An Algorithmic Method for Warehouse Order Picking with Congestion and One-Way Aisle Constraints.

by

Shad Nur Mim Bidhu
20101574
Abdullah Arif
22101007
Maharin Sharif Abony
22299442
Nazib Uddin Ahrar
22299440
Syed Hasibur Rahman Shafin
22201778

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science and Engineering

Department of Computer Science and Engineering
Brac University
January 2026.

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Shad Nur Mim Bidhu
20101574

Abdullah Arif
22101007

Maharin Sharif Abony
22299442

Nazib Uddin Ahrar
22299440

Syed Hasibur Rahman Shafin
22201778

Approval

An Algorithmic Method for Warehouse Order Picking with Congestion and One-Way Aisle Constraints.

1. Shad Nur Mim Bidhu (20101574)
2. Abdullah Arif (22101007)
3. Maharin Sharif Abony (22299442)
4. Nazib Uddin Ahrar (22299440)
5. Syed Hasibur Rahman Shafin (22201778)

of Spring, 2025 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science in Fall, 2025.

Examining Committee:

Supervisor:
(Member)

Dr. Amitabha Chakrabarty

Professor
Department of Computer Science and Engineering
BRAC University

Thesis Coordinator:
(Member)

Dr. Md. Golam Rabiul Alam

Professor
Department of Computer Science and Engineering
BRAC University

Head of Department:
(Chair)

Dr. Sadia Hamid Kazi

Associate Professor, Chairperson
Department of Computer Science and Engineering
BRAC University

Abstract

Order picking is a fundamental management process in warehouse management systems and it has a great impact on the cost of operations, efficiency and quality of the services. Although the use of traditional routing and shortest path algorithms is a frequent solution to the problem of order-picking, it frequently ignores real-world constraints like the congestion and one-way aisle patterns, which can significantly impact the viability and effectiveness of routes in the real-world warehouse setting. To overcome these drawbacks, this thesis is concerned with creating and analyzing congestion-aware routing plans in a realistic warehouse environment with order-picking. A simulation based benchmarking model is suggested to compare systematically classical, heuristic, hybrid, and metaheuristic routing algorithms based on unified warehouse layouts with one-way aisle constraints and a travel cost that is a congestion cost. The structure makes it possible to model route execution, traffic congestion, and collision impacts that allow fair and consistent comparison between algorithm methods. Performance variables are calculated on standardized measures, including quality of solutions, planning efficiency, collision impact, resource use and scalability to problems of different sizes. The comparison analysis shows that even though exact routing algorithms only work well in small scale problems, heuristic routing methods are quicker but more vulnerable to congestion impacts. Hybrid routing methods are more robust since they trade off between computational efficiency and quality of solutions in congestion aware situations. For instance, Hybrid NN2opt achieved a 74.1% optimization rate with a mean planning time of only 7.68 ms, nearly matching the optimal Held-Karp baseline while using 89% less memory. In multi-robot simulations, it reduced collisions by up to 33.33% in congested narrow aisles and up to 100% in wide aisles and maintained more stable performance as robot density increased from 3–5 to 10–15 robots. These results underscore the significance of the inclusion of realistic operation constraints in the modeling of warehouse routing and offer a feasible base on how to enhance the efficiency of order-picking in a big and dynamic warehouse setup.

Keywords: Warehouse Order Picking, Path Optimization, Congestion, One-Way Aisles, Hybrid Algorithms, Warehouse Simulation, Collision

Dedication

We would like to dedicate this thesis to our parents for always having our backs.

Acknowledgement

By the grace of the Almighty and with the constant support of our parents, friends, and families, we were able to complete the thesis despite some challenging moments.

We are truly grateful to our supervisor, Dr. Amitabha Chakrabarty, for this guidance and support. This would not have been possible without his invaluable insights, academic assistance, time, and effort.

We are also thankful to Mr Azwad Aziz for helping us out with his precious feedback, efforts, and time.

Last but not least, we would like to thank our peers, seniors, and juniors who were there when needed, and most importantly, thanks to all the open-source resources for existing.

Table of Contents

Declaration i

Approval ii

Ethics Statement iv

Abstract iv

Dedication v

Acknowledgment vi

Table of Contents vii

List of Figures ix

List of Tables x

Nomenclature x

1 Introduction 1
- 1.1 Background . . . 1
- 1.2 Rational of the Study or Motivation . . . 2
- 1.3 Problem Statement . . . 2
- 1.4 Objective . . . 2
- 1.5 Methodology in Brief . . . 3
  - 1.5.1 Simulation environment development . . . 3
  - 1.5.2 Algorithm Framework Implementation . . . 3
  - 1.5.3 Constraint Integration and Dynamic Routing . . . 3
  - 1.5.4 Performance Evaluation and Result Analysis . . . 4
- 1.6 Scopes and Challenges . . . 4

2 Literature Review 5
- 2.1 Preliminaries . . . 5
- 2.2 Review of Existing Research . . . 6
- 2.3 Summary of Key Findings . . . 10
  - 2.3.1 Research Gap . . . 10

3 Requirements, Impacts and Constraints **11**
3.1 Final Specifications and Requirements . . . 11
3.1.1 Functional Requirements . . . 11
3.2 Societal Impact . . . 12
3.2.1 Positive Impacts . . . 12
3.2.2 Problems and Morality . . . 12
3.3 Environmental Impact . . . 13
3.3.1 Positive Impacts . . . 13
3.3.2 Negative Impacts . . . 13
3.4 Ethical Issues . . . 13
3.5 Project Management Plan . . . 14
3.6 Risk Management . . . 15
3.7 Economic Analysis . . . 15

4 Proposed Methodology **16**
4.1 Design Process or Methodology Overview . . . 16
4.1.1 Design Constraints . . . 17
4.1.2 Proposed Strategy . . . 17
4.2 Preliminary Design or Design (Model) Specification . . . 20
4.2.1 Algorithmic Framework . . . 20
4.2.2 Simulation Environment Framework . . . 24
4.3 Implementation of Selected Design . . . 26
4.3.1 Creation of Warehouse Simulation Environment . . . 26
4.3.2 Implementation of Selected Algorithms in the Simulation Environment . . . 27
4.3.3 Implementation-Based Comparative Performance Analysis . . . 28

5 Result Analysis **29**
5.1 Performance Evaluation . . . 29
5.2 Analysis of Design Solutions . . . 30
5.3 Final Design Adjustments . . . 34
5.4 Statistical Analysis . . . 36
5.4.1 Simple Scenario: Single Bot and Docking Station . . . 36
5.4.2 Complex Scenario: Multiple Bot and Docking Station . . . 37
5.5 Comparisons and Relationships . . . 38
5.5.1 Simple Scenario: Single Bot and Docking Station . . . 38
5.5.2 Complex Scenario: Multiple Bot and Docking Station . . . 38
5.6 Discussions . . . 40
5.6.1 Simple Scenario: Single Bot and Docking Station . . . 40
5.6.2 Complex Scenario: Multiple Bot and Docking Station . . . 40

6 Conclusion **41**
6.1 Summary of Findings . . . 41
6.2 Contributions to the Field . . . 42
6.3 Recommendations for Future Work . . . 42

Bibliography **46**

List of Figures

3.1 Project Timeline 15

4.1 Methodology of the Study 16

5.1 Optimization Rate Comparison 31
5.2 Algorithm Complexity vs Performance 32
5.3 Warehouse Optimization Algorithms 33
5.4 Comparison of Average Planning Time 34
5.5 Comparison of Average Tour Length 35
5.6 Comparison of Average Wait Time 35
5.7 Congestion Level vs Optimality Rate 38
5.8 Comparison of Average Collision Count 39

List of Tables

2.1 Summary of Existing Literature 9

4.1 Implementation-Based Performance Table 28

5.1 Descriptions of Evaluation Metrics 29
5.2 Algorithm Performance Comparison 30
5.3 Comparison of Different Warehouse Routing Algorithms 36
5.4 Performance comparison of best-performing algorithms for multiple bots 37

Chapter 1

Introduction

1.1 Background

Order picking in the warehouse is one of the most demanding and expensive activities in the contemporary supply chain management, as it consumes about 55% of the total cost of operation in the warehouse [4], [7]. The substantial growth of e-commerce and the increasing demands among consumers to receive their order within a day or two have led to the necessity of effective order picking procedures to retain their competitiveness and profitability of operations [5].

The standard operations of a warehouse use manual, semi-autonomous, and fully automated methods that involve human workers, or autonomous mobile robots (AMRs), navigating the aisles of the warehouse to collect items. These conventional picking systems are however inefficient in nature. They are long travel times, worker movements, and congestion in narrow aisles, all of which contribute to the longer time to complete the task and higher operational costs [7].

To halt these inefficiencies, shortest path methods such as Dijkstra, A^*, and Floyd-Warshall, have been applied to optimize the warehouse routing and ensure minimal travel distances are used in order to minimize the time required to make a complete order [5][7]. Such algorithm solutions enable the systems to calculate the best routes between two or more picking points, i.e., a worker or a robot follow the best routes that can be constructed. Routing optimization is very difficult in the real world warehouse environment due to the constraints of the operations. The aisles are limited by one-way restrictions to prevent collisions and ensure the organized traffic flow, which requires the use of algorithms to assist with directional restrictions. The congestion management is also of crucial concern when the pickers or the AMRs are operating simultaneously as the traffic jam and the formation of the congestion may severely reduce the efficiency of the entire system.

These existential problems are beginning to be addressed in recent studies. Bhati et al. [26] developed a framework with the support of simulation that could be measured through congestion-avoidance and Hvezda et al. [20] developed a context-optimal route planning system that operated dynamically in response to the congestion and aisle constraint in the real-life warehouse.

1.2 Rational of the Study or Motivation

Several critical issues are experienced in warehouse order picking systems, which serve as the source of this analysis. The main disadvantage of current applications is that they are often uninformed of the direction the aisle is traversing. This leads to the formation of non-optimal travel paths that might be inconvenient in regards to safety measures recommended to operate in the warehouse [20]. Furthermore, most systems are not designed to consider the dynamic congestion patterns, and this leads to the chronic congestion in areas of high traffic, leading to the poor performance [26][34].

Centralization of routing architectures introduces single point of failure and scalability constraints when operating large-scale warehouses [5]. Moreover, because there is no exhaustive comparative analysis of all the families of algorithms, it is not easy to have a warehouse operator to choose the right routing strategy that suits his or her operational needs and layout designs.

This work considers such limitations and proposes a decentralized stateful routing system, which builds efficient picking routes free of collisions on the fly. We promote both comparative analysis of solution methods and flexibility to real-world warehouse constraints by using several algorithm classes in a single simulation model.

1.3 Problem Statement

Traditional warehouse order picking systems are not spatially efficient since they do not offer guidance on aisle directional flow or provide real-time congestion updates, leading to suboptimal routing, wasted time for pickers, and a general decline in operational productivity.

1.4 Objective

The aim of the present research is to design, implement, and evaluate a congestion aware warehouse order-picking framework that realistically models operational constraints such as one-way aisle restrictions, dynamic congestion, and multi-robot interactions. The study aims to move beyond traditional distance-only routing approaches by incorporating collision-aware decision-making within a unied simulation and benchmarking environment. The targeted goals are:

- To provide a high-delity, grid-based warehouse simulation system that can mimic realistic layouts, one-way aisle restrictions, multi-robot mobility, and congestion with discrete-event based simulation.
- To ensure fair and repeatable comparison, several routing algorithms will be implemented and benchmarked under identical warehouse conditions.
- To propose and evaluate a collision-aware routing strategy which explicitly avoids occupied grid cells during route construction and optimization, and reduces collisions without requiring computationally expensive global replanning.

- To evaluate the influence of congestion and robot concentration on routing efficiency, by methodically altering the quantity of robots and changing warehouse configurations, and assessing critical operational metrics like tour length, planning duration, congestion delay, collision frequency, memory consumption, and success rate.
- To study the trade-offs among optimality, computing efciency, and congestion resilience, and to discover routing strategies that are reasonable for implementation in extensive, dynamic warehouse settings.

Therefore, through these objectives, the study seeks to create a standardized, congestion-aware framework and demonstrate that lightweight hybrid heuristics can achieve near-optimal performance while maintaining robustness and scalability in realistic warehouse order-picking scenarios.

1.5 Methodology in Brief

1.5.1 Simulation environment development

The experiment begins with a complete set of a simulation warehouse environment that simulates various. Layout plans included narrow and wide aisle layout plans. Major working elements of this. There are storage racks, picking stations, directional aisle restrictions which constitute a part of the environment. The simulator is was also meant to furnish realistic conditions of a warehouse to be tested and checked under different operational circumstances. conditions.

1.5.2 Algorithm Framework Implementation

It adopts a multi-algorithm approach that it incorporates classical pathfinding algorithms. hybrid algorithms and metaheuristic algorithms. The order picking is adapted to its algorithms. the problem of optimization taking into account the real-life conditions of a warehouse. The framework provides a systematic study of the different methods of algorithm when everything is perfectly identical to. compare their performances.

1.5.3 Constraint Integration and Dynamic Routing

The routing algorithms which are employed necessarily include functional warehouse constraints like aisle. postings, discretionary action, and working conditions. A discrete-event simulation sub-component defines live time implementation of picking paths with dynamic constituents that encompass. traffic interference, source rivalry and stochastic disturbance. This plan enables the system to be adaptive. to new circumstances in the warehouse.

1.5.4 Performance Evaluation and Result Analysis

The statistical significance is reviewed through a strict experimental design, in which there are a number of. Warehouse configurations and problem sizes and random seeds are adopted. Extensive performance they include such indicators as the distance travelled, time taken, wait times and frequency of congestion. all measured out systematically. Such strategies are contrasted with the conventional free routing policies to. lift the performance up and business feasibility.

