A Multi-Mobile-Robot Control Framework for Parcel Sorting in Sorting Centres

By

Ch'ng Chee Henn

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONS)

Faculty of Information and Communication Technology
(Kampar Campus)

MAY 2019

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

Title:

A Multi-Mobile-Robot Control Framework for Parcel Sorting in Sorting Centres

Academic Session: May 2019

I , CH'NG CHEE HENN

declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

_________________________ _________________________

(Author's signature) (Supervisor's signature)

Address:

____

_________________________ _________________________

_________________________ Supervisor's name

Date: ____________________ **Date**: ____________________

DECLARATION OF ORIGINALITY

I declare that this report entitled "**A Multi-Mobile-Robot Control Framework for Parcel Sorting in Sorting Centres**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : ___________________________

Name : ___________________________

Date : ___________________________

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisors, Dr Liew Soung Yue who has given me the opportunity to engage in this multi-mobile robot system. He also gave me guidance so that I able to complete this project smoothly.

Finally, I must say thanks to my family for their love and support.

ABSTRACT

The era of automation is coming. Thus, this project will investigate the multi-mobile robots for parcel sorting in the sorting centre. The reason for this is because logistics is one of the areas that will benefit from automation and the multi-robot system is one of the solutions to automation. The logistics processes are complex, and each process might be conducted in a different location. However, this project only focuses on the sorting centre. Besides, the multi-mobile robot system can be distinguished based on the approach used to manage the robots. For example, decentralized approaches, centralized approaches or a combination between them. The distributed approach is normally more robust to the partially known environment, but the sorting centre designed for multi-robot does not belong to this case. Thus, it is better to focus on improving the performance of the multi-mobile robot system by utilizing all the information. This can be achieved by adopting a centralized system, but the centralized system is normally complex, and it is difficult to manage a huge number of robots. Therefore, this project proposed a hybrid multi-mobile robot system that maintains the simplicity of the system and provides the ability to utilize global information. Apart from that, a discrete event simulator with a time-advance mechanism was created to study the performance of the proposed multi-robot system. Next, this project shows that it is possible to control the robot's path planning behaviour by modifying the moving cost between two grids. This is a novel way with high extendibility to control multi-robot behaviour. This project shows a few examples that utilizing this method to boost up the total throughput. The results show that the proposed framework able to achieve 31339 throughputs per hour in term of the sorted parcel.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

CBS	Conflict Based Search
MAPF	Multi-agent pathfinding

Chapter 1 Introduction

1.1 Project Background

1.1.1 Overview of Logistics Automation

With the maturity of the Internet, online shopping and e-commerce have gradually become popular. According to the statistical data provided by Statista (2018), the e-commerce share is increasing day by day and expected to achieve 17.5% in the year 2021. Moreover, global electronic retail sales reached 2.3 trillion in 2017, this thanks to online shopping has become a popular worldwide activity. This brought a significant change to the market. For example, the competition has become crucial, a life cycle of a product is shortened. All these situations state the importance of an efficient supply chain to fulfil the requirements and expectations of customers (Wessman & Bärring, 2014). To meet the customer needs the logistics operations such as receiving, storage, picking, and shipping of goods need to be improved (Constantiono, Dotolim, Fanti, & Mangini, 2012). However, the warehouse resources such as labour, equipment and space are always limited (Wessman & Bärring, 2014). Thus, automated robotics solutions have become one of the viable investments for optimizing logistics processes. (Husain, Kennedy, & Krebs, 2016).

1.1.2 The Challenge of Logistics Industries Today

Labour availability is the biggest challenge in logistics industries (Bonkenburg, 2016). Thus, it's not an easy task for logistic companies to find many high-quality employees. Kennedy, & Krebs (2016) state that the single largest expense for logistics companies is labour cost. As an evidence, in September of 2016, total employment in the warehouse and logistics industry have increased 5.94% compared to the same period on 2015. On the other hand, the average hourly earnings of all employees are also increasing year by year. The increasing of labour cost is driving the logistic companies to shift to the high level of automation. However, the research of Husain, Kennedy, & Krebs (2016) claimed that only under 10% of companies deploy robotics technologies in their facilities and most of the organizations are conservative in investing in this technology. This shows that the logistics automation is still infancy and has great potential in the future.

