal path to ensure the path has a minimal cost compared to other paths. Costs can be calculated based on the actual costs or estimated costs. UAVs' cost functions usually include the consideration of path length, flight altitude, danger zones, energy consumption, threats, flight time, and path segments [2], [63], [64], [67], [233], [238].

This section solves the problem of finding the optimal path to the destination for UAVs. The section is organized as follows. Sections 3.1.2 and 3.1.3 present the path planning algorithm considering cost functions. Section 3.1.4 presents the simulation results to validate the approach. It is concluded in Section 3.1.5.

3.1.2 Description of The Algorithm

For applying the planning algorithm, the airspace will be discretised to meaningful representations for the algorithm. Waypoints are defined in the specified order. Each waypoint is represented as (x, y, z) for 3D navigation. The matrix with n waypoints is described as Equation (3.1):

$$Waypoints = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots \\ x_n & y_n & z_n \end{bmatrix} \tag{3.1}$$

Based on the literature, determining the flight path segment can be achieved by Quintic Hermite interpolation [62]. Using $(x(t), y(t), z(t))$ to locate the position of the UAV at the specific time t. The definition of t_{max} is the longest flight time from Point 1 to Point 2. For the path segment between Point 1 and Point 2, polynomials $x(t), y(t), z(t)$ are as follows.

$$x(0) = x_1 \tag{3.2}$$

$$x(t_{max}) = x_2 \tag{3.3}$$

$$x^{'}(0) = v_{1x} \tag{3.4}$$

$$x^{'}(t_{max}) = v_{2x} \tag{3.5}$$

$$x^{''}(0) = 0 \tag{3.6}$$

$$x^{''}(t_{max}) = 0 \tag{3.7}$$

$$y(0) = y_1 \tag{3.8}$$

$$y(t_{max}) = y_2 \tag{3.9}$$

$$y^{'}(0) = y_{1x} \tag{3.10}$$

$$y^{'}(t_{max}) = y_{2x} \tag{3.11}$$

$$y^{''}(0) = 0 \tag{3.12}$$

$$y^{''}(t_{max}) = 0 \tag{3.13}$$

$$z(0) = z_1 \tag{3.14}$$

$$z(t_{max}) = z_2 \tag{3.15}$$

$$z^{'}(0) = z_{1x} \tag{3.16}$$

$$z^{'}(t_{max}) = z_{2x} \tag{3.17}$$

$$z^{''}(0) = 0 \tag{3.18}$$

$$z^{''}(t_{max}) = 0 \tag{3.19}$$

Based on Quintic Hermite interpolation, the equation of x(t) is:

$$x(t) = a_5t^5 + a_4t^4 + a_3t^3 + a_2t^2 + a_1t + a_0 \tag{3.20}$$

$y(t)$ and $z(t)$ is calculated by the same formula as Equation (3.20). The coefficients can be calculated by calculating the six conditions for polynomial $x(t)$ with the equation of $x(t)$, and the linear system (3.21) is the equations for solving the coefficients for $x(t), y(t), z(t)$.

$$\begin{bmatrix} t_{max}^3 & t_{max}^4 & t_{max}^5 \\ 3t_{max}^2 & 4t_{max}^3 & 5t_{max}^4 \\ 6t_{max} & 12t_{max}^2 & 20t_{max}^3 \end{bmatrix} * \begin{bmatrix} a_3 \\ a_4 \\ a_5 \end{bmatrix} = \begin{bmatrix} x_2 - x_1 - v_{1x}t_{max} \\ v_{2x} - v_{1x} \\ 0 \end{bmatrix} \tag{3.21}$$

The t_{max} of Point 1 and Point 2 is calculated by:

$$Distance = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \tag{3.22}$$

$$t_{max} = \frac{Distance}{v} \tag{3.23}$$

Where (x_1, y_1, z_1) is the coordinate of the start point P1, (x_2, y_2, z_2) is coordinate of the destination point P2. v is the constant speed.

With autonomous navigation and defined waypoints, UAVs can perform aerial photography and environmental monitoring of specified locations by equipping the camera.

3.1.3 Implementation of The Algorithm

The algorithm is implemented using the following process. The waypoints are recorded in the Waypoint-Matrix. The algorithm uses iterations to calculate the path to get a smooth path curve. The Euclidean distance of the waypoints is calculated to calculate the flight time, then using distance to get t_{max}. For a more precise trajectory, t is set from 0 to t_{max} with a line space of 0.01s. t is being input to find the value of $\begin{bmatrix} a_3 \\ a_4 \\ a_5 \end{bmatrix}$, and with the value of the coefficients, the path segment

can be created. The points of the path are represented in the same way as the Waypoint-Matrix.

Algorithm 1 shows the algorithm for generating the path, and the matrix imports the waypoints. M is getting the size of the matrix, then sets the values of $X1$, $Y1$, $Z1$, $X2$, $Y2$, and $Z2$ by locating the value within the matrix. Call the $GetTime$ function to calculate each path segment's t_{max}. The $GetTime$ function calls the $GetPathSegments$ function to get the positions. Using an empty matrix to store the positions of each path segment. The algorithm involves iterations to get path segments of every two waypoints.

Algorithm 2 presents the algorithm for getting time. The values of $X1$, $Y1$, $Z1$, $X2$, $Y2$, $Z2$ are passed by the parameters, and the constant speed V is input as 18. Calculate the Euclidean distance between two waypoints, then calculate t_{max} by using the distance to divide the speed for generating the path segments. Then get the max value of t, and set the line space as 0.01. Call the $GetPathSegments$ function to get the positions of the path segment between every two waypoints.

Algorithm 3 presents the algorithm for getting path segments. The parameters pass the values of $X1, Y1, Z1, X2, Y2, Z2$, and $V1X, V2X, V1Y, V2Y, V1Z, V2Z$ are input as 18 m/s, which is within the regular cruising speed for civilian UAV. Calculate the coefficients between two waypoints, then calculate the points of the path segment by the coefficients. Then storing the positions of the path segment in a matrix.

Algorithm 1: GenerationOfPath

Input: $matrix$

Output: Path, $KeyPoints$

$M \leftarrow size(matrix, 1)$

$KeyPoints \leftarrow []$

if $M > 1$ **then**

  for $i \leftarrow 1 : M - 1$ **do**

    $X1 \leftarrow matrix(i, 1)$

    $Y1 \leftarrow matrix(i, 2)$

    $Z1 \leftarrow matrix(i, 3)$

    $X2 \leftarrow matrix(i + 1, 1)$

    $Y2 \leftarrow matrix(i + 1, 2)$

    $Z2 \leftarrow matrix(i + 1, 3)$

    $Points \leftarrow$
    $GetTime(X1, Y1, Z1, X2, Y2, Z2, V1X, V2X, V1Y, V2Y, V1Z, V2Z)$

    $SizeOfPoints \leftarrow size(Points, 1)$

  end

  if $i < 2$ **then**

    $KeyPoints \leftarrow [Keypoints; Points]$

  else

    if $SizeOfPoints = 2$ **then**

      $Points \leftarrow Points(2, :)$

      $KeyPoints \leftarrow [Keypoints; Points]$

    else

      $Points \leftarrow Points(2 : SizeOfPoints, :)$

      $KeyPoints \leftarrow [Keypoints; Points]$

    end

  end

end

Algorithm 2: GetTime

Input: $X1, Y1, Z1, X2, Y2, Z2, V1X, V2X, V1Y, V2Y, V1Z, V2Z$

$Equ \leftarrow (X2 - X1)^2 + (Y2 - Y1)^2 + (Z2 - Z1)^2$

$Distance \leftarrow sqrt(Equ)$

$tmax \leftarrow Dis/V$

$t \leftarrow 0 : 0.01 : ceil(tmax)$

$PathSegments \leftarrow GetPathSegments(t, X1, Y1, Z1, X2, Y2, Z2,$
$V1X, V2X, V1Y, V2Y, V1Z, V2Z)$

Algorithm 3: GetPathSegments

Input: $t, X1, Y1, Z1, X2, Y2, Z2, V1X, V2X, V1Y, V2Y, V1Z, V2Z$

Output: $Points$

$A \leftarrow [t(end)^3 \quad t(end)^4 \quad t(end)^5;$
$\quad 3 * t(end)^2 \quad 4 * t(end)^3 \quad 5 * t(end)^4;$
$\quad 6 * t(end) \quad 12 * t(end)^2 \quad 20 * t(end)^3];$

$XB \leftarrow [X2 - X1 - (V1X. * t(end)); V2X - V1X; 0];$

$x \leftarrow A \backslash XB$

$Xpoints \leftarrow x(3,1) * t^5 + x(2,1) * t^4 + x(1,1) * t^3 + V1 * t + X1$

$YB \leftarrow [y2 - y1 - (V1Y. * t(end)); V2Y - V1Y; 0];$

$y \leftarrow A \backslash YB$

$Ypoints \leftarrow y(3,1) * t^5 + y(2,1) * t^4 + y(1,1) * t^3 + V1 * t + Y1$

$ZB \leftarrow [z2 - z1 - (V1Z. * t(end)); V2Z - V1Z; 0];$

$z \leftarrow A \backslash ZB$

$Zpoints \leftarrow z(3,1) * t^5 + z(2,1) * t^4 + z(1,1) * t^3 + V1 * t + Z1$

$Points \leftarrow []$

for $i \leftarrow 0 : size(t, 2) - 1$ **do**

$\quad Points \leftarrow [points; Xpoints(i+1)Ypoints(i+1)Zpoints(i+1)]$

end

Cost Functions

The cost function evaluates a generated path, and it is optimal with minimal costs. Each cost function value can be defined within [0, 1] by a fraction. The total costs add all values to compare the paths.

The costs function f_{cost} is defined as:

$$f_{cost} = f_{length} + f_{time} + f_{altitude} + f_{collision} \tag{3.24}$$

Where f_{length} is the cost function of the path length, f_{time} is the cost function of flight time, $f_{altitude}$ is the cost function of mean altitude, $f_{collision}$ is the cost function of a collision.

Path Length The cost function f_{length} for path length is developed as follows:

$$f_{length} = \frac{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}}{\sum_{i=1}^{n} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 + (z_{i+1} - z_i)^2}} \tag{3.25}$$

Where (x_1, y_1, z_1) is the start point, (x_2, y_2, z_2) is the destination point, and n is the number of the points of the path segment. $\sum_{i=1}^{n} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 + (z_{i+1} - z_i)^2}$ is the actual cost of path length. $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$ is the length of the straight line between Point 1 and Point 2.

Flight Time The cost function f_{time} for flight time is developed as follows:

$$ActualT = \frac{\sum_{i=1}^{n} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 + (z_{i+1} - z_i)^2}}{V_{avg}} \tag{3.26}$$

$$Time = \frac{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}}{v_0} \tag{3.27}$$

$$f_{time} = 1 - \frac{Time}{ActualT} \tag{3.28}$$

Where $ActualT$ is the actual cost of time when going through the path, and v_{avg} is the average velocity during the flight. Time is the estimated cost of going through the straight line between Point 1 and Point 2, and v_0 is the initial velocity.

Altitude The cost function $f_{altitude}$ is proposed as:

$$f_{altitude} = 1 - \frac{\sqrt{(z_2 - z_1)^2}}{\sum_{i=1}^{n} \sqrt{(z_{i+1} - z_i)^2}} \tag{3.29}$$

Where z_i is the altitude of the point located in the path. $\sqrt{(z_2 - z_1)^2}$ is the altitude difference between Point 1 and 2, and $\sum_{i=1}^{n} \sqrt{(z_{i+1} - z_i)^2}$ is the sum of the altitude difference between pairs of points in the path.

Collision The cost function $f_{collision}$ for collision is as:

$$f_{collision} = \frac{\sum_{c=1}^{cn} \sqrt{(x_{c+1} - x_c)^2 + (y_{c+1} - y_c)^2 + (z_{c+1} - z_c)^2}}{\sum_{i=1}^{n} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 + (z_{i+1} - z_i)^2}} \tag{3.30}$$

Where cn is the number of points that have collisions with the terrain or objects, and $\sum_{c=1}^{cn} \sqrt{(x_{c+1} - x_c)^2 + (y_{c+1} - y_c)^2 + (z_{c+1} - z_c)^2}$ is the sum of the path segments which hit the terrain.

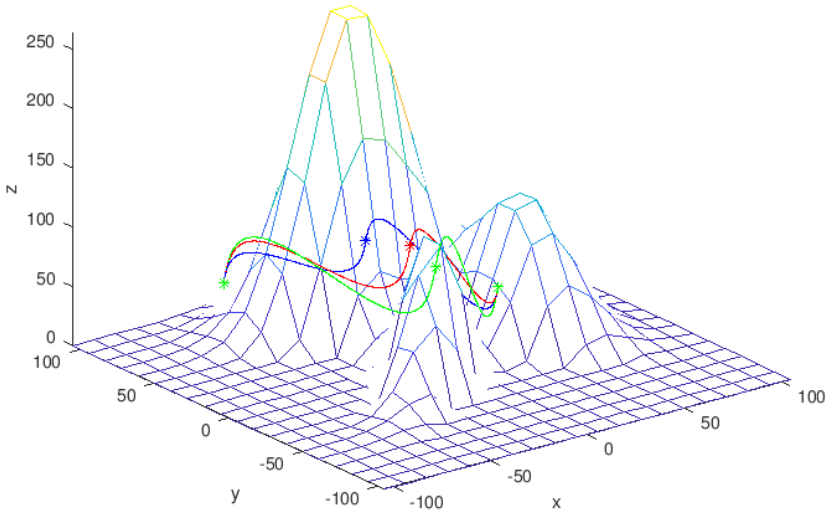
3.1.4 Simulation Results

We use simulation to compute UAV paths and calculate cost functions for different path designs to validate our approach. Simulation of the algorithm was implemented with MATLAB. The constant flight speed is set to 18 m/s. Because the safety margin for UAVs is 50 m above the ground, the path should be close to the altitude. Figure 3.1a integrated the paths with the terrain, using different colours to indicate the different paths. It presents the paths from a three-dimensional view.

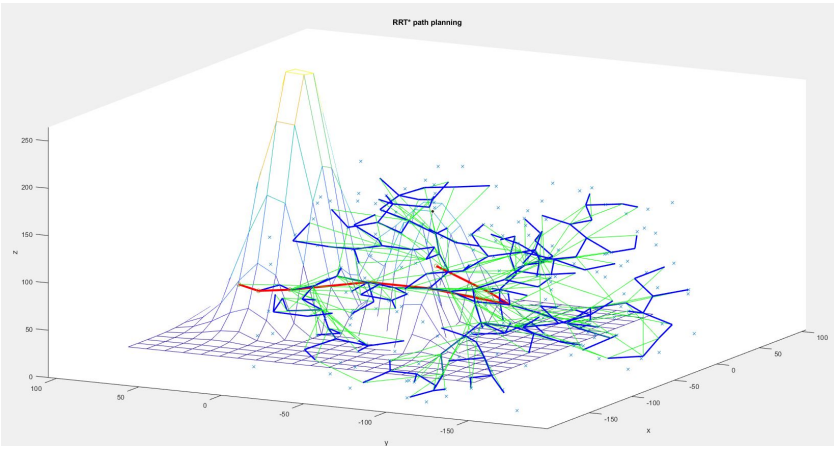
Figure 3.1b shows the path generated by the RRT* algorithm with obstacle avoidance. It is used to make a comparison with the Quintic Hermite interpolation approach. The path is marked red. When generating the path, it is more time-consuming than the Quintic Hermite interpolation approach. The costs function evaluates the paths, as shown in Table 3.1. From the comparisons of cost values among the four paths, the path with minimal cost is the Blue Path. The minimal cost is 2.4239.

3.1.5 Conclusion of Path Planning for The Terrain Case

The 3D path planning algorithm considers the terrain and protects the paths from collisions. The algorithm implements Quintic Hermite interpolation for planning the path. To generate the path segments, iterations and matrixes are involved.



(a) The generated paths from the side viewpoint



(b) The RRT* generated path from the side viewpoint

Figure 3.1: The generated paths

Table 3.1: Simulated paths for the robot

Costs	Blue Path	Red Path	Green Path	RRT* Path
f_{length}	0.4726	0.5218	0.6452	0.6272
f_{time}	0.9986	0.9989	0.9994	0.9184
$f_{altitude}$	0.9526	0.9549	0.9667	0.9075
$f_{collision}$	0	0	0	0
f_{cost}	2.4239	2.4755	2.6113	2.4531

The waypoints are used to improve the possibility of successful navigation and are defined in a specific order. The paths are required to pass these waypoints to reach the destination. The cost function is proposed in this section, and each cost is limited to the range of [0, 1], then sum the costs up to get the total cost. The cost function considers path length, time, altitude and collision. This section can be used for UAV 3D path planning. Future work should recognize UAVs' dynamic and kinematic behaviours and automatically generate the waypoints.

3.2 Helix-HPSO Approach

3.2.1 Introduction

Unmanned Aerial Vehicles (UAVs) can provide remote inspections and obtain different digital imagery by sensors or cameras, such as thermal, ultrasonic, laser scanners, high-resolution, and near-infrared [239], [240]. Computer vision and other technologies can process the collected data to detect surface defects of infrastructure, including distortion, spalling, cracking, excessive movements, rusting, and misalignment [240]. An informative and efficient path is required to perform UAV-based path planning inspection with data from different views [240].

This research was conducted because of an accident in Notre Dame de Paris, which attracted attention to protect the historical buildings and collect the building data in case of an accident. Historic buildings have historical value as symbols of specific eras and can exhibit the architect's aesthetic and past people's lives. Monitoring historic buildings' condition and regular inspections of historic buildings are essential.

The Helix-HPSO approach is proposed as UAV-based path planning for building inspection of multiple historical buildings. It uses the helix-shaped path for inspecting each building, which is a smooth path at a low cost and suitable for UAV constraints. Also, the Helix path is compared with the traditional inspection path. The proposed HPSO algorithm generates the path for flying to another building, considering the distance and collision avoidance. The HPSO has been compared with other bio-inspired algorithms through benchmark functions and with PSO for path planning. The contributions of this section include the following:

- Improve PSO with the inspiration from the Harmony search algorithm as a new algorithm, HPSO.
- A novel 3D path planning approach based on the helix-shaped path and proposed HPSO for a smooth and safe path.
- Provide the generic building inspection approach in the multi-building environment.

This section proposes a UAV-based path planning for a building inspection as the Helix-HPSO approach, organized as follows. Section 3.2.2 reviews the UAV path planning algorithms and building inspection approaches. Section 3.2.3 describes the problem formulation, and Section 3.2.4 proposes the Helix-HPSO approach. The result of the computational experiment is in Section 3.2.5. This section is concluded in Section 3.2.6.

3.2.2 Related Work

The path planning algorithms can be classified as node-based [241], sampling-based [242], mathematical mode [243], multi-fusion-based [239] and bio-inspired algorithms [240]. A* and DWA is integrated for global path optimization for UAV path planning to improve safety and efficiency [241]. A genetic algorithm and A* algorithm are integrated to solve the travelling salesman problem for LiDAR-equipped UAV path planning [244]. An improved A* is combined with a gravitational search algorithm for the optimal path with several optimization objectives, such as the return point and the heading angle [245].

Moreover, for UAV path planning, a biased sampling potentially guided intelligent bidirectional RRT* algorithm is proposed to overcome the slow convergence rate of exploration [242]. UAV path planning can be transformed into a nonlinear optimal control problem, and sequential convex programming is proposed to solve it by approximating the non-convex parts [243]. An improved Artificial Potential field (APF) is proposed for UAV path planning with additional control force, and it avoids local minima [246]. The chaotic bat algorithm is combined with an improved APF for faster position update and adaptive inertia weight [247].

An odd-even layered genetic algorithm (GA) is proposed for cooperative inspection path planning in [248]. Particle swarm optimization (PSO) and the mutation operator of GA are combined to improve the central force optimization algorithm and aim for complicated path optimization [249]. In a proposed random chemical reaction optimization algorithm [250], PSO and elitist selections are combined to act as the subsets of the set of molecules. It is used for UAV inspection path planning.

Additionally, a heuristic evolutionary algorithm in [251] implements mutation, substitution, crossover, smoothness and length operations to build the path for UAV. The reinforcement learning is improved based on the grey wolf optimizer algorithm to control the switch operations for individuals during UAV path planning [252]. The differential evolution algorithm is modified with symbiotic organism search for impressive local and global search ability [201]. A modified Mayfly algorithm is proposed in [253] with an enhanced crossover operator, adaptive Cauchy mutation and an exponent decreasing inertia weight to search the configuration space and get the path with the lowest cost for UAV path planning.

UAVs offer excellent flexibility in many fields with the development of technologies and remote photogrammetry, including security, rural environment monitoring, urban planning, and, recently, infrastructure inspection [254]–[256]. UAV applications in construction include site surveying, safety inspection, building inspection, damage assessment, urban monitoring, road assessment, geo-hazard investigations, and progress monitoring [257], [258].

UAV infrastructure inspections can be classified into five categories: bridges, power lines, buildings and facades, railways and sewers, and geographical inspections [254]. A regular building inspection is necessary to ensure their condition is safe. Maintenance and detection of building façade anomalies require periodic safety inspections and examinations [259]. They are more critical to the historical buildings as heritage [256].

Detailed three-dimensional heritage reconstruction is important for interpretation, analysis, and physical reconstruction [260]. Heritage documentation includes post-catastrophe damage assessment and modelling facades, monuments or entire buildings [260]. Inspectors usually take photos directly of every element in general prac-

tice to record damage or dilapidation, which is time-consuming and expensive [239], [256]. UAV technology overcomes risky, time-consuming, and costly inspection practices for monitoring and inspecting infrastructure [261].

Additionally, the research of building inspection is for evaluating the condition of a building and checking the appearance of a building. The UAV survey aims to model the structure and recognize the state [262]. UAV-based photogrammetry evaluates the condition of facilities and allows complete documentation of buildings with fewer human resources and time, particularly for the areas difficult to access [240], [256], [263]. A UAV-based project's geometric result relies on the UAV manufacturing quality and the onboard sensor system, materials, shutter modes, and digital sensor types [260].

The autonomous navigation of UAVs relies on local information in the GNSS-denied zones, and an RGB-D Kinect camera or visual information can obtain such local information to compute an analytical path [264]. Distance measuring units can also calculate the angle and the range between the structure and the UAV during the inspection [265]. The pre-processing algorithm processes the local information obtained from the point cloud and then uses the path planning algorithm to generate the path for the indoor environment [264].

Range-based and image-based techniques are employed widely in the three-dimensional documentation of buildings, and UAVs have advantages for image-based techniques with aerial views [260]. UAVs follow a programmed flight path to record and survey historical buildings, obtaining survey data for unreachable areas with a terrestrial platform [240], [255], [266]. UAVs are equipped with sensors for navigation, 3D data acquisition, and obstacle avoidance [239], [240], [255]. It achieves higher flexibility with low cost [267]. Survey data can vary from sensors such as Forward-Looking Infrared technology, images, separate strips, multi-attributed point cloud, 3D point cloud, and laser scanners [239], [256], [267], [268]. UAV-derived point clouds can extract the surface after image processing [269]. The building information model and the UAV inspection workflow are integrated for an augmented reality solution with aerial video [270], [271].

Laser scanning data, close-range photogrammetric images, or a combination of both

are generally applied for building inspection documentation [240], [263]. Close-range facade images are employed to inspect and document facade anomalies, such as corrosion and cracks [259]. 2D photographs generated by UAVs enable the creation of a precise model of buildings with proper path planning and using dedicated software to identify the defects [263]. Close-range high-resolution inspections can achieve 3D modelling of a historical building [266]. Documentation of heritage buildings with 3D reconstruction and photogrammetry is based on motion with a dense matching algorithm and optical sensors [260]. LiDAR, stereo camera, IMU, and wide-angle camera are implemented for the UAV-UGV system for indoor and cluttered scenes [272].

Moreover, a GIS-based two-step procedure supports building facade inspection by processing the management of UAV-collected images [259] in the inspection, measuring the geometry of buildings and requiring documentation in the form of photogrammetric images [255], [273]. Advanced photogrammetric techniques and deep learning algorithms are applied to record building damages autonomously by a true-orthophoto [273]. A Convolutional Neural Network is fine-tuned to detect damage and surface cracks [274], [275]. Ultrasonic Beacon system replaces GPS for autonomous flight, and a geotagging method can locate damages [274].

For buildings with planar surfaces, particle swarm optimization is adapted as enhancing discrete particle swarm optimization to plan the path for UAV [239]. The most common path planning method for a building inspection is back-and-forth paths. The inspection of the Perak Museum used back-and-forth paths; the UAV flew from the bottom of the building, then moved to the top, and flew to the right side, then reached the ground [256].

From the literature, the popular algorithms used in UAV path planning are based on bio-inspired algorithms. The bio-inspired algorithms provide fast path generation with optimal solutions. For the building inspection, the related work lacks the consideration of multiple buildings. This section proposes the Helix-HPSO approach to address the problem of the multi-building environment, generating the path for each building and between the buildings.

3.2.3 Problem Formulation

The basic components of UAV path planning include the start, the goal and the environment. The shape of most buildings can be modelled as rectangular or cylinder. The building inspection requires broad coverage with a short path. For proceeding between buildings, collision avoidance and flight distance should be considered. It is aimed at generating a safe and optimal path. The developed cost functions to evaluate the paths are as below.

Cost Functions for Building Inspection

The cost function is defined as Equation (3.31), and it is to evaluate the inspection paths.

$$f_{cost} = w_1 * f_{distance} + w_2 * f_{time} + w_3 * f_{altitude} \tag{3.31}$$

Where w_1, w_2 and w_3 are the weights of the cost functions of distance, flight time, and the change of altitude. The cost functions are listed as Equations (3.32) – (3.34). The sum of the weights is 1.

$$f_{distance} = \sum_{k=1}^{n} \sqrt{(x_{k+1} - x_k)^2 + (y_{k+1} - y_k)^2 + (z_{k+1} - z_k)^2} \tag{3.32}$$

Where n is the size of the path points, and x_k is the current path point.

$$f_{altitude} = \sum_{k=1}^{n} \sqrt{(z_{k+1} - z_k)^2} \tag{3.33}$$

Where $\sqrt{(z_{k+1} - z_k)^2}$ is the change of altitude.

$$f_{time} = \sum_{k=1}^{n} \frac{\sqrt{(x_{k+1} - x_k)^2 + (y_{k+1} - y_k)^2 + (z_{k+1} - z_k)^2}}{v} \tag{3.34}$$

Where v is the current velocity.

Cost Functions for Path Planning

The cost functions are developed to evaluate the optimal path during path planning, considering the distance and collision avoidance. The best solution would be the

generated path for flying to another building.

$$f_{path\ cost} = w_4 * f_{length} + w_5 * f_{collision} \tag{3.35}$$

Where w_4 and w_5 are the weights of the length and collision cost functions. The cost functions are listed as Equations (3.36) – (3.39). The sum of the weights is 1.

$$f_{length} = \sum_{k=1}^{n} \sqrt{(x_{k+1} - x_k)^2 + (y_{k+1} - y_k)^2 + (z_{k+1} - z_k)^2} \tag{3.36}$$

Where n is the number of path points, and x_k is the current path point.

$$d_k = \sqrt{(x_k - x_c)^2 + (y_k - y_c)^2 + (z_k - z_c)^2} \tag{3.37}$$

Where d_k is the distance from the current path point to the centre of an obstacle.

$$c_k = \sum_{c=1}^{cn} r_c - d_k \tag{3.38}$$

Where cn is the number of obstacles. The collision occurs if $r_c - d_k < 0$, which means that the current path point overlaps with an obstacle c.

$$f_{collision} = \sum_{k=1}^{n} c_k \tag{3.39}$$

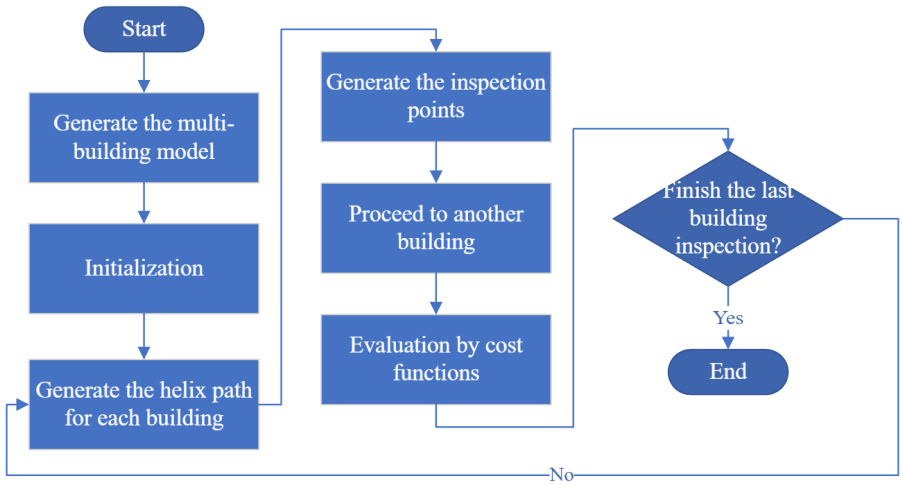
Where $f_{collision}$ sums up the violation value for collisions between each path point and obstacle.

3.2.4 Helix-HPSO Approach

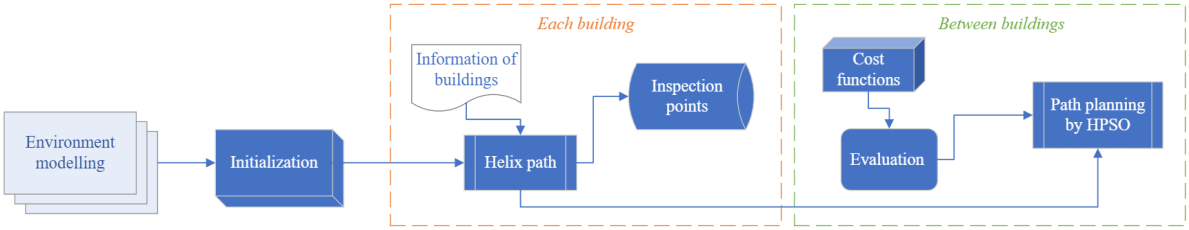
Overview

The helix-HPSO approach generates the helix path for each building, defining the points where info is collected with reasonable time slots. After inspecting one building, the UAV proceeds to another building with an optimal path based on a visited vector and a cost matrix, as shown in Figure 3.2a. Cost functions evaluate several factors, including distance, time, and altitude. The helix-HPSO approach is demonstrated in Figure 3.2b. The steps of the proposed algorithm are as follows.

1. Build the model for the multi-building environment.
2. Generate the inspection path and the points where info is collected for each building.
3. Proceed to another building after inspecting one building.



(a) The flow chart of the Helix-HPSO approach



(b) The principle of the Helix-HPSO approach

Figure 3.2: The Helix-HPSO

Path planning for Each Building: Helix Path

Helix Path The helix path [276] for inspecting each building is generated through Equations (3.40) – (3.42). The position of the UAV is denoted as $(x(t), y(t), z(t))$ at the specific time t. The time slot is set to 0.01 for t. Recording current t as t_{max} and terminating the iteration once $z(t)$ reaches the determined height based on the inspection type.

$$x(t) = r * sin(\frac{vt}{2\pi r}) + x_g \tag{3.40}$$

$$y(t) = r * cos(\frac{vt}{2\pi r}) + y_g \tag{3.41}$$

$$z(t) = (\frac{vh}{2\pi r}) * t + z_g + 1 \tag{3.42}$$

Where v is the UAV flight speed, (x_g, y_g, z_g)represents the centre of the ground floor of a building, r is the radius of the helix path, and h is the height of each floor.

Algorithm 4 presents the algorithm for generating the interior and exterior inspection paths. The exterior and interior inspection paths are connected through the door, inspecting the entrance hall.

Algorithm 4: Generation Of Inspection Path

Input: $xg, yg, zg, floorNum, radiusOfBuilding,$
$heightOfEachFloor, heightOfRoof, inspectionType$

Output: Helix Path

```
if inspectionType == "interior" then
    h ← heightOfEachFloorForInterior
    r ← radiusOfBuildingForInterior
    tMax ← getTmaxForInterior()
else
    h ← heightOfEachFloorForExterior
    r ← radiusOfBuildingForExterior
    tMax ← getTmaxForExterior()
end
for time ← 0 : 0.01 : tMax do
    xt ← r * sin(v * time/2 * pi * r) + xg
    yt ← r * cos(v * time/2 * pi * r) + yg
    zt ← (v * h)/(2 * pi * r) * time + zg + 1
    points ← [points; xt, yt, zt]
end
```

The time required for an n-floor building inspection is defined by Equation (3.43), with a flight speed v.

$$T_s = \frac{2\pi rn}{v} \tag{3.43}$$

The points of the path are stored as Equation (3.44).

$$Points = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots \\ x_n & y_n & z_n \end{bmatrix} \tag{3.44}$$

Path Planning Between Buildings: HPSO

Cost Matrix After completing the inspection of one building, the UAV proceeds to another building for inspection. Each building is marked with numbers, and the cost matrix evaluates the order of inspecting buildings as Equation (3.45). If a building has been visited, update the visited vector and the related costs as $infinity$.

$$Cost = \begin{bmatrix} infinity & cost(1,2) & ... & cost(1,m) \\ cost(2,1) & infinity & ... & cost(2,m) \\ \vdots & \vdots & \vdots & \vdots \\ cost(m,1) & cost(m,2) & ... & infinity \end{bmatrix} \tag{3.45}$$

where m stands for the number of buildings in the environment.

Preliminary Knowledge

Particle Swarm Optimization (PSO) Particle swarm optimization (PSO) is proposed by [277] and is inspired by the social behaviour of fish schooling or bird flocking. It uses the global and local-oriented values to track each particle's coordinates in hyperspace. PSO uses initial random solutions and updates the velocity and position for each particle as Equations (3.46) – (3.47) to seek the optimal solutions in the configuration space. Algorithm 5 shows the principle of PSO.

$$v_i^{t+1} = \omega v_i^t + c_1 r_1 (pbest_i^t - x_i^t) + c_2 r_2 (gbest^t - x_i^t) \tag{3.46}$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{3.47}$$

Where v stands for particle velocity and ω is the inertial weight. $t+1$ is the current timeslot, and i is the current particle. *pbest* is the best local-oriented value, and *gbest* is the best global-oriented value. x is the position of the particle.

Harmony Search (HS) Harmony search (HS) is proposed by mimicking the improvisation of music players [278]. It aims to get better solutions but with fewer iterations than other heuristic algorithms. Music harmony consists of several sound waves with different frequencies, and performances seek the best state. HS initializes a harmony memory (HM). Set the harmony memory considering rate (HMCR), harmony memory size (HMS), and pitching adjusting rate (PAR). Then, improvise a new harmony from HM as Equations (3.48) – (3.50). Add the better harmony in HM and exclude the old harmony. Keep seeking harmony with the minimum harmony until satisfying the stopping criterion. HS is described by Algorithm 6.

$$x_i^{'} = \begin{cases} x_i^{'} \in \left\{x_i^1, x_i^2, ..., x_i^{HMS}\right\}, \text{ with probability HMCR} \\ x_i^{'} \in X_i, \text{ with probability (1-HMCR)} \end{cases} \tag{3.48}$$

Where HMCR is the probability of getting one historic value in the HM

$$\textit{Pitch adjusting decision of } x_i^{'} = \begin{cases} YES, \text{ with probability PAR} \\ NO, \text{ with probability (1-PAR)} \end{cases} \tag{3.49}$$

If a value is from the HM, the pitching process happens. If the decision is YES, then updating $x_i^{'}$ as Equation (3.50).

$$x_i^{'} = x_i^{'} + \alpha \tag{3.50}$$

Where α can be the result of the arbitrary distance bandwidth multiplying a uniform distribution in [-1,1] or the arbitrary distance bandwidth multiplying the standard normal distribution.

Algorithm 5: PSO algorithm

Input: x,v

Output: PSO

```
1  Initialization
2  for t ← 1 : t_Max do
3      for i ← 1 : numberOfParticle do
4          Update Velocity
5          Update Position
6          Evaluation
7          if cost(particle_i^t) ≤ particle_i.pbest then
8              particle_i.pbest ← particle_i^t
9          end
10         if cost(particle_i^t) ≤ gbest then
11             gbest ← particle_i
12         end
13     end
14 end
```

Algorithm 6: HS algorithm

Input: VarMin, VarMax, VarSize

Output: HS

```
Initialization
for it ← 1 : it_Max do
    for k ← 1 : nNew do
        x_i^new ← unifrnd(VarMin, VarMax, VarSize)
        for j ← 1 : nVar do
            if rand<HMCR then
                x_i^new ← x_i^fromHM
                if rand<PAR then
                    x_i^new ← x_i^fromHM + α
                end
            end
        end
        Evaluation
        Merge HM
        Update Best Solution
    end
end
```

Harmony Particle Swarm Optimization (HPSO)

From Section 3.2.2, the bio-inspired algorithms are widely used in UAV path planning, especially PSO and GA. Due to the requirement of fast computation, PSO is chosen to be improved for UAV path planning between buildings. HS is proposed for better solutions with fewer iterations, inspiring the improvement of the PSO approach. Thus, a hybrid HPSO is proposed for fast convergence and fewer iterations. The inertial weight in HPSO is adjusted based on Equations (3.51) – (3.53). Algorithm 7 demonstrates the proposed HPSO algorithm.

Algorithm 7: HPSO algorithm

Input: v,x,damping,PAR,BW

Output: HPSO

```
Initialization
for t ← 1 : t_Max do
    for i ← 1 : N do
        Update Velocity
        Update Position
        Evaluation
        if cost(particle_i^t) ≤ particle_i.pbest then
            particle_i.pbest ← particle_i^t
        else
            Alpha ← BW * randn()
            if rand ≤ PAR then
                w ← w + Alpha
            end
        end
        if cost(particle_i^t) ≤ gbest then
            gbest ← particle_i
        end
    end
    w ← w * damping
    BW ← BW * damping
end
```

$$\alpha = BW + randn() \tag{3.51}$$

Where BW is the arbitrary distance bandwidth inspired by the principle of HS and $randn()$ generates a Gaussian distribution.

$$\omega = \omega + \alpha \tag{3.52}$$

Where ω is the inertial weight of HPSO, the adjustment processing only occurs when the current cost is larger than the personal best value, and the probability is less

than PAR.

$$\omega = \omega * damping\ ratio \tag{3.53}$$

Where *damping ratio* is the ratio of updating the inertial weight.

The algorithm initializes the velocities and positions of the particles. Then update the positions and velocities with Equations (3.46) – (3.47) and evaluate the solution by the cost function. If the current cost is less than the personal best value, it becomes the local best-oriented value. Otherwise, adjusting inertial weight for the algorithm based on the probability of PAR and α is calculated by the bandwidth multiplying the standard normal distribution. The proposed algorithm allows better solutions with fewer iterations and adjusts the inertial weight randomly for better search abilities to avoid getting trapped in local optima. The global best value keeps updating until the termination criteria reach.

