-

# Adaptive Storage Strategy and Multi-Agent Path Finding for Robotic Fulfillment Centers

*Jeffrey Ryu*

The Graduate School
School of Electrical and Electronic Engineering
Yonsei University

December 2024

# Adaptive Storage Strategy and Multi-Agent Path Finding for Robotic Fulfillment Centers

A Master's Thesis
Submitted to the Department of Electrical and Electronic Engineering
and the Graduate School of Yonsei University
in partial fulfillment of the
requirements for the degree of
Master of engineering

Jeffrey Ryu

December 2024

This certifies that the master's thesis of
Jeffrey Ryu is approved.

Thesis Supervisor: DaeEun Kim

Euntai Kim

Dongjun Shin

The Graduate School
School of Electrical and Electronic Engineering
Yonsei University
December 2024

# Abstract

This thesis presents two novel approaches for improving efficiency in robotic mobile fulfillment centers: an Agent Density Based Congestion Detection system for path planning and an Adaptive Stochastic Class Based Storage (AS-CBS) strategy for dynamic storage optimization. The congestion detection system utilizes spatial convolution operations with varying kernel sizes and thresholds to analyze local robot densities, dynamically adjusting A* pathfinding costs to enable proactive congestion avoidance. Simultaneously, AS-CBS implements a queue-based system to track real-time demands of incoming items, creating adaptive storage patterns that optimize access to high-demand items while maintaining operational flexibility. Through extensive simulations in a 29 by 32 grid environment with 20 robots, both systems demonstrated consistent performance improvements across different configurations. The congestion detection system achieved a 26-percent reduction in collision frequency while maintaining 25-percent faster completion times, while AS-CBS showed remarkable adaptiveness to shifting demand patterns while improving task completion rates. Physical validation experiments using seven robots on a 6 by 6 grid demonstrated the real-world feasibility of both approaches, though with some implementation challenges due to scale limitations. While the robotic experiments revealed gaps between simulation and physical implementation, they provided valuable insights into practical considerations for future deployment. This research contributes to the advancement of warehouse automation by addressing both path planning and storage optimization, demonstrating balanced improvements in efficiency and operational stability.

**Key words :** Multi Agent Path Finding, Warehouse, Swarm Robotics, Multi Agent System, Robotic Fulfillment, Storage

# Acknowledgements

First and foremost, I would like to express my sincere gratitude to my advisor, Professor DaeEun Kim, for his continuous support, patience, and guidance throughout my graduate studies. His deep insights and encouragement have been invaluable to my research and academic growth.

I am deeply grateful to my thesis committee members for their time, constructive feedback, and expertise that helped shape this research. Their valuable suggestions have significantly improved the quality of this work.

Special thanks to my fellow laboratory members at the Biological Cybernetics Lab for their collaboration, stimulating discussions, and friendship. The countless hours we spent troubleshooting experiments and sharing ideas have made this journey both enriching and memorable.

I would also like to acknowledge the support of the School of Electrical and Electronic Engineering at Yonsei University for providing the resources and facilities necessary for conducting this research.

Finally, I want to express my heartfelt appreciation to my family for their unwavering support, understanding, and encouragement throughout my academic journey. Their belief in me has been a constant source of motivation.

# Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Jeffrey Ryu*)

# Table of Contents

**List of Abbreviations** **xiii**

**1 Introduction** **1**

1.1 Motivation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

1.2 Organization . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4

**2 Background** **7**

2.1 Storage Strategy . . . . . . . . . . . . . . . . . . . . . . . . . . 10

2.1.1 Random Storage . . . . . . . . . . . . . . . . . . . . . . 11

2.1.2 Class Based Storage . . . . . . . . . . . . . . . . . . . . 12

2.1.3 Storage Layouts . . . . . . . . . . . . . . . . . . . . . . 15

2.2 Routing Strategy . . . . . . . . . . . . . . . . . . . . . . . . . . 18

2.2.1 Difficulty of Routing . . . . . . . . . . . . . . . . . . . . 18

2.2.2 Commonly Used Strategies . . . . . . . . . . . . . . . . . 19

2.3 Multi Agent Path Finding . . . . . . . . . . . . . . . . . . . . . . 22

**3 Gait-Based Human Identification Using iPhone Video and Skeleton Analysis** **25**

3.1 Method . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 27

3.1.1 Storage Methods . . . . . . . . . . . . . . . . . . . . . . 28

3.1.2 Path Planning . . . . . . . . . . . . . . . . . . . . . . . . 30

3.1.3 Simulation Environment . . . . . . . . . . . . . . . . . . 33

3.2 Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 35

3.2.1 Random Storage . . . . . . . . . . . . . . . . . . . . . . 35

3.2.2 Traditional Storage . . . . . . . . . . . . . . . . . . . . . 39

3.2.3 Class Based Storage . . . . . . . . . . . . . . . . . . . . 42

3.3 Summary of Chapter 3 . . . . . . . . . . . . . . . . . . . . . . . 47

**4 Adaptive Stochastic Class Based Storage** **49**

4.1 Method . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 50

4.2 Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 56

4.3 Summary of Chapter 4 . . . . . . . . . . . . . . . . . . . . . . 67

**5 Robotic Application of Congestion Detection and Adaptive Storage Strategy** **71**

5.1 Method . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 72

5.2 Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 78

5.3 Summary of Chapter 5 . . . . . . . . . . . . . . . . . . . . . . 86

**6 Conclusion** **89**

6.1 Congestion Detection Based on Convolutional Filter . . . . . . . . . 89

6.2 Adaptive Stochastic Class Based Storage . . . . . . . . . . . . . . . 90

6.3 Robotic Application of Congestion Detection and Adaptive Storage Strategy . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 91

6.4 Future Works . . . . . . . . . . . . . . . . . . . . . . . . . . 92

**Bibliography** **94**

# List of Figures


2.1 A picture of traditional pallet stacker (Rücker et al., 2020). . . . . . . 9

2.2 Operation of a pod/shelf lifting warehouse AGV (Allgor et al., 2023). 10

2.3 Optimized storage layouts. (a) Petersen's storage layout assumption. (b) Different Volume Based Storage policies. Figures from (Petersen and Schmenner, 1999). . . . . . . . . . . . . . . . . . . . . . . . . 13

2.4 Storage structure of a warehouse. (a) Two-lane aisle design. (b) One-lane intersection design (Figures from (Roser, 2021)). . . . . . . . . . 15

2.5 Optimized cross-aisles in Fishbone Shape. (a) 3D view of a Fish Bone layout (b) Fish Bone layout implementation in a business warehouse (Figures from (Meller and Gue, 2009)). . . . . . . . . . . . . . . . . 16

2.6 Storage layout incorporating cross-aisles. (a) Long aisle without cross-aisle. (b) Long aisle with horizontal cross-aisle. (c) Horizontal long aisle with vertical cross-aisle. Figure reproduction of Pohl et al (Pohl et al., 2009). . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

2.7 Possible movements between a storage point to another (a) Fish bone design. (b) Traditional long aisle design (Figures from (Meller and Gue, 2009)). . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17

2.8 Different routing strategies for parallel aisles layout (Roodbergen, 2001). 20

2.9 Five types of conflicts. (a) Edge Conflict. (b) Vertex Conflict. (c) Following Conflict. (d) Cycling Conflict. (e) Swapping Conflict. (Stern et al., 2019). . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 22

2.10 Grid representation of warehouse MAPF problem. (a) A Routing and Task-Allocation Algorithm for Robotic Groups in Warehouse Environments (Chatzisavvas et al., 2022). (b) Mobile Robot Path Optimization Technique Based on Reinforcement Learning Algorithm in Warehouse Environment (Lee and Jeong, 2021). (c) Rapid Randomized Restarts for Multi-Agent Path Finding: Preliminary Results (Cohen et al., 2018). (d) Experimenting with robotic intra-logistics domains (Gebser et al., 2018). . . . . . . . . . . . . . . . . . . . . . . . . 24

3.1 Capture of a picking operation with 50 agents and traditionally organized racks. (a) Location of agents on the field. Light blue represent the storage racks and the yellow dots represent the agents. (b) Visualization of congestion map grid. Congestion identified by the convolution filter are marked by the corresponding color to the kernel sizes, light blue (2), yellow(3), and red color(5). . . . . . . . . . . . . . . . . . . . 30

3.2 Simulation running with 15 agents. Racks of different types are distinguished by their color. (a) Racks are randomly dispersed across the field. (b) Racks of the same kind are organized in their designated storage area. (c) Racks are organized by Manhattan distances from the stations. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 33

3.3 Heat maps representing the agent activity (a, b, c) and congestion (d, e, f) across the field with no path planning. (a,d) Random Storage. (b,e) Traditionally Organized Storage. (c,f) CBS Storage. . . . . . . . . . . 35

3.4 Heat maps representing the agent activity (a, b, c) and congestion (d, e, f) across the field with no path planning. (a,d) Random Storage. (b,e) Traditionally Organized Storage. (c,f) CBS Storage. . . . . . . . . . . 36

3.5 ANOVA of total volumes for random storage configuration. (a) Uniform item distribution (b) Biased item distributions. . . . . . . . . . . 36

3.6 ANOVA of total collisions for random storage configuration. (a) Uniform item distribution (b) Biased item distributions. . . . . . . . . . . 37

3.7 ANOVA of total congestion counts for random storage configuration. (a) Uniform item distribution (b) Biased item distributions. . . . . . . 38

3.8 ANOVA of total volumes for traditional storage configuration. (a) Uniform item distribution (b) Biased item distributions. . . . . . . . . . . 40

3.9 ANOVA of total collisions for traditional storage configuration. (a) Uniform item distribution (b) Biased item distributions. . . . . . . . . 40
3.10 ANOVA of total congestion counts for traditional storage configuration. (a) Uniform item distribution (b) Biased item distributions. . . . 41
3.11 Duration of non-movement of picking agents over time. The scatter plots are binned into intervals, then histograms were drawn with respect to Update Cycle and Duration. (a) No path planning. (b) Agent density based path planning. . . . . . . . . . . . . . . . . . . . . . . . 43
3.12 ANOVA of total volumes for CBS configuration. (a) Uniform item distribution (b) Biased item distributions. . . . . . . . . . . . . . . . . . 44
3.13 ANOVA of total collisions for CBS configuration. (a) Uniform item distribution (b) Biased item distributions. . . . . . . . . . . . . . . . 44
3.14 ANOVA of total congestion counts for CBS configuration. (a) Uniform item distribution (b) Biased item distributions. . . . . . . . . . . . . . 45
3.15 Volume plotted against Congestion and Collision plotted against Collision to discover the relationship (**A, C**) uniform distribution, (**B, D**) biased distribution. . . . . . . . . . . . . . . . . . . . . . . . . 46

4.1 Simulation running with 20 agents in a randomly scattered storage. Racks of different classes are distinguished by their color, and the agents are numbered with the blue line indicating their path to the destination. Magenta marks the grid cell the agent. . . . . . . . . . . . 53
4.2 Flowchart of a picking agent behavior. . . . . . . . . . . . . . . . . . 54
4.3 Activity heat maps across storage methods (red: high, blue: low) under uneven Pareto's turnover rates. (a) Random. (b) Fixed CBS. (c) N-CBS. (d) AS-CBS. . . . . . . . . . . . . . . . . . . . . . . . . . . 56
4.4 Activity heat maps across storage methods (red: high, blue: low) under dynamic Pareto's turnover rates. (a) Random. (b) Fixed CBS. (c) N-CBS. (d) AS-CBS. . . . . . . . . . . . . . . . . . . . . . . . . . . 57
4.5 Simulation running with 20 agents in a different CBS strategies to fill up an empty storage. (a) Fixed CBS. (b) N-CBS. (c) AS-CBS. . . . . 58
4.6 AS-CBS queue over time during uneven distributed turn over rates, when (a) the distribution is kept constant, (b) distribution is flipped mid-point. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 59

4.7 Box plots as a result of a one-way ANOVA of distances of each class from the nearest P/D station for AS-CBS methods: (a) Uniform Demands. (b) Pareto's Demands. . . . . . . . . . . . . . . . . . . . . . 59
4.8 Reorganizing behavior of Fixed CBS method. (a) Initial state. (b) Final state. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 60
4.9 Reorganizing behavior of N-CBS method. (a) Initial state. (b) Final state. 60
4.10 Reorganizing behavior of AS-CBS method in Figure 4.6 LaTeX Error: Can be used only in preambleSee the LaTeX manual or LaTeX Companion for explanation.Your command was ignored.Type I ¡command¿ ¡return¿ to replace it with another command,or ¡return¿ to continue without it.4.6(b)'s simulation. (a) Reorganization. (b) Reorganization after the demand flip. . . . . . . . . . . . . . . . . . . . . . . . . 61
4.11 ANOVA comparison of processed volumes across storage types with uniform turnover rates: (a) Box plots. (b) Pairwise comparisons. . . . 62
4.12 ANOVA comparison of processed volumes across storage types with Pareto's turnover rates: (a) Box plots. (b) Pairwise comparisons. . . . 63
4.13 ANOVA comparison of processed volumes across storage types with dynamic Pareto's turnover rates: (a) Box plots. (b) Pairwise comparisons. 64
4.14 ANOVA comparison of collision and congestion counts across storage types with uniform turnover rates: (a) Collision box plots. (b) Collision pairwise comparisons. (c) Congestion box plots. (d) Congestion pairwise comparisons. . . . . . . . . . . . . . . . . . . . . . . . . 66
4.15 ANOVA comparison of collision and congestion counts across storage types with Pareto's turnover rates: (a) Collision box plots. (b) Collision pairwise comparisons. (c) Congestion box plots. (d) Congestion pairwise comparisons. . . . . . . . . . . . . . . . . . . . . . . . . 67
4.16 ANOVA comparison of collision and congestion counts across storage types with dynamic Pareto's turnover rates: (a) Collision box plots. (b) Collision pairwise comparisons. (c) Congestion box plots. (d) Congestion pairwise comparisons. . . . . . . . . . . . . . . . . . . . . . 68

5.1 Schematic of the experiment setup. . . . . . . . . . . . . . . . . . . . 72
5.2 Placement of items on the six-by-six grid. (a) Randomly placed items. (b) Items re-organized according to AS-CBS. . . . . . . . . . . . . . 75
5.3 Images of the Robot Experiment Environment. . . . . . . . . . . . . 76

5.4 Line following grid environment with four Zumo Robots. Their paths to their respective destination are drawn. . . . . . . . . . . . . . . . . . 77
5.5 Movement of robots to their destination. Purple and green reach their destination at (c), yellow reaches its destination at (d), and blue reaches its destination at (f). . . . . . . . . . . . . . . . . . . . . . . . . . . . 77
5.6 Bar plot of collision counts marked blue and completion counts marked red for four robots. (a) Without congestion avoidance. (b) With convolutional congestion avoidance. . . . . . . . . . . . . . . . . . . . . . . 78
5.7 Bar plot of collision counts marked blue and completion counts marked red for an experiment with three robots. . . . . . . . . . . . . . . . . . 79
5.8 Boxplots for larger scale six-by-six grid with seven robots with different algorithms applied. (a) Volume, or number of path planning completion. (b) Number of collisions. . . . . . . . . . . . . . . . . . . . . . . 80
5.9 Boxplots for six-by-six grid with seven robots with different algorithms applied. (a) Collisions per minute. (b) Interval between completing a path. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 82
5.10 Placement of items on the six-by-six grid. (a) Randomly placed items. (b) Items re-organized according to AS-CBS. . . . . . . . . . . . . . 83
5.11 Boxplots for larger scale six-by-six grid with seven robots with different storage strategies applied. (a) Volume, or number of cycle completion. (b) Number of collisions. . . . . . . . . . . . . . . . . . . . . . . 84
5.12 Boxplots comparing the Random and AS-CBS method with different metrics. (a) Collision per Minute. (b) Time to First Completion. . . . . 85

# List of Tables


3.1 Mean Collision, Congestion, and Volume counts for each path planning algorithm when randomly stored with uniform item demands. . . 38

3.2 Mean Collision, Congestion, and Volume counts for each path planning algorithm when randomly stored with biased item demands. . . . 38

3.3 Mean Collision, Congestion, and Volume counts for each path planning algorithm when traditionally stored with uniform item demands. 41

3.4 Mean Collision, Congestion, and Volume counts for each path planning algorithm when traditionally stored with biased item demands. . 41

3.5 Mean Collision, Congestion, and Volume counts for each path planning algorithm when stored according to CBS with uniform item demands. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 42

3.6 Mean Collision, Congestion, and Volume counts for each path planning algorithm when stored according to CBS with biased item demands. 44

4.1 Mean Volume and Standard Deviation for Different Storage Methods under Various Demand Patterns . . . . . . . . . . . . . . . . . . . . . 62

4.2 Mean Collision Count and Standard Deviation for Different Storage Methods under Various Demand Patterns . . . . . . . . . . . . . . . . 65

4.3 Mean Congestion Count and Standard Deviation for Different Storage Methods under Various Demand Patterns . . . . . . . . . . . . . . . . 65

5.1 Table of average time (in seconds) in between each collision (first row) and in between reaching a destination (second row) for default configuration. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 79

5.2 Table of average time (in seconds) in between each collision (first row) and in between reaching a destination (second row) for congestion avoidance configuration. . . . . . . . . . . . . . . . . . . . . . . . 80

5.3 Comparison of different algorithms, mean and standard deviation of collision and number of completion (volume). . . . . . . . . . . . . . . 81
5.4 Performance Metrics of Different Path Planning Methods. Best values are in bold. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 82
5.5 Comparison of different algorithms, mean and standard deviation of collision and number of completion (volume). . . . . . . . . . . . . . . 84
5.6 Performance Comparison between Random and AS-CBS Methods with regards to time. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 85

# List of Abbreviations

| | |
|---|---|
| AGV | Automated Guided Vehicles |
| AMR | Autonomous Mobile Robots |
| ANOVA | Analysis of Variance |
| AS-CBS | Adaptive Stochastic Class-Based Storage |
| AS/RS | Automated Storage and Retrieval Systems |
| CBS | Class-Based Storage |
| GTP | Goods to Person |
| HRC | Human-Robot Collaboration |
| LRA* | Local Repair A* |
| MAPF | Multi-Agent Path Finding |
| N-CBS | Normal Distribution Class-Based Storage |
| P/D Point | Pick and Delivery Point |
| RMFS | Robotic Mobile Fulfillment System |
| SKU | Stock Keeping Unit |
| SLAM | Simultaneous Localization and Mapping |
| TSP | Travelling Salesman Problem |
| VBS | Volume Based Storage |
| WHCA* | Windowed Hierarchical Cooperative A* |
| WMS | Warehouse Management System |

# Chapter 1

# Introduction

Amazon.com's notable example of successful design of Automated Storage and Retrieval Systems (AS/RS) motivated many businesses to follow its treads to implement similar systems. These businesses often do not fathom the complexity of constructing such system, and hasty implementation often introduced bottle necks and jams, causing more losses then benefits.

A notable example showcasing the difficulty of large scale AS/RS is Denver International Airport (DIA)'s 193 million dollar ambitious vision of automated baggage handling system, which shared some core discipline with warehouse management system, identifying and routing the correct baggage to their destination akin to routing stocks into a storage location within a warehouse.

This system consisted of 4000 carts and 21 miles of subterranean network interlinked into a network of more than 100 computers and scanners, anticipated to be one of the most advanced system in the world. It costed 560 million dollars and 16 months for engineers to fix the system and additional costs to hastily construct a manual baggage handling system.

Despite the time and funds poured into the system, the inherent design flaws hindered reliable operation, eventually scrapping the entire project in 2005 (Gibbs, 1994; Ltd, 2008). While DIA's example pertains to the difficulty of coordination and routing, another significant challenge of managing a large scale logistics is the storage of items.

Traditionally, same items were stored and organized together at a dedicated area, and this seems to be the most intuitive way of managing multiple articles of items. It is also easier for human operators to locate and manage items without the use of computers. Yet, this method is disadvantageous in space management, as the dedicated sufficiently large storage space for each items must be maintained.

Conversely, if the volume of an article of items exceed the dedicated storage space, the excess items must be stored somewhere else, deteriorating the efficiency of item management. In large scale industrial settings, it is quite usual to encounter situations where certain items exceed or falls short of anticipation, leading to added cost and inefficiency (de Koster et al., 2007).

Recently, the dynamic nature of order picking operations in warehouses introduces significant challenges regarding the efficient placement of various items to minimize travel distances. This aspect of warehouse management is crucial because the travel cost and time associated with picking activities can account for up to 75 percent of the entire operational cost and time (Tompkins et al., 2010; Riedel, 2011; Azadeh et al., 2019; Feng and Ye, 2021; Liu et al., 2021).

Efficient planning and coordination in this context are essential for optimizing overall performance. Several factors, including product placement strategies and routing algorithms, play a significant role in reducing these costs.

## 1.1 Motivation

In large operations such as Amazon.com, Inc., where their Fulfillment centers average 800,000 square feet with 185 locations globally, employing more than 100000 robots in their Robotic Mobile Fulfillment System (RMFS), these Fulfillment centers are located near population centers to minimize delivery time, and storage efficiency becomes a quintessential part of the operation to minimize the size of the facility as well as order-picking speed and efficiency (Amazon.com, 2021; Merschformann et al., 2019).

By the virtue of computerized stock-keeping systems, mobile robots can now accurately locate desired items within a warehouse. This technological advancement has led certain warehouses, such as Amazon Robotic Fulfillment Centers, to adopt random storage methods (Hausman et al., 1976; Petersen et al., 2004; Petersen and Schmenner, 1999). The computerized Stock Keeping Unit (SKU) management systems provide real time information of item locations to the order-picking robots, dedicated storage spaces for articles of items are no longer needed and the space can be utilized more efficiently (Zou et al., 2017).

As previously mentioned, dedicated storage spaces lack flexibility in utilizing storage spaces, while random storage method can simply put an item anywhere available. Most importantly, random storage offer decreased travelling expenses, because the picker is more likely to locate its desired item nearby regardless of its location.

Amazon.com, Inc.'s RMFS employ random storage method due to their wide range of items they store, and this method becomes more efficient as the variety of the items increase. In retrospect, the adoption of computerized storage management enabled location of individual items, and the random storage method not only maximize space utilization, but also optimizes picking operation by decreasing the travelling expenses (Malmborg and Al-Tassan, 2000; Furmans and Dehdari, 2014).

Conversely, operations that handle more predictable logistics may implement a more tailored approach such as the Class Based Storage (CBS) method, operating on the principle of categorizing items into classes based on their turnover rates, placing high-turnover items closer to the picking and delivery (p/d) station, thus reducing travel distances for frequently requested items.

Research by Gibson and Sharp (1992) has demonstrated that the CBS method can significantly shorten travel distances compared to random storage methods (Gibson and Sharp, 1992). This method enhances operational efficiency by strategically positioning items according to their demand levels, which optimizes the order-picking process and reduces the overall time and cost associated with warehouse operations.

Albeit a research suggests that a well-designed storage system can lead to up to a 60 percent improvement in travel distances, attempts at optimizing a order-picker's path planning have been made to further reduce the travelling costs (Caron et al., 2000).

For example, Öztürkoğlu (2010) and Zhou (2022) suggested a novel storage strategy and its optimal routing technique, however, the design lacks consideration for multi agent picking, where the aisles leading to a singular P/D point may induce congestion (Zhou et al., 2022; Öztürkoğlu et al., 2012).

Accordingly, some warehouses implement a P/D station design to increase its surface area, handling multiple pickers to reduce bottle necking. Previous researches mainly explored routing and storage strategies, however, were limited assuming single picker carrying out a single operation (Petersen and Schmenner, 1999; Ratliff and Rosenthal, 1983).

Although there are routing strategies and warehouse designs that may optimize the travel expenses, the interaction between picking agents may introduce another layer of complexity in evaluating their effectiveness.

As noted by a survey by Roodbergen et al, the researches pertain to static environments, but solutions applicable for dynamic nature of the AS/RS requirements are increasingly more needed (Roodbergen and Vis, 2009).

## 1.2 Organization

To address the consequences of multi-agent picking operation, Chapter 3 proposes a routing heuristic that focuses on detection of congestion and avoidance of the said congestion by reconstructing a path. Based on the congestion detection, an agent may recognize where congestion have occurred or likely occur and re-route their path to the destination.

To minimize communication and computational demands for each agent, a central entity may manage the storage and the real time information of the agents and provide appropriate instruction based on them upon request. Such employment of would be most effective in a predetermined environment such as warehouses where the layout remain relatively static, and robustness in variety of settings is subordinate against other considerations.

Three methods of congestion mitigating path reconstruction were devised, the first based on the local density of the agents in the grid, where a convolution filter with varying thresholds and kernel sizes determine areas with congestion, the second utilizing the density of overlapping paths of the agents in the grid, generating a new path that avoids areas and corridors exceeding a set threshold, and finally, the third method generates a path that does not intersect with pre-existing paths.

To further add complexity, many businesses opt to widen their scope of service by treating broader variety of items and sometimes ones that require special care such as those that are time-sensitive. In such businesses, the RMFS require additional versatility and adaptability to rapidly changing stock keeping requirements, such as fluctuation of demands for certain items.

For example, it is evident that seasonal items will have different demands and subsequent turn-over rates by each month, and the optimized storage strategy, such as CBS, would consequently differ each season. While a complete random storage method eliminate the need to consider this issue, one might possibly further optimize the storage strategy by considering the temporal element in the CBS method by assigning different classes in response to the turn-over rates.

Chapter 4 presents a queue based classification method that utilizes recent order-picking operations to rank and assign classes to incoming orders, and additionally introduces a stochastic element to the class-based storage system, where the placement of items by class type follows a normal distribution.

In this system, items in classes with higher turnover rates are preferentially positioned closest to the picking and delivery (p/d) stations. However, items in classes with lower turnover rates are occasionally placed near the p/d stations as well. This approach ensures that picking agents benefit from shorter travel distances even when less frequently requested items are needed and aims to enhance flexibility and efficiency in warehouse operations.

The normal distribution model allows for a more dynamic and adaptive storage strategy that can better accommodate the unpredictable nature of item demand. This method not only reduces the likelihood of congestion in specific storage areas but also optimizes the overall travel distances for picking agents.

Consequently, the warehouse can maintain high performance and efficiency even when dealing with fluctuating turnover rates. Furthermore, this research will explore the implementation and impact of stochastic class-based storage through simulation.

By comparing the performance of this method against traditional storage systems, we aim to demonstrate its advantages in terms of reduced travel times, improved order-picking accuracy, and overall operational cost savings. The findings from this research could provide valuable insights for warehouse managers seeking to optimize their storage strategies in the face of dynamic demand patterns.

# Chapter 2

# Background

In recent years, the rapid growth of e-commerce has necessitated significant advancements in order processing and fulfillment systems. The integration of computerization and automation technologies has dramatically accelerated order processing speeds, leading to an unprecedented demand for faster and larger-scale order-picking operations.

A prime example of this trend is Amazon.com's Same-Day and Two-Day Delivery services in the United States, which exemplify the pinnacle of automation and stock management efficiency, enabling the timely delivery of even the most mundane items.

The seeming paradox of expedited delivery across vast distances is resolved through a strategic network of local warehouses. Similar to supermarket supply chains, products are not shipped across the continent for each order but are instead stored in proximity to potential customers.

Amazon.com's local fulfillment centers maintain extensive inventories of diverse products, including an expanding range of time-sensitive perishables. To manage the high volume and variety of orders efficiently, Amazon.com has developed innovative practices and heavily automated both their physical and managerial operations. Within the scope of this research, we focus primarily on the physical aspects of warehouse automation, specifically optimizing multi-agent control systems.

Key elements of this automation include Goods to Person (GTP) systems, Automated Storage and Retrieval Systems (AS/RS), and Autonomous Mobile Robots (AMRs). Goods to Person (GTP) technology forms a fundamental and essential component in enhancing picking operation speed for high-volume orders, particularly in warehouses with spatial constraints. In GTP systems, inventory items are transported to human operators rather than requiring workers to navigate the warehouse.

This approach significantly increases picking speed, minimizes human errors, and improves industrial safety by separating automated workspaces from human-occupied areas. Automated Storage and Retrieval Systems (AS/RS) represent a subset of GTP technology, where robotic elements further automate the storage and retrieval processes. These robotic components primarily consist of Automated Guided Vehicles (AGVs) and Autonomous Mobile Robots (AMRs).

While these terms may seem interchangeable, they denote distinct levels of autonomy and operational capability:

- **Automated Guided Vehicles (AGVs)** : AGVs possess limited autonomy and processing power, designed for simple, repetitive tasks. They navigate using predefined pathways, typically guided by markers on floors or walls, such as magnetic tapes or simple close-range communication beacons. AGVs generally have restricted obstacle avoidance capabilities and are usually designed to operate in environments segregated from human workers. These vehicles typically function within a more centralized control structure and are relatively cost-effective.

- **Autonomous Mobile Robots (AMRs)** : AMRs exhibit enhanced autonomy with independent sensing and navigation capabilities, necessitating more advanced computational capacity. They commonly employ onboard Simultaneous Localization and Mapping (SLAM) technology, enabling them to perform complex tasks and facilitate Human-Robot Collaboration (HRC). AMRs possess superior decision-making capabilities, allowing for a more decentralized operational structure, and offer greater versatility in their applications.

The implementation of these automated systems has revolutionized warehouse operations, significantly improving efficiency, accuracy, and scalability. As e-commerce continues to evolve, the role of these technologies in shaping the future of logistics and supply chain management cannot be overstated. Further research into the optimization and integration of these systems will be crucial in meeting the ever-increasing demands of the global marketplace.

Amazon Robotics, formerly known as Kiva Systems first commercialized automated warehouse system with GTP using AGVs, and sold their system to companies such as Walgreens and Staples. Many automated warehouse systems use AGVs than AMRs because streamlining the operation and the structure is possible. The SLAM

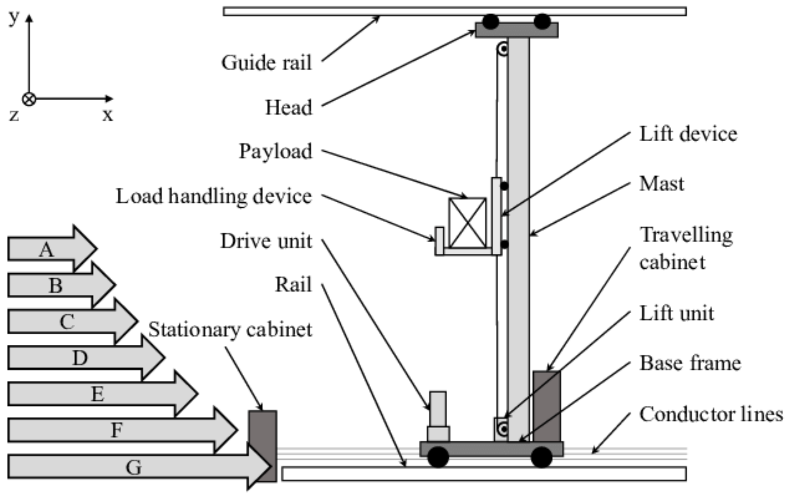



Figure 2.1: A picture of traditional pallet stacker (Rücker et al., 2020).

capabilities of AMRs are often not needed in such operations because the workspace is predefined with simple repetitive tasks, and unforeseen disturbances and hazards are unwarranted.