1.1.3 Distribution Centre and Warehouse

Warehouse and distribution centre are similar for a normal person. However, they have different responsibilities and internal operation. Normally, warehouses are only used for storage. However, distribution centres are building designed to store the products for retailers and wholesalers and redistribute these products to customers or another location. According to Cannon Hill Logistics (2017), distribution centres are an integral part of the order fulfilment process, which are receives, processes, and delivers orders to customers. The normal way to utilize a distribution centres is that the retailers ship the products to the distribution centre, after, then the products are shipped from the distribution centre to the customer. Since products move in and out frequently, the main consideration in building a distribution centre is to make the pickup and drop off process efficient. In short, although warehouses and distribution centres look similar, warehouses are designed for long-term storage while distribution centres are designed to provide rapid intake and shipment of items. However, no matter how well the distribution centre is designed, if the rest of the logistics process is not designed properly, the overall delivery speed will still slow down.

1.1.4 Automation of Sorting Centres

Sorting centres is an important part of logistics processes. In sorting centres, items that need to be delivered will be sorted according to the specific location. The correctness and speed of sorting process can impact the overall speed of logistics processes significantly. However, a few years ago, parcels sorting could only be done by humans. For some large companies, they may use cross-belt to facilitate this process. However, cross-belts are expensive heavy machinery that small and medium-sized business can't afford. In addition, if any part of the cross-belt fails, the sorting centre will stop working. As a result, some companies have begun to introduce new powerful and cheaper solutions for parcel sorting. In most of the recent solutions, multi-mobile robot system plays an important role. This is because multi- robots are cheaper compared to heavy machinery and have good scalability and performance. Besides, it is more fault tolerance, that means when one robot fails to work, it will not affect others. One of the instances of multi-mobile robot system is the parcel-sorting robotics solution created by LiBiao Robotics (LiBiao, 2019). LiBiao Robotics claimed that their robotics system can save up to 70% human resources and the error rate of their system in sorting the parcel is much lower than humans, which are 0.5% and 0.01% respectively. On the other hand, robots will not be tired, they can work 24 hours per day, therefore this solution able to increase the throughput of a sorting centre significantly.

1.1.5 Multi-Mobile Robot System

Apart from the examples above, there are many solutions that are implemented using multi-mobile robots. For example, the Kiva system is a goods-to-pick solution that is widely used in Amazon warehouses (Guizzo, 2008). As the hardware costs are declining year by year, and technology is becoming more mature, multi-mobile robot solutions are becoming popular. Thus, it is important to design a suitable multi-mobile robot coordination algorithm so that these robots can perform their task efficiency.

1.1.6 Challenges in Multi-Mobile-Robot Coordination

When multiple robots work in the same workplace, there are many difficulties in coordinating the robot. These difficulties include task assignment, multi-mobile robot path planning, obstacle avoidance etc. Therefore, algorithm design for multi-mobile robots has become an important and interesting issue in the field of computer science. For example, how to coordinate multiple robots so that they can complete their tasks in the shortest total time. Besides, the thing that make the problem more interesting is most of the multi-mobile robots problems such as multi-mobile robots path planning and multi-mobile robots task allocation had been proved be NP-Hard (Pinkam, Bonnet, & Chong, 2016). This can lead to the creation of various approximate algorithms and multi-mobile robot control systems in order to achieve better performance.

1.2 Problem Statement and Motivation

The pace of automation is unavoidable, and this is especially true in the logistics world. However, most of the studies about multi-mobile robot automation and control framework focus in a partially observable environment. Although multi-mobile robot coordination system developed for the partially observable environment is more robust and can be implemented in the sorting centre, this sacrifices the algorithm performance because the known information of the environment is too limited.

In fact, the partially observable environment assumption is not that suitable for the environment that purposely designed to implement a multi-mobile robot system. Thus, by eliminating this assumption, more powerful algorithms that utilized all the environment information can be developed and increase the performance of a multi-mobile robot system. However, the research in this area is limited although there still have much room for development. Thus, this project is conducted to have more understanding about the multi-mobile robot control framework and multi-mobile robot control system in a known environment and the known environment in this project is sorting centres.