3.2.5 Computational Experiment

Performance Measurements

The proposed HPSO was compared with other bio-inspired methods, such as PSO, HS, GA [279] and Firefly algorithm (FA) [280]. The test functions are listed in Table 3.2 to compare the performance of the proposed algorithm with other bio-inspired algorithms. Each test function is a typical benchmark function with different characteristics. Ackley function has many local minima. The shapes of Rosenbrock and Sphere functions are like valley and bowl, respectively. The shape of Michalewicz function is a steep ridge. The number of the maximum iteration was 200.

Table 3.2: Test functions

Name	**Type**	**Test function**
Ackley	Many Local Minima	$f_1(x) = -a\ exp\left(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - exp\left(\sqrt{\frac{1}{d}\sum_{i=1}^{d} cos(cx_i)}\right) + a + exp(1)$
Rosenbrock	Valley-Shaped	$f_2(x) = \sum_{i=1}^{d-1}[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$
Sphere	Bowl-Shaped	$f_3(x) = \sum_{i=1}^{d} x_i^2$
Michalewicz	Steep Ridges/Drops	$f_4(x) = -\sum_{i=1}^{d} sin(x_i) sin^{2m}(\frac{ix_i^2}{\pi})$

Table 3.3 lists the mean iteration times when getting the optimal solution, the runtime of each algorithm, and fitness values for each benchmark function with different algorithms. Each algorithm has been run 20 times. The best value is highlighted in bold. The proposed HPSO had the best performance for Ackley, which has many local minima. HPSO had minimal iteration times for the remaining test functions, while PSO had minimal runtime. Nevertheless, the difference between the runtime of PSO and HPSO was insignificant. It can be concluded that HPSO has great performance compared with other algorithms, especially for problems with many local minima. Getting rid of local optima and fast convergence are important for path planning, so HPSO is suitable for UAV path planning.

Table 3.3: Performance of test functions

Function		HPSO	PSO	HS	GA	FA
Ackley	Iterations	**127.6**	177.25	179.35	199.6	191.25
	Runtime (s)	**0.007485**	0.008081	0.106657	0.275348	1.008479
	Fitness value	**0.00**	**0.00**	0.01	**0.00**	**0.00**
Rosenbrock	Iterations	**1**	**1**	190.15	199.1	197.5
	Runtime (s)	0.005447	**0.003763**	0.107351	0.329114	0.970046
	Fitness value	**0.00**	**0.00**	4.05	7.86	0.40
Sphere	Iterations	**102.8**	192.4	174.6	199.1	187.45
	Runtime (s)	0.0057645	**0.0045109**	0.103878	0.2901371	0.9958323
	Fitness value	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Michalewicz	Iterations	**1**	**1**	174.3	192.75	188.55
	Runtime (s)	0.005332	**0.003449**	0.115909	0.335478	1.070354
	Fitness value	-0.86	**-0.84**	-4.87	-9.31	-4.41

Figure 3.3 presents the test functions' mean fitness value's convergence curve. Figure 3.3 and Table 3.3 show that the proposed HPSO converges faster than other bio-inspired algorithms. It has much fewer iteration times to get the optimization solutions, which is useful for path planning.

Scenario: Fujian Tulou

Modelling A Tulou is an enclosed, large and fortified earthen building, most commonly circular or rectangular in configuration. It has thick, load-bearing, and

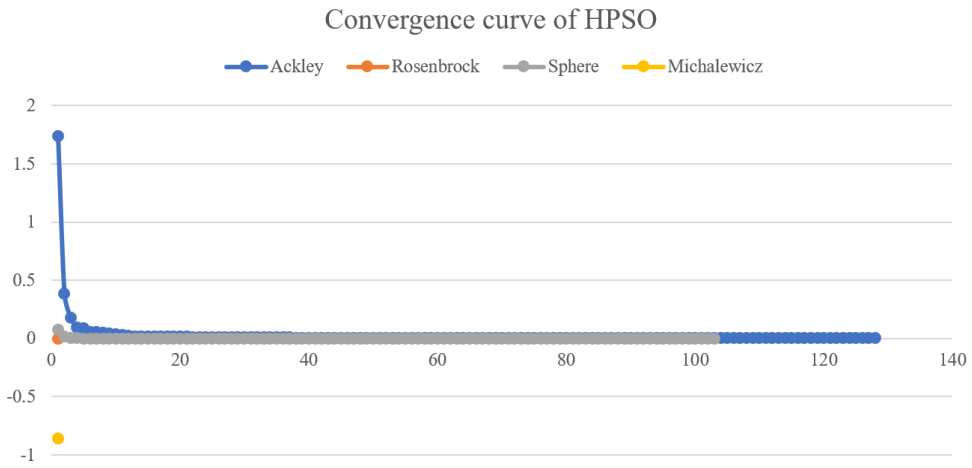


Figure 3.3: Convergence curve of HPSO

rammed earthen walls, housing up to 800 people. It represents the living style from Song and Yuan Dynasty, around C.E. 960 to C.E. 1368. Figure 3.4a displays the photo of the Tianluokeng Tulou cluster comprising five main buildings. Figure 3.4b presents the model with the marked number of each building.

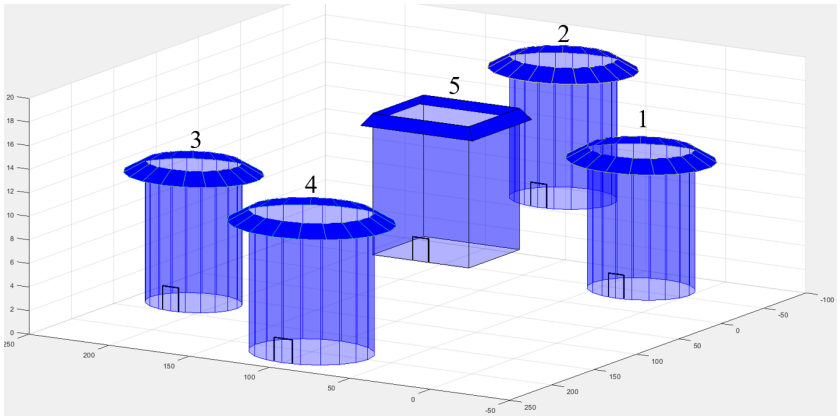
Simulation Results MATLAB validates the simulation of the path planning. The radius and the centre coordinate are inputted to generate the inspection path. The assumptions are as below:

- The speed is 18 m/s, the speed of ascent is 6 m/s, and the descent speed is 4 m/s.
- The UAV hovers when the UAV reaches the point of generating inspection data.
- UAV has been charged fully and operates in excellent condition.

Figure 3.5 demonstrates the flight path of each building, with the points where info is collected marked by "*". The interior and exterior inspection paths are connected through the hall, and it provides the possibility to check the condition of the entrance hall.

The start building is Building 1. From the visited vector and the cost matrix, the order of inspecting the buildings is 1 2 5 4 3, which is determined by Dijkstra's algorithm. The start is the exterior path of Building 1, which is close to the top.

(a) The Tianluokeng Tulou cluster



(b) Model of the cluster

Figure 3.4: The Fujian Tulou

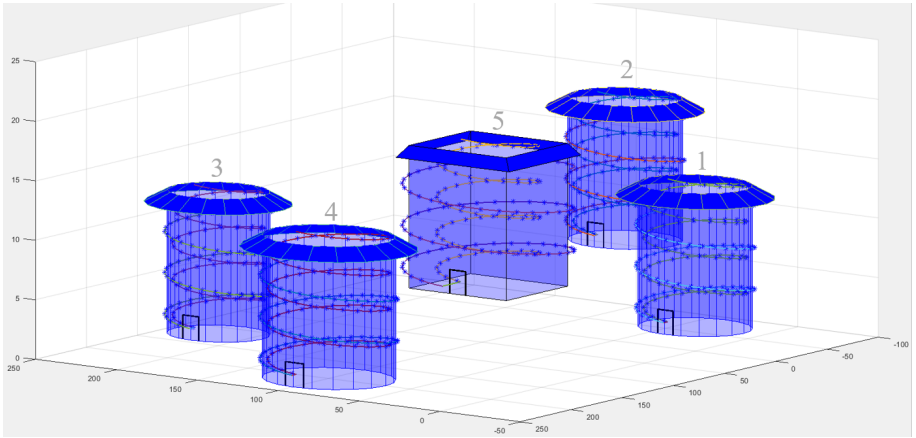


Figure 3.5: The flight path for each building

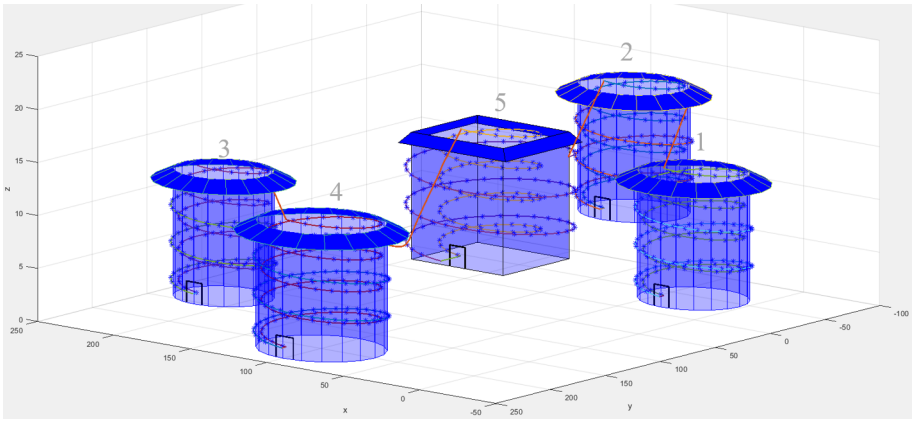


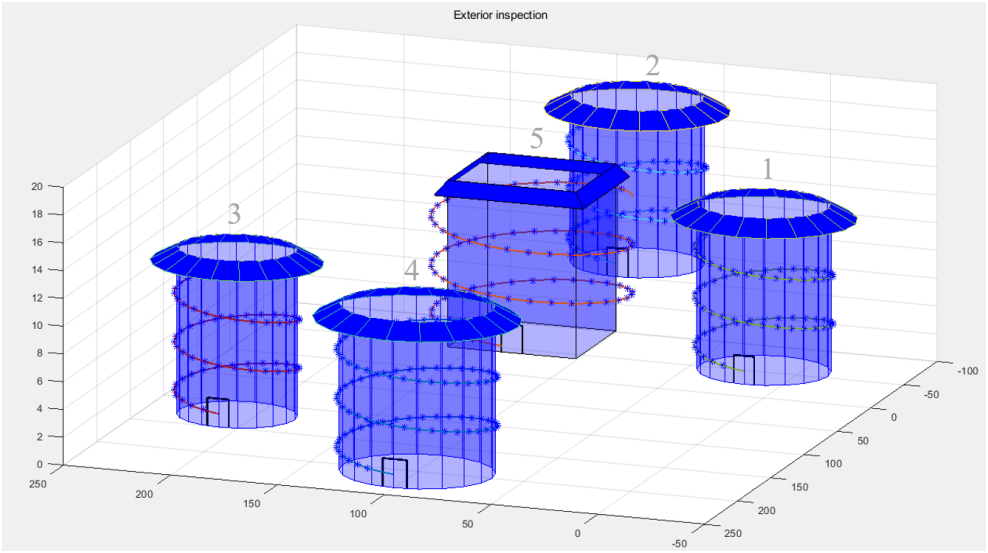
Figure 3.6: The Helix-HPSO path

The Helix-HPSO approach integrates the flight path; the entire flight path is shown in Figure 3.6 and Figure 3.8a. The Helix-HPSO path starts from the exterior inspection path of Building 1, following the interior path inspection path. After reaching the end of the interior path, the UAV flies to the next building defined in the order. After getting to Building 2, the UAV inspects the exterior and interior areas. The loops of inspection keep until the UAV reaches the last building. Table 3.4 lists the details for the generated path.

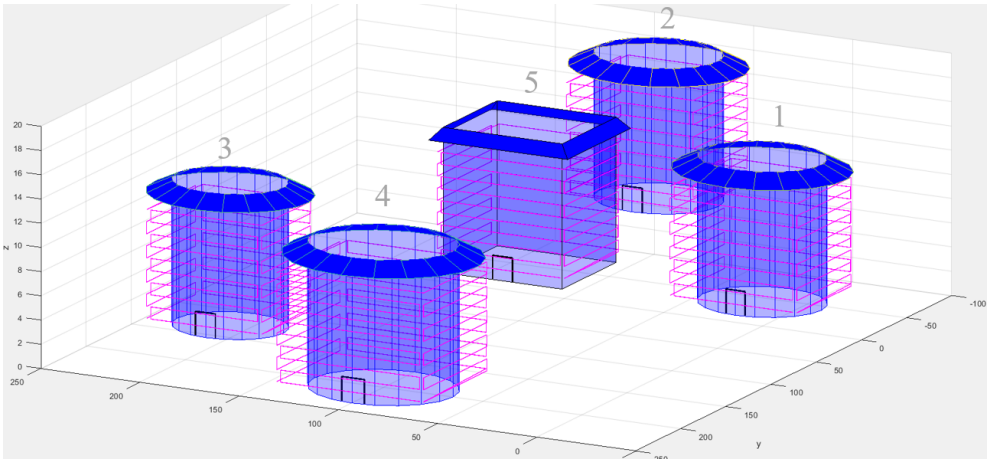
Table 3.4: Details of the Helix-HPSO path

Steps	Description	Path	Start	Destination
1	1 (From exterior to interior)	Building 1 inspection	(-29.97,-7.932,12.48)	(0.196,27,15)
2	From 1 (interior) to 2 (exterior)	Proceed to Building 2	(0.196,27,15)	(44.74,-56.73,17.5)
3	2 (From exterior to interior)	Building 2 inspection	(44.74,-56.73,17.5)	(75.2,-23,20)
4	From 2 (interior) to 5 (exterior)	Proceed to Building 5	(75.2,-23,20)	(46.07,50.23,15.5)
6	5 (From exterior to interior)	Building 5 inspection	(46.07,50.23,15.5)	(89.66,87,17.99)
7	From 5 (interior) to 4 (exterior)	Proceed to Building 4	(89.66,87,17.99)	(65.29,190.4,10.48)
8	4 (From exterior to interior)	Building 4 inspection	(65.29,190.4,10.48)	(100.2,232,13)
9	From 4 (interior) to 3 (exterior)	Proceed to Building 3	(100.2,232,13)	(173.2,141.8,11.46)
10	3 (From exterior to interior)	Building 3 inspection	(173.2,141.8,11.46)	(200.2,174,14)

Comparison of Path Planning for Each Building The back-and-forth path is widely implemented for a building inspection to get the inspection data. It is programmed to compare with the Helix-HPSO approach. The distance between a UAV and a building is less than 2m. The path length of the vertical flight is 1m. The back-and-forth path is only for the exterior inspection without considering the



(a) The exterior inspection paths by Helix-HPSO



(b) The back-and-forth for each building

Figure 3.7: The exterior inspection paths

costs of taking inspection photos, as shown in Figure 3.7b. Figure 3.7a displays the exterior inspection path by defining the data generation points through the proposed Helix-HPSO approach.

The proposed approach implements the helix-shaped path for coverage inspection. Table 3.5 compares the costs of the exterior inspection of each building for different paths. The cost only considers the exterior inspection path, and they are calculated by Equation (3.31), considering the distance, altitude, and time. For getting complete documentation of the building, coverage path planning is required to get complete documentation of the building. If comprehensive interior and exterior coverage are required, traditional back-and-forth paths' costs rise rapidly.

Table 3.5: Costs of the exterior paths

Path	**Building 1**	**Building 2**	**Building 3**	**Building 4**	**Building 5**
The Helix-HPSO path	**236.8405**	**236.8405**	**209.1750**	**280.4919**	**299.7959**
The back-and-forth path	600.1694	600.1694	542.8375	695.7231	600.1694

Comparison of Path Planning Between the Buildings PSO is compared with the proposed HPSO for path planning between the buildings. Figure 3.8 compares the path generated by PSO and HPSO. The HPSO path is marked orange, and the PSO path is marked blue.

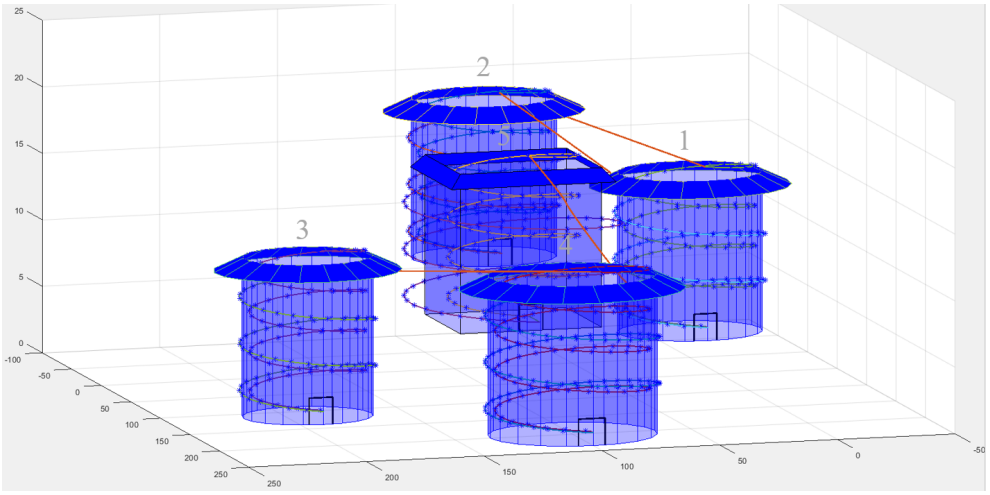
Table 3.6 compares the performance of PSO and HPSO for path planning. If the algorithm gets the best solution ten times, the algorithm terminates. For the path between Buildings 1 and 2, Buildings 2 and 5, and Buildings 4 and 3, PSO and HPSO have the same performance. While for the path between 5 and 4, HPSO has fewer iteration times and costs than PSO. HPSO reduces by around 53.62% iteration times to get the optimal path.

Table 3.6: Comparison of PSO and HPSO paths

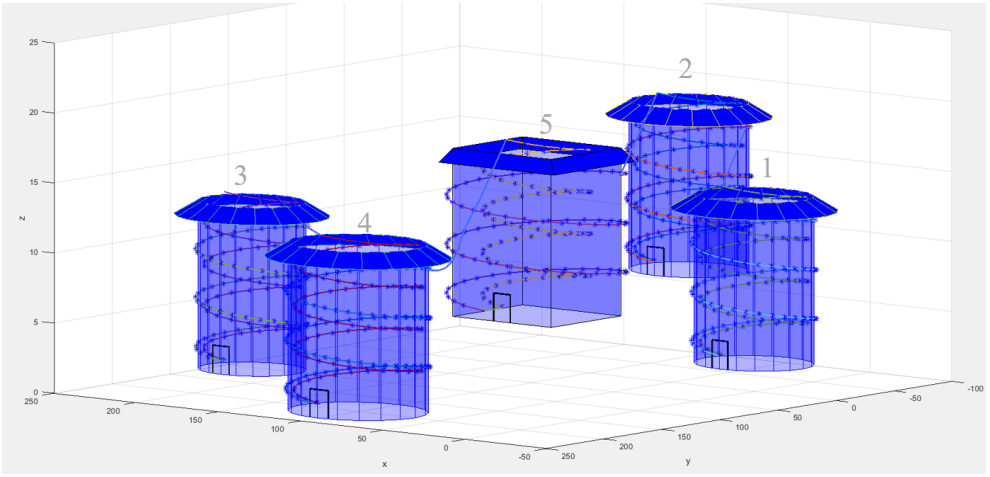
Path		From 1 to 2	From 2 to 5	From 5 to 4	From 4 to 3
The HPSO path	Iteration times	**11**	**11**	**32**	**11**
	Best cost	**47.4371**	**39.4697**	**53.5707**	**58.0246**
The PSO path	Iteration times	**11**	**11**	69	**11**
	Best cost	**47.4371**	**39.4697**	53.5756	**58.0246**

3.2.6 Conclusion of Path Planning for The Multi-building Case

The Helix-HPSO path planning approach is developed for UAV multi-building inspection, considering the paths travelling between buildings. Usually, the building inspection algorithm is designed for one building, while our approach is for a multi-building environment. The proposed method can be used for interior inspection, extending the traditional inspection area. The proposed helix path provides broader coverage and smooth turn angles at lower costs than traditional inspection paths.



(a) The path generated by HPSO



(b) The path generated by PSO

Figure 3.8: The generated paths by HPSO and PSO

Cost function factors include time, distance, and altitude, as altitude change affects energy consumption and speed.

A visited vector and a cost matrix are implemented to determine the order of proceeding inspections for travelling between buildings. The inspiration of HS improves the PSO algorithm, which enhances the search abilities by updating the inertial weight, and the improved global search prevents trapping on the local optima. The proposed HPSO exhibits outstanding performance when dealing with many local optima in the benchmark functions compared to other bio-inspired algorithms. For UAV path planning, the cost functions consider the distance and achieve collision avoidance. The HPSO provides faster convergence than PSO during UAV path planning, so it is used in the path planning between buildings.

Chapter 4

An Intelligence-based Hybrid PSO-SA for Mobile Robot Path Planning in Warehouse

4.1 Introduction

Mobile robots have various applications, such as entertainment, cleaning, surveillance, and object delivery [19]. The requirements of the AGV navigation system in flexible manufacturing systems include versatility, flexibility, no human interaction, scalability, performance, adaptability, and robustness [281], [282]. Path planning, real-time monitoring, task scheduling, and traffic coordination are the primary consideration for AGV operations [283]. The requirements of AGV path planning for enterprises are feasibility and practicability, which consists of no line or rail navigation, low latency and cost, remotely controllable, and precise positioning [282], [284].

The path planning approaches can be categorized into geometric, grid-based, reward-based, random incremental, and Next Best View [285]. Robot path planning algorithms can also be classified as evolutionary and non-evolutionary algorithms [18]. The commonly used robot path planning algorithms consist of the artificial potential field [101], the random search ant colony algorithm, the genetic algorithm [286], grid maps [19], the search algorithms [118], particle swarms [135], and reinforcement

learning and neural network [177].

Multi-criteria decision-making is proposed in [118] for crowd-based path planning in the unknown environment, using the full consistency method and implementing the D* Lite algorithm. D* algorithm extends the A* algorithm, an incremental graph search algorithm, and D* Lite is algorithmically simpler and different from D* for a partially known environment [118]. EA* is used in path planning, then implementing assignment techniques to inform the robots and fault-detection algorithms to handle robots that fail path planning [116]. Symbiotic navigation for multiple robots is proposed in [19], which enables a knowledge-sharing mechanism with the D* algorithm, and for minimum communication, the map is represented by nodes. A linear temporal logic formula and a weighted transition system for high-level mission specification are presented for automatic path planning for multi-robot [287].

Additionally, machine learning is a technology for implementing human learning abilities, and reinforced learning can make a sequence of decisions for robots to achieve goals in complex and uncertain environments. Model-free approaches are implemented in robot motion planning problems, requiring much training data [186]. Convolution Neural Network is combined with Deep q learning for strengthening the learning algorithm to analyse the situations and information in the images with reward function [177]. In reinforcement learning, the Q-learning algorithm can establish interactive relationships and build a dynamic environment without prior knowledge of the environment. An empirical playback mechanism is combined with a Q-learning algorithm in a Deep Q-network for multi-robot path planning. It solves the problems of excessive randomness and slow convergence [185]. Reinforcement learning can be integrated with meta-learning to enhance the generalization ability, and transfer learning is involved in the testing process [186].

The complete coverage and path planning (CCPP) techniques can be classified as online and offline approaches. The online method generates the information from the environment, while the offline algorithms are designed for the known environment. Advanced techniques such as cellular decomposition, grid-based, and topological coverage are introduced [142]. [288] proposes a collaborative CCPP approach for unknown environments with maximizing incremental coverage. The Voronoi di-

agram generates various points as the cores to separate a flat space into multiple areas for robot path planning. The Voronoi diagram is extended to multi-robot path planning and sets a path-priority order for each robot, using the Dijkstra algorithm to generate all navigation points [113].

Evolutionary algorithms are one of the well-researched path-planning approaches. They are inspired by the laws of biological survival and natural phenomena and originated from physics and mathematics [18]. Artificial intelligent algorithms convert the path search problem to functional optimization by realizing the path planning of robots with self-renewal and self-learning abilities [18]. Multi-robot path planning requires addressing the validity issues of non-holonomic constants, dynamic changes in the robotic plan, and memory and execution time complexity [286]. Meta-heuristic and heuristic algorithms achieve efficient local and global search by balancing intensification and diversification [130]. A jumping mechanism particle swarm optimization (PSO) is proposed with a safety gap obstacle avoidance algorithm. This approach uses a fitness function to measure the convergence and then control the update of velocity [18]. The fuzzy inference system and the artificial potential field (APF) are implemented for collision avoidance strategy [101]. A genetic algorithm is modified to handle partitioning and routing sensor-based coverage path planning [289].

The optimization of the evolutionary algorithms has a significant chance of being struck with local optima and getting slow when the dimensionality rises, but the cooperative evolutionary algorithm divides the problem into more minor issues with smaller dimensionality [286]. Evolutionary operators and the improved version of PSO are combined to compute the optimal path for multi-robot [290]. Modified genetic algorithms and improved cooperatively coevolving PSO are introduced in a cooperative path planning approach to address multi-robot persistent coverage [142]. Co-evolutionary grammar-based genetic programming is developed with a maze-like map in [286], and a master evolutionary algorithm achieves overall path optimality. A coevolution-based PSO is presented with evolutionary game theory for a self-adaptive approach, which improves optimization efficiency and guarantees convergence, addressing the stagnation issues and adjusting local and global search abilities [135].

In a continuous environment, the APF generates feasible paths based on a time-efficient deterministic scheme and uses an enhanced genetic algorithm to modify the positions for multi-objective multi-robot path planning [98]. The proposed hybridization of kidney-inspired and sine-cosine algorithm chooses subsequent optimal positions for robots, avoiding collisions with other robots and dynamic obstacles [162]. The genetic algorithm is introduced to optimize the goal points, and the boundary node method and path enhancement method are combined to get an optimal collision-free path [139]. An improved PSO is integrated with the gravitational search algorithm inspired by nature, and the proposed co-evolutionary algorithms maintain the balance between exploitation and exploration [136]. A hybridization of improved PSO and differentially perturbed velocity algorithm is proposed in [130], aiming to minimize the maximum path length and arrival time.

From the literature, path planning algorithms consist of grid-based, reward-based, geometric-based, and evolutionary-based approaches. The grid-based and geometric-based method usually implements graph search or evolutionary algorithms while it wastes the available zones for path planning. The reward-based approach makes a sequence of robot decisions, but it requires enormous computation space, time, and reliable train data. Evolutionary algorithms have robust and straightforward implementations, but they will likely be trapped in local optima or require huge computation space and time for optimization. The co-evolutionary approach can overcome the shortness of each evolutionary algorithm.

This chapter proposes a new hybrid meta-heuristic algorithm for AGV path planning to explore the optimal global solution with enhanced search abilities. It aims to provide an approach for robot path planning and ensure flexibility, scalability, adaptability, performance, and practicability for AGV path planning. It is aimed to reduce the computation space and runtime for generating the path with faster convergence and not getting trapped in the local optima. It also uses coordinators to produce more available zones during path planning. Compared with other well-known evolutionary algorithms, it shows excellent performance in optimization. The approach provides faster convergence and high flexibility in the challenging static environment with the developed cost function for path planning. The proposed approach requires less computation time and iterations to get the optimal global so-

lution in the simulation and experiment. The chapter's main contributions include

- proposing a novel hybrid heuristic algorithm that can obtain globally optimal solutions with great performance and compared with other heuristic algorithms.
- applying the proposed hybrid PSO-SA algorithm for a path planning application, which has faster convergence and less runtime.
- developing cost functions that consider path length, smoothness, and collision avoidance for AGV path planning to provide flexibility, scalability, adaptability, performance, and practicability.

The chapter provides a robot path-planning approach based on the hybrid intelligence algorithm. The chapter is organized as follows. Section 4.2 explains the hybrid PSO-SA algorithm. The simulation and experiment results are provided in Section 4.3 and concluded in Section 4.4.

4.2 Hybrid PSO-SA

4.2.1 Preliminary Knowledge

Particle Swarm Optimization (PSO)

The social behaviour inspires particle swarm optimization (PSO), which is the population-based stochastic optimization approach. As a meta-heuristic optimization approach, it can gain global or near-global optimum solutions [277]. Each particle is treated as a potential solution for PSO, exploring an optimum within the searching space. PSO has various applications, such as optimization problems [291]–[294], robot path planning and navigation [130], [295]–[300], and network applications [301].

PSO is a robust optimization algorithm with fast convergence, but it may get trapped in a local optimal in multi-modal problems. The hybrid meta-heuristic approach is introduced to overcome the trapping issue; combining it with another algorithm can enhance searching and exploring abilities to obtain the global optimum solutions. The velocity and position for particles of PSO are updated in each

iteration as Equations (4.1) and (4.2). PSO algorithm is demonstrated in Algorithm 8. The flowchart of the PSO algorithm is shown in Figure 4.1.

$$v_i^{t+1} = \omega v_i^t + c_1 r_1 (pbest_i^t - x_i^t) + c_2 r_2 (gbest^t - x_i^t) \tag{4.1}$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{4.2}$$

Where v_i^{t+1} denotes the velocity for ith particle in the $t+1$ timeslot, and x_i^{t+1} is the position. Consider ω as the weighting factor, r_1 and r_2 are random numbers, and c_1 and c_2 are cognitive and social parameters. $pbest_i^t$ is the particle's best position, and *gbest* is the global best position for all particles.

Algorithm 8: PSO algorithm

```
Initialization
for iteration = 1 : iteration_max do
    for particle = 1 : particle_max do
        Update velocity and velocity bounds
        Update position
        Evaluation by the cost function
        if pcost_i^t < pbest_i^t then
            Update Personal Best
            if pbest_i^t < gbest^t then
                Update Global Best
            end
        end
    end
    Update Global Best Cost
end
```

Simulated Annealing (SA)

Simulated Annealing (SA) algorithm is proposed to reduce the possibility of getting trapped in one local minimum and accepting other solutions. SA is inspired by the annealing process of crystals, which can reach the minimum energy, and if the

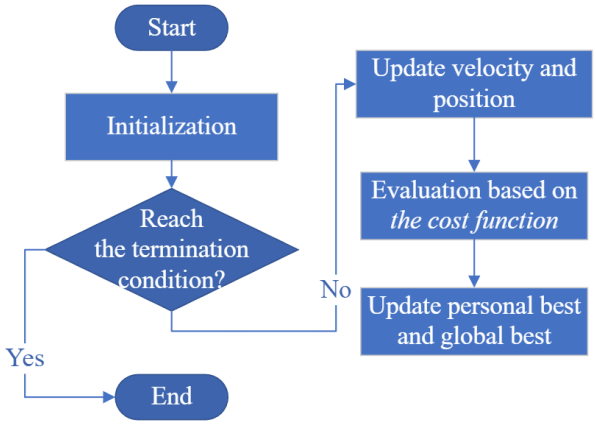


Figure 4.1: The flowchart of PSO algorithm

temperature reduces slower, the energy state will reach lower [116]. SA algorithm is shown in Algorithm 9, and the flowchart of the SA algorithm is indicated in Figure 4.2. The probability of accepting a new solution is as Equation (4.3).

Algorithm 9: SA algorithm

```
Initialization
while t > t_min do
    for particle = 1 : particle_max do
        Generate a new solution
        delta ← cost_new − cost
        if delta <= 0 then
            particle ← particle_new
        else
            p ← exp(−delta/kt)
            if p > rand then
                particle ← particle_new
            end
        end
    end
    t ← t ∗ alpha
end
```

$$\rho = \begin{cases} 1, & newcost \leq cost \\ e^{\frac{newcost-cost}{t}} & newcost > cost \end{cases} \tag{4.3}$$

Where *newcost* is the cost of the new state, and t is the temperature in the SA. ρ indicates p in Algorithm 9 and Figure 4.2. The result of $newcost - cost$ is *delta*.

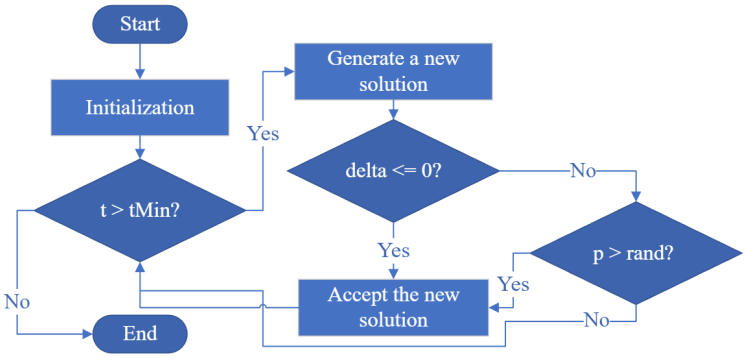


Figure 4.2: The flowchart of SA algorithm

4.2.2 Hybrid PSO-SA

Description

Hybrid PSO-SA is proposed for optimization problems, and the application of path planning is developed. The algorithm of the proposed hybrid PSO-SA is in Algorithm 10. The cost function is designed to evaluate the results; initialization is involved. The generated particles have the initial status, then updating velocities and positions for particles based on Equations (4.1) and (4.2). The cost is estimated by the cost function for each particle and compared with its personal best. Then the swarm gets the best solutions from the particles as the global best. Gaining the statuses of particles and updating the local-oriented and global-oriented best values are inspired by the PSO algorithm.

PSO only accepts the lower-cost solution, which will likely get trapped on local optima. SA usually reaches the maximum iteration times to get the solution. However, the proposed PSO-SA may take the new solution even with a higher cost, and it overcomes the shortness of each algorithm. SA algorithm inspires accepting the new solution, while it updates the local best-oriented value in the proposed algorithm rather than accepting the new solution. The PSO-SA will calculate the probability based on Equations (4.4) and (4.5). The new solution is accepted if the probability is larger than a random number between 0 and 1. It allows getting rid of one local optimum to find the global optimization result.

$$\delta = \frac{pcost_i^t - pbest_i^t}{pbest_i^t} \tag{4.4}$$

$$\rho = e^{-\frac{\delta}{kt}} \tag{4.5}$$

Where $pcost_i^t$ is the current cost of particle i in the t timeslot, and $pbest_i^t$ is the personal best value. δ in Equation (4.5) is calculated by Equation (4.4). *delta* is calculated by δ, and p is calculated by ρ in Algorithm 10.

Algorithm 10: Hybrid PSO-SA algorithm

```
Initialization
for iteration = 1 : iteration_max do
    for particle = 1 : particle_max do
        Update velocity and velocity bounds
        Update position
        Evaluation by the cost function
        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/kt)
            if p > rand then
                pbest_i^t.cost ← pcost_i^t.cost
            end
        end
    end
    Update Global Best Cost
    Update inertia weight
    t ← t ∗ alpha
end
```

Path Planning

The proposed PSO-SA is aimed at path planning, and path planning is formulated as the optimization problem. For the mobile robots' navigation, path planning is the crucial part. We develop the cost function that evaluates path length, collision, and path smoothness.

The path length is one of the primary considerations during path planning in our scenario. The position of one particle is (x, y), and the next position is (x_{k+1}, y_{k+1}). The path consists of n particles, and the cost function of path length is as Equation (4.6).

$$f_{length} = \sum_{k=1}^{n} \sqrt{(x_{k+1} - x_k)^2 + (y_{k+1} - y_k)^2} \tag{4.6}$$

Where n is the number of particles, k is the current particle, and $k+1$ is the following particle.

For robots' operation, a smoother path is easier to follow and consumes less energy. The smoothness of the path is calculated by the slopes from particles. The cost function of smoothness is calculated by Equations (4.7) and (4.8).

$$s_k = \begin{cases} 0, & if\ x_{k+1} - x_k = 0 \\ \left| \frac{y_{k+1} - y_k}{x_{k+1} - x_k} \right| & else \end{cases} \tag{4.7}$$

$$f_{smoothness} = \sum_{k=1}^{n} s_k \tag{4.8}$$

Where s_k is the smoothness of the kth particle.

The cost function of collision is evaluated by Equations (4.9) and (4.10). The particle k has the collision cost represented by c_k.

$$c_k = \sum_{c=1}^{j} r_c - d_c,\ if\ d_c < r_c \tag{4.9}$$

$$f_{collision} = \sum_{k=1}^{n} c_k \tag{4.10}$$

Where r_c is the influence radius of the obstacle, and d_c is the distance from the particle to the centre of the obstacle for calculating c_k. j is the number of obstacles. The cost function calculates every collided obstacle with the current particle and then sums it up as c_k.

The cost function considers each factor as Equation (4.11).

$$f_{cost} = w_1 \cdot f_{length} + w_2 \cdot f_{smoothness} + w_3 \cdot f_{collision} \tag{4.11}$$

Where w_1, w_2, w_3 are the weight factors for each cost function, and their sum is 1.

The scenario is AGV path planning in the warehouse. The flowchart of PSO-SA path planning is shown in Figure 4.3. Using a warehouse map to generate the path, then developing the cost function and initialization for the parameters and the swarm, updating the particles' states, and evaluating the path based on the cost function. The initialization of parameters includes setting the maximum iterations, initial temperature, population size, inertia weight and damping ratio, temperature reduction rate, learning coefficients, and bounds. The complexity of the application determines the weights and parameters, depending on the considered factors.

If the new cost is more minor, the personal best of the current particle is updated; otherwise, the probability is involved. If the cost is less than the global best, the cost becomes the global best solution. The optimal global path is determined if the global best cost occurs ten times continuously. The iterations can be terminated in advance once the globally optimal path is determined.