With the freedom of configuring AGVs and their operating environment, Kiva Systems further innovated warehouse automation by moving an entire shelf or a pod containing many items. At first, it may seem sub-optimal to carry an entire pod to look for a few, however, a picking station may not treat a single order but multiple, and because the pods carry random items, it is likely that it contains items for another order. This approach further enabled standardization of storage pods and space savings, coupled with storage designs facilitating AGV movement.

Traditionally, warehouses used stacker cranes instead of AGVs to move pallets along the aisles to store large volumes of items and utilize vertical space, however, the rigidity of this structure poses challenges in the contemporary rapid and dynamic order-picking environment.

AGVs are flexible and scalable, and a warehouse layout can be reconfigured easily, while being able to add more units if needed. This flexibility offers unmatched advantage when automating in a pre-existing structure, because less significant renovation is needed due to having a smaller profile.

Moreover, cranes require more work to reconfigure their routes and position, and can be more expensive for smaller warehouses and maintenance costs are lower when AGV units can individually be replaced or excluded from operation without disrupting the entire system. Using more individual units with freedom of movement offer redundancy and resilience to failure, because an operation along an entire aisle need to be

Figure 2.2: Operation of a pod/shelf lifting warehouse AGV (Allgor et al., 2023).

halted if a crane fails.

Still, the complexity and scale of order picking operations in RMFS centers, storage optimization and Multi-Agent Path Finding (MAPF) become quintessential components of reducing operational costs and quality of service.

One of the leaders in RMFS operation is Amazon.com, which has been bench marked by many other businesses. With its acquirement of Kiva Systems, Amazon Robotics division has been heavily investing into researching optimal storage strategies and robot systems tailored to their business model, and has made many innovative practices that now became industry standards such as random storage.

A warehouse layout optimized exclusively for their own AGVs combined with digital stock keeping system, Amazon.com effectively automated their order-picking operation while maximizing the storage efficiency.

## 2.1 Storage Strategy

As previously demonstrated by Caron et al, design of a storage dictate a large part in reducing travelling expense (Caron et al., 2000). The significance of travelling in order-picking operation entails an efficient storage method that accounts for the nature of stock articles as demonstrated by Amazon.com.

Traditionally, a dedicated storage method was preferred, with its structure akin to shopping in a supermarket. This method has been the choice for warehouse storage due to its ease of navigation and search for a human picker, but problems such as inefficient space utilization has always existed.

Furthermore, stock keeping and throughput is not always predictable, thus the dedication of storage space often end up becoming disorganized as business operates.

The introduction of Warehouse Management System (WMS) eliminated the need for operators to memorize the item's location and eventually led to the development of random storage method. The extensive and fluctuating inventory of their business led to the development of random storage method, however, it is possible that a more optimal solution exists when the inventory is smaller and more predictable.

For example, if certain items have higher demands, consequently the turn-over rate of those items will be higher at the warehouse. Therefore, one would reasonably be inclined to place such items at a more accessible location. Petersen suggested a Volume Based Storage method to address this issue and developed upon it to propose Class Based Storage to group and rank items by their turn-over rates for optimized placement.

In understanding warehouse layout, one of the deciding factors is the type of picking operation. there are two types of operations in a warehouse order-picking. One is a single command operation, where as the name suggests, a picker receives either a storage or retrieval order, and makes a travel per order, and the second is a dual command operation where a picker carries out both a storage and a retrieval order, storing an item from a p/d point then moving to an item location to retrieve and return to the p/d point.

### 2.1.1 Random Storage

A popular method of storage assignment is a random/chaotic storage method, where an empty storage space is chosen randomly for an incoming item to be stored (Petersen, 1997). This method's advantage lies in high utilization of space in contrast to traditional dedicated storage method where a space needs to be reserved for items regardless of low inventory, and conversely, sometimes need to be expanded to accept more items (Sharp et al., 1991; de Koster et al., 2007).

The advantage of the traditional storage against other methods is the ease of locating items for human operators and treatment of special items such as thermoregulated products, but the former advantage deteriorates when the stock keeping adapts Warehouse Management Systems (WMS) to digitally manage the location of the items. Such is why robotic fulfillment with the foundation of WMS only use dedicated storage space with limited application, while utilizing efficient storage strategy.

Within the random storage method there is another variation, storing the items in the nearest available location. At first, this may not seem like a random storage method, however, Hausman explained that if goods were stored and treated by pallets or pods, it performs similar to the random storage method (Hausman et al., 1976).

In Sharp's research, the travel distance reducing properties of random storage method appeared to be sub-optimal, and sometimes extended the distance, and Hausman also discovered that it lagged behind other methods such as the CBS (Sharp et al., 1991; Hausman et al., 1976).

To further develop the random storage policy, Han et al propose "nearest neighbor" strategy in deciding item to be retrieved after completing a storage in dual-command operations, by selecting from storage and retrieval points with minimum distance in between, resulting in around 5 to 8 percent range within optimal performance (Han et al., 1987; Johnson and Brandeau, 1996).

Nevertheless, an overlooked aspect in random storage method is the elimination of need for storage organization. In a dedicated storage method or in CBS, regular organization of items and updates to reflect demand patterns is required, which may cause additional resource consumption.

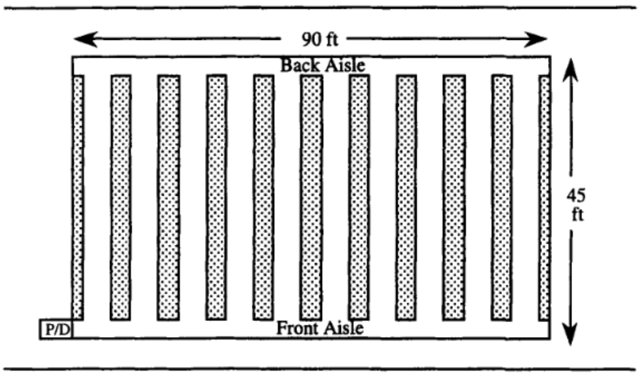
One may wonder why a random storage method increased the travel distance when items are scattered, but it may be due to increased average distance to every items. For any given pick, a chance of having the desired item close by exists, however, vice versa is also valid. This becomes more significant if the demands for items are uneven, because high demand items and low demand items are equally scattered around the warehouse while the other methods can strategically place from the p/d point to reduce the travel distance.

Another challenge posed by random storage method is its potential difficulty in routing, due to the scattered nature of goal points, and this is further exacerbated by the space saving efforts which is often applied in conjunction.

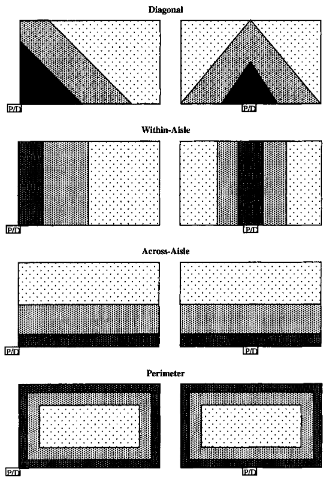
As mentioned earlier, use of singe lane layouts could save significant space, but reduce the freedom of movement. Regardless, random storage strategy may outweigh its disadvantages in contexts where demands of items rapidly change or remains uniform, inventory is extensive, and flexibility and space efficiency is the primary concern.

### 2.1.2 Class Based Storage

A Volume Based Storage (VBS) method was proposed by Petersen et al, where items of high order volume are placed closer to the p/d point (Petersen and Schmenner, 1999; Jarvis and McDowell, 1991). The aim of this method is to decrease the travel distances




(a)




(b)

Figure 2.3: Optimized storage layouts. (a) Petersen's storage layout assumption. (b) Different Volume Based Storage policies. Figures from (Petersen and Schmenner, 1999).

for items that are picked often, reducing the overall distance needed to travel.

Petersen studied four storage policies applicable to the volume based storage as shown at the top of Fig 2.3(b), with darker regions showing items with higher volume. The first one from the top, diagonal strategy was proposed by Gibson et al, showing high volume items placed within the closest Manhattan distance from the p/d point, and also illustrate how the shape varies by p/d location (Gibson and Sharp, 1992).

Conversely, Jarvis et al, devised a within-aisle plan, arranging the storage area by each aisle as seen in the second image in Fig 2.3(b). The third image of Fig 2.3(b) show across-aisle policy where items are stored closer to the p/d point for each aisle, and the fourth shows storage around the perimeter with high demand items outward.

Petersen evaluated the effectiveness of these four storage policies of their travel distances with different pick list sizes and various demand skewness by having 20 percent of the stocks having low, medium, or high demands. As a result of his experiment, Jarvis et al's within-aisle policy with a p/d point in the middle yielded the best results across all settings. This result is somewhat foreseeable from the geometry of Petersen's assumption, because the pickers are bound to moving along the aisles, thus placing items along them will be sensible.

Except for the diagonal policy, having the p/d at the middle was significantly better in all cases, but Petersen remarks that the difference narrows as the pick list becomes greater. In his research, he discovered that within-aisle policy resulted in 10 to 20 percent travel distance savings and middle p/d point saving up to 4 percent in comparison to the corner p/d (Petersen and Aase, 2004).

Petersen noted that within-aisle and diagonal policy concentrates high demand items within the first few aisles near the p/d point, therefore across-aisle and perimeter policy could result in outcomes when congestion and balanced space utilization is of concern (Petersen and Schmenner, 1999).

In further improving the order picking performance of the VBS has been evaluated, called Class Based Storage method. This method further fine tunes VBS by grouping items by their pick rate or turnover rate, and assigns storage areas closer to the p/d point for classes with higher demands.

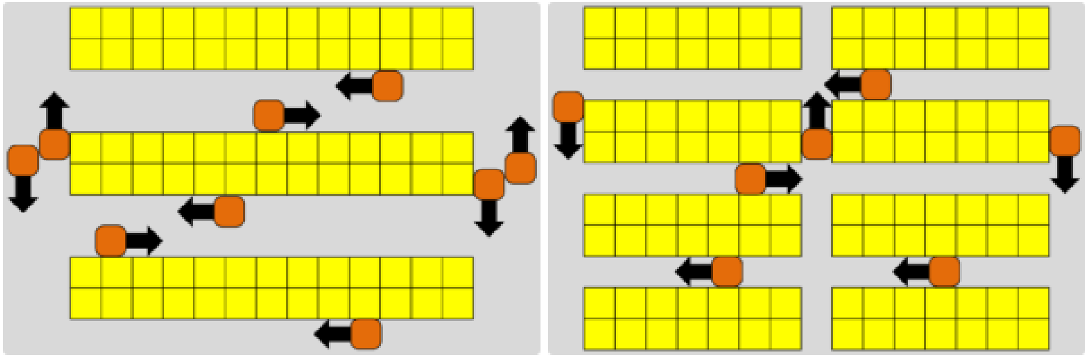
This is essentially a VBS but it simplifies the assignment process with aims to reduce administrative expenses such as analysis of demand patterns and item characteristics, while attaining the benefit of shortened travel distances. While relatively small number of classes yield good performance and Van den Berg and Gademann found that six classes resulted in optimal, several researches demonstrate that CBS approaches the performance of VBS proportionally to number of class division (Eynan and Rosenblatt, 1994; van den Berg and Gademann, 2000; Hausman et al., 1976).

While it may seem as increasing the number of classes is favorable, but the number of classes will eventually converge to number of items, rendering the new method meaningless. Contrarily, Petersen implemented partitioning strategies to divide the items into two classes, then assigning top 30 percent items into class 1 then the rest into class 2.

Similarly, he tested other proportions and concluded that 30-70 and 40-60 partitioning performed the best. The said partitioning performance resulted in 80 percent performance compared to the VBS, exhibiting the effectiveness while requiring less management and care (Petersen et al., 2004).

Indeed, ideal utilization of VBS policy would require ranking all item types by their demand and placing them accordingly, but its difficulty lies in realization when demands often fluctuate and assignment of storage space to wide articles of items become difficult. Another problem with a Class Based Storage is that it requires more space than random storage method to admit incoming items to their corresponding classes, thus more classes require more space (Graves et al., 1977).

Though, in attempts to undermine this ramification, optimal partitioning strategies have been established and generalized for rectangular layouts using one-dimensional search (Eynan and Rosenblatt, 1994; Rosenblatt and Eynan, 1989).



(a) (b)

Figure 2.4: Storage structure of a warehouse. (a) Two-lane aisle design. (b) One-lane intersection design (Figures from (Roser, 2021)).

Interestingly, Petersen argued in favor of within-aisle CBS, but Le-Duc's research contradict his results by demonstrating across-aisle CBS's close-to-optimal superiority, but then discovered that routing heuristics play a pivotal role doing so, along with pick list size (Le-Duc * and De Koster, 2005; Le-Duc, 2005).

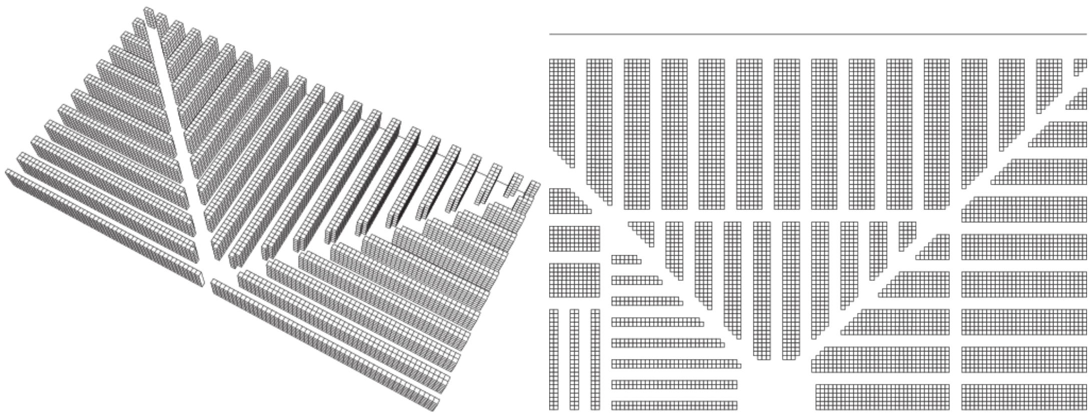
While classes can be assigned using various heuristics, one method is grouping items that are often demanded together in a same class (Frazele and Sharp, 1989). By using this method, the average distance to the paired item can decrease even if their demand differ, however, requires periodic analysis of the demand patterns.

Similarly to the implementation of ”nearest neighbor” policy in random storage method, Eynan and Rosenblatt applied this policy in CBS environment to discover significant increase in throughput (Eynan and Rosenblatt, 1993; Johnson and Brandeau, 1996).

### 2.1.3 Storage Layouts

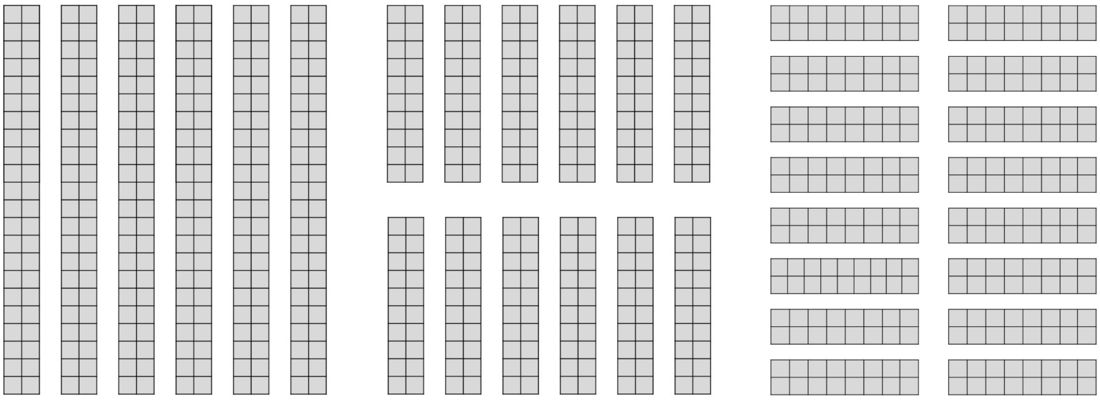
Traditional long aisles designs are most desirable for accessing items while maximizing storage space and is most intuitive for human operators to navigate through. Unlike traditional long aisle designs as demonstrated in Fig 2.4(a), businesses are starting to use single lane designs with intersecting longitudinal lanes for better accessibility.

The storage efficiency is typically below 50 percent with long two-way aisles, while the one-way method usually yield 40-60 percent storage efficiency depending on the number of cross aisles, and as a result, businesses such as Amazon.com employ one-way design to simultaneously maintain adequate traffic flow. To further increase the throughput, cross aisles are introduced, and they are especially useful when using



(a) (b)

Figure 2.5: Optimized cross-aisles in Fishbone Shape. (a) 3D view of a Fish Bone layout (b) Fish Bone layout implementation in a business warehouse (Figures from (Meller and Gue, 2009)).



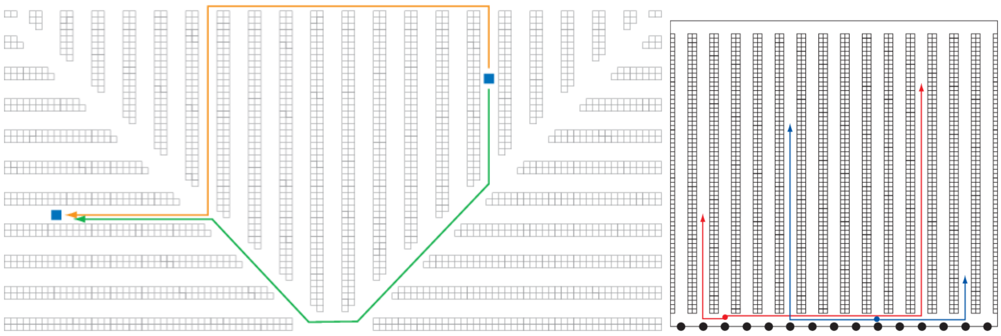
(a) (b) (c)

Figure 2.6: Storage layout incorporating cross-aisles. (a) Long aisle without cross-aisle. (b) Long aisle with horizontal cross-aisle. (c) Horizontal long aisle with vertical cross-aisle. Figure reproduction of Pohl et al (Pohl et al., 2009).

AGVs instead of cranes, because they are easier to move across aisles.

As a result, while cranes rarely have the capability to move perpendicularly to an aisle due to following rails to navigate, the compactness and freedom of movement enables dual command operations for AGVs to transition from storage to retrieval, thus utilizing cross aisles. Additionally, Roodbergen et al. showed that single command operations do not benefit from cross aisles due to increased travel distances (Roodbergen and de Koster, 2001).

How one should implement cross aisles has been a debated subject, but it should



(a) (b)

Figure 2.7: Possible movements between a storage point to another (a) Fish bone design. (b) Traditional long aisle design (Figures from (Meller and Gue, 2009)).

ultimately be dependent of the nature of operation, however for dual command operations, having a cross aisle is always resulted in better performance, and while layout C resulted in the best performance in most cases, it is more dependent on the assumption that the p/d point is a central singular position (Pohl et al., 2009).

The use of cross aisles maximize when a p/d point is not singular, because AGVs they increase the accessibility from different directions. The construction with aisles and cross aisles depend on the nature of the picking operation and the location of p/d points, and many warehouses that have a singular p/d point often use a ”Fish bone” design (Fig 2.5(a)), with aisles extending outwards from the p/d point (Meller and Gue, 2009).

In evaluating the fish bone layout, Gue notes its limitations that the layout assumes a single p/d point, single order operation, and random storage method (Gue and Meller, 2009).

Accordingly, the fish bone layout would benefit single command operations, where travel occur mostly between sections that benefit from shortened travel distances from a storage location and a p/d point. While the fish bone layout demonstrated up to 23.5 percent improvement in shortening the travel distances depending on the weight and height ratio of the warehouse, Fig 2.7 demonstrates the effect of long aisles in a fish bone layout and traditional layout, where moving from one point to another in the warehouse result in prolonged travel (Gue and Meller, 2009; Meller and Gue, 2009).

To address this issue, Pohl implemented additional cross aisles at the top of the fish bone layout to create a ”Fish Bone Triangle” layout with further improvements for dual command operations (Pohl et al., 2010).

Nonetheless, businesses with single p/d operation adopted the fish bone layout with cross aisles to facilitate movement across different aisles, because in reality, p/d point is not the only access point by workers and sometimes orders, thus necessitating additional cross aisles while sacrificing some storage efficiency.

To further improve upon the fish bone layout, Zhou et al implemented CBS to further utilize the improved accessibility. In his research, there has been up to 70 percent difference and minimum 2 percent difference in travel distance when CBS was adopted against the random storage method, indicating that the CBS method always yielded a positive result in reducing travel distances (Zhou et al., 2022).

This result extends from Petersen's study on effective class assignment in long aisle storage, highlighting the effectiveness of the CBS method in different warehouse layouts.

## 2.2 Routing Strategy

### 2.2.1 Difficulty of Routing

In an order picking situation, route planning becomes a complex and expensive problem especially for batch picking operations, because this problem can be reduce to a Travelling Salesman Problem (TSP). The challenge lies in finding the shortest cycle visiting all nodes, similarly, an order picker leaves the p/d point then needs to visit all points with the corresponding order then return to the p/d point, and shortening this as much as possible will reduce the cost of operation (Robinson, 1949). This problem is particularly challenging because it can be reduced into an NP-hard problem, where no known general solution that can solve in a polynomial time exists, meaning that it becomes explosively computationally consuming as the size of the pick-list increases (Karp, 1972). Thus, using approximation heuristics is a reasonable approach to solving a TSP, motivating researchers to apply them into warehouse picker routing (Theys et al., 2010). One of the earliest and simplest methods of approximation is the nearest neighbor heuristic, and Lin-Kernighan algorithm is often regarded as the best one.

- Nearest Neighbor Algorithm : Start from one of the nodes, connect to the nearest node and repeat until all nodes have been visited. This is a greedy heuristic, where the algorithm finds the best immediate solution for each step, without regards to the entire scheme.
  - Results in a 25 percent longer path on average compared to the shortest

possible path (Johnson and McGeoch, 2008).

  - Worst case performance yields $O(N^2)$, and worst case space complexity of $O(N)$.

- Lin-Kernighan (LK) Algorithm : Developed by Lin S. and Kernighan B. at Bell Laboratory and modifies local tours from a given complete Hamiltonian tour (Lin and Kernighan, 1973). This method is largely based on k-opt algorithm, where the algorithm replaces k number of edges to find the best solution.

  - Approximation ratio, how accurate compared to the optimal varies by $k$ number of changed edges, generally within single digit percentage from the optimal solution.
  - Observed runtime also varies by implementation, however, empirically observed $O(N^{2.2})$, and generally space complexity of $O(N)$ is reasonable (Helsgaun, 2000; Papadimitriou, 1992).

The order picking operation is essentially a Steiner TSP, an extension of TSP with additional Steiner points, which may or may not be visited to reduce the total length of travel (Cornuéjols et al., 1985).

Because of this, a literature implemented a variation of LK, Lin-Kernighan-Helsgaun heuristic, for effective routing, and reported about 47 percent saving in travel distances compared to S-shaped routing and 0.1 percent deviation from optimum on average (Theys et al., 2010).

In certain cases like parallel long aisle layouts, Ratliff and Rosenthal proved that there is a linear time solvable solution for the TSP, however, the optimal solution acquired from this method posses limitations, which will be discussed next before discussing commonly used heuristics (Ratliff and Rosenthal, 1983).

### 2.2.2 Commonly Used Strategies

Mainly, it was established that there are no generally optimal solution for every layout, and that optimum routing heuristics do not take congestion or multiple picking operations into account (de Koster et al., 2007). One of the greatest hindrances in implementing the optimal route is in the difficulty and cost of the solution and lack of transparency in the algorithm. In practice, there are several commonly used heuristics: S-shaped, Mid-point, Return, and Largest-Gap.

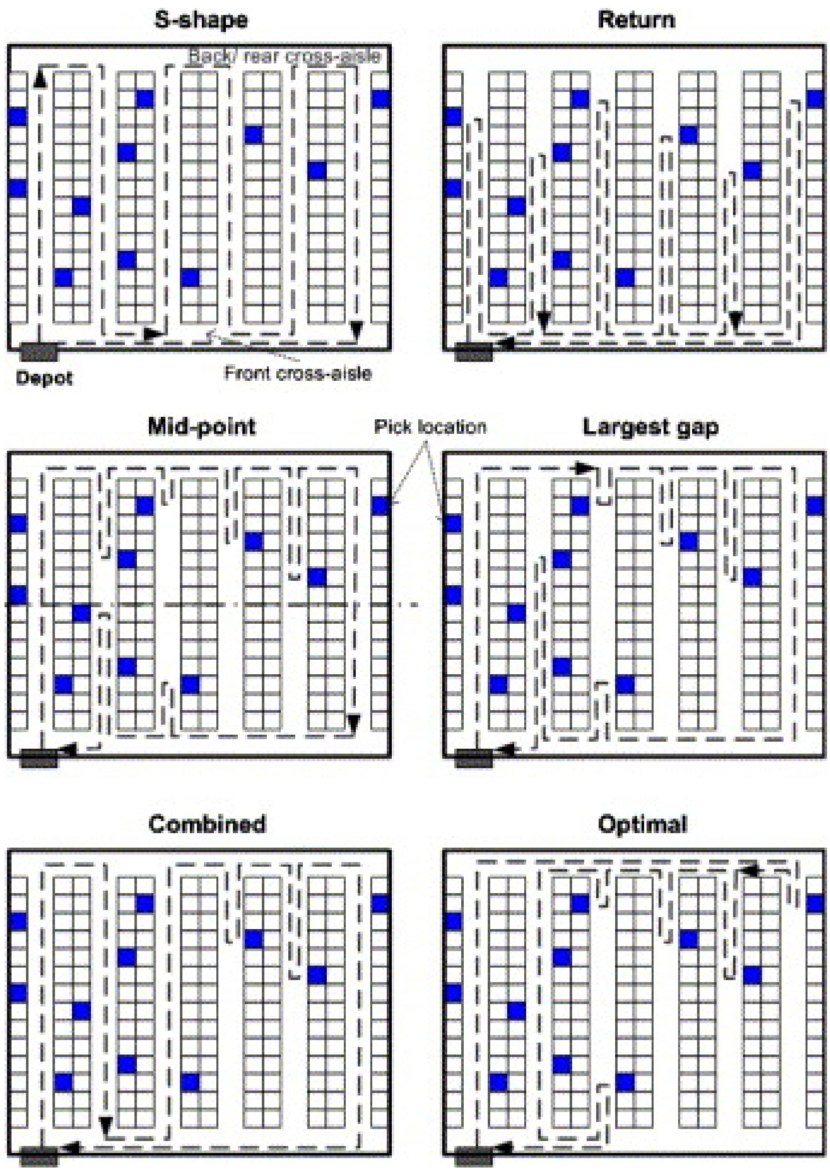



Figure 2.8: Different routing strategies for parallel aisles layout (Roodbergen, 2001).

Figure 2.8 show different heuristics, and first, the S-shaped heuristic travels through an aisle with a pick order, leaving from the other end of the aisle every time there it enters one.

Second, the warehouse is divided into two area, a picker enters an aisle then picks orders up until the mid point of the aisle, then exits the same point where it entered. The picker crosses over to the other half of the warehouse at the aisle of the last order in the first area, when using the Mid-point heuristic.

Third, in the Return heuristic, the picker enters and exits from the same end of the aisle and only travels until the pick order. Lastly, the Largest-Gap heuristic only travels as far into the aisle as the largest gap between pick orders in an aisle.

To put simply, assume an aisle with two pick locations A and B, the layout is in series: end of aisle - A - B - end of aisle. If the gap between an end of aisle to a pick location is the greatest gap, picker does not enter the aisle from that end. If the gap between A and B is the largest, picker will pick A or B depending on which end it entered, then turn around to exit.

Additionally, combined method decides whether to traverse the aisle entirely or return (Roodbergen and de Koster, 2001). In comparison of these heuristics, Hall discovered that the Mid-Point heuristic outperforms S-shaped heuristic when pick list is small, and that the Largest-Gap heuristic always outperformed the Mid-Point, and Petersen further evaluated these strategies in random storage and discovered that the combined method yielded 8.9 percent greater than optimal and 41.1 percent greater than optimal for the return heuristic (Hall, 1993; Petersen, 1997).

In evaluating the performance of routing and warehouse layout, two main metrics are travel time and travel distance for a pick operation. These both measure the travel of the picking effort, because minimizing travel effort in a pick operation is essential in minimizing losses and potentially increasing the throughput of the warehouse AS/RS.

Researches attempt to estimate the travel time of a picking operation in different layouts by Bassan and Larson, while Bozer and Cho presented evaluation under stochastic demand (Bassan et al., 1980; Larson et al., 1997; Bozer and Cho, 2005).

To further evaluate the performance of the routing heuristics, it needs to be compared against the optimal travel time, and Ratliff and Rosenthal provides the lower bound in this case (Ratliff and Rosenthal, 1983). There are also cases where the p/d points are not centralized, which influence the routing and the consequent travel duration, de Koster presented estimation of average travel time and compares the gain in average travel against S-shaped heuristic, evaluating the worthiness of implementing optimal routing despite the complexity and unforeseen risks (de Koster et al., 1998).

From his literature, the performance improvement of an optimal routing depended upon the number of aisles and pick list, presenting its limits especially when picker vehicles discussed in the literature value flexibility of carrying out other operations than order picking in a practical context.