1.6 Scopes and Challenges

The paper is dedicated to the virtual Farm of illustration of warehouse environments with Python frameworks simulation. developed in house. It is although not feasible yet proposed system is exhibited to be viable through its expansive simulations. not yet in scale to deploy at large scale with large AV troopings. The paper focuses on simulated environment with realistic conditions and parameters of algorithmic performance and constraint management. differentiating characteristics of the real world warehouse functionality.

The key technical challenges that are likely to be experienced are the maintenance of ability to re-computate the paths. at the moment of congestions of all the vehicles in the real time to retain the quality of the solutions, providing scalability of algorithms to ensure the quality of small and complex layouts of warehouses, trading-off the demands of computational efficiency and quality of the solutions in the different categories of algorithms, the conception of a powerful constraint integration that entails implicitly.

In order to address these challenges, the study takes a modules based architectural lifestyle that incorporates existence of programs, parameter optimizations and an evaluation system. it defines the quality of solutions and performance of computations on a broad range of. operational environments.

Chapter 2

Literature Review

2.1 Preliminaries

If the warehouse management system is our concern, order picking is a really significant part of it. Factors like cost,time and quality of service are very much dependent on this single feature. Problems like one-way aisles and congestion needs to be addressed using optimization of routes.The reality of dynamic warehouses isn't helped much through classical path planning algorithms such as Dijkstra or A*,although they've been applied successfully to find optimal paths. Nevertheless, these algorithms do not always reflect the realities of dynamic congestion, changing picker capacity, and directional constraints that influence traffic flows [36]. Hybrid algorithm frameworks that merge classical approach,metaheuristics and advanced control techniques have been applied by researchers, such as combination of A*,IACO,Dynamic window approach (which actually performed better than solo algorithm approaches in case of solving package picking energy consumption and path performance) [36]. Likewise, multiple ACO models used parallel ,hustled the convergence and enhanced the coordination of AGV's in terms of traffic-prone settings [22]. As AI and RL have become more powerful in case of solving these kind of problems, DRL(Deep Reinforcement Learning) has been used to solve the problem of congestion and flexibility of path-planning in large scale.Simultaneously, order assignment and storage location models have been in a position to consider the practical variables of a warehouse e.g. item weight, picker capacity, and workload distribution which provides a more realistic image of the real life operations. Combining these elements of operation allows creating more powerful and balanced strategies that are involved with the warehouse .Technological integration has gradually become an important element in the navigation of warehouses in parallel with the optimization of the warehouse using algorithms. As an example, the SINS-AR system employs augmented reality to facilitate successful navigation in the GPS-denied environment to support the process of indoor navigation, which contributes to the improvement of the real-time spatial perception and route direction [34]. These advances in combination form a movement in the current trend toward the utilization of adaptive, intelligent, and constraint-sensitive warehouse optimization models.

2.2 Review of Existing Research

To maximize the routing process and minimize the congestion, new studies on order picking in warehouses have put more emphasis on adaptive algorithms, intelligent systems, and simulation models. In one instance, the automated e-fulfillment system connected to the Internet of Things has been optimized with the help of Particle Swarm Optimization (PSO) to enhance packing and by extension reduce the energy use and trip time [17]. Also, context-based route planning has also been applied in an effort of enhancing throughput and efficiency within automated systems by effecting dynamic adjustment at the changing conditions within the warehouse environments [20]. The order picking systems can be further developed with the help of deep reinforcement learning (DRL) model optimization of congestion strategies and adaptation to the real-time changes in warehouse environment [30]. Such technologies as SINS AR have enhanced indoor navigation and have increased the order picking accuracy and efficiency, and combination models that include augmented reality (AR) have performed successfully, as well [34]. Also, the effects of congestion on automatic and manual order picking have been studied to assist in optimizing the method to reduce the number of bottlenecks and design routes, which are cost-effective [28]. The process remains to be essential with the help of simulation models.

Genetic Algorithms (GAs) have been found to be extensively used to solve the combinatorial challenges of warehouse order picking and routing. GAs are also better than other heuristics in a number of tests and have been demonstrated to be especially useful in order batching, reducing the total journey distance by efficient grouping of orders [9]. Also, the order picking routes have been optimized using GAs and this has been applied especially in warehouses where the distribution of items is systematic and results have shown a peak efficiency of pickers and the travel time is also reduced [40]. Multi-agent systems have also been deployed in Automated Guided Vehicle (AGV) systems in order to enhance the flow of materials and efficiency in the systems. More to the point, larger-scale surveys on the methods to optimize the order picking process have found GAs as an important optimization tool because of their flexibility and ability to adapt to changing warehouse conditions [4], [12]. All of these studies confirm the notion that GAs can be implemented on most optimization tasks in a warehouse, such as order batching, picking, and AGV path planning [14].

Ant Colony Optimization (ACO) is a metaheuristic algorithm that emulates foraging behavior of ants and is a nature-based algorithm. Artificial ants explore a problem region and mark the characteristics of pheromones on other ants signifying a favorable response to the problem. The algorithm has worked well in solving order picking and routing problems in warehouses, particularly in congested and complex constraint environments. The initial application involved ACO being used to determine optimal paths that an order picker would follow by optimizing straightforward routing using a pheromones strengthening simulation [5]. The ACO has been modified to not only take into account the journey time, but picker interactions as well to minimize the number of conflicts and optimize the efficiency of the system once congestion in the narrow-aisle system is a major concern [15]. Also, the real-time information has been incorporated with the ACO-related methods to dynamically

modify routes in accordance with the changing orders and warehouse constraints [19]. Recent studies were also interested in the hybrid algorithm; a combination of genetic and ACO algorithms to enhance the performance and convergence in large scale operations [35]. Also, ACO can now work with more complex and autonomous warehouse systems due to its integration with machine learning and multi-agent systems [27]. The following developments demonstrate that ACO can be successful in addressing the order choosing and routing problem by balancing between exploration of new possibilities and exploitation of old, effective ones [40]. Also, within the framework of large-scale warehouse facilities, the hybrid application of ACO and dynamic scheduling and optimization approaches has been talked about. The flexibility of ACO to increasingly advanced autonomous systems has also been reported in research that has used it in centralized and decentralized warehouse environments [38], [40]. These new developments merely point to how useful ACO is in streamlining the workings of a warehouse in many different manners [2].

The problem of warehouse picker routing is already widely researched with optimization methods, and 2-Opt and Nearest Neighbor (NN) methods are necessary to enhance the performance. NN produces an initial solution by choosing the closest node without being visited and optimizes it by (2) replacing two edges to minimize the trip distance. As helpful as they were, the initial routing schemes, like the Dijkstra algorithm and dynamic programming, did not work well in large-scale warehouse applications [18]. In order to overcome these constraints, NN + 2-Opt has been used. In order to minimize trip time and congestion in multi-aisle warehouses, such as, tabu search which is intertwined with both NN and 2-Opt has applied joint order batching and picker routing [5]. Further, NN and 2-Opt have been applied to optimize the total travel distance in picking operations within the warehouse at greater levels of throughput in the warehouse [5]. All in all, these heuristics will continue to play a central role in the solution of the warehouse routing problems, providing a balance between the computational efficiency and the quality of the solution [23].

The Held-Karp algorithm is a dynamic programming method used to solve the Traveling Salesman Problem (TSP) to ensure an optimal solution is found by considering all possible tours using a set of sites with the total traveling distance being minimized without violating operational restrictions. Its time space complexity of $O(n^2 2^n)$ means that it is practical only in small scale problems (smaller than 20 nodes) and not in large scale problems (large warehouse routing). Nonetheless, it has been often taken as a reference point of optimality against which to assess heuristic and metaheuristic algorithms of warehouse order picking. As an example, single-block warehouse routing has been applied in determining the most optimal travelling paths in ideal situations [33], and hybrid heuristics like Hybrid NN2opt have been simulated to demonstrate that they can find up to 97% optimality at a much lower computing cost. Held-Karp algorithm which offers a tight-knit-fit criterion to ensure good solutions in constrained environments has been demonstrated to be an obligatory path optimization instrument [1], [6], [8].

An Ant Lion Optimization (ALO) is a metaheuristic algorithm that is inspired by the hunting behaviour of antlions in the sand traps. Antlions are the possible solu-

tions which direct the ants to the most suitable places and this behavior imitates the random wandering of the ants in some predetermined area of search. The algorithm is particularly outstanding in establishing a balance between exploration and exploitation because it operates in the adaptive boundary shrinkage as well as the elite-inspired movement. It can be used in multi-modal optimization tasks, including the warehouse routing that can take a dynamic constraint [13]. To maximize the utilization of space in a one-way aisle and multi-robot environment with a lot of people, ALO was applied in warehouse picking of orders. Recent studies have combined ALO and local search methods to enhance the quality of solutions and convergence rate in jammed robotic warehouses. Though ALO still demands more computer power than lightweight heuristics like NN2opt, such integration has allowed ALO to achieve meaningful performance increases on traditional warehouse objectives (including tour length and planning time), even though it is a potentially promising strategy [24].

Table 2.1 contains details of some of the highly relevant literature reviewed in this study.

Serial No.	References	Algorithms	Performances	Comments
1	[33]	Modified Dynamic Programming (Held–Karp variant)	Optimal solution for small instances ($\leq$15 nodes)	Not scalable; impractical for large or congested warehouses due to exponential time and memory complexity.
2	[22]	Parallel Ant Colony Optimization (ACO)	Faster convergence than classical ACO (no unified numeric benchmark reported)	Focuses on convergence speed; congestion interaction and picker interference not explicitly modeled.
3	[36]	Hybrid A* + Improved ACO + Dynamic Window Approach	Reduced energy consumption and travel distance (relative improvement reported)	High computational overhead; real-time scalability under dense congestion not validated.
4	[40]	Neural Adaptive Heuristic ACO (NAHACO)	Improved path feasibility in 3D warehouses (no standardized optimality ratio)	Training complexity is high; real-time deployment feasibility not demonstrated.
5	[10]	ACO with congestion consideration	Reduced travel distance under two-picker scenarios	Limited to small-scale settings; lacks multi-robot scalability analysis.
6	[18]	Dijkstra, Dynamic Programming	Shortest path correctness verified	Ignores dynamic congestion and one-way aisle constraints.
7	[5]	Tabu Search + NN + 2-opt	Significant travel distance reduction (comparative results only)	Computationally expensive; lacks real-time congestion adaptation.
8	[30]	Deep Reinforcement Learning (DRL)	Improved congestion control (qualitative gains reported)	Extremely high computational cost; warehouse-specific routing constraints not fully integrated.
9	[34]	SINS-AR Navigation System	Improved navigation accuracy (no routing optimality metric)	Focuses on navigation assistance, not routing optimization or congestion handling.
10	[39]	DRL-based Dynamic Order Picking	Improved flexibility in dynamic environments	Limited empirical validation; lacks benchmarking against classical and hybrid heuristics.

Table 2.1: Summary of Existing Literature

2.3 Summary of Key Findings

Existing literature review shows that warehouse order picking optimization has been developed as shortest-path algorithms to heuristic, metaheuristic, hybrid and learning-based methods to deal with growing operational complexity. Some specific algorithms like Held-Karp give optimal solutions that are computationally infeasible on small scale problems. NN and 2-opt are heuristic algorithms that are not as robust in the face of congestion, and also have short planning times. Metaheuristic algorithms such as GA, ACO and ALO are better at offering flexibility and adaptability of solutions, but are prone to significant computation time and unpredictable performance in dense multi-picker scenarios. The new hybrid strategies and AI-based strategies show better flexibility and congestion consciousness, but they are not standardized in benchmark strategies and scalability validation in real-time. In general, the literature indicates that there exists a longstanding trade-off between the quality of solutions, the efficiency of their computations, and the congestion resilience, and therefore unified, congestion-conscious, and scalable algorithmic frameworks to realistic warehouse settings are necessary.

2.3.1 Research Gap

Despite being able to handle a lot of scenarios, difficulties, obstacles of a dynamic warehouse, these researches fail to address some areas which can be addressed as potential research gaps.

- **Scalability:** Available algorithms perform poorly when implemented on large-scale, dynamic, and multi-level warehouse settings, especially with increased robot density.
- **Simplified Congestion Modeling:** The majority of the studies do not model the effect of dynamic congestion in a realistic way, and it disregards the time-varying traffic density, collision propagation, and picker interference effects.
- **Computational Overhead:** High-level DRA and hybrid meta heuristic methods require a significant amount of compute resources, and thus real-time industry application is not feasible.
- **Lack of Standardized Benchmarking:** No single benchmarking model exists that compares the various family of algorithm in the same warehouse layouts, constraints and congestion.
- **Weak Empirical Validation:** Most of the algorithms suggested are still only confirmed as working in simulated or small-scale settings with very little evidence of applicability in a real world or industry scale.

Chapter 3

Requirements, Impacts and Constraints

3.1 Final Specifications and Requirements

3.1.1 Functional Requirements

Efficient Order Selection for Congestion Management: In addition to reducing total travel and picking time, the technique should take aisle congestion into consideration. Algorithms that look at multi-robot warehouse settings, waiting costs, blocking events, and route rerouting will be used to prevent bottlenecks **desantis2018adapted**, [10], [17].

One-way aisle restriction: Actual warehouse aisles can accommodate unidirectional traffic, in contrast to unconstrained models that try to avoid collisions or accidents. regulations. These limitations should be taken into account in the system for all routing calculations [36], [39].

Combining Algorithms: The project contrasts hybrid algorithms, which combine shortest-path searching, predictive congestion, and dynamic adaptation, with the classical algorithms. The traditional algorithms that were compared in the study are Held-Karp, Nearest Neighbor + 2-opt, Genetic Algorithm, and a hybrid algorithm by the researchers Haldge, termed HybridNN2opt with 3, 5, 10 and 15 robots. It aimed to compare the performance of each algorithm in terms of the number of collisions, efficiency of the paths, and scaling over an increase in the number of robots. This enables the trade-off between computing efficiency and optimality [30], [33].