4.3 Experiment

4.3.1 Simulation with Test Functions

The proposed PSO-SA is validated through tests with MATLAB and compared with various optimization algorithms, including PSO, SA, Harmony Search (HS) [302], Firefly Algorithm (FA) [280], Artificial Bee Colony (ABC) [303], and Genetic Algorithm (GA) [279]. The benchmark test functions are the standard optimization problems to test the performance of meta-heuristic algorithms. The test functions

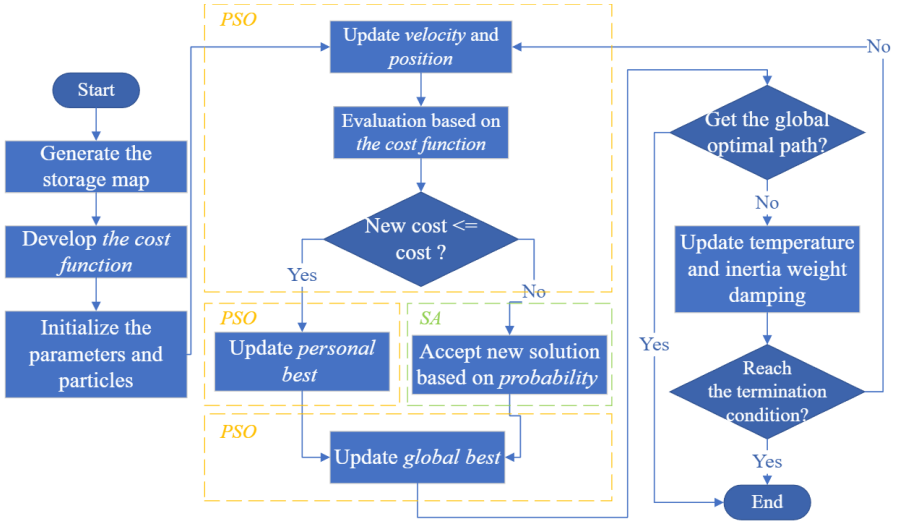


Figure 4.3: The flowchart of path planning by Hybrid PSO-SA

are listed in Table 4.1, and the characters of each function are listed. The graph of valley-shaped, bowl-shaped, plate-shaped, and Steep Ridges/Drops functions are like the valley, bowl, plate, and steep ridges/drops, respectively. The many local minima functions have many local optima. The number of iterations is 200, and each algorithm runs 20 times.

Table 4.2 compares the mean iteration times for each test function when the algorithms get the best global solution and list the final fitness value for the algorithms. The best mean value for each test function is highlighted. The results show that PSO-SA uses the least mean iteration times to get the best answer in most test functions. For Rosenbrock $f_1(x)$ and Michalewicz function $f_7(x)$, PSO-SA and PSO can easily find the optimal solution.

Table 4.3 compares the run time for the algorithms, with the minimum one highlighted. It can draw that the proposed PSO-SA usually has less mean run time for most functions, except the functions $f_4(x)$ and $f_6(x)$, where PSO has a bit less run time than PSO-SA. The algorithm HS, FA, ABC and GA are very slow compared to PSO, SA and PSO-SA. As mobile robot path planning in real-time, the algorithm's speed is critical for practical application.

Figure 4.4 presents the mean fitness value's convergence curve for each test function in 200 iterations. PSO-SA can get the optimal global solution in most cases. The runtime and iteration numbers are also considered. It can conclude that the proposed

Table 4.1: Test functions

Function	Type	Name	Test function
$f_1(x)$	Valley-Shaped	Rosenbrock	$f_1(x) = \sum_{i=1}^{d-1}[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$
$f_2(x)$	Many Local Minima	Ackley	$f_2(x) = -a\ exp\left(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right)$ $-exp\left(\sqrt{\frac{1}{d}\sum_{i=1}^{d} cos(cx_i)}\right) + a + exp(1)$
$f_3(x)$	Many Local Minima	Levy	$f_3(x) = sin^2(\pi w_1)$ $+\sum_{i=1}^{d-1}(w_i - 1)^2[1 + 10sin^2(\pi\omega_i + 1)]$ $+(w_d - 1)^2[1 + sin^2(2\pi\omega_d)]$ $where\ w_i = 1 + \frac{x_i - 1}{4}, for\ all\ i = 1, ..., d$
$f_4(x)$	Bowl-Shaped	Sphere	$f_4(x) = \sum_{i=1}^{d} x_i^2$
$f_5(x)$	Bowl-Shaped	Sum squares	$f_5(x) = \sum_{i=1}^{d} ix_i^2$
$f_6(x)$	Plate-Shaped	Zakharov	$f_6(x) = \sum_{i=1}^{d} x_i^2 + (\sum_{i=1}^{d} 0.5ix_i^2)^2 + (\sum_{i=1}^{d} 0.5ix_i^2)^4$
$f_7(x)$	Steep Ridges/Drops	Michalewicz	$f_7(x) = -\sum_{i=1}^{d} sin(x_i)sin^{2m}(\frac{ix_i^2}{\pi})$

PSO-SA has an excellent performance in most optimization problems with faster convergence and less consumed time.

4.3.2 Path Planning

Simulation

The environment is generated from an existing warehouse, as shown in Figure 4.5, and walls and pillars are exhibited. The AGVs depart from Area G to different storage rooms for operation and return to Area G from storage rooms after completing the tasks. Black dots indicate the pillars, and the walls are annotated with black.

The maximum number of iterations is 150, and the population size is 150. The population size should be within $[100, 200]$ for the path planning problem. For simulation, we tested the population size for the path from Area G to E as the size is 100, 150, and 200. When the population size is 150, PSO-SA has the best performance, with the least runtime for each iteration and fewer iteration times.

Table 4.2: Mean iteration times and fitness value

Function		PSO-SA	PSO	SA	HS	FA	ABC	GA
$f_1(x)$	Iterations	**1.00**	**1.00**	200.00	184.75	197.55	131.95	199.10
	Value	**0.00**	**0.00**	-0.21	1.98	0.40	0.77	6.97
$f_2(x)$	Iterations	181.20	187.60	193.20	**179.35**	183.25	196.10	199.30
	Value	0.18	**0.00**	0.05	0.01	**0.00**	**0.00**	**0.00**
$f_3(x)$	Iterations	60.75	**48.60**	199.90	181.35	193.50	196.00	199.40
	Value	0.02	**0.00**	0.83	**0.00**	**0.00**	**0.00**	**0.00**
$f_4(x)$	Iterations	**173.25**	188.50	199.85	185.05	186.95	197.60	199.45
	Value	**0.00**	**0.00**	0.03	**0.00**	**0.00**	**0.00**	**0.00**
$f_5(x)$	Iterations	**180.10**	188.00	199.80	186.20	189.70	197.15	199.45
	Value	**0.00**	**0.00**	0.06	**0.00**	**0.00**	**0.00**	**0.00**
$f_6(x)$	Iterations	**178.15**	187.20	199.80	193.85	189.95	182.15	198.85
	Value	**0.00**	**0.00**	0.05	0.02	**0.00**	**0.00**	0.14
$f_7(x)$	Iterations	1.05	**1.00**	1.30	185.60	192.70	114.70	190.70
	Value	**-0.89**	-0.99	2.19	-4.88	-4.45	-3.47	-9.24

The simulation generates the paths from Area G to Area E and Area F and the path from Area D to Area G by PSO and PSO-SA. The start and target position, the best costs, the mean runtime for each iteration, and the iteration times for the paths are shown in Table 4.4. The cost functions (4.6) – (4.11) are defined to evaluate the path length, collision, and smoothness. Because the path length and collision are the primary considerations, w_1 is set as 0.5, w_2 is set as 0.4, and w_3 is set as 0.1.

Table 4.4 shows that PSO-SA performs better in path planning than PSO. Paths for PSO and PSO-SA are shown in Figure 4.6, and the convergence curve of PSO-SA is shown in Figure 4.7. The areas, the source and target positions, and the directions for the paths are marked. It can draw that the proposed algorithm can get the optimal way with an outstanding performance by Table 4.4 and Figure 4.6. The algorithm terminates when it gets the best cost ten times and records the convergence curve. For the paths, the proposed PSO-SA has less runtime than 30.50%, 51.68%, and 34.43%, respectively. It also achieves path planning with fewer

Table 4.3: Runtime of each algorithm

Runtime (ms)		PSO-SA	PSO	SA	HS	FA	ABC	GA
$f_1(x)$	Mean	**3.25**	4.29	4.88	115.15	933.90	641.93	500.12
	Std. dev	**3.05**	3.52	1.67	21.43	28.60	78.29	104.81
$f_2(x)$	Mean	**4.31**	7.61	14.24	116.18	944.02	660.24	465.68
	Std. dev	**2.65**	2.79	12.75	18.03	64.44	39.11	56.01
$f_3(x)$	Mean	**3.99**	6.41	30.08	149.02	1183.89	853.02	603.02
	Std. dev	**2.01**	3.04	4.57	24.94	47.26	45.79	49.97
$f_4(x)$	Mean	4.18	**4.16**	8.43	125.35	925.05	675.87	479.10
	Std. dev	2.59	**2.55**	5.79	21.81	45.72	43.05	51.24
$f_5(x)$	Mean	**4.03**	4.17	6.86	119.92	947.17	652.37	471.05
	Std. dev	2.72	**2.57**	5.79	17.87	50.20	51.32	38.50
$f_6(x)$	Mean	4.57	**4.12**	10.34	115.60	901.46	638.35	478.45
	Std. dev	0.59	**0.35**	1.30	14.09	41.23	21.30	20.71
$f_7(x)$	Mean	**7.08**	11.79	12.33	127.69	981.54	686.64	528.40
	Std. dev	**2.94**	3.42	3.32	22.96	35.61	52.84	52.18

iteration times and better costs than PSO in most situations.

Experiment

The test scenario is shown in Figure 4.8. The AGVs are operating at Area A for moving goods to and from the gate of other areas. The three-dimensional storage system is assumed to be utilized in Area B to Area F. The AGVs depart from Area A and return to Area A for parking once they finish the tasks.

The experiment employed a robot with a Decawave Ultra-Wide Band (UWB) sensor and Raspberry Pi, as shown in Figure 4.9. UWB is the positioning sensor that can provide centimetre-level accuracy for the indoor environment. The robot follows the defined path for collecting position data, and the arrow of each path indicates the direction. The robot departs from Area A to the storage room for inbound delivery and moves to Area A for outbound delivery.

There are some obstacles in the simulation to create a more challenging path-

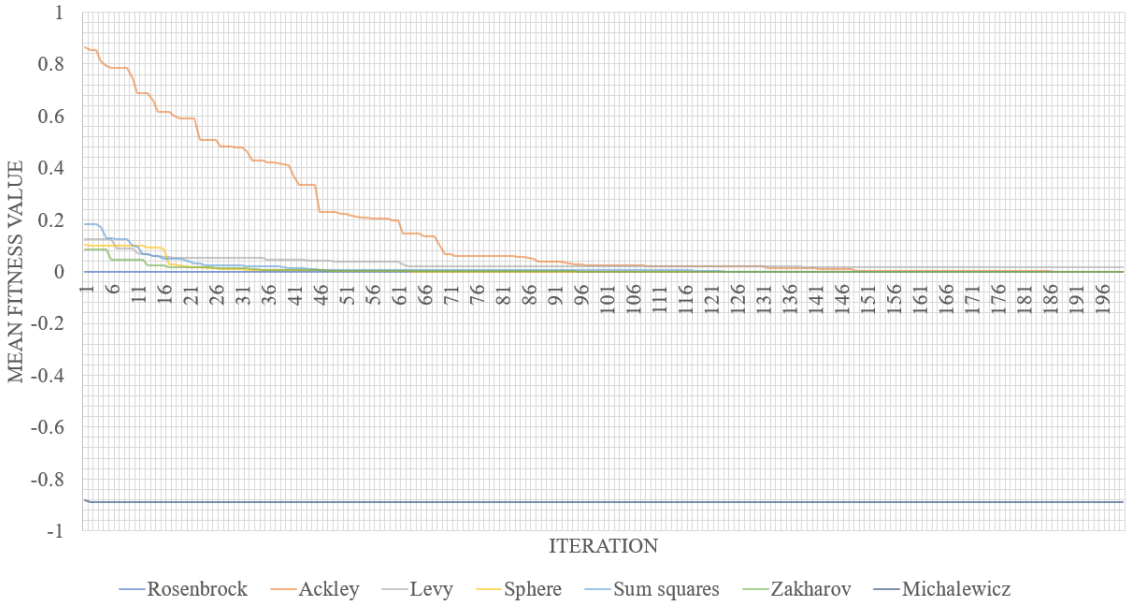


Figure 4.4: Convergence curve for PSO-SA

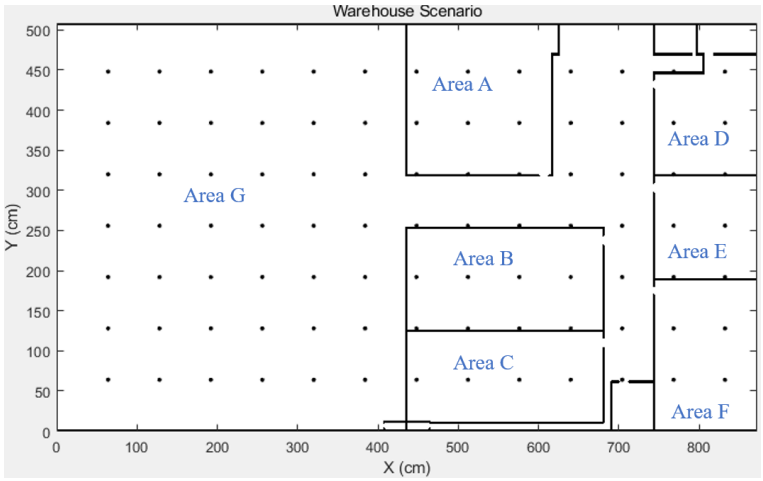
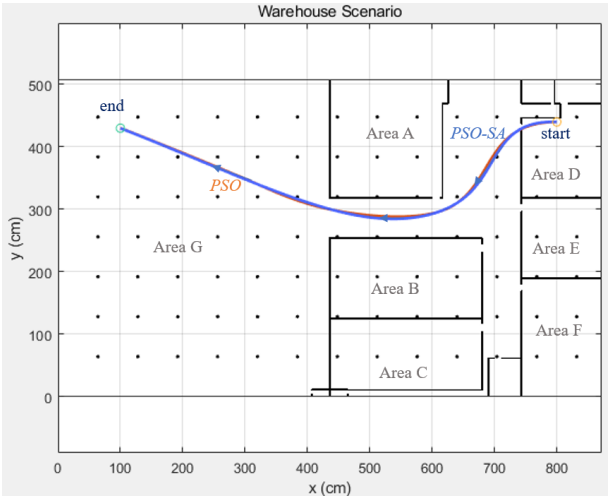


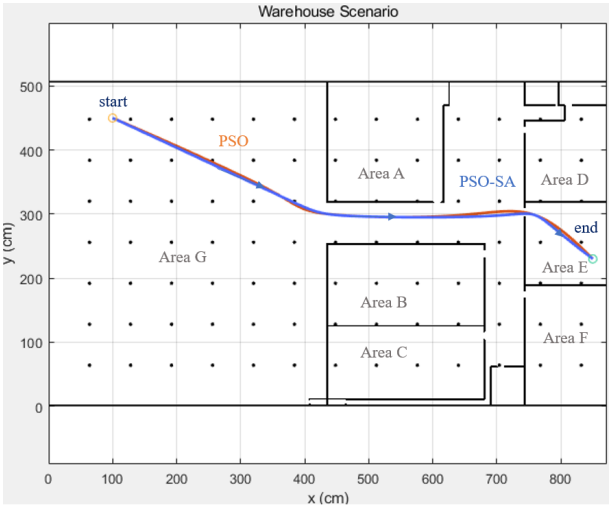
Figure 4.5: The scenario of a warehouse

planning environment. Table 4.5 lists the start and target positions and the best cost for each path. The maximum number of iterations is set as 150. Figure 4.10 exhibits the path and the experiment data in the map, and the simulated path is blue, while the UWB positions are marked yellow. The start position is highlighted in orange, and the target is highlighted in blue. The UWB positions have measurement bias at the centimetre level, and the bias correction would be the future work.

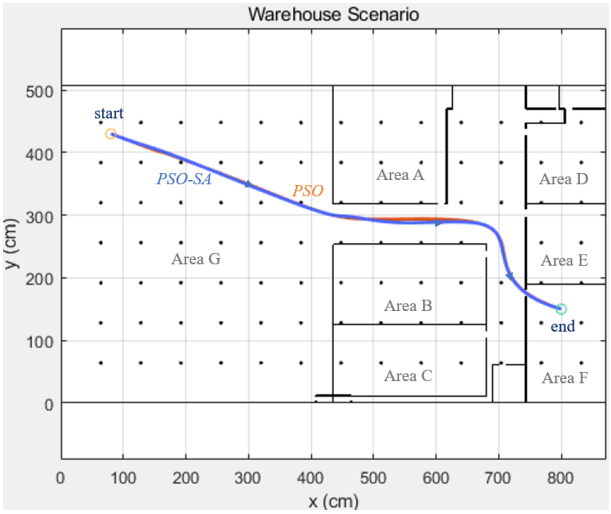
The proposed PSO-SA evaluates the cost for the path in each iteration and reduces the global best to get the optimal path. If the exact best cost of the optimal path



(a) From D to G



(b) From G to E



(c) From G to F

Figure 4.6: The paths for PSO and PSO-SA

Table 4.4: Performance measurements of the simulated paths

Path	Start position	End position	Algorithm	Mean runtime (s)	Iteration times	Best Cost
From D to G	(800,440)	(100,430)	PSO-SA	**1.0915**	**69**	400.1210
			PSO	1.5704	129	**398.9587**
From G to E	(100,450)	(850,230)	PSO-SA	**1.0301**	**74**	**406.4598**
			PSO	2.1319	99	407.4978
From G to F	(80,430)	(800,150)	PSO-SA	**1.0922**	**82**	**415.2469**
			PSO	1.6658	97	415.4940

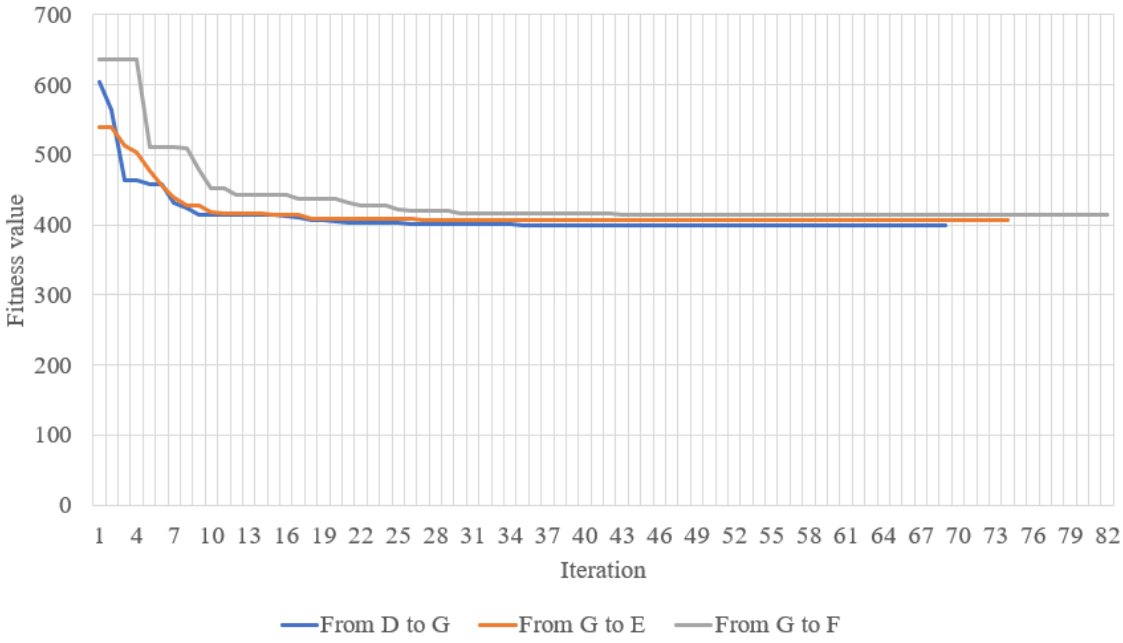


Figure 4.7: The convergence curve of PSO-SA path planning

occurs ten times continuously, the algorithm will treat the solution as the best solution and then terminate the iterations. The termination threshold is selected based on the simulation results of the optimization problem. If a solution occurs continuously, acting as the optimal solution is highly possible. In this simulation, the maximum iteration number is around 70, and the minimum iteration number is about 60.

4.4 Conclusion

The hybrid PSO-SA algorithm improves the PSO algorithm by jumping out of the local optimum to get the best global solution with inspiration from the SA algorithm.

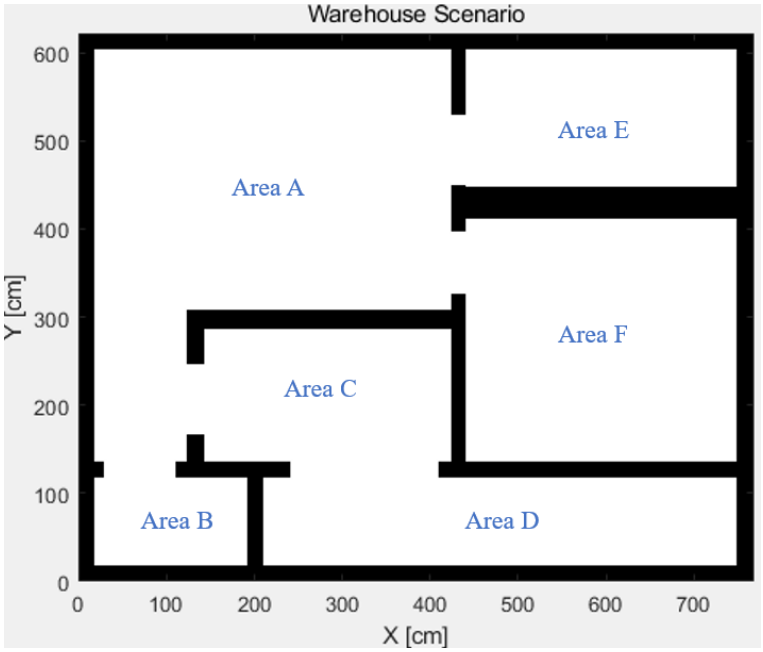


Figure 4.8: The scenario of the test side

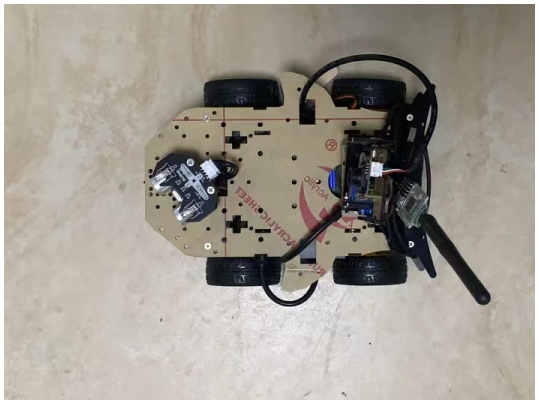


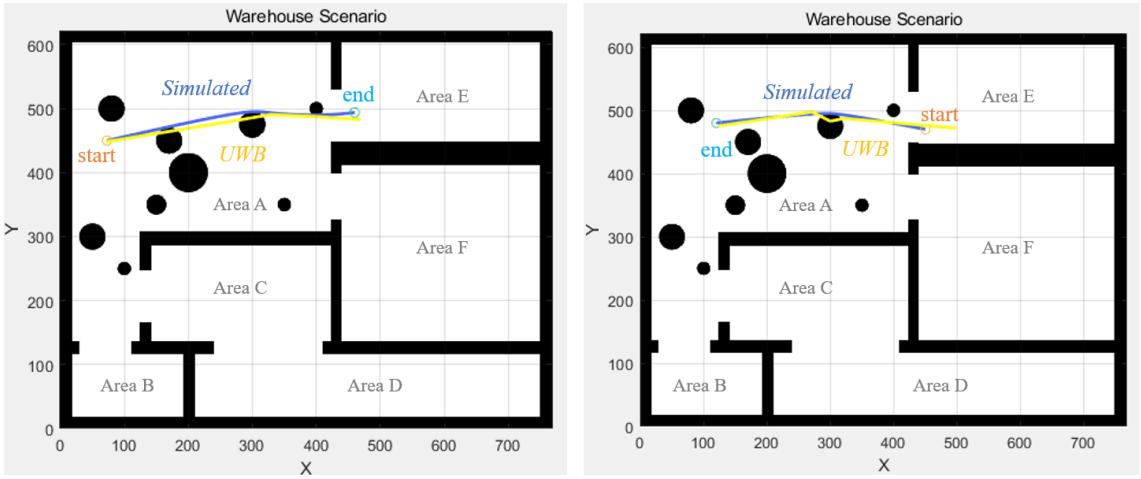
Figure 4.9: Robot with UWB

It achieves enhanced performance by updating the particles' personal best variable probabilistically. It is compared with well-known evolutionary algorithms, including HS, GA, FA, and ABC, by benchmark functions. The proposed PSO-SA algorithm has less 3%, 35%, 40%, 42%, 36%, 44% mean iterations times than PSO, SA, HS, FA, ABC, and GA algorithms, respectively. For the runtime of each algorithm, the proposed PSO-SA algorithm has less 26%, 64%, 96%, 100%, 99% and 99% mean values than PSO, SA, HS, FA, ABC, and GA algorithms respectively. The result indicates that it has faster convergence, high accuracy, and less runtime and iterations to get the best solution.

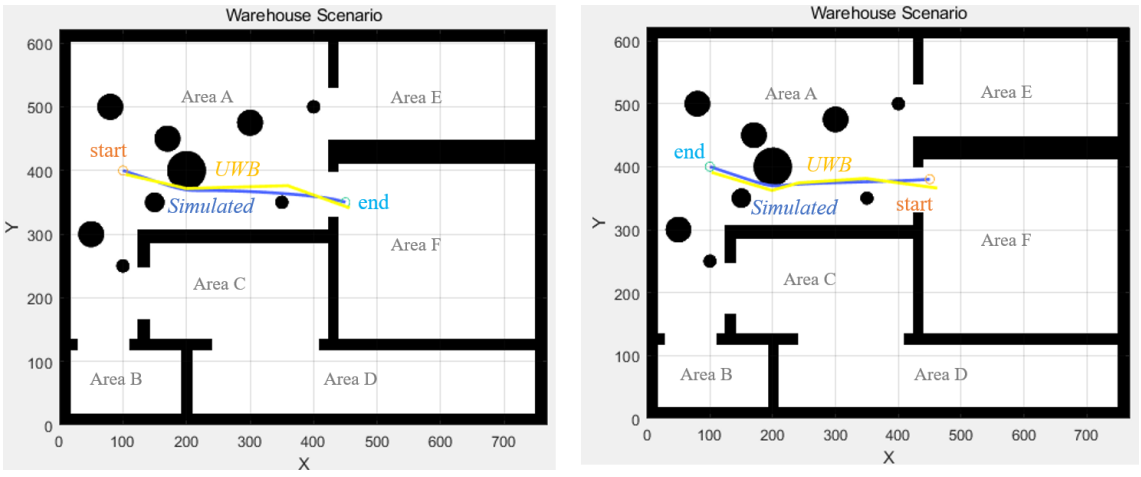
The evolutionary-based approach is the primary path-planning approach, providing

Table 4.5: Simulated paths

Path	Start	Target	Best Cost
From A to E	(73,450)	(460,494)	196.9564
From A to F	(100,400)	(450,350)	179.3558
From F to A	(450,380)	(100,400)	178.5797
From E to A	(450,470)	(120,480)	167.2975



(a) From A to E and from E to A



(b) From A to F and from F to A

Figure 4.10: The paths for UWB

flexibility and scalability. The proposed hybrid PSO-SA approach is used for AGV path planning. Collision avoidance, smoothness, and path length are considered the cost function for the optimal path. The approach has been compared with the PSO for path planning. The model is validated through the storage/warehouse scenario with simulation and experiment, and it can obtain the best path with improved convergence performance. The proposed algorithm can be adapted to different environments with the developed cost function and can be applied to more robots easily with robustness. The approach will be modified to adapt to the dynamic environment with moving obstacles as the future work. Distance sensors or visual sensors may be employed to detect obstacles. Implementing multiple robots' collaboration could also be a further consideration with real-time implementation.

Chapter 5

A Dual-Layer Weight-Leader-Vicsek-Model for Path Planning of Multiple Automatic Guided Vehicles in Warehouse

5.1 Introduction

With the development of robotic technologies, mobile robots are implemented in commerce and industry. Automated Guided Vehicles (AGVs) enhance transportation efficiency with less cost [38], [43]. They are utilized in the industry as a part of industrial intelligence, intelligent logistics, and intelligent factories [37], [43]. AGVs comprise the industry's unit load vehicles, towing vehicles, forklifts, and pallet trucks [47]. AGVs are applied to diminish labour costs and improve safety for the high demands in a production environment [45], [46]. Servo handling systems, warehousing systems, logistics, and storage industries employ AGVs in various areas [46], [49], including transportation, transhipment, distribution, and material handing in manufacturing [3].

AGVs can transfer products at high speeds in a chaotic situation. They are implemented in production lines with flexible development in modern manufacturing, incorporating automated intelligent control systems [46]–[48]. With sensors' help, detecting obstacles and automatically eliminating problems ensure the intelligence and adaptability of AGVs [46]. AGV navigation in the industrial environment usually implements the fixed line or the coupled approach, which adapts the single-robot navigation methods. It lacks the flexibility and the possibility of real-time implementation. It has highly demanded that AGV can have a quick and flexible route setting and adapt to the dynamic environment. The motivation of this study is to provide fast computed path planning with flexibility and scalability for multi-AGV systems, addressing most situations.

This chapter introduces a new algorithm for this purpose. Its main contribution is as below:

- A novel path planning approach for multi-AGV systems based on the improved Vicsek model with a leader-follower structure. The bio-inspired approaches are widely used in multi-robot path planning. The Vicsek model is adapted for this case because the AGVs aim to move as a swarm operation and provide quick computation for the entire system.
- Offering the fast path setting for multiple AGVs in one calculation step, which differs from the other path planning algorithms that repeat the algorithms for every robot.
- For real-time implementation, it provides faster computation and adaption to the environment.

A set of virtual leaders are introduced in the AGV swarms to navigate all the AGVs, which combines search and intelligent algorithm advantages. Biological patterns and the leader's principle are the main concepts in this model, and it integrates the coupled and decoupled approach. The hybrid centralized decentralized is proposed for determining the leaders' path, providing flexibility for AGVs. Each AGV follower collects data from its neighbours and is led by the virtual leaders without the restriction of the current group. One of the significant advantages of the proposed multi-AGV path planning system is that it requires less computational load for

real-time implementation.

Leader-Vicsek-Model was published in [235], and this chapter is an extension. The published paper introduces the concept of the novel Weight-Leader-Vicsek-Model, but it uses simple, straightforward path planning for the leader and traditional Vicsek update equations for followers. Nevertheless, this chapter improves the model as an enhanced Weight-Leader-Vicsek-Model (WLVM) with a dual layer. It generates the dynamic virtual leader by the hybrid A* algorithm and uses a start-destination matrix to determine the swarms' integration, and collision avoidance is achieved by priority. Also, the followers' position and angle updates are improved by adding the weight for the average angle and considering the current leader's angle. The simulations are conducted in a warehouse scenario, and WLVM is compared with the RRT* algorithm.

This chapter offers a multi-AGV path planning approach for optimizing automatic transportation in commercial or industrial warehouses. The paper is organized as follows. Section 5.2 introduces the problem statement. Section 5.3 reviews path planning algorithms and the multi-AGV navigation approaches. Weight-Leader-Vicsek-Model is proposed in Section 5.4 for multi-AGV path planning and navigation. Experiment results are demonstrated in Section 5.5 to validate the approach, and the conclusion is in Section 5.6.

5.2 Problem Statement

The basic components of AGV path planning include the start, the target, and the environment. Path planning aims to generate the path from the start to the target without collisions. Figure 5.1 indicates an example of AGV path planning. As the occupancy map, the map uses the binary number to represent the locations in a 2D space. It can be transformed into a grid map or nodes for performing the heuristic or other path-planning algorithms. The obstacles or the walls are annotated in black.

The AGV is supposed to move from the start to the target, and green points demonstrate the path points. The obstacles or walls are treated as static obstacles, and the generated path should not be overlapped with the static obstacles. The path

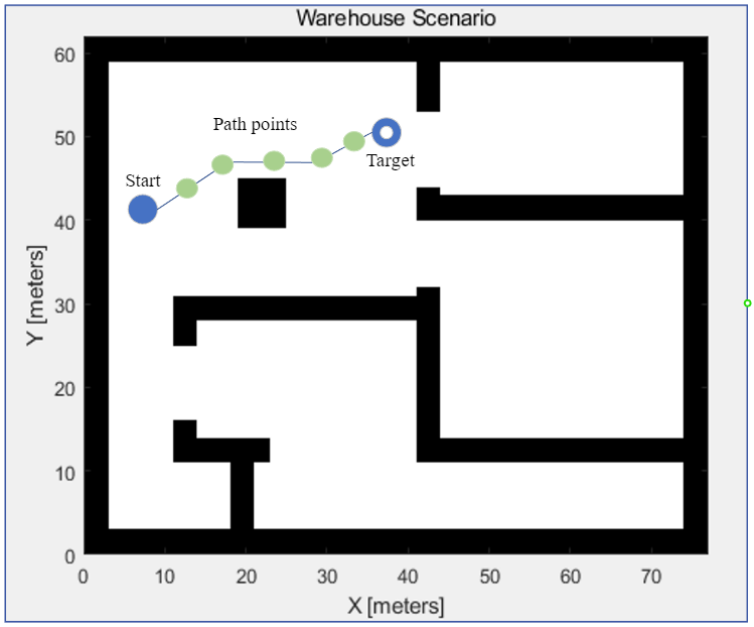


Figure 5.1: Path planning in the map

points are represented by $r_1^k, r_2^k, \ldots, r_n^k$. r is the position of the current iteration k for nth AGV, and r_n^k is (x_n^k, y_n^k).

Collision and deadlock avoidance is necessary for the multi-AGV system. Each AGV is assumed to communicate with other AGVs treated as dynamic obstacles. The deadlock of AGVs should be avoided for great system performance, and the adjusting progress is based on priority. The adjusted position of AGV n is represented by r_{n-new}.

5.3 Related Work

The Automatic Guided Vehicle (AGV) plays a crucial role in the intelligent transportation system. AGV is the primary automated equipment that carries materials and processes unmanned distribution and sorting in an unmanned storage environment [304]. Path planning has been the most crucial consideration of mobile robot navigation, which plans the path from the start to the target for mobile robots [37].

Obstacle avoidance functions must be developed to operate AGVs when considering dynamical limitations and dynamical safety [38], [39]. Meta-heuristic-based methods [305], graph-search-based methods [43], [44], mathematical optimization-based

methods [38], and potential field and navigation-based methods [40] are the four main categories for navigation algorithms.

Additionally, adapting the classic graph search algorithm is implemented widely for AGV path planning, such as the A* and Dijkstra [306]. Different types of algorithms can be combined, for example, the Dijkstra algorithm is for initial static path planning, and the virtual potential function algorithm is for dynamic path planning [42]. Sampling-based methods are also significant for single AGV path planning, such as the Voronoi graph and rapidly exploring random trees (RRT) methods [307].

The mathematical optimization-based approach consists of open-loop and closed-loop strategies [38]. A nonlinear model predictive control (MPC) algorithm has been proposed for large-size AGVs with onboard LIDAR sensors and a 14 DoF vehicle dynamics model [38]. Artificial potential field (APF) methods [308] and the probabilistic road map [309] are also proposed for planning. The AGV path planning algorithms introduced above are mainly focused on individual AGVs. For the venues with multi AGVs operated simultaneously, the dynamic environment due to other AGVs and people needs to be handled safely and efficiently.

Multi-AGV systems have become more popular because of their powerful task-solving complexity, continuous operation, reduced maintenance and operational costs, broad convergence, flexibility, and versatility [219], [310]. Multi-AGV optimization scheduling can improve the logistics transportation system structure and system operation efficiency and reduce transportation costs [311], [312].

Finding the best path quickly with avoiding collision is worth studying in AGV operation [313]. AGVs obtain the paths from the multi-AGV scheduling system, and they sense the surroundings independently and communicate with others by sending poses [314]. Routing, scheduling, and layout are the main factors to be considered for designing and controlling the AGV system [312]. AGVs are usually guided by optical, electromagnetic, and laser navigation technologies or combinations of them, following the arranged path and avoiding collisions [315]. The research on multi-AGV routing can be classified as semi-dynamic, fully-dynamic, and static routing [316].

Vehicle dispatching, positioning, vehicle routing, scheduling, and collision and deadlock avoidance are considered for designing vehicle transportation systems [317]. Current multi-AGV systems commonly implement the centralized control architecture to perform tasks, such as motion coordination, mission allocation, and path planning [318]. The centralized control system is referred to as the warehouse management system (WMS) [13]. However, each AGV plans its path for the distributed approaches and resolves deadlocks or collisions by communicating with its neighbours. Vehicle autonomy and distributed computation are characteristics of decentralized methods [219].

Moreover, a decentralized approach for determining the shortest paths and motion coordination based on nonholonomic vehicle constraints is presented in [219]. A regional control model is introduced for distributed control for the multi-AGV system in [319] to minimize the complexity of scheduling issues. An ant-agent optimized by a repulsive potential field is developed to combine centralized and decentralized control and avoid path conflicts with stability and efficiency [316]. A multi-AGV path planning method improves ant colony algorithms according to prioritized planning, considering battery management in [320] as a decentralized algorithm.

The metaheuristic algorithms are widely used for optimization problems for AGV systems, such as task allocation [312] and path planning [321]. Multi-agent path planning algorithms can be divided into rule-based, search-based, and learning-based [322]. Rule-based algorithms use mature solutions for path planning by transferring the problem to other problems; search-based algorithms implement heuristic search algorithms and are classified as decoupled search, and coupled search algorithms [138], and learning-based algorithms get optimal solutions from suboptimal solutions [322].

Additionally, a multi-AGV probabilistic time-constrained based path planning algorithm is based on the A* heuristic algorithm with dynamic stochastic network theory in [310]. An improved A* path planning algorithm is introduced for a grid-shaped network, ensuring locating and execution of motion commands [313]. The unidirectional directed graph method combined with the A* algorithm for AGV path planning in a multi-AGV scheduling system is presented [323].

Heuristic information and elastic time window are considered in the improved ant colony algorithm [324], and the conflict resolution strategies are based on the priority of AGV task scheduling. A hybrid genetic algorithm-particle swarm optimization is proposed for multi-AGV path planning with a fuzzy logic controller in [138], combining scheduling and path planning. A genetic algorithm is improved to consider the highest charging utilization rate and the shortest path to plan the optimal path for multi-AGV [304].

Deadlock avoidance is the primary consideration during multi-AGV path planning. Nodes describe the physical locations, while the grids are independent spaces in the environment [311]. The node-based coordination strategies strictly avoid the AGVs occupying a common node. [311] proposes the deadlock strategies by a combination of nodes and grids. A structural online control policy is proposed for multi-AGV deadlock resolution based on analysing the system as discrete events [325]. The topological graph and roadmap work for the AGVs' subsequent coordination by local negotiation and shared resources as a holistic approach in industrial warehouses [217].