1.3 Project Scope

This project is aimed to develop multi-mobile robot control frameworks with different algorithms in order to figure out the best framework to speed up the throughput of automation sorting centre. The control framework should solve the typical problems in multi-mobile robot coordination such as multi-mobile robot path planning. On the other hand, this project will develop a simple simulator that simulates parcel sorting centres in order to know the performance of different multi-mobile robot control frameworks.

1.4 Project Objectives

The aim of this project is to develop an efficient control framework for parcel sorting in sorting centre.

The objectives of the proposed project are:

- To develop an efficient multi-mobile robots control framework for parcel sorting centre.
- To develop a simulator to simulate the multi-mobile robots sorting centre in order to verify the performance of the developed multi-mobile robots control framework.
- To understand how the number of robots affects the efficiency of the sorting centre that implementing the proposed framework.
- To understand how the configuration of the environment affects the efficiency of the proposed framework in a multi-mobile robot environment.

1.5 Impact, Significance and Contribution

The pace of automation is unavoidable, one of the automation means is to utilize multi-mobile robot to complete repetitive tasks, such as parcel sorting in sorting centre. However, the related study in this area is limited.

Thus, this project builds a control framework for multi-mobile robots for parcel sorting in sorting centres and provides an understanding on how the multi-mobile robots surpass the performance of traditional sorting centre. Besides, this project investigates how the different algorithms implemented in the multi-mobile robot control framework can affect the system performance. On the other hand, this project also investigates how the configuration on the environment affect multi-mobile robot performance.

We believe this project will bring a deeper understanding of how to utilize multi-mobile robots to increase productivity in the future world. In addition, the results of the project can provide some insight into how to design efficient multi-mobile robot control framework for other environments. Therefore, this project can also be viewed as a preparation for the automation era.

1.6 Report Organization

This report consists of a total of 6 chapters. The first chapter introduces the project background, the problem statement as well as the project scope and objective. Chapter 2 study the related literature that might be useful in this project. Next, Chapter 3 describe the project flow as well as the overall system design. Besides, Chapter 4 discuss some consideration when designing the multi-robot control framework as well as other configurations. Apart from that, Chapter 5 shows the result of the experiments that investigate the proposed framework and configuration. Finally, Chapter 6 conclude all the previous chapters and describe future work.

Chapter 2 Literature Review

2.1 Decentralized and Centralized Multi-Mobile Robot System

According to Zhang, Guo, Chen, and Yuan (2018), most of the multi-mobile robot control systems can be classified into centralized and decentralized approach. Normally decentralized approaches better in handing multi-mobile robot but it might cause collisions and deadlocks. Besides, it is often more desirable to heterogeneous multi-mobile robot in decentralized manner because this can simplify the design of multi-mobile robot system and allow the task specification and constraints to remain private to each robot. On the other hand, decentralized approach will be useful if robots in a multi-mobile robot system is controlled by different entities that interested in minimizing the disclosure of private information (Cap, 2016). However, conflicts can easily occur in distributed multi-mobile robot systems. If occurred conflicts cannot be solved correctly, the overall performance of the system will reduce or even deadlock might occur (Wang & Chen, 2002).

In a centralized approach, the system gathers global information received from all robots and always keep track of the positions of the robots in the environment. Normally, this system is in a stationary host or a robot that is assigned as a master (Stanek, 2012). Besides, Xu (2010) claimed that centralized systems are typically implemented in well structural environments and for a small number of robots. He also stated the centralized systems is not robust to a dynamic environment, failures in communications and other uncertainties. However, the central agent in centralized approach has a global view of the environment, thus it can produce a globally optimal plans (Yan, Jouandeau, & Cherif, 2013).

2.2 Multi-mobile robot System Design

Wurman, D'Andrea, & Mountz (2008) found 4 golden rules in deciding how to partition the multi-mobile robot system into agents:

1. Physical correspondence.
2. Information encapsulation.
3. Single-agent ownership.
4. Separation by job.

The first point means that for each physically distinct object there to be one agent. For the second point which is information encapsulation state that each agent should know the minimum information to perform its tasks. This reduces bandwidth and system database workload. Next, the single-agent ownership means that only one agent owns all the data elements, that is only one agent can permanently change or write the database, although multiple agents may need the information. The last rule is separation by jobs which is when there is a resource allocation that needs to be done, it can be encapsulated as a separate agent.