To summarize, it seems that there does not seem to be a generally optimal solution in either heuristics and optimal routing for every pick order and layout. These literature only discussed long aisle layouts, therefore operation in fish bone layout or its variants would further jeopardize the evaluation of routing strategies. All in all, selection of a routing strategy needs to account for these various factors and make adequate trade

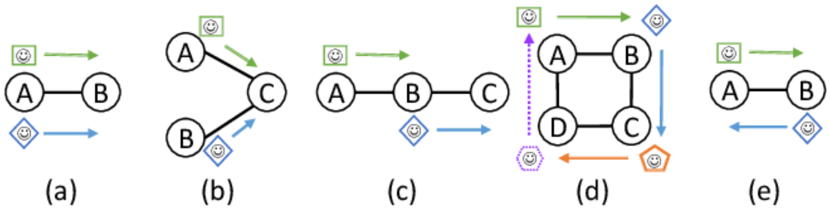



Figure 2.9: Five types of conflicts. (a) Edge Conflict. (b) Vertex Conflict. (c) Following Conflict. (d) Cycling Conflict. (e) Swapping Conflict. (Stern et al., 2019).

offs.

## 2.3 Multi Agent Path Finding

Modern warehouses like RMFS are vastly different from previously researched designs, often having decentralized p/d points and omni-directional layouts, due to incorporation of multiple simultaneously working picking agents.

While previous researches considered routing strategy and warehouse layout fitted to a single or few pickers, RMFS utilize AGVs to maximize throughput of picking operation, posing a new important factor of picker interaction into the equation. Without coordination, individual routing may lead to congestion and deadlocks when routes converge or cross, to address this issue, Multi Agent Path Finding (MAPF) is studied.

Stern establishes some important definitions in MAPF (Stern et al., 2019). To put simply, for $k$ number of agents, there exists a tuple $G$ comprised of set of vertexes $V$ and set of edges $E$, a set of starting vertexes $s \rightarrow V$, a set of destination vertexes $t \rightarrow V$. For an agent, $\pi$ is a set of moves along the vertexes at each discrete time step $x$, and an agent may either move or wait at each time step. A solution is defined as $k$ number of $pi$s do not have a conflict at every step of $x$. There are five types of conflicts that may occur:

- Edge Conflict : Agents travel along the same edge and direction at the same time step (Fig 2.9(a)).
- Vertex Conflict : Agents travel to the same vertex at the same time step (Fig 2.9(b)).
- Following Conflict : An agent intends to travel to a vertex previously occupied by another agent (Fig 2.9(c)).

- Cycling Conflict : Agents intend to travel to a vertices previously occupied by other agents, which also intend to travel to a previously occupied vertices to create a ”cycle” (Fig 2.9(d)).
- Swapping Conflict : Two or more agents travel to vertices previously occupied by each other (Fig 2.9(e)).

It becomes increasingly difficult to obtain a conflict free solution, thus an algorithm decides which of the five conflicts are to be ignored. According to Stern, there are two types of objective functions that an algorithm aims to optimize.

- Makespan : This is the number of time steps needed to reach the destination for all $k$ agents.
- Sum of Costs : The sum of total time steps needed to reach the destination for all $k$ agents.

Though these objective functions provide a guideline for MAPF, others may construct unique objective functions to fit their needs, such as a function that separates ”move” and ”wait” actions to minimize either action, and in the context of warehouse order picking, the bounds go beyond classical MAPF.

Ma et al proposed a subclass of Online MAPF as ”Lifelong MAPF” for warehouse MAPF, where fixed number of agents solve sequence of MAPF problems, thus upon reaching their destination, agents are assigned to a new destination.

In warehouse situations, the graph mimics the layout and movement of AGVs, thus often incorporate a grid-like design as seen from Cohen et al, and Asprilo benchmark framework often used in MAPF evaluation (Fig 2.10) (Cohen et al., 2018; Gebser et al., 2018).

Yu and LaValle proved that classical MAPF is NP-hard for finding an optimal solution to minimize makespan ans sum of costs, and similarly Online MAPF problems and Lifelong MAPF are NP-hard as well (Yu and LaValle, 2013; Ma, 2021).

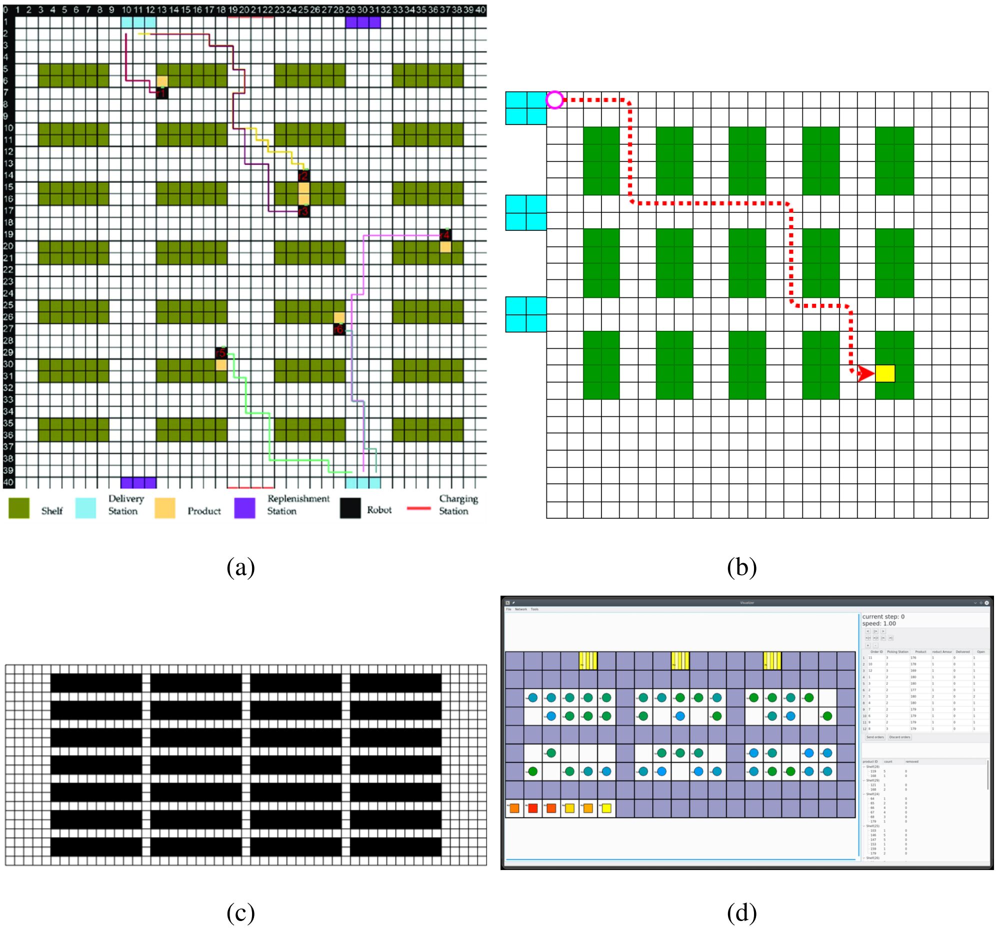



Figure 2.10: Grid representation of warehouse MAPF problem. (a) A Routing and Task-Allocation Algorithm for Robotic Groups in Warehouse Environments (Chatzisavvas et al., 2022). (b) Mobile Robot Path Optimization Technique Based on Reinforcement Learning Algorithm in Warehouse Environment (Lee and Jeong, 2021). (c) Rapid Randomized Restarts for Multi-Agent Path Finding: Preliminary Results (Cohen et al., 2018). (d) Experimenting with robotic intra-logistics domains (Gebser et al., 2018).

# Chapter 3

# Gait-Based Human Identification Using iPhone Video and Skeleton Analysis

Understanding human gait patterns holds significant potential in various domains, including healthcare monitoring, security, and behavioral analysis. However, most existing gait recognition methods rely heavily on specialized equipment such as multiple cameras or motion capture systems, which limit their practical applicability in real-world environments. In contrast, this study aims to explore whether gait-based identification can be achieved using only monocular video data captured under unconstrained settings.

To this end, we recorded videos of 89 individuals walking from left to right using an iPhone 15 Pro in HD at 60 frames per second. Each participant was asked to walk naturally over a 10-step sequence, maintaining a constant direction and speed. The camera was fixed at a distance of approximately 12 meters from the walking path and at a height of 1 meter, ensuring that the walking direction was perpendicular to the camera's line of sight when the subject reached the center of the frame. This setup was deliberately chosen to avoid experimental constraints that arise in environments where the camera follows the subject, thereby demonstrating the feasibility of gait identification even under simpler and more accessible recording conditions.

We utilized the OpenPose BODY25 model to extract 25 keypoint skeleton data from each video frame. Among these, eight features were selected for gait analysis: knee, pelvis, neck, body, ankle, nose-to-knee, nose-to-ankle, and pelvis-to-ankle angles. These features were chosen based on their ability to represent the dynamic structure of walking. Each participant's gait was segmented into four full walking cycles, derived from 10 steps, using the knee joint angle peaks to detect step boundaries. From each cycle, the segment corresponding to foot-ground contact—between the local minimum of the knee angle and the maximum of the pelvis angle—was further divided into four equal parts. The average joint angles in these intervals were used as representative scores, yielding four scores per feature and thus eight features × 4 scores = 32-dimensional feature vectors per cycle.

Our observations revealed that as more features were incorporated, the ability to distinguish between individuals improved. Initially, only two features—knee and pelvis joint angles—were used, but as we added features such as body inclination, ankle dynamics, and head posture, the identification accuracy increased. Moreover, we found that different observers intuitively focus on different aspects of movement—some emphasize leg movement while others focus on torso posture or head alignment. This motivated a multi-perspective analysis, where we defined six "views" of gait features based on anatomical regions and their respective relevance in cluster formation.

We conducted a series of experiments to evaluate the effectiveness of the proposed features and methodology. First, we measured gait similarity between individuals using Euclidean distance and visually confirmed that people with similar scores exhibited comparable gait patterns. Second, we performed self-identification experiments, comparing a participant's single cycle to the average of their remaining three cycles using joint angle scores. The results showed that identification using only one cycle was generally challenging, whereas comparing to the average of four cycles significantly improved accuracy. Finally, we applied K-means clustering and entropy-based decision trees using different feature combinations to classify gait types. These experiments confirmed that distinct gait patterns can be captured and quantified using only joint angles extracted from monocular video, without the need for specialized recording environments.

In summary, this study demonstrates that gait-based identification is feasible using only OpenPose-based skeleton data extracted from monocular video. By analyzing the temporal patterns of key joint angles across walking cycles and exploring the role of multiple features, we provide a practical framework for gait analysis that balances simplicity in data acquisition with robustness in feature representation. (Ryu et al., 2024b).

## 3.1 Method

Pertaining to the focus of this research, the path shortening property of the random storage will be investigated in congestion reduction. It is expected that shorter paths will reduce the likelihood of colliding with another agent, while the scattered item storage will avoid bottle-necks when multiple agents attempt to reach a single article of item.

In a random storage environment, the matter of locating the desired item does not pose a significant challenge as the storage management system can locate the necessary items, thus items need not to be at a specific place. Rather, having the items accessible from anywhere in the warehouse would be more desirable. Furthermore, if the items can always be located, there does not seem to be a substantial advantage in having the items grouped.

Another commonly used storage method is Class-Based Storage (CBS), which is used when different items have different demands. Items are ranked by class based on their picking activity, and items that are picked more frequently are placed near the picking area, and items with less demands are placed farther away (Petersen et al., 2004; Petersen and Schmenner, 1999). This method aims to optimize the path lengths and consequently the picking time, and it is most suited when items can be divided into classes by their demand.

While storage methods aim to passively reduce the likelihood of congestion occurrences, an active method employs a routing the agents. It is conjectured that the agent based method would perform better as the number of agents increase, while the path density and intersection based method would perform better when there are less agents. When there are more agents, the path based method could have difficulty avoiding areas determined to be congested as the absolute number of paths on the grid increases, and this also applies to the intersection based method.

Since the convolution filter determines a congestion at the kernel center, the determined congestion area is generally a point or cluster of points on a grid, whereas the path based counterpart would have a point of congestion area when paths intersect perpendicularly, and a line of congestion area when paths overlap. Similarly, the intersection based method entirely avoids intersection with pre-existing paths, resulting in multiple paths that need to be avoided.

Consequently, the intersection based method is generally expected to have more number of grids to avoid, while the agent based method would have the least. This problem is also expected to be more apparent as the number of agents grow. The resilience against increasing number of agents in the agent based method is expected to perform better with more agents, utilizing the size of the fleet to perform more picking operations. Another aspect of the storage method and the path planning is the occurrences of bottlenecks, and regardless of the configuration, they are expected to occur near the stations by the incoming and outgoing agents.

Additionally, bottlenecks may occur in the picking area, especially for the organized storage strategy, because agents moving between different designated areas are likely to intersect and collide. Such introduction of bottlenecks may likely trigger the path planning more frequently.

In contrast, the CBS storage would be less prone to such bottlenecks even though the items are designated to their storage area, due to the introduction of varying chances of selection. While bottlenecks may occur among agents picking high demand items and agents picking low demand items, there would be less agents picking low demand items, and most agents would operating near the high demand items. Likewise, high demand items are stored in greater volume in larger area, thus freedom of movement is provided proportionally to the demand.

### 3.1.1 Storage Methods

In the random storage method, each of the 360 racks within the warehouse grid is assigned a uniform random chance of storing one of four item types: red, blue, green, or purple. This approach ensures that approximately equal numbers of each item type are distributed randomly across the storage area, as illustrated in Figure 3.2(a). Consequently, items of different types are interspersed throughout the warehouse, promoting a more even distribution and potentially reducing travel distances for agents during the retrieval process.

Conversely, the traditionally organized storage method adopts a structured approach, designating specific rectangular storage areas within the warehouse grid for each item type. As depicted in Figure 3.2(b), distinct regions of the grid are allocated to accommodate items of different types(colors): blue items occupy the grid cells within the range (0-14, 0-18), purple items are stored in cells (0-14, 19-35), green items occupy cells (16-29, 0-18), and red items are positioned in cells (16-29, 19-35). This method facilitates efficient item retrieval by grouping similar items together, thereby reducing the search time and streamlining the order picking process.

The CBS method incorporates considerations of demand variability and proximity to stations, and implements a non-uniform distribution strategy, wherein items with higher demand are strategically positioned closer to the stations for expedited retrieval. To implement this bias in item selection probability, an additional setting was introduced, wherein each of the four types of racks possesses a distinct "bias" or probability of being selected as a target.

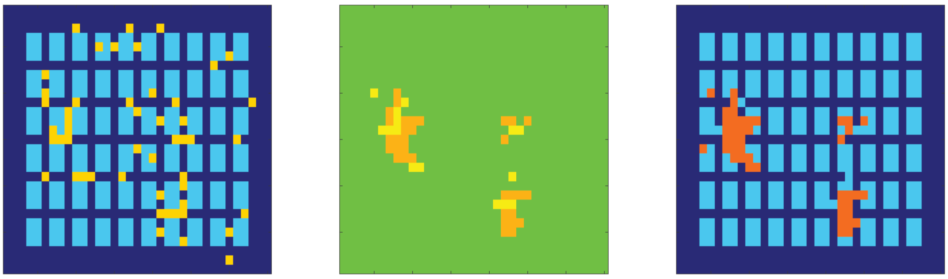
The biased demand has its basis in ABC analysis following a Pareto's Distribution, also known as the 80-20 rule where 80 percent of outcomes are typically attributed to 20 percent of causes, reflecting the uneven distribution of demand across different item types (Farrington and Lysons, 2012).

$$r_{n-k} = \sqrt{r_{n-k+1}^2 - P(X = x) r_n^2} \tag{3.1}$$

The distribution has a scale parameter $x_m = 1$ and shape parameter $\alpha = 1$ which govern the distribution's behavior. The storage area for each item type is partitioned by the Pareto's Distribution, with the bounds of each storage area calculated using the following equation, where $r_n$ denotes the radius of the largest storage area considering the total number of item varieties $n$. $P(X = x)$ is the probability density function of the Pareto's distribution, and $r_{n-k}$ represents the $k$th item type. By mapping the storage areas based on the Manhattan distance from the center of the grid, the resulting distribution yields diamond-shaped storage areas, as depicted in Figure 3.2(c).

While different storage strategies were proposed by Petersen, the diagonal storage method was selected, as drawing a boundary from a point with a fixed Manhattan distance results in a diamond or diagonal shape (Petersen et al., 2004). In his literature, within-aisle storage and rectangular storage performed better than diagonal storage, however, in consideration of different aisle and geometry and routing strategy, diagonal storage seemed more practical pertaining to this simulation as Petersen's configuration employs long continuous aisles, whereas this research place the racks in clusters with more passages.

Finally, the effect of different item varieties were explored, because the choice between random storage and CBS storage depends heavily on the specific characteristics of the business model and the diversity of product offerings. Companies such as Amazon.com Inc., renowned for their extensive product catalog, often opt for random storage due to its adaptability to diverse item types and unpredictable demand patterns. Conversely, businesses with fewer item varieties and more predictable demand may find CBS storage more suitable, as it allows for strategic placement of high-demand



(a) (b) (c)

Figure 3.1: Capture of a picking operation with 50 agents and traditionally organized racks. (a) Location of agents on the field. Light blue represent the storage racks and the yellow dots represent the agents. (b) Visualization of congestion map grid. Congestion identified by the convolution filter are marked by the corresponding color to the kernel sizes, light blue (2), yellow(3), and red color(5).

items closer to stations, thereby optimizing order picking efficiency.

To investigate the effects of varying item varieties, simulations were conducted with the number of agents fixed at 20, with item types varying 2, 3, 6, and 12, and additional 30 and 60 item types for the random and traditional storage, and CBS storage was excluded from higher item variety due to the storage boundaries becoming too fine and dense, the difference between a boundary radius $r_{n-k+1}$ and its subsequent radius $r_{n-k}$ becomes too small and fails to create sufficient area for item storage.

Ultimately, the findings of these investigations contribute to the development of best congestion avoiding behaviors in different business environments and allows for a comprehensive understanding of the trade-offs and considerations within the warehouse management.

### 3.1.2 Path Planning

The decision to employ a locally repairing path search rather than searching for optimal paths with the least intersection is rooted in practical considerations regarding computational complexity and efficiency. Searching for paths with the least intersection, while conceptually appealing, presents significant computational challenges.

This problem is akin to the multi-commodity flow problem, which is known to be NP-complete. NP-complete problems are notoriously difficult to solve optimally, often requiring exponential time to find the optimal solution. Although there exist efficient approximation algorithms and heuristics for solving NP-complete problems in polynomial time, selecting the optimal approach introduces another layer of complexity (Goldberg et al., 1998; Robinson, 1949; Kizilateş and Nuriyeva, 2013; Johnson and McGeoch, 2008).

The task of identifying the most suitable algorithm or heuristic becomes a challenging optimization problem in itself, requiring careful consideration of trade-offs between solution quality and computational efficiency. Therefore, a much simpler method called Local Repair A* (LRA*) was devised involving re-routing an agent consequently to a collision by avoid the visited cell, expected to decrease the number of collision and ultimately remedy congestion (Hart et al., 1968; Ma et al., 2018).

This method would effectively prevent agents from repeatedly re-attempting to enter a congested area and facilitate vacation of paths for agents in higher priority. This method will not only be tested against its default counterpart without such feature, but further congestion avoidance algorithms will implement this re-routing technique along with their novel congestion detection and avoidance routing heuristics.

The agent density-based path planning method is a heuristic that leverages a convolution filter to calculate the local density of agents within the warehouse grid. This technique involves several key steps to effectively identify and navigate around congested areas.

The congestion detection and avoidance process begins with the initialization of a ”congestion map,” an empty grid matching the warehouse environment’s dimensions. This map serves as the foundation for tracking and responding to congestion patterns throughout the operational space.

To detect congestion at different scales, the system employs multiple kernel sizes (2, 3, and 5) along with corresponding thresholds (1, 3, and 4). These parameters enable the system to identify and respond to varying levels of congestion density and distribution across the warehouse grid.

The core of the detection mechanism lies in its convolution operation. For each kernel size, the system creates a square kernel with ones in all positions except the center, which remains zero. This kernel is then convolved with the grid containing agent locations, following the equation:

$$\text{Convolution Result}[i,j] = \sum_{m=-k}^{k}\sum_{n=-k}^{k} \text{Agent Location Grid}[i-m, j-n] \times \text{Kernel}[m,n] \tag{3.2}$$

Where Convolution Result$[i, j]$ represents the output grid value at position [i, j], Agent Location Grid$[i-m, j-n]$ refers to the input grid value after offset [m,n], Kernel$[m,n]$ denotes the kernel value at position [m,n], and $k$ is the floor of $\frac{\text{Kernel Size}}{2}$.

The convolution operation calculates local agent density across the warehouse grid. When this density exceeds the kernel's corresponding threshold, the system marks congestion on the congestion map. Finally, these marked cells are assigned higher traversal costs, prompting the A* algorithm to construct alternative paths avoiding congested areas.

The path density-based path planning method extends the congestion detection principle to account for the density of agents' paths within the warehouse grid. This approach aims to identify and avoid areas where multiple agents' paths overlap excessively, signaling potential congestion hotspots.

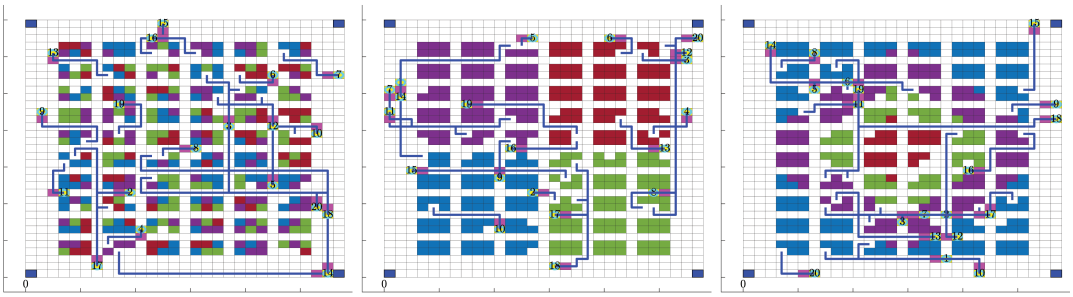
Similar to the agent density-based method, the path density-based approach initializes a congestion map to track areas of high path density. However, instead of marking grid cells based on the presence of individual agents using a convolutional filter, the congestion map is updated to reflect overlapping paths of multiple agents.

The congestion avoidance process begins with the creation of a congestion map that mirrors the simulation field's dimensions. As agents plan their movements, their intended paths are continuously recorded and overlaid onto this map, with each grid cell along an agent's path being marked accordingly.

The system then analyzes the congestion map to identify potential traffic bottlenecks by scanning for areas where multiple paths converge. This analysis focuses on detecting grid cells where the density of planned paths exceeds a predefined threshold, indicating potential congestion points. When such high-density areas are identified, the corresponding grid cells are marked as congested, highlighting zones where multiple agents' paths intersect and congestion is likely to occur.

During path planning, the A* algorithm incorporates this congestion information by assigning higher traversal costs to marked grid cells. This cost adjustment encourages agents to seek alternative routes around congested areas while maintaining efficient paths to their destinations. This dynamic path planning approach enables proactive congestion avoidance while preserving overall system efficiency.

To mitigate the risk of agents re-entering congested areas after escaping from them, an avoidance mechanism is implemented, where, upon successfully navigating away from a congested region, agents mark the grid where congestion escape was activated, thereby avoiding its re-entry during path planning. By incorporating this avoid-



(a)

(b)

(c)

Figure 3.2: Simulation running with 15 agents. Racks of different types are distinguished by their color. (a) Racks are randomly dispersed across the field. (b) Racks of the same kind are organized in their designated storage area. (c) Racks are organized by Manhattan distances from the stations.

ance mechanism, agents are less likely to backtrack into congested areas they previously escaped from, enhancing overall navigation efficiency and reducing the likelihood of prolonged delays.

In instances where an alternate path cannot be found due to the absence of feasible detours, the reconstructed path may temporarily ignore the avoidance mechanism to ensure that agents continue to make progress towards their goals. On the other hand, the path planning hueristics are based on A* Search method, with different Density-Based methods applying weights to nodes for congestion avoidance.

To further inspect how the proposed Agent Density Based Path Planning performs, newer and improved path planning called Windowed Hierarchical Cooperative A* (WHCA*) was implemented into the simulation and its collision avoiding and throughput increasing performances will be evaluated (Ma, 2021).

In this method, the agents cooperate, do not plan the entire path from start to finish, but plans the path within a set window. This algorithm is hierarchical, meaning that it plans a rough path then elaborates upon it (Silver, 2021).

### 3.1.3 Simulation Environment

The research methodology employed in this study involves the utilization of a simulation framework designed to replicate the intricate dynamics of order picking within a warehouse environment. This simulation model encompasses a detailed representation of the physical layout, comprising a grid measuring 29 units horizontally and 32 units vertically. Within this grid, the warehouse space is subdivided into 60 clusters, each consisting of 2 by 3 cells, mirroring the spatial configuration commonly observed in contemporary warehousing systems, as exemplified by the organizational structure employed by industry giant Amazon.com, Inc. (Roser, 2021).

In instances where agents become trapped between obstructing entities or encounters an agent with higher priority travelling in opposite direction, the agent retreats to its previous grid cell. If the previous cell is occupied by another agent, the occupying agent is also signalled to retreat to its previous location.

Furthermore, to preemptively address potential unanticipated congestion scenarios, each agent is equipped with an internal counter mechanism, tracking the duration of grid position stagnation. Upon reaching a predefined threshold, agents autonomously trigger congestion escape protocols, thereby proactively resolving congestion and restoring the fluidity of operations.

Each agent keeps an individual map of the field, and its own path represented by a list of continuous grid indices on the map. The agents have three picking states from 1 to 3, where the aforementioned ”priority” of traversal in collision correspond, with higher picking states having higher priority, The first picking state carrying a rack from a station to the closest empty cell, the second state moving to the target rack after returning a rack to an empty cell, and the third state carrying the target rack to the station.

Circulating through these three states, the agents essentially conduct a dual-command operation, where agents are instructed to return an item to its storage, while also ordered to retrieve an item to the P/D station (Pohl et al., 2009). The agents are provided with a path constructed by the central coordinator to their destination whenever the agent state changes.

The paths are discovered using A* algorithm, and is designed to search for a path that has the shortest Manhattan (rectilinear) distance to the destination. Assuming that the agents are two-wheeled differential drive Automated Guided Vehicles (AGV)s, the agents move in a constant speed in a straight path, and the agents make a stationary pivot turn when changing its direction.

Each item within the warehouse is categorized by one of four integers between 1 and 4, which correspond to their respective color: red, blue, green, and purple. A list maintains the status of each cell within the warehouse grid, with 0 and 1 to denote the vacancy of the cell. As a results, agents may identify empty cells to place their item.

Of three main metrics used to evaluate the effectiveness of a path planning method,

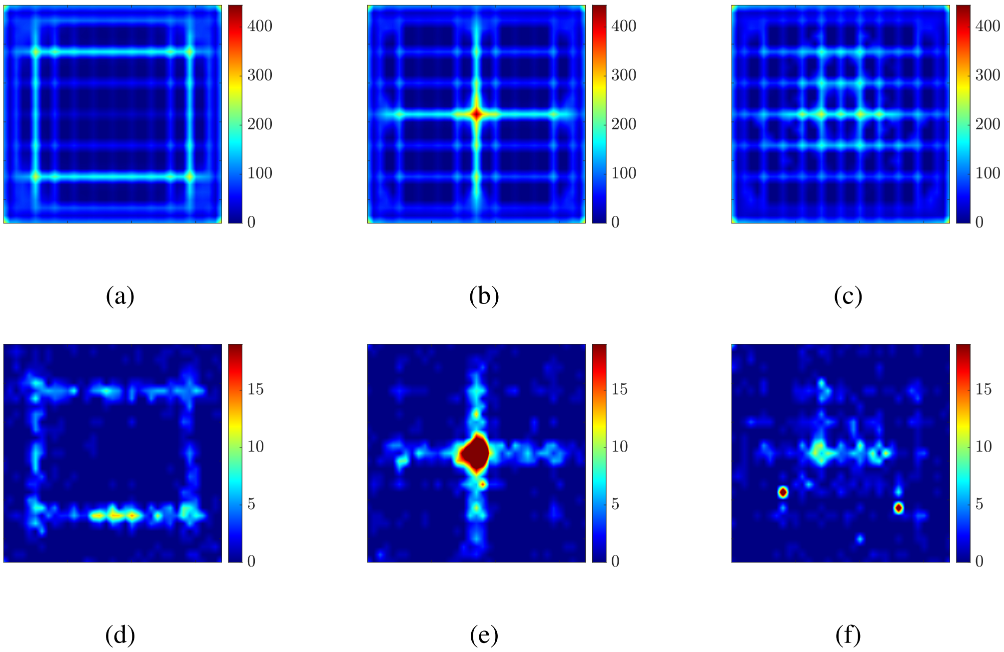



Figure 3.3: Heat maps representing the agent activity (a, b, c) and congestion (d, e, f) across the field with no path planning. (a,d) Random Storage. (b,e) Traditionally Organized Storage. (c,f) CBS Storage.

the volume represents the number of items processed through the P/D points within the predefined duration of the simulation. Collision and congestion are also the number of total collision during the run of simulation, defined by an agent's next step blocked by another, and congestion defined by an agent determining that its path is completely blocked towards the destination and its escape algorithm is triggered. This congestion count is also recorded and used to evaluated the effectiveness of rerouting and bottleneck prevention.

## 3.2 Results

### 3.2.1 Random Storage

The comparison of completed picking operation volumes in random storage revealed notable differences in performance among the path planning methods, particularly with respect to the agent density-based approach. Despite some overlap in the standard deviations of the completed picking volumes, the boxplot and multi-comparison analysis from the ANOVA in Figure 3.5 provide compelling evidence of the superior perfor-

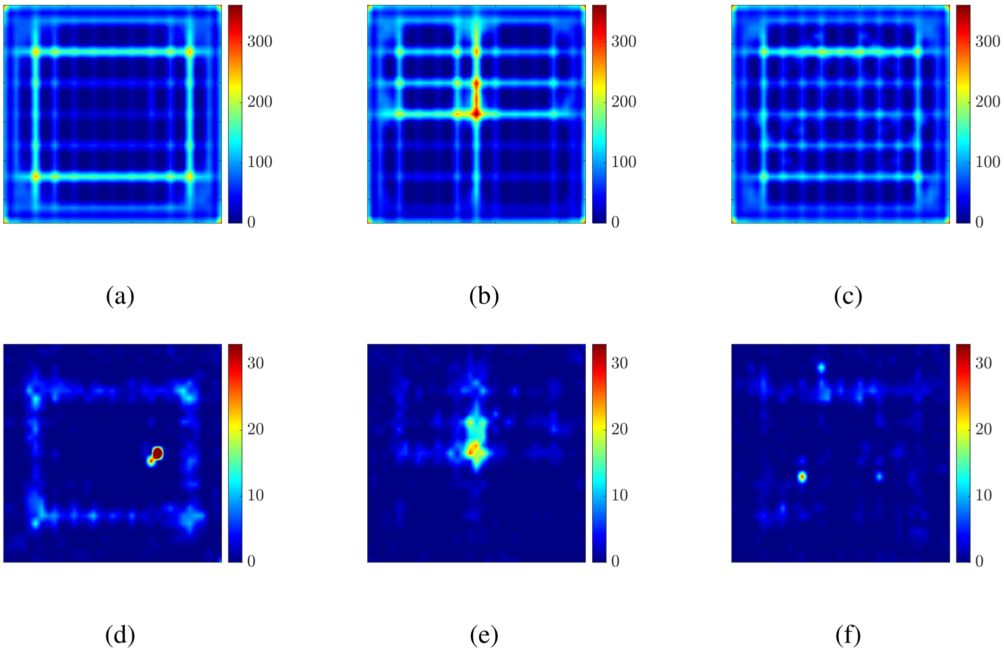



Figure 3.4: Heat maps representing the agent activity (a, b, c) and congestion (d, e, f) across the field with no path planning. (a,d) Random Storage. (b,e) Traditionally Organized Storage. (c,f) CBS Storage.