However, the multi-agent algorithms plan the path independently and lack consideration of moving obstacles and real-time implementation. Most evolution-based and swarm-based algorithms are bio-inspiration, and the biological pattern is considered when developing the new algorithm. This paper proposes a Weight-Leader-Vicsek-Model algorithm, which incorporates the advantages of decentralized and centralized approaches. Each AGV collects data from relevant AGVs, determines its path, and achieves collision avoidance. At the same time, a central decision-maker assigns the multi-AGV groups and virtual leaders for the defined groups. AGV control variables can be gained with faster computational speed and less complexity. This model provides path planning functionality simultaneously for grouped AGVs and treats each group as different swarms.

5.4 WLVM with Virtual Leaders and Weight

5.4.1 Preliminary Knowledge: Vicsek Model

The Vicsek model is introduced for particles with biologically motivated interaction with self-ordered motion and purpose in [326]. The Vicsek model can be expressed by Equations (5.1) – (5.2), and the algorithm is shown in Algorithm 11. Biological subjects spin in the same direction and move as their neighbourhood for interaction in an $L * L$ square region [326], [327]. For a multi-agent system, the Vicsek model has been improved by taking a fixed number of neighbours and a percentage of neighbours into account as the remote neighbours' strategy [328]. The hierarchical Weighting Vicsek model is proposed for flocking navigation, and it assigns different layers for the drones with weights to enhance the convergence speed, analyzing the involved parameters [329].

Algorithm 11: Vicsek model

```
Initialize parameters
for time ← 1 to time_max do
    Calculate averageTheta
    x ← x + vel * cos(theta) * dt
    y ← y + vel * sin(theta) * dt
    theta ← averageTheta + noise
end
```

$$x_i(t+1) = x_i(t) + v_i(t)\Delta t \tag{5.1}$$

$$\theta_i(t+1) = \langle \theta(t) \rangle_r + \Delta\theta \tag{5.2}$$

Where $x_i(t+1)$ is the location of the particle i, and the velocity v_i is gained through an absolute velocity and the angle $\theta_i(t+1)$. $\langle \theta(t) \rangle_r$ represents the average angle of the neighbouring particles within a circle with radius r. $\Delta\theta$ denotes a random number from $[-\frac{\eta}{2}, \frac{\eta}{2}]$ as noise.

5.4.2 Algorithm Description

Overview

Weight-Leader-Vicsek-Model is proposed for a multi-AGV path planning algorithm to improve the Vicsek model as the Vicsek model cannot achieve practical path planning in the industry. WLVM assigns the virtual leaders to collaborate with the grouped AGVs and guide the followers to reach their destinations, considering collision avoidance.

Figure 5.2 describes the process of WLVM. According to a real industrial environment, the storage map is generated. Points represent the map; 0 for the space allowed to move, while 1 for the wall or the collision-free area. The number of AGVs, velocities, angles and locations are set for model initialization. The AGVs are divided into swarms based on their locations and destinations with assigned virtual leaders. The positions and angles of virtual leaders are computed by the hybrid A* algorithm. The follower-AGVs use the status of the leader in the current group to get the average angle within the defined path for WLVM. The AGVs implement a segment delay function to be separated by a certain distance for optimal arrangements. The AGVs avoid vehicle congestion and deadlock. [235] presents the design of the model for the AGV system.

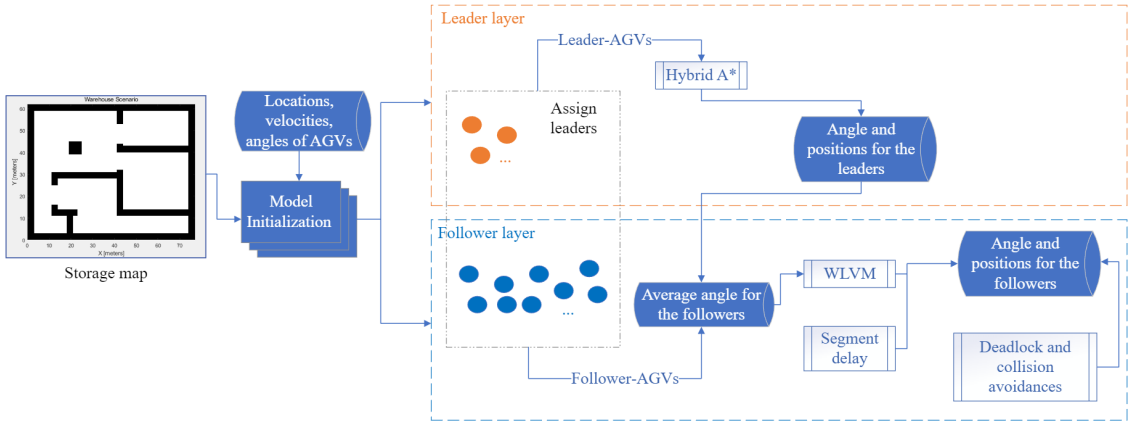


Figure 5.2: WLVM process

Table 5.1 saves the positions and angles of each swarm during the path planning process. N stands for the number of the AGVs, and $Virtual\ Leader$ represents the leader in the group. WLVM saves the positions and angles for follower-AGVs, and

Table 5.1: AGV navigation data in the same group

Iteration	AGV 1	AGV 2	$\cdots$	AGV n	Virtual Leader
1	$\begin{bmatrix} x_1^1 \\ y_1^1 \\ \theta_1^1 \end{bmatrix}$	$\begin{bmatrix} x_2^1 \\ y_2^1 \\ \theta_2^1 \end{bmatrix}$	$\cdots$	$\begin{bmatrix} x_n^1 \\ y_n^1 \\ \theta_n^1 \end{bmatrix}$	$\begin{bmatrix} x_L^1 \\ y_L^1 \\ \theta_L^1 \end{bmatrix}$
2	$\begin{bmatrix} x_1^2 \\ y_1^2 \\ \theta_1^2 \end{bmatrix}$	$\begin{bmatrix} x_2^2 \\ y_2^2 \\ \theta_2^2 \end{bmatrix}$	$\cdots$	$\begin{bmatrix} x_n^2 \\ y_n^2 \\ \theta_n^2 \end{bmatrix}$	$\begin{bmatrix} x_L^2 \\ y_L^2 \\ \theta_L^2 \end{bmatrix}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
k	$\begin{bmatrix} x_1^k \\ y_1^k \\ \theta_1^k \end{bmatrix}$	$\begin{bmatrix} x_2^k \\ y_2^k \\ \theta_2^k \end{bmatrix}$	$\cdots$	$\begin{bmatrix} x_n^k \\ y_n^k \\ \theta_n^k \end{bmatrix}$	$\begin{bmatrix} x_L^k \\ y_L^k \\ \theta_L^k \end{bmatrix}$

virtual leaders in each iteration and is used for further updating.

Dynamic Virtual Leader

Dynamic virtual leaders are implemented in Weight-Leader-Vicsek-Model for shorter paths, faster convergence, and more accurate direction for planning the path to arrive at the destination. Each multi-AGV group has one virtual leader. Figure 5.3 demonstrates the principle of the dynamic virtual leader. When a new AGV joins the current group, the AGVs of the multi-AGV group will treat it as part of the current group, and the group dynamically generates the virtual leader. The virtual leaders are generated in a static environment based on the start and target location. When the AGVs aim for different areas, the start-destination matrix makes the separation, which refers to the Section "Start-destination".

The positions for each AGV are calculated by Equation (5.3). The equations for calculating the angles for follower AGVs are as Equation (5.4).

$$\begin{pmatrix} x_i^{k+1} \\ y_i^{k+1} \end{pmatrix} = \begin{pmatrix} x_i^k \\ y_i^k \end{pmatrix} + v\Delta t \cdot \begin{pmatrix} cos\theta_i^k \\ sin\theta_i^k \end{pmatrix} \tag{5.3}$$

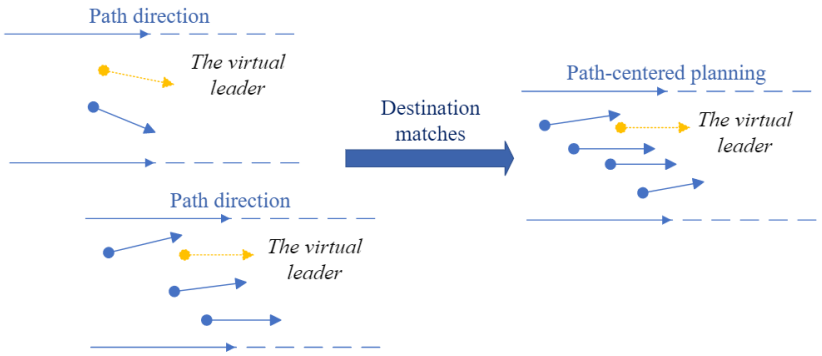


Figure 5.3: Principle of Weight-Leader-Vicsek-Model

$$\theta_i^{k+1} = \omega_1 \cdot arctan \frac{\left\langle sin(\theta_i^k) \right\rangle_p}{\left\langle cos(\theta_i^k) \right\rangle_p} + \omega_2 \cdot \theta_l^{k+1} + \eta_i^k \tag{5.4}$$

Where the average direction $arctan \frac{\left\langle sin(\theta_i^k) \right\rangle_p}{\left\langle cos(\theta_i^k) \right\rangle_p}$ is estimated along the path p, and the travel distance is represented by $v\Delta t$. η_i^k denotes the noise. w_1 is a random number in $(0, 1)$, and the sum of w_1 and w_2 is 1. The swarm only considers the AGVs in the defined path in the same direction. The hybrid A* algorithm is integrated to calculate the position and angles for virtual leaders, and θ_l^{k+1} denotes the angle of the leader in the current iteration. It is adapted for multi-task implementation for virtual leaders.

When an AGV enters a new group, the virtual leader of the group changes if the AGV's destination is the same as the group. Otherwise, the group remains. The leaders are predefined, while the followers keep gaining statuses from the neighbourhoods and then updating their status. It provides the possibility of real-time implementation.

For virtual leaders, angles and positions are generated by the hybrid A* algorithm, and the pseudo-code is indicated in Algorithm 12. The hybrid A* algorithm is proposed in [330], which guarantees kinematic feasibility and continuous nature [331]. The heuristics are the maximum non-holonomic-without-obstacles and obstacle map, ignoring the nonholonomic nature [330]. The MATLAB navigation toolbox has the function for the hybrid A* algorithm.

Algorithm 12: Hybrid A*

```
Initialization of openset, closeset
openset.push(start)
while openset is not empty do
    x_current ← openset.popMinCostNode
    if exists RS path then
        return the path
    end
    for x_next calculated by kinematic equation do
        Collision avoidance
        if x_next not exists in closeset then
            g ← g(x_current) + l(x_current, x_next)
            if x_next not exists in openset or g < g(x_next) then
                g(x_next) ← g
                h(x_next) ← Heuristic(x_next, x_goal)
                Pred(x_next) ← x_current
                if x_next not exists in openset then
                    openset.push(x_next)
                else
                    openset.update(next node)
                end
            end
        end
    end
end
```

The combination of the hybrid A* and WLVM is shown in Figure 5.4, and the proposed WLVM is in Algorithm 13. The leaders are generated for each swarm, and they are unique. For generated path for followers, the path needs to be smooth by the Spline curve for AGVs to operate. The leaders will be regenerated if the environment or group formation changes. The statuses of AGVs are defined as Equation (5.5). The statuses of AGVs are dynamically assigned based on their roles in the current swarm. Once the AGV arrives at the destination, the status is set as

-1.

Algorithm 13: Weight-Leader-Vicsek-Model(WLVM)

Data: x, y, leaderStart, leaderTarget

Initialize parameters

`// n - the number of particles`

$n \leftarrow size(x, 2) + 1$

$dt \leftarrow 1$

$x \leftarrow \begin{bmatrix} x & leaderStart_x \end{bmatrix}$

$y \leftarrow \begin{bmatrix} y & leaderStart_y \end{bmatrix}$

$theta \leftarrow zeros(1, n)$

for $i \leftarrow 1$ *to* n **do**

  $theta(z) \leftarrow leaderStart_{theta}$

end

for *each swarm* **do**

  `// getting the positions and angles for the virtual leader`

  $[LeaderPos, LeaderAngle] \leftarrow HybridAstar$

  for $iter \leftarrow 1$ *to* k **do**

    Calculate averageTheta

    $x \leftarrow x + vel * cos(theta) * dt$

    $y \leftarrow y + vel * sin(theta) * dt$

    $theta \leftarrow w1 * averageTheta + w2 * leaderAngle(k) + noise$

    Smoothen path

    Deadlock and collision avoidance

  end

  Segment delay

end

$$status = \begin{cases} 1, \text{if } leader \\ 0, \text{if } follower \\ -1, \text{if } arrives \end{cases} \tag{5.5}$$

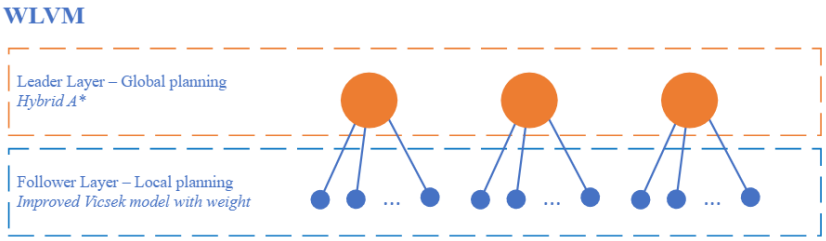


Figure 5.4: Dual layer of Weight-Leader-Vicsek-Model

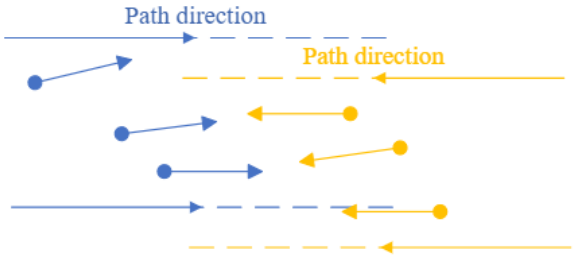


Figure 5.5: Groups with a different direction

Start-destination

Weight-Leader-Vicsek-Model can handle the multi-agent motion directed to different areas, as shown in Figure 5.5. The start-destination matrix is saved in Table 5.2. It assigns the multi-AGV groups, determines each group's destination, treats them as other swarms, and does not integrate them. M Stands for the number of leader-AGVs, which is much less than the number of follower-AGVs. *Start* and *Destination* represent each leader's origin and destination locations.

Keep updating positions and directions within the defined group based on the destination flag. When AGVs are in operation, they only consider the AGVs along the path in the same direction. Even if the other path with a different direction is closer, the AGVs would not be generated. It only concerns the AGVs on the current path.

Table 5.2: Start-destination matrix

	Virtual leaders			
Location	$Leader_1$	$Leader_2$	$\cdots$	$Leader_M$
Start	$Start_1$	$Start_2$	$\cdots$	$Start_M$
Destination	End_1	End_2	$\cdots$	End_M

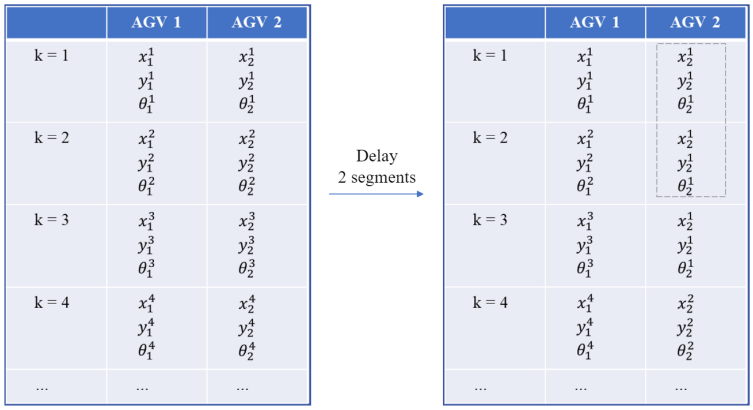


Figure 5.6: The positions after setting segment delay as 2 segments

Segment Delay

The proposed Weight-Leader-Vicsek-Model enables AGVs to travel as a swarm and reach the target, which results in inefficiency for loading and unloading, so segment delay is introduced to solve this problem. Figure 5.6 indicates the operation of segment delay, and it sets segment delay as 2 segments. Each AGV follows the defined path while having a different segment delay for departure in each swarm. The segment delay is set as 5 segments in the computational experiments to provide the buffer area among AGVs.

Collision Avoidance

Obstacle Avoidance Collision avoidance is necessary during the updating iterations to keep AGVs safe. There usually are some obstacles on the map in the practical implementation, so collision avoidance with the obstacles should be achieved. Collision avoidance of the virtual leaders is achieved by the hybrid A* algorithm. The follower-AGV utilizes the leader angle in the next iteration to determine the direction of the movement.

Figure 5.7 demonstrates the movement of the AGV. Each obstacle or wall sets the buffer area as 1m. The leader angle θ indicates the movement of the path, and AGV is represented by i. If a path point represented by $r_i(x_i, y_i)$ overlaps with the buffer area or the restricted area as the obstacles, it requires adjusting positions. The steps for achieving obstacle avoidance are as below.

First, comparing θ with 0. If $\theta \geq 0$, it means the path is aimed to move upper/-

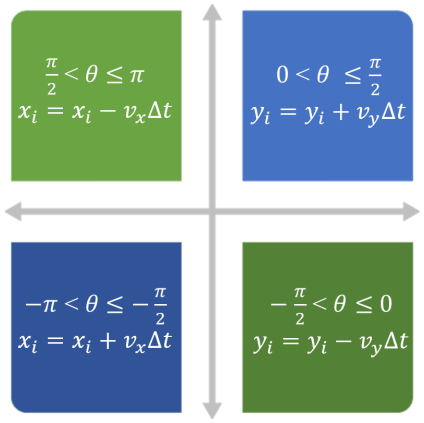


Figure 5.7: The AGV's new position after Collision avoidance

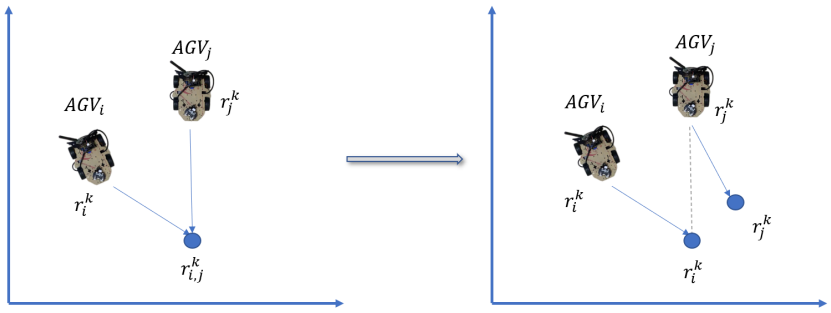


Figure 5.8: The AGV's new position after Deadlock avoidance

forward, then $y_i = y_i + v_y \Delta t$. While θ<0, which means the path is moving lower, then $y_i = y_i - v_y \Delta t$. The AGVs follow the dotted line to change the Y locations. Second, comparing θ with $\frac{\pi}{2}$ or $-\frac{\pi}{2}$. If $\theta \geq \frac{\pi}{2}$, it means the path is aimed to move left, then $x_i = x_i - v_x \Delta t$. While $\theta \geq -\frac{\pi}{2}$, which means the path is moving right, then $x_i = x_i + v_x \Delta t$. The AGVs follow the dotted line to change the X locations.

Deadlock Avoidance The other AGVs in the predictable path or the moving obstacles are treated as dynamic obstacles. Vehicle congestion must be avoided, as shown in Figure 5.8. The target location reserves only one vehicle in each iteration to avoid deadlock. The strategies to deal with the moving obstacles, except for other AGVs, refer to the previous section.

Deadlock avoidance involves priorities for AGVs, and high priority is assigned when AGVs carry goods close to the destination or have urgent tasks. The AGV with higher priority remains following the predicted path, while the other AGV moves to an available position. The priority is calculated based on Equation (5.6), and

the new position is gained by Equation (5.7). The new position is updated in the robot's path after ensuring it is available.

$$priority_i = \omega_1 \cdot priority_{task} + \omega_2 \cdot distance \tag{5.6}$$

$$r_{i-new}^{k+1} = r_i^{k+1} + rand \tag{5.7}$$

Where i stands for the current robot, and *distance* is the distance from the current position to the destination. $priority_{task}$ represents the priority of each assigned task, and if the task is more urgent than others, $priority_{task}$ is higher. w_1 and w_2 are the weight of each factor, the sum of w_1 and w_2 is 1. r_{i-new} stands for a new position of the robot i.

5.4.3 Comparison

The proposed WLVM implements dynamic swarms and virtual leaders to ensure accurate direction and faster convergence, considering collision avoidance, the start-destination matrix to distinguish the destinations of swarms and the applications of multi-objective algorithms in the industrial environment. Figure 5.9 shows the advanced functionality of the Weight-Leader-Vicsek-Model (WLVM). It also provides flexibility because of the dynamic swarms involved, and the follower-AGV can join or disconnect from the current group. The AGVs are assumed to exchange information during operations; if one AGV enters another area, it becomes a member of the new group.

The WLVM is novel for multi-AGV navigation, and it adapts the biological pattern because it achieves the path planning of several AGVs in one step. The traditional Vicsek model can describe the multi-agent movement, which is enhanced to improve WLVM. Following the biological pattern, AGVs move automatically as their neighbours in the same direction if operating the same task. It is typical to involve several AGVs for one task, the improved WLVM achieves fast multi-AGV path planning by updating the positions and angles. It only requires calculation for the virtual leaders in the leader layer and one calculation step in the follower layer. Even though the number of AGVs is large, it obtains the path with a quick calculation.

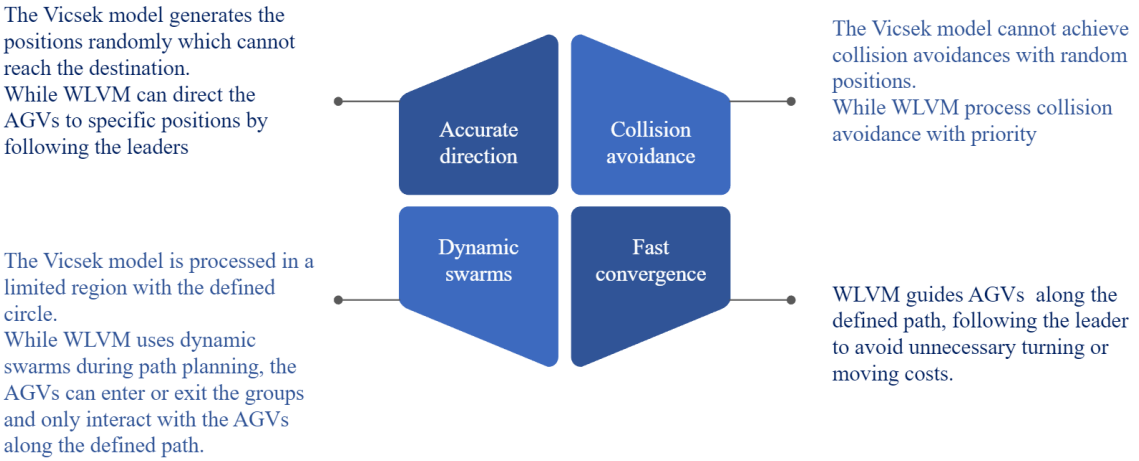


Figure 5.9: Comparison of Vicsek Model and Weight-Leader-Vicsek-Model

5.5 Computational Experiments

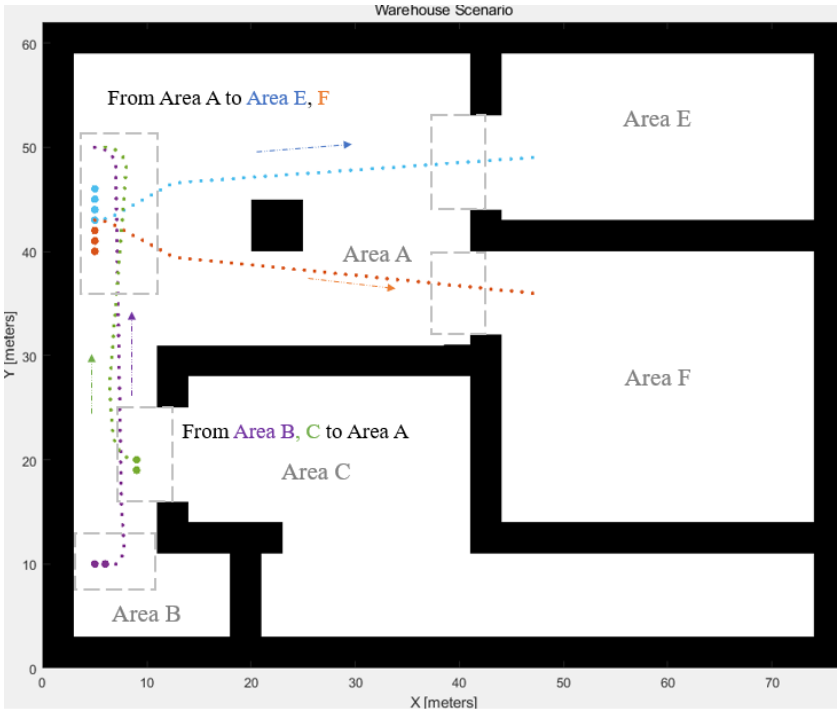
5.5.1 Experiment Settings

Figure 5.10 generates the warehouse map and denotes the initial locations of AGVs; different colours indicate the different swarms. Weight-Leader-Vicsek-Model is validated through MATLAB. The start-destination matrix separates the swarms. Each delivery group is operating in Area A.

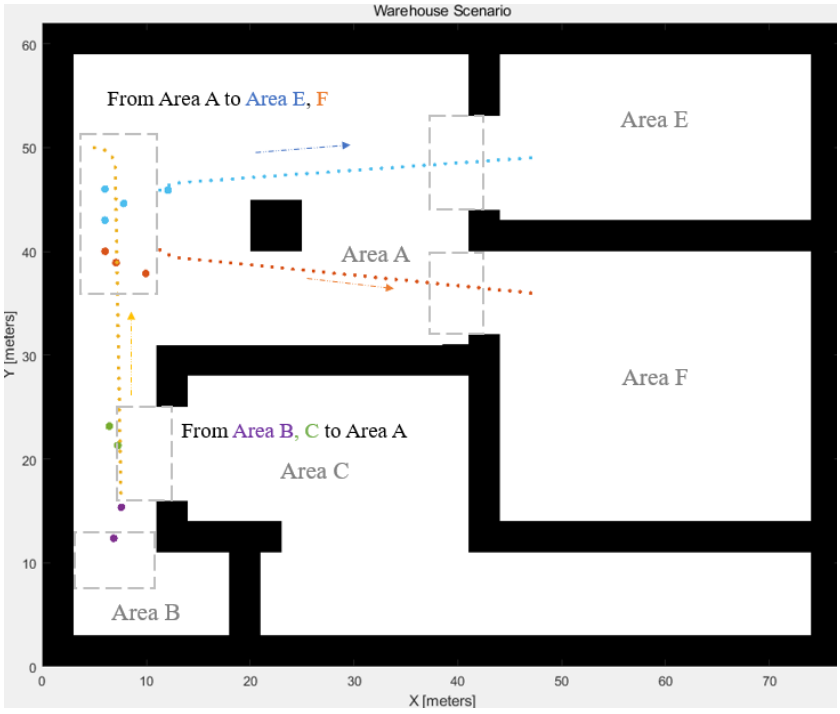
The multi-AGV system is engaged for deliveries from Area A because the three-dimensional storage system is implemented in the primary storage rooms: Area B to Area F. The materials are placed on the platform to transfer to the defined location by the pallet in the three-dimensional storage system.

Assumptions in the simulation are as follows:

- 7 AGVs departure from Area A to different storage areas for processing tasks, and 4 AGVs move to Area A for parking;
- Set segment delay for 5 segments in each swarm;
- AGVs with an absolute velocity of 1m/s;
- Each AGV is equipped with a board that can operate WLVM and under good conditions;
- Each AGV has onboard sensors for localization and obstacle detection;



(a) After t = 0 (1^{st} group change)



(b) After t = 8 (2^{nd} group change)

Figure 5.10: The path for virtual leaders

Table 5.3: Group settings

Group no.	Group	Priority	Number of AGVs	Color
1	Area A to E	1	4	Blue
2	Area A to F	2	3	Orange
3	Area B to A	2	2	Purple
4	Area C to A	3	2	Green

- Each AGV communicates with other AGVs, sending its positions, angles and statuses.

The groups' settings are listed in Table 5.3.

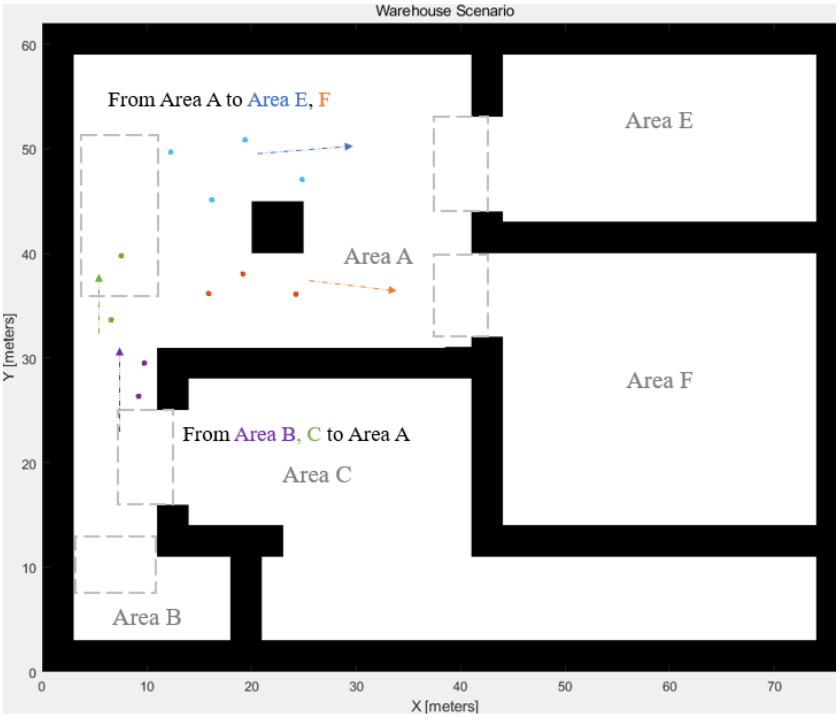
5.5.2 Results

Simulation

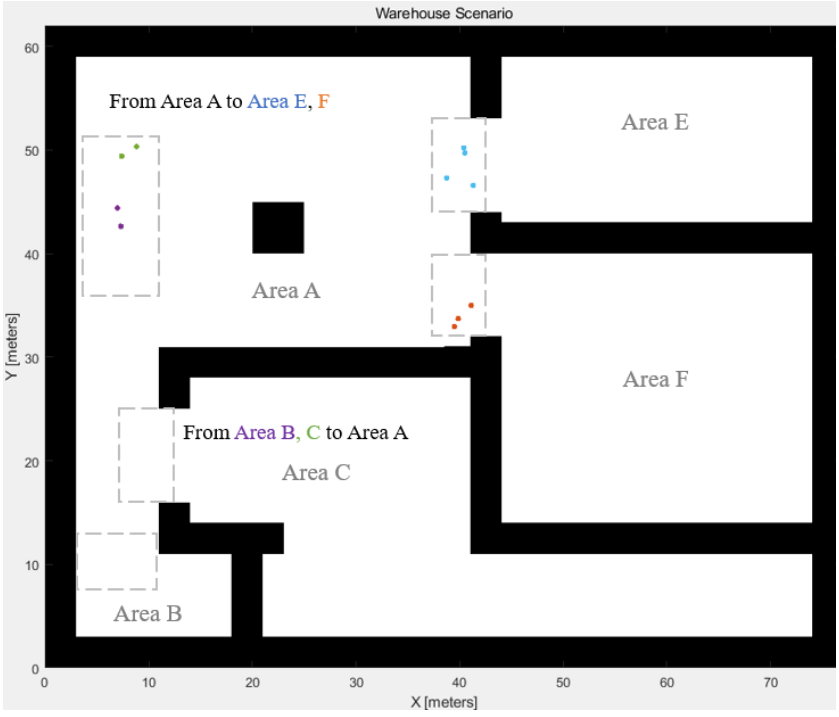
Figure 5.10 displays the path for the virtual leaders generated by the hybrid A* algorithm after the group changes. The directions are indicated along with the path. It also outlines the path area for each swarm. The blue path is from Area A to Area E, and the orange path is from Area A to Area F. The purple path is from Area B to Area A, and the green path is from Area C to Area A. Groups 1 and 2 from Area A are separated into two groups based on the simulation's destination. Groups 3 and 4 are merged into one group when they are close to each other and aim for the same destination.

Figure 5.11 shows the AGVs' path. When two delivery groups pass the same area, collision avoidance ensures that AGVs operate safely. The Start-destination matrix has played a role in distinguishing the multi-AGV group. The groups of inbound delivery are aimed at different main storage rooms with different virtual leaders. When the group arrives at the destination, the group reminds of the current state and the moving AGVs achieve collision avoidance based on the priorities.

The performance measurements are listed in Table 5.4. The distance is the travelling distance for each AGV in the group, and the time is the completion time of each task.



(a) t = 25



(b) t = 49

Figure 5.11: AGV positions during the path planning process and the grey areas indicate the packing areas

Table 5.4: Performance measurements

Group no.	Group	Time	Distance	Group changes
1	Area A to E	49s	34.84, 36.46, 35.84, 34.02	Separated to Group 1
2	Area A to F	44s	36.13, 37.36, 35.97	and 2 when t = 0
3	Area B to A	37s	33.00, 34.21	Group 3 and 4 merged
4	Area C to A	35s	30.82, 30.86	when t = 8

Experiment

The results of the simulation section are validated by the experiment with the Raspberry Pi robot and Ultra-Wide Band (UWB) for positioning. UWB provides centimetre-level positioning and high positioning accuracy in the indoor environment. The robot follows the designed path gained by the simulation.

The experiment used the AGVs from Group 1, the inbound delivery group from Area A to Area E, with 4 AGVs. They followed the defined paths and gained positioning data from the positioning sensor. The positioning results were collected from the Decawave UWB sensors, and the robot carries the target for getting locations. The iterations of the results shown in Figure 5.12 are $t = 0$, $t = 25$, and $t = 49$. The start positions are indicated as $t = 0$, and the destinations are shown when $t = 49$. The locations have some bias due to the sensor accuracy, which can be fixed by sensor fusion in future work. The other collaborative sensors could be Inertial Measurement Unit (IMU), Lidar, or cameras. Then implement the Particle Filter or Extended Kalman Filter for a nonlinear system to eliminate the sensor errors.

5.5.3 Comparison with Another Algorithm

The proposed WLVM is compared with the RRT* algorithm [332] for path planning for four AGVs in Group 1, with a segment delay of 5 segments. The comparison of the runtime is listed in Table 5.5. The RRT* algorithm needs to repeat the calculation for each AGV, while WLVM calculates the path for all AGVs at one step. WLVM provides fast path planning due to the computational speed. Figure 5.13 shows the AGV positions generated by the RRT* algorithm for t = 25 marked

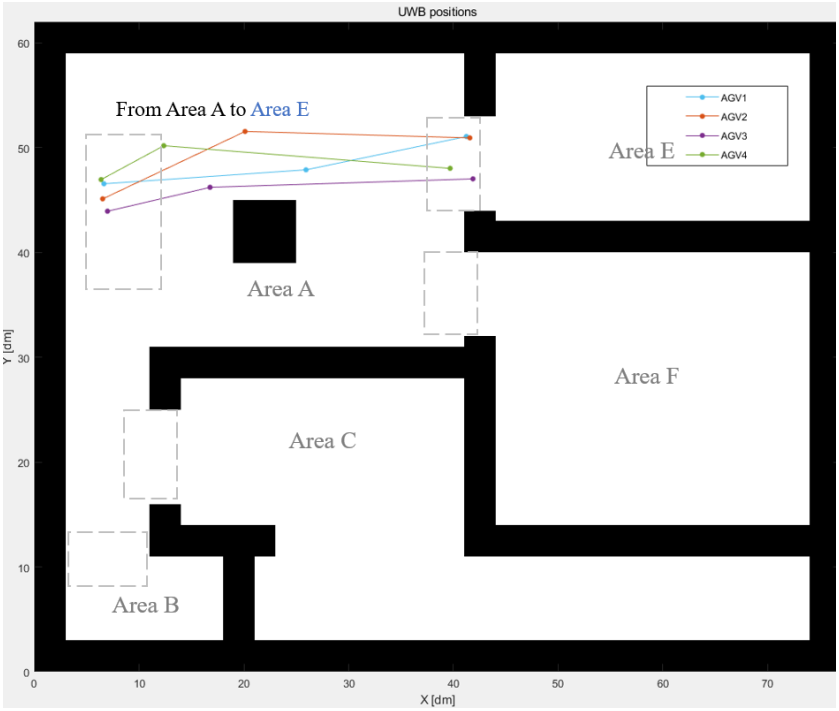


Figure 5.12: UWB Positions following the generated path

by different colours and the last positions when t = 64 marked by blue. If the number of AGVs dramatically rises, the computation time for the RRT* algorithm will rise rapidly, while it would affect WLVM.