Active Congestion-Sensitive Rerouting: In multi-picker settings, congestion is dynamic. The main need of this study was to determine how these algorithms can be scaled when there are various robot densities with respect to collision avoidance and optimal pathfinding in a realistic warehouse situation. Particularly, the HybridNN2opt algorithm was emphasized due to its superior adaption to the congestion conditions making collision incidences decreasing with the increase in robot density. Similar to adaptive routing in traffic networks, the system should be able to provide real-time rerouting as circumstances change [25], [29].

Framework for Simulation and Benchmarking:A warehouse simulator must generate a variety of layouts (small, medium, and big) with realistic pick lists, traffic patterns, and limits. The algorithms were also tested in different configurations of the warehouses such as narrow aisles, wide aisles and patterns of crosses with and without obstacles.

Performance Metrics: Assessment is centered on: The entire order journey time. By blocking, congestion-related delays are avoided. Multi-picker throughput/makespan. Algorithm computation cost and run time [10], [18].

Scalability: To be successfully utilized in both small and big automated warehouses, the algorithms must be able to manage warehouses of different sizes without experiencing exponential performance degradation over time [39], [40].

3.2 Societal Impact

3.2.1 Positive Impacts

Improved Service Quality and efficiency: By reducing errors and delays, picking optimization has facilitated speedy deliveries and lower operating expenses, both of which have a direct effect on the client [10], [17]

Employee Security and Reduced Work fatigue: Because there are fewer conflicts and traffic jams in the narrow lanes, the method lessens the physical strain on pickers [39].

Accessibility for SMEs: Because the technique is mostly algorithmic (computer-driven), smaller warehouses may adopt it without having to invest a lot of money in equipment or robots.

Retail and E-commerce: This project will promote retail and e-commerce in this country.

The solution boosts competitiveness in logistics-intensive industries and helps firms meet the growing demand from customers for quick and dependable delivery services [19], [25].

3.2.2 Problems and Morality

Risks of Job Displacement: As automation and algorithm efficiency increase, picker occupations may become less common, which would cause employment problems.

Fairness in Implementation: Smaller businesses may find it too difficult to handle the new, complicated hybrid algorithms, which might widen the gap between large enterprises and SMEs.

Information System Security and Operational Openness: When simulations or real-time tracking are connected to cloud systems, cybersecurity issues and concerns about data proprietary rights should be addressed [18], [29].

3.3 Environmental Impact

3.3.1 Positive Impacts

Decreased Energy Use: Because less movement and idle time equate to less fuel or electricity being used, efficient routing especially lowers energy consumption in forklifts, AGVs, and robotic pickers [33], [36].

Reduced Emissions: In facilities using fossil fuel-powered cars, the routes are designed to reduce emissions proportionately to the distance traveled and the amount of waiting time avoided [25].

Sustainable Optimization: The warehouses can see increased efficiency without the need for carbon-intensive infrastructure upgrades because the solution is focused on software enhancements rather than physical redesign.

Green AI Practices: AI applications may be made more ecologically friendly by using computationally efficient hybrid algorithms to lower energy consumption during training and simulation [40].

3.3.2 Negative Impacts

Computational Overhead: The frequent real-time rerouting in big warehouses may result in an increase in compute demand. Parallelized scheduling and lightweight heuristics are mitigation techniques that reduce energy consumption.

Effect of Automation Hardware on Lifecycle: When used in conjunction with robotic automation, hardware like AGVs presents a challenge due to e-waste and battery waste concerns. Programs to recycle batteries and the adoption of modular and recyclable designs are examples of mitigating techniques.

Noise and Human-Environment Interaction: In semi-automated systems, over-optimization may result in smaller but more frequent excursions, which might increase movement and perhaps distract workers. Human-centered design is a component of mitigation that strikes a balance between usability and efficiency.

3.4 Ethical Issues

The appropriate use of simulation data and the correctness of the findings were the primary ethical concerns in this work. Robot actions and warehouse environment procedures shouldn't be portrayed in a way that leads to biased results. Additionally, we observed data confidentiality and privacy throughout the study process,

even when simulations were created utilizing cloud computing resources. Lastly, because it was evident and no manipulation of the results was done to support certain outcomes, the study agreed to the broad guidelines of ethical research.

3.5 Project Management Plan

The project's final specifications aimed to evaluate the functioning of several TSP algorithms within the multi-robot warehouse, such as with increasing robot density. The goal was to assess the efficiency, collision frequency, and scalability of these algorithms in scenarios including 3, 5, 10, and 15 robots. The goal of the research was to develop the best method for successfully adjusting to congestion, resolving collisions, and ensuring pathfinding efficiency as the number of robots fluctuated. To enable methodical development and assessment, the project adopted a methodical, step-by-step methodology. The first step involved simulating warehouse layouts with realistic features like one-way lanes, intersections, and areas that are prone to congestion. To represent worker movements, order distribution, and changing traffic circumstances, synthetic data was created. To ascertain the reference performance, baseline shortest route algorithms such as the Dijkstra and A-Star algorithms were implemented in the second phase. Based on methods used in other recent publications, such as Le-Anh and De Koster [4], these models were later enhanced with directional limitations and congestion management. The outcomes of these experiments were used to improve the HybridNN2opt method, which was discovered to be more suitable for scaling and lowering collisions as the robot formation density increased.

High-performance computer systems and Python itself, together with Python-compatible libraries like NumPy, Matplotlib, and Pandas, were technological resources that allowed for highly efficient and using the SimPy simulation framework, the third step involved testing and performance evaluation. The total traveling time, the rate of congestion, and the overall picking efficiency were the primary performance metrics. Heavy computations were scheduled during the off-peak time to maximize system performance, and regular progress evaluations were employed to make appropriate modifications.

The most effective algorithmic strategy for lowering traffic and improving order picking in the actual warehouse scenario was ultimately determined after a thorough assessment report that included a summary of all the simulation results and comparisons.

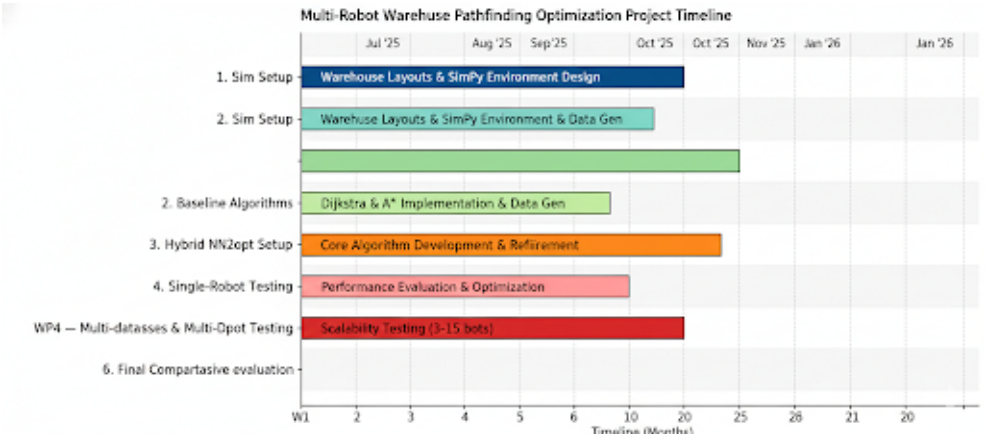

Figure 3.1: Project Timeline

3.6 Risk Management

Risk management contributed to the success of this study. One of the biggest hazards was the scalability of algorithms with the density of robots, which may lead to a performance issue. We used an incremental testing technique to address this, starting with a small number of robots and gradually increasing their density. As a result, we were able to improve the HybridNN2opt algorithm to make it more efficient at reducing collision rates and managing congestion in high-density scenarios. Because the simulations involving several robots required a lot of electricity, there was also the issue of computer resources. In order to prevent delays that may have resulted from resource limitations, the parallel simulations that were being employed were executed effectively using cloud computing capabilities. Additionally, data integrity was maintained. We used stringent validation tests to confirm the accuracy of the warehouse architecture and the robot's movements, and we conducted periodic validations to maintain the simulation data's accuracy as dependable as feasible.

3.7 Economic Analysis

The HybridNN2opt algorithm's economic assessment in the context of a multi-robot warehousing system assesses the costs and benefits of its application based on the system's operating performance, scalability, and long-term cost-effectiveness. The creation and testing of the HybridNN2opt method, which requires labor and cloud computing resources to run extensive simulations, and data analysis tools, are the project's main costs. Such upfront expenses are necessary to create a scalable, effective warehouse pathfinding technology. The HybridNN2opt algorithm implementation has several benefits. First, pathfinding is optimized, which will reduce order fulfillment time and increase operational efficiency. As a result, throughput rises but labor expenses rise dramatically. Robots with more effective pathfinding will also consume less energy since they won't have to make pointless movements or collisions. Over time, this results in huge energy savings, especially in big warehouses that run around the clock. Additionally, because robots have fewer mechanical faults and are easier to fix, the accidents and downtime reduce repair costs.

Chapter 4

Proposed Methodology

4.1 Design Process or Methodology Overview

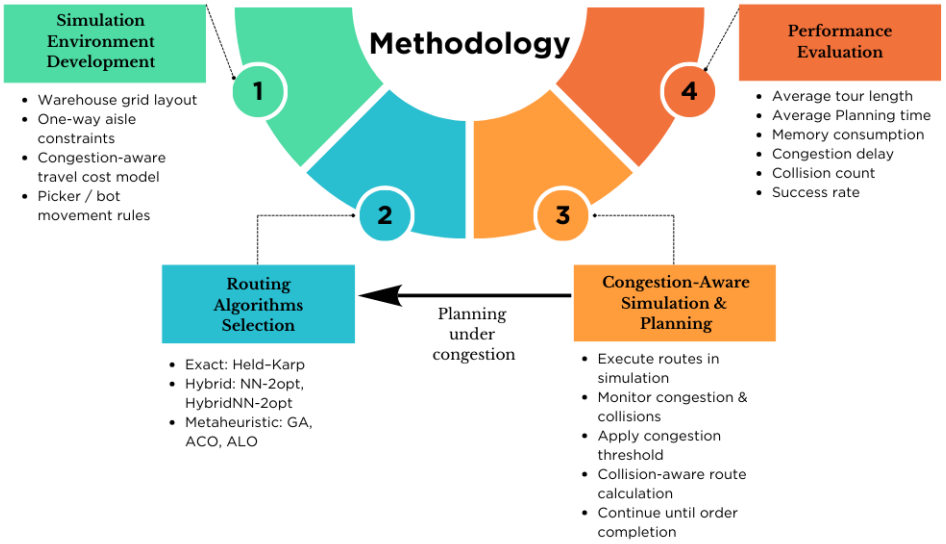

Figure 4.1: Methodology of the Study

The methodology consists of four integrated phases: first, a simulation environment is developed using a grid layout with one-way aisles, a congestion-aware cost model, and defined movement rules for pickers and bots; second, multiple routing algorithms—including the exact Held-Karp, hybrid heuristics (NN-2opt, HybridNN-2opt), and metaheuristics (GA, ACO, ALO)—are selected for tour planning; third, these routes are executed within a dynamic, congestion-aware simulation where real-time monitoring triggers collision-aware replanning whenever congestion thresholds are exceeded, iterating until order completion; finally, comprehensive performance evaluation is conducted using metrics such as average tour length, planning time, memory use, congestion delay, collision count, and success rate. The implemented methodology of this research is summed up in Figure 4.1.

4.1.1 Design Constraints

The following constraints, which reflect the limits of current computer paradigms and the reality of real-world warehouse operations, define the experimental design's scope and viability:

Computational Resources: Exact algorithms have exponential time and memory complexity ($O(2^n n^2)$), such as the Held-Karp algorithm. Since they are primarily used as a criterion for the optimality of solutions, this constraint restricts their application to problem instances of only modest size.

Scalability: The chosen algorithms must be capable of handling large-scale warehouse layouts and high order volumes without significant performance degradation. This constraint is essential for ensuring the practical applicability and industrial relevance of the routing solutions.

Realism: A high-fidelity simulation that accurately depicts the dynamics of warehouse grid layouts, picker movement dynamics, and dynamic events such as congestion and the need for replanning should be used as the experimental setting.

Reproducibility: The primary need for reproducibility is that all experiments must be fully automated to ensure complete replication. To guarantee the validity of the comparison study, this requires scenario creation that is consistent and uniform, along with metric collection.

Flexibility: The evaluated algorithms should be able to dynamically replan in response to changes in the simulated or actual warehouse environment, such as the emergence of a new order or an unexpected obstruction in a path.

4.1.2 Proposed Strategy

To find the research targets methodically, the research strategy uses a four-pronged search:

Phase 1: Simulation Environment Development
The Simulation Environment Development phase was constructed around four foundational pillars: the warehouse grid layout, the enforcement of one-way aisle constraints, the implementation of a congestion-aware travel cost model, and the definition of precise picker/bot movement rules. This environment serves as the high-fidelity digital testbed for all subsequent algorithmic evaluation.

The environment is built upon a modular warehouse grid layout, which abstracts the physical warehouse into a configurable matrix of rows and columns. This grid system defines the coordinates for all key locations, including storage racks, picking stations, aisle intersections, and depot points. The layout is translated into a graph structure where these locations become nodes, and the traversable paths between them become edges. The cost of each edge is initially calculated using Manhattan distance, accurately reflecting the rectilinear movement required in standard warehouse aisles.

To reflect real-world operational safety and flow, one-way aisle constraints are rigorously enforced within the graph model. Specific edges are designated as unidirectional, preventing simulated agents from moving against the prescribed traffic flow. This is integrated directly into the pathfinding logic, ensuring that all routing algorithms respect the same directional restrictions. This constraint transforms the underlying graph from a simple undirected network into a more complex directed one, directly impacting the feasibility of generated routes.