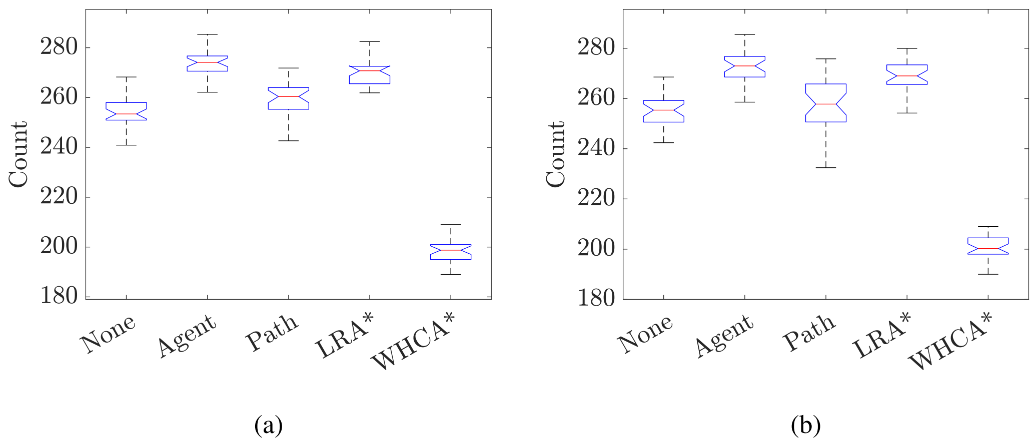



Figure 3.5: ANOVA of total volumes for random storage configuration. (a) Uniform item distribution (b) Biased item distributions.

mance of the agent density-based path planning method over having no implementation, path density-based, and the WHCA* algorithm.

However, LRA* seems to perform as good as the agent density-based method, and it is perhaps due to how the agent density based method operates. Because the

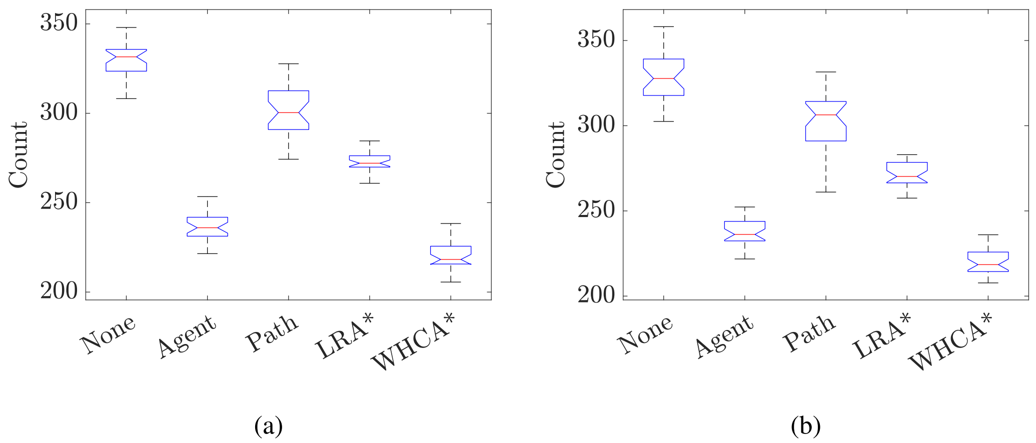



Figure 3.6: ANOVA of total collisions for random storage configuration. (a) Uniform item distribution (b) Biased item distributions.

congestion detection by convolutional filter is triggered by a collision with another agent, it effectively performs similarly to the LRA*, finding a new path that avoids the location of collision. Unlike the LRA*, the agent based method also accounts for congestion and other agents' density across the map, however the effectiveness of this key difference seems to be diminished by the intrinsic nature of the random storage method that minimizes congestion.

Overall, the findings emphasize the significant influence of path planning methods on completed picking volumes in random storage scenarios. Conversely, the decreased performance of the WHCA* method highlights the need for further investigation into its limitations and potential refinements to enhance its efficacy in real-world warehouse environments.

Furthermore, the comparison between uniform and biased demands for items revealed no noticeable deviance of picking performances and exhibited similar increase in performance nor resulted in changes in relative performances among the different path planning methods except for in path based reconstruction method.

The comparison of collision and congestion counts across different settings revealed significant improvements with the agent density-based path planning method. The effectiveness of the agent density-based method in alleviating bottlenecks and spreading out agent activity is particularly noteworthy and can be seen in Figure 3.6 and 3.7. Regardless of the biased or uniform demand of items the performances of the path planning methods seem consistent.

Though the WHCA* method exhibited the lowest number of collision and conges-

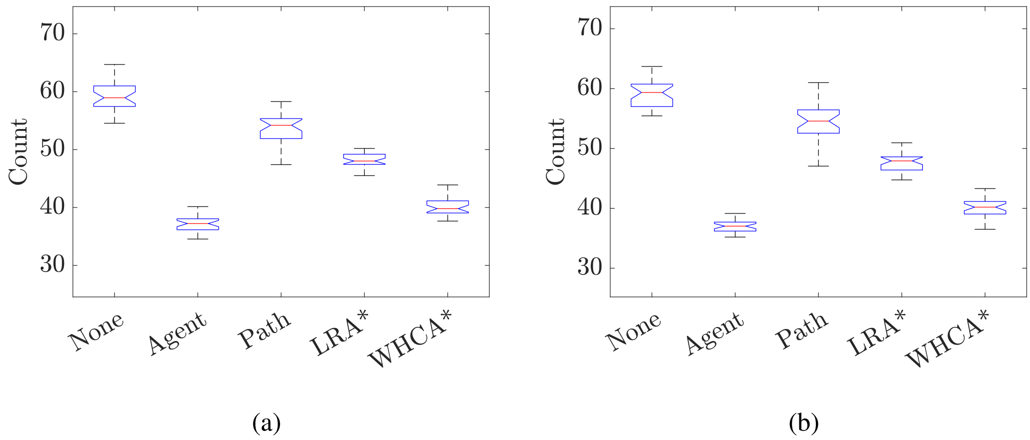



Figure 3.7: ANOVA of total congestion counts for random storage configuration. (a) Uniform item distribution (b) Biased item distributions.

Table 3.1: Mean Collision, Congestion, and Volume counts for each path planning algorithm when randomly stored with uniform item demands.

| | **Collision** | **Congestion** | **Volume** |
|---|---|---|---|
| None | 334.6(±22.46) | 60.74(±9.054) | 253.9(±8.784) |
| Agent Density | 236.9(±7.520) | 37.22(±1.257) | 274.1(±6.797) |
| Path Density | 301.3(±15.64) | 53.50(±2.807) | 259.9(±8.320) |
| LRA* (Stout, 1998) | 272.3(±6.235) | 48.15(±1.140) | 270.0(±6.150) |
| WHCA* (Silver, 2021) | 220.0(±8.167) | 40.28(±1.837) | 198.2(±4.854) |

Table 3.2: Mean Collision, Congestion, and Volume counts for each path planning algorithm when randomly stored with biased item demands.

| | **Collision** | **Congestion** | **Volume** |
|---|---|---|---|
| None | 327.7(±13.24) | 59.24(±2.341) | 254.8(±6.357) |
| Agent Density | 237.3(±8.172) | 37.33(±1.766) | 272.9(±6.158) |
| Path Density | 306.7(±29.34) | 56.34(±12.44) | 257.3(±10.14) |
| LRA* (Stout, 1998) | 270.8(±7.702) | 47.65(±1.562) | 268.7(±5.724) |
| WHCA* (Silver, 2021) | 220.5(±8.219) | 40.56(±1.630) | 200.1(±4.841) |

tion counts among all algorithms, comparing this results to the previous findings about total volume of items processed, the improved collision avoidance do not seem to translate into increased throughput. While LRA* showed improvements in both throughput increase and collision/congestion mitigation, the effects were not as dramatic as the agent density based method.

Interestingly, though the agent based method and the WHCA* exhibited similar reduction in both collision and congestion, the agent density method seems to result in slightly less congestion despite having insignificantly greater collision count, as shown in Figure 3.6 and 3.7.

All in all, the path density based method did not seem to yield impressive performance improvement over the LRA*. As mentioned earlier, the density based algorithms share similar operation with the LRA*, searching for a new path upon collision, but with additional rules. While the agent density based method demonstrated superior performance than the LRA*, the path density method did not. Across all performance evaluation metrics, it performed worse, and the reason could be conjectured that avoiding areas with high path density induced additional collisions and congestion, worsening the total throughput of items.

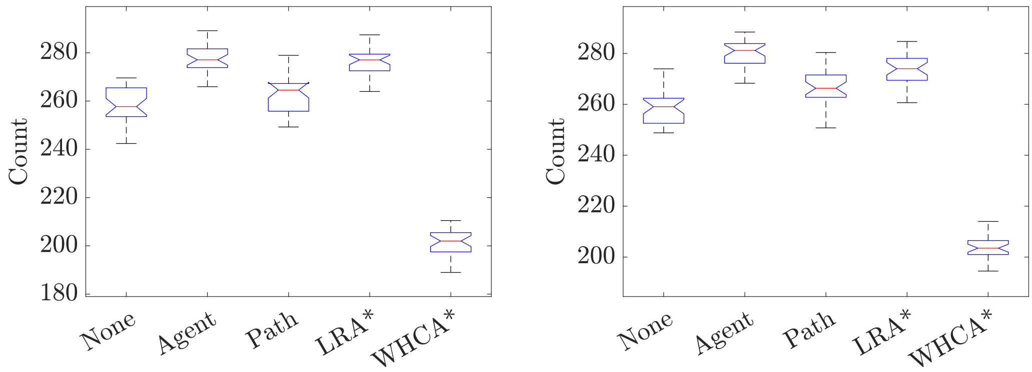
So far, the findings ephasize the complex tradeoffs and interactions in path planning methods across different storage methods and item variation. Certain methods such as WHCA* method may reduce collision and congestion to a meaningful degree, however, sacrifices the picking performance. In contrast, LRA* method finds a balance in lowering congestion while maintaining or in some storage methods, surpassing the picking performance of the default non-implementation.

It seems that the agent density based path planning does not have such compromises between the picking performance and the congestion mitigation, yielding improvements in all aspects consistently in any storage configuration when the items have uniform demands.

### 3.2.2 Traditional Storage

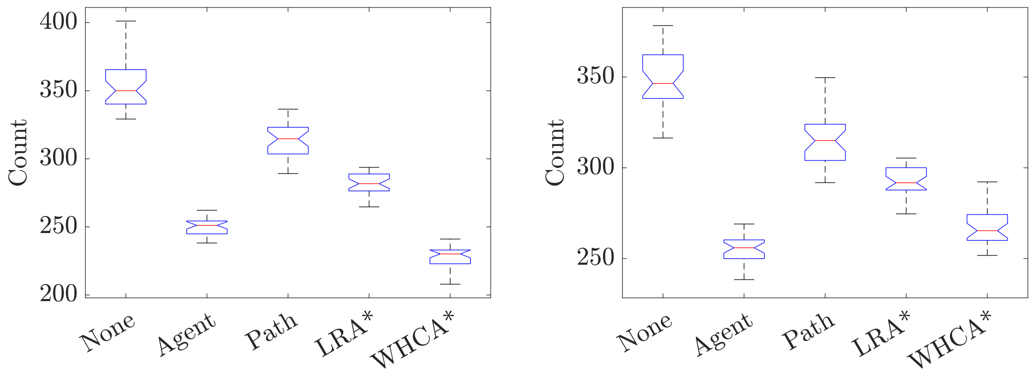
The overall trend of the picked volume, collision, and congestion counts seem consistent to the uniform distribution. At a glance, The path density based method matches having no implementation as can be seen from Figure 3.8, and this is similar to the observation in the uniform case. The observations of the picked volume in biased distribution concur with the uniform counterpart, with agent density method performing well.

Furthermore, the differences in picked volume from the default no-reconstruction and the implementations are smaller in the Class Based Storage, and the differences are




(a) (b)

Figure 3.8: ANOVA of total volumes for traditional storage configuration. (a) Uniform item distribution (b) Biased item distributions.



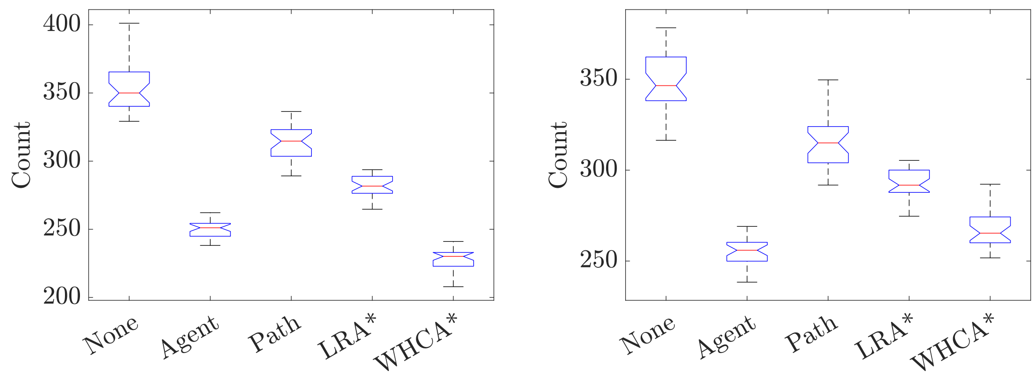

(a) (b)

Figure 3.9: ANOVA of total collisions for traditional storage configuration. (a) Uniform item distribution (b) Biased item distributions.

the greatest in the traditionally organized method. It can be extruded that in traditionally organized method induce congestion at the center of the map where different types of items interface, and this static and constant bottlenecks amplify the effectiveness of constructing circumventing routes.

In evaluation of the uniform and the biased setting, biased item demands induced discrepency between the LRA* and agent density based method in total picked volume. This is perhaps due to the biased setting exacerbating bottlenecks by leading the pickers to a fixed storage area for high demand items.

Upon further investigation of collision and congestion counts in different scenar-




(a) (b)

Figure 3.10: ANOVA of total congestion counts for traditional storage configuration. (a) Uniform item distribution (b) Biased item distributions.

Table 3.3: Mean Collision, Congestion, and Volume counts for each path planning algorithm when traditionally stored with uniform item demands.

| | **Collision** | **Congestion** | **Volume** |
|---|---|---|---|
| None | 354.6(±19.69) | 60.42(±3.014) | 257.6(±8.484) |
| Agent Density | 249.4(±7.021) | 37.87(±1.244) | 277.6(±5.860) |
| Path Density | 314.5(±15.26) | 54.26(±2.475) | 262.8(±7.013) |
| LRA* (Stout, 1998) | 281.4(±7.514) | 48.33(±1.151) | 276.0(±6.117) |
| WHCA* (Silver, 2021) | 228.3(±8.091) | 41.39(±2.235) | 201.7(±5.287) |

Table 3.4: Mean Collision, Congestion, and Volume counts for each path planning algorithm when traditionally stored with biased item demands.

| | **Collision** | **Congestion** | **Volume** |
|---|---|---|---|
| None | 348.8(±17.12) | 60.50(±2.559) | 259.2(±7.683) |
| Agent Density | 255.0(±6.619) | 38.66(±1.408) | 279.9(±4.977) |
| Path Density | 315.0(±14.90) | 54.90(±2.559) | 266.5(±6.317) |
| LRA* (Stout, 1998) | 292.5(±7.581) | 49.37(±1.475) | 273.7(±6.038) |
| WHCA* (Silver, 2021) | 267.6(±12.37) | 47.93(±2.692) | 204.2(±5.323) |

ios, all path planning method made meaningful effort in reduction. The WHCA* once again demonstrated superior reduction of collision counts in exchange for the picking performance.

Table 3.5: Mean Collision, Congestion, and Volume counts for each path planning algorithm when stored according to CBS with uniform item demands.

| | **Collision** | **Congestion** | **Volume** |
|---|---|---|---|
| None | 308.2(±46.48) | 59.62(±16.04) | 273.9(±10.90) |
| Agent Density | 225.0(±13.89) | 36.60(±7.878) | 289.7(±4.940) |
| Path Density | 275.2(±15.92) | 49.83(±5.029) | 271.1(±5.578) |
| LRA* (Stout, 1998) | 255.7(±15.59) | 46.44(±9.482) | 287.9(±5.079) |
| WHCA* (Silver, 2021) | 237.6(±58.62) | 48.62(±25.69) | 213.4(±4.373) |

However, unlike the random storage method, Figure 3.9 show that the collision reduction was outperformed by the agent density based method once the item demands possessed bias. As seen from Figure 3.4(b) and e, picker activities are dense at a certain location, and congestion occur as a consequence. While WHCA* is effective, the agent density based method's convolutional filter considers other agents' activity across the map, thus creating better circumventing paths.

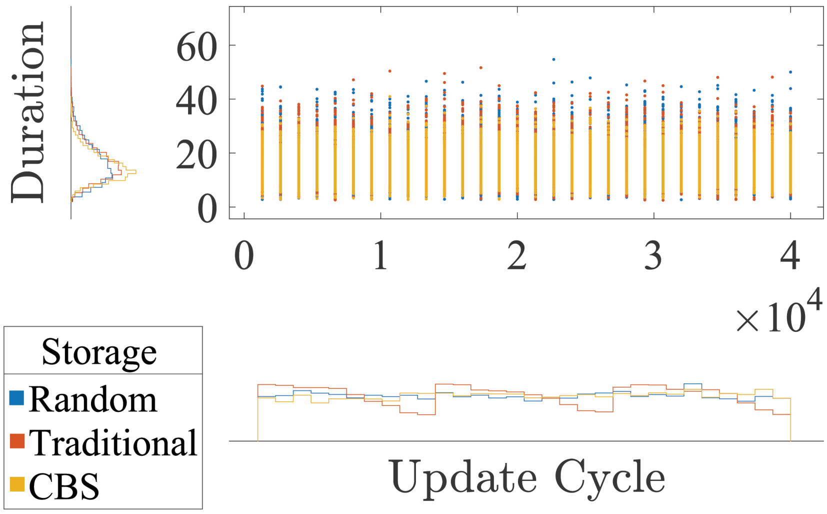
Additionally, the congestion mitigation of the WHCA* suffer, and its effectiveness closely follows LRA* in this aspect. While, the path density based method was expected to perform better with a somewhat predictable and static bottleneck, it exhibited collision and congestion reduction, but, the improvements do not seem to be in a meaningful degree for increasing picking throughput.

As demonstrated in Figure 3.11(a), the fluctuation nature of the congestion formation and resolution when there are no implementations to prevent the formation of bottlenecks in a traditionally configured storage. Such geometry is especially prone to formation of congestion, highlighting the importance of not only the geometry, but also the preventative algorithms.

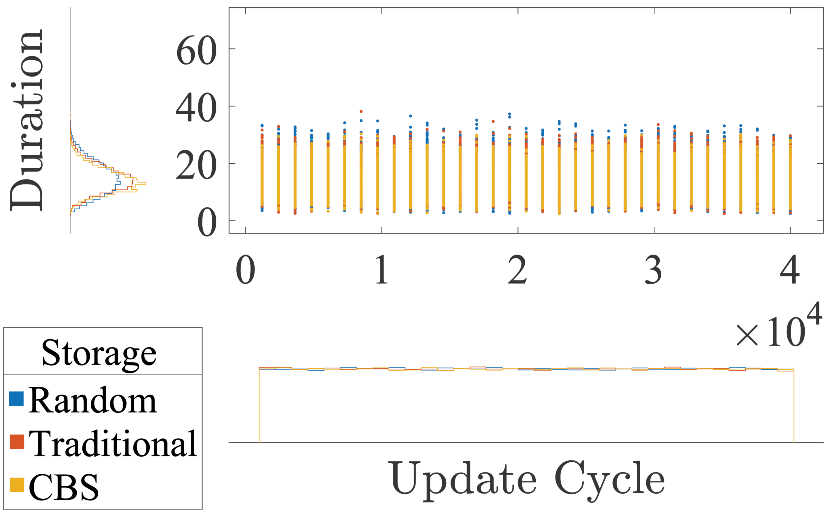
In contrast, Figure 3.11(b) demonstrates the effectiveness of its traffic managing ability, where the duration of non-movement is uniformly managed compared to the other three plots.

### 3.2.3 Class Based Storage

As depicted in Figure 3.12, the introduction of bias into item demands appears to have minimal impact on the dynamics between different path planning methods. In both cases, the picked volume follow a similar trend as the previous random case, where the path density method have no meaningful change in mean completed volume as hav-




(a)



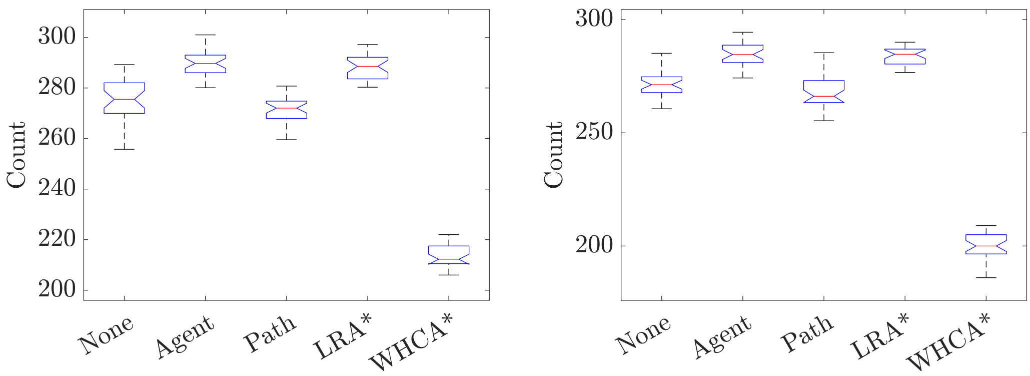

(b)

Figure 3.11: Duration of non-movement of picking agents over time. The scatter plots are binned into intervals, then histograms were drawn with respect to Update Cycle and Duration. (a) No path planning. (b) Agent density based path planning.

ing no implementation, and agent density based and LRA* having improved picking volumes.

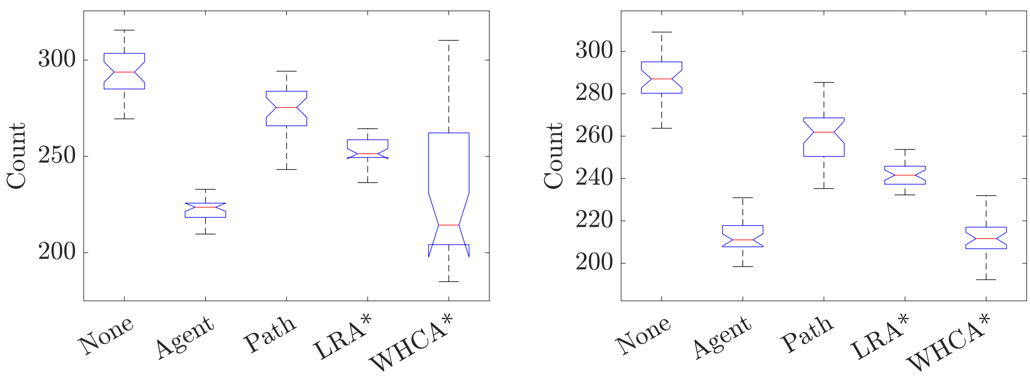
The WHCA* also underperformed in picking operation. A more in-depth statistical analysis, as illustrated in Figure 3.13 and 3.14, reveals intriguing insights into the performance of these methods. Specifically, when no bias was introduced in the Class-Based Storage (CBS) method, the WHCA* showed wider variation in its total number of collisions.

Such behavior is also observed when investigating the total number of congestion,




(a) (b)

Figure 3.12: ANOVA of total volumes for CBS configuration. (a) Uniform item distribution (b) Biased item distributions.




(a) (b)

Figure 3.13: ANOVA of total collisions for CBS configuration. (a) Uniform item distribution (b) Biased item distributions.

Table 3.6: Mean Collision, Congestion, and Volume counts for each path planning algorithm when stored according to CBS with biased item demands.

| | **Collision** | **Congestion** | **Volume** |
|---|---|---|---|
| None | 288.0(±12.03) | 54.19(±5.020) | 271.9(±6.018) |
| Agent Density | 215.8(±20.62) | 36.33(±11.82) | 284.2(±5.748) |
| Path Density | 262.1(±16.97) | 48.31(±4.779) | 268.1(±6.964) |
| LRA* (Stout, 1998) | 241.9(±8.443) | 43.58(±2.554) | 284.0(±3.921) |
| WHCA* (Silver, 2021) | 213.0(±9.633) | 39.27(±3.611) | 199.4(±6.086) |

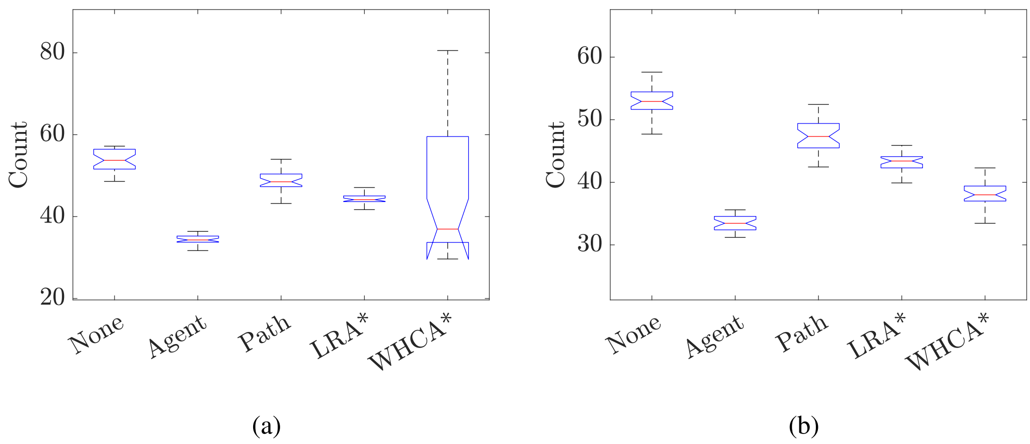



Figure 3.14: ANOVA of total congestion counts for CBS configuration. (a) Uniform item distribution (b) Biased item distributions.

also having a wide variety that translated from the collision counts, and it seems that the particular geometry of the product placements influence WHCA*, and the introduction of bias limited the agents' picking activities near the p/d points and away from potential bottlenecks within the storage space.

Because WHCA* also performed the worst in previous experiments pertaining to the picked volumes, it is unclear as if the collision and congestion performance impacted the picking operation. On the other hand, it is noteworthy that the agent-based path planning method emerged as the most statistically significant in terms of enhancing total picked volume, as indicated by the results of comparisons in Figure 3.12.

In the subsequent analysis, collision counts resulting from various path planning methods were meticulously examined, as depicted in Figure 3.13 and 3.14. Figure 3.13 and 3.14 unveils a noteworthy trend: all path plannings yielded a decrease in collision counts when demands were uniform across items.

Notably, the LRA* and the path density-based method exhibited comparable performance, with no significant discrepancy between them. Meanwhile, the agent density-based method outperformed every other method, consistently achieving lower collision counts. Conversely, the path-based method exhibited under-performance compared to the previous scenario involving biased demand distributions. This divergence in performance can be attributed to the paths converging towards storage areas housing highly demanded items, thereby limiting freedom of movement near stations and reducing the availability of alternative routes.

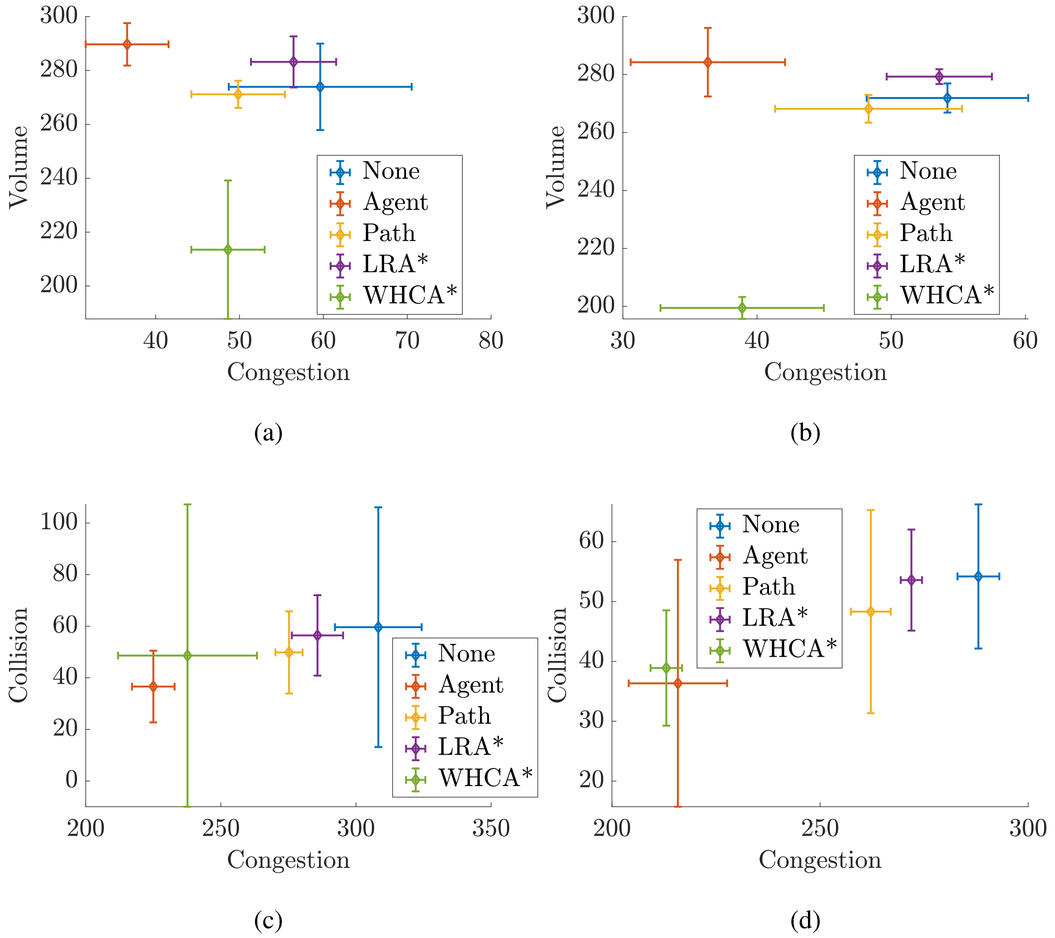



Figure 3.15: Volume plotted against Congestion and Collision plotted against Collision to discover the relationship (**A, C**) uniform distribution, (**B, D**) biased distribution.