Table 5.5: Runtimes for the RRT* and WLVM algorithm

Algorithm	Runtime	Total Runtime
WLVM	Leader: 0.0757s AGV 1-4: 0.0056s	0.0813s
RRT*	AGV 1: 1.5522s AGV 2: 1.1777s AGV 3: 1.1550s AGV 4: 1.1533s	5.0382s

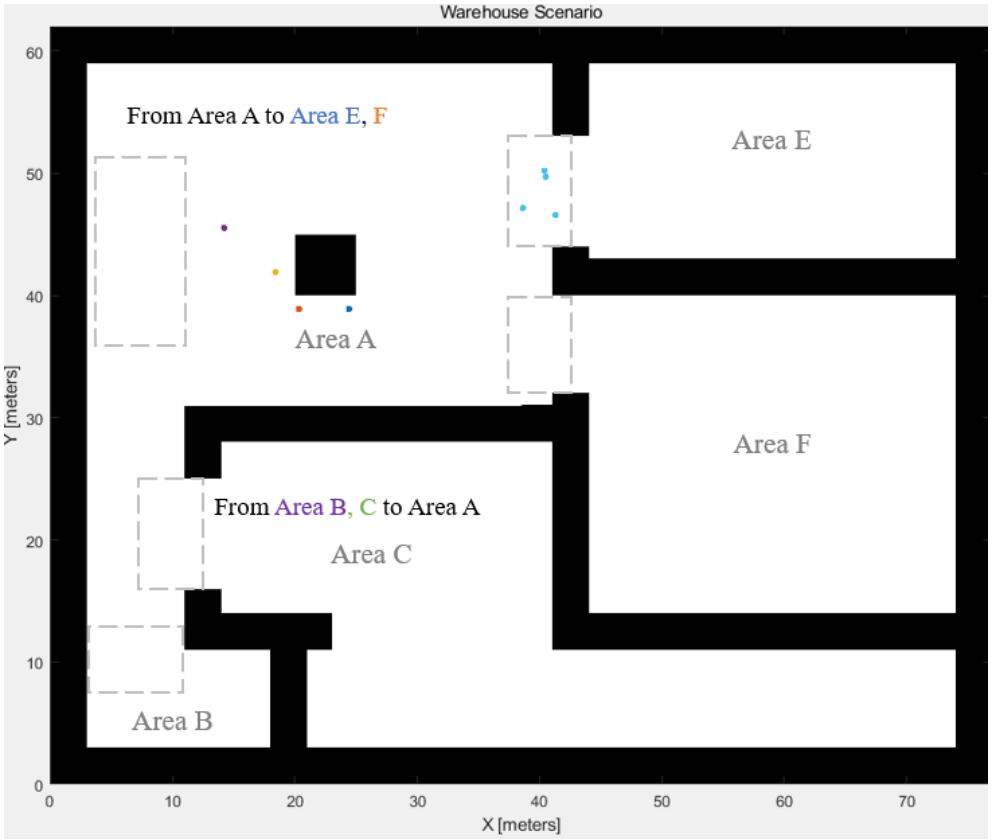


Figure 5.13: AGV positions generated by the RRT* algorithm

5.6 Conclusion and discussion

5.6.1 Discussion

Weight-Leader-Vicsek-Model is proposed to provide scalability and flexibility for multi-AGV systems with fast and flexible path settings. The path planning problem is formulated as a 2D space with the start and target locations and avoids static and dynamic obstacles. From the literature, most path-planning approaches plan the path independently.

However, the proposed algorithm can offer the path settings in one calculation step. Weight-Leader-Vicsek-Model implements the virtual leader to navigate the follower-AGV in each multi-AGV group. The leader's positions and angles are generated by the hybrid A* algorithm. Unlike the traditional Vicsek model, the proposed approach updates the statutes of AGVs with iterations, and the angles of AGVs consider the neighbour and the leader.

For swarm integration or separation, the start-destination matrix plays a role. It

determines whether the multi-AGV group is aimed at the same destination and makes changes in the decentralized follower layer. Segment delay is implemented for optimal arrangement between AGVs for loading. Also, collision avoidance is introduced with obstacle and deadlock avoidance.

The proposed Weight-Leader-Vicsek-Model has the following benefits: accurate direction, dynamic swarms, fast convergence, and collision avoidance. It can achieve real-time implementation due to its computational speed and robust implementation. Four groups with different settings demonstrate swarm separation and integration for the computational experiment. The proposed algorithm is compared with the RRT* algorithm for path planning in the simulation, Weight-Leader-Vicsek-Model saves 98.39% computational time.

The limitation of the proposed approach is that it does not consider the cost value during the path planning as the heuristic methodologies. Therefore, the generated path cannot be measured or estimated with the specific costs to determine whether the path is globally optimal. Also, the avoidance of dynamic obstacles could be improved with sensor fusion algorithms in future studies.

5.6.2 Conclusion

The traditional Vicsek model is unsuitable for path planning because of its random direction and ignoring obstacle avoidance with the environment. Weight-Leader-Vicsek-Model is proposed to improve the Vicsek model, and it develops dynamic virtual leaders and a start-destination matrix, considering the leaders' direction with weight. It makes the model possible to reach the destination based on the map and considers obstacle avoidance. It has much less computational load and more flexibility than graph search and sampling-based algorithms with faster convergence. The relevant sensors and programmable robots can directly apply the results.

Model and system initialization and multi-AGV group formation are completed through a centralized method, while Weight-Leader-Vicsek-Model achieves the dynamic decentralized approach for each AGV. One virtual leader leads each multi-AGV group directly or closes to the destination. Path planning of leaders is achieved by the hybrid A* algorithm, and for the followers is achieved by the improved updat-

ing equation. It computes the follower-AGVs' path with a quick computation, even though the number of AGVs is large. Unnecessary turning costs and path segments are avoided in this model. Each AGV only considers its neighbours in the path, and it is tended to move as its neighbours. When an AGV enters a new group, if the destination matches, it will follow a newly generated virtual leader like the other group particles.

The proposed Weight-Leader-Vicsek-Model is robust and simple for implementation during the AGVs' or robots' operation. The proposed algorithm can be applied to various scenarios involving the system of multiple robots, such as warehouses, logistics systems, ports, and airports. For future work, Weight-Leader-Vicsek-Model would be more practical in the industrial environment with the following considerations. The convergence of the particles should be considered in the application. Mission planning and task allocation can be included in the further improvement of this model. The multi-AGV system would be more practical if it involves fault tolerance during implementation. The combination of sensors and the sensor-fusion algorithm could be considered in the further real experiment to estimate the angle and the position.

Chapter 6

An Fault-tolerant Cultural-PSO for Multi-AGV Path Planning

6.1 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 [312]. 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. [318]. Multi-AGV systems have been more common in warehouses, compared with the single-AGV counterparts, achieving efficiency and robustness of operations [325].

For the multi-AGV system, path planning is the primary consideration for robot navigation. 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, as mentioned in Chapter 2. This chapter aims to provide solutions for multi-AGV path planning, considering task allocation and fault tolerance for better practice. 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 metaheuristic algorithm to enhance search abilities and performance. It is organized as follows. Section 6.2 describes the problem formulation with cost functions. Section 6.3 reviews the related work. Section 6.4 introduces the cultural algorithm, particle swarm optimization algorithm and metropolis rule. Section 6.5 proposes C-PSO with a dual-layer framework, and the experiment results are in Section 6.6. It is concluded in Section 6.7.

6.2 Problem Formulation

The path planning problem of the multi-AGV system consists of multiple starts and destinations for the involved AGVs and the two-dimensional environment. It is required that path planning cannot occur collisions or deadlocks. A binary map indicates the two-dimensional environment: zero is allowed to move, while one stands for the wall or obstacle marked by black.

Figure 6.1 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 in the same timeslot during operation. The cost functions of path planning consider the path length and collision as Equations (6.1) – (6.4). The best solution is treated as the globally optimal path with minimal costs. Each position r_i^t is represented by (x_i^t, y_i^t). The particle is indicated by i, and the iteration is indicated by t. The total number of particles is n.

$$f^t_{length} = \sum_{i=1}^{n} \sqrt{(x^t_i - x^t_{i+1})^2 + (y^t_i - y^t_{i+1})^2} \tag{6.1}$$

Where the current particle is i, and the next particle is $i+1$.

The violation cost of collision is evaluated by Equations (6.3) and (6.4).

$$d^t_i = \sqrt{(x^t_i - x_c)^2 + (y^t_i - y_c)^2} \tag{6.2}$$

$$c^t_i = \sum_{c=1}^{z} r_c - d^t_i, \quad if \quad d^t_i < r_c \tag{6.3}$$

$$f^t_{collision} = \sum_{i=1}^{n} c^t_i \tag{6.4}$$

Where (x_c, y_c) is the central coordination of an obstacle, the total number of obstacles is z, and c^t_i is the collision cost for the particle i, which sums up the total collision cost of the current particle. Then summing all collision costs of all particles as the collision cost.

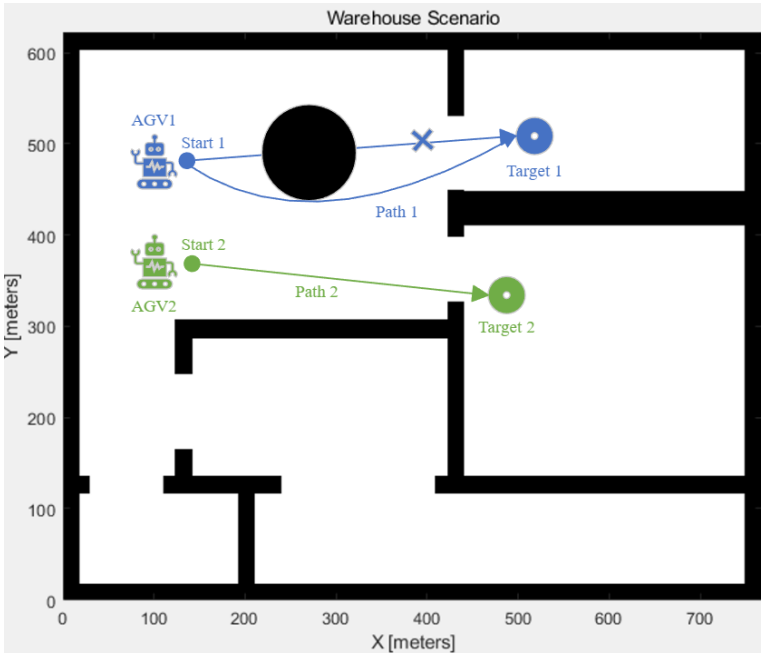


Figure 6.1: The scenario of problem formulation

6.3 Related Work

The deadlock resolution methods consist of zone control, time windows, and Petri-net according to traffic control strategies [318]. The time window is one of the most popular deadlock resolution methods, which refers interval of time when an operation occurs. For the multi-AGV system in intelligent warehousing, the time window is used for task assignment, and the A* algorithm finds the path in [333]. A dynamic routing method is applied for multi-AGVs for supervisory control, and it evaluates paths by appropriate time windows and window overlapping tests [334]. A conflict graph is introduced to solve the task assignment problem to deal with deadlocks [335].

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 [315]. The hybrid algorithm of Dijkstra and Floyd-warshall is presented with time windows in [317]. 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 [322]. Grid blocking degree is proposed for multi-AGV path planning to minimize the handling completion time with a conflict-free path [124].

Moreover, 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 [336]. Probabilistic time constraints and queuing mechanisms are combined with an A* algorithm based on a dynamic random network in a warehouse multi-AGV system [310]. 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 [337]. The time-enhanced A* algorithm provides path planning in real-time with temporal estimation and a supervision system [338].

Three-exchange crossover heuristic operators are implemented in an improved genetic algorithm with double-path constraints to minimize the distance for multi-AGV path planning [339]. Ant colony algorithm is enhanced with prioritized planning for path coordination and optimization, considering the remaining battery charge and the fitness value [320]. The genetic algorithm is improved for multi-

AGV scheduling planning based on the operation of power evaluation and change of mutation operators, considering power consumption [340]. Spinning drawing frames improve the genetic algorithm for multi-AGV path planning and maneuvering scheduling decisions [341]. A particle swarm optimization algorithm is used for AGV scheduling with a path time window [342].

Two-staged scheduling is introduced to handle collision for multi-AGV systems. In [149], the offline scheduling stage uses the genetic algorithm 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 is based on a genetic algorithm [343], and the genetic algorithm is processed with constraints to get a stable path in the online stage. [344] 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 [318]. [314] 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 [13] for supporting local path planning and a flexible and cooperative route assignment with knowledge extension. The particle swarm optimization algorithm is improved to reduce congestion and provide efficiency for multi-AGV path optimization [321].

Deep reinforcement learning can process high-dimensional environment data, such as images, and it has intelligent decision-making ability and powerful perception ability [37], [345]. A neural network structure and Dueling DDQN-PER has been implemented as AGV path planning for multi-modal sensing environments information, such as GPS, cameras, and speed sensors [37]. For large-scale space, a reinforcement learning algorithm combines with a deep q-network in a complex environment [345]. Multi-agent reinforcement learning is proposed as a deep deterministic policy gradient for anti-conflict multi-AGV path planning in [346], considering conflict

situations as an integer programming model.

The literature shows that graph-based search approaches, bio-inspired, and AI-based 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 scheduling and path planning system. However, from [347], fault tolerance is barely considered during multi-AGV real-time path planning. This chapter proposes a fault-tolerant multi-AGV path planning algorithm based on the hybridization of cultural algorithm and particle swarm optimization.

6.4 Preliminary Knowledge

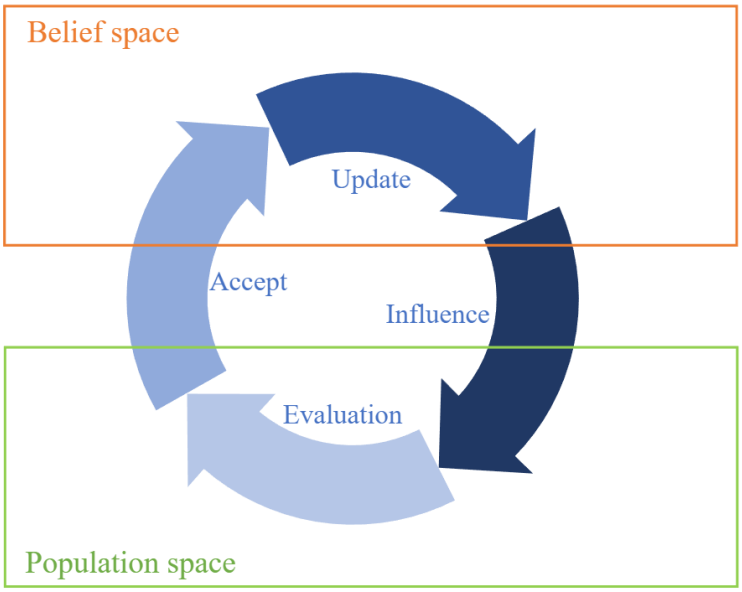
6.4.1 Cultural Algorithm (CA)

A cultural algorithm (CA) is proposed in [62] as an evolutionary algorithm by gaining solutions through normative knowledge and situational knowledge. The principle of CA is illustrated in Figure 6.2 and Algorithm 14. It consists of belief space and population space and uses evolutionary knowledge to determine the solutions.

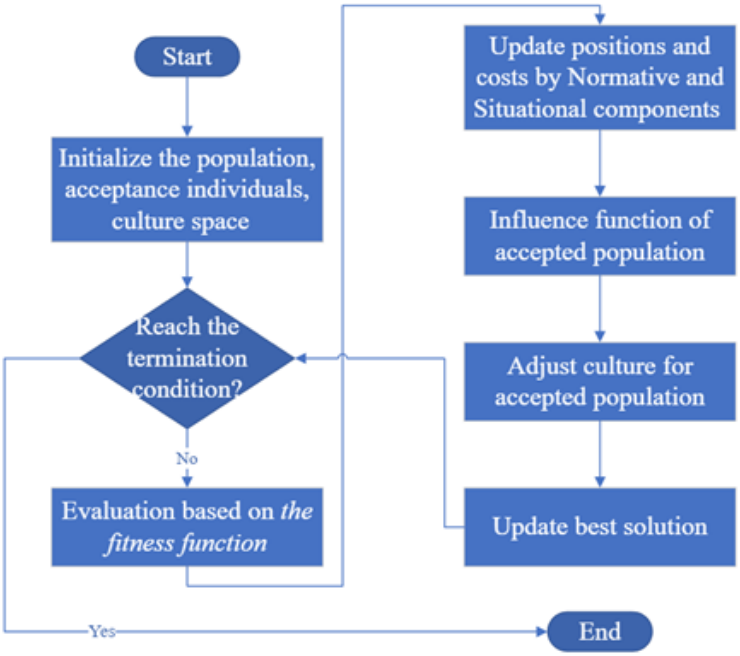
Algorithm 14: Cultural algorithm

```
Initialization of population and belief space
for iteration = 1 : iteration_max do
    for i = 1 : N do
        Influence()
        Evaluation
        Sort Population
        Update() by Selected Population
    end
end
```

CA uses the influence function to get the new population as Equation (6.5). The accept function usually selects n individuals with better solutions from the population. The update function implements Equations (6.6) – (6.9) to update the positions and fitness values.



(a) The principle of CA



(b) The flowchart of CA

Figure 6.2: The principle of CA

$$x_{i,j}^{t+1} = \begin{cases} x_{i,j}^{t} + |(u_j^t - l_j^t) \cdot rand|, \text{if } x_{i,j}^t < l_j^t \\ x_{i,j}^{t} - |(u_j^t - l_j^t) \cdot rand|, \text{if } x_{i,j}^t > l_j^t \end{cases} \tag{6.5}$$

Where $x_{i,j}^t$ is the position, t is the iteration number, i indicates the individual, and j means the jth value of i. u_j^t is the maximum position limit, 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 (6.6) – (6.9) to update the positions and fitness values.

$$l_j^{t+1} = \begin{cases} x_{i,j}^t, \text{if } x_{i,j}^t \leq l_j^t \text{ or } f_{cost}(x_i^t) < L_j^t \\ l_j^t, \text{otherwise} \end{cases} \tag{6.6}$$

$$L_j^{t+1} = \begin{cases} f_{cost}(x_i^t), \text{if } x_{i,j}^t \leq l_j^t \text{ or } f_{cost}(x_i^t) < L_j^t \\ L_j^t, \text{otherwise} \end{cases} \tag{6.7}$$

Where f_{cost} is the objective function to evaluate the fitness value. l_j^{t+1} denotes lower bound for the jth value of individual i at iteration $t+1$. L_j^{t+1} presents the fitness value of the lower bound. The lower bound updates when the fitness value is lower than L_j^t or $x_{i,j}^t$ is less than l_j^t.

$$u_j^{t+1} = \begin{cases} x_{i',j}^t, \text{if } x_{i',j}^t \geq u_j^t \text{ or } f_{cost}(x_{i'}^t) < L_j^t \\ u_j^t, \text{otherwise} \end{cases} \tag{6.8}$$

$$U_j^{t+1} = \begin{cases} f_{cost}(x_{i'}^t), \text{if } x_{i,j}^t \geq u_j^t \text{ or } f_{cost}(x_{i'}^t) < L_j^t \\ L_j^t, \text{otherwise} \end{cases} \tag{6.9}$$

Where individual i' impacts the upper bound. u_j^{t+1} is upper bound for the jth value of individual i' at iteration $t+1$. U_j^{t+1} is the maximum fitness value in the $t+1$ iteration for jth value.

6.4.2 Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) [277] is widely implemented in optimization, especially for robot path planning. The particles explore the searching space to get

the solutions for the optimization problems. Algorithm 15 demonstrates the PSO algorithm. The equations of velocity and position of the particles are updated as Equations (6.10) - (6.11). The personal best of a particle and the global best is updated as Equations (6.12) - (6.13).

$$v_i^{t+1} = \omega v_i^t + c_1 r_1 (pbest_i^t - x_i^t) + c_2 r_2 (gbest^t - x_i^t) \tag{6.10}$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{6.11}$$

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 $t+1$.

$$pbest_i^{t+1} = \begin{cases} pbest_i^t, \text{if } f_{cost}(x_i^{t+1}) \geq f_{cost}(pbest_i^t) \\ x_i^{t+1}, \text{otherwise} \end{cases} \tag{6.12}$$

$$f_{cost}(gbest^t) = \min(f_{cost}(x_1^t)), f_{cost}(x_2^t)), ..., f_{cost}(x_i^t))) \tag{6.13}$$

Where $pbest_i$ is the personal best of particle i, when dealing with minimal optimization, if the fitness value of the current particle is less than the personal best, the personal best gets the current particle. $gbest$ indicates the global best, and it gets the minimal fitness value of the particle as the best solution.

6.4.3 Metropolis Rule

The Metropolis rule is proposed in Simulated Annealing (SA) [348]. The probabilistic of the Metropolis rule is as (6.14). SA reduces "temperature" to get a lower energy state for reaching the solutions to the optimization problem.

$$\rho = \begin{cases} 1, \text{if } f_{cost}(x_i^{t+1}) \leq f_{cost}(x_i^t) \\ e^{\frac{f_{cost}(x_i^{t+1}) - f_{cost}(x_i^t)}{T}}, \text{otherwise} \end{cases} \tag{6.14}$$