On the other hands, Wang and Chen (2002) proposed a queue coordination strategy. The basic idea of the strategy is based on the behaviour of ant colony in transporting foods. In this strategy, a robot will observe the status of robots near to it. Based on its observation, a pre-defined algorithm will be used to determine the next action. The robot might follow the robots near to it and control the speed and the distance between itself and other robots (Wang & Chen, 2002). This strategy reduces the conflict in using the limited space of the warehouse and improve the overall performance.

Apart from that, Zhang, Guo, Chen, and Yuan (2018) separate their multi-mobile robot system into 2 layers. In their paper, the upper layer planning and the lower level logical control are separated. There have three modules in upper layer: a scheduling module, a planning module, and a supervisory module. Besides, each mobile robot in the lower system layer includes motion, speed and charge module.

2.3 Motion Coordination for Multi-Mobile Robot System

The multi-mobile robot motions coordination can be approached either reactive paradigm or deliberative paradigm. In the reactive paradigm, each robot moves in the shortest path to its destination and the robots resolve collision locally if it happens. In this paradigm, the robots attempt to avoid the collision by adjusting its heading and velocity by observes positions and velocities of other robots periodically. This method is widely used in practice because of the excellent computational efficiency and it decentralized nature (Cap, 2016).

In the deliberative paradigm, the system first plans a collision-free trajectory for each robot. After that, the robots start following their planned trajectories. This method is guaranteed all robots can reach their destination without any collisions with other robots. However, even the simplest variant of this problem is intractable (Cap, 2016). Another alternative formulation of this problem is pebble motion problem. In pebble motion problem, each robot is assumed to occupy one vertex on a given graph. The objective of this problem is to find a trajectory for each robot from its start vertex to its destination vertex so that two robots never occupy one vertex at any time step. However, it still a NP-hard problem in finding the optimal solution in this formulation (Yu, 2016). Although non-optimal solutions can be solved in polynomial time but in term of travel time, it is significantly suboptimal (Cap, 2016).

2.4 A* Algorithm

Path planning is a typical search problem. There have many well-developed informed search algorithms that use for path planning such as Uniform Cost Search (UCS) Algorithm, Greedy Algorithm and A* Algorithm. The UCS algorithm always finds the optimal solution but it has higher complexity compare to others. On the other hands, the greedy algorithm can find a path with short computation time, but it results not always optimal (Tsang, Che, Zhang, & Song, 2017). Fortunately, by introducing heuristics function, A* algorithm able to combine the benefits of UCS algorithm and A* algorithm which is finding a path between two points with low computation time and high optimality. According to E.Hart & J.Nilsson (1968), A* can always find a optimal path if the heuristic function always less than the actual minimum cost from current node to destination node. Yet, A* can perform its task faster if the heuristic function have a bigger value. Thus, it’s important to find a suitable heuristic function.

2.5 More Path Planning Algorithms

Dijkstra’s algorithm is a well-known algorithm in solving shortest path problems. However, it can only find one shortest route because it only stores one intermediate node. To address this problem, Zhang, Guo, Chen & Yuan (2018) proposed an improved Dijkstra’s algorithm which it can found more than one shortest part and it’s useful for collision-free solutions. However, the drawback of this algorithm is the high time complexity, it's difficult to search for shortest paths for large area warehouses in an acceptable time. Also, A* algorithm can be a better alternative for a grid environment.

Multi-agent pathfinding (MAPF) problem can be view as a generalization of the single-agent pathfinding problem where its task is finding paths for all agents from their start vertex to goal vertex. Normally, an additional goal will be introduced to minimizing the total cost of every agent to reach its goal is required in order to increase the performance of multi-agent systems. (Sharon, Stern, Felner, & R. Sturtevant, 2014).