Consequently, the path-based method proved ineffective, resembling a scenario with no implementation at all. These findings underscore the intricate interplay between demand distributions, path planning methods, and operational dynamics within warehouse environments.

By elucidating the performance nuances of different path planning strategies under varying demand scenarios, this analysis provides valuable insights for optimizing collision mitigation efforts and enhancing operational efficiency in Class-Based Storage systems.

## 3.3 Summary of Chapter 3

This chapter presented and evaluated a novel Agent-Density Based Path Finding algorithm that uses convolutional filters to detect and respond to congestion in warehouse environments. The evaluation demonstrated that the agent density-based method achieved balanced improvements across all performance metrics compared to other approaches, reducing collisions by 29.2-percent compared to no implementation while simultaneously increasing throughput by 8-percent. The method showed notably consistent performance with lower variance in both collision and completion counts, maintaining its effectiveness across different storage configurations and demand patterns.

Performance comparisons with alternative methods revealed distinct trade-offs: while WHCA* achieved the lowest collision rates, this came at a significant cost to throughput performance. LRA* showed moderate improvements in both metrics but not as substantial as the agent density approach, while the path density based method showed minimal improvements over baseline. These comparisons highlighted the agent density method's unique ability to improve multiple metrics simultaneously without significant trade-offs.

The agent density approach demonstrated robust performance across all storage types, including random, traditional, and CBS configurations. It proved particularly effective with traditional storage where static bottlenecks typically occur, and maintained its performance even with biased item demands. Notably, the method showed remarkable resilience to increased agent counts compared to path-based methods, suggesting superior scalability in high-density operations.

The success of the agent density approach can be attributed to several key factors: its consideration of agent distribution across the entire warehouse space, ability to identify and respond to emerging congestion patterns, and effective balance between path optimization and congestion avoidance. The implementation demonstrated important practical advantages, with the convolutional filter approach providing computationally efficient congestion detection while requiring minimal parameter tuning compared to alternative approaches.

This research establishes the agent density-based method as a promising approach for warehouse robotics, offering consistent performance improvements while maintaining operational stability across diverse scenarios. The method's balanced optimization of both safety and efficiency metrics, coupled with its adaptability to different storage strategies, suggests strong potential for practical implementation in real-world warehouse automation systems. Its successful integration with existing A* pathfinding frameworks and robust performance across various warehouse configurations further supports its viability for industrial applications.

# Chapter 4

# Adaptive Stochastic Class Based Storage

In addition to path finding algorithms, this chapter further explores the effect of storage placement to decrease the travel distances of multiple picking agents and also the throughput of item picking and storage, collision counts, and congestion occurrences. A novel storage strategy is proposed with adaptive storage placement, which utilizes a queue to observe recent picking activities and modulate the distance of storage location based on the determined demands of different types of items.

The aim of this method is to determine the demands of items in a dynamic environment and place high demand items closer and low demand items farther from the P/D points to decrease the overall travel distances in real time. Furthermore, this method aims to introduce stochastic behaviors by randomly storing items within the storage area to decrease the travel distances in spontaneous cases where low demand items are needed.

In evaluating the performance of the proposed storage policy, its performance will be compared with other methods in metrics such as throughput of items, collision counts, congestion occurrences, and achieved storage distances' gain. As a result, the proposed method resulted in increased performance in throughput and mitigation of collision and congestion against other storage methods, especially when the demands and input of different item types varied. It also demonstrated meaningful modulation of item storage location in response to changing demands. This chapter's contents are available on arXiv and is in preparation for submission to International Journal of Physical Distribution & Logistics Management (Ryu et al., 2024a).

## 4.1 Method

The simulation comprises four distinct storage modes designed to operate under different turnover rate configurations. The first storage mode is the random storage method, similar to the system utilized by Amazon Inc. In this mode, picking agents place items in the nearest available spot without regard to their class or turnover rate, resulting in a random distribution of articles.

The second mode is a fixed Class-Based Storage (CBS) system, where storage areas for each class are predetermined. Within these designated areas, picking agents choose the closest available spot for item placement. The third method introduces a predetermined CBS system that incorporates probabilities based on a normal distribution (N-CBS). This mode aims to balance the storage distribution more effectively by leveraging statistical models.

The final method, which is the primary focus of this research, is termed Adaptive Stochastic Class-Based Storage (AS-CBS). This approach uses a queue to dynamically assess turnover rates and adjust item storage distances accordingly. In AS-CBS, item placement follows a normal distribution with its mean determined by the storage distance, and the variation is influenced by the queue capacity. This system records the flow of articles into the warehouse, updating the queue with the number of items in each article class. These counts are then used to determine how turnover rates are distributed, subsequently calculating the mean storage distance for each class of items.

At the initial stage, the queue is empty, leading to a broad standard deviation and a mean placed at the midpoint between the P/D station and the farthest Manhattan point in the warehouse. As the queue fills up, the variation narrows, and the ratio of the size of each class relative to the total items recorded in the queue is used to calculate the mean, or the center, of the normal distribution for placement. The determined normal distribution is then used to randomly select a storage location. If the selected location is unavailable, the nearest empty spot to the initially selected location is chosen instead. The mean of the distribution is determined by the proportion of the item in the queue.

$$P_i = \frac{l}{2} + \frac{d_i}{\max(|d_1|, |d_2|, |d_3|, |d_4|)} \cdot (0.45 l_{max}) \tag{4.1}$$

$$d_i = \frac{c_i}{q_n} - \frac{1}{4} \tag{4.2}$$

Building upon the above equation, the equation for obtaining the mean distance $P_i$ for a class $i$ from the P/D station was constructed, where $l_{max}$ denotes the farthest

Manhattan distance from the P/D station and $c_i$ denotes the number of items in class $i$ and $q_n$ is the size of the queue. $d_i$ is the deviation of the proportion from mean, calculated by 4.2, and $\max(|d_1|,|d_2|,|d_3|,|d_4|)$ is the maximum absolute deviation among all points. $.45l_{max}$ is the scaling factor to ensure points stay within the line bounds.

$$s_i = 1 + (l_{max} - 1) \cdot (1 - \frac{|N \cdot \frac{c_i}{q_n} - 1|}{max(1, N \cdot \frac{c_i}{q_n} - 1, 1 - N \cdot \frac{c_i}{q_n})}) \tag{4.3}$$

Using equation 4.1, the mean for placing the item is obtained and the standard deviation for the normal distribution is calculated by 4.3. With $N$ denoting the total number of classes, the deviation $s_i$ approaches $l_max$ as the number of items within a queue becomes proportionate, and approaches 1 vice versa. This ensures that as the demands become more uniform and balanced, the normal distribution's stochastic placement becomes wider across the storage space to behave more like a random storage.

On the other hand, if demands are polarized, the separation of storage space becomes stricter due to the narrow distribution. This enables the AS-CBS to not only place the items by demand, but also adapt so that the probabilistic travel distance reduction expected from random storage may occur.

The turn over rate configurations are divided into three, with the turn over rates for each class are uniform, Pareto's distribution, and Pareto's distribution with a reversal. Pareto's distribution is commonly known as the ”80-20” rule where 80 percent of the outcomes are due to the 20 percent of the causes. This distribution is commonly seen in many observable phenomena, and it was used to describe the unequal turn over rates for different classes of articles in this research (Farrington and Lysons, 2012).

In this configuration, the turn over rates are in descending order from the first class to the last, following a Pareto curve, ultimately replicating the intended use case situation of the CBS. It is expected that the random distribution is more effective of all when the turn over rates are uniform, and the other three CBS-like methods effective when the turn over rates follow a Pareto's distribution. Finally, the third configuration of the turn over rate is a Pareto's distribution based method, but the turn over rates for the classes change during the simulation. This setting reflects the dynamic nature of product demands and potentially lifespan of perishable items. Ultimately, this setting aims to discover the performance of the AS-CBS and how well it responds to the changing turn over rate compared to other measures.

There are 20 identical picking agents in the environment, and they can move in the four primary directions. The mapping system is based on a grid and the distances

**Algorithm 1** AS-CBS Storage Location Assignment

**for** Each incoming item $i$ of type $t \in \{1, 2, 3, 4\}$ **do**
  Add item type $t$ to FIFO queue $Q$ of size $q_n$
  **if** queue is full **then**
    Remove oldest entry from $Q$
  **end if**
  Calculate counts $c_t$ for each type in $Q$
  Calculate proportions $prop_t = \frac{c_t}{q_n}$
  Calculate deviations $d_t = prop_t - \frac{1}{4}$
  **for** Each type $t$ **do**
    Calculate position $P_t = \frac{l}{2} + \frac{d_t}{\max(|d_1|,|d_2|,|d_3|,|d_4|)} \cdot (0.45l)$
    Calculate normalized proportion $np_t = 4 \cdot prop_t$
    Calculate deviation factor $dev_t = \frac{|np_t - 1|}{\max(1, np_t - 1, 1 - np_t)}$
    Calculate storage spread $s_t = 1 + (s_{max} - 1)(1 - dev_t)$
  **end for**
  Generate random offset $r \sim \mathcal{N}(0, \frac{s_t}{3})$ ▷ Normal distribution
  Assign storage location $L_i = P_t + r$
**end for**

are therefore calculated in Manhattan distances. When planning a path from the initial position to an arbitrary point on the grid, A* is employed. A single grid can be occupied by a single agent, thus collision and congestion may occur, thus a collision avoidance measures are employed.

When an agent detect another agent occupying the adjacent grid cell it intended to move to, it will wait until it is cleared. However, if the agents are head on, that is, the intended moves are opposite, one of the agents will retreat to its previous position and clear the way, and find a new path avoiding the location where the collision occurred to its original destination. This simple collision avoidance measure proved to be effective in the context of this research, significantly reducing the congestion and increasing the picking performance across all settings.

To simulate a physical warehouse environment, a 29 by 32 grid was constructed to map the storage locations and passages for picking agents' navigation. each grid cell is a 1 by 1 cell labeled either a storage "rack" or a passage (van den Berg and Gademann, 2000). At the four corners of the map ar P/D stations which the agents retrieve and receive storage items. In the center area of the map, there are clusters of

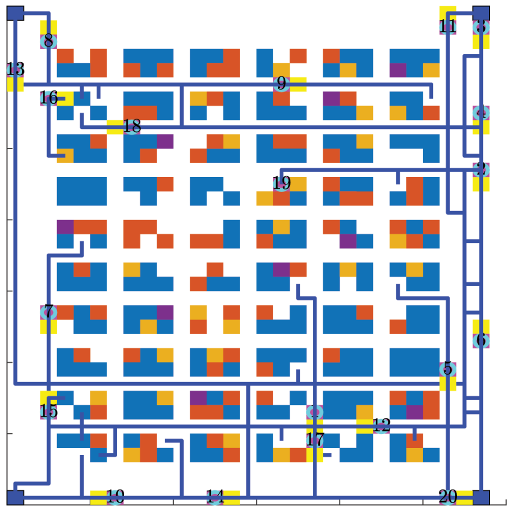



Figure 4.1: Simulation running with 20 agents in a randomly scattered storage. Racks of different classes are distinguished by their color, and the agents are numbered with the blue line indicating their path to the destination. Magenta marks the grid cell the agent.

storage racks of 3 by 2 dimension, totaling 60 clusters across the map. There are 360 available storage racks, and each rack can store one of four classes of item. Traditional warehouses employ long shelves or long corridors to move along, however, Petersen et al (2017) demonstrated that implementation of cross-aisles can reduce the travel distances of picking agents (Petersen and Aase, 2017; Gue and Meller, 2009; Gue et al., 2012; Meller and Gue, 2009; Hall, 1993).

A research showed that within-aisle configuration was the most effective for CBS, however the research employed a single P/D station with long aisle configurations to perform simulations, and as it can be seen from the configuration of this research, equal Manhattan distances naturally produce a diagonal configuration; furthermore, multiple P/D points and the prevalence of cross aisles in this research makes the diagonal configuration more suitable, and because of this, certain organizations such as Amazon Inc. employ a similar strategy using cross aisles and clustered storage (Roodbergen and de Koster, 2001; Roodbergen, 2001; Roodbergen and Vis, 2009; Roser, 2021; Eynan and Rosenblatt, 1993).

On the other hand, because the agents can freely move in four adjacent cells, the passages are inherently bidirectional, and though come routing strategies prefer unidirectional passages to reduce congestion, a research suggest that bidirectional paths may facilitate minimizing travel distances for picking agents and ultimately improve throughput of the picking activity (Pohl et al., 2010; Hsueh, 2010; Han et al., 1987;

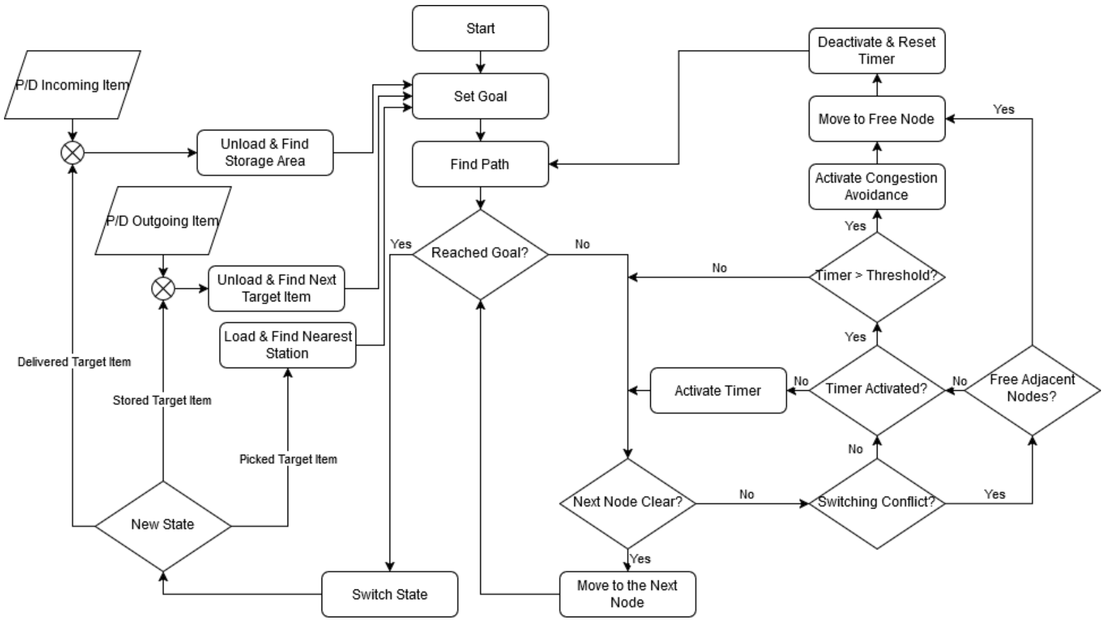



Figure 4.2: Flowchart of a picking agent behavior.

Frazele and Sharp, 1989).

In opposition, having bidirectional passages pose potential risk of increased collisions and congestion, thus a simple rerouting technique has been devised in response.

Figure 4.2 shows the operation of individual picking agents, and the individual picking agent keep a list of grid coordinates to follow as a result of A* path construction from its starting point to the destination point. This list contains the shortest Manhattan path, however, if the next grid position the agent intends to travel to is occupied by another agent, the agent will pause until the intended grid cell of movement is cleared (Stern et al., 2019).

Nevertheless, if both agents discover each other's position as the desired grid cell to travel to, i.e. if the agents intend to move in opposite directions to each other, one of the agents will move to the adjacent cell available to move into. This is denoted as "collision avoidance behavior," and the agent who "avoided" the collision will mark the location where the collision occurred and construct an new shortest Manhattan path to its original destination. If an agent does not find an adjacent cell available for avoidance, it will signal an adjacent agent to activate an avoidance by changing its intended cell to travel to in the opposite direction.

This behavior is named "congestion avoidance" behavior. Counting the occurrences of these two behaviors is an important measure in determining the effectiveness of the geometry of class based storage from the simulation settings, because although the shortened travel distances may reduce the theoretical travel paths, densely stored items with high turn over rate can induce bottlenecks, elongating the actual duration of picking operation.

Additionally, the throughput of the items is another important aspect of assessing the effectiveness of the system, as the optimized geometry would yield shorter travel distances, coupled with reduced collision to further expedite the picking agents travel along the shortened paths.

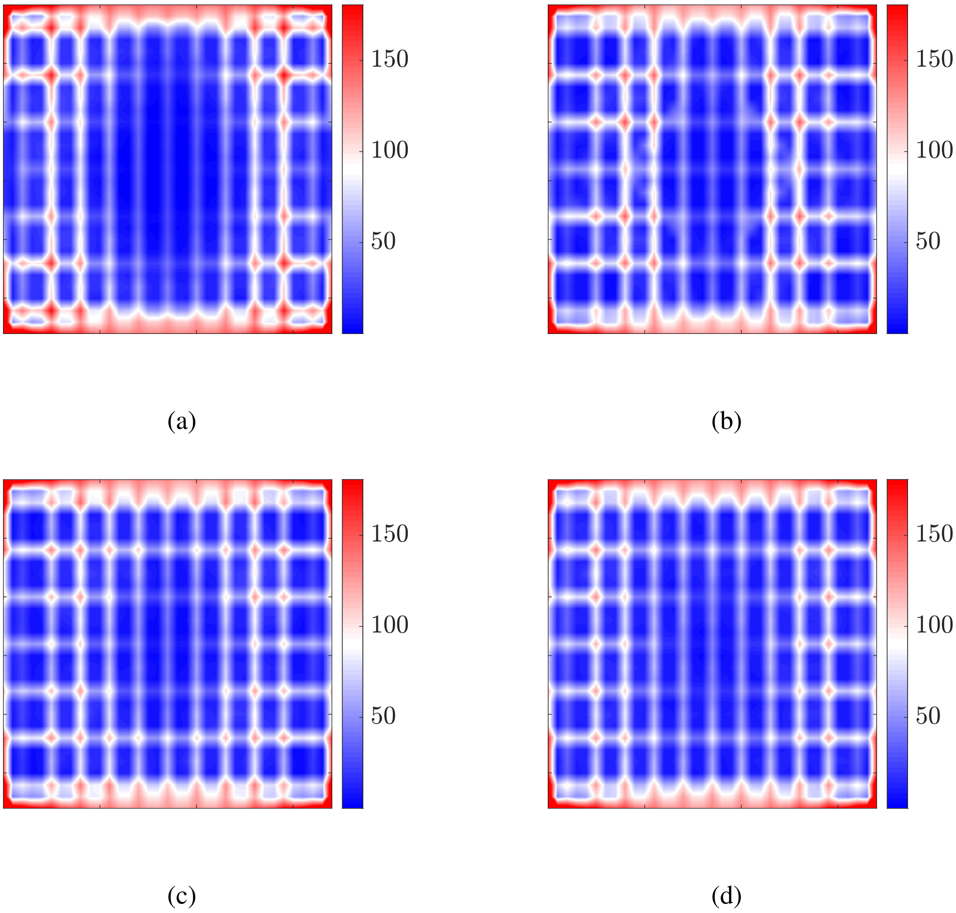
During the picking operation,there are three different tasks that the agents may perform. First is the storage of items from the P/D station to the storage area, second is the travel to a nearest item that matches the requested article, and finally the third is the retrieval of the requested item to the P/D station. Accordingly, having both the storage and retrieval operation is analogous to dual-command operation used in actual warehouse application due to their effectiveness (Pohl et al., 2009).

The four classes of items are enumerated from 1 to 4, and each are assigned corresponding colors to visually differentiate. For class 1, blue is assigned, class 2, orange, class 3, yellow, and finally class 4 is purple. In the default setting, the turn over rates for each class follows the Pareto's distribution, so that class 1 has the highest, then decreases towards class 4. For the management of the storage cells, The storage cells either contain one of the four classes of items, or is empty, and additional markers are used to mark cells that are targeted by a picking agent to prevent duplicate orders for the same cell.

Meanwhile, the cells can only contain one item, likewise, the picking agents replicate the Kiva's shelf lifting mobile robots, thus can load a single item at once. In real-life operations of such robots, the shelves contain multiple items within, however, the entire shelf is carried to the P/D station for an operator to pick the needed items from it.

The simulation consists of loop operations performed to carry out calculations and update the environment and variables, thus the duration of the simulation is counted by the number of updates occurred, denoted by $k$, rather than a timer, which is sensitive to the code and the hardware's performance. The duration of the simulation is 120000 update cycles, and the aforementioned inversion of turn over rates occur at $k = 60000$.

To ensure consistent results and statistically sufficient samples for evaluation, each combination of storage methods (random, Fixed CBS, N-CBS, and AS-CBS) paired with turn over rate configuration (uniform, Pareto's, Pareto's inversion) will be run 30 times. Overall, the simulation addresses the quintessential components of real




(a) (b) (c) (d)

Figure 4.3: Activity heat maps across storage methods (red: high, blue: low) under uneven Pareto's turnover rates. (a) Random. (b) Fixed CBS. (c) N-CBS. (d) AS-CBS.

life warehouses with simplicity to evaluate the performance of AS-CBS and other storage strategies.

## 4.2 Results

The activities of the picking agents across the grid were recorded for each storage method when the turn over rates for classes of items followed a fixed Pareto's distribution throughout the simulation and revealed that the random storage method generally had even activities across, with less activities near the center and the activities were closer to the P/D stations as seen from Figure 4.3(a).

Conversely, the Fixed CBS method demonstrated a distinct pattern with heightened picking activities concentrated near the center of the storage area. This clustering effect suggests that items were predominantly stored farther away from the P/D sta-

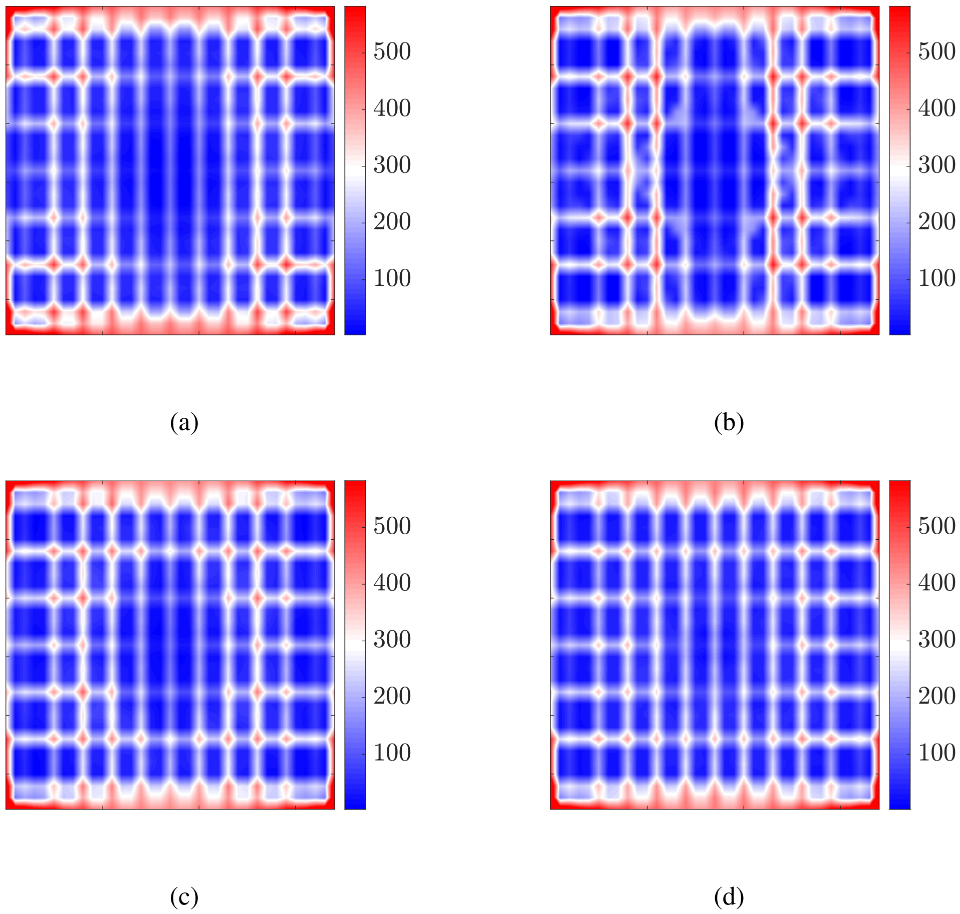
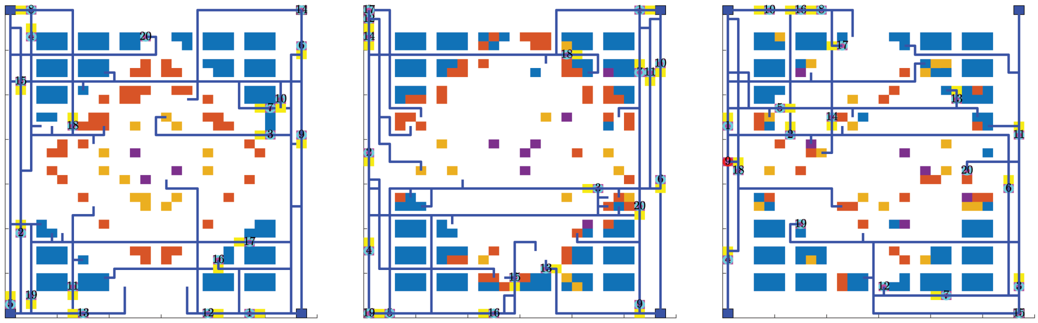



Figure 4.4: Activity heat maps across storage methods (red: high, blue: low) under dynamic Pareto's turnover rates. (a) Random. (b) Fixed CBS. (c) N-CBS. (d) AS-CBS.

tions (Figure 4.3(b)), potentially resulting in longer travel distances for picking operations. The observed dispersion of activities in the Fixed CBS configuration highlights the need for further investigation into its impact on operational efficiency and travel distances within the warehouse environment, potentially resulting in longer travel distances for picking operations. The observed dispersion of activities in the Fixed CBS configuration highlights the need for further investigation into its impact on operational efficiency and travel distances within the warehouse environment.

Nonetheless, the case of turn over rates inverting midway through the simulation was evaluated for picking agents' activities. At a glance, Figure 4.3(a) and Figure 4.4(a)'s similarities in activities concentrated towards the P/D station indicate that random storage method is possibly not influenced by the turn over rates.

Notwithstanding, Fixed CBS exhibited notable change in agents' activities as can be seen from Figure 4.4(b), where indications of increased activities near the farthest



(a) (b) (c)

Figure 4.5: Simulation running with 20 agents in a different CBS strategies to fill up an empty storage. (a) Fixed CBS. (b) N-CBS. (c) AS-CBS.

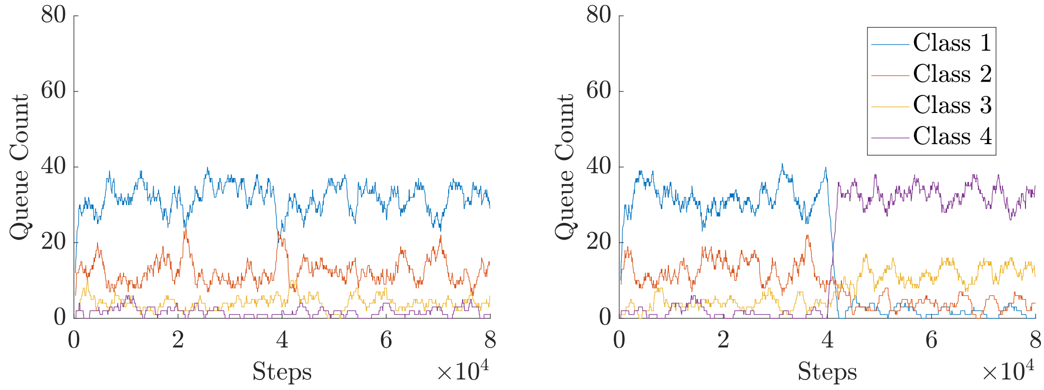
point from the P/D station is pronounced. Similarly, Figure 4.4(c) indicate increased activity near the center of the map. In both Fixed CBS and N-CBS, the storage methods follow a predetermined storage area or a distribution of storage area, thus their failure to respond to the change in turn over rate potentially harmed their performances.

Regardless, the AS-CBS show little differences in between Figure 4.3(d) and Figure 4.4(d), presumably indicating its adaptiveness to changing turn over rates.

Initial tests were conducted to observe the behaviors of the different CBS strategies when the turn over rates followed the Pareto's distribution as shown in Figure 4.5. The Fixed CBS configuration yielded a very organized storage, prioritizing filling up the nearest cells in the corresponding class storage area. The N-CBS showed a more disorganized behavior compared to the Fixed CBS, however, still showed the CBS's characteristics of storing high turn over rates near the P/D stations, and lastly, AS-CBS showed a more random dispersion of classes across the storage area, but gradually started to store high turn over classes near the P/D stations.