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

Algorithm 15: 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
        end
    end
end
```

6.5 Cultural-PSO (C-PSO) with Metropolis Rule

6.5.1 Overview

The proposed C-PSO consists of centralized and decentralized layers, as shown in Figure 6.3. 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.

6.5.2 C-PSO

Inspired by CA and the proposed PSO-SA in Chapter 4, Cultural-PSO (C-PSO) is proposed. It combines the characteristics of CA to update the inertia weight

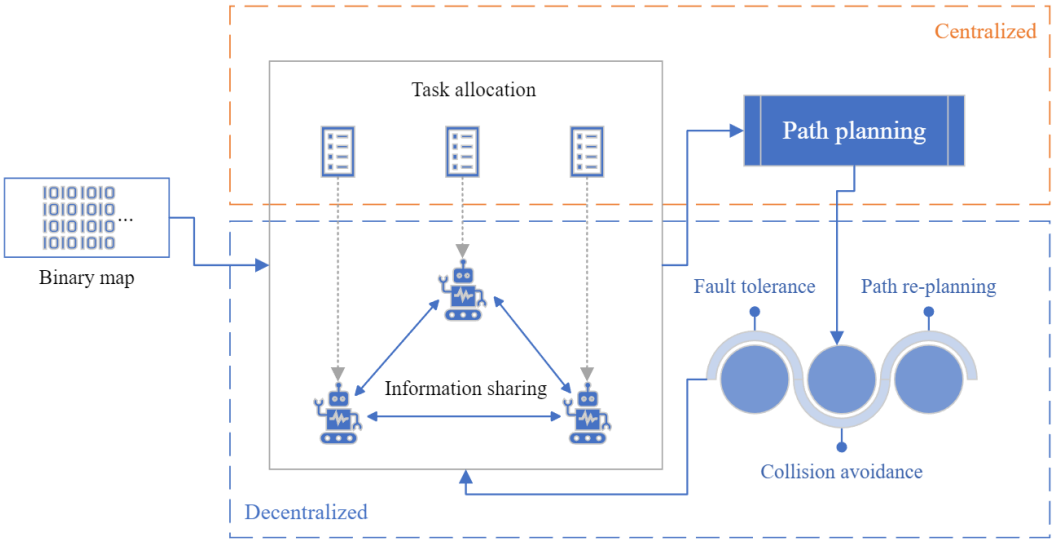


Figure 6.3: Dual-layer of the C-PSO algorithm

for PSO. 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 16. Compared to DWLVM in Chapter 5, C-PSO guarantees path optimality, while DWLVM provides a one-step calculation for all robots.

The probability is calculated by Equations (6.15) - (6.16) 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 Equation (6.17). 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.

$$\delta = \frac{f_{cost}(i_t) - f_{best}(i_t)}{f_{best}(i_t)} \tag{6.15}$$

$$\rho = e^{-\frac{\delta}{T}} \tag{6.16}$$

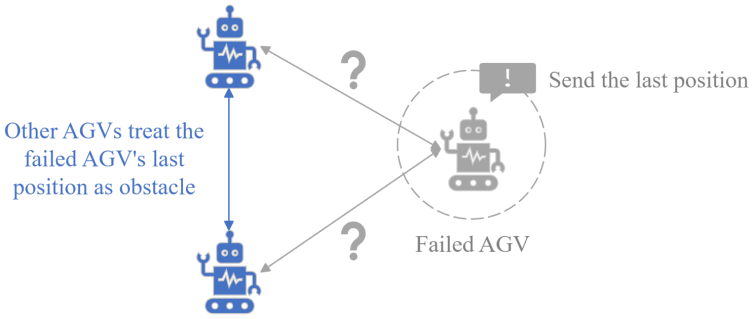
Where ρ is the calculated probability, and δ is calculated by Equation (6.15). $f_{cost}(i_t)$ is the cost of the particle i in the current iteration t, and $f_{best}(i_t)$ is the personal best value. T is the temperature.

Algorithm 16: C-PSO algorithm

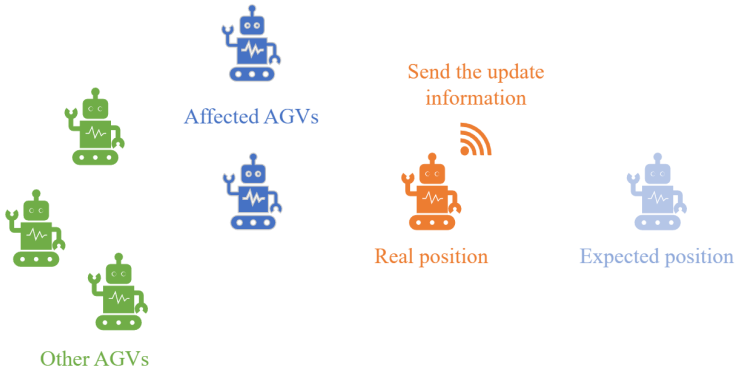
```
Initialization
for iteration = 1 : iteration_max do
    for particle = 1 : particle_max do
        Update Velocity
        Update Position
        Evaluation
        if pcost_i^t < pbest_i^t then
            Update Personal Best
            if pbest_i^t < gbest^t then
                Update Global 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
```

$$\omega = \begin{cases} \omega + |(f_{cost}(i_t) - f_{bestCost}(i_t))|, \text{if } f_{cost}(i_t) < f_{cost}(avg_t) \\ \omega_{min}, \text{if } f_{cost}(i_t) \geq f_{cost}(avg_t) \end{cases} \tag{6.17}$$

Where ω is the inertia weight for PSO. ω is the current value for the inertia weight,



(a) When one AGV fails



(b) When one AGV is not operated as expected

Figure 6.4: The scenario of fault tolerance

and ω_{min} is the minimum value. $f_{cost}(avg_t)$ 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.

6.5.3 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 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 6.4 and follows.

1. When one AGV fails

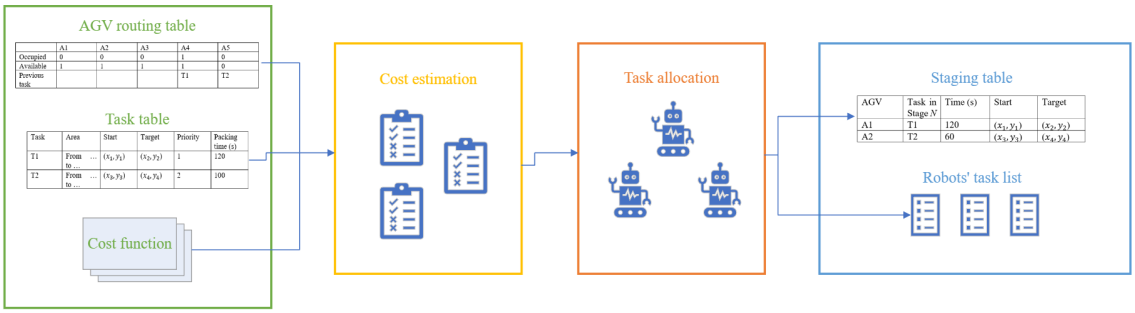


Figure 6.5: The flow of task allocation

- 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.

6.5.4 Task Allocation

Task allocation is essential for the multiple robot system, as demonstrated in Figure 6.5. It considers the task priority, path costs and the time of completing the previous task as Equations (6.18) – (6.19).

$$f_{pathCost} = w_1 * f_{length} + w_2 * f_{collision} \tag{6.18}$$

Where $f_{pathCost}$ is the path's cost function, considering the path's length and collision. The definitions of f_{length} and $f_{collision}$ refer to Section 6.2. 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 path length and collision avoidance is the same important factors for path planning.

$$f_{taskCost} = w_3 * f_{pathCost} + w_4 * f_{taskPriority} + w_5 * f_{timeOfPreviousTask} \quad (6.19)$$

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 6.1. The status of occupied and available is indicated by 0 or 1. For the occupied status, "0" 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 6.1: 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 6.2 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, and the cost is refreshed after a robot or a task is assigned

Table 6.2: 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	$PackingTime_n$

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 6.3 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.

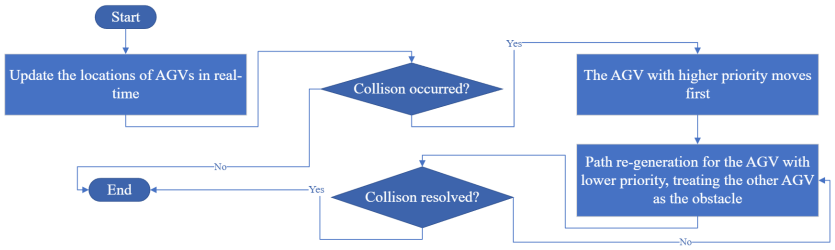
Table 6.3: A example of Staging table

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})

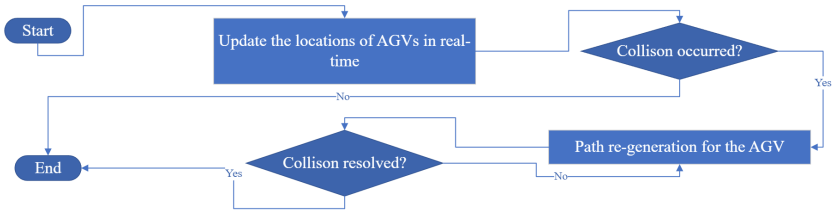
6.5.5 Collision Avoidance

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 6.6.

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



(a) Collision resolution for AGVs



(b) Collision resolution for dynamic obstacles

Figure 6.6: Flowcharts of collision resolution

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.

6.6 Computation Experiments

6.6.1 Performance Measurements

The proposed C-PSO has been compared with other bio-inspired algorithms, including PSO, CA, Harmony Search (HS) [302], and Artificial Bee Colony (ABC) [303]. The test functions include Rosenbrock, Sphere, and Michalewicz, as listed in Table 6.4. The maximum iteration is set as 200, and each algorithm runs 20 times with a population size of 50. For the HS algorithm, the harmony memory size is set as 50. Table 6.5 analyses each benchmark function's iteration times, runtime, and fitness

values with different algorithms. The best value is highlighted in bold. The iteration times are recorded when it equals the best solution. The runtime is calculated by the mean runtime of each main loop when getting the best solutions.

Table 6.4: Test functions

Name	**Type**	**Test function**
Ackley	Many Local Minima	$f_1(x) = -a\ exp\left(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - exp\left(\sqrt{\frac{1}{d}\sum_{i=1}^{d} cos(cx_i)}\right) + a + exp(1)$
Levy	Many Local Minima	$f_2(x) = sin^2(\pi w_1) + \sum_{i=1}^{d-1}(w_i - 1)^2[1 + 10sin^2(\pi\omega_i + 1)]$ $+(w_d - 1)^2[1 + sin^2(2\pi\omega_d)], where\ w_i = 1 + \frac{x_i - 1}{4}, for\ all\ i = 1, ..., d$
Sum squares	Bowl-Shaped	$f_3(x) = \sum_{i=1}^{d} ix_i^2$
Sphere	Bowl-Shaped	$f_4(x) = \sum_{i=1}^{d} x_i^2$
Zakharov	Plate-Shaped	$f_5(x) = \sum_{i=1}^{d} x_i^2 + (\sum_{i=1}^{d} 0.5ix_i^2)^2 + (\sum_{i=1}^{d} 0.5ix_i^2)^4$

C-PSO has significantly fewer iterations when generating the best solution with the best fitness value for each benchmark function than other metaheuristic algorithms, proving the proposed C-PSO has great optimization performance. Also, when C-PSO gets the best solution, the average consumed runtime is the least. In most optimization functions, C-PSO is a steady algorithm to perform the searching ability. Therefore, it is suitable for the optimization problem. Figure 6.7 presents each test function's fitness value. From Figure 6.7 and Table 6.5, it can draw that the proposed C-PSO has faster convergence than PSO. It has much fewer iteration times to get the optimization solutions, and it is useful for online path planning due to its computational speed.

6.6.2 Experiment of Path Planning

Experiment Settings

The AGVs are initially located in the loading area. The pillars are annotated with grey, and the AGVs are marked with different colours. Table 6.6 lists the tasks for the multi-robot system. The locations of the AGVs are shown in Table 6.7.

The assumptions of the simulation are listed below:

- AGVs have the map, and the speed is 0.25 m/s;

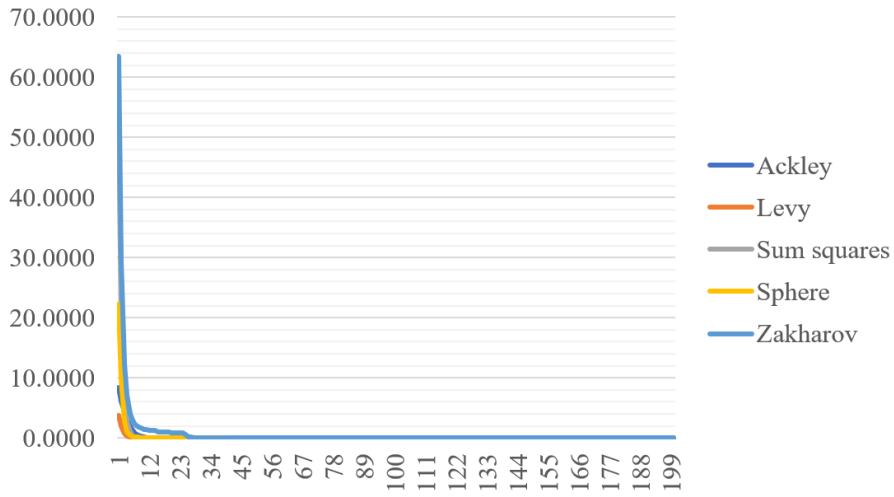


Figure 6.7: Convergence curve of C-PSO

- 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.

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. Task 1 is the most urgent, so it would be assigned the first robot during the planning. Table 6.8 compares the task cost of robots for each task. Task 1 was assigned to AGV 5 based on the combination of task priority and cost considerations, adding to the staging table. Then the system searched for the second lowest task priority, and Tasks 2 and 4 were recorded. Task 2 was assigned to AGV 4 according to the costs. From the costs, AGV 1 has the minimal cost of Task 3, but AGV 1 is broken, so AGV 2, with the second minimal cost, is chosen for Task 3.

For Task 4, the minimal cost is with AGV 5, but AGV 5 is assigned to Task 1 in

the first stage of operation. The second minimal cost is AGV 4. Even though the priority of Tasks 2 and 4 is the same, but the cost of AGV 4 to operate Task 2 is lower than Task 4, so AGV 4 is assigned to Task 2 in the first stage. Therefore, the path cost of AGVs 4 and 5 are updated based on the new location and the time of completing the previous task. AGV 3 is available, and it has two options: one is to operate the failed AGV 2's task, and the other one is to perform Task 4. AGV 3 requires 32s to perform Task 4, but it only needs 9s to reach AGV 2's location to perform Task 3. Therefore, AGV 3 is assigned to Task 3 to resume AGV 2's task list, and Task 4 is assigned to AGV 5 in the second stage from the updated cost. Table 6.9 lists the expected completion time for each task. The stages of the tasks are as follows.

1. Stage 1:
    - Task 1: AGV 5 (Yellow path)
    - Task 2: AGV 4 (Green path)
    - Task 3: AGV 2 (Orange path) $\rightarrow$ AGV 3 (Purple path)
2. Stage 2:
    - Task 4: AGV 5 (Yellow path)

AGV 2 shut down when $t = 5$. The system checks whether there is an available or unoccupied AGV. As AGV 1 is broken and AGVs 4 and 5 are occupied, AGV 2's task table and the last location is transferred to AGV 3. AGV 3 reaches AGV 2 last location and performs the remaining tasks. Figure 6.8 shows the entire AGVs' path and AGVs' locations highlighted by dots when $t = 0, t = 5, t = 10, t = 20$ and $t = 150$. The initial positions of AGVs are denoted when $t = 0$. The stage number is annotated below the time slots. In stage 1, AGV 5 performs Task 1, AGV 4 performs Task 2, and AGV 2 performs Task 3. But AGV 2 shuts down, then AGV 3 promotes fault-tolerance to perform the reminding task of AGV 2. In stage 2, AGV 5 performs Task 4. When AGV 2 shuts down, AGV 3 reaches AGV 2's last location and then moves to the destination of Task 3, and the orange path is the path of AGV2, and the purple path is the path of AGV 3.

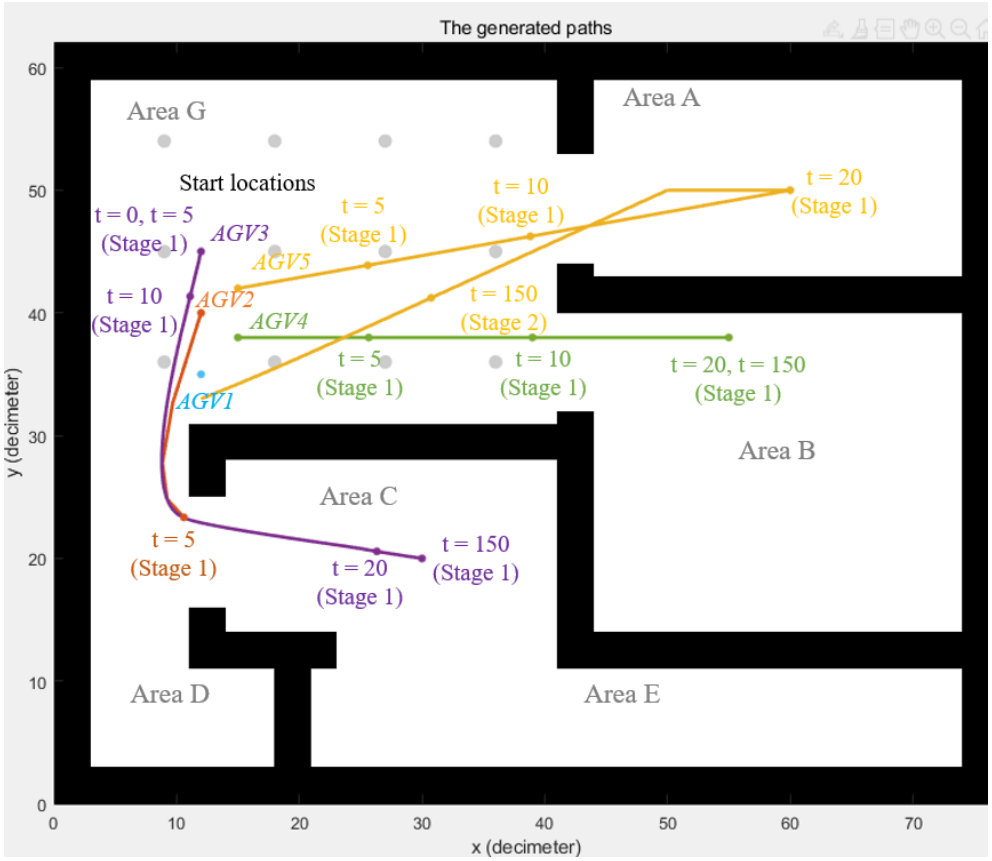


Figure 6.8: AGVs' locations of different timeslot. The paths of AGV 1, AGV 2, AGV 3, AGV 4 and AGV 5 are marked by blue, orange, purple, green and yellow, respectively.

6.7 Conclusion

This chapter 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 fewer $74\%, 47\%, 80\%, 53\%$ iterations than PSO, CA, HS, and ABC, respectively. C-PSO has fewer $69\%, 59\%, 34\%, 57\%$ consumed runtime than the mentioned algorithms with the best solution for different benchmark functions.

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.

Table 6.5: Iterations, runtime, and fitness value

Function			C-PSO	PSO	CA	HS	ABC
Ackley	Iterations	Average	**25.5500**	94.9000	49.6500	193.2500	117.5000
		Std. dev	**1.9358**	2.9182	3.3604	1.9358	7.2684
	Runtime (s)	Average	**0.0341**	0.1067	0.0876	0.1080	0.2550
		Std. dev	**0.0039**	0.0138	0.0075	0.0156	0.1056
	Fitness value	Average	**0.0000**	**0.0000**	**0.0000**	0.0162	**0.0000**
		Std. dev	**0.0000**	**0.0000**	**0.0000**	0.0058	**0.0000**
Levy	Iterations	Average	**13.5000**	60.3000	27.4500	165.1500	50.6500
		Std. dev	**1.3572**	3.7290	2.4165	23.5803	2.6213
	Runtime (s)	Average	**0.0214**	0.0816	0.0535	0.1026	0.1251
		Std. dev	**0.0026**	0.0081	0.0046	0.0258	0.0076
	Fitness value	Average	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
		Std. dev	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
Sum squares	Iterations	Average	**15.4000**	67.0500	28.7000	179.2000	50.3500
		Std. dev	2.0876	2.4597	**2.0287**	13.9608	3.1834
	Runtime (s)	Average	**0.0228**	0.0739	0.0512	0.0883	0.0960
		Std. dev	**0.0044**	0.0076	0.0049	0.0073	0.0244
	Fitness value	Average	**0.0000**	**0.0000**	**0.0000**	0.0002	**0.0000**
		Std. dev	**0.0000**	**0.0000**	**0.0000**	0.0001	**0.0000**
Sphere	Iterations	Average	**15.2000**	63.6000	26.0500	170.2000	47.1000
		Std. dev	3.1556	3.0848	**1.8489**	21.8743	2.8266
	Runtime (s)	Average	**0.0203**	0.0717	0.0466	0.0862	0.0879
		Std. dev	0.0050	0.0076	**0.0034**	0.0170	0.0078
	Fitness value	Average	**0.0000**	**0.0000**	**0.0000**	0.0001	**0.0000**
		Std. dev	**0.0000**	**0.0000**	**0.0000**	0.0001	**0.0000**
Zakharov	Iterations	Average	**24.0000**	71.0000	47.6000	193.7500	170.3000
		Std. dev	6.1044	**4.1422**	4.7143	5.6835	16.2095
	Runtime (s)	Average	**0.0318**	0.0798	0.0862	0.1004	0.3100
		Std. dev	0.0076	**0.0056**	0.0084	0.0118	0.0408
	Fitness value	Average	**0.0000**	**0.0000**	**0.0000**	0.2549	0.0002
		Std. dev	**0.0000**	**0.0000**	**0.0000**	0.3936	0.0002

Table 6.6: Task list in the simulation

Task	Area	Start	Target	Priority	Packing time (s)
T1	From G to A	Loading zone	(60,50)	1	120
T2	From G to B	Loading zone	(55,38)	2	100
T3	From G to C	Loading zone	(30,20)	3	120
T4	From A to G	(50,50)	(12,33)	2	110

Table 6.7: AGVs' initial location

AGV	Area	Position	Color
AGV1	Loading zone	(12,35)	Blue
AGV2	Loading zone	(12,40)	Orange
AGV3	Loading zone	(12,45)	Purple
AGV4	Loading zone	(15,38)	Green
AGV5	Loading zone	(15,42)	Yellow

Table 6.8: Path costs of robots for each task

Tasks	AGV1	AGV2	AGV3	AGV4	AGV5
T1	20.2167	19.7153	19.4039	18.7290	**18.3822**
T2	17.4489	17.4186	17.6356	**16.2000**	16.2798
T3	**13.5524**	**15.2965**	17.2452	15.2677 → 40.1142	16.5331 → 45.6910
T4	37.3578	**36.7339**	**36.3474**	35.8184 → 44.9468	**35.3924 → 38.8986**

Table 6.9: Schedule of tasks

Tasks	T1	T2	T3	T4
AGV	AGV5	AGV4	AGV2 → AGV3	AGV5
Time	18	16 → 17	14 → 20	20
Packing time	120	100	120	110
Total	138	116 → 117	134 → 140	130

Chapter 7

Conclusions and Future Work

Mobile robots have been increasingly implemented in civil areas recently, and their safe and efficient automatic operation drew the researchers' attention. The widely utilized mobile robots include UAVs and AGVs. The navigation system is an essential part of the autonomous system, and path planning is the basis of robot navigation. This thesis reviews the previous studies, proposes the path planning algorithms for a single UAV/AGV, and then the multi-robot path planning approach.

7.1 Single Mobile Robot Path Planning

Chapter 3 presents the mathematical-based UAV path planning. As the UAV is widely applied for aerial photography or environmental monitoring, the research consists of two case studies, one for the terrain and the other one for the multiple building environment. Section 3.1 uses the Quintic Hermite interpolation with the developed cost function to generate the path through waypoints, comparing with the RRT* algorithm.

In previous path-planning research, the popular approaches include classical approaches, heuristic algorithms, bio-inspired methods, and AI-based approaches. The most popular algorithm is PSO and GA based on a survey [9]. From the comparison by the benchmark functions, PSO is chosen as the primary optimization algorithm in this research, but PSO has the weakness of trapping in local optima as other metaheuristic algorithms.

Section 3.2 proposes the Helix-HPSO approach for the multi-building environment for building inspection. It is a new application of building inspection, as most previous studies focus on a single building. The path planning between buildings uses the improved PSO algorithm with HS. It uses the update ratio from HS to update the inertial weight to perform faster computation with fewer iterations when compared to other algorithms. For the single-building inspection, a helix-based path is generated.

Additionally, for industrial applications, PSO-SA is proposed for AGV path planning. It is an intelligent evolutionary algorithm inspired by PSO and SA in Chapter 4. The proposed approach is aimed at avoiding the local optima. It is inspired by the principle of SA with the Metropolis rule to accept new solutions with probabilistic. PSO usually accept better solutions, but the proposed approach allows it to update the personal best by the Metropolis rule. It would not slow down the convergence speed of the algorithm. The proposed approach is compared with other evolutionary algorithms by the benchmark functions and with PSO for path planning. The proposed PSO-SA algorithm has less 80.67% mean runtime than other algorithms.

7.2 Multi-robot Path Planning

The multi-robot system is required in the current industry for operating collaborative robots. Path planning algorithms consist of classical, heuristic algorithms, bio-inspired and AI-based approaches, as concluded in Chapter 2. From the recent surveys in [9], [12], [35], [36], meta-heuristic approaches perform much better than classical methods, regardless of the extension of the search space or the computation complexity. Therefore, the metaheuristic/bio-inspired methods recently became popular in optimising path planning areas. This thesis provides the solution for a multi-robot system based on Chapters 5 and 6.

For the multi-AGV system, another bio-inspired model, the dual layer Weight-Leader-Vicsek-Model (DWLVM), is proposed in Chapter 5. It consists of two layers: centralized and decentralized. For the centralized layer, the approach generates the path for virtual leaders. For the decentralized layer, the AGVs perform deadlock and collision avoidance. Also, the follower-AGVs update their locations and angles

based on the swarm's average angle and the virtual leader's status in the current group. The segment delay function is developed for practical application to separate the specific distance between AGVs. Weight-Leader-Vicsek-Model saves 98.39% computational time than the RRT* algorithm during the computational experiment.

Additionally, from the review of previous studies in Chapter 2, one significant gap is lacking fault tolerance of the multi-robot system. Cultural-PSO is proposed for the multi-AGV system in Chapter 6, considering task allocation, path planning, fault tolerance and collision avoidance. It improves the proposed PSO-SA algorithm with the cultural algorithm to change the inertia weight. The C-PSO updates inertia weight with probabilistic calculated by the Metropolis rule. The proposed algorithm has fewer 63.50% mean iterations than other bio-inspired algorithms. It proposes some rules for task allocation and fault tolerance. When an AGV is turned down, if there is an available AGV, the available AGV gets the remaining task; otherwise, the closest AGV will take it in place after completing the current task list. If an AGV is not operated as expected, the neighbouring AGVs will re-generating the paths.

7.3 Future Work

Most studies only consider single robotics systems [349], so the recommended future work should be focused on the multi-robot system. The requirements of the multi-robot system are not only about functionality, such as collaboration and safety, but also the quality of path planning. The quality of path planning should further achieve completeness, robustness and flexibility by producing an optimal path with a fast and complete algorithm, which can be suitable for most situations.

From the previous Chapter 2, it can be concluded that lacking consideration of real-time implementation is the future work of the path planning algorithms for mobile robots. Most studies achieve online implementation by computational speed or robust algorithms. It should consider the situation that occurred in the practical applications. AI-based approaches require much prior data from the environment, but it still attracts great attention [12], [350].

Moreover, the generic problem formation could be developed for planning the path, including more than path length and collision violations. The cost functions could further consider the energy consumption or kinetics of the robots during operations. Also, the dynamic environment or obstacles could be modelled to achieve more flexible planning.

The future work is not limited to providing more practical algorithms but is also more able to deal with the unknown or dynamic environment. Machine learning approaches can classify or detect changes in the environment or obstacles. Therefore, AI-based approaches such as deep learning or neural networks could be combined with bio-inspired approaches for deterministic strategies.

For further improving experimental performance, the information from robot positioning and perception sensors can be integrated with the proposed path-planning algorithms for more reliable and efficient implementations. Multiple sensors should be employed during the experiment to produce more accurate positions. The non-linear sensor-fusion algorithm should be used, such as the Extended Kalman Filter or the Unscented Kalman Filter.

References

[1] E. Falomir, S. Chaumette, and G. Guerrini, "A 3d mobility model for autonomous swarms of collaborative uavs," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, pp. 196–204, ISBN: 2575-7296. DOI: 10.1109/ICUAS.2019.8798199.

[2] C. Wang, P. Liu, T. Zhang, and J. Sun, "The adaptive vortex search algorithm of optimal path planning for forest fire rescue uav," in *2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC 2018)*, IEEE, pp. 400–403.

[3] P. T. Zacharia and E. K. Xidias, "Agv routing and motion planning in a flexible manufacturing system using a fuzzy-based genetic algorithm," *The International Journal of Advanced Manufacturing Technology*, vol. 109, no. 7-8, pp. 1801–1813, 2020, ISSN: 0268-3768 1433-3015. DOI: 10.1007/s00170-020-05755-3.

[4] G. K. Tevyashov, M. V. Mamchenko, A. N. Migachev, *et al.*, "Algorithm for multi-drone path planning and coverage of agricultural fields," in *Agriculture Digitalization and Organic Production* (Smart Innovation, Systems and Technologies), Smart Innovation, Systems and Technologies. 2022, ch. Chapter 25, pp. 299–310, ISBN: 978-981-16-3348-5 978-981-16-3349-2. DOI: 10.1007/978-981-16-3349-2_25.

[5] V. T. Hoang, M. D. Phung, T. H. Dinh, and Q. P. Ha, "Angle-encoded swarm optimization for uav formation path planning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, pp. 5239–5244, ISBN: 2153-0858. DOI: 10.1109/IROS.2018.8593930.

[6] P. Yao, H. Wang, and Z. Su, "Cooperative path planning with applications to target tracking and obstacle avoidance for multi-uavs," *Aerospace Science and Technology*, vol. 54, pp. 10–22, 2016, ISSN: 12709638. DOI: 10.1016/j.ast.2016.04.002.

[7] M. N. A. Wahab, S. Nefti-Meziani, and A. Atyabi, "A comparative review on mobile robot path planning: Classical or meta-heuristic methods?" *Annual reviews in control*, vol. 50, pp. 233–252, 2020, ISSN: 1367-5788. DOI: 10.1016/j.arcontrol.2020.10.001.

[8] X. Mu, Y. Liu, L. Guo, J. Lin, and R. Schober, "Intelligent reflecting surface enhanced indoor robot path planning: A radio map-based approach," *IEEE transactions on wireless communications*, vol. 20, no. 7, pp. 4732–4747, 2021, ISSN: 1536-1276. DOI: 10.1109/TWC.2021.3062089.

[9] H.-y. Zhang, W.-m. Lin, and A.-x. Chen, "Path planning for the mobile robot: A review," *Symmetry*, vol. 10, no. 10, 2018, ISSN: 2073-8994. DOI: 10.3390/sym10100450.

[10] V. Sathiya, M. Chinnadurai, and S. Ramabalan, "Mobile robot path planning using fuzzy enhanced improved multi-objective particle swarm optimization (fimopso)," *Expert systems with applications*, vol. 198, p. 116 875, 2022, ISSN: 0957-4174. DOI: 10.1016/j.eswa.2022.116875.

[11] T. B. Ionescu, "Adaptive simplex architecture for safe, real-time robot path planning," *Sensors (Basel, Switzerland)*, vol. 21, no. 8, p. 2589, 2021, ISSN: 1424-8220. DOI: 10.3390/s21082589.

[12] Y. Zhao, Y. Zhang, and S. Wang, "A review of mobile robot path planning based on deep reinforcement learning algorithm," *Journal of physics. Conference series*, vol. 2138, no. 1, p. 12 011, 2021, ISSN: 1742-6588. DOI: 10.1088/1742-6596/2138/1/012011.

[13] E. Cardarelli, V. Digani, L. Sabattini, C. Secchi, and C. Fantuzzi, "Cooperative cloud robotics architecture for the coordination of multi-agv systems in industrial warehouses," *Mechatronics*, vol. 45, pp. 1–13, 2017, ISSN: 09574158. DOI: 10.1016/j.mechatronics.2017.04.005.

[14] N. Ahmed, C. J. Pawase, and K. Chang, "Distributed 3-d path planning for multi-uavs with full area surveillance based on particle swarm optimiza-

tion," *Applied Sciences*, vol. 11, no. 8, 2021, ISSN: 2076-3417. DOI: 10.3390/app11083417.

[15] J. Berger and N. Lo, "An innovative multi-agent search-and-rescue path planning approach," *Computers and Operations Research*, vol. 53, pp. 24–31, 2015, ISSN: 03050548. DOI: 10.1016/j.cor.2014.06.016.

[16] R. Nagasawa, E. Mas, L. Moya, and S. Koshimura, "Model-based analysis of multi-uav path planning for surveying postdisaster building damage," *Sci Rep*, vol. 11, no. 1, p. 18 588, 2021, ISSN: 2045-2322 (Electronic) 2045-2322 (Linking). DOI: 10.1038/s41598-021-97804-4.

[17] T. Pereira, A. P. G. M. Moreira, and M. Veloso, "Multi-robot planning for perception of multiple regions of interest," in *ROBOT 2017: Third Iberian Robotics Conference* (Advances in Intelligent Systems and Computing), Advances in Intelligent Systems and Computing. 2018, ch. 23, pp. 275–286, ISBN: 978-3-319-70832-4 978-3-319-70833-1. DOI: 10.1007/978-3-319-70833-1_23.

[18] S. Tian, Y. Li, Y. Kang, and J. Xia, "Multi-robot path planning in wireless sensor networks based on jump mechanism pso and safety gap obstacle avoidance," *Future Generation Computer Systems*, vol. 118, pp. 37–47, 2021, ISSN: 0167739X. DOI: 10.1016/j.future.2020.12.012.

[19] A. Ravankar, A. A. Ravankar, Y. Kobayashi, and T. Emaru, "Symbiotic navigation in multi-robot systems with remote obstacle knowledge sharing," *Sensors (Basel)*, vol. 17, no. 7, 2017, ISSN: 1424-8220 (Electronic) 1424-8220 (Linking). DOI: 10.3390/s17071581.

[20] H. Li, T. Zhao, and S. Dian, "Prioritized planning algorithm for multi-robot collision avoidance based on artificial untraversable vertex," *Applied Intelligence*, vol. 52, no. 1, pp. 429–451, 2021, ISSN: 0924-669X 1573-7497. DOI: 10.1007/s10489-021-02397-0.

[21] D. L. Cruz and W. Yu, "Path planning of multi-agent systems in unknown environment with neural kernel smoothing and reinforcement learning," *Neurocomputing*, vol. 233, pp. 34–42, 2017, ISSN: 09252312. DOI: 10.1016/j.neucom.2016.08.108.

[22] G. Kyprianou, L. Doitsidis, and S. A. Chatzichristofis, "Towards the achievement of path planning with multi-robot systems in dynamic environments," *Journal of Intelligent and Robotic Systems*, vol. 104, no. 1, 2021, ISSN: 0921-0296 1573-0409. DOI: 10.1007/s10846-021-01555-3.

[23] Y. Liu, C. Jiang, X. Zhang, *et al.*, "Reliability-based multivehicle path planning under uncertainty using a bio-inspired approach," *Journal of Mechanical Design*, vol. 144, no. 9, 2022, ISSN: 1050-0472 1528-9001. DOI: 10.1115/1.4053217.

[24] W. Shi, Z. He, W. Tang, W. Liu, and Z. Ma, "Path planning of multi-robot systems with boolean specifications based on simulated annealing," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6091–6098, 2022, ISSN: 2377-3766 2377-3774. DOI: 10.1109/lra.2022.3165184.

[25] S. D. Han, E. J. Rodriguez, and J. Yu, "Sear: A polynomial- time multi-robot path planning algorithm with expected constant-factor optimality guarantee," IEEE, pp. 1–9, ISBN: 2153-0866. DOI: 10.1109/IROS.2018.8594417.

[26] S. MahmoudZadeh, A. Abbasi, A. Yazdani, H. Wang, and Y. Liu, "Uninterrupted path planning system for multi-usv sampling mission in a cluttered ocean environment," *Ocean Engineering*, vol. 254, 2022, ISSN: 00298018. DOI: 10.1016/j.oceaneng.2022.111328.

[27] W. Cai, M. Zhang, and Y. R. Zheng, "Task assignment and path planning for multiple autonomous underwater vehicles using 3d dubins curves (dagger)," *Sensors (Basel)*, vol. 17, no. 7, 2017, ISSN: 1424-8220 (Electronic) 1424-8220 (Linking). DOI: 10.3390/s17071607.

[28] H. Lurz, T. Recker, and A. Raatz, "Spline-based path planning and reconfiguration for rigid multi-robot formations," *Procedia CIRP*, vol. 106, pp. 174–179, 2022, ISSN: 2212-8271. DOI: 10.1016/j.procir.2022.02.174.

[29] A. C. Kapoutsis, S. A. Chatzichristofis, L. Doitsidis, *et al.*, "Real-time adaptive multi-robot exploration with application to underwater map construction," *Autonomous Robots*, vol. 40, no. 6, pp. 987–1015, 2015, ISSN: 0929-5593 1573-7527. DOI: 10.1007/s10514-015-9510-8.

[30] J. Yu and D. Rus, "An effective algorithmic framework for near optimal multi-robot path planning," in *Robotics Research* (Springer Proceedings in Ad-

vanced Robotics), Springer Proceedings in Advanced Robotics. 2018, ch. 30, pp. 495–511, ISBN: 978-3-319-51531-1 978-3-319-51532-8. DOI: 10.1007/978-3-319-51532-8_30.

[31] S. Öztürk and A. E. Kuzucuoğlu, "Optimal bid valuation using path finding for multi-robot task allocation," *Journal of Intelligent Manufacturing*, vol. 26, no. 5, pp. 1049–1062, 2014, ISSN: 0956-5515 1572-8145. DOI: 10.1007/s10845-014-0909-4.

[32] T. Regev and V. Indelman, "Decentralized multi-robot belief space planning in unknown environments via identification and efficient re-evaluation of impacted paths," *Autonomous Robots*, vol. 42, no. 4, pp. 691–713, 2017, ISSN: 0929-5593 1573-7527. DOI: 10.1007/s10514-017-9659-4.

[33] S. Veeramani and S. Muthuswamy, "Hybrid type multi-robot path planning of a serial manipulator and swarmitfix robots in sheet metal milling process," *Complex and Intelligent Systems*, 2021, ISSN: 2199-4536 2198-6053. DOI: 10.1007/s40747-021-00499-3.

[34] F. Gul, I. Mir, L. Abualigah, P. Sumari, and A. Forestiero, "A consolidated review of path planning and optimization techniques: Technical perspectives and future directions," *Electronics*, vol. 10, no. 18, 2021, ISSN: 2079-9292. DOI: 10.3390/electronics10182250.

[35] B. K. Patle, G. Babu L, A. Pandey, D. R. K. Parhi, and A. Jagadeesh, "A review: On path planning strategies for navigation of mobile robot," *Defence Technology*, vol. 15, no. 4, pp. 582–606, 2019, ISSN: 22149147. DOI: 10.1016/j.dt.2019.04.011.

[36] J. R. Sanchez-Ibanez, C. J. Perez-Del-Pulgar, and A. Garcia-Cerezo, "Path planning for autonomous mobile robots: A review," *Sensors (Basel)*, vol. 21, no. 23, 2021, ISSN: 1424-8220 (Electronic) 1424-8220 (Linking). DOI: 10.3390/s21237898.

[37] X. Guo, Z. Ren, Z. Wu, J. Lai, D. Zeng, and S. Xie, "A deep reinforcement learning based approach for agvs path planning," in *2020 Chinese Automation Congress (CAC)*, pp. 6833–6838. DOI: 10.1109/cac51589.2020.9327532.

[38] J. Liu, P. Jayakumar, J. L. Stein, and T. Ersal, "A nonlinear model predictive control formulation for obstacle avoidance in high-speed autonomous ground vehicles in unstructured environments," *Vehicle System Dynamics*, vol. 56, no. 6, pp. 853–882, 2017, ISSN: 0042-3114 1744-5159. DOI: 10.1080/00423114.2017.1399209.

[39] A. Yadav, A. Gaur, S. M. Jain, D. K. Chaturvedi, and R. Sharma, "Development navigation, guidance and control program for gps based autonomous ground vehicle (agv) using soft computing techniques," *Materials Today: Proceedings*, vol. 29, pp. 530–535, 2020, ISSN: 22147853. DOI: 10.1016/j.matpr.2020.07.309.

[40] H. Kong, J. Sun, and J. Hu, "Real-time motion planning based on layered cost map for agv navigation," in *2020 Chinese Automation Congress (CAC)*, pp. 7624–7628. DOI: 10.1109/cac51589.2020.9327401.

[41] J. Li, G. Deng, C. Luo, Q. Lin, Q. Yan, and Z. Ming, "A hybrid path planning method in unmanned air/ground vehicle (uav/ugv) cooperative systems," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 12, pp. 9585–9596, 2016, ISSN: 0018-9545 1939-9359. DOI: 10.1109/tvt.2016.2623666.

[42] H. A. F. Almurib, P. T. Nathan, and T. N. Kumar, "Control and path planning of quadrotor aerial vehicles for search and rescue," in *SICE Annual Conference 2011*, IEEE, pp. 700–705, ISBN: 9781457707148.

[43] Z. Yuan, Z. Yang, L. Lv, and Y. Shi, "A bi-level path planning algorithm for multi-agv routing problem," *Electronics*, vol. 9, no. 9, 2020, ISSN: 2079-9292. DOI: 10.3390/electronics9091351.

[44] V. Digani, M. A. Hsieh, L. Sabattini, and C. Secchi, "Coordination of multiple agvs: A quadratic optimization method," *Autonomous Robots*, vol. 43, no. 3, pp. 539–555, 2018, ISSN: 0929-5593 1573-7527. DOI: 10.1007/s10514-018-9730-9.

[45] M. Cramer, J. Cramer, D. De Schepper, P. Aerts, K. Kellens, and E. Demeester, "Benchmarking low-cost inertial measurement units for indoor localisation and navigation of agvs," *Procedia CIRP*, vol. 86, pp. 204–209, 2019.

[46] M. Kostov, V. Kostova, and R. Markoska, "Agv guidance system simulations with a programmable robotics kit," *International journal of reasoning-based intelligent systems*, vol. 7, no. 1-2, pp. 42–46, 2015.

[47] L. Lynch, T. Newe, J. Clifford, J. Coleman, J. Walsh, and D. Toal, "Automated ground vehicle (agv) and sensor technologies- a review," in *Twelfth International Conference on Sensing Technology (ICST)*, pp. 347–352.

[48] P. Li, Y. Xu, T. Shen, and S. Bi, "Ins/uwb integrated agv localization employing kalman filter for indoor los/nlos mixed environment," in *International Conference on Advanced Mechatronic Systems (ICAMechS)*, pp. 294–298.

[49] Z. Cui, H. Xu, Z. Chen, H. Yang, S. Huang, and M. Gong, "Design of a novel agv with automatic pick-and-place system based on scissor lifting platform," in *2020 Chinese Automation Congress (CAC)*, pp. 4435–4440. DOI: 10.1109/cac51589.2020.9327003.

[50] S.-Y. Lee and H.-W. Yang, "Navigation of automated guided vehicles using magnet spot guidance method," *Robotics and Computer-Integrated Manufacturing*, vol. 28, no. 3, pp. 425–436, 2012, ISSN: 07365845. DOI: 10.1016/j.rcim.2011.11.005.

[51] W. Xing, L. Peihuang, Y. Jun, Q. Xiaoming, and T. Dunbing, "Intersection recognition and guide-path selection for a vision-based agv in a bidirectional flow network," *International Journal of Advanced Robotic Systems*, vol. 11, no. 3, 2014, ISSN: 1729-8814 1729-8814. DOI: 10.5772/58218.

[52] Z. Fu, X. Feng, X. Duan, and Z. Fu, "An improved integrated navigation method based on rins, gnss and kinematics for port heavy-duty agv," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 234, no. 8, pp. 2135–2153, 2020, ISSN: 0954-4070 2041-2991. DOI: 10.1177/0954407019900031.

[53] H. Wang, X. Zhang, X. Li, L. Han, and J. Zhang, "Gps/dr information fusion for agv navigation," in *World Automation Congress*, 2012, pp. 1–4.

[54] J. Lee, C. H. Hyun, and M. Park, "A vision-based automated guided vehicle system with marker recognition for indoor use," *Sensors (Basel)*, vol. 13, no. 8, pp. 10 052–73, 2013, ISSN: 1424-8220 (Electronic) 1424-8220 (Linking). DOI: 10.3390/s130810052.

[55] G. G. Rigatos, "Derivative-free distributed filtering for integrity monitoring of agv navigation sensors," in *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pp. 299–304.

[56] T.-c. Wang, C.-s. Tong, and B.-l. Xu, "Agv navigation analysis based on multi-sensor data fusion," *Multimedia Tools and Applications*, vol. 79, no. 7-8, pp. 5109–5124, 2018, ISSN: 1380-7501 1573-7721. DOI: 10.1007/s11042-018-6336-3.

[57] L. Liu, R. Shi, S. Li, and . J. Wu, "Path planning for uavs based on improved artificial potential field method through changing the repulsive potential function," in *2016 IEEE Chinese Guidance, Navigation and Control Conference*, IEEE, pp. 2011–2015.

[58] J. Verbeke, J. Vantilt, D. Vanthienen, M. Vochten, S. Debruyne, and J. D. Schutter, "A constraint-based flight control system architecture for uavs using the itasc framework," in *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, pp. 310–319.

[59] L. Babel, "Three-dimensional route planning for unmanned aerial vehicles in a risk environment," *Journal of Intelligent and Robotic Systems*, vol. 71, no. 2, pp. 255–269, 2012, ISSN: 0921-0296 1573-0409. DOI: 10.1007/s10846-012-9773-7.

[60] X. Wang, Y. Shi, D. Ding, and X. Gu, "Double global optimum genetic algorithm–particle swarm optimization-based welding robot path planning," *Engineering Optimization*, vol. 48, no. 2, pp. 299–316, 2015, ISSN: 0305-215X 1029-0273. DOI: 10.1080/0305215x.2015.1005084.

[61] I. Khademi, B. Maleki, and A. N. Mood, "Optimal three dimensional terrain following/terrain avoidance for aircraft using direct transcription method," in *19th Mediterranean Conference on Control and Automation*, IEEE, pp. 254–258.

[62] L. Babel, "Flight path planning for unmanned aerial vehicles with landmark-based visual navigation," *Robotics and Autonomous Systems*, vol. 62, no. 2, pp. 142–150, 2014, ISSN: 09218890. DOI: 10.1016/j.robot.2013.11.004.

[63] X. Yang, M. Ding, and C. Zhou, "Fast marine route planning for uav using improved sparse a* algorithm," in *2010 Fourth International Conference on*

Genetic and Evolutionary Computing, IEEE, pp. 190–193. DOI: 10.1109/icgec.2010.54.

[64] B. Meng and X. Gao, "Uav path planning based on bidirectional sparse a* search algorithm," in *2010 International Conference on Intelligent Computation Technology and Automation*, IEEE, pp. 1106–1109. DOI: 10.1109/ICICTA.2010.235.

[65] L. Babel, "Flight path optimization with application to in-flight replanning to changing destinations," *Aircraft Engineering and Aerospace Technology*, pp. 1192–1202, 2018, ISSN: 0002-2667. DOI: 10.1108/aeat-05-2016-0088.