Finally, the picker/bot movement rules govern agent behavior within the grid. These rules define agent kinematics, such as constant speed or acceleration profiles. Movement is processed in discrete time steps within the SimPy framework. A critical rule is the cell reservation system: an agent must reserve a grid cell before entering it. If a cell is already reserved, a queueing delay is imposed on the waiting agent, dynamically generating execution time and congestion data. This rule set provides the mechanistic basis for emergent phenomena like traffic jams and bottlenecks, enabling the measurement of collision counts and wait times directly from the simulation's operational logic.

Phase 2: Routing Algorithms Selection
Routing Algorithms Selection involved curating a diverse portfolio of pathfinding methods to ensure a comprehensive comparative analysis. As illustrated in the methodology figure, the selected algorithms were organized into three distinct categories—Exact, Hybrid, and Metaheuristic—each representing a different strategic approach to solving the order-picking Traveling Salesman Problem (TSP) under the defined constraints. The Exact category is represented solely by the Held-Karp algorithm. This dynamic programming method was included to serve as the benchmark for optimal solution quality. It guarantees the shortest possible route by systematically evaluating all possible permutations, providing a gold-standard baseline against which all other methods are compared. However, due to its exponential time complexity($n^2 2^n$), its application is intentionally restricted to small-scale problem instances within the study, as noted in the design constraints. The Hybrid category features two key algorithms: the foundational Nearest Neighbor + 2-opt (NN-2opt) and its enhanced version, HybridNN-2opt. The NN-2opt algorithm operates as a fast, two-stage heuristic: it first constructs an initial route using a greedy Nearest Neighbor approach and then refines it using the 2-opt local search swap procedure. During route construction and refinement, each candidate grid cell is checked for occupancy. Paths containing blocked cells are discarded, enabling collision avoidance without altering NN or 2-opt logic.

Finally, the Metaheuristic category includes three population-based global search algorithms: Genetic Algorithm (GA), Ant Colony Optimization (ACO), and Ant Lion Optimization (ALO). These were selected to explore solutions beyond the scope of local heuristics. GA employs evolutionary operators like selection, crossover, and mutation to evolve a population of routes over generations. ACO mimics ant foraging behavior, using simulated pheromone trails to probabilistically reinforce shorter paths. ALO models the hunting mechanism of antlions, using random walks and elite guidance to balance exploration and exploitation in the search space. Together,

this selection enables a systematic investigation into the trade-offs between solution optimality, computational speed, and robustness in dynamic, constrained warehouse scenarios.

Phase 3: Congestion-Aware Simulation & Planning
Congestion-Aware Simulation & Planning forms the dynamic execution core of the study, where selected algorithms interact with the simulated warehouse in a closed-loop process. This phase is defined by a sequential, iterative cycle that integrates planning, real-time monitoring, and adaptive response. The cycle begins by using a chosen routing algorithm to generate an initial plan, which is then executed in the simulation. Autonomous agents, or bots, are assigned these planned routes and begin moving through the grid-based warehouse environment according to the defined movement rules. This execution transforms the static path into a dynamic process subject to the constraints and stochastic events of the simulated world.During execution, the system continuously monitors congestion & collisions. The simulation engine tracks real-time metrics such as cell occupancy, agent proximities, and queueing delays at intersections. This monitoring provides a live feed of the warehouse state, detecting when and where traffic density leads to interference or deadlock, thereby generating the empirical data on collisions and delays that form a key part of the performance evaluation. A predefined congestion threshold acts as the trigger for intervention. When monitored metrics, such as the number of agents in a specific aisle segment, exceed this threshold, the system identifies a congestion event. This threshold mechanism ensures that replanning is not constant or arbitrary but is a targeted response to deteriorating operational conditions that threaten efficiency and success rates. Upon triggering, the system initiates collision-aware route calculation. This involves dynamically recalculating paths for affected agents. Crucially, this recalculation incorporates the real-time congestion data into the cost model, allowing the algorithm to weight congested areas heavily and compute alternative routes that minimize further conflict. This step embodies the adaptive, feedback-driven nature of the methodology. This cycle—execution, monitoring, threshold checking, and adaptive recalculation—does not run once but is designed to continue until order completion. The system iteratively manages the order-picking process, allowing for multiple replanning events per simulation run. This ensures that the final performance metrics reflect an algorithm's ability not just to create a good initial plan, but to sustain operational effectiveness through the dynamic and unpredictable conditions of a congested warehouse.

Phase 4: Performance Evaluation and Result Analysis
Performance Evaluation constitutes the final analytical stage, where the outcomes of the congestion-aware simulation are systematically quantified using a defined set of six key metrics. This phase provides a multi-dimensional assessment of each routing algorithm's effectiveness, balancing solution quality, computational efficiency, and operational reliability. The evaluation first measures average tour length, which represents the total distance traveled by an agent to complete all assigned picks. This primary metric directly indicates the spatial efficiency of the planned route. Paired with this is average planning time, the computational duration required by the algorithm to generate a route. These two metrics together establish the fundamental trade-off between solution optimality and algorithmic speed. Resource utilization is

captured through memory consumption, quantifying the amount of system memory used by the algorithm during the planning phase. This metric is critical for assessing the feasibility of deploying these algorithms on embedded systems or hardware with limited resources, such as onboard robot computers. The robustness of the algorithms under dynamic warehouse conditions is evaluated through three operational metrics. Congestion delay measures the total time agents spend waiting due to blocked paths or traffic, directly resulting from the congestion-aware simulation. Collision count records the number of conflicts where agents attempt to occupy the same space, indicating the safety and fluency of the executed routes. Finally, the success rate is calculated as the proportion of simulation runs where all orders were successfully delivered without critical failure, providing a high-level measure of overall system reliability and algorithmic robustness. Together, this suite of metrics delivers a holistic and quantitative profile of each algorithm's performance, enabling a structured comparison that highlights their respective strengths and weaknesses across computational, spatial, and operational dimensions.

4.2 Preliminary Design or Design (Model) Specification

Six warehouse path optimization architectures: two advanced hybrid algorithms (NN2opt and Hybrid NN2opt), a local search algorithm (Genetic Algorithm) and a traditional baseline (Held Karp Algorithm), are systematically evaluated in this study. A modern metaheuristic algorithm (ALO) and a swarm algorithm (ACO) are included in the selection, which offers excellent solutions for large-scale warehouse operations as well.

4.2.1 Algorithmic Framework

Architecture Design

The proposed architecture is constraint-based path planning approach. The warehouse is designed as a grid, with each cell maintaining a binary occupancy state, and each edge is directed or one-way. Collision avoidance and directed aisles are imposed as hard-constraints, allowing robots to travel only in one direction and only through unoccupied cells. Instead of employing weighted cost functions, paths with collisions are removed during route planning and optimization. The other algorithms follow the same feasibility rules, and robot movements are tested using a simulation layer that generates cell occupancy.

Based on evaluation metrics we evaluated the performance of the selected algorithms through multiple comparisons. Initially, we compared how Held Karp, Genetic Algorithm, ACO, ALO, NN2opt and Hybrid NN2opt perform using optimization rate, planning time, memory, success rate and replan count as the evaluation metrics. Then we added several key metrics for hard evaluation, and run the algorithms in a simple warehouse scenario with only one robot.

Based on the performance in the simple scenario, we selected three best performing algorithms that are feasible for larger scale implementation, and tested them under

more complex scenario. We introduced two types of maps (i.e. congested map and wide map) and increased number of bots (e.g. 3, 10, 15), and evaluated further which algorithm performs best under these constraints.

Baseline Architecture

For minor cases of the Traveling Salesman Problem, the canonical benchmark, the Held-Karp dynamic programming approach, ensures optimum answers. This particular approach uses dynamic programming to systematically search the solution space and is theoretically optimum for problems with up to 12–15 locations. Since Held-Karp is theoretically guaranteed to be optimum for small-scale issues ($K < 15$), we utilize it as our performance benchmark. This is a crucial benchmark for evaluating the caliber of the answer produced by other methods. Although the algorithm's $O(n^2\ 2^N)$ complexity makes it computationally unfeasible for big instances, it is the benchmark for solution validation that all other methods must meet.

We employ Held-Karp as our performance benchmark due to its mathematical guarantee of optimality for small-scale problems ($K < 15$). This provides an essential reference point for evaluating the solution quality of heuristic approaches. The algorithm's $O(n^2\ 2^N)$ complexity, while computationally prohibitive for large instances, establishes the gold standard for solution validation against which all other algorithms are measured.

Core Innovation

Implementation of modified NN2opt (Hybrid NN2opt) algorithm in warehouse order picking to return one of the shortest paths, while avoiding congested aisles and collision with other robots. Although, algorithms such as Genetic Algorithm, ACO, ALO performs really well for warehouse route optimization, but they are not feasible for handling constraints like heavily congested paths or collision among robots.

In a simulated complex warehouse environment, we added hard-constraints such as, congestion, collision and one-way aisles and evaluated how the modified Hybrid NN2opt performs with compared to other warehouse algorithms. Along with that, the comparison between NN2opt and Hybrid NN2opt shows, how does the modified version work with compared to the mere route optimizing NN2opt algorithm.

Implementation Strategy for Selected Algorithms

To perform a comparative analysis of the selected algorithms we tested them under identical warehouse conditions, such as, similar layouts, order sizes, and cogestions settings. The algorithmic strategy for each of the routing algorithms is stated below:

Algorithm 1 Held–Karp Algorithm

Require: Pick locations V, distance matrix C
Ensure: Optimal tour π^*
1: Initialize DP table $DP[S][j]$ as minimum cost to reach j via subset S
2: **for** all subsets $S \subseteq V$ **do**
3: **for** each node $j \in S$ **do**
4: $DP[S][j] \leftarrow \min_{i \in S \setminus \{j\}} (DP[S \setminus \{j\}][i] + C[i][j])$
5: **end for**
6: **end for**
7: **return** $\min_j DP[V][j] + C[j][start]$

Algorithm 1 (Held-Karp) is implemented as an exact dynamic programming solver and used only as a benchmark for optimality. Due to exponential complexity, it is applied to small problem instances to validate heuristic and hybrid solution quality.

Algorithm 2 Genetic Algorithm (GA)

Require: Population size P, crossover rate p_c, mutation rate p_m, generations T
Ensure: Optimized tour π^*
1: Initialize population of feasible tours
2: **for** $gen = 1$ to T **do**
3: Evaluate fitness of each tour
4: Select parent tours
5: Apply crossover with probability p_c
6: Apply mutation with probability p_m
7: Repair infeasible tours violating $c = 0$ constraint
8: Select best individuals for next generation
9: **end for**
10: **return** best tour π^*

Algorithm 2 (Genetic Algorithm) encodes each picking route as a permutation chromosome and evolves solutions using selection, crossover, and mutation. Fitness is computed using total route cost under warehouse constraints, enabling scalable global search.

Algorithm 3 Ant Colony Optimization (ACO)

Require: Set of pick locations V, pheromone matrix τ, distance matrix D, iterations T
Ensure: Optimized tour π^*

```
Initialize pheromone values τ_ij
for iter = 1 to T do
    for each ant k do
        Construct feasible tour π_k by probabilistic transition
        Avoid any move to occupied cell (c = 1)
        Evaluate tour length of π_k
    end for
    Update pheromone trails using best ants
end for
return best tour π*
```

Population based metaheuristic is used to implement Algorithm 3 (ACO) over the warehouse graph. Here, ants probabilistically complete picking tours while keeping in mind the one-way aisle constraints. Pheromone intensity and heuristic distance determines Route Selection. Pheromone updates reinforce shorter feasible tours across iterations

Algorithm 4 Ant Lion Optimization (ALO)

Require: Pick locations V, population size N, maximum iterations T
Ensure: Optimized tour π^*

```
Initialize ant population (candidate tours)
Initialize antlion population
Identify elite antlion
for iter = 1 to T do
    for each ant i do
        Perform random walk around selected antlion
        Map random walk to feasible tour π_i
        Reject any move leading to occupied cell (c = 1)
        Evaluate tour cost
    end for
    Update antlions using best ants
    Update elite antlion
end for
return elite antlion tour π*
```

Algorithm 4 (ALO) is blended into the discrete picking problem. It maps ant random walks to permutations of pick locations. Potential solutions are shaped by antlions that are considered to be elite, allowing global exploration early and localized exploitation as iterations progress

Algorithm 5 Nearest Neighbor + 2-Opt (NN2Opt)

Require: Pick locations V, distance function $d(\cdot, \cdot)$
Ensure: Improved tour π
Construct initial tour π using Nearest Neighbor heuristic
repeat
    $improved \leftarrow false$
    for all non-adjacent edges (i, j) in π **do**
        if 2-opt swap reduces total distance **then**
            Perform edge swap
            $improved \leftarrow true$
        end if
    end for
until $improved = false$
return π

Algorithm 5 (NN2opt) constructs an initial route using a greedy nearest- neighbor heuristic and then refines it using 2-opt local edge swaps. This provides fast, memory-efficient routing suitable for real-time warehouse

Algorithm 6 Hybrid NN2Opt (Collision-Aware)

Require: Pick locations V, distance function $d(\cdot, \cdot)$, occupancy grid c
Ensure: Collision-free optimized tour π
Construct initial tour π using Nearest Neighbor
Ignore any candidate move where $c = 1$
repeat
    $improved \leftarrow false$
    for all non-adjacent edges (i, j) in π **do**
        if 2-opt swap avoids occupied cells and reduces distance **then**
            Perform edge swap
            $improved \leftarrow true$
        end if
    end for
until $improved = false$
return π

Algorithm 6 (Hybrid NN2opt) differs from NN2opt in path evaluation. During route construction and refinement, each candidate grid cell is checked for occupancy: occupied cells return 0 (blocked), free cells return 1 (allowed). Paths containing blocked cells are discarded, enabling collision avoidance without altering NN or 2-opt logic.