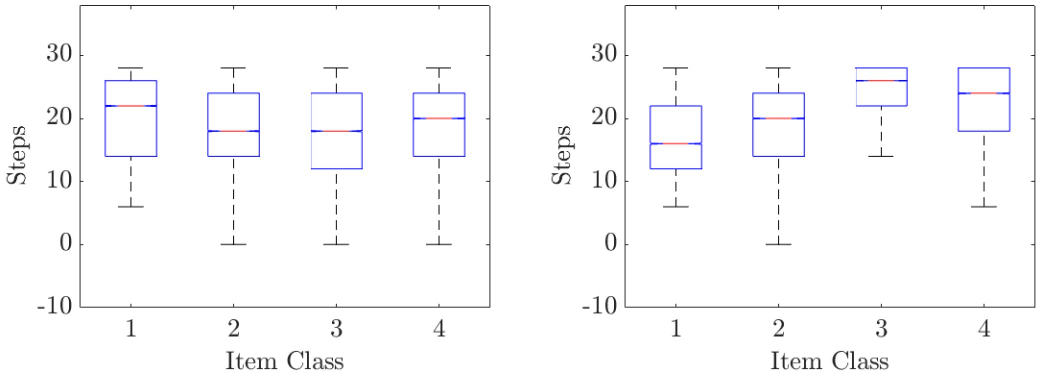
This phenomenon was somewhat expected due to the nature of the AS-CBS, the adaptive nature of the queue based construction of normal distribution constitute the necessity of a warm-up duration. While the queue fills up to its capacity, there are not enough samples to meaningfully discriminate the proportions, leading to a wide standard devation.

Consequently, such warm up time is dictated by the size of the queue as shown in Equation 4.3, and smaller queue size may decrease the warm up, but it results in a lower resolution to the underlying distribution of the turn over rates, hence making the adaptiveness overly sensitive to changes and insufficient for capturing the underlying




(a) (b)

Figure 4.6: AS-CBS queue over time during uneven distributed turn over rates, when (a) the distribution is kept constant, (b) distribution is flipped mid-point.



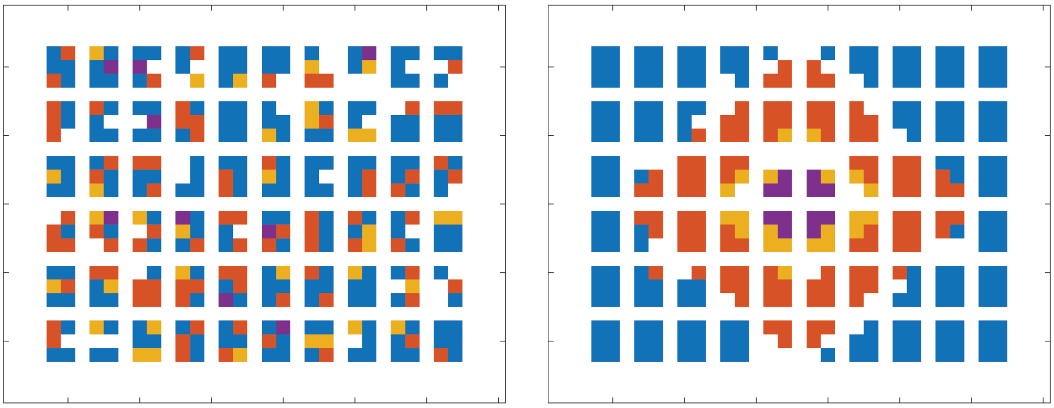

(a) (b)

Figure 4.7: Box plots as a result of a one-way ANOVA of distances of each class from the nearest P/D station for AS-CBS methods: (a) Uniform Demands. (b) Pareto's Demands.

distribution to reflect upon the placement of the classes.

The contents of queue over time in the AS-CBS was recorded (Figure 4.6) to visualize how the proportion of classes changed over time when the turn over rates were uneven. The queue seemed to reflect the demand changes sufficiently, accurately capturing when the inversion of turn over rates occurred. The mean distances of the placed items according to the Adaptive CBS method was evaluated to further investigate their effectiveness in reflecting the turn over rates.

For the AS-CBS to be effective, the item placement needs to be modulated relative to the demands, placing higher demand items nearer to the P/D points, meaning that classes 1, 2, 3, and 4 needs to be in ascending order when plotting the average distances



(a) (b)

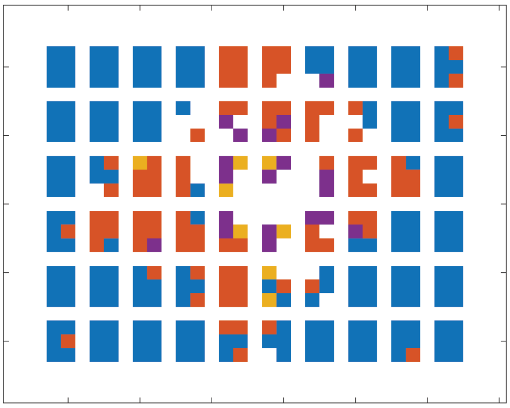
Figure 4.8: Reorganizing behavior of Fixed CBS method. (a) Initial state. (b) Final state.

(a) (b)

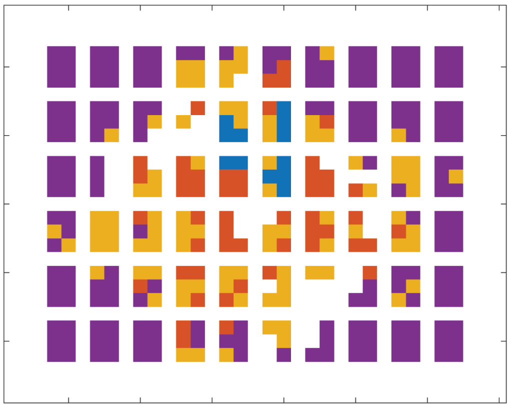
Figure 4.9: Reorganizing behavior of N-CBS method. (a) Initial state. (b) Final state.

to the P/D. As a result, Figure 4.7 shows that the placement of items under AS-CBS was modulated and corresponded to the predictions, having items of class 1 closest to the P/D, followed by 2, and 3.

A notable case is class 4, having a seemingly shorter average distance from P/D than class 3, and this is most likely due to the fact that the stochastic element played a role, and that the lower demand of class 4 sporadically placed them close to P/D. Regardless, AS-CBS also seem to perform similarly to random storage strategy when the demands of item classes are uniform, each class having similar average distance from P/D. In conclusion, analyzing the average placement distances of the different classes under uniform and Pareto demands revealed that AS-CBS effectively discriminated the different demands for each class and placed the items according to it.



(a)



(b)

Figure 4.10: Reorganizing behavior of AS-CBS method in Figure 4.6(b)'s simulation. (a) Reorganization. (b) Reorganization after the demand flip.

The primary focus of this research was to investigate the reorganization behavior of the Adaptive Stochastic Class-Based Storage (AS-CBS) method. Similarly, the reorganization behaviors of the Fixed CBS and N-CBS methods were also examined, with the initial state comprising randomly scattered classes across the storage area. Analgous to filling up the empty storage area in Figure 4.5, the transition of the storage area from an initial state of random scattering to a structured configuration was observed for the Fixed CBS method.

Each class gradually formed distinct storage areas, resulting in a static organization. In contrast, the N-CBS method displayed a less structured storage pattern, with a more dispersed arrangement of classes throughout the storage area. In comparison, the AS-CBS method exhibited a more dynamic and adaptive reorganization process. Despite the initial wide variation in the normal distribution, resulting in diverse storage geometries, the AS-CBS method consistently showcased a higher turnover rate of items closer to the picking and delivery (P/D) stations.

Moreover, the AS-CBS method demonstrated adaptiveness in response to sudden changes in turnover rate distributions. By dynamically adjusting the placement of items based on real-time data, the AS-CBS method proved its capability to efficiently adapt to varying demand patterns. As seen from Figure 4.10(b), the placement of different classes were reflected when the demand shifted according to the queue in Figure 4.6(b).

Beyond these examinations, the measured performances were evaluated such as the number of total items processed, number of total collisions, and number of total congestion occurred throughout the duration of the simulation. These evaluations

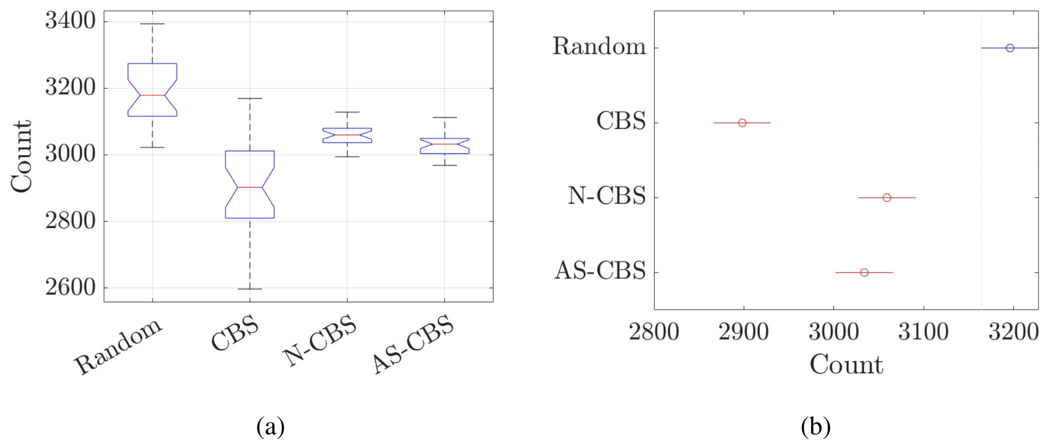



Figure 4.11: ANOVA comparison of processed volumes across storage types with uniform turnover rates: (a) Box plots. (b) Pairwise comparisons.

Table 4.1: Mean Volume and Standard Deviation for Different Storage Methods under Various Demand Patterns

| | **Uniform** | **Pareto** | **Invert Pareto** |
|---|---|---|---|
| Random | 3195.9(±106.17) | 3158.1(±159.19) | 3008.7(±205.75) |
| CBS (Petersen et al., 2004) | 2898.1(±139.48) | 2994.9(±86.72) | 2614.2(±51.22) |
| N-CBS | 3059.0(±34.73) | 3076.4(±104.15) | 2756.6(±72.11) |
| AS-CBS | 3033.9(±41.60) | 3139.6(±95.10) | 2929.0(±109.03) |

ultimately determine how the aforementioned reorganization benefitted the order picking operation. Due to the reorganization, shortened travel distances were expected that would lead to faster picking operations, leading to the warehouse being able to treat more orders for a given time.

Evaluating the total number of items processed by each storage method when the turn over rates are uniform from Figure 4.11, the random storage method seems to perform decently, because randomly scattered classes across the storage area could decrease the travel distances since all classes of items have an equal probability of being requested for storage and retrieval.

Moreover, AS-CBS was expected to behave in a similar manner to the random storage method when the turn over rates are uniform due to the fact that all classes will have the same proportion within the queue. On the flip side, the AS-CBS seems to have a narrower minimum and maximum, as well as a narrower inter quartile range. While the Fixed CBS method was not expected to perform well in this configuration, it

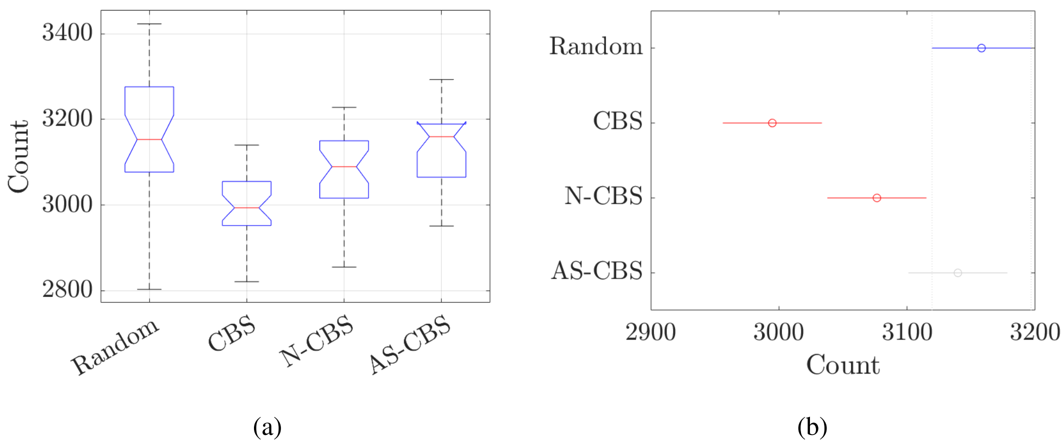



Figure 4.12: ANOVA comparison of processed volumes across storage types with Pareto's turnover rates: (a) Box plots. (b) Pairwise comparisons.

not only demonstrated overall lower mean processed number of items, it also showed wide variation.

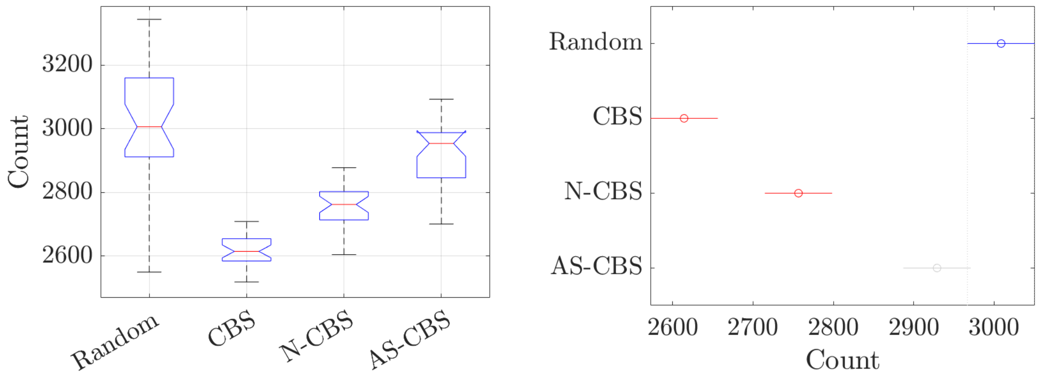
Remarkably, N-CBS was not expected to perform on par as the AS-CBS due to its lack of adaptiveness, perhaps, the stochastic nature of storage contributed to shortening travel distances by having certain degree of randomness in storage locations unlike the rigid nature of the Fixed CBS. Moving onto the pair-wise comparison of the ANOVA statistics, it was revealed that storage methods random storage, N-CBS, and AS-CBS do not have a significant difference in mean number of total items processed, while the Fixed CBS underperformed compared to all other methods.

Building upon the preceding observations, the investigation into turnover rates following Pareto's distribution revealed intriguing findings, particularly highlighting the comparative performance of the Fixed CBS method. As illustrated in Figure 4.12(b) similar trends to those observed in Figure 4.11(b), emerged, with random storage, N-CBS, and AS-CBS displaying minimal differences in mean values.

This outcome was somewhat unexpected, as the Fixed CBS method was initially anticipated to excel in this configuration. However, it failed to meet expectations, indicating potential limitations in its adaptability to varying turnover rate distributions.

Moreover, while the random storage method exhibited substantial variability in performance, as evidenced by the wide minimum and maximum differences and interquartile ranges shown in Figure 4.12(a), the other three methods showcased more consistent results.

To further probe the adaptability and robustness of the Adaptive Stochastic Class-




(a) (b)

Figure 4.13: ANOVA comparison of processed volumes across storage types with dynamic Pareto's turnover rates: (a) Box plots. (b) Pairwise comparisons.

Based Storage (AS-CBS) method in response to dynamic turnover rates, an inversion of turnover rates was introduced. This inversion served as a means to simulate scenarios where items with historically low turnover rates suddenly experience high demand, thereby challenging the adaptiveness of storage methods.

As illustrated in Figure 4.13(b), the analysis revealed intriguing insights into the comparative performance of the storage methods. Surprisingly, both the random storage method and AS-CBS exhibited mean total items processed statistically comparable to each other, outperforming both Fixed CBS and N-CBS. However, a notable disparity emerged when examining the variability of performance across the methods, as depicted in Figure 4.13(a).

In particular, the random storage method displayed a significantly wider range of performance variability, as evidenced by the stark contrast between the minimum and maximum values and the interquartile range. This variability suggests that while the random storage method and AS-CBS may yield similar mean performance outcomes, the consistency and reliability of AS-CBS may surpass that of the random storage method.

This finding underscores the importance of not only considering mean performance metrics but also evaluating the stability and consistency of performance across diverse scenarios.

As mentioned previously, another impoertant element that could potentially influence the performance of order picking is collision among agents and consequent congestion. To address how different storage strategies influence the behaviors of agents,

Table 4.2: Mean Collision Count and Standard Deviation for Different Storage Methods under Various Demand Patterns

| | Uniform | Pareto | Invert Pareto |
|---|---|---|---|
| Random | 14944(±1700.3) | 15962(±1287.2) | 18456(±3304.2) |
| CBS (Petersen et al., 2004) | 17158(±3243.3) | 15202(±944.8) | 15972(±874.0) |
| N-CBS | 14049(±523.6) | 14435(±496.1) | 15116(±332.0) |
| AS-CBS | 13563(±604.5) | 14119(±352.5) | 15468(±918.2) |

Table 4.3: Mean Congestion Count and Standard Deviation for Different Storage Methods under Various Demand Patterns

| | Uniform | Pareto | Invert Pareto |
|---|---|---|---|
| Random | 2469.5(±391.97) | 2584.2(±534.71) | 3353.8(±1003.9) |
| CBS (Petersen et al., 2004) | 2758.1(±942.01) | 2264.2(±306.26) | 2613.0(±197.9) |
| N-CBS | 2001.9(±78.27) | 2319.3(±114.50) | 2381.6(±90.3) |
| AS-CBS | 1890.2(±49.94) | 2140.6(±110.58) | 2315.4(±87.3) |

ANOVA was performed under uniform turn over rates.

Similarly to the previous observations in total number of items processed, Fixed CBS method yielded the most collision and congestion compared to other methods, meanwhile N-CBS and AS-CBS resulted in narrow range of number of both collisions and congestion as seen from Figure 4.14(a) and (c).

The previously observed underperformance of the Fixed CBS method might be explained by the increased variance in collision and congestion counts, where the rigid nature of the storage led to bottlenecks and points of slow downs.

In contrast to previous observations, where the Fixed CBS method exhibited the widest variance in the number of collisions and congestion, the random storage method demonstrated the widest variability in these metrics when turnover rates followed a Pareto distribution. This notable variability may stem from the uneven distribution of turnover rates, which can lead to increased travel distances and a heightened probability of collisions, particularly when items with low turnover rates are sporadically requested.

As depicted in Figure 4.15(b), the implementation of AS-CBS resulted in a statistically significant decrease in the number of collisions, indicating its efficacy in mitigating collision incidents within the warehouse environment. However, intriguingly,

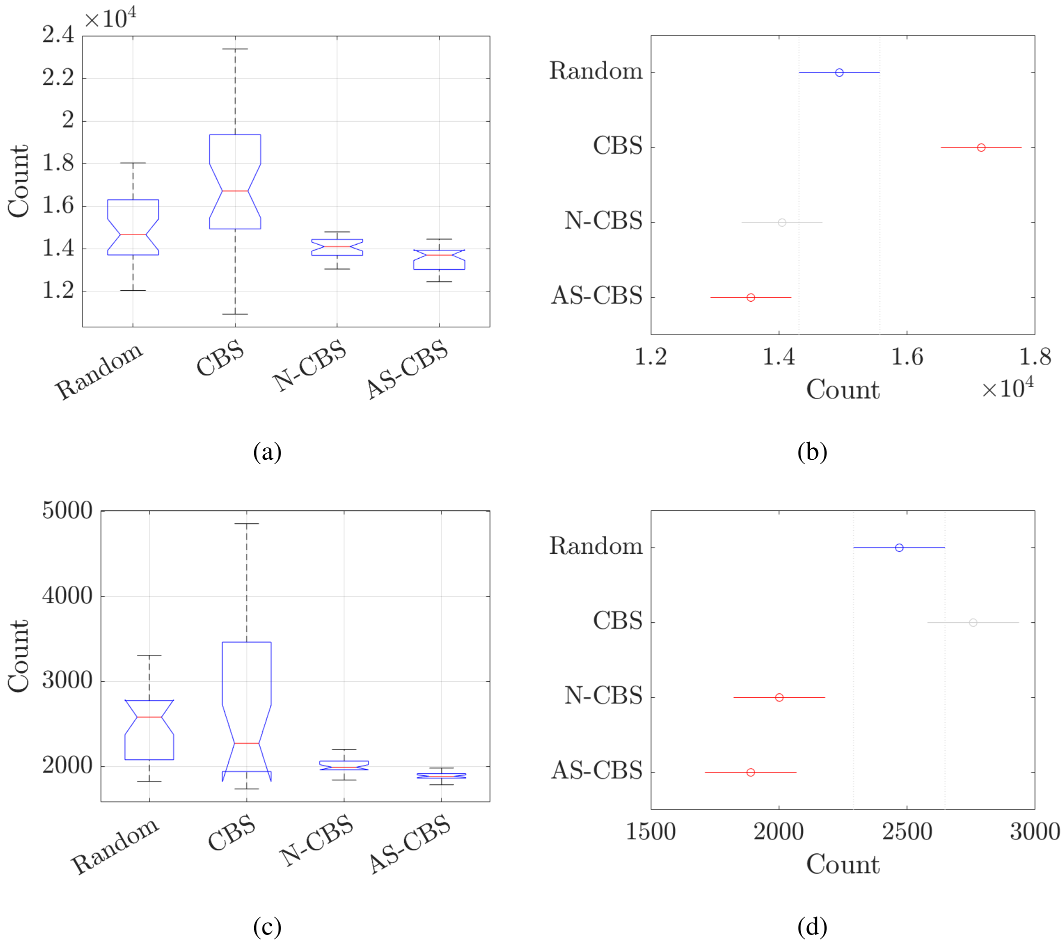



Figure 4.14: ANOVA comparison of collision and congestion counts across storage types with uniform turnover rates: (a) Collision box plots. (b) Collision pairwise comparisons. (c) Congestion box plots. (d) Congestion pairwise comparisons.

this reduction in collisions did not necessarily translate to a corresponding decrease in congestion levels, as illustrated in Figure 4.15(d).

Similarly to the preceding case characterized by uneven turnover rates, the random storage method exhibited significant variability in the number of collisions and congestion, as evident from the data presented in Figure 4.16(a) and (c). Once again, both the N-CBS and AS-CBS methods displayed notable effectiveness in managing collision incidents and congestion levels.

However, a particularly intriguing observation emerged regarding the Fixed CBS method. In contrast to its performance in the previous scenario, where it demonstrated wide variance in collision and congestion counts (as depicted in Figure 4.15(a) and (c)), the Fixed CBS method exhibited a reduction in variance in both metrics. This unexpected finding may be attributed to the unique characteristics of the turnover rate

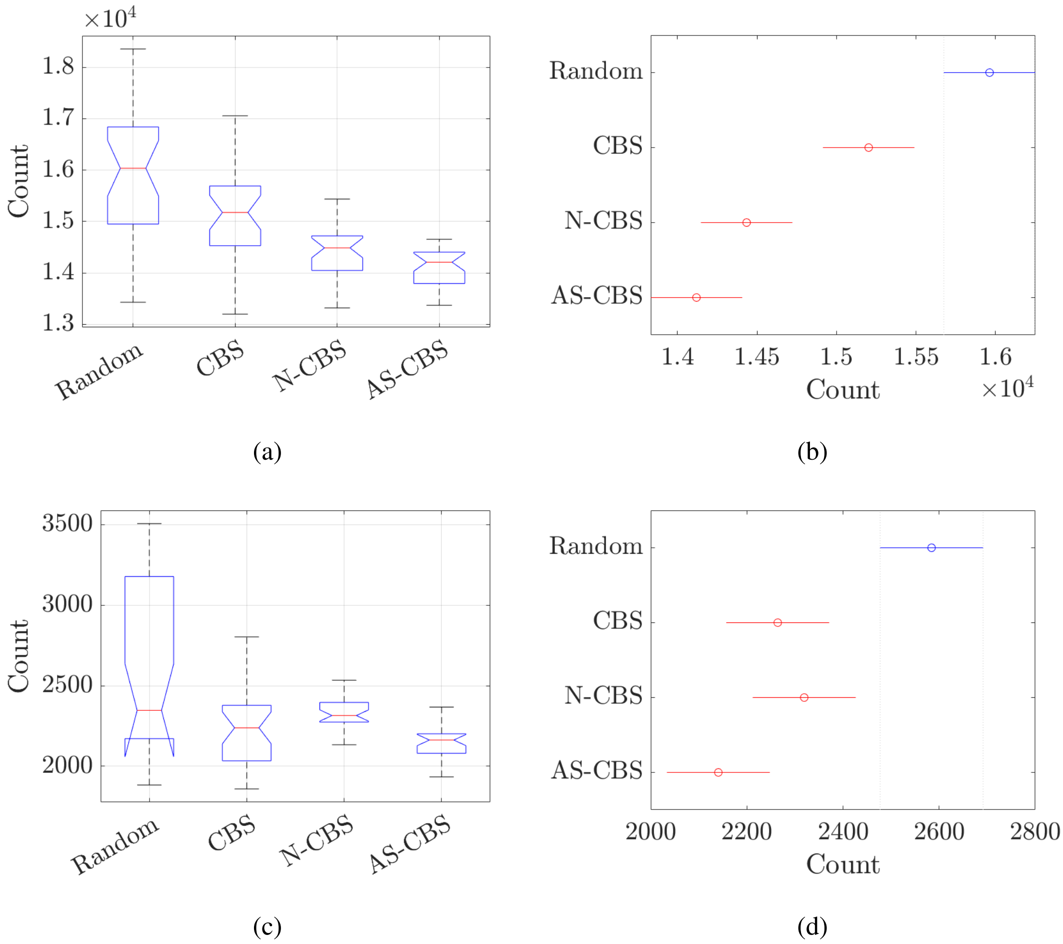



Figure 4.15: ANOVA comparison of collision and congestion counts across storage types with Pareto's turnover rates: (a) Collision box plots. (b) Collision pairwise comparisons. (c) Congestion box plots. (d) Congestion pairwise comparisons.

inversion scenario. The inversion of turnover rates likely played a role in alleviating bottlenecks within the warehouse environment.

Specifically, the increased demand for items located at the center of the map, where the inverted turnover rate is highest, necessitates longer travel distances for picking agents. Consequently, this extended travel distance offers more alternative routes and pathways in the event of congestion or collision incidents, thereby reducing the likelihood of bottlenecks occurring.

## 4.3 Summary of Chapter 4

This chapter introduced and evaluated the Adaptive Stochastic Class-Based Storage (AS-CBS) method, demonstrating its effectiveness in dynamically optimizing ware-

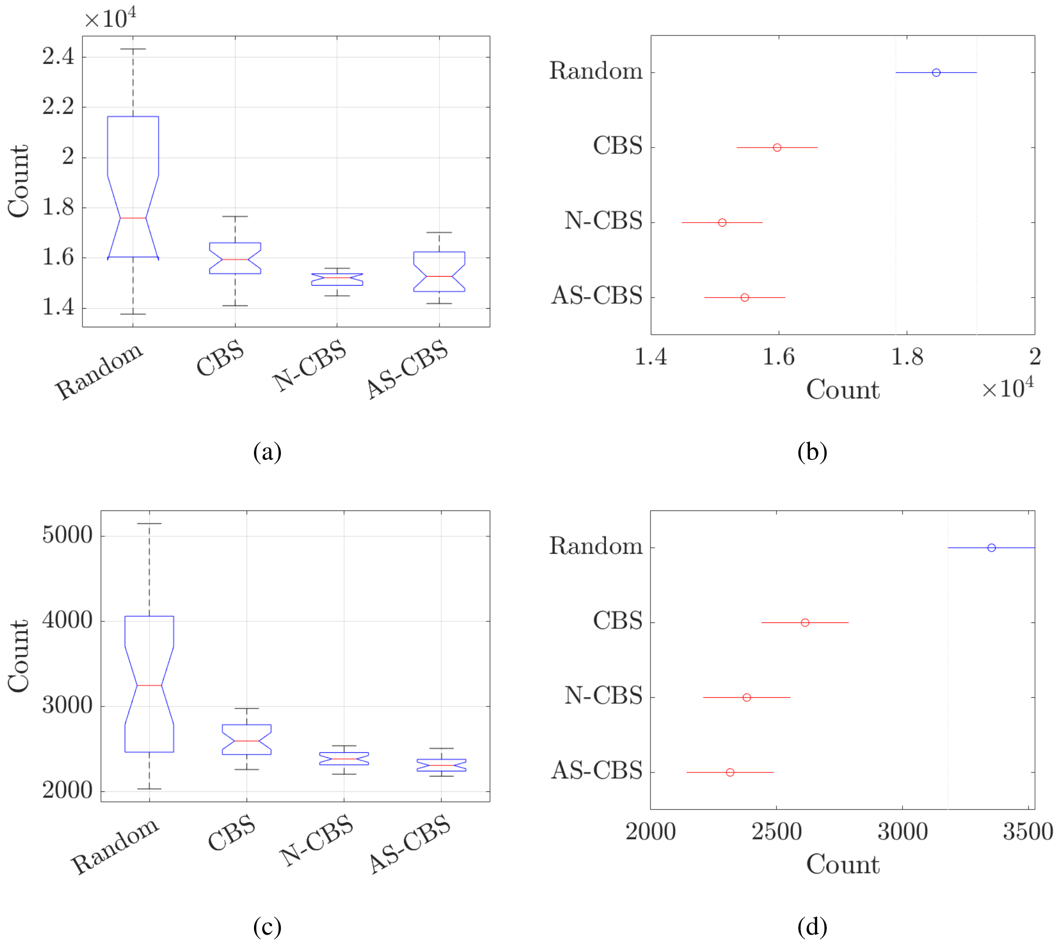



Figure 4.16: ANOVA comparison of collision and congestion counts across storage types with dynamic Pareto's turnover rates: (a) Collision box plots. (b) Collision pairwise comparisons. (c) Congestion box plots. (d) Congestion pairwise comparisons.

house storage locations. The research revealed that AS-CBS successfully adapted storage locations based on real-time demand patterns, demonstrating robust performance across uniform, Pareto, and dynamic demand scenarios. The method achieved notably consistent performance with lower variance compared to random storage, while maintaining higher throughput during demand pattern changes compared to fixed methods.

Analysis of storage behavior showed that AS-CBS effectively modulated storage distances based on item demand, displaying appropriate stochastic behavior when demands were uniform and successfully reorganizing storage patterns in response to demand changes. The queue-based demand estimation proved particularly effective for adapting to changing patterns, providing a reliable mechanism for dynamic storage optimization.

Comparative analysis revealed distinct characteristics among different storage methods. Random storage showed high variability but good mean performance, while Fixed CBS surprisingly underperformed in dynamic scenarios despite its theoretical advantages. N-CBS achieved intermediate performance but lacked adaptability, highlighting the unique advantage of AS-CBS in combining the benefits of both random and structured storage approaches.