[66] B. M. Sathyaraj, L. C. Jain, A. Finn, and S. Drake, "Multiple uavs path planning algorithms: A comparative study," *Fuzzy Optimization and Decision Making*, vol. 7, no. 3, pp. 257–267, 2008, ISSN: 1568-4539 1573-2908. DOI: 10.1007/s10700-008-9035-0.

[67] V. Roberge, M. Tarbouchi, and G. Labonte, "Comparison of parallel genetic algorithm and particle swarm optimization for real-time uav path planning," in *IEEE Transactions on Industrial Informatics*, vol. 9, British Crown, pp. 132–141, ISBN: 1551-3203 1941-0050. DOI: 10.1109/tii.2012.2198665.

[68] P. B. Sujit and R. Beard, "Multiple uav path planning using anytime algorithms," in *2009 American Control Conference*, AACC, pp. 2978–2983.

[69] J. Jamieson and J. Biggs, "Path planning using concatenated analytically-defined trajectories for quadrotor uavs," *Aerospace*, vol. 2, no. 2, pp. 155–170, 2015, ISSN: 2226-4310. DOI: 10.3390/aerospace2020155.

[70] Z. Dong, W. Li, and Y. Zhou, "An autonomous navigation scheme for uav in approach phase," in *2016 IEEE Chinese Guidance, Navigation and Control Conference*, IEEE, pp. 982–987.

[71] D. Lee and D. H. Shim, "Spline-rrt* based optimal path planning of terrain following flights for fixed-wing uavs," in *The 11th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI 2014)*, IEEE, pp. 257–261.

[72] K. Suzuki, R. Shigeta, Y. Kawahara, and T. Asami, "Bilateration-based position estimation of sensor nodes for uav-assisted wireless power transfer systems," in *the 13th International Conference on Mobile and Ubiquitous Sys-*

tems: Computing Networking and Services - MOBIQUITOUS 2016, ACM, pp. 66–71. DOI: 10.1145/3004010.3004015.

[73] B. Liu, G. Wei, W. Xi, and X. Fu, "Multi-uavs cooperative area search with no-fly zones constraints," in *the 34th Chinese Control Conference*, 2015, pp. 6038–6042.

[74] X. Li, F. Liu, J. Liu, and S. Liang, "Obstacle avoidance for mobile robot based on improved dynamic window approach," *TURKISH JOURNAL OF ELECTRICAL ENGINEERING and COMPUTER SCIENCES*, vol. 25, pp. 666–676, 2017, ISSN: 1300-0632. DOI: 10.3906/elk-1504-194.

[75] L. Babel, "Trajectory planning for unmanned aerial vehicles: A network optimization approach," *Math Meth Oper Res (2011)*, vol. 74, pp. 343–360, 2011. DOI: 10.1007/s00186-011-0366-1.

[76] T. W. McLain, P. R. Chandler, S. Rasmussen, and M. Pachter, "Cooperative control of uav rendezvous," in *the American Control Conference*, AACC, 2001, pp. 2309–2314.

[77] L. Zhang, J. Z. Lai, P. Shi, S. Bao, and J. Y. Liu, "An improved mcs/ins integrated navigation algorithm for multi-rotor uav in indoor flight," in *2016 IEEE Chinese Guidance, Navigation and Control Conference*, IEEE, 2016, pp. 2099–2104.

[78] P. Dong, X. Ruan, J. Huang, X. Zhu, and Y. Xiao, "A rgb-d slam algorithm combining orb features and bow," in *the 2nd International Conference on Computer Science and Application Engineering - CSAE '18*, 2018, pp. 1–6. DOI: 10.1145/3207677.3278061.

[79] F. Vanegas and F. Gonzalez, "Enabling uav navigation with sensor and environmental uncertainty in cluttered and gps-denied environments," *Sensors (Basel)*, vol. 16, no. 5, pp. 1–17, 2016, ISSN: 1424-8220 (Electronic) 1424-8220 (Linking). DOI: 10.3390/s16050666. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/27171096.

[80] P. Agrawal, A. Ratnoo, and D. Ghose, "Inverse optical flow based guidance for uav navigation through urban canyons," *Aerospace Science and Technology*, vol. 68, pp. 163–178, 2017, ISSN: 12709638. DOI: 10.1016/j.ast.2017.05.012.

[81] G. Fornari, V. A. de Santiago Júnior, and E. H. Shiguemori, "A self-adaptive approach for autonomous uav navigation via computer vision," in *Computational Science and Its Applications – ICCSA 2018* (Lecture Notes in Computer Science), Lecture Notes in Computer Science. 2018, ch. 19, pp. 268–280, ISBN: 978-3-319-95164-5 978-3-319-95165-2. DOI: 10.1007/978-3-319-95165-2_19.

[82] D.-Y. Gu, C.-F. Zhu, J. Guo, S.-X. Li, and H.-X. Chang, "Vision-aided uav navigation using gis data," in *Proceedings of 2010 IEEE International Conference on Vehicular Electronics and Safety*, 2010, pp. 78–82.

[83] N. Stefas, H. Bayram, and V. Isler, "Vision-based uav navigation in orchards," *FAC PapersOnLine*, vol. 49, no. 16, pp. 10–15, 2016.

[84] F. J. Perez-Grau, R. Ragel, F. Caballero, A. Viguria, and A. Ollero, "An architecture for robust uav navigation in gps-denied areas," *Journal of Field Robotics*, vol. 35, no. 1, pp. 121–145, 2018, ISSN: 15564959. DOI: 10.1002/rob.21757.

[85] D. O. Nitti, F. Bovenga, M. T. Chiaradia, M. Greco, and G. Pinelli, "Feasibility of using synthetic aperture radar to aid uav navigation," *Sensors (Basel)*, vol. 15, no. 8, pp. 18 334–59, 2015, ISSN: 1424-8220 (Electronic) 1424-8220 (Linking). DOI: 10.3390/s150818334.

[86] E. Michaelsen, K. Jäger, D. Roschkowski, L. Doktorski, and M. Arens, "Object-oriented landmark recognition for uav-navigation," *Pattern Recognition and Image Analysis*, vol. 21, no. 2, pp. 152–155, 2011, ISSN: 1054-6618 1555-6212. DOI: 10.1134/s1054661811020763.

[87] S. Ashraf, P. Aggarwal, P. Damacharla, H. Wang, A. Y. Javaid, and V. Devabhaktuni, "A low-cost solution for unmanned aerial vehicle navigation in a global positioning system–denied environment," *International Journal of Distributed Sensor Networks*, vol. 14, no. 6, pp. 1–17, 2018, ISSN: 1550-1477 1550-1477. DOI: 10.1177/1550147718781750.

[88] S. Bao, J. Lai, Z. Chen, P. Lyu, and W. Chen, "Aerodynamic model/ins/gps failure-tolerant navigation method for multirotor uavs based on federated kalman filter," in *2017 Chinese Automation Congress (CAC)*, IEEE, 2017, pp. 1121–1125. DOI: 10.1109/CAC.2017.8242934.

[89] J. Bu, R. Sun, H. Bai, *et al.*, "Integrated method for the uav navigation sensor anomaly detection," *IET Radar, Sonar and Navigation*, vol. 11, no. 5, pp. 847–853, 2017, ISSN: 1751-8784 1751-8792. DOI: 10.1049/iet-rsn.2016.0427.

[90] S. T. Goh, O. Abdelkhalik, and S. A. Zekavat, "A weighted measurement fusion kalman filter implementation for uav navigation," *Aerospace Science and Technology*, vol. 28, no. 1, pp. 315–323, 2013, ISSN: 12709638. DOI: 10.1016/j.ast.2012.11.012.

[91] A. F. Scannapieco, A. Renga, G. Fasano, and A. Moccia, "Ultralight radar for small and micro-uav navigation," *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLII-2/W6, pp. 333–338, 2017, ISSN: 2194-9034. DOI: 10.5194/isprs-archives-XLII-2-W6-333-2017.

[92] R. P. Padhy, F. Xia, S. K. Choudhury, P. K. Sa, and S. Bakshi, "Monocular vision aided autonomous uav navigation in indoor corridor environments," *IEEE Transactions on Sustainable Computing*, vol. 4, no. 1, pp. 96–108, 2019, ISSN: 2377-3782 2377-3790. DOI: 10.1109/tsusc.2018.2810952.

[93] B. M. Miller, K. V. Stepanyan, A. K. Popov, and A. B. Miller, "Uav navigation based on videosequences captured by the onboard video camera," *Automation and Remote Control*, vol. 78, no. 12, pp. 2211–2221, 2017, ISSN: 0005-1179 1608-3032. DOI: 10.1134/s0005117917120098.

[94] Y. A. Nijsure, G. Kaddoum, N. Khaddaj Mallat, G. Gagnon, and F. Gagnon, "Cognitive chaotic uwb-mimo detect-avoid radar for autonomous uav navigation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 11, pp. 3121–3131, 2016, ISSN: 1524-9050 1558-0016. DOI: 10.1109/tits.2016.2539002.

[95] M. Liu and H. Wang, "Research on agriculture survey and evaluation uav navigation system," *INMATEH - Agricultural Engineering*, vol. 41, no. 3, pp. 31–38, 2013.

[96] Y. Chen, J. Yu, X. Su, and G. Luo, "Path planning for multi-uav formation," *Journal of Intelligent and Robotic Systems*, vol. 77, no. 1, pp. 229–246, 2014, ISSN: 0921-0296 1573-0409. DOI: 10.1007/s10846-014-0077-y.

[97] H. Chen, Q. Wang, M. Yu, J. Cao, and J. Sun, "Path planning for multi-robot systems in intelligent warehouse," in *Internet and Distributed Computing Systems* (Lecture Notes in Computer Science), Lecture Notes in Computer Science. 2018, ch. Chapter 13, pp. 148–159, ISBN: 978-3-030-02737-7 978-3-030-02738-4. DOI: 10.1007/978-3-030-02738-4_13.

[98] M. Nazarahari, E. Khanmirza, and S. Doostie, "Multi-objective multi-robot path planning in continuous environment using an enhanced genetic algorithm," *Expert Systems with Applications*, vol. 115, pp. 106–120, 2019, ISSN: 09574174. DOI: 10.1016/j.eswa.2018.08.008.

[99] H. Wang and W. Chen, "Multi-robot path planning with due times," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4829–4836, 2022, ISSN: 2377-3766 2377-3774. DOI: 10.1109/lra.2022.3152701.

[100] B. Wang, K. Zhou, and J. Qu, "Research on multi-robot local path planning based on improved artificial potential field method," in *Proceedings of the Fifth Euro-China Conference on Intelligent Data Analysis and Applications* (Advances in Intelligent Systems and Computing), Advances in Intelligent Systems and Computing. 2019, ch. 77, pp. 684–690, ISBN: 978-3-030-03765-9 978-3-030-03766-6. DOI: 10.1007/978-3-030-03766-6_77.

[101] T. Zhao, H. Li, and S. Dian, "Multi-robot path planning based on improved artificial potential field and fuzzy inference system," *Journal of Intelligent and Fuzzy Systems*, vol. 39, no. 5, pp. 7621–7637, 2020, ISSN: 10641246 18758967. DOI: 10.3233/jifs-200869.

[102] C. He, Y. Wan, Y. Gu, and F. L. Lewis, "Integral reinforcement learning-based multi-robot minimum time-energy path planning subject to collision avoidance and unknown environmental disturbances," *IEEE Control Systems Letters*, vol. 5, no. 3, pp. 983–988, 2021, ISSN: 2475-1456. DOI: 10.1109/lcsys.2020.3007663.

[103] E. Turki and H. Al-Rawi, "Multi-robot path-planning problem for a heavy traffic control application: A survey," *International journal of advanced computer science and applications*, vol. 7, no. 6, 2016, ISSN: 2158-107X. DOI: 10.14569/IJACSA.2016.070623.

[104] K. Solovey, O. Salzman, and D. Halperin, "Finding a needle in an exponential haystack: Discrete rrt for exploration of implicit roadmaps in multi-robot motion planning," *The International Journal of Robotics Research*, vol. 35, no. 5, pp. 501–513, 2016, ISSN: 0278-3649 1741-3176. DOI: 10.1177/0278364915615688.

[105] L. Shen, Y. Wang, K. Liu, *et al.*, "Synergistic path planning of multi-uavs for air pollution detection of ships in ports," *Transportation Research Part E: Logistics and Transportation Review*, vol. 144, 2020, ISSN: 13665545. DOI: 10.1016/j.tre.2020.102128.

[106] A. Pintado and M. Santos, "A first approach to path planning coverage with multi-uavs," in *15th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2020)* (Advances in Intelligent Systems and Computing), Advances in Intelligent Systems and Computing. 2021, ch. 64, pp. 667–677, ISBN: 978-3-030-57801-5 978-3-030-57802-2. DOI: 10.1007/978-3-030-57802-2_64.

[107] J. Yu, "Intractability of optimal multirobot path planning on planar graphs," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 33–40, 2016, ISSN: 2377-3766 2377-3774. DOI: 10.1109/lra.2015.2503143.

[108] A. Nedjati, G. Izbirak, B. Vizvari, and J. Arkat, "Complete coverage path planning for a multi-uav response system in post-earthquake assessment," *Robotics*, vol. 5, no. 4, 2016, ISSN: 2218-6581. DOI: 10.3390/robotics5040026.

[109] G. S. Avellar, G. A. Pereira, L. C. Pimenta, and P. Iscold, "Multi-uav routing for area coverage and remote sensing with minimum time," *Sensors (Basel)*, vol. 15, no. 11, pp. 27 783–803, 2015, ISSN: 1424-8220 (Electronic) 1424-8220 (Linking). DOI: 10.3390/s151127783.

[110] S.-W. Cho, J.-H. Park, H.-J. Park, and S. Kim, "Multi-uav coverage path planning based on hexagonal grid decomposition in maritime search and rescue," *Mathematics*, vol. 10, no. 1, 2021, ISSN: 2227-7390. DOI: 10.3390/math10010083.

[111] E. Turki and H. Al-Rawi, "Mrppsim: A multi-robot path planning simulation," *International journal of advanced computer science and applications*, vol. 7, no. 8, 2016, ISSN: 2158-107X. DOI: 10.14569/IJACSA.2016.070821.

[112] A. Dutta, A. Bhattacharya, O. P. Kreidl, A. Ghosh, and P. Dasgupta, "Multi-robot informative path planning in unknown environments through continuous region partitioning," *International Journal of Advanced Robotic Systems*, vol. 17, no. 6, 2020, ISSN: 1729-8814 1729-8814. DOI: 10.1177/1729881420970461.

[113] S.-K. Huang, W.-J. Wang, and C.-H. Sun, "A path planning strategy for multi-robot moving with path-priority order based on a generalized voronoi diagram," *Applied Sciences*, vol. 11, no. 20, 2021, ISSN: 2076-3417. DOI: 10.3390/app11209650.

[114] H. Zheng and J. Yuan, "An integrated mission planning framework for sensor allocation and path planning of heterogeneous multi-uav systems," *Sensors (Basel)*, vol. 21, no. 10, 2021, ISSN: 1424-8220 (Electronic) 1424-8220 (Linking). DOI: 10.3390/s21103557. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/34065325.

[115] X. Sun, Y. Liu, W. Yao, and N. Qi, "Triple-stage path prediction algorithm for real-time mission planning of multi-uav," *Electronics Letters*, vol. 51, no. 19, pp. 1490–1492, 2015, ISSN: 0013-5194 1350-911X. DOI: 10.1049/el.2015.1244.

[116] A. K. Singh, "Fault-detection on multi-robot path planning," *International Journal of Advanced Research in Computer Science*, vol. 8, no. 8, pp. 539–543, 2017, ISSN: 09765697. DOI: 10.26483/ijarcs.v8i8.4832.

[117] Z. Wang and S. Zlatanova, "Multi-agent based path planning for first responders among moving obstacles," *Computers, Environment and Urban Systems*, vol. 56, pp. 48–58, 2016, ISSN: 01989715. DOI: 10.1016/j.compenvurbsys.2015.11.001.

[118] Zagradjanin, Pamucar, and Jovanovic, "Cloud-based multi-robot path planning in complex and crowded environment with multi-criteria decision making using full consistency method," *Symmetry*, vol. 11, no. 10, 2019, ISSN: 2073-8994. DOI: 10.3390/sym11101241.

[119] G. Serpen and C. Dou, "Automated robotic parking systems: Real-time, concurrent and multi-robot path planning in dynamic environments," *Applied Intelligence*, vol. 42, no. 2, pp. 231–251, 2014, ISSN: 0924-669X 1573-7497. DOI: 10.1007/s10489-014-0598-x.

[120] M. Salerno, Y. E-Martín, R. Fuentetaja, A. Gragera, A. Pozanco, and D. Borrajo, "Train route planning as a multi-agent path finding problem," in *Advances in Artificial Intelligence* (Lecture Notes in Computer Science), Lecture Notes in Computer Science. 2021, ch. 23, pp. 237–246, ISBN: 978-3-030-85712-7 978-3-030-85713-4. DOI: 10.1007/978-3-030-85713-4_23.

[121] J. Bae and W. Chung, "Efficient path planning for multiple transportation robots under various loading conditions," *International Journal of Advanced Robotic Systems*, vol. 16, no. 2, 2019, ISSN: 1729-8814 1729-8814. DOI: 10.1177/1729881419835110.

[122] D. Gujarathi and I. Saha, "Mt : Multi-robot path planning for temporal logic specifications," *arXiv.org*, 2021. DOI: 10.48550/arxiv.2103.02821.

[123] V. Modi, Y. Chen, A. Madan, S. Sueda, and D. I. W. Levin, "Multi-agent path planning with asymmetric interactions in tight spaces," *arXiv.org*, 2022. DOI: 10.48550/arxiv.2204.00567.

[124] N. N. Yu, T. K. Li, B. L. Wang, S. P. Yuan, and Y. Wang, "Reliability oriented multi-agvs online scheduling and path planning problem of automated sorting warehouse system," *IOP conference series. Materials Science and Engineering*, vol. 1043, no. 2, p. 22 035, 2021, ISSN: 1757-8981. DOI: 10.1088/1757-899X/1043/2/022035.

[125] M. A. Luna, M. S. Ale Isaac, A. R. Ragab, P. Campoy, P. Flores Pena, and M. Molina, "Fast multi-uav path planning for optimal area coverage in aerial sensing applications," *Sensors (Basel)*, vol. 22, no. 6, 2022, ISSN: 1424-8220 (Electronic) 1424-8220 (Linking). DOI: 10.3390/s22062297.

[126] H. Kim, D. Kim, H. Kim, J.-U. Shin, and H. Myung, "An extended any-angle path planning algorithm for maintaining formation of multi-agent jellyfish elimination robot system," *International Journal of Control, Automation and Systems*, vol. 14, no. 2, pp. 598–607, 2016, ISSN: 1598-6446 2005-4092. DOI: 10.1007/s12555-014-0349-0.

[127] E. N. Mobarez, A. Sarhan, and M. M. Ashry, "Obstacle avoidance for multi-uav path planning based on particle swarm optimization," *IOP conference series. Materials Science and Engineering*, vol. 1172, no. 1, p. 12 039, 2021, ISSN: 1757-8981. DOI: 10.1088/1757-899X/1172/1/012039.

[128] Z. Chen, H. Wu, Y. Chen, L. Cheng, and B. Zhang, "Patrol robot path planning in nuclear power plant using an interval multi-objective particle swarm optimization algorithm," *Applied Soft Computing*, vol. 116, 2022, ISSN: 15684946. DOI: 10.1016/j.asoc.2021.108192.

[129] Y. Chen, S. Ren, Z. Chen, M. Chen, and H. Wu, "Path planning for vehicle-borne system consisting of multi air–ground robots," *Robotica*, vol. 38, no. 3, pp. 493–511, 2019, ISSN: 0263-5747 1469-8668. DOI: 10.1017/s0263574719000808.

[130] P. K. Das, H. S. Behera, S. Das, H. K. Tripathy, B. K. Panigrahi, and S. K. Pradhan, "A hybrid improved pso-dv algorithm for multi-robot path planning in a clutter environment," *Neurocomputing*, vol. 207, pp. 735–753, 2016, ISSN: 09252312. DOI: 10.1016/j.neucom.2016.05.057.

[131] W. He, X. Qi, and L. Liu, "A novel hybrid particle swarm optimization for multi-uav cooperate path planning," *Applied Intelligence*, vol. 51, no. 10, pp. 7350–7364, 2021, ISSN: 0924-669X 1573-7497. DOI: 10.1007/s10489-020-02082-8.

[132] M. R. Panda, P. Das, and S. Pradhan, "Hybridization of iwo and ipso for mobile robots navigation in a dynamic environment," *Journal of King Saud University - Computer and Information Sciences*, vol. 32, no. 9, pp. 1020–1033, 2020, ISSN: 13191578. DOI: 10.1016/j.jksuci.2017.12.009.

[133] Z. Shao, F. Yan, Z. Zhou, and X. Zhu, "Path planning for multi-uav formation rendezvous based on distributed cooperative particle swarm optimization," *Applied Sciences*, vol. 9, no. 13, 2019, ISSN: 2076-3417. DOI: 10.3390/app9132621.

[134] H. K. Paikray, P. K. Das, and S. Panda, "Optimal multi-robot path planning using particle swarm optimization algorithm improved by sine and cosine algorithms," *Arabian Journal for Science and Engineering*, vol. 46, no. 4, pp. 3357–3381, 2021, ISSN: 2193-567X 2191-4281. DOI: 10.1007/s13369-020-05046-9.

[135] B. Tang, K. Xiang, M. Pang, and Z. Zhanxia, "Multi-robot path planning using an improved self-adaptive particle swarm optimization," *International Journal of Advanced Robotic Systems*, vol. 17, no. 5, 2020, ISSN: 1729-8814 1729-8814. DOI: 10.1177/1729881420936154.

[136] P. K. Das, H. S. Behera, and B. K. Panigrahi, "A hybridization of an improved particle swarm optimization and gravitational search algorithm for multi-robot path planning," *Swarm and Evolutionary Computation*, vol. 28, pp. 14–28, 2016, ISSN: 22106502. DOI: 10.1016/j.swevo.2015.10.011.

[137] B. Sahu, P. Kumar Das, and M. r. Kabat, "Multi-robot cooperation and path planning for stick transporting using improved q-learning and democratic robotics pso," *Journal of Computational Science*, vol. 60, 2022, ISSN: 18777503. DOI: 10.1016/j.jocs.2022.101637.

[138] M. Zhong, Y. Yang, Y. Dessouky, and O. Postolache, "Multi-agv scheduling for conflict-free path planning in automated container terminals," *Computers and Industrial Engineering*, vol. 142, 2020, ISSN: 03608352. DOI: 10.1016/j.cie.2020.106371.

[139] R. A. Saeed, D. Reforgiato Recupero, and P. Remagnino, "The boundary node method for multi-robot multi-goal path planning problems," *Expert Systems*, vol. 38, no. 6, 2021, ISSN: 0266-4720 1468-0394. DOI: 10.1111/exsy.12691.

[140] J. Song, L. Liu, Y. Liu, J. Xi, W. Zhai, and G. Yang, "Path planning for multi-vehicle-assisted multi-uavs in mobile crowdsensing," *Wireless Communications and Mobile Computing*, vol. 2022, pp. 1–21, 2022, ISSN: 1530-8677 1530-8669. DOI: 10.1155/2022/9778188.

[141] J. Ru, S. Yu, H. Wu, *et al.*, "A multi-auv path planning system based on the omni-directional sensing ability," *Journal of Marine Science and Engineering*, vol. 9, no. 8, 2021, ISSN: 2077-1312. DOI: 10.3390/jmse9080806.

[142] G. Sun, R. Zhou, B. Di, Z. Dong, and Y. Wang, "A novel cooperative path planning for multi-robot persistent coverage with obstacles and coverage period constraints," *Sensors (Basel)*, vol. 19, no. 9, 2019, ISSN: 1424-8220 (Electronic) 1424-8220 (Linking). DOI: 10.3390/s19091994.

[143] S. Yanes Luis, F. Peralta, A. Tapia Córdoba, Á. Rodríguez del Nozal, S. Toral Marín, and D. Gutiérrez Reina, "An evolutionary multi-objective path planning of a fleet of asvs for patrolling water resources," *Engineering Applications of Artificial Intelligence*, vol. 112, 2022, ISSN: 09521976. DOI: 10.1016/j.engappai.2022.104852.

[144] R. Sun, C. Tang, J. Zheng, Y. Zhou, and S. Yu, "Multi-robot path planning for complete coverage with genetic algorithms," in *Intelligent Robotics and Applications* (Lecture Notes in Computer Science), Lecture Notes in Computer Science. 2019, ch. 29, pp. 349–361, ISBN: 978-3-030-27540-2 978-3-030-27541-9. DOI: 10.1007/978-3-030-27541-9_29.

[145] M. Xu, B. Xin, L. Dou, and G. Gao, "A cell potential and motion pattern driven multi-robot coverage path planning algorithm," in *Bio-inspired Computing: Theories and Applications* (Communications in Computer and Information Science), Communications in Computer and Information Science. 2020, ch. 36, pp. 468–483, ISBN: 978-981-15-3424-9 978-981-15-3425-6. DOI: 10.1007/978-981-15-3425-6_36.

[146] R. Sarkar, D. Barman, and N. Chowdhury, "A cooperative co-evolutionary genetic algorithm for multi-robot path planning having multiple targets," in *Computational Intelligence in Pattern Recognition* (Advances in Intelligent Systems and Computing), Advances in Intelligent Systems and Computing. 2020, ch. 63, pp. 727–740, ISBN: 978-981-13-9041-8 978-981-13-9042-5. DOI: 10.1007/978-981-13-9042-5_63.

[147] B. Farooq, J. Bao, H. Raza, Y. Sun, and Q. Ma, "Flow-shop path planning for multi-automated guided vehicles in intelligent textile spinning cyber-physical production systems dynamic environment," *Journal of Manufacturing Systems*, vol. 59, pp. 98–116, 2021, ISSN: 02786125. DOI: 10.1016/j.jmsy.2021.01.009.

[148] Z. Han, D. Wang, F. Liu, and Z. Zhao, "Multi-agv path planning with double-path constraints by using an improved genetic algorithm," *PLoS One*, vol. 12, no. 7, e0181747, 2017, ISSN: 1932-6203 (Electronic) 1932-6203 (Linking). DOI: 10.1371/journal.pone.0181747. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/28746355.

[149] W. Xu, "Path planning for multi-agv systems based on two-stage scheduling," *International Journal of Performability Engineering*, 2017. DOI: 10.23940/ijpe.17.08.p16.13471357.

[150] H. Huang and T. Zhuo, "Multi-model cooperative task assignment and path planning of multiple ucav formation," *Multimedia Tools and Applications*,

vol. 78, no. 1, pp. 415–436, 2017, ISSN: 1380-7501 1573-7721. DOI: 10.1007/s11042-017-4956-7.

[151] G. Yi, Z. Feng, T. Mei, P. Li, W. Jin, and S. Chen, "Multi-agvs path planning based on improved ant colony algorithm," *The Journal of Supercomputing*, vol. 75, no. 9, pp. 5898–5913, 2019, ISSN: 0920-8542 1573-0484. DOI: 10.1007/s11227-019-02884-9.

[152] J. Liu, S. Anavatti, M. Garratt, and H. A. Abbass, "Modified continuous ant colony optimisation for multiple unmanned ground vehicle path planning," *Expert Systems with Applications*, vol. 196, 2022, ISSN: 09574174. DOI: 10.1016/j.eswa.2022.116605.

[153] L. Huang, H. Qu, P. Ji, X. Liu, and Z. Fan, "A novel coordinated path planning method using k-degree smoothing for multi-uavs," *Applied Soft Computing*, vol. 48, pp. 182–192, 2016, ISSN: 15684946. DOI: 10.1016/j.asoc.2016.06.046.

[154] N. Botteghi, A. Kamilaris, L. Sinai, and B. Sirmacek, "Multi-agent path planning of robotic swarms in agricultural fields," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. V-1-2020, pp. 361–368, 2020, ISSN: 2194-9050. DOI: 10.5194/isprs-annals-V-1-2020-361-2020.

[155] D. Zhang and H. Duan, "Social-class pigeon-inspired optimization and time stamp segmentation for multi-uav cooperative path planning," *Neurocomputing*, vol. 313, pp. 229–246, 2018, ISSN: 09252312. DOI: 10.1016/j.neucom.2018.06.032.

[156] B. H. Wang, D. B. Wang, and Z. A. Ali, "A cauchy mutant pigeon-inspired optimization–based multi-unmanned aerial vehicle path planning method," *Measurement and Control*, vol. 53, no. 1-2, pp. 83–92, 2020, ISSN: 0020-2940. DOI: 10.1177/0020294019885155.

[157] C. Xu, M. Xu, and C. Yin, "Optimized multi-uav cooperative path planning under the complex confrontation environment," *Computer Communications*, vol. 162, pp. 196–203, 2020, ISSN: 01403664. DOI: 10.1016/j.comcom.2020.04.050.

[158] M. Zhou, Z. Wang, J. Wang, and Z. Dong, "A hybrid path planning and formation control strategy of multi-robots in a dynamic environment," *Journal of Advanced Computational Intelligence and Intelligent Informatics*, vol. 26, no. 3, pp. 342–354, 2022, ISSN: 1883-8014 1343-0130. DOI: 10.20965/jaciii.2022.p0342.

[159] G. Huang, Y. Cai, J. Liu, Y. Qi, and X. Liu, "A novel hybrid discrete grey wolf optimizer algorithm for multi-uav path planning," *Journal of Intelligent and Robotic Systems*, vol. 103, no. 3, 2021, ISSN: 0921-0296 1573-0409. DOI: 10.1007/s10846-021-01490-3.

[160] K. Shi, X. Zhang, and S. Xia, "Multiple swarm fruit fly optimization algorithm based path planning method for multi-uavs," *Applied Sciences*, vol. 10, no. 8, 2020, ISSN: 2076-3417. DOI: 10.3390/app10082822.

[161] P. K. Das, H. S. Behera, P. K. Jena, and B. K. Panigrahi, "Multi-robot path planning in a dynamic environment using improved gravitational search algorithm," *Journal of Electrical Systems and Information Technology*, vol. 3, no. 2, pp. 295–313, 2016, ISSN: 23147172. DOI: 10.1016/j.jesit.2015.12.003.

[162] P. K. Das, "Hybridization of kidney-inspired and sine–cosine algorithm for multi-robot path planning," *Arabian Journal for Science and Engineering*, vol. 45, no. 4, pp. 2883–2900, 2019, ISSN: 2193-567X 2191-4281. DOI: 10.1007/s13369-019-04193-y.

[163] M. R. Panda, S. Dutta, and S. Pradhan, "Hybridizing invasive weed optimization with firefly algorithm for multi-robot motion planning," *Arabian Journal for Science and Engineering*, vol. 43, no. 8, pp. 4029–4039, 2017, ISSN: 2193-567X 2191-4281. DOI: 10.1007/s13369-017-2794-6.

[164] K. Y. Kok and P. Rajendran, "Differential-evolution control parameter optimization for unmanned aerial vehicle path planning," *PLoS One*, vol. 11, no. 3, e0150558, 2016, ISSN: 1932-6203 (Electronic) 1932-6203 (Linking). DOI: 10.1371/journal.pone.0150558. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/26943630.

[165] Y. Zhang, P. Wang, L. Yang, Y. Liu, Y. Lu, and X. Zhu, "Novel swarm intelligence algorithm for global optimization and multi-uavs cooperative path

planning: Anas platyrhynchos optimizer," *Applied Sciences*, vol. 10, no. 14, 2020, ISSN: 2076-3417. DOI: 10.3390/app10144821.

[166] J. Zhang, M. Liu, S. Zhang, R. Zheng, and S. Dong, "Multi-auv adaptive path planning and cooperative sampling for ocean scalar field estimation," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–14, 2022, ISSN: 0018-9456 1557-9662. DOI: 10.1109/tim.2022.3167784.

[167] Z. Chen, J. Alonso-Mora, X. Bai, D. D. Harabor, and P. J. Stuckey, "Integrated task assignment and path planning for capacitated multi-agent pickup and delivery," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5816–5823, 2021, ISSN: 2377-3766 2377-3774. DOI: 10.1109/lra.2021.3074883.

[168] L. Deng, X. Ma, J. Gu, Y. Li, Z. Xu, and Y. Wang, "Artificial immune network-based multi-robot formation path planning with obstacle avoidance," *International Journal of Robotics and Automation*, vol. 31, no. 3, 2016, ISSN: 1925-7090. DOI: 10.2316/Journal.206.2016.3.206-4746.

[169] Y.-T. Kang, W.-J. Chen, D.-Q. Zhu, and J.-H. Wang, "Collision avoidance path planning in multi-ship encounter situations," *Journal of Marine Science and Technology*, vol. 26, no. 4, pp. 1026–1037, 2021, ISSN: 0948-4280 1437-8213. DOI: 10.1007/s00773-021-00796-z.

[170] J.-H. Liang and C.-H. Lee, "Efficient collision-free path-planning of multiple mobile robots system using efficient artificial bee colony algorithm," *Advances in Engineering Software*, vol. 79, pp. 47–56, 2015, ISSN: 09659978. DOI: 10.1016/j.advengsoft.2014.09.006.

[171] R. Al-Jarrah, A. Shahzad, and H. Roth, "Path planning and motion coordination for multi-robots system using probabilistic neuro-fuzzy," *IFAC-PapersOnLine*, vol. 48, no. 10, pp. 46–51, 2015, ISSN: 2405-8963. DOI: 10.1016/j.ifacol.2015.08.106.

[172] A. Pandey and D. R. Parhi, "Optimum path planning of mobile robot in unknown static and dynamic environments using fuzzy-wind driven optimization algorithm," *Defence Technology*, vol. 13, no. 1, pp. 47–58, 2017, ISSN: 22149147. DOI: 10.1016/j.dt.2017.01.001.

[173] R. K, B. R, P. Panchu K, and R. M, "A novel fuzzy and reverse auction-based algorithm for task allocation with optimal path cost in multi-robot systems,"

Concurrency and Computation: Practice and Experience, vol. 34, no. 5, 2021, ISSN: 1532-0626 1532-0634. DOI: 10.1002/cpe.6716.

[174] T. I. Zohdi, "The game of drones: Rapid agent-based machine-learning models for multi-uav path planning," *Computational Mechanics*, vol. 65, no. 1, pp. 217–228, 2019, ISSN: 0178-7675 1432-0924. DOI: 10.1007/s00466-019-01761-9.

[175] D. Zhu, Y. Liu, and B. Sun, "Task assignment and path planning of a multi-auv system based on a glasius bio-inspired self-organising map algorithm," *Journal of Navigation*, vol. 71, no. 2, pp. 482–496, 2017, ISSN: 0373-4633 1469-7785. DOI: 10.1017/s0373463317000728.

[176] X. Cao and D. Zhu, "Multi-auv task assignment and path planning with ocean current based on biological inspired self-organizing map and velocity synthesis algorithm," *Intelligent Automation and Soft Computing*, vol. 23, no. 1, pp. 31–39, 2015, ISSN: 1079-8587 2326-005X. DOI: 10.1080/10798587.2015.1118277.

[177] H. Bae, G. Kim, J. Kim, D. Qian, and S. Lee, "Multi-robot path planning method using reinforcement learning," *Applied Sciences*, vol. 9, no. 15, 2019, ISSN: 2076-3417. DOI: 10.3390/app9153057.

[178] D. Zhu, R. Lv, X. Cao, and S. X. Yang, "Multi-auv hunting algorithm based on bio-inspired neural network in unknown environments," *International Journal of Advanced Robotic Systems*, vol. 12, no. 11, 2015, ISSN: 1729-8814 1729-8814. DOI: 10.5772/61555.

[179] D. Zhu, B. Zhou, and S. X. Yang, "A novel algorithm of multi-auvs task assignment and path planning based on biologically inspired neural network map," *IEEE Transactions on Intelligent Vehicles*, vol. 6, no. 2, pp. 333–342, 2021, ISSN: 2379-8904 2379-8858. DOI: 10.1109/tiv.2020.3029369.

[180] M. Çetinkaya, "Multi-agent path planning using deep reinforcement learning," *arXiv.org*, 2021. DOI: 10.48550/arxiv.2110.01460.

[181] H. Hu, X. Yang, S. Xiao, and F. Wang, "Anti-conflict agv path planning in automated container terminals based on multi-agent reinforcement learning," *International Journal of Production Research*, pp. 1–16, 2021, ISSN: 0020-7543 1366-588X. DOI: 10.1080/00207543.2021.1998695.

[182] B. Li and H. Liang, "Multi-robot path planning method based on prior knowledge and q-learning algorithms," *Journal of physics. Conference series*, vol. 1624, no. 4, p. 42 008, 2020, ISSN: 1742-6588. DOI: 10.1088/1742-6596/1624/4/042008.

[183] H. Chang, Y. Chen, B. Zhang, and D. Doermann, "Multi-uav mobile edge computing and path planning platform based on reinforcement learning," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 6, no. 3, pp. 489–498, 2022, ISSN: 2471-285X. DOI: 10.1109/tetci.2021.3083410.

[184] T. Wang, B. Zhang, M. Zhang, S. Zhang, and D. Guo, "Multi-uav collaborative path planning method based on attention mechanism," *Mathematical Problems in Engineering*, vol. 2021, pp. 1–8, 2021, ISSN: 1563-5147 1024-123X. DOI: 10.1155/2021/6964875.

[185] Y. Yang, L. Juntao, and P. Lingling, "Multi-robot path planning based on a deep reinforcement learning dqn algorithm," *CAAI Transactions on Intelligence Technology*, vol. 5, no. 3, pp. 177–183, 2020, ISSN: 2468-2322 2468-2322. DOI: 10.1049/trit.2020.0024.

[186] S. Wen, Z. Wen, D. Zhang, H. Zhang, and T. Wang, "A multi-robot path-planning algorithm for autonomous navigation using meta-reinforcement learning based on transfer learning," *Applied Soft Computing*, vol. 110, 2021, ISSN: 15684946. DOI: 10.1016/j.asoc.2021.107605.

[187] H. Shiri, H. Seo, J. Park, and M. Bennis, "Attention based communication and control for multi-uav path planning," *arXiv.org*, 2021. DOI: 10.48550/arxiv.2112.12584.

[188] D. Luviano and W. Yu, "Continuous-time path planning for multi-agents with fuzzy reinforcement learning," *Journal of Intelligent and Fuzzy Systems*, vol. 33, no. 1, pp. 491–501, 2017, ISSN: 10641246 18758967. DOI: 10.3233/jifs-161822.

[189] T. Guo and J. Yu, "Sub-1.5 time-optimal multi-robot path planning on grids in polynomial time," *arXiv.org*, 2022.

[190] B. Lopez, J. Munoz, F. Quevedo, C. A. Monje, S. Garrido, and L. E. Moreno, "Path planning and collision risk management strategy for multi-uav systems

in 3d environments," *Sensors (Basel)*, vol. 21, no. 13, 2021, ISSN: 1424-8220 (Electronic) 1424-8220 (Linking). DOI: 10.3390/s21134414. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/34203160.

[191] J. Munoz, B. Lopez, F. Quevedo, C. A. Monje, S. Garrido, and L. E. Moreno, "Multi uav coverage path planning in urban environments," *Sensors (Basel)*, vol. 21, no. 21, 2021, ISSN: 1424-8220 (Electronic) 1424-8220 (Linking). DOI: 10.3390/s21217365. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/34770670.

[192] E. T. S. Alotaibi and H. Al-Rawi, "A complete multi-robot path-planning algorithm," *Autonomous Agents and Multi-Agent Systems*, vol. 32, no. 5, pp. 693–740, 2018, ISSN: 1387-2532 1573-7454. DOI: 10.1007/s10458-018-9391-2.

[193] J. Yu, "Average case constant factor time and distance optimal multi-robot path planning in well-connected environments," *Autonomous Robots*, vol. 44, no. 3-4, pp. 469–483, 2019, ISSN: 0929-5593 1573-7527. DOI: 10.1007/s10514-019-09858-z.

[194] A. C. Kapoutsis, S. A. Chatzichristofis, and E. B. Kosmatopoulos, "Darp: Divide areas algorithm for optimal multi-robot coverage path planning," *Journal of Intelligent and Robotic Systems*, vol. 86, no. 3-4, pp. 663–680, 2017, ISSN: 0921-0296 1573-0409. DOI: 10.1007/s10846-016-0461-x.

[195] J. Olofsson, G. Hendeby, T. R. Lauknes, and T. A. Johansen, "Multi-agent informed path planning using the probability hypothesis density," *Autonomous Robots*, vol. 44, no. 6, pp. 913–925, 2020, ISSN: 0929-5593 1573-7527. DOI: 10.1007/s10514-020-09904-1.

[196] W. Wang and W. B. Goh, "An iterative approach for makespan-minimized multi-agent path planning in discrete space," *Autonomous Agents and Multi-Agent Systems*, vol. 29, no. 3, pp. 335–363, 2014, ISSN: 1387-2532 1573-7454. DOI: 10.1007/s10458-014-9259-z.

[197] Y. Choi, Y. Choi, S. Briceno, and D. N. Mavris, "Energy-constrained multi-uav coverage path planning for an aerial imagery mission using column generation," *Journal of Intelligent and Robotic Systems*, vol. 97, no. 1, pp. 125–139, 2019, ISSN: 0921-0296 1573-0409. DOI: 10.1007/s10846-019-01010-4.

[198] A. Koval, S. Sharif Mansouri, and G. Nikolakopoulos, "Multi-agent collaborative path planning based on staying alive policy," *Robotics*, vol. 9, no. 4, 2020, ISSN: 2218-6581. DOI: 10.3390/robotics9040101.

[199] C. Tatino, N. Pappas, and D. Yuan, "Multi-robot association-path planning in millimeter-wave industrial scenarios," *IEEE Networking Letters*, vol. 2, no. 4, pp. 190–194, 2020, ISSN: 2576-3156. DOI: 10.1109/lnet.2020.3037741.

[200] H. Zhang, J. Luo, J. Long, Y. Huang, and W. Wu, "Multi-robot path planning using petri nets," in *Verification and Evaluation of Computer and Communication Systems* (Lecture Notes in Computer Science), Lecture Notes in Computer Science. 2020, ch. 2, pp. 15–26, ISBN: 978-3-030-65954-7 978-3-030-65955-4. DOI: 10.1007/978-3-030-65955-4_2.

[201] L. Huo, J. Zhu, Z. Li, and M. Ma, "A hybrid differential symbiotic organisms search algorithm for uav path planning," *Sensors (Basel, Switzerland)*, vol. 21, no. 9, p. 3037, 2021, ISSN: 1424-8220. DOI: 10.3390/s21093037.

[202] M. Haciomeroglu, "Congestion-free multi-agent navigation based on velocity space by using cellular automata," *Adaptive Behavior*, vol. 24, no. 1, pp. 18–26, 2015, ISSN: 1059-7123 1741-2633. DOI: 10.1177/1059712315612917.

[203] J. Melin, M. Lauri, A. Kolu, J. Koljonen, and R. Ritala, "Cooperative sensing and path planning in a multi-vehicle environment," *IFAC-PapersOnLine*, vol. 48, no. 9, pp. 198–203, 2015, ISSN: 2405-8963. DOI: 10.1016/j.ifacol.2015.08.083.

[204] K. Jose and D. K. Pratihar, "Task allocation and collision-free path planning of centralized multi-robots system for industrial plant inspection using heuristic methods," *Robotics and Autonomous Systems*, vol. 80, pp. 34–42, 2016, ISSN: 09218890. DOI: 10.1016/j.robot.2016.02.003.