According to Sharon, Stern, Felner, and R. Sturtevant (2014), algorithms for solving MAPF can be categorized into two categories: optimal and sub-optimal solvers, and they propose an algorithm called Conflict Based Search (CBS) for searching optimal solutions. This algorithm has a better performance compared to traditional approach such as A* for MAPF. However, the state space of MAPF problem grows exponentially with the number of agents, the CBS require an unacceptable time to find a solution if there have many robots. Therefore, Barer, Sharon, Stern, and Felner (2014) proposed a few ways to relax the optimality of CBS. According to the paper, the new algorithms able to sacrifice a minor loss in solution quality but have are a massive reduction in running time. Yet, the proposed algorithm is still not suitable for sorting centre environment because its computation time is still high, and it is impossible to adjust all the trajectories time by time.

2.6 Collision between Multi-Mobile Robot

According to Zhang, Guo, Chen, & Yuan (2018), potential collisions in the automated warehouse system can be classified into 4 categories, however, only 3 categories will occur in the sorting centre. The 3 categories are listed in below:

- Head-on collision
- Cross collision
- Node occupancy collision

Head-on collision means that two mobile-robot travel on the same path at the time but in opposite direction. Next, a cross collision happens when two mobile robots compete for a given node at the intersections. Apart from that, the node occupancy happens when a mobile robot stops at the path of another robot.

Zhang, Guo, Chen & Yuan (2018) proposed some solutions to handle collision problem such as selecting the candidate route, modifying the route and waiting a short period of time before starting. For the first solution only change the schedule of the current route, and the total travel time is the same as previous because one or two shortest routes might be found. For the second solution, the path of later mobile-robot will be re-planned if the route of previous mobile-robots collision with the later mobile-robot. The third solution, which is waiting a short period of time before starting is an improved version of the traditional solution which is waiting when a collision occurs. The third solution is proposed because the traditional solution will cause waste of energy during the period of stop and restart of mobile robots.

2.7 Simulation Modelling

2.7.1 Type of Models

Law (2007) claimed that in order to study a system scientifically, it is important to build a model to understand how a system. If the relationships of a model are simple, mathematical methods might be used but it's difficult to evaluate a realistic model for complex system analytically. Thus, simulation can be used to estimate the characteristics of a model.

Law (2007) classify simulation models along three different dimensions:

1. *Static vs. Dynamic Simulation*
2. *Deterministic vs. Stochastic Simulation*
3. *Continuous vs. Discrete Simulation*

A static simulation model represents the system at a given time, or time is no playing any role in the model, while a dynamic simulation model indicates that the system evolves over time. Besides, deterministic simulation model represents a model does not contain any probabilistic components. On the other hands, continuous simulation simulates the state variables of a system change continuously with respect to time while discrete simulation simulates the state of a system change instantaneously if an event occurs.

2.7.2 Time-Advance Mechanisms

Multi-mobile robot sorting centre simulation can be modelling as discrete-event simulation and it is important to keep track of the simulated time in discrete-event simulation models. There have two principal methods to advance the simulation clock which is next-event time advance and fixed-increment time advance. The next-event time advance approach is widely used by all major simulation software. This approach always advances the time to the next future events then update the event list based on the system state change. Thus, all the inactive periods are skipped over (Law, 2007).

Chapter 3 Project / System Design

3.1 Project Flow

It is costly and impractical to study the multi-mobile robot control framework based on real environment and the multi-mobile robot system is also too complicated for mathematical formulation. Thus, simulation is the best way to capture and analyse the important characteristics of a multi-mobile robot system. Therefore, the first stage of this project is to build a simulator to simulate the multi-mobile robot system and investigate potential issues.

Although there have many multi-mobile robot simulator platforms existing in the market, most of them focus on the mechanical part of robots and it is too time-consuming to create an experiment environment with different layouts and configuration. Thus, a discrete-event simulator that simulates multi-mobile robot system was built in this project. The simulator allows the rapid modification of system layout and configuration at the same time capture the most important characteristics of the multi-robot system. For instance, the collision between robots, the competition for resources and the most important thing is the throughput of the system, that is the number of parcels sorted per hour.

After that, the simulator allows us to implement various time of algorithms to overcome some problems in multi-mobile robot system such as path planning algorithm, charge scheduler algorithm and so on. Though the observation of the simulation result, the bottleneck and weakness of the system, algorithm or the layout and configuration of sorting centre can be spotted and thus some improvement can be done to further boost up the system performance. Also, this iterative process might provide some important insight into how to design an efficient multi-mobile robot control framework or how the environment layout and configuration affect the performance of multi-robot system.