In terms of operational metrics, AS-CBS showed consistent improvement in collision reduction across all demand patterns while achieving lower congestion counts compared to other methods. The method maintained competitive processing volumes while reducing disruptions, demonstrating superior performance particularly during demand pattern changes. This balanced performance across multiple metrics distinguishes AS-CBS from other approaches.

Implementation considerations revealed several important factors. The queue size significantly impacts adaptation speed and stability, and the method requires a warm-up period for optimal performance. However, once established, AS-CBS successfully balances deterministic and stochastic storage behaviors without requiring extensive parameter tuning. The research establishes AS-CBS as a viable approach for dynamic warehouse environments, offering improved operational stability and adaptation to changing demand patterns while maintaining competitive throughput performance. The method's ability to combine structured storage benefits with stochastic flexibility makes it particularly suitable for modern warehouse operations with varying demand patterns.

# Chapter 5

# Robotic Application of Congestion Detection and Adaptive Storage Strategy

In this chapter, a robotic experiment is devised to investigate the feasibility of real life adaptation of proposed path finding algorithm and adaptive storage policy proposed in chapter 3 and 4. This experiment serves as a validation for core mechanics of the proposed methods, while further exploring the discrepancies between simulation and robotic implementation. It also aims to find underlying factors in observed results and identify the challenges of implementation. The experiment also serves as a pioneering endeavor to reveal the potential solutions to overcome the identified challenges.

The experiment aims to discover these by measuring the performance of the algorithms in number of task completion, navigation completion, and collision counts. As a result, the challenges of robotic implementation such as dependency on reliable visual acquisition and communication have been discovered, and also the limitation of scaled down experiment in measuring the performances. Regardless, meaningful evidences of performance increase due to novel path finding and storage algorithm have been discovered and this research serves as a first step to practical implementation in real-world robotic fulfillment centers. This chapter's contents are available on arXiv and is in preparation for submission to Swarm and Evolutionary Computation (Ryu et al., 2024b).

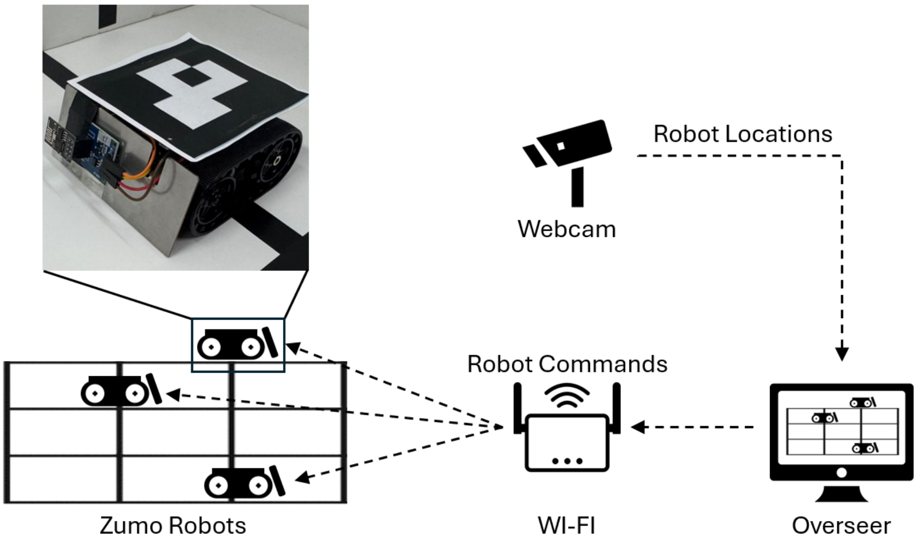



Figure 5.1: Schematic of the experiment setup.

## 5.1 Method

A real-world simulation consisting of robots was conducted to discover the effectiveness of the previously devised Agent Density Based Routing heuristics and Adaptive Stochastic Class Based Storage (AS-CBS) (Ryu et al., 2024b; Ryu et al., 2024a). The robots within the simulation obey the rules identical to the MATLAB simulation and operate on a grid environment and but simplified and scaled.

The robots move in four cardinal directions, one step at a time, and a node shall not quarter more than one unit. The robotic platform is a Zumo Shield V1.2 developed by Pololu, which is controlled by an Arduino UNO R3 unit (Pololu Corporation, 2024; Arduino LLC, 2024). This platform was chosen due to its compactness and integrated sensor and motor control suite, and uniform nature of operation. This platform is limited by its inconsistency of movement due to the tracked design, thus requiring external observer to correct its movement and location.

Because of its limited capabilities, virtual navigation is not applicable, but instead a physical method to also correct the movement. Though many businesses are now transitioning to virtual navigation such as SLAM (Simultaneous Localization and Mapping) and dead-reckoning, tags and line followers are still widely in use. Zumo Shield's line following capabilities were utilized when devising a physical navigation environment due to its course correcting nature to compensate for the inconsistent movement by using the integrated infrared sensor array.

To coordinate multiple Zumo Shields, each robot needed to be identified and remotely controlled. This feature was enabled by marking the robots with an ArUco markers to determine their orientation and location on the grid, while offering robust and quick recognition with simplicity (Garrido-Jurado et al., 2014). These markers also serve as reference points to define the boundaries of the simulation environment and calibration of image processing.

For wireless control of the Zumo Shield robots, three common methods are used in practice: radio, bluetooth, and WIFI, and, WIFI was chosen because of its availability at the time. Each unit houses an ESP8266 WIFI module connected to an adapter for the Arduino UNO R3, and offer adequate connectivity.

Normally, the ESP8266 WIFI modules need additional circuitry to be compatible with Arduino UNOs, due to them taking 3.3V power while Arduino UNO supply 5V, so third party adapters with built in resistors and circuits were applied to eliminate the necessity of additional work and to achieve compactness.

While the Zumo Shield already comes equipped with necessary pins and features to accommodate Arduino UNOs, the WIFI module had to be installed. The pins of an Arduino UNO are still usable despite being connected to the Zumo Shield. Fig B shows how an ESP8266 is supposed to be connected to an Arduino UNO, with pins 6 and 3 from Arduino UNO each connected to TX and RX of ESP8266, and 5V and GND connected to the VCC and GND via jumper cables.

The Zumo Shields operate by four AA sized batteries, and it was discovered that the robots' performance highly depend on the voltage they provide. Therefore, traditional alkaline batteries with a gradual voltage descent with repeated use presented challenges by affecting the consistency of WIFI connectivity and motor power, often leading to errors and halts during simulations.

As a solution, Nickel-Metal-Hydrogen (NiMH) rechargeable AA batteries replaced the alkaline power sources due to their consistent voltage output despite a slight drop (1.5V vs 1.2V). While the NiMH batteries did not deliver enhanced run times, however, consistent and reliable operation regardless of repeated use.

The grid environment consists of 4 by 3 nodes, and the grid cells correspond to nodes where the lines intersect. Zumo Shield robots will move from a node to another at a time if available, but moving to an already occupied node is prohibited, similar to the simulation environment. The robot occupies the beginning node and the destination node until its traversal is completed, and this effectively prevents most of the conflicts defined by Stern (Fig 2.9).

If the desired destination node is not vacated, a robot does not move, and if a robot encounters a swapping conflict, one of the robots involved in the conflict will evacuate to a nearest empty node to unblock the path. Determining which robot shall evacuate is determined randomly, and in case where a robot is surrounded and cannot find an adjacent empty node, it will trigger one of the surrounding robots to evacuate, and move into its node.

As demonstrated in the previous chapter 3, this method was proven to eventually solve any congestion. The construction of the path to the destination is achieved by using A* algorithm, and utilized Manhattan Distance heuristics to minimize turns and promote construction of straight paths.

The Zumo robots and an overseeing computer exchange information for operation, and establishes a two-way communication via WIFI. The role of the overseer is to observe the movements of the robots and correct their path, send commands to robots to reach their destination, and calculates and plans paths for each robot.

The map is perceived as a grid for the overseer and the operation of robots are marked on the grid. Consequently, visual recognition of the robots become and important aspect of the operation, and a top-view camera identifies the ArUco markers to find the location of robots, as well as their heading direction.

The equipped camera operates at 1280 by 720 resolution for faster recognition and processing, and image adjustments such as color/contrast adjustments, warping, and cropping are applied to correctly identify the markers. With ArUco markers placed at the edges of the grid environment, the overseer can correctly identify the intersections of the grid, as well as the robots regardless of how the grid was deployed.

The nodes are identified by the overseer and marked red, and distortions are corrected. For the top-view camera, a generic Universal Serial Bus (USB) webcam was used, presenting the challenge of inconsistent capture rates and low resolution.

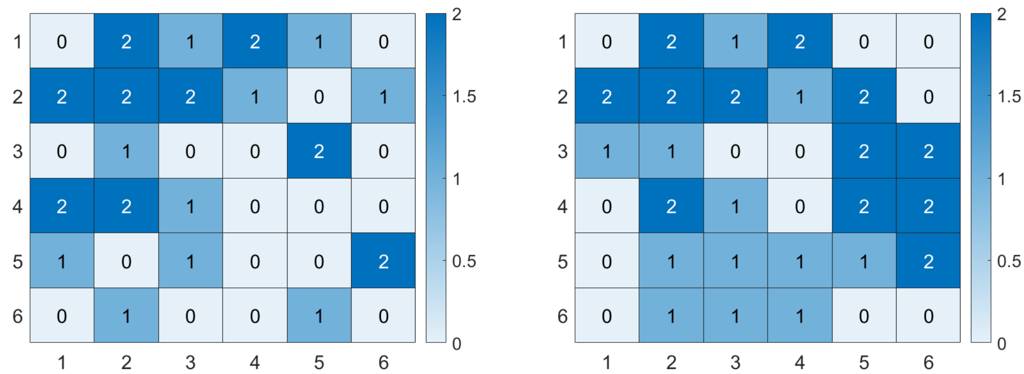
Meanwhile the issue of having a low resolution was solved by image processing, the inconsistent capture rates and shutting down during a prolonged usage was not solved completely, thus the overseer had to wait for completion signals from all of the robots and often attempt multiple times to capture a correct image.

All in all, the overseer first ensures the completion of all robot operations, identifies the robots, calculates the next move for each robots, then sends out commands. The following pseudo code explains the operation of the overseer.

Unlike the original computer simulation, experimenting with real robots presented multiple unforeseen challenges due to physical and software constraints. Consequently,

**Algorithm 2** Overseer algorithm

```
while running do
    while !all complete do
        for all robots do
            check completion signal received
        end for
    end while
    visual robot recognition
    for all robots do
        determine next move
    end for
    send commands for robots
end while
```



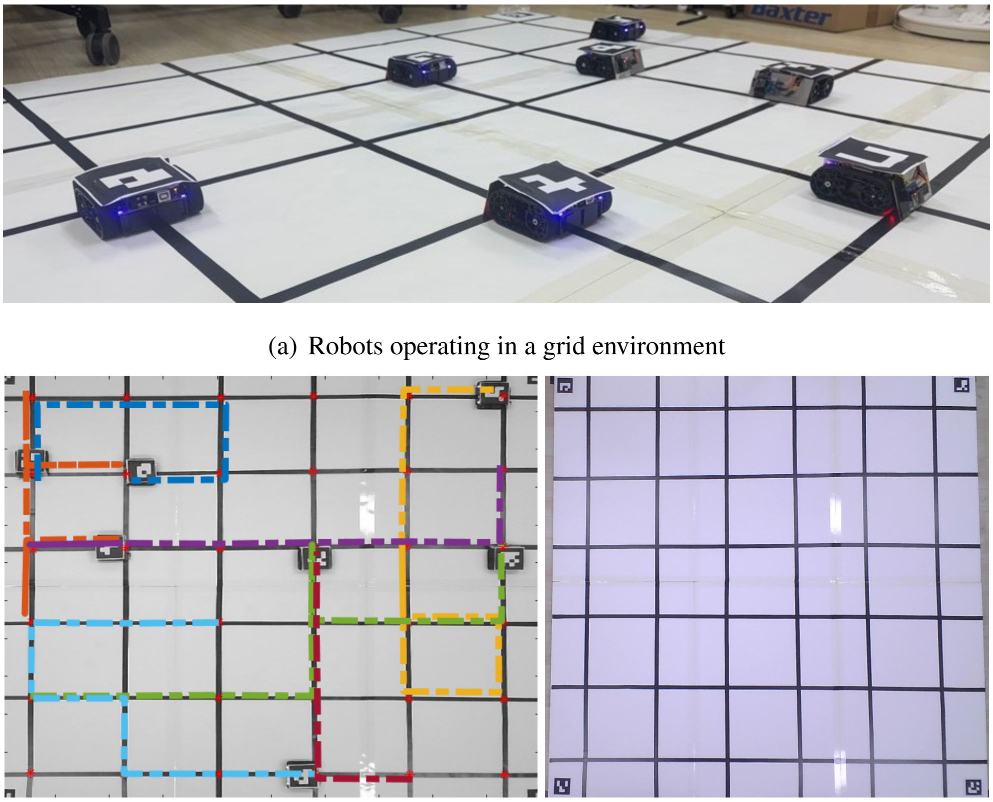

(a) (b)

Figure 5.2: Placement of items on the six-by-six grid. (a) Randomly placed items. (b) Items re-organized according to AS-CBS.

modifications such as sending commands collectively after coordinating moves for robots instead of each robot operating in serial.

Nevertheless, the grand scheme of the experiment does not deviate largely from the computer simulation, incorporating path reconstructing algorithm and congestion avoidance based on convolutional congestion detection.

Building upon the evaluation of different path finding algorithms, the effect of Adaptive Storage Strategy is further evaluated. To account for the smaller scope of the experiment, the number of classes were decreased to two classes along with a smaller queue size for estimating the input demand.



(a) Robots operating in a grid environment

(b) Path drawn on the perceived view by the overseer

(c) 6 by 6 line following grid with corner ArUco markers for calibration

Figure 5.3: Images of the Robot Experiment Environment.

In this case, the completion is measured by a dual order cycle, where a robot starting from a corner begin by placing a cargo at a designated spot, then travels to another location to retrieve a cargo back to the corner. Similarly, robots that do not start at the corner will begin with carrying a cargo to a location. This set up emulates a situation where robots are continuously working in a cycle, as the P/D point at the corners can only have one robot at a time.

Same as the path finding experiment, the runtime was ten minutes, and the number of repetition was three per configuration. The items classified into either class 1 or 2 were randomly placed in the random setting, while the AS-CBS setting followed the placement determining calculation according to its algorithm.

The placement of items for each setting are shown in Figure 5.10, and illustrates the placement of items by demand. Because of the setup replicating a continuous operation, the queue was provided pre-filled before commencing the experiment according to an uneven distribution, with class 1 comprising 66.6 percent and class 2 taking 33.3 percent of the total demands. As a result, the placement of different classes were mod-

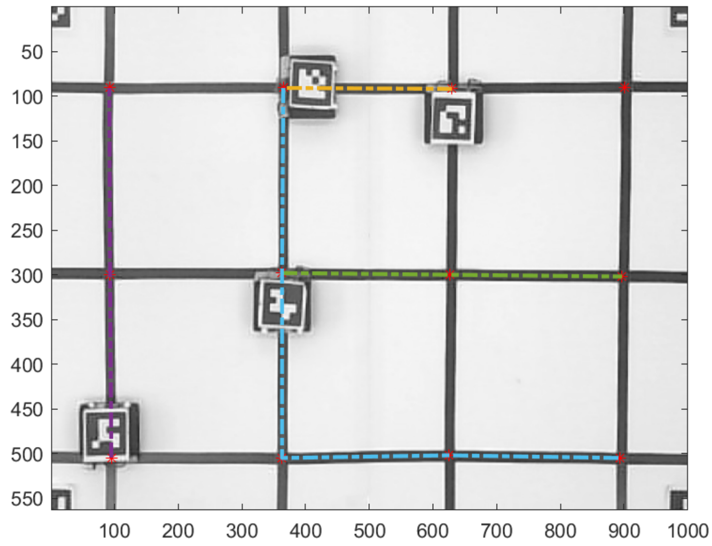



Figure 5.4: Line following grid environment with four Zumo Robots. Their paths to their respective destination are drawn.

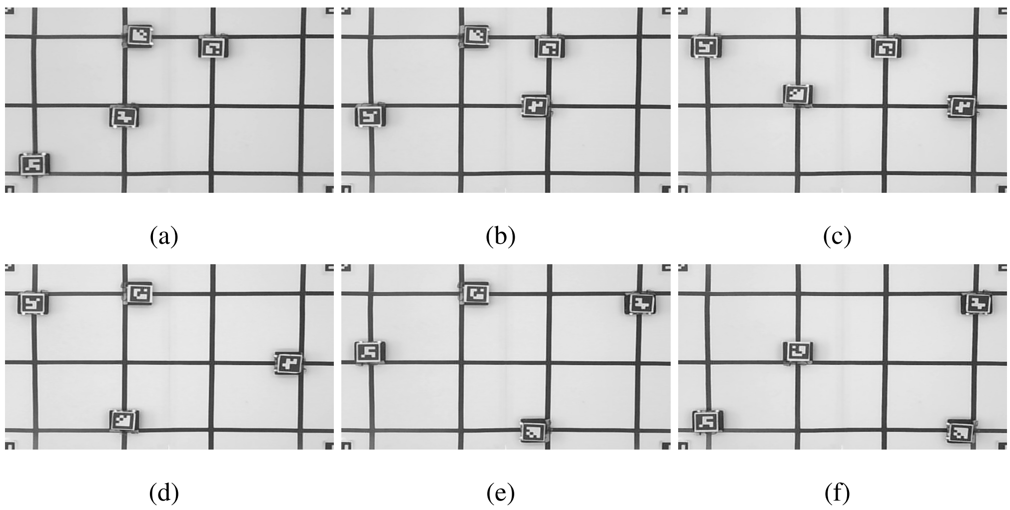


Figure 5.5: Movement of robots to their destination. Purple and green reach their destination at (c), yellow reaches its destination at (d), and blue reaches its destination at (f).

ulated according to the AS-CBS calculation.

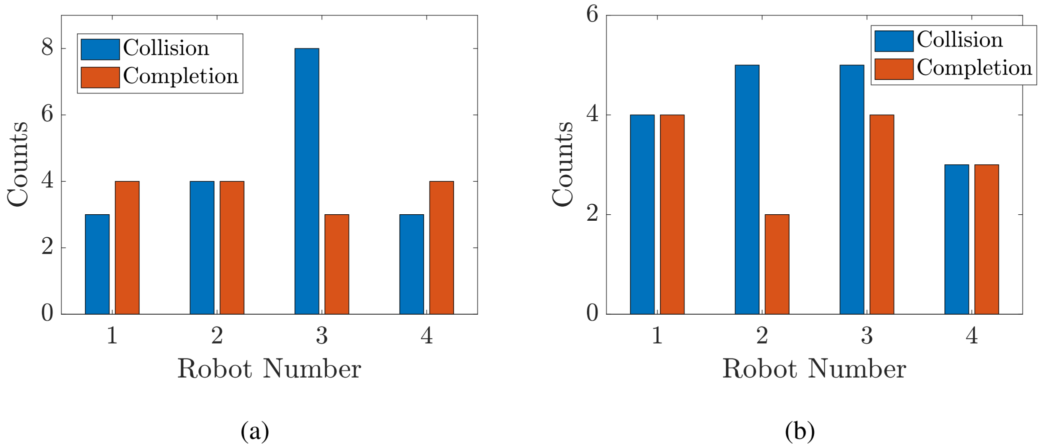



Figure 5.6: Bar plot of collision counts marked blue and completion counts marked red for four robots. (a) Without congestion avoidance. (b) With convolutional congestion avoidance.

## 5.2 Results

Fig 5.4 shows the destination and an A* path constructed by the overseer overlaid upon the captured top-view image. One may notice that some robots are expected to have blockage when following through the path.

Fig 5.5 demonstrates the process of robot movements by each step, where the robots wait and avoid conflicts while moving along the path. Some robots are seen exploring different paths after reaching the initial destination because the overseer produces a new destination if the robot reaches the end.

Initially, three settings have been devised for robotic experiments on a same 3 by 4 grid, with two, three, and four robots, each with and without convolutional congestion avoidance and path reconstruction.

Because of the simplifications made, notably, without cargo collection akin to a warehouse situation, collisions and completion to the destination became key metrics in determining the effectiveness of navigation. Additionally, due to being a robotic environment in real life, the absolute time it takes for navigating through the grid to the destination have been recorded.

Having two robots did not yield significantly meaningful data, with the simulation sometimes but rarely ending without a single collision. Meanwhile, as seen in Fig 5.7, completion count is usually higher than the collision counts when the number of robots is relatively small, but still the collision count increased when compared to having only

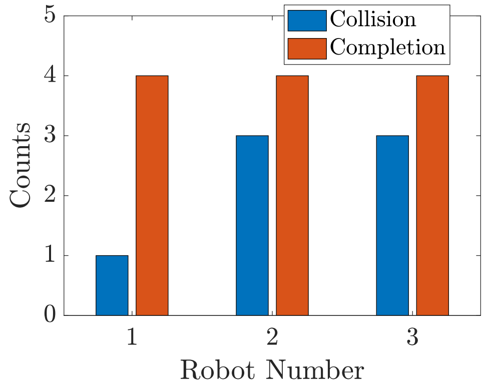



Figure 5.7: Bar plot of collision counts marked blue and completion counts marked red for an experiment with three robots.

Table 5.1: Table of average time (in seconds) in between each collision (first row) and in between reaching a destination (second row) for default configuration.

| Robot 1 | Robot 2 | Robot 3 | Robot 4 |
|---|---|---|---|
| 25.4218 | 26.8991 | 15.3905 | 23.0658 |
| 26.8989 | 30.7806 | 25.4218 | 28.9245 |

two robots.

On the other hand, the collision count increased significantly in contrast to the completion count regardless of having a congestion avoidance, as demonstrated in Fig 5.6 by the blue bars. It seems that having four robots would induce more collision in a confined environment such as a 3 by 4 grid, and the contrast between having an algorithm and without should become clear.

As seen from Fig 5.6, A's robot number 3 has a significantly higher collision count compared to other robots, indicating that it was struggling to find a new path that avoids congestion, and likely got stuck until other robots cleared the path. In contrast, B shows a more evenly distributed number of collision across the four robots.

The Overall average time between collision and reaching a destination is 22.6943 seconds and 28.0065 respectively in default configuration, and 25.7924 seconds and 34.8896 seconds when convolution based congestion avoiding measures were taken.

As far as we know, there seems to be lack of significant differences in performance in between these two configurations, and this is more evident when comparing the average number of collision and completion for each robot in these two settings. The

Table 5.2: Table of average time (in seconds) in between each collision (first row) and in between reaching a destination (second row) for congestion avoidance configuration.

| Robot 1 | Robot 2 | Robot 3 | Robot 4 |
|---|---|---|---|
| 29.4418 | 16.7174 | 22.1075 | 34.9031 |
| 27.6342 | 52.3541 | 17.4000 | 42.1703 |

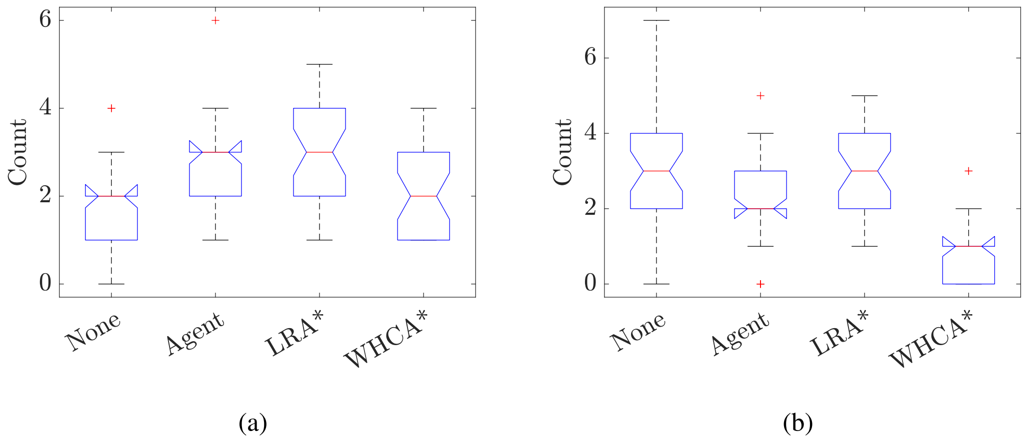



Figure 5.8: Boxplots for larger scale six-by-six grid with seven robots with different algorithms applied. (a) Volume, or number of path planning completion. (b) Number of collisions.

average number of collision is 4.5 and the average completion count is 3.75 for a single robot when no congestion avoiding measures were implemented, and 4.25 and 3.25 respectively for its congestion avoiding counterpart.

There certainly seems to be possibility that these two settings yield different results, however, unlike the computer simulation spanning 40000 steps of duration on a 30 by 40 grid with 20 to 50 agents, the experiment is conducted on a relatively smaller scale and due to the limited duration and size of the experiment, the difference may not be discernible. Further experiments were conducted to evaluate the effectiveness of various routing heuristics.

From Table 5.5 and Figure 5.8, the Agent Based path planning resulted in a balanced improvement in path planning completion and collision counts. In comparison to the baseline "None" configuration, the collision was reduced from 3.142(±1.396) to 2.285(±1.177), while simultaneously achieving a 43-percent improvement of completion count to 2.571(± 1.119).These enhancements in both collision and completion are

Table 5.3: Comparison of different algorithms, mean and standard deviation of collision and number of completion (volume).

| | **Collision** | **Volume** |
|---|---|---|
| None | 3.142(±1.396) | 1.800(±0.9331) |
| Agent Density | 2.285(±1.177) | 2.571(±1.119) |
| LRA* | 2.885(±1.078) | 2.971(±1.248) |
| WHCA* | 0.885(±0.8321) | 1.971(±0.9544) |

particularly noteworthy when examining the performance trade-offs exhibited by other methods.

Compared to WHCA* that also showed improved performances, WHCA* showed better improvements in reducing the collision, yet came at a cost of reduced completion performances. On the other hand, LRA* resulted in the highest completion counts at 2.971(±1.248) but showed minimal improvement in collision reduction at 2.885(±1.078), compared to the "None."

In can be said that the Agent Based Method found the compromise in between the extremes of these two methods, similar to the observation in simulation experiment. Additionally, the Agent Based Method demonstrate a tighter variation compared to the WHCA* and LRA*, showcasing consistent performance that indicate a reliable and stable path planning.

The remarkable aspect of the Agent Based Method's results is the balanced improvement of both the collision and the completion count without the extreme trade offs that other methods exhibit. The consistency in performance across different trials, combined with the substantial improvements in both safety and completion metrics, suggests robust operation even when facing real-world implementation challenges such as sensor noise, communication delays, and physical constraints.

Furthermore, the balanced improvements are particularly promising in real world operations when the two aspects are crucial considerations, such as in warehouse automation, manufacturing environments, or service robotics applications.

Comprehensive analysis with respect to time was conducted to reveal how the different path planning methods perform when implemented into robots. All methods except for the "None" completed its path to the goal for all 35 path planning tasks. While it yielded a reasonably good success rate at 97.1-percent, there were cases where a robot failed to reach its destination in time.

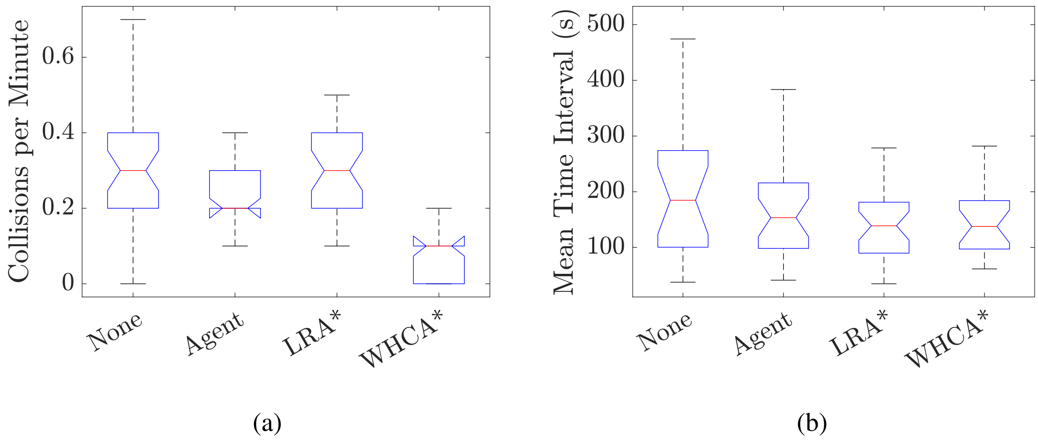



Figure 5.9: Boxplots for six-by-six grid with seven robots with different algorithms applied. (a) Collisions per minute. (b) Interval between completing a path.

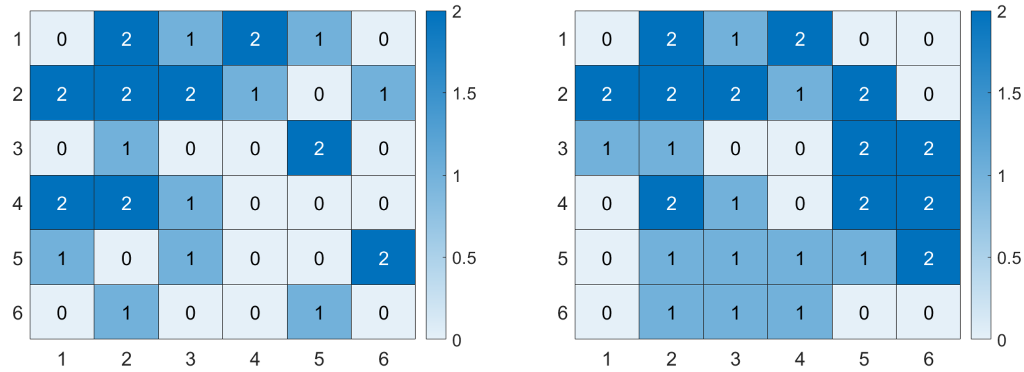
Table 5.4: Performance Metrics of Different Path Planning Methods. Best values are in bold.

| **Method** | **Collisions per min** | **Success Rate** | **First Completion (s)** | **Mean Interval (s)** |
|---|---|---|---|---|
| None | 0.31 ± 0.14 | 97.1% | 184.96 ± 124.70 | 196.64 ± 127.79 |
| Agent | 0.23 ± 0.12 | 100% | **144.50 ± 108.82** | 172.30 ± 108.19 |
| LRA* | 0.29 ± 0.11 | 100% | 165.25 ± 122.94 | 157.43 ± 105.52 |
| WHCA* | **0.09 ± 0.08** | 100% | 211.80 ± 152.05 | **147.96 ± 58.82** |

To surmise, all additional measures seem to be successful at avoiding congestion and finding an alternate route. The results demonstrate that the Agent Density Based Method provided similar achievement of balancing the collision and completion performances to the simulation results.