[205] T. Yamauchi, Y. Miyashita, and T. Sugawara, "Path and action planning in non-uniform environments for multi-agent pickup and delivery tasks," in *Multi-Agent Systems* (Lecture Notes in Computer Science), Lecture Notes in Computer Science. 2021, ch. 3, pp. 37–54, ISBN: 978-3-030-82253-8 978-3-030-82254-5. DOI: 10.1007/978-3-030-82254-5_3.

[206] S. D. Han and J. Yu, "Ddm: Fast near-optimal multi-robot path planning using diversified-path and optimal sub-problem solution database heuristics," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1350–1357, 2020, ISSN: 2377-3766 2377-3774. DOI: 10.1109/lra.2020.2967326.

[207] J. Olofsson, C. Veibäck, G. Hendeby, and T. A. Johansen, "Outline of a system for integrated adaptive ice tracking and multi-agent path planning," 2017, p. 13. DOI: 10.1109/RED-UAS.2017.8101636.

[208] G. Best, J. Faigl, and R. Fitch, "Online planning for multi-robot active perception with self-organising maps," *Autonomous Robots*, vol. 42, no. 4, pp. 715–738, 2017, ISSN: 0929-5593 1573-7527. DOI: 10.1007/s10514-017-9691-4.

[209] I. Nielsen, G. Bocewicz, and S. Saha, "Multi-agent path planning problem under a multi-objective optimization framework," in *Distributed Computing and Artificial Intelligence, Special Sessions, 17th International Conference* (Advances in Intelligent Systems and Computing), Advances in Intelligent Systems and Computing. 2021, ch. 1, pp. 5–14, ISBN: 978-3-030-53828-6 978-3-030-53829-3. DOI: 10.1007/978-3-030-53829-3_1.

[210] S. Hayat, E. Yanmaz, C. Bettstetter, and T. X. Brown, "Multi-objective drone path planning for search and rescue with quality-of-service requirements," *Autonomous Robots*, vol. 44, no. 7, pp. 1183–1198, 2020, ISSN: 0929-5593 1573-7527. DOI: 10.1007/s10514-020-09926-9.

[211] M. Kiadi, J. R. Villar, and Q. Tan, "Synthesized a* multi-robot path planning in an indoor smart lab using distributed cloud computing," in *15th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2020)* (Advances in Intelligent Systems and Computing), Advances in Intelligent Systems and Computing. 2021, ch. 56, pp. 580–589, ISBN: 978-3-030-57801-5 978-3-030-57802-2. DOI: 10.1007/978-3-030-57802-2_56.

[212] G. Han, X. Qi, Y. Peng, C. Lin, Y. Zhang, and Q. Lu, "Early warning obstacle avoidance-enabled path planning for multi-auv-based maritime transportation systems," *IEEE Transactions on Intelligent Transportation Sys-*

tems, pp. 1–12, 2022, ISSN: 1524-9050 1558-0016. DOI: 10.1109/tits.2022.3157436.

[213] X. Dai, Q. Fan, and D. Li, "Research status of operational environment partitioning and path planning for multi - robot systems," *Journal of physics. Conference series*, vol. 887, no. 1, p. 12 080, 2017, ISSN: 1742-6588. DOI: 10.1088/1742-6596/887/1/012080.

[214] S. D. Han and J. Yu, "Optimizing space utilization for more effective multi-robot path planning," *arXiv.org*, 2021. DOI: 10.48550/arxiv.2109.04677.

[215] K. Okumura, F. Bonnet, Y. Tamura, and X. Défago, "Offline time-independent multi-agent path planning," *arXiv.org*, 2021. DOI: 10.48550/arxiv.2105.07132.

[216] F. Causa, G. Fasano, and M. Grassi, "Multi-uav path planning for autonomous missions in mixed gnss coverage scenarios," *Sensors (Basel)*, vol. 18, no. 12, 2018, ISSN: 1424-8220 (Electronic) 1424-8220 (Linking). DOI: 10.3390/s18124188. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/30501114.

[217] V. Digani, L. Sabattini, C. Secchi, and C. Fantuzzi, "Ensemble coordination approach in multi-agv systems applied to industrial warehouses," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 3, pp. 922–934, 2015, ISSN: 1545-5955 1558-3783. DOI: 10.1109/tase.2015.2446614.

[218] A. Andreychuk and K. Yakovlev, "Applying mapp algorithm for cooperative path finding in urban environments," in (Lecture Notes in Computer Science), Lecture Notes in Computer Science. Cham: Springer International Publishing, 2017, pp. 1–10, ISBN: 0302-9743. DOI: 10.1007/978-3-319-66471-2_1.

[219] I. Draganjac, D. Miklic, Z. Kovacic, G. Vasiljevic, and S. Bogdan, "Decentralized control of multi-agv systems in autonomous warehousing applications," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 4, pp. 1433–1447, 2016, ISSN: 1545-5955 1558-3783. DOI: 10.1109/tase.2016.2603781.

[220] S. S. Chouhan and R. Niyogi, "Dimpp: A complete distributed algorithm for multi-agent path planning," *Journal of Experimental and Theoretical Artifi-*

cial Intelligence, vol. 29, no. 6, pp. 1129–1148, 2017, ISSN: 0952-813X 1362-3079. DOI: 10.1080/0952813x.2017.1310142.

[221] X. Huang, Q. Cao, and X. Zhu, "Mixed path planning for multi-robots in structured hospital environment," *The Journal of Engineering*, vol. 2019, no. 14, pp. 512–516, 2019, ISSN: 2051-3305 2051-3305. DOI: 10.1049/joe.2018.9409.

[222] A. Ravankar, A. A. Ravankar, Y. Kobayashi, and T. Emaru, "Shp: Smooth hypocycloidal paths with collision-free and decoupled multi-robot path planning," *International Journal of Advanced Robotic Systems*, vol. 13, no. 3, 2016, ISSN: 1729-8814 1729-8814. DOI: 10.5772/63458.

[223] M. Abdelkader, H. Jaleel, and J. S. Shamma, "A distributed framework for real time path planning in practical multi-agent systems," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 10 626–10 631, 2017, ISSN: 2405-8963. DOI: 10.1016/j.ifacol.2017.08.1035.

[224] Q. Li, F. Gama, A. Ribeiro, and A. Prorok, "Graph neural networks for decentralized multi-robot path planning," IEEE, pp. 11 785–11 792, ISBN: 2153-0866. DOI: 10.1109/IROS45743.2020.9341668.

[225] Y. Chen, U. Rosolia, and A. D. Ames, "Decentralized task and path planning for multi-robot systems," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4337–4344, 2021, ISSN: 2377-3766 2377-3774. DOI: 10.1109/lra.2021.3068103.

[226] Q. Li, W. Lin, Z. Liu, and A. Prorok, "Message-aware graph attention networks for large-scale multi-robot path planning," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5533–5540, 2021, ISSN: 2377-3766 2377-3774. DOI: 10.1109/lra.2021.3077863.

[227] H. Bayerlein, M. Theile, M. Caccamo, and D. Gesbert, "Multi-uav path planning for wireless data harvesting with deep reinforcement learning," *IEEE Open Journal of the Communications Society*, vol. 2, pp. 1171–1187, 2021, ISSN: 2644-125X. DOI: 10.1109/ojcoms.2021.3081996.

[228] A. Trudeau and C. M. Clark, "Multi-robot path planning via genetic programming," *arXiv.org*, 2019. DOI: 10.48550/arxiv.1912.09503.

[229] C. Wei, K. V. Hindriks, and C. M. Jonker, "Altruistic coordination for multi-robot cooperative pathfinding," *Applied Intelligence*, vol. 44, no. 2, pp. 269–281, 2015, ISSN: 0924-669X 1573-7497. DOI: 10.1007/s10489-015-0660-3.

[230] T.-M. Liu and D. M. Lyons, "Leveraging area bounds information for autonomous decentralized multi-robot exploration," *Robotics and Autonomous Systems*, vol. 74, pp. 66–78, 2015, ISSN: 09218890. DOI: 10.1016/j.robot.2015.07.002.

[231] F. Matoui, B. Boussaid, and M. N. Abdelkrim, "Distributed path planning of a multi-robot system based on the neighborhood artificial potential field approach," *Simulation*, vol. 95, no. 7, pp. 637–657, 2018, ISSN: 0037-5497 1741-3133. DOI: 10.1177/0037549718785440.

[232] A. A. Neto, D. G. Macharet, and M. F. M. Campos, "Multi-agent rapidly-exploring pseudo-random tree," *Journal of Intelligent and Robotic Systems*, vol. 89, no. 1-2, pp. 69–85, 2017, ISSN: 0921-0296 1573-0409. DOI: 10.1007/s10846-017-0516-7.

[233] Y. Wang and X. Zeng, "Three dimensional unmanned aerial vehicle path planning by a continuous optimization method," in *the 34th Chinese Control Conference*, 2015, pp. 2379–2383.

[234] M. A. Guney and I. A. Raptis, "Dynamic prioritized motion coordination of multi-agv systems," *Robotics and Autonomous Systems*, vol. 139, 2021, ISSN: 09218890. DOI: 10.1016/j.robot.2020.103534.

[235] S. Lin, A. Liu, X. Kong, and J. Wang, "Development of swarm intelligence leader-vicsek-model for multi-agv path planning," in *2021 20th International Symposium on Communications and Information Technologies (ISCIT)*, IEEE, 2021, pp. 49–54, ISBN: 2643-6175. DOI: 10.1109/ISCIT52804.2021.9590578.

[236] M. Pelosi, C. Kopp, and M. Brown, "Range-limited uav trajectory using terrain masking under radar detection risk," *Applied Artificial Intelligence*, vol. 26, no. 8, pp. 743–759, 2012, ISSN: 0883-9514 1087-6545. DOI: 10.1080/08839514.2012.713308.

[237] J. S. Bellingham, M. Tillerson, M. Alighanbari', and J. P. How, "Cooperative path planning for multiple uavs in dynamic and uncertain environments," in

the 41st IEEE Conference on Decision and Control, IEEE, 2002, pp. 2816–2822.

[238] L. Li, J. Wu, Y. Xu, J. Che, and J. Liang, "Energy-controlled optimization algorithm for rechargeable unmanned aerial vehicle network," in *2017 12th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, IEEE, 2017, pp. 1337–1342.

[239] D. Mader, R. Blaskow, P. Westfeld, and C. Weller, "Potential of uav-based laser scanner and multispectral camera data in building inspection," in *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLI-B1, pp. 1135–1142, ISBN: 2194-9034. DOI: 10.5194/isprsarchives-XLI-B1-1135-2016.

[240] M. D. Phung, C. H. Quach, T. H. Dinh, and Q. Ha, "Enhanced discrete particle swarm optimization path planning for uav vision-based surface inspection," *Automation in Construction*, vol. 81, pp. 25–33, 2017, ISSN: 09265805. DOI: 10.1016/j.autcon.2017.04.013.

[241] X. Bai, H. Jiang, J. Cui, K. Lu, P. Chen, and M. Zhang, "Uav path planning based on improved a and dwa algorithms," *International journal of aerospace engineering*, vol. 2021, pp. 1–12, 2021, ISSN: 1687-5966. DOI: 10.1155/2021/4511252.

[242] X. Wu, L. Xu, R. Zhen, and X. Wu, "Biased sampling potentially guided intelligent bidirectional rrt algorithm for uav path planning in 3d environment," *Mathematical problems in engineering*, vol. 2019, pp. 1–12, 2019, ISSN: 1024-123X. DOI: 10.1155/2019/5157403.

[243] Z. Zhang, J. Li, and J. Wang, "Sequential convex programming for nonlinear optimal control problems in uav path planning," *Aerospace science and technology*, vol. 76, pp. 280–290, 2018, ISSN: 1270-9638. DOI: 10.1016/j.ast.2018.01.040.

[244] N. Bolourian and A. Hammad, "Lidar-equipped uav path planning considering potential locations of defects for bridge inspection," *Automation in Construction*, vol. 117, 2020, ISSN: 09265805. DOI: 10.1016/j.autcon.2020.103250.

[245] R. Tian, M. Cao, F. Ma, and P. Ji, "Agricultural uav path planning based on improved a and gravity search mixed algorithm," *Journal of Physics: Conference Series*, vol. 1631, no. 1, p. 12 082, 2020, ISSN: 1742-6588. DOI: 10.1088/1742-6596/1631/1/012082.

[246] G.-c. Luo, J.-q. Yu, Y.-s. Mei, and S.-y. Zhang, "Uav path planning in mixed-obstacle environment via artificial potential field method improved by additional control force," *Asian journal of control*, vol. 17, no. 5, pp. 1600–1610, 2015, ISSN: 1561-8625. DOI: 10.1002/asjc.960.

[247] N. Lin, J. Tang, X. Li, and L. Zhao, "A novel improved bat algorithm in uav path planning," *Computers, materials and continua*, vol. 61, no. 1, pp. 323–344, 2019, ISSN: 1546-2218. DOI: 10.32604/cmc.2019.05674.

[248] J. Wang, G. Wang, X. Hu, H. Luo, and H. Xu, "Cooperative transmission tower inspection with a vehicle and a uav in urban areas," *Energies*, vol. 13, no. 2, 2020, ISSN: 1996-1073. DOI: 10.3390/en13020326.

[249] Y. Chen, J. Yu, Y. Mei, Y. Wang, and X. Su, "Modified central force optimization (mcfo) algorithm for 3d uav path planning," *Neurocomputing (Amsterdam)*, vol. 171, pp. 878–888, 2016, ISSN: 0925-2312. DOI: 10.1016/j.neucom.2015.07.044.

[250] Q. Yang, Z. Yang, T. Zhang, and G. Hu, "A random chemical reaction optimization algorithm based on dual containers strategy for multi-rotor uav path planning in transmission line inspection," *Concurrency and computation*, vol. 31, no. 12, 2019, ISSN: 1532-0626. DOI: 10.1002/cpe.4658.

[251] Z. Fu, J. Yu, G. Xie, Y. Chen, and Y. Mao, "A heuristic evolutionary algorithm of uav path planning," *Wireless communications and mobile computing*, vol. 2018, pp. 1–11, 2018, ISSN: 1530-8669. DOI: 10.1155/2018/2851964.

[252] C. Qu, W. Gai, M. Zhong, and J. Zhang, "A novel reinforcement learning based grey wolf optimizer algorithm for unmanned aerial vehicles (uavs) path planning," *Applied soft computing*, vol. 89, p. 106 099, 2020, ISSN: 1568-4946. DOI: 10.1016/j.asoc.2020.106099.

[253] X. Wang, J.-S. Pan, Q. Yang, L. Kong, V. Snášel, and S.-C. Chu, "Modified mayfly algorithm for uav path planning," *Drones (Basel)*, vol. 6, no. 5, p. 134, 2022, ISSN: 2504-446X. DOI: 10.3390/drones6050134.

[254] S. Jordan, J. Moore, S. Hovet, *et al.*, "State-of-the-art technologies for uav inspections," *IET Radar, Sonar and Navigation*, vol. 12, no. 2, pp. 151–164, 2018, ISSN: 1751-8792 1751-8792. DOI: 10.1049/iet-rsn.2017.0251.

[255] D. Roca, S. Lagüela, L. Díaz-Vilariño, J. Armesto, and P. Arias, "Low-cost aerial unit for outdoor inspection of building façades," *Automation in Construction*, vol. 36, pp. 128–135, 2013, ISSN: 09265805. DOI: 10.1016/j.autcon.2013.08.020.

[256] M. J. Zainorizuan, M. Kaamin, N. A. Idris, *et al.*, "Visual inspection of historical buildings using micro uav," in *MATEC Web of Conferences*, vol. 103, 2017, ISBN: 2261-236X. DOI: 10.1051/matecconf/201710307003.

[257] Y. Ham, K. K. Han, J. J. Lin, and M. Golparvar-Fard, "Visual monitoring of civil infrastructure systems via camera-equipped unmanned aerial vehicles (uavs): A review of related works," *Visualization in Engineering*, vol. 4, no. 1, 2016, ISSN: 2213-7459. DOI: 10.1186/s40327-015-0029-z.

[258] G. Rachele, M. Umberto, M. Giuseppe, P. Francesco, and R. Manuela, "Collecting built environment information using uavs: Time and applicability in building inspection activities," *Sustainability (Basel, Switzerland)*, vol. 12, no. 4731, p. 4731, 2020, ISSN: 2071-1050. DOI: 10.3390/su12114731.

[259] K. Chen, G. Reichard, A. Akanmu, and X. Xu, "Geo-registering uav-captured close-range images to gis-based spatial model for building façade inspections," *Automation in Construction*, vol. 122, 2021, ISSN: 09265805. DOI: 10.1016/j.autcon.2020.103503.

[260] A. Murtiyoso and P. Grussenmeyer, "Documentation of heritage buildings using close-range uav images: Dense matching issues, comparison and case studies," *The Photogrammetric Record*, vol. 32, no. 159, pp. 206–229, 2017, ISSN: 0031868X. DOI: 10.1111/phor.12197.

[261] J. Seo, L. Duque, and J. Wacker, "Drone-enabled bridge inspection methodology and application," *Automation in Construction*, vol. 94, pp. 112–126, 2018, ISSN: 09265805. DOI: 10.1016/j.autcon.2018.06.006.

[262] M. Markova and D. Kravchenko, "3d photogrammetry application for building inspection of cultural heritage objects," *Bulletin of Prydniprovs'ka State*

Academy of Civil Engineering and Architecture, vol. 1, pp. 91–96, 2018, ISSN: 2312-2676. DOI: 10.30838/j.Bpsacea.2312.170118.82.44.

[263] G. Buffi, P. Manciola, A. Gambi, and G. Montanari, "Unmanned aerial vehicle (uav) and building information modelling (bim) technologies in concrete dam management: The case of ridracoli," *In bo*, vol. 9, no. 13, pp. 36–43, 2018.

[264] L. M. Gonzalez de Santos, E. Frias Nores, J. Martinez Sanchez, and H. Gonzalez Jorge, "Indoor path-planning algorithm for uav-based contact inspection," *Sensors (Basel)*, vol. 21, no. 2, 2021, ISSN: 1424-8220 (Electronic) 1424-8220 (Linking). DOI: 10.3390/s21020642. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/33477623.

[265] L. M. González-deSantos, J. Martínez-Sánchez, H. González-Jorge, F. Navarro-Medina, and P. Arias, "Uav payload with collision mitigation for contact inspection," *Automation in Construction*, vol. 115, 2020, ISSN: 09265805. DOI: 10.1016/j.autcon.2020.103200.

[266] A. Murtiyoso, M. Koehl, P. Grussenmeyer, and T. Freville, "Acquisition and processing protocols for uav images: 3d modeling of historical buildings using photogrammetry," *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. IV-2/W2, pp. 163–170, 2017, ISSN: 2194-9050. DOI: 10.5194/isprs-annals-IV-2-W2-163-2017.

[267] N.-H. Pan, C.-H. Tsai, K.-Y. Chen, and J. Sung, "Enhancement of external wall decoration material for the building in safety inspection method," *Journal of Civil Engineering and Management*, vol. 26, no. 3, pp. 216–226, 2020.

[268] G. Vacca, G. Furfaro, and A. Dessì, "The use of the uav images for the building 3d model generation," *Remote Sensing and Spatial Information Sciences*, vol. XLII-4/W8, pp. 217–223, 2018.

[269] S. Biçici and M. Zeybek, "An approach for the automated extraction of road surface distress from a uav-derived point cloud," *Automation in Construction*, vol. 122, 2021, ISSN: 09265805. DOI: 10.1016/j.autcon.2020.103475.

[270] H. Freimuth and M. König, "Planning and executing construction inspections with unmanned aerial vehicles," *Automation in Construction*, vol. 96, pp. 540–553, 2018, ISSN: 09265805. DOI: 10.1016/j.autcon.2018.10.016.

[271] D. Liu, X. Xia, J. Chen, and S. Li, "Integrating building information model and augmented reality for drone-based building inspection," *Journal of Computing in Civil Engineering*, vol. 35, no. 2, 2021, ISSN: 0887-3801 1943-5487. DOI: 10.1061/(asce)cp.1943-5487.0000958.

[272] K. Asadi, A. Kalkunte Suresh, A. Ender, *et al.*, "An integrated ugv-uav system for construction site data collection," *Automation in Construction*, vol. 112, 2020, ISSN: 09265805. DOI: 10.1016/j.autcon.2019.103068.

[273] F. Nex, D. Duarte, A. Steenbeek, and N. Kerle, "Towards real-time building damage mapping with low-cost uav solutions," *Remote Sensing*, vol. 11, no. 3, 2019.

[274] D. Kang and Y.-J. Cha, "Autonomous uavs for structural health monitoring using deep learning and an ultrasonic beacon system with geo-tagging," *Computer-Aided Civil and Infrastructure Engineering*, vol. 33, no. 10, pp. 885–902, 2018, ISSN: 10939687. DOI: 10.1111/mice.12375.

[275] F. Kucuksubasi and A. Sorguc, "Transfer learning-based crack detection by autonomous uavs," in *35th International Symposium on Automation and Robotics in Construction (ISARC 2018)*.

[276] S. Lin, X. Kong, J. Wang, A. Liu, G. Fang, and Y. Han, "Development of a uav path planning approach for multi-building inspection with minimal cost," in *Parallel and Distributed Computing, Applications and Technologies*, Y. Zhang, Y. Xu, and H. Tian, Eds., Springer International Publishing, pp. 82–93, ISBN: 978-3-030-69244-5.

[277] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," IEEE, 1995, pp. 39–43, ISBN: 0780326768. DOI: 10.1109/MHS.1995.494215.

[278] G. Zong Woo, K. Joong Hoon, and G. V. Loganathan, "A new heuristic optimization algorithm: Harmony search," *Simulation (San Diego, Calif.)*, vol. 76, no. 2, pp. 60–68, 2001, ISSN: 0037-5497. DOI: 10.1177/003754970107600201.

[279] D. E. Goldberg and J. H. Holland, "Genetic algorithms and machine learning," *Machine learning*, vol. 3, no. 2, pp. 95–99, 1988, ISSN: 0885-6125. DOI: 10.1023/A:1022602019183.

[280] X.-S. Yang, "Firefly algorithm, lévy flights and global optimization," in London: Springer London, 2009, pp. 209–218, ISBN: 1848829825. DOI: 10.1007/978-1-84882-983-1_15.

[281] R. Kumar, A. Haleem, S. K. Garg, and R. K. Singh, "Automated guided vehicle configurations in flexible manufacturing systems: A comparative study," *International journal of industrial and systems engineering*, vol. 21, no. 2, pp. 207–226, 2015, ISSN: 1748-5037. DOI: 10.1504/IJISE.2015.071510.

[282] S. G. Kumbhar, R. B. Thombare, and A. B. Salunkhe, "Automated guided vehicles for small manufacturing enterprises: A review," *SAE International journal of materials and manufacturing*, vol. 11, no. 3, pp. 253–258, 2018, ISSN: 1946-3979. DOI: 10.4271/05-11-03-0024.

[283] Y. Yang, Y. Quan, and Y. He, "Research on multi-agv management system of autonomous navigation agvs for manufacturing environment," *Journal of physics. Conference series*, vol. 1910, no. 1, p. 12 025, 2021, ISSN: 1742-6588. DOI: 10.1088/1742-6596/1910/1/012025.

[284] H.-W. Cheong and H. Lee, "Requirements of agv (automated guided vehicle) for smes (small and medium-sized enterprises)," *Procedia computer science*, vol. 139, pp. 91–94, 2018, ISSN: 1877-0509. DOI: 10.1016/j.procs.2018.10.222.

[285] R. Almadhoun, T. Taha, L. Seneviratne, and Y. Zweiri, "A survey on multi-robot coverage path planning for model reconstruction and mapping," *SN Applied Sciences*, vol. 1, no. 8, 2019, ISSN: 2523-3963 2523-3971. DOI: 10.1007/s42452-019-0872-y.

[286] R. Kala, "Multi-robot path planning using co-evolutionary genetic programming," *Expert Systems with Applications*, vol. 39, no. 3, pp. 3817–3831, 2012, ISSN: 09574174. DOI: 10.1016/j.eswa.2011.09.090.

[287] A. Ulusoy, S. L. Smith, X. C. Ding, C. Belta, and D. Rus, "Optimality and robustness in multi-robot path planning with temporal logic constraints,"

The International Journal of Robotics Research, vol. 32, no. 8, pp. 889–911, 2013, ISSN: 0278-3649 1741-3176. DOI: 10.1177/0278364913487931.

[288] H. Y. Lin and Y. C. Huang, "Collaborative complete coverage and path planning for multi-robot exploration," *Sensors (Basel)*, vol. 21, no. 11, 2021, ISSN: 1424-8220 (Electronic) 1424-8220 (Linking). DOI: 10.3390/s21113709. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/34073565.

[289] M. Kapanoglu, M. Alikalfa, M. Ozkan, A. Yazıcı, and O. Parlaktuna, "A pattern-based genetic algorithm for multi-robot coverage path planning minimizing completion time," *Journal of Intelligent Manufacturing*, vol. 23, no. 4, pp. 1035–1045, 2010, ISSN: 0956-5515 1572-8145. DOI: 10.1007/s10845-010-0404-5.

[290] P. K. Das and P. K. Jena, "Multi-robot path planning using improved particle swarm optimization algorithm through novel evolutionary operators," *Applied Soft Computing*, vol. 92, 2020, ISSN: 15684946. DOI: 10.1016/j.asoc.2020.106312.

[291] H. Garg, "A hybrid pso-ga algorithm for constrained optimization problems," *Applied Mathematics and Computation*, vol. 274, pp. 292–305, 2016, ISSN: 00963003. DOI: 10.1016/j.amc.2015.11.001.

[292] A. R. Jordehi, "Enhanced leader pso (elpso): A new pso variant for solving global optimisation problems," *Applied Soft Computing*, vol. 26, pp. 401–417, 2015, ISSN: 15684946. DOI: 10.1016/j.asoc.2014.10.026.

[293] P. Liu and J. Liu, "Multi-leader pso (mlpso): A new pso variant for solving global optimization problems," *Applied Soft Computing*, vol. 61, pp. 256–263, 2017, ISSN: 15684946. DOI: 10.1016/j.asoc.2017.08.022.

[294] Z. Meng, J.-S. Pan, and H. Xu, "Quasi-affine transformation evolutionary (quatre) algorithm: A cooperative swarm based algorithm for global optimization," *Knowledge-Based Systems*, vol. 109, pp. 104–121, 2016, ISSN: 09507051. DOI: 10.1016/j.knosys.2016.06.029.

[295] N. Geng, D. W. Gong, and Y. Zhang, "Pso-based robot path planning for multisurvivor rescue in limited survival time," *Mathematical Problems in Engineering*, vol. 2014, pp. 1–10, 2014, ISSN: 1024-123X 1563-5147. DOI: 10.1155/2014/187370.

[296] E. Krell, A. Sheta, A. P. R. Balasubramanian, and S. A. King, "Collision-free autonomous robot navigation in unknown environments utilizing pso for path planning," *Journal of Artificial Intelligence and Soft Computing Research*, vol. 9, no. 4, pp. 267–282, 2019, ISSN: 2449-6499. DOI: 10.2478/jaiscr-2019-0008.

[297] S. Sahu and B. B. Choudhury, "Pso based path planning of a six-axis industrial robot," in *Computational Intelligence in Data Mining* (Advances in Intelligent Systems and Computing), Advances in Intelligent Systems and Computing. 2020, ch. 19, pp. 213–220, ISBN: 978-981-13-8675-6 978-981-13-8676-3. DOI: 10.1007/978-981-13-8676-3_19.

[298] K. Sameshima, K. Nakano, T. Funato, and S. Hosokawa, "Strrt-based path planning with pso-tuned parameters for robocup soccer," *Artificial Life and Robotics*, vol. 19, no. 4, pp. 388–393, 2014, ISSN: 1433-5298 1614-7456. DOI: 10.1007/s10015-014-0177-6.

[299] B. Song, Z. Wang, and L. Zou, "An improved pso algorithm for smooth path planning of mobile robots using continuous high-degree bezier curve," *Applied Soft Computing*, vol. 100, 2021, ISSN: 15684946. DOI: 10.1016/j.asoc.2020.106960.

[300] S. Tian, Y. Li, J. Li, *et al.*, "Robot global path planning using pso algorithm based on the interaction mechanism between leaders and individuals," *Journal of Intelligent and Fuzzy Systems*, vol. 39, no. 4, pp. 4925–4933, 2020, ISSN: 10641246 18758967. DOI: 10.3233/jifs-179978.

[301] S. Kathpal, R. Vohra, J. Singh, and R. S. Sawhney, "Hybrid pso – sa algorithm for achieving partitioning optimization in various network applications," *Procedia Engineering*, vol. 38, pp. 1728–1734, 2012, ISSN: 18777058. DOI: 10.1016/j.proeng.2012.06.210.

[302] K. S. Lee and Z. W. Geem, "A new meta-heuristic algorithm for continuous engineering optimization: Harmony search theory and practice," *Computer methods in applied mechanics and engineering*, vol. 194, no. 36, pp. 3902–3933, 2005, ISSN: 0045-7825. DOI: 10.1016/j.cma.2004.09.007.

[303] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (abc) algorithm," *Journal of*

global optimization, vol. 39, no. 3, pp. 459–471, 2007, ISSN: 0925-5001. DOI: 10.1007/s10898-007-9149-x.

[304] J. Wang, J. Pan, J. Huo, R. Wang, L. Li, and T. Nian, "Research on optimization of multi-agv path based on genetic algorithm considering charge utilization," *Journal of Physics: Conference Series*, vol. 1769, no. 1, p. 12 052, 2021, ISSN: 1742-6588. DOI: 10.1088/1742-6596/1769/1/012052.

[305] U. Cekmez, M. Ozsiginan, and O. K. Sahingoz, "A uav path planning with parallel aco algorithm on cuda platform," in *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, 2014, pp. 347–354, ISBN: 1479923761. DOI: 10.1109/ICUAS.2014.6842273.

[306] Z. Xu, X. Liu, and Q. Chen, "Application of improved astar algorithm in global path planning of unmanned vehicles," IEEE, pp. 2075–2080, ISBN: 2688-0938. DOI: 10.1109/CAC48633.2019.8996720.

[307] Y. Shi, Q. Li, S. Bu, J. Yang, and L. Zhu, "Research on intelligent vehicle path planning based on rapidly-exploring random tree," *Mathematical Problems in Engineering*, vol. 2020, p. 14, 2020, ISSN: 1024123X. DOI: http://dx.doi.org/10.1155/2020/5910503.

[308] T. Huang, D. Huang, N. Qin, and Y. Li, "Path planning and control of a quadrotor uav based on an improved apf using parallel search," *International journal of aerospace engineering*, vol. 2021, pp. 1–14, 2021, ISSN: 1687-5966. DOI: 10.1155/2021/5524841.

[309] J. Xu, Z. Tian, W. He, and Y. Huang, "A fast path planning algorithm fusing prm and p-bi-rrt," IEEE, 2020, pp. 503–508, ISBN: 2166-5656. DOI: 10.1109/PHM-Jinan48558.2020.00098.

[310] Y. Lian, W. Xie, and L. Zhang, "A probabilistic time-constrained based heuristic path planning algorithm in warehouse multi-agv systems," vol. 53, 2020, pp. 2538–2543, ISBN: 2405-8963. DOI: 10.1016/j.ifacol.2020.12.293.

[311] X. Chen, W. Wu, and R. Hu, "A novel multi-agv coordination strategy based on the combination of nodes and grids," *IEEE Robotics and Automation Letters*, pp. 1–1, 2022, ISSN: 2377-3766 2377-3774. DOI: 10.1109/lra.2022.3164754.

[312] P. Xia, A. Xu, and Y. Zhang, "A multi-agv optimal scheduling algorithm based on particle swarm optimization," in *Artificial Intelligence and Security* (Communications in Computer and Information Science), Communications in Computer and Information Science. 2020, ch. 47, pp. 527–538, ISBN: 978-981-15-8082-6 978-981-15-8083-3. DOI: 10.1007/978-981-15-8083-3_47.

[313] Y. Lian and W. Xie, "Improved a multi-agv path planning algorithm based on grid-shaped network," vol. 2019-, Technical Committee on Control Theory, Chinese Association of Automation, 2019, pp. 2088–2092, ISBN: 1934-1768. DOI: 10.23919/ChiCC.2019.8865840.

[314] Q. Yang, Y. Lian, Y. Liu, W. Xie, and Y. Yang, "Multi-agv tracking system based on global vision and apriltag in smart warehouse," *Journal of Intelligent and Robotic Systems*, vol. 104, no. 3, 2022, ISSN: 0921-0296 1573-0409. DOI: 10.1007/s10846-021-01561-5.

[315] L. Xing, Y. Liu, H. Li, C.-C. Wu, W.-C. Lin, and X. Chen, "A novel tabu search algorithm for multi-agv routing problem," *Mathematics*, vol. 8, no. 2, 2020, ISSN: 2227-7390. DOI: 10.3390/math8020279.

[316] J. Chen, X. Zhang, X. Peng, D. Xu, and J. Peng, "Efficient routing for multi-agv based on optimized ant-agent," *Computers and Industrial Engineering*, vol. 167, 2022, ISSN: 03608352. DOI: 10.1016/j.cie.2022.108042.

[317] S. Solichudin, A. Triwiyatno, and M. A. Riyadi, "Conflict-free dynamic route multi-agv using dijkstra floyd-warshall hybrid algorithm with time windows," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 4, 2020, ISSN: 2722-2578 2088-8708. DOI: 10.11591/ijece.v10i4.pp3596-3604.

[318] Y. Zhao, X. Liu, S. Wu, and G. Wang, "Spare zone based hierarchical motion coordination for multi-agv systems," *Simulation Modelling Practice and Theory*, vol. 109, 2021, ISSN: 1569190X. DOI: 10.1016/j.simpat.2021.102294.

[319] K. Zheng, D. Tang, W. Gu, and M. Dai, "Distributed control of multi-agv system based on regional control model," *Production Engineering*, vol. 7, no. 4, pp. 433–441, 2013, ISSN: 0944-6524 1863-7353. DOI: 10.1007/s11740-013-0456-4.

[320] Y. Zhang, F. Wang, F. Fu, and Z. Su, "Multi-agv path planning for indoor factory by using prioritized planning and improved ant algorithm," *Journal of Engineering and Technological Sciences*, vol. 50, no. 4, pp. 534–547, 2018, ISSN: 2338-5502 2337-5779. DOI: 10.5614/j.eng.technol.sci.2018.50.4.6.

[321] X. Cao and M. Zhu, "Research on global optimization method for multiple agv collision avoidance in hybrid path," *Optimal Control Applications and Methods*, vol. 42, no. 4, pp. 1064–1080, 2021, ISSN: 0143-2087 1099-1514. DOI: 10.1002/oca.2716.

[322] L. Tao, S. Zhang, S. Chen, and N. Zheng, *Multi-agv pathfinding for automatic warehouse applications*, Conference Paper, 2021. DOI: 10.1109/cac53003.2021.9728597.

[323] C. Liu, J. Tan, H. Zhao, Y. Li, and X. Bai, "Path planning and intelligent scheduling of multi-agv systems in workshop," Technical Committee on Control Theory, CAA, 2017, pp. 2735–2739, ISBN: 1934-1768. DOI: 10.23919/ChiCC.2017.8027778.

[324] Y. Yang, J. Zhang, Y. Liu, and X. Song, "Multi-agv collision avoidance path optimization for unmanned warehouse based on improved ant colony algorithm," in *Bio-inspired Computing: Theories and Applications* (Communications in Computer and Information Science), Communications in Computer and Information Science. 2020, ch. 41, pp. 527–537, ISBN: 978-981-15-3424-9 978-981-15-3425-6. DOI: 10.1007/978-981-15-3425-6_41.

[325] J. Zajac and W. Małopolski, "Structural on-line control policy for collision and deadlock resolution in multi-agv systems," *Journal of Manufacturing Systems*, vol. 60, pp. 80–92, 2021, ISSN: 02786125. DOI: 10.1016/j.jmsy.2021.05.002.

[326] T. Vicsek, A. Czirók, E. Ben-Jacob, I. Cohen, and O. Shochet, "Novel type of phase transition in a system of self-driven particles," *Physical Review Letters*, vol. 75, no. 6, pp. 1226–1229, 1995, ISSN: 0031-9007 1079-7114. DOI: 10.1103/PhysRevLett.75.1226.

[327] X. Lu, C. Zhang, C. Huang, and B. Qin, "Research on swarm consistent performance of improved vicsek model with neighbors' degree," *Physica A*,

vol. 588, p. 126 567, 2022, ISSN: 0378-4371. DOI: 10.1016/j.physa.2021.126567.

[328] X. Lu, C. Zhang, and B. Qin, "An improved vicsek model of swarm based on remote neighbors strategy," *Physica A*, vol. 587, p. 126 553, 2022, ISSN: 0378-4371. DOI: 10.1016/j.physa.2021.126553.

[329] X. Liu, X. Xiang, Y. Chang, Y. Chao, and D. Tang, "Hierarchical weighting vicsek model for flocking navigation of drones," *Drones*, vol. 5, no. 3, p. 74, 2021. DOI: http://dx.doi.org/10.3390/drones5030074.

[330] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Practical search techniques in path planning for autonomous driving," *AAAI Workshop - Technical Report*, 2008.

[331] V. D. E. Verlag, "Application of hybrid a to an autonomous mobile robot for path planning in unstructured outdoor environments," in VDE Verlag, 2012, pp. 1–1, ISBN: 3800734184.

[332] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011. DOI: 10.1177/0278364911406761. [Online]. Available: https://journals.sagepub.com/doi/abs/10.1177/0278364911406761.

[333] W. Zhang, Y. Peng, W. Wei, and L. Kou, "Real-time conflict-free task assignment and path planning of multi-agv system in intelligent warehousing," vol. 2018-, Technical Committee on Control Theory, Chinese Association of Automation, 2018, pp. 5311–5316, ISBN: 1934-1768. DOI: 10.23919/ChiCC.2018.8483306.

[334] N. Smolic-Rocak, S. Bogdan, Z. Kovacic, and T. Petrovic, "Time windows based dynamic routing in multi-agv systems," *IEEE Transactions on Automation Science and Engineering*, vol. 7, no. 1, pp. 151–155, 2010, ISSN: 1545-5955. DOI: 10.1109/tase.2009.2016350.

[335] L. Sabattini, V. Digani, C. Secchi, and C. Fantuzzi, "Optimized simultaneous conflict-free task assignment and path planning for multi-agv systems," IEEE, 2017, pp. 1083–1088, ISBN: 2153-0866. DOI: 10.1109/IROS.2017.8202278.

[336] Y. Lian and W. Xie, "Improved a multi-agv path planning algorithm based on grid-shaped network," Technical Committee on Control Theory, Chinese Association of Automation, pp. 2088–2092, ISBN: 2161-2927. DOI: 10.23919/ChiCC.2019.8865840.

[337] D. Yu, X. Hu, K. Liang, and J. Ying, "A parallel algorithm for multi-agv systems," *Journal of ambient intelligence and humanized computing*, vol. 13, no. 4, pp. 2309–2323, 2022, ISSN: 1868-5137. DOI: 10.1007/s12652-021-02987-3.

[338] D. Matos, P. Costa, J. Lima, and P. Costa, "Multi agv coordination tolerant to communication failures," *Robotics (Basel)*, vol. 10, no. 2, p. 55, 2021, ISSN: 2218-6581. DOI: 10.3390/robotics10020055.

[339] Z. Han, D. Wang, F. Liu, and Z. Zhao, "Multi-agv path planning with double-path constraints by using an improved genetic algorithm," *PloS one*, vol. 12, no. 7, e0181747–e0181747, 2017, Competing Interests: The authors have declared that no competing interests exist., ISSN: 1932-6203. DOI: 10.1371/journal.pone.0181747.

[340] Y. Zhang and M. Li, "A multi-agv scheduling planning method based on improved ga," *Journal of physics. Conference series*, vol. 1550, no. 2, p. 22 014, 2020, ISSN: 1742-6588. DOI: 10.1088/1742-6596/1550/2/022014.

[341] B. Farooq, J. Bao, H. Raza, Y. Sun, and Q. Ma, "Flow-shop path planning for multi-automated guided vehicles in intelligent textile spinning cyber-physical production systems dynamic environment," *Journal of manufacturing systems*, vol. 59, pp. 98–116, 2021, ISSN: 0278-6125. DOI: 10.1016/j.jmsy.2021.01.009.

[342] S. Sun, J. Li, J. Zhu, *et al.*, "Optimization of waste smoke recovery scheduling strategy based on multi agv," *IOP conference series. Materials Science and Engineering*, vol. 719, no. 1, p. 12 082, 2020, ISSN: 1757-8981. DOI: 10.1088/1757-899X/719/1/012082.

[343] J. Liu, Z. Wang, Q. Xu, and Q. Huang, "Path scheduling for multi-agv system based on two-staged traffic scheduling scheme and genetic algorithm," *Journal of Computational Methods in Sciences and Engineering*, vol. 15, no. 2, pp. 163–169, 2015, ISSN: 14727978 18758983. DOI: 10.3233/jcm-150530.

[344] T. Mu, J. Zhu, X. Li, and J. Li, "Research on two-stage path planning algorithms for storage multi-agv," in *Bio-inspired Computing: Theories and Applications* (Communications in Computer and Information Science), Communications in Computer and Information Science. 2020, ch. 35, pp. 418–430, ISBN: 978-981-15-3414-0 978-981-15-3415-7. DOI: 10.1007/978-981-15-3415-7_35.

[345] X. Liao, Y. Wang, Y. Xuan, and D. Wu, "Agv path planning model based on reinforcement learning," IEEE, 2020, pp. 6722–6726, ISBN: 2688-0938. DOI: 10.1109/CAC51589.2020.9326742.

[346] H. Hu, X. Yang, S. Xiao, and F. Wang, "Anti-conflict agv path planning in automated container terminals based on multi-agent reinforcement learning," *International journal of production research*, vol. ahead-of-print, no. ahead-of-print, pp. 1–16, 2021, ISSN: 0020-7543. DOI: 10.1080/00207543.2021.1998695.

[347] S. Lin, A. Liu, J. Wang, and X. Kong, "A review of path-planning approaches for multiple mobile robots," *Machines*, vol. 10, no. 9, p. 773, 2022, ISSN: 2075-1702. [Online]. Available: https://www.mdpi.com/2075-1702/10/9/773.

[348] M. Steinbrunn, G. Moerkotte, and A. Kemper, "Heuristic and randomized optimization for the join ordering problem," *The VLDB journal*, vol. 6, no. 3, pp. 191–208, 1997, ISSN: 1066-8888. DOI: 10.1007/s007780050040.

[349] M. N. Zafar and J. C. Mohanta, "Methodology for path planning and optimization of mobile robots: A review," *Procedia computer science*, vol. 133, pp. 141–152, 2018, ISSN: 1877-0509. DOI: 10.1016/j.procs.2018.07.018.

[350] Z. Xie, Q. Zhang, Z. Jiang, and H. Liu, "Robot learning from demonstration for path planning: A review," *Science China. Technological sciences*, vol. 63, no. 8, pp. 1325–1334, 2020, ISSN: 1674-7321. DOI: 10.1007/s11431-020-1648-4.