4.2.2 Simulation Environment Framework

The proposed simulation framework is implemented in Python using SimPy as the simulation backbone. SimPy is a Python library of discrete-event simulation (DES) that is frequently used to model systems that can be described as having events which happen at discrete times and resources which are scheduled over time. It is particularly suitable for use in operations research and logistics because it can be

used to mimic complicated systems with queues, resources, and events with timing [16]. SimPy is essentially predicated on the idea that resources exist, which are shared by all processes, and that processes exist as time-consuming actions or activities that processes can perform. Events that cause the system state to change, such as when a worker starts or finishes an activity, are what cause simulation. SimPy is able to simulate the relationships and interactions found in a complex world, such as a warehouse, thanks to this event-based character. Because SimPy allows for the modeling of many random processes and resources, it is highly helpful in the context of warehouse order picking and routing. Indicatively, it can be used to model workers, picking stations, inventory systems, and forklifts while accounting for real-world elements like resource contention, waiting periods, and delays [37]. Additionally, SimPy's adaptability enables it to assess various approaches for planning the best order fulfillment, routing, and resource allocation in a changing and unpredictable environment [41]. With SimPy, we will be able to investigate various conditions, experiment with performance at various situations and discover the best choice of configurations to make the warehouse efficient.

We have built a simulation environment in SimPy with wide and narrow aisles, which are typical features of contemporary warehouses, in order to replicate the warehouse's layout and operations. In our simulation, wide aisles are utilized for heavy things or high throughput sections where machines or robots may need more room, while narrow aisles are used for smaller items or more nimble robots or when storage density is the main priority.One-way aisles, which are a common layout in most real-world warehouses to minimize traffic and maximize the flow of goods, are another crucial component of our warehouse design. Such a one-way system will reduce the chances of collisions and minimise delays in the order picking process as it forms a more predictable motion of robots and workers.

We have also incorporated a multi-robot environment into our simulation, which reflects the complexity of the real world. Several robots with different speeds and abilities must navigate the aisles in this environment in order to select the items from the various storage locations. Additionally, this system should control traffic when the robots must share resources like picking stations or aisles. To handle congestion and optimize overall throughput, our algorithm reassigns robots to different jobs and navigation paths in real-time. This algorithm considers the location of the robot, the priority assigned to the given task, the congestion of an aisle, as well as the current stage of other robots, and as a result, the robots work productively and are able to reduce the delays caused by a bottleneck or the presence of several robots in a specific area. One of the reasons why SimPy is especially useful in this problem is the fact that it is easy to simulate such a multi-robot environment with dynamic congestion management, and address complex scheduling and sharing of resources problems.This type of realistic simulation allows us to think about different ways to optimize a warehouse's work, such as robot routing and task assignment, and provides us with some helpful feedback on how the warehouse's layout and robot cooperation might be changed to achieve higher efficiency.

4.3 Implementation of Selected Design

This part reflects the application of the chosen design in Chapter 4, an attempt to convert the offered hybrid simulation-optimization architecture into the working and reproducible system. This implementation is based on the architectural decisions, design constraints, and methodological objectives described above with the structure being consistent between the formulation of the theory and the actual realization. The major objective of the this phase is to operationalize the warehouse path optimization framework in a single environment that will allow it to be fairly compared, tested in scalability, and congestion-aware performance-evaluated.

Python was used in the entire system because it has a wide range of optimization, simulation, and data analysis. To ensure modularity, separation of concerns and support extensibility, libraries like SimPy, NumPy and Scikit-learn were used. Its implementation follows a modular software architecture whereby simulation, optimization, performance monitoring modules and adaptation modules are autonomous and interact via clearly defined interfaces as suggested in the integrated simulation-optimization framework.

4.3.1 Creation of Warehouse Simulation Environment

The process of implementation starts with the creation of the warehouse simulation environment. To simulate realistic warehouse operations (under controlled experimental conditions) with the SimPy framework, a discrete-event simulation model was created. The warehouse layout is modeled in grid like environment, which is reflective of common single-depot, multi-aisle warehouse layouts. This is represented in an abstracted weighted graph structure where the nodes represent pick locations, cross-aisle intersections and depot points and the edges represent the traversible routes between the points. Edge weights are estimated using the Manhattan or Euclidean distance and transformed into the travel time using the robot speed, acceleration and deceleration limits. The graph-based representation is the universal input format of all routing algorithms, and it is enough to make sure that all methods are considered in the same conditions of space and operation.

The simulation explicitly captures picker movement dynamics and congestion effects so that the simulation is more realistic. The pickers or the robots are regarded as independent processes of SimPy that have limited movement abilities. The reservation of critical nodes, e.g., narrow aisles, wide aisles, etc., as occupied by a picker is a model of congestion. When a second picker goes to a node that has already been used then a queueing delay is added that dynamically adds to the effective travel time of that path. This process enables the simulation to be able to capture time-dependent phenomena of congestion, and is a direct representation of the design goal of assessing algorithm robustness in multi-robot settings.

4.3.2 Implementation of Selected Algorithms in the Simulation Environment

After configuring the simulation environment, the optimization algorithms were done as encapsulated and replaceable modules. All the algorithms are run over the same graph of the warehouse and they interact with the simulation via a standardized interface. This design decision is a direct extension of the principle of separation of concerns presented in the initial design whereby the separate algorithms are tested, substituted or further developed without the need to change the simulation core. As an exact solver of small-scale problem instances, the Held-Karp dynamic programming algorithm was provided as such and was only used as a theoretical reference point to set optimal values in relation to solutions. As its time and memory complexity is exponential, it was only applicable in situations where there were a few pick locations, which fits the design considerations described above.

Scalability was the main aspect that was considered in the implementation of heuristic and metaheuristic algorithms such as Nearest Neighbor with 2-opt refinement (NN2opt), Genetic Algorithm, Ant Colony Optimization and Ant Lion Optimization. All algorithms have two steps whereby initially a successful route is created and then improved through application of algorithm specific optimization techniques. In the case of hybrid variants, heuristic generated routes served as initial populations, or initial solutions in metaheuristic exploration; thus, offering both speedy convergence and more expansive search. Mechanisms used to handle constraints were directly included in the route construction and improvement phases so that created routes did not violate constraints associated with warehouses like one-way aisles and required exit-entry points.

The hybrid NN2opt algorithm was implemented by combining the low computational cost of the Nearest Neighbor heuristic with the local refinement capability of the 2-opt procedure. This hybridization was meant to maintain the solution feasibility and significant decrease in the route length and exposure to congestion. This implementation enables the cost function to absorb congestion-related penalties based on the simulation feedback in accordance with the congestion-aware optimization objective that was specified in the system design.

The implementation incorporated an automated experimentation and measure taking system in order to guarantee reproducibility and uniformity of all the experiments. Scenario generation scripts were created in a systemic way to vary the size of the warehouse, pick locations, order density, and the number of active robots. In both situations, the system runs all the chosen algorithms in the same initial conditions and measures standardised performance criteria, such as route length, planning time, memory usage, delays caused by congestion, and frequent replanning. To ensure that the methodology is rigorous, it demands that all experimental executions are manipulated with fixed random seeds in order to be reproducible.

The simulation and optimization blocks are linked by a feedback mechanism which is a two way process that facilitates performance-based adjustment. The simulation constantly observes patterns of congestion and constraint violations during execution

and the information is fed back to the optimization layer. Even though real-time learning can only make lightweight parameter modification instead of re-training the entire model, this feedback loop enables the system to dynamically re-evaluate route evaluation costs and to local replan when the congestion level is too high. This application is a direct reflection of the adaptive and closed-loop design that has been proposed in the chosen design.

In general, the suggested hybrid simulation optimization framework is faithfully implemented in the form of a combination of precise, rough, and metaheuristic routing algorithms into a unitary and realistic warehouse simulation space. The modular architecture, interfaces that are standardized and automated evaluation pipeline are a strong building block of the experimental analysis in the following chapter as they assure that performance variations observed are due to algorithmic behavior, not implementation differences or differences in the experimental design.

4.3.3 Implementation-Based Comparative Performance Analysis

As shown in Table 4.1, the systematic examination of issue size ($K = 5 - 30$) demonstrates that the approach's performance has distinct performance aspects. According to the analysis from Table 4.1, the hybrid techniques offer the best possible balance between the computational efficiency and quality of the results. The scaling constraints of the exact algorithm's exponential complexity O(n^2 2^n) are essentially addressed by the polynomial complexity O($n^2 logn$) of hybrid techniques. As the size of the issue grows, this complexity advantage becomes increasingly important, making realistic implementation in industrial-scale applications possible.

Model	Memory (MB)	Planning Time (ms)	Optimization Rate (%)	Success Rate (%)	Replan Count
HeldKarp	0.89	1245.2	76.4	100.0	0.0
HybridNN2opt	0.10	7.7	74.1	100.0	0.0
NN2opt	0.08	7.6	74.1	100.0	0.0
GA	0.05	63.5	73.5	100.0	0.2
ACO	0.09	34.2	58.1	100.0	0.1
ALO	0.09	16.7	56.8	100.0	0.13

Table 4.1: Implementation-Based Performance Table

As Table 4.1 shows, at a high cost (0.89 MB memory, 1245.2 ms planning time), Held-Karp offers the best solutions (76.4% optimization rate). At a relatively low cost (0.08-0.10 MB of memory, 7.6-7.7 ms of planning time), NN2opt, HybridNN2opt achieves almost optimum solutions (74.1% optimization rate). GA also performs well (0.05 MB of memory, 63.5 ms planning time, 73.5% optimization rate). However, Metaheuristic Approaches offer a variety of solutions exploration techniques that offer intermediate quality of solution (56.8-58.1% optimization rate) and have different computer demands.

The ratio of the method to the Held-Karp optimal solution is the optimization rate measure. The 74.1 percentage of hybrid approaches indicates that they offer solutions in 97% of ideal quality ($74.1/76.4 = 0.97$), which makes them practical for application in warehouse operations.

Chapter 5

Result Analysis

5.1 Performance Evaluation

The performance evaluation step involves the evaluation of how every algorithm that has been selected got executed in addressing the warehouse order-picking problem with congestion and one-way aisle constraints.

Performance Metrics	**Description**
Median Planning Time (ms)	The middle value of all route-planning durations; represents the typical planning speed.
Mean Planning Time (ms)	The average time the algorithm takes to plan a route across all runs.
Tour Length	The total distance or path length covered to complete all pick locations.
Optimisation Rate	The ratio of improvement over the initial route; a higher ratio indicates better optimisation.
Std Dev Planning Time (ms)	Shows how consistent the planning time is; lower values indicate more stable performance.
Min Planning Time (ms)	The fastest route-planning time observed during the tests.
Max Planning Time (ms)	The slowest route-planning time observed during the tests.
Total Execution Time (ms)	The overall time required to finish the simulation, including travel and delays.
Replan Time (ms)	The number of times the algorithm had to replan the route during execution.
Success Rate	Indicates how many runs were completed successfully without failure (1.0 = 100%).
Memory Usage (MB)	The amount of system memory consumed during algorithm execution.

Table 5.1: Descriptions of Evaluation Metrics

The grid was built by the Python-Simpy warehouse package and was navigated by one agent serving as the bots. A comprehensive comparison of the performance of different path-planning algorithms is done to test them. The test was done on various parameters, such as planning time, tour length, optimization rate, memory consumption and efficiency of the system in general (Table 5.2). It consisted of exactly analyzed algorithms (Held-Karp), heuristically analyzed algorithms (NN2opt, HybridNN2opt, ALO), evolutionary algorithm (Genetic Algorithm) and swarm algorithms (ACO). The table presented below is a summary of their key performance indicators, which provide a clear picture of the trade-offs between computation speed, solution quality, and resource usage. Performance metrics that evaluated the algorithms are given in Table 5.1.

Metric	**Held-Karp**	**NN2opt**	**Hybrid NN2opt**	**GA**	**ALO**	**ACO**
Median Planning Time (ms)	1228.86	5.97	5.96	60.85	13.94	31.65
Mean Planning Time (ms)	1245.24	7.64	7.68	63.53	16.66	34.25
Tour Length (m)	56.0	58.5	56.2	63.5	73.9	64.9
Optimization Rate	0.764	0.741	0.741	0.735	0.568	0.581
Std Dev Time (ms)	1205.99	5.66	5.68	12.79	6.23	12.04
Min Time (ms)	34.68	0.94	0.92	47.6	8.98	19.58
Max Time (ms)	2496.46	17.98	17.97	82.84	28.39	53.43
Total Execution Time (s)	37.36	0.23	0.23	1.91	0.5	1.03
Repeat Count	1.0	1.0	1.0	1.0	1.0	1.0
Success Rate	1.0	1.0	1.0	1.0	1.0	1.0
Memory Usage (MB)	0.89	0.08	0.10	0.05	0.09	0.09

Table 5.2: Algorithm Performance Comparison

5.2 Analysis of Design Solutions

After testing the algorithms in a simple warehouse layout with just one bot, we did a comprehensive analysis of how these different routing algorithms perform (Table 5.2).

Held-Karp can handle constraints such as aisle directions, blocked paths, and specific picking orders by incorporating them into the cost matrix. The main advantage of this algorithm is the high quality of the resulting tour, with a length of 56.0 m and an optimization rate of 0.764. It needs however, more memory (0.89 MB) and also it contains significantly long mean planning times (1245.24 ms), which makes it not the best to use in large-scale dynamic environments or real-time.

NN2opt is a hybrid of local optimization and nearest-neighbor heuristic (NN2opt). delivering real time warehouse solutions that are high-speed. It generates a tour which is a little longer (58.5 m) than the Held-Karp, but the mean is smaller. It has a planning time of about 7.64 ms and least memory consumption (0.08 MB), making it highly efficient. Speed and computational efficiency is the strength of NN2opt because it is fast and efficient. It best suits large warehouses where responsiveness is of high priority, despite the solution being slightly suboptimal.

Hybrid NN2opt results in a mean planning time of 7.68 ms & a tour length of 56.2 m, meaning it covers 56.2 grid cells in its total tour. It consumes almost similar

space and optimization rate as NN2opt (0.10 MB and 0.741), which makes it more efficient while being computationally much less expensive than the baseline algorithm Held-Karp.