3.2 Overview of Multi-Mobile robot Sorting Centre Processes

1. Parcels from Inbound Unit Load Device (ULD) are placed to pick-up points.
2. Workers pick up a parcel from pickup points then scan it in order to know where to place the parcel.
3. Assign the parcel to a robot for sorting.
4. Robot drops the parcel to the correct location or says outbound ULD.

The information above is based on the report from Mauro (2017) and the observation of multi-mobile robot parcel sorting solution from LiBiao Robotics.

Figure 3.1 Sorting Centre that Implements Multi-mobile robot System (Online Source, retrieved from https://kknews.cc/zh-sg/tech/pq3qn38.html)

3.3 Language & Library

In this project, Python is the language used to develop and test the multi-mobile robot control framework. This is because Python is widely supported by the machine learning community and there have many well-developed machine learning libraries for Python such as TensorFlow and Scikit-Learn. These libraries allow the AI method to implement in multi-mobile robot control framework easily. Also, the Python may have the most concise syntax among all programming language. This shortens the development time for simulator and algorithms so that this project can more focus on multi-mobile robot control framework. One the other hand, there have two important python libraries are used in this project which is SimPy and Matplotlib. SimPy is a library that provides comprehensive tools for discrete-event simulation whereas the Matplotlib is a popular 2D visualization tool that used to plot the environment status so that we can have a deeper understanding about the status of the simulation environment.