In the simulation results, the Agent Based Method resulted in a meaningful reduction of collision while increasing the throughput (volume) performance of the P/D process. This was reflected in the robotic experiment, while WHCA* showed the lowest collision rate (0.09 ± 0.08 per minute), this comes at a significant cost of longer completion times (mean time 211.80s to first completion) while the Agent Based Method decreased the collision rate to 0.23 ± 0.12 per minute, compared to having none at 0.31 ± 0.14, while having the shortest completion time performance of 144.50s.

This translate to 25-percent faster than baseline (none) and 35-percent faster than WHCA*. The collision frequency was lowered by 26-percent than the baseline, and




(a) (b)

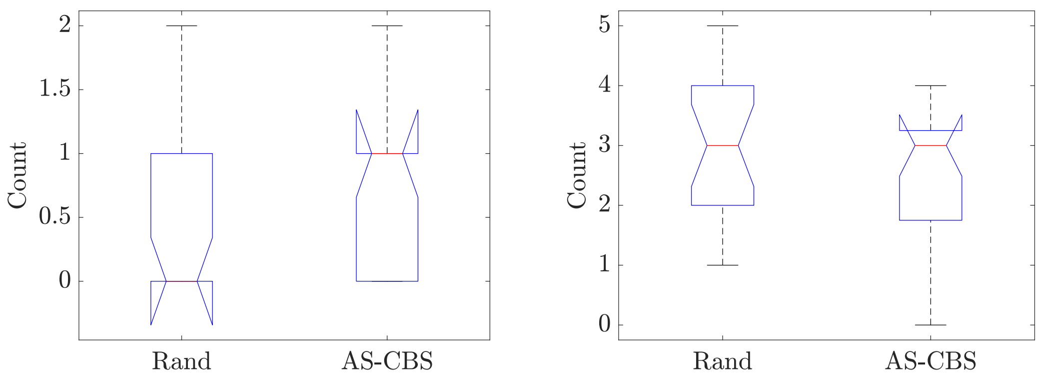
Figure 5.10: Placement of items on the six-by-six grid. (a) Randomly placed items. (b) Items re-organized according to AS-CBS.

the reliability of path planning was not compromised. In continued experiment, the Agent Based method also showed improved completion interval against the baseline (172.30s vs 196.64s).

The observation of lacking completion to the goal in WHCA* aligns with the simulation results where it sacrificed the navigational performance that resulted in a lower processed item volumes.

The effect of Adaptive Storage Strategy is further evaluated with a robotic experiment. To account for the smaller scope of the experiment, the number of classes were decreased to two classes along with a smaller queue size for estimating the input demand. In this case, the completion is measured by a dual order cycle, where a robot starting from a corner begin by placing a cargo at a designated spot, then travels to another location to retrieve a cargo back to the corner. Similarly, robots that do not start at the corner will begin with carrying a cargo to a location. This set up emulates a situation where robots are continuously working in a cycle, as the P/D point at the corners can only have one robot at a time.

The runtime was ten minutes, and the number of repetition was three per configuration. The items classified into either class 1 or 2 were randomly placed in the random setting, while the AS-CBS setting followed the placement determining calculation according to its algorithm. The placement of items for each setting are shown in Figure 5.10, and though limited, illustrates the placement of items by demand. In Figure 5.10(a), the randomly placed items were reorganized into B with AS-CBS in effect, locating class 1 items closer to the corners where the P/D points were defined.




(a) (b)

Figure 5.11: Boxplots for larger scale six-by-six grid with seven robots with different storage strategies applied. (a) Volume, or number of cycle completion. (b) Number of collisions.

Table 5.5: Comparison of different algorithms, mean and standard deviation of collision and number of completion (volume).

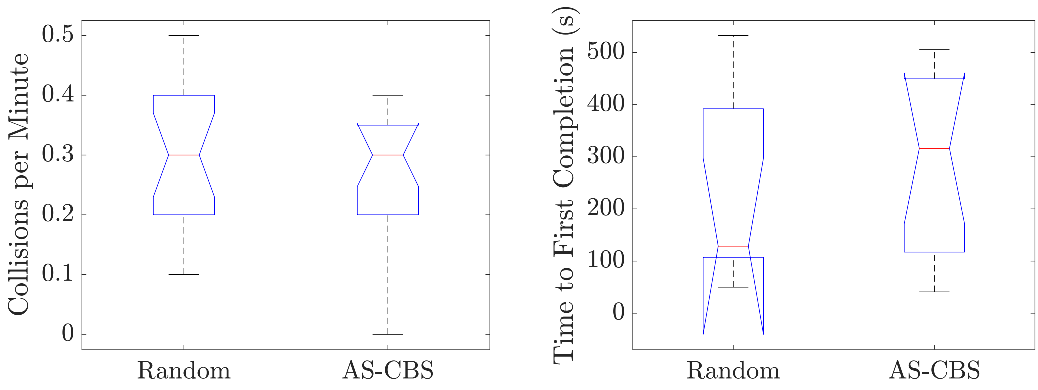
| | **Collision** | **Volume** |
|---|---|---|
| Random | 3.000(±1.183) | 0.3810(±0.5896) |
| AS-CBS | 2.524(±1.289) | 0.7619(±0.6249) |

The resulting reorganization do not seem as apparent as the simulation results due to the scope of the robotic experiment and its shortened duration. Nevertheless, increased picking and delivery activity of class 1 items was observed, as well as placement of class 2 items further from the P/D points.

Because of the setup replicating a continuous operation, the queue was provided pre-filled before commencing the experiment according to an uneven distribution, with class 1 comprising 66.6 percent and class 2 taking 33.3 percent of the total demands. As a result, the placement of different classes were modulated according to the AS-CBS calculation.

It was demonstrated that the number of completing a dual order cycle was low and robots often failed to finish a complete cycle, hence, no more than two cycles were completed by a robot. The number of collisions also appear to have little differences, and in both the volume and collision counts, there seem to be no significant evidence to believe that the means differ between the Random and AS-CBS setting.

The lack of discrimination most likely originate from the limited scale and sam-




(a) (b)

Figure 5.12: Boxplots comparing the Random and AS-CBS method with different metrics. (a) Collision per Minute. (b) Time to First Completion.

Table 5.6: Performance Comparison between Random and AS-CBS Methods with regards to time.

| **Metric** | **Random** | **AS-CBS** |
|---|---|---|
| Collision Frequency (per minute) | 0.30 ± 0.12 | 0.26 ± 0.13 |
| Completion Rate | 7/20 (35.0%) | 13/20 (65.0%) |
| Mean Time to First Completion (s) | 244.67 ± 188.74 | 284.88 ± 169.07 |
| Min Time to First Completion (s) | 49.94 | 40.83 |
| Max Time to First Completion (s) | 532.54 | 505.98 |

ple size of the experiment. In the simulation setting, the field was considerably larger with 29 by 32 grid, with up to 50 robots as opposed to the robotic experiment. Because of these results, the effectiveness of the AS-CBS policy remains inconclusive in real applications, and its performance when the demands of different items fluctuate, therefore further experiments in greater scale are warranted to better clarify the dynamics of the system and identify the underlying issues that led to these obtained results. Regardless, Figure 5.10 revealed that placement of higher demand class 1 was modulated closer to the corner P/D point, validating the AS-CBS implementation was in effect.

To further evaluate the underlying dynamics of robotic implementation and to discover deeper from the completion and collision counts, additional analysis using the time data was conducted. Mainly the collision frequency and time it takes to first completion was the focus in studying how the claimed performance gains translate into real time. From the Table 5.6, the Random method and the AS-CBS method show marginal differences in the mean (Random: 0.30 ± 0.12, AS-CBS: 0.26 ± 0.13 per minute), and also in the median as seen from box plots in Figure 5.12.

Regardless, the most notable advantage of AS-CBS method appeared when completing a P/D cycle, having 86-percent improvement in completion rate in comparison to the Random method. Out of 20 item placement and delivery cycle, AS-CBS method completed only failed to complete 7 times, while the Random method failed 13 times. It is noteworthy that the AS-CBS achieved this higher completion rate while also achieving a slight decrease in collision frequency.

The analysis using the time metric also revealed the reliable and consistent performance of AS-CBS, with both lower minimum completion time (40.83s vs 49.94s) and maximum completion time (505.98s vs 532.54s), with the smaller standard deviation in completion times (169.07s vs 188.74s) further supporting the consistency of the operation. Such consistent performance in contrast to the Random method is somewhat foreseen, because randomly stored items have the chance of having the needed item nearby, but could also potentially be farther away, especially if the item demands are non-uniform.

Furthermore, AS-CBS may result in a more optimized and shortened route as demonstrated in the simulation results, leading to less collisions and a more reliable and structured path to the goal, and the experimental results seem to support these claims. The outstanding aspect of the AS-CBS implementation is the demonstration of reliability and consistency while maintaining the collision frequency lower or similar to the Random method, under the real-world uncertainties and physical constraints that often degrade algorithmic performance.

A more optimized setting with less reliance to visual robot location and increased individual autonomy could further fine tune the performance of the AS-CBS method, thus this robotic implementation sets the foundation for continued improvements with further refinements.

## 5.3 Summary of Chapter 5

This chapter evaluated the real-world implementation of the proposed path finding and storage algorithms through robotic experiments, successfully demonstrating the agent-based path finding in a physical robot environment and validating the AS-CBS storage strategy implementation. The experiments achieved meaningful reductions in collisions while maintaining throughput, demonstrating the practical feasibility of the proposed algorithms in real-world conditions.

Performance results were particularly encouraging for both main components of the system. The Agent Density Method showed balanced improvements with a 26-percent reduction in collision frequency and completion times 25-percent faster than baseline and 35-percent faster than WHCA*. Similarly, AS-CBS demonstrated an impressive 86-percent improvement in completion rate, along with more consistent performance and lower variance, while successfully adapting to changing demands throughout the experiments.

The implementation process revealed several critical technical challenges. Visual recognition reliability proved crucial for system performance, while communication consistency played a vital role in coordination. Physical movement variations required compensation, and battery voltage consistency significantly affected robot performance. These challenges led to the development of specific solutions, including the use of line following for movement correction, ArUco markers for robust position tracking, and NiMH batteries for consistent power delivery. The coordination approach also required modification to account for real-world constraints.

Several limitations and considerations emerged during the experimental phase. The scaled-down environment affected measurement precision, while limited sample size impacted statistical significance. Physical constraints introduced new variables that required consideration, and real-world factors necessitated algorithm adaptations. These limitations provide valuable insights for future implementations and scaling considerations.

The experiments established clear pathways for future implementation, identifying the need for robust visual acquisition systems and reliable communication infrastructure. The demonstrated value of physical movement compensation and the established foundation for larger-scale implementations provide crucial guidance for future development. This research successfully validated the core mechanics of the proposed methods in a physical environment while identifying key challenges and solutions for practical implementation. The results provide valuable insights for future deployment in real-world robotic fulfillment centers, establishing a foundation for continued development and scaling of these approaches.

# Chapter 6

# Conclusion

This research investigated three interconnected aspects of automated warehouse operations: congestion detection using convolutional filters, adaptive storage strategies, and their practical implementation in robotic systems. The investigation began with the development of an agent density-based path finding algorithm that uses convolutional filters to detect and respond to congestion patterns.

This was complemented by the introduction of Adaptive Stochastic Class-Based Storage (AS-CBS), a novel approach to dynamically optimize storage locations based on real-time demand patterns. Finally, these theoretical developments were validated through physical robotic experiments that provided crucial insights into the challenges and opportunities of real-world implementation.

Together, these three components represent a comprehensive approach to improving warehouse automation, addressing both the theoretical foundations and practical considerations of robotic warehouse operations. This chapter summarizes the key findings and implications from each aspect of the research, discussing their significance for future warehouse automation development.

## 6.1 Congestion Detection Based on Convolutional Filter

In the context of evolving trends towards automated smart factories and paperless inventory management, this research endeavors to investigate the intricate dynamics of AGVs in warehouse environments with varying storage techniques. The primary objective is to assess the performance of AGVs under different storage configurations reflective of diverse business needs and to propose an effective congestion mitigating routing heuristic.

The findings of this study demonstrate that the agent density-based path reconstruction model consistently outperforms other methods in terms of order picking efficiency within a given timeframe. Notably, this approach also significantly reduces the incidence of collisions and congestion compared to scenarios without such measures in place.

Importantly, this characteristic holds true across different numbers of agents, item varieties, and storage methods, including random organization, traditional organization, and Class Based Storage (CBS). While perfect coordination among AGVs may theoretically yield high performance while eliminating congestion, the agent density-based approach offers resilience and simplicity in adapting to rapidly changing demands.

Moreover, it integrates the adaptability and robustness of Autonomous Mobile Robots (AMRs) designed to operate in dynamic human-robot interaction environments.

Future research endeavors may warrant exploring alternative implementations of routing heuristics and warehouse layouts to comprehensively assess the advantages and limitations of agent density-based path reconstruction. By conducting detailed investigations, tailored warehouse management strategies can be developed to optimize operational efficiency and effectively address evolving operational challenges.

## 6.2 Adaptive Stochastic Class Based Storage

The comprehensive simulation results underscore the consistent performance of the Adaptive Stochastic Class-Based Storage (AS-CBS) method across various turnover rate scenarios. While AS-CBS may not consistently yield the highest performance in terms of total items picked, collision counts, and congestion counts compared to other storage methods, it demonstrates remarkable reliability and predictability across a diverse range of situations within the scope of this research.

The integration of user-adjustable and controlled stochastic characteristics equips AS-CBS with adaptiveness and resilience to unforeseen variables that may otherwise impact performance, while maintaining stability in warehouse operations. However, for AS-CBS to be practically implemented, several additional considerations must be addressed.

These include accommodating separate turnover rates for incoming and outgoing order pickings, optimizing predetermined storage geometries such as within-aisle configurations, determining the optimal number of picking and delivery (p/d) points, and exploring potential enhancements to further refine the system's performance. By addressing these considerations, AS-CBS can be tailored to meet the specific requirements and operational constraints of individual warehouse environments, thereby enhancing its practicality and effectiveness in real-world applications.

Moreover, continued research and development efforts are essential to refine and optimize AS-CBS further, paving the way for its widespread adoption and integration into modern warehouse management systems. By leveraging the insights gained from this study and addressing the identified considerations, AS-CBS holds viability in improving warehouse storage strategies and optimizing operational efficiency in dynamic and evolving logistics environments.

## 6.3 Robotic Application of Congestion Detection and Adaptive Storage Strategy

The application of convolution filter based routing into real robots proved its feasibility in implementation and demonstrated its improvements. Despite the robot experiments showed gaps with the simulation results, it provided valuable insight into identifying issues it faced and ones that may potentially arise.

The algorithm needed to accumulate activity for its benefits to become distinct in contrast to having no collision and congestion avoiding measures, as seen from traces of performance discrepancies among different settings. In the simulation environment, the scale of the picking activity was longer at 40000 steps on a larger 29 by 32 grid with 20 robots, conversely, the robotic experiment was scaled with limited duration due to the battery capacity and smaller field.

The limitation of the robotic experiment most prominently appeared when testing the AS-CBS implementation. Although lacking, path finding algorithms showed some improvements and traces of performance differences, but the virtual order picking activity often collected less than sufficient number of cycles to draw a meaningful conclusion despite the elongated duration. Nevertheless, future experiment that better replicates the simulated environment still holds possibility of revealing the effect of AS-CBS in real world environment.

## 6.4 Future Works

In the domain of congestion detection and path planning, future research should explore alternative implementations of the convolutional filter approach. This includes investigating different kernel sizes and configurations, developing adaptive threshold mechanisms that respond to varying traffic densities, and exploring the integration of machine learning techniques to optimize filter parameters based on historical congestion patterns. Additionally, the investigation of alternative warehouse layouts specifically designed to complement the agent density-based path reconstruction could potentially yield further improvements in operational efficiency.

For the AS-CBS storage strategy, several key areas warrant further investigation. First, the development of a more sophisticated model that can handle asymmetric turnover rates between incoming and outgoing operations could enhance the system's practical applicability. Second, research into optimizing storage geometries, particularly focusing on within-aisle configurations and their interaction with the adaptive storage strategy, could improve space utilization. Third, investigating the optimal number and placement of P/D points in relation to storage zones could further enhance system performance. Lastly, exploring methods to reduce the warm-up period required for the queue-based demand estimation could improve the system's responsiveness to changing demand patterns.

Regarding robotic implementation, future work should focus on scaling up the experimental validation. This includes conducting longer-duration experiments with larger robot fleets and storage areas to better match simulation conditions. Development of more energy-efficient robots with extended battery life would enable such extended testing. Additionally, investigation into improved visual recognition systems and communication protocols could enhance the reliability of real-world implementations. Research into robust error recovery mechanisms and fault-tolerant operation would also be valuable for practical deployments.

Integration and scaling aspects also present important research opportunities. This includes investigating the scalability of the combined system (congestion detection, adaptive storage, and robotic implementation) in larger warehouse environments, developing methods to handle dynamic reconfigurations of warehouse layouts, and exploring the integration of human operators in hybrid human-robot warehouse environments. The development of standardized performance metrics and testing scenarios would also facilitate comparative evaluation of future improvements to these systems.

Finally, theoretical extensions of this work could explore the formal properties of the agent density-based congestion detection method, including proofs of convergence and optimality under various conditions. The relationship between storage location assignment and congestion formation could also be analyzed mathematically to provide insights for future system optimization.

# Bibliography


Allgor, R., Cezik, T., and Chen, D. (2023). Algorithm for robotic picking in amazon fulfillment centers enables humans and robots to work together effectively. *INFORMS Journal on Applied Analytics*, 53(4):266–282.

Amazon.com (2021). Our facilities and offices.

Arduino LLC (2024). Arduino Uno Rev3. Italy.

Azadeh, K., De Koster, R., and Roy, D. (2019). Robotized and automated warehouse systems: Review and recent developments. *Transportation Science*, 53(4):917–945.

Bassan, Y., Roll, Y., and Rosenblatt, M. J. (1980). Internal layout design of a warehouse. *A I I E Transactions*, 12(4):317–322.

Bozer, Y. A. and Cho, M. (2005). Throughput performance of automated storage/retrieval systems under stochastic demand. *IIE Transactions*, 37(4):367–378.

Caron, F., Marchet, G., and Perego, A. (2000). Optimal layout in low-level picker-to-part systems. *International Journal of Production Research*, 38(1):101–117.

Chatzisavvas, A., Chatzitoulousis, P., Ziouzios, D., and Dasygenis, M. (2022). A routing and task-allocation algorithm for robotic groups in warehouse environments. *Information*, 13(6):288.

Cohen, L., Koenig, S., Kumar, T. K. S., Wagner, G., Choset, H., Chan, D., and Sturtevant, N. (2018). Rapid randomized restarts for multi-agent path finding: Preliminary results. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '18, page 1909–1911, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

Cornuéjols, G., Fonlupt, J., and Naddef, D. (1985). The traveling salesman problem on a graph and some related integer polyhedra. *Mathematical Programming*, 33(1):1–27.

de Koster, R., Le-Duc, T., and Roodbergen, K. J. (2007). Design and control of warehouse order picking: A literature review. *European Journal of Operational Research*, 182(2):481–501.

de Koster, R., van der Poort, E., and Roodbergen, K. J. (1998). *When to apply optimal or heuristic routing of orderpickers*, page 375–401. Springer Berlin Heidelberg.

Eynan, A. and Rosenblatt, M. J. (1993). An interleaving policy in automated storage/retrieval systems. *International Journal of Production Research*, 31:1–18.

Eynan, A. and Rosenblatt, M. J. (1994). Establishing zones in single-command class-based rectangular as/rs. *IIE Transactions*, 26(1):38–46.

Farrington, B. and Lysons, K. (2012). *Purchasing and supply chain management*. Pearson Education, London, England, 8 edition.

Feng, B. and Ye, Q. (2021). Operations management of smart logistics: A literature review and future research. *Frontiers of Engineering Management*, 8(3):344–355.

Frazele, E. A. and Sharp, G. P. (1989). Correlated assignment strategy can improve any order-picking operation. *Ind. Eng.*, 21(4):33–37.

Furmans, K. and Dehdari, P. (2014). An empirical study about the effectiveness of leanan empirical study about the effectiveness of lean empowerment in warehousesempowerment in warehouses. In *13th IMHRC Proceedings (Cincinnati, Ohio. USA – 2014)*. Progress in Material Handling Research.

Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F., and Marín-Jiménez, M. (2014). Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280–2292.

Gebser, M., Obermeier, P., Otto, T., Schaub, T., Sabuncu, O., Nguyen, V., and Son, T. C. (2018). Experimenting with robotic intra-logistics domains. *Theory and Practice of Logic Programming*, 18(3–4):502–519.

Gibbs, W. W. (1994). Software's chronic crisis. *Scientific American*, 271(3):86–95.

Gibson, D. R. and Sharp, G. P. (1992). Order batching procedures. *European Journal of Operational Research*, 58(1):57–67.

Goldberg, A. V., Oldham, J. D., Plotkin, S., and Stein, C. (1998). *An Implementation of a Combinatorial Approximation Algorithm for Minimum-Cost Multicommodity Flow*, page 338–352. Springer Berlin Heidelberg.

Graves, S. C., Hausman, W. H., and Schwarz, L. B. (1977). Storage-retrieval interleaving in automatic warehousing systems. *Management Science*, 23(9):935–945.

Gue, K. R., Ivanović, G., and Meller, R. D. (2012). A unit-load warehouse with multiple pickup and deposit points and non-traditional aisles. *Transportation Research Part E: Logistics and Transportation Review*, 48(4):795–806.

Gue, K. R. and Meller, R. D. (2009). Aisle configurations for unit-load warehouses. *IIE Transactions*, 41(3):171–182.

Hall, R. W. (1993). Distance approximations for routing manual pickers in a warehouse. *IIE Transactions*, 25(4):76–87.

Han, M.-H., McGinnis, L. F., Shieh, J. S., and White, J. A. (1987). On sequencing retrievals in an automated storage/retrieval system. *Iie Transactions*, 19:56–66.

Hart, P., Nilsson, N., and Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107.

Hausman, W. H., Schwarz, L. B., and Graves, S. C. (1976). Optimal storage assignment in automatic warehousing systems. *Management Science*, 22(6):629–638.

Helsgaun, K. (2000). An effective implementation of the lin–kernighan traveling salesman heuristic. *European Journal of Operational Research*, 126(1):106–130.

Hsueh, C.-F. (2010). A simulation study of a bi-directional load-exchangeable automated guided vehicle system. *Computers & Industrial Engineering*, 58(4):594–601.

Jarvis, J. M. and McDowell, E. D. (1991). Optimal product layout in an order picking warehouse. *IIE Transactions*, 23(1):93–102.

Johnson, D. S. and McGeoch, L. A. (2008). The traveling salesman problem: A case study in local optimization.

Johnson, M. E. and Brandeau, M. L. (1996). Stochastic modeling for automated material handling system design and control. *Transportation Science*, 30(4):330–350.

Karp, R. M. (1972). *Reducibility among Combinatorial Problems*, page 85–103. Springer US.

Kizilateş, G. and Nuriyeva, F. (2013). On the nearest neighbor algorithms for the traveling salesman problem. In Nagamalai, D., Kumar, A., and Annamalai, A., editors, *Advances in Computational Science, Engineering and Information Technology*, pages 111–118, Heidelberg. Springer International Publishing.

Larson, T. N., March, H., and Kusiak, A. (1997). A heuristic approach to warehouse layout with class-based storage. *IIE Transactions*, 29(4):337–348.

Le-Duc *, T. and De Koster, R. (2005). Travel distance estimation and storage zone optimization in a 2-block class-based storage strategy warehouse. *International Journal of Production Research*, 43(17):3561–3581.

Le-Duc, T. (2005). *Design and control of efficient order picking processes*.

Lee, H. and Jeong, J. (2021). Mobile robot path optimization technique based on reinforcement learning algorithm in warehouse environment. *Applied Sciences*, 11(3):1209.

Lin, S. and Kernighan, B. W. (1973). An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, 21(2):498–516.

Liu, W., Hou, J., Yan, X., and Tang, O. (2021). Smart logistics transformation collaboration between manufacturers and logistics service providers: A supply chain contracting perspective. *Journal of Management Science and Engineering*, 6(1):25–52.

Ltd, C. C. (2008). Case study - denver international airport baggage handling system - an illustration of ineffectual decision making.

Ma, H. (2021). A competitive analysis of online multi-agent path finding.

Ma, H., Hönig, W., Kumar, T. K. S., Ayanian, N., and Koenig, S. (2018). Lifelong path planning with kinematic constraints for multi-agent pickup and delivery.

Malmborg, C. J. and Al-Tassan, K. (2000). An integrated performance model for orderpicking systems with randomized storage. *Applied Mathematical Modelling*, 24(2):95–111.

Meller, R. D. and Gue, K. R. (2009). The application of new aisle designs for unit-load warehouses.

Merschformann, M., Lamballais, T., de Koster, M., and Suhl, L. (2019). Decision rules for robotic mobile fulfillment systems. *Operations Research Perspectives*, 6:100128.

Öztürkoğlu, O., Gue, K. R., and Meller, R. D. (2012). Optimal unit-load warehouse designs for single-command operations. *IIE Transactions*, 44(6):459–475.

Papadimitriou, C. H. (1992). The complexity of the lin–kernighan heuristic for the traveling salesman problem. *SIAM Journal on Computing*, 21(3):450–465.

Petersen, C. G. (1997). An evaluation of order picking routeing policies. *International Journal of Operations & Production Management*, 17(11):1098–1111.

Petersen, C. G. and Aase, G. (2004). A comparison of picking, storage, and routing policies in manual order picking. *International Journal of Production Economics*, 92(1):11–19.

Petersen, C. G. and Aase, G. R. (2017). Improving order picking efficiency with the use of cross aisles and storage policies. *Open Journal of Business and Management*, 05(01):95–104.

Petersen, C. G., Aase, G. R., and Heiser, D. R. (2004). Improving order-picking performance through the implementation of class-based storage. *International Journal of Physical Distribution & Logistics Management*, 34(7):534–544.

Petersen, C. G. and Schmenner, R. W. (1999). An evaluation of routing and volume-based storage policies in an order picking operation. *Decision Sciences*, 30(2):481–501.

Pohl, L. M., Meller, R. D., and Gue, K. R. (2009). An analysis of dual-command operations in common warehouse designs. *Transportation Research Part E: Logistics and Transportation Review*, 45(3):367–379.

Pohl, L. M., Meller, R. D., and Gue, K. R. (2010). A new aisle design for dual-command travel. *IISE Annual Conference.Proceedings*, pages 1–6.

Pololu Corporation (2024). Zumo shield for Arduino, v1.2. Las Vegas, NV, USA.

Ratliff, H. D. and Rosenthal, A. S. (1983). Order-picking in a rectangular warehouse: A solvable case of the traveling salesman problem. *Operations Research*, 31(3):507–521.

Riedel, R. (2011). Facilities planning – 4th edition by j.a. tompkins, j.a. white, y.a. bozer and j.m.a. tanchoco. *International Journal of Production Research*, 49(24):7519–7520.

Robinson, J. B. (1949). *On the Hamiltonian game (a traveling-salesman problem).* RAND Corporation, Santa Monica, CA.

Roodbergen, K.-J. (2001). *Layout and Routing Methods for Warehouses*. PhD thesis.

Roodbergen, K. J. and de Koster, R. (2001). Routing order pickers in a warehouse with a middle aisle. *European Journal of Operational Research*, 133(1):32–43.

Roodbergen, K. J. and Vis, I. F. (2009). A survey of literature on automated storage and retrieval systems. *European Journal of Operational Research*, 194(2):343–362.

Rosenblatt, M. J. and Eynan, A. (1989). Note—deriving the optimal boundaries for class-based automatic storage/retrieval systems. *Management Science*, 35(12):1519–1524.

Roser, C. (2021). Amazon kiva storage strategies.

Rücker, A., Rief, J., and Fottner, J. (2020). Development of a method for the energy efficiency determination of stacker cranes in automated high-bay warehouses. *FME Transactions*, 48(4):753–760.

Ryu, H., Kang, B., Park, S., and Kim, D. (2024a). Demand-driven adaptive storage strategy for improved warehouse efficiency. *International Journal of Physical Distribution & Logistics Management*. In preparation for submission.

Ryu, H., Kang, B., Park, S., and Kim, D. (2024b). Spatial convolution-based congestion detection and avoidance for multi-robot systems. *Swarm and Evolutionary Computation*. In preparation for submission.

Sharp, G. P., Il-Choe, K., and Yoon, C. S. (1991). *Small Parts Order Picking: Analysis Framework and Selected Results*, page 317–341. Springer Berlin Heidelberg.

Silver, D. (2021). Cooperative pathfinding. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 1(1):117–122.

Stern, R., Sturtevant, N., Felner, A., Koenig, S., Ma, H., Walker, T., Li, J., Atzmon, D., Cohen, L., Kumar, T. K. S., Boyarski, E., and Bartak, R. (2019). Multi-agent pathfinding: Definitions, variants, and benchmarks.

Stout, B. (1998). Smart moves: Intelligent pathfinding.

Theys, C., Bräysy, O., Dullaert, W., and Raa, B. (2010). Using a tsp heuristic for routing order pickers in warehouses. *European Journal of Operational Research*, 200(3):755–763.

Tompkins, J. A., White, J. A., Bozer, Y. A., and Tanchoco, J. (2010). *Facilities Planning*. Wiley-Blackwell, Hoboken, NJ, 4 edition.

van den Berg, J. P. and Gademann, A. (2000). Simulation study of an automated storage/retrieval system. *International Journal of Production Research*, 38(6):1339–1356.

Yu, J. and LaValle, S. (2013). Structure and intractability of optimal multi-robot path planning on graphs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 27(1):1443–1449.

Zhou, L., Zhao, J., Liu, H., Wang, F., Yang, J., and Wang, S. (2022). Stochastic models of routing strategies under the class-based storage policy in fishbone layout warehouses. *Scientific Reports*, 12(1).

Zou, B., Gong, Y. y., Xu, X., and Yuan, Z. (2017). Assignment rules in robotic mobile fulfilment systems for online retailers. *Int. J. Prod. Res.*, 55(20):6175–6192.