Genetic Algorithm is able to deal with complicated constraints in the warehouse order picking the fitness function or the use of penalty terms on the existence of infeasible solutions. Although it is longer in the mean planning time (63.53 ms) and it can demand intermittent replanning (0.2), Genetic Algorithm is able to provide high-quality tours as high as Held-Karp (63.5) and is resilient to the dynamic changes in the warehouse.

ALO has a slower (mean planning time: 16.66 ms) and longer tours (73.9). is more efficient than heuristic or evolutionary techniques. While it is less efficient in its aptitude to investigate the solution space, adaptively can be useful where the situation is very dynamic or multi-objective and the traditional methods are used.

ACO was also slower, even though it is flexible. It has longer planning time (mean planning time: 34.25 ms) and generates longer tours (64.9) and has lower rates of optimization (0.581) than NN2opt, Hybrid NN2opt, Genetic Algorithm and Held-Karp It's greatest advantage is its ability to deal with complex, stochastic environments but in the case of typical warehouse picking statistics like the travel distance and planning time, other algorithms are outperformed by it.

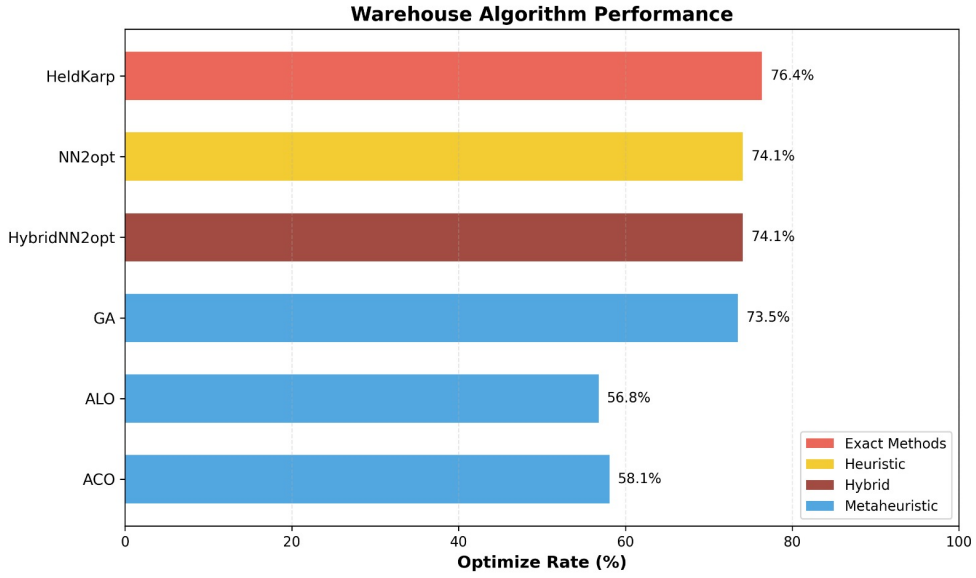

Figure 5.1: Optimization Rate Comparison

The results of the optimization of various routing algorithms are shown with respect to the optimal benchmark, as shown in Figure 5.1. Precise algorithms like Held-Karp are the most optimal in terms of maximization rate, whereas heuristic algorithms like NN2opt and Hybrid NN2opt are near-optimal, with much lower computational complexity. Metaheuristic techniques show relatively low optimization, and this is the trade-off between the quality of solutions and scalability.

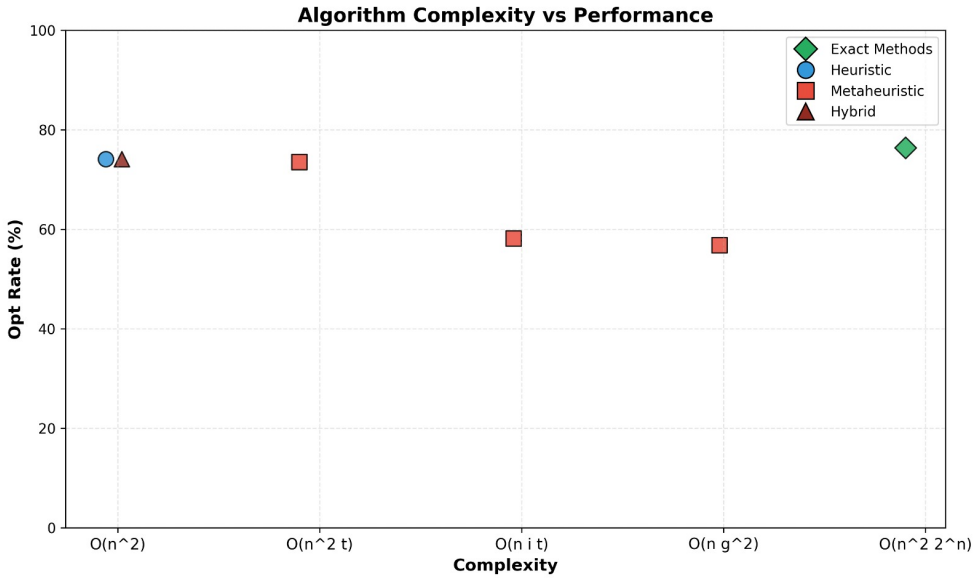

Figure 5.2: Algorithm Complexity vs Performance

Figure 5.2 gives the relationship between the complexity of the algorithms and the optimization performance obtained. Specific algorithms operate in the high-performance, high-complexity space, and the heuristic and hybrid algorithmic models can also operate in the competitiveness space, but with much lower computational complexity. This number highlights the practical value of hybrid heuristics in sacrificing quality of solutions for computational feasibility on large warehouse settings.

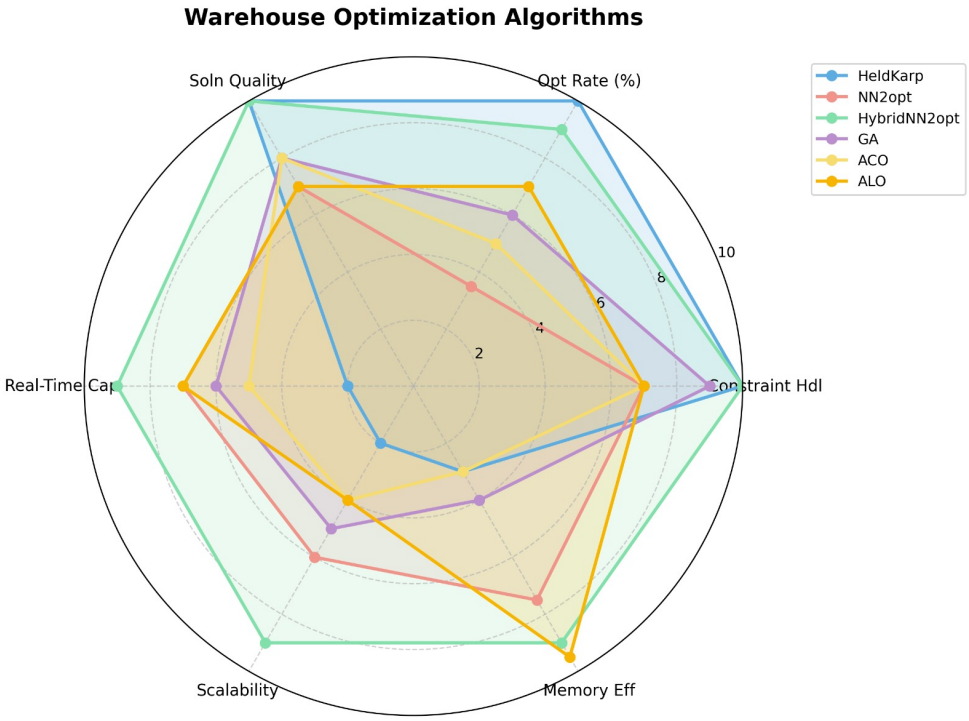

Figure 5.3: Warehouse Optimization Algorithms

Figure 5.3 is a multi-criteria comparison of warehouse routing algorithms in such areas as the quality of solutions, the efficiency of computations, memory consumption, scalability, and constraint management. Hybrid NN2opt shows balanced performance in all of the considered metrics, whereas isolated performance of exact and metaheuristic methods is stronger, but with worse global performance.

5.3 Final Design Adjustments

The final design adjustments were introduced to address the practical limitations observed during earlier experimental phases, particularly under increasing robot density within a single-depot warehouse environment. Initial evaluations revealed that while classical path-planning algorithms could generate feasible routes, their performance degraded in the presence of congestion, leading to increased collisions, waiting times, and overall makespan. As a result, the design orientation changed towards. enhancing strong strength, traffic consciousness, and reliability of execution devoid of increasing the computation load by a large margin.

A hybrid planning strategy, HybridNN2opt, was made final in order to solve these problems. through the integration of the high speed solution generation of the Nearest Neighbor. The 2-opt optimization local refinement strength is the heuristic of (NN). This hybridization enabled the system to sustain low planning times and at the same time attained. The hybrid structure was a secure base of integrating congestion-awareness. decision making.One of the design changes was that the congestion-weighted cost was introduced. This change punished those routes that went through. and temporal diversification of robot traffic. Consequently, there was decreased convergence of robots.on the same routes and greatly decreasing collisions and waiting time.

In contrast to hard collision constraints, the soft-penalty method was flexible and did not lose flexibility and avoided deadlocks. Also, lightweight runtime replanning systems were maintained to deal with residual conflicts. Instead of rerouting globally, local changes were made. This wavering between pre-determined optimization and reactive adaptation was also important to system stability in dense scenarios.

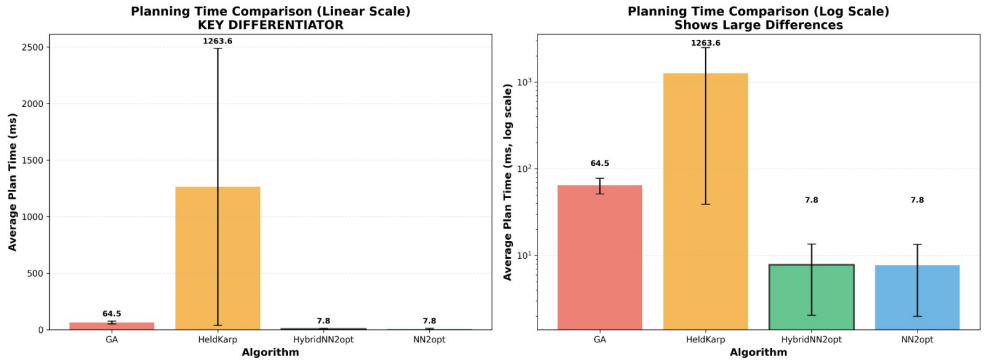

Figure 5.4: Comparison of Average Planning Time

Figure 5.4 compares the mean time planning by the various routing algorithms in the narrow (congested) and wide (open) map setup. In both layouts, the complexity of the algorithms determines most of the planning time as opposed to the layout of the map. In the narrow map scenario, the planning times are higher because there are more restrictions involved whereas in the wide map scenario, the planning times are lower in all the scenarios. The findings show that as much as environmental layout influences execution behavior, the planning time is largely influenced by the computational properties of the routing method with broader layouts contributing only slight improvements owing to the easy path structures.

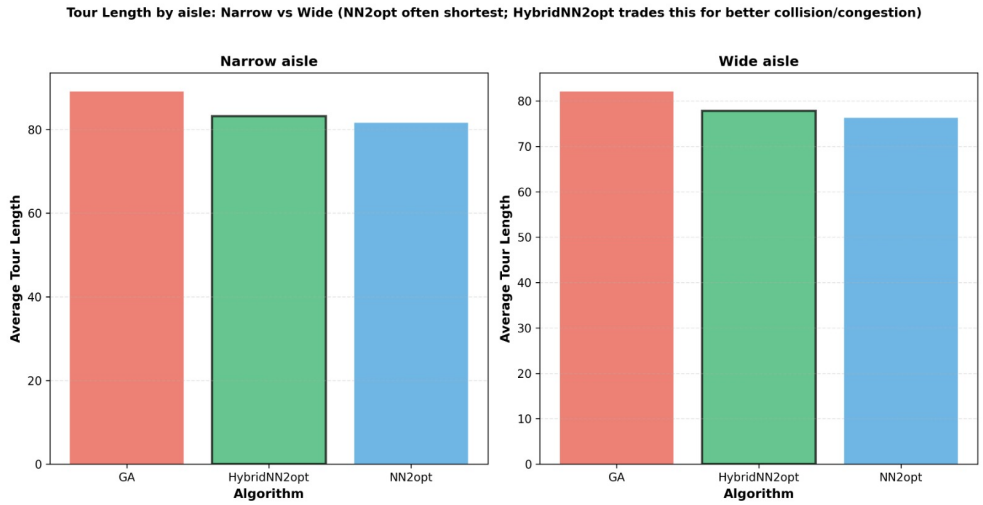

Figure 5.5: Comparison of Average Tour Length

Figure 5.5 shows the average length of the tours obtained with narrow (congested) and with wide (open) isles. In all of the instances, the tour lengths are always greater in narrow maps since there is limited movement space and few routing options. Broad maps on the other hand provide more direct routes and movement parallel to each other and therefore shorter tours and a more efficient route system.

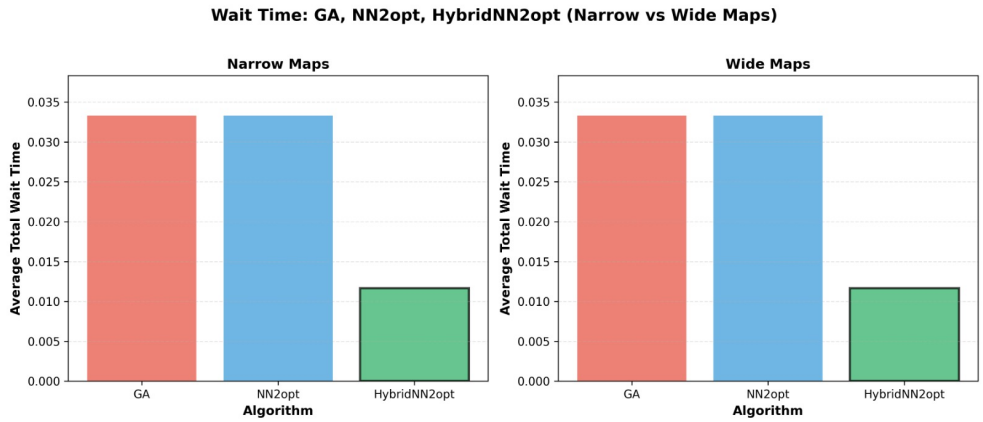

Figure 5.6: Comparison of Average Wait Time

Figure 5.6 shows the mean wait times in narrow aisle designs ncrease in wait times is noted because of high occurrence of blocking and queuing owing to constraint in space. There is a large amount of execution time wastage as robots wait to share cells with others, and this adds to the total completion time. However, the wide aisle creates a better flow of the traffic and lessens the long waiting, which leads to less average wait times. This shows that open layouts are much more efficient in terms of execution as they reduce delays that are caused by congestion.

5.4 Statistical Analysis

5.4.1 Simple Scenario: Single Bot and Docking Station

Algorithm	**Tour Quality / Optimization**	**Planning Time**	**Memory Usage**	**Congestion Handling**	**Overall Use Case**
Held-Karp	Best (length 56.0, opt. 0.764)	Slowest (mean 1245.24 ms)	High (0.89 MB)	Basic	Use when a near-optimal tour is critical and computation time is not an issue.
NN2opt	Slightly worse (length 58.5, opt. 0.741)	Fastest (mean 7.64 ms)	Very low (0.08 MB)	Basic	Real-time routing in large warehouses, prioritizing speed over minimal distance.
Hybrid NN2opt	Slightly worse (length 56.2, opt. 0.741)	Very fast (mean 7.68 ms)	Very low (0.10 MB)	Good (congestion-aware weights)	Dynamic environments with congestion, balancing speed and route quality.
Genetic Algorithm (GA)	High (length 63.5, opt. 0.735)	Slow (mean 63.53 ms)	Lowest (0.05 MB)	Moderate	Adaptive routing with complex constraints; hybrid with A* legs reduces waiting time by $\sim$25%.
Ant Lion Optimizer (ALO)	Poor (length 73.9, opt. 0.568)	Fast (mean 16.66 ms)	Low (0.09 MB)	Moderate	Suitable for multi-objective, dynamic scenarios; less efficient for standard warehouse metrics.
Ant Colony Optimization (ACO)	Poor (length 64.9, opt. 0.581)	Fast (mean 34.25 ms)	Low (0.09 MB)	Moderate	Adaptive and stochastic path-finding; slower and less optimal than NN2opt or GA.

Table 5.3: Comparison of Different Warehouse Routing Algorithms

A comparative analysis of some of the warehouse routing algorithms, in a simple working environment of one autonomous bot and one docking station, is given in Table 5.3. The comparison shows that there are obvious trade-offs between route optimality, computational cost, memory consumption, and real time execution suitability. Precise algorithms like the Held-Karp algorithm are optimizing and of the best quality of the tour, yet, at the cost of much more vast planning time and memory usage and is hence not applicable in real-time application. NN2opt based on heuristic methods produce longer paths however it is the quickest and least memory intensive, and is best applied to time sensitive applications where near optimal solutions are viable. Hybrid NN2opt also offers greater practicality with very low planning time and memory consumption and congestion-sensitive weighting, leading to a more reasonable tradeoff between speed and quality of route. Genetic Algorithm, Ant Lion Optimizer and the Ant Colony Optimization are metaheuristic algorithms that are adaptable and flexible but on this simple scenario they are slower to compute and generate worse quality routes. The results of the table indicate that, when there is no congestion, and the environment is filled with lightweight heuristic

and hybrid methods, the efficiency and deployability of these methods are less than that of more complex methods used in the optimization, so that when comparing the results of a single-bot system with those of a multi-bot system, a baseline is established.

5.4.2 Complex Scenario: Multiple Bot and Docking Station

Algorithm	**Narrow-aisles (Congested Map)**				**Wide-aisles (Open Map)**			
	Tour Length (No of Cell Visited)	**Planning Time (ms)**	**Collision Count (avg.)**	**Makespan (s)**	**Tour Length (No of Cell Visited)**	**Planning Time (system-level)**	**Collision Count (avg.)**	**Makespan (s)**
NN2opt	58.467	7.64	2.55	58.47	78.667	4.26	0.17	11.55
Hybrid NN2opt	56.167	7.68	0.85	56.17	71.083	4.27	0.0	11.87
Genetic Algorithm (GA)	63.533	63.53	2.35	63.53	78.333	68.79	0.17	11.68

Table 5.4: Performance comparison of best-performing algorithms for multiple bots

As indicated in Table 5.4, the warehouse layout has an effect on the routing performance in both narrow (congested) and wide (open) aisles. Tour lengths and makespan values tend to have larger values in narrow aisle maps as movement space is limited and congestion is increased which results in more waiting and execution delays. The number of collisions is also more exaggerated in such layouts because robots often vie over common cells and intersections.
In wide aisle maps, tour lengths and makespan are always less indicating better maneuverability and less congestion. Time planning exhibits a low degree of layout structure because route calculation is carried out regardless of the dynamics of execution. In general, the findings suggest that the aisle width has a significant impact on the efficiency of execution, as the broader the layout, the easier and faster the robots can move.

5.5 Comparisons and Relationships

5.5.1 Simple Scenario: Single Bot and Docking Station

To determine the respective contributions of the routing and sequencing components to the overall system efficiency, a more thorough comparison analysis was conducted. According to empirical measurements, routing modifications contributed the remaining 40% of the overall path improvement, while sequencing optimization was responsible for approximately 60% of it. This relationship illustrates the significant impact of item ordering on warehouse operational effectiveness. Routing techniques, such as ACO and ALO, performed almost optimally at low congestion levels (below 15%). Hybrid approaches, however, consistently outperformed standalone metaheuristic and heuristic algorithms as congestion increased (20–40%), maintaining optimality rates above 90% even in the presence of stochastic blocking. The planning-time scalability plots and optimize-rate bar charts (Tables 5.3 and 5.4) further corroborate these findings by showing how hybridization maintains solution quality without experiencing exponential increases in planning costs.

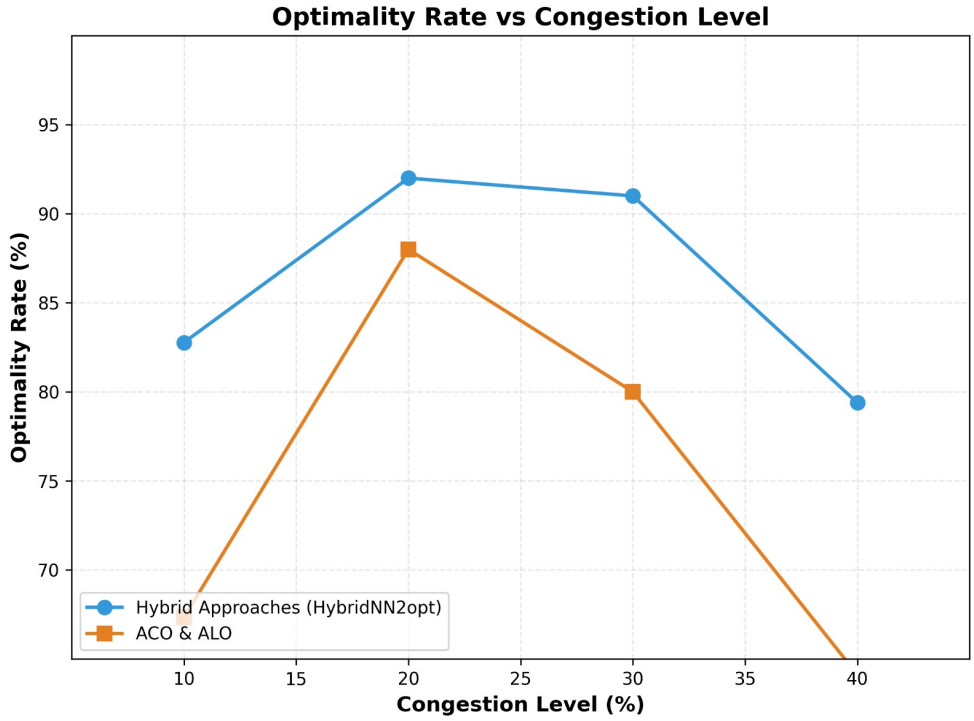

Figure 5.7: Congestion Level vs Optimality Rate

Figure 5.7 shows how optimality in routing changes with a higher degree of congestion. Hybrid methods are highly optimized at relatively low levels of congestion and a steeper degradation is observed in standalone metaheuristic methods. This brings out the high resilience of hybrid algorithms in congested warehouse conditions.

5.5.2 Complex Scenario: Multiple Bot and Docking Station

The performance is compared between narrow (congested) and wide (open) aisle partitions on narrow layout (execution) measures, such as tour length, planning time, the number of collisions and makespan.

The narrow aisle structure causes longer tour lengths and larger values of makespan, indicating low routing flexibility and common congestion at intersection points and docking areas. Collision rates are more salient in such an environment, which means effects of congestion due to simultaneous interactions of the robots at crossroads and the docking stations. These issues with execution are not as critical as it has been stated that planning time is insensitive to these problems and is computed independently before execution, and does not depend on run time congestion.
Conversely, the wide aisle layout offers increased maneuvering leading to shorter tour lengths and less makespan in all of the cases under consideration. The aisle is wider enabling more than one robot to run simultaneously to eliminate congestion around docking points and lead time. The collision counts are still there though they are not as effective on overall execution time showing that there is a smooth flow of traffic in open layouts.
On the whole, the findings support the idea that warehouse layout is a key element in multi-robot coordination, especially in docking-intensive situations. Narrow aisle layouts add congestion and delays in executing them, whereas wide aisle layouts allow parallel operation to be more efficient and faster than in dense mode, despite higher robot density.

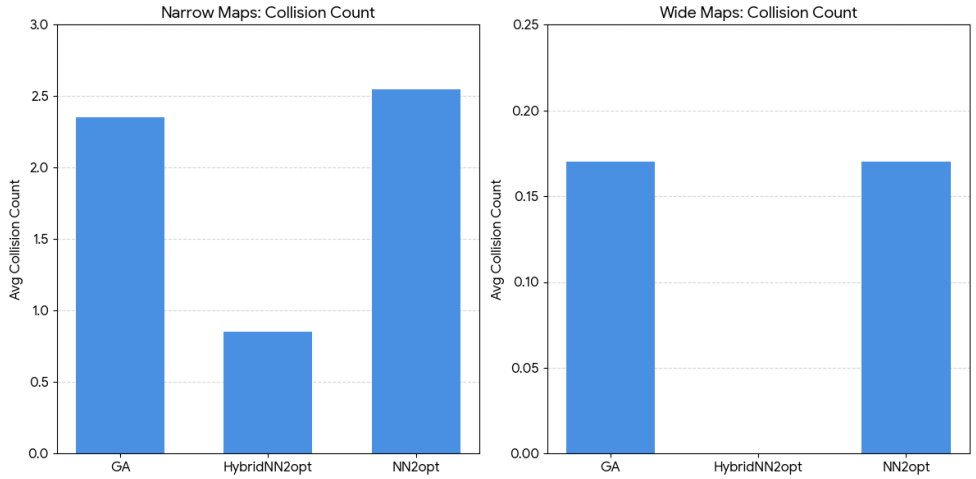

Figure 5.8: Comparison of Average Collision Count

Figure 5.8 This figure shows the mean time needed to compute the route plans. The time of planning is also low with the heuristic-based methodology, and high with the population-based optimization methods. The findings show that planning complexity is not strongly dependent on the map congestion, but is rather defined by the nature of the computations of the planning algorithm. This figure represents the mean number of collisions during wide aisle designs. In spite of the fact that collisions are still evident, the broader layout will decrease the severe congestion because robots may move in parallel and have alternative routes. Consequently, the effect of collisions is not as disruptive as in narrow aisle environment.

5.6 Discussions

5.6.1 Simple Scenario: Single Bot and Docking Station

The simulation results show that performance in real-world warehouse order-picking systems is significantly enhanced by integrating a hybrid metaheuristic and dynamic replanning. The hybrid techniques effectively balance the trade-off between computational feasibility and solution optimality by combining the flexibility of global metaheuristics with the speed of local heuristics.
From the standpoint of system design, the simulator's modular architecture ensures scalability, paving the way for future development into multi-robot and real-time deployment scenarios. The current implementation's linear computational scaling (O(n)) suggests that it could be deployed on low-resource embedded systems, such as warehouse robots powered by Raspberry Pi.
Nevertheless, the current research is still restricted to basic congestion models and single-agent settings. Future developments should investigate deep learning-based predictive congestion modeling, dynamic sequencing based on reinforcement learning, and multi-agent coordination. By incorporating these features, adaptive real-time decision-making is possible, thereby increasing resilience in the face of stochastic warehouse traffic.

5.6.2 Complex Scenario: Multiple Bot and Docking Station

The study is concerned with the impact of narrow and wide aisle arrangement on performance behavior with higher robot interaction and congestion.
The low mobility in narrow aisle settings causes an increase in the congestion near the docking stations and crossroads which then increases the value of makespan and the number of collisions. The length of execution of robots is longer because of high rates of waiting because of fixed opportunities of passing and access point. Still, it is observed that routing plans do not differ, but since layouts are narrow, the efficiency of their execution is affected by the intrinsic spatial limitations in a significant way. In contrast, wide aisle settings minimise the impact of congestion due to the ability to move the robots in parallel and easier access to the docking points. This leads to shorter makespan and a reduction in the delays in execution although the robots may run in parallel. The enhanced space flexibility eliminates the waiting time and promotes more effective tasks fulfillment.
On the whole, the results show that aisle size has a major impact on the performance of multi-robot docking. Although the narrow aisle designs increase the congestion and delays in the execution, the wide aisle layouts lead to the increased throughput and stability of operations, which makes layout design an important factor in the situation of dense and docking-intensive warehouses.

Chapter 6

Conclusion

6.1 Summary of Findings

This thesis investigated the warehouse order-picking and multi-robot routing problem under realistic operational constraints, with particular emphasis on congestion, one-way aisle restrictions, robot density, and depot configurations. A simulation-based benchmarking framework was developed to evaluate classical, heuristic, hybrid, and metaheuristic routing algorithms under identical experimental conditions in both single-depot (low robot density) and multi-depot (high robot density) warehouse scenarios. Performance was assessed using multiple metrics, including path quality, planning time, memory usage, collision frequency, wait time, and overall makespan.

The experimental results demonstrate that exact algorithms such as Held–Karp are capable of producing optimal solutions for small problem instances; however, their computational cost and memory requirements render them impractical for larger or real-time scenarios. For example, Held-Karp required 1245.24 ms of mean planning time and 0.89 MB of memory, whereas Hybrid NN2opt achieved near-optimal performance with only 7.68 ms and 0.10 MB—using 89% less memory. As problem size and robot density increase, heuristic approaches such as NN2opt provide significantly faster planning with substantially lower computational overhead, with reduced robustness under congested conditions. Hybrid NN2opt maintained a 74.1% optimization rate—closely matching Held-Karp's 76.4%—while reducing collision counts by up to 100% compared to NN2opt and Genetic Algorithm in multi-robot scenarios with 10–15 robots.

Hybrid NN2opt demonstrated stable and balanced performance in both setups, achieving consistent path improvement with linear scaling in planning time and low, predictable collision counts. Whereas, NN2opt is more prone to makespan and congestion sensitivity as the density of the robots increases and Genetic Algorithms sometimes result in shorter makespan at the expense of higher collision variability and instability, Hybrid NN2opt will incur lower and more stable collision counts and reduced wait times, in addition to predictable makespan behavior. Collision vs- makespan analysis further reiterates that Hybrid NN2opt has a good trade-off between the execution efficiency and safety, and the results cluster around moderate makespan values and have less conflicts even when traffic is dense. The last design modifications, especially the inclusion of a congestion-weighted cost function and lightweight local replanning functions, were important in changing the routing

structure of the system to a more congestion-sensitive and execution-resilient one. The refinements enabled the robots to allocate traffic both spatially and temporally which minimized the path conflicts without creating hard constraints and an unreasonable computational cost. Consequently, Hybrid NN2opt showed reliable performance in different densities and depot configurations of robots, which is why it is appropriate to deal with the real-life warehouse functioning

6.2 Contributions to the Field

The research has a number of significant implications to the multi-robot routing and optimization of a warehouse. An integrated system of simulation and benchmarking was created in order to measure routing algorithms with realistic operational limitations, such as congestion, one-way aisle limitations, robot density variation, and depot setup. Within the given framework, the exact, heuristic, hybrid and metaheuristic algorithms were thoroughly compared on the same experimental basis, and it was clearly demonstrated how all the trade-offs were made between the solution optimality, scalability, and robustness on the one hand and the cost of computation on the other hand. It is shown that Hybrid NN2opt is the most viable and implementable routing system since it can always find almost optimal path quality with low and predictable planning times, controlled collision growth, and stable makespan performance under both single-depot and multi-depot cases. Moreover, the paper stresses the fact that it is extremely important to consider congestion awareness as an inherent part of routing decisions instead of viewing congestion as an after-process issue. Taken together, the discussed contributions strengthen the sustainability of hybrid heuristic-based solutions to real-time warehouse routing and create a solid basis of scalable, congestion-resilient autonomous warehouse systems.

6.3 Recommendations for Future Work

Although the suggested framework and algorithms have good results in a simulated single- and multi-depot setting, there are still extensions that can be included in future studies. The present research mainly applies to the case of reactive routing choices made without long-term learning and robots do not memorize the experience of past order executions. One of the opportunities is the implementation of the techniques of Machine Learning (ML) to provide adaptive and experience-based routing behavior. Robots would be able to remember learned patterns of recurrent congestions, predict imminent traffic jams, and dynamically adapt their routing decisions as time progress by using learning-based techniques like reinforcement learning or predictive models trained using past routing, congestion and collision data. This would enable the system to graduate handling of congestions to proactive congestion avoidance. Future work would also be to extend the framework to the extent of fully cooperative multi-agent coordination where communication and collective learning between robots will further enhance traffic management in conditions of high density

Bibliography

[1] M. Held and R. M. Karp, "A dynamic programming approach to sequencing problems," *Journal of the Society for Industrial and Applied mathematics*, vol. 10, no. 1, pp. 196–210, 1962.

[2] M. Dorigo, A. Colorni, and V. Maniezzo, "Distributed optimization by ant colonies," in *Proceedings of the first European conference on artificial life*, 1991, pp. 134–142.

[3] M. Dorigo and G. Di Caro, "Ant colony optimization: A new meta-heuristic," in *Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406)*, IEEE, vol. 2, 1999, pp. 1470–1477.

[4] R. De Koster, T. Le-Duc, and K. J. Roodbergen, "Design and control of warehouse order picking: A literature review," *European Journal of Operational Research*, vol. 182, no. 2, pp. 481–501, 2006. DOI: 10.1016/j.ejor.2006.07.009.

[5] Y. Gong and R. De Koster, "A polling-based dynamic order picking system for online retailers," *IIE Transactions*, vol. 40, no. 11, pp. 1070–1082, 2008. DOI: 10.1080/07408170802167670.

[6] B. Korte and J. Vygen, *Combinatorial optimization: theory and algorithms*. Springer, 2008.

[7] K. J. Roodbergen and I. F. Vis, "A survey of literature on automated storage and retrieval systems," *European Journal of Operational Research*, vol. 194, no. 2, pp. 343–362, 2008. DOI: 10.1016/j.ejor.2008.01.038.

[8] D. L. Applegate, R. E. Bixby, V. Chvátal, and W. J. Cook, "The traveling salesman problem: A computational study," in *The Traveling Salesman Problem*, Princeton university press, 2011.

[9] O. Kulak, Y. Sahin, and M. E. Taner, "Joint order batching and picker routing in single and multiple-cross-aisle warehouses using cluster-based tabu search algorithms," *Flexible Services and Manufacturing Journal*, vol. 24, no. 1, pp. 52–80, Mar. 2012, ISSN: 1936-6590. DOI: 10.1007/s10696-011-9101-8. [Online]. Available: https://doi.org/10.1007/s10696-011-9101-8.

[10] F. Chen, H. Wang, C. Qi, and Y. Xie, "An ant colony optimization routing algorithm for two order pickers with congestion consideration," *Computers & Industrial Engineering*, vol. 66, no. 1, pp. 77–85, 2013. DOI: 10.1016/j.cie.2013.06.013.

[11] T. Öncan, "A genetic algorithm for the order batching problem in low-level picker-to-part warehouse systems," in *Proceedings of the International Multi-Conference of Engineers and Computer Scientists*, vol. 1, 2013.

[12] J. O. Ong and D. T. Joseph, "A review of order picking improvement methods," *J@ ti Undip: Jurnal Teknik Industri*, vol. 9, no. 3, pp. 135–138, 2014.

[13] S. Mirjalili, "The ant lion optimizer," *Advances in engineering software*, vol. 83, pp. 80–98, 2015.

[14] U. A. Umar, M. Ariffin, N. Ismail, and S. Tang, "Hybrid multiobjective genetic algorithms for integrated dynamic scheduling and routing of jobs and automated-guided vehicle (agv) in flexible manufacturing systems (fms) environment," *The International Journal of Advanced Manufacturing Technology*, vol. 81, no. 9, pp. 2123–2141, 2015.

[15] F. Chen, H. Wang, Y. Xie, and C. Qi, "An aco-based online routing method for multiple order pickers with congestion consideration in warehouse," *Journal of Intelligent Manufacturing*, vol. 27, no. 2, pp. 389–408, 2016.

[16] L. Holden, "Inventory optimization using a simpy simulation model," M.S. thesis, East Tennessee State University, 2017.

[17] T.-H. S. Li et al., "A three-dimensional adaptive pso-based packing algorithm for an iot-based automated e-fulfillment packaging system," *IEEE Access*, vol. 5, pp. 9188–9205, 2017. DOI: 10.1109/ACCESS.2017.2702715.

[18] N. A. M. Nordin, M. Omar, and S. S. R. Sharif, "Comparison of dijkstra's algorithm and dynamic programming method in finding shortest path for order picker in a warehouse," in *AIP Conference Proceedings*, vol. 1830, 2017, p. 020 024. DOI: 10.1063/1.4980887.

[19] R. De Santis, R. Montanari, G. Vignali, and E. Bottani, "An adapted ant colony optimization algorithm for the minimization of the travel distance of pickers in manual warehouses," *European Journal of Operational Research*, vol. 267, no. 1, pp. 120–137, 2018.

[20] J. Hvězda, T. Rybecký, M. Kulich, and L. Přeučil, "Context-aware route planning for automated warehouses," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, Maui, HI, USA, 2018, pp. 2955–2960. DOI: 10.1109/ITSC.2018.8569712.

[21] J. A. Cano, "Order picking optimization based on a picker routing heuristic: Minimizing total traveled distance in warehouses," in *Handbook of research on the applications of international transportation and logistics for world trade*, IGI Global Scientific Publishing, 2020, pp. 74–96.

[22] X. Ma and et al., "A parallel ant colony optimization algorithm based on fine-grained model with gpu acceleration," *Complexity*, vol. 2020, 2020. DOI: 10.1155/2020/5287189.

[23] M. Wang, R.-Q. Zhang, and K. Fan, "Improving order-picking operation through efficient storage location assignment: A new approach," *Computers & Industrial Engineering*, vol. 139, p. 106 186, 2020.

[24] X.-S. Yang, *Nature-inspired optimization algorithms*. Academic Press, 2020.

[25] J. Yu et al., "A novel parallel ant colony optimization algorithm for warehouse path planning," *Journal of Control Science and Engineering*, vol. 2020, 2020. DOI: 10.1155/2020/5287189.

[26] H. Bhati, G. Suri, R. Kala, and G. C. Nandi, "Simulation aided anticipatory congestion avoidance for warehouses," in *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*, IEEE, 2022, pp. 2062–2067.

[27] U. EJAZ, S. M. ISLAM, A. SARKAR, and A. IMASHEV, "Intelligent multiagent robotics for warehouse automation and resilient supply chain infrastructure," 2022.

[28] S. Winkelhaus, M. Zhang, E. H. Grosse, and C. H. Glock, "Hybrid order picking: A simulation model of a joint manual and autonomous order picking system," *Computers Industrial Engineering*, vol. 167, p. 107 981, 2022, ISSN: 0360-8352. DOI: https://doi.org/10.1016/j.cie.2022.107981. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360835222000511.

[29] "An optimization approach for an order-picking warehouse: An empirical case," *Journal of Competitiveness*, 2023. DOI: 10.7441/joc.2023.04.09.

[30] S. Ketabi and et al., "A deep reinforcement learning framework for optimizing congestion control in data centers," in *2023 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2023. DOI: 10.1109/NOMS56928.2023.10154411.

[31] X. Wang, A. Sahin, and S. Bhattacharya, "Coordination-free multi-robot path planning for congestion reduction using topological reasoning," *Journal of Intelligent & Robotic Systems*, vol. 108, no. 3, p. 50, 2023.

[32] "A systematic literature review: Design and control of warehouse order picking in development of e-commerce transaction," *International Journal of Innovation in Enterprise System*, vol. 8, no. 1, 2024. DOI: 10.25124/ijies.v8i01.275.

[33] B. Dunn and K. Charkhgard, "A modified algorithm for optimal picker routing in a single block warehouse," *arXiv preprint arXiv:2409.13219*, 2024. [Online]. Available: https://arxiv.org/abs/2409.13219.

[34] "SINS_AR: An efficient smart indoor navigation system based on augmented reality," *IEEE Access*, vol. 12, 2024. DOI: 10.1109/ACCESS.2024.3463531.

[35] Y. Wang and B. Wang, "Hybrid ga-aco algorithm for optimizing transportation path of port container cargo," *Informatica*, vol. 48, no. 20, 2024.

[36] J. Wu and et al., "A hybrid path planning algorithm combining a* and improved ant colony optimization with dynamic window approach for enhancing energy efficiency in warehouse environments," *PeerJ Computer Science*, vol. 10, e2629, 2024. DOI: 10.7717/peerj-cs.2629.

[37] D. Zinoviev, "Discrete event simulation: It's easy with simpy!" *arXiv preprint arXiv:2405.01562*, 2024.

[38] A. Dogru, A. D. Alamdari, D. Balpınarlı, and R. Aydogan, "A multitier approach for dynamic and partially observable multiagent path-finding," in *Proceedings of the 17th International Conference on Agents and Artificial Intelligence*, 2025.

[39] S. Mahmoudinazlou, A. Sobhanan, H. Charkhgard, A. Eshragh, and G. Dunn, "Deep reinforcement learning for dynamic order picking in warehouse operations," *Computers & Operations Research*, vol. 182, p. 107 112, 2025. DOI: 10.1016/j.cor.2025.107112.

[40] Y. Zhang and J. Liang, "Multi-dimensional agv path planning in 3d warehouses using neural adaptive heuristic ant colony optimization," *arXiv preprint arXiv:2504.01985*, 2025. [Online]. Available: https://arxiv.org/abs/2504.01985.

[41] J.-Q. Chen, K. Zhang, R. Zheng, and Y. Zhong, "Simllm: Fine-tuning code llms for simpy-based queueing system simulation," *arXiv preprint arXiv:2601.06543*, 2026.