3.4 Overview of Environment Modelling

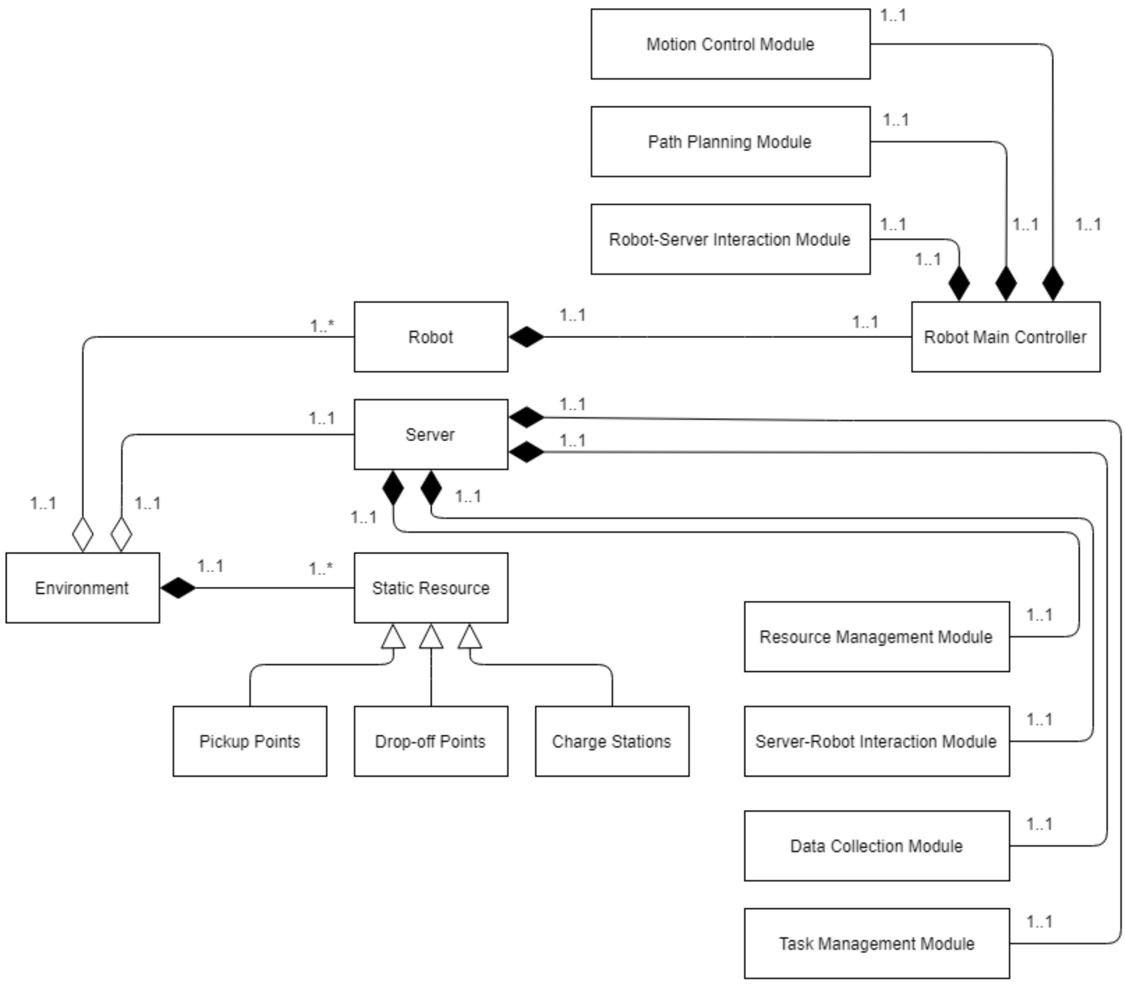


Figure 3.2 Framework & Environment Modelling

It is important to design and test with new algorithms and new frameworks, thus it's important to have a general structure that allows us to test different prototype rapidly. Figure 3.2 provides a high-level view of the simulator environment by using a simple class diagram. The usage of some important classes is shown below:

Environment: This is a container to build our simulator. It provides some basic ability to conduct a discrete-event simulation. Also, it's used to maintain some physical constraints, for example, each position can only occupy by at most one robot. Please note that the environment treats a grid as the basic unit of space and each grid is viewed as a resource. Theoretically, we can tread each grid as a vertex in a graph, and the graph is built by an adjacency matrix.

Static Resource: It is an abstract class that used to build a different type of resources, for example, pickup point, drop-off point and charge stations. Concrete classes of this class have their own unique properties and behaviours for server or robots to trigger.

Server: In our design, there must have one and only one server for the environment. The server is used to coordinate different objects in the environment.

Difference Modules in Server: Figure 3.2 shows the server has many modules. This is kind of modularization and it provides high flexibility to change the framework. Since this section is only aimed to provide a high-level view of environment modelling, the responsibility of these modules will not be discussed in this section. In fact, the modules can be different if we have a different framework.

Robot: The instance of the Robot class is the main agent in our environment, their interact to the environment and the server to perform its tasks. Like the server, a robot has many modules, the modules can also be different if we have a different framework.

3.5 Simulation Approach

3.5.1 Overview

In this project, we use discrete-event simulation to simulate the multi-robot environment. This type of simulation is commonly used and thus there have many libraries created to provide rich functionality for discrete-event simulation. One of them is SimPy in Python, thus the library is used to facilitate the simulator development process.

In this simulation approach, for all the object that actively interact with the environment, we need to build a generator or called coroutine to continuously interact with the environment. These generators keep generating an event, wait for the event to trigger, and generate new event based on the results. For example, after a robot generates an event that represents it is moving to next grid, the robot will wait until the event to trigger, and then it performs further action based on the result of the previous event and then generate a new event. On the other hand, the simulator always triggered the event nearest to current simulation time. When the event is triggered, the simulation time also jumps to the time of the event to occur.

3.5.2 Events

This section lists out the important events in our simulation and discusses how to design generators to trigger these events.

1. *A robot moves from one grid to another.*

2. *A robot releases a grid that is reserved.*

3. *A robot drops its parcel at a drop-off point.*

4. *A robot picks up a parcel at a pick-up point.*

5. *A robot request grid resource from the server.*

6. *A robot charges its battery at a charge station.*

7. *A robot's charge process is interrupted.*

Some of the processes of a robot are parallel. For instance, while the robot is moving, it can also receive the notice from the server such as the robot success or fail to request a future grid in its path. In this situation, the robot requires adjusting its velocity or make a new plan based on the obtained information. Such situations are difficult to model by using a single thread, but it is also difficult to use multiple threads while running the discrete-event simulation. Thus, both robot-server interaction module and motion control module is implemented with a generator/coroutine that responsible to generate the next event corresponding to the environmental situation.

The generator in robot-server interaction module is deal with the communication to the server and responsible in decision making based on the information obtained from the server and thus it also triggers the path planning module. Next, the motion module only manages the movement of the robot or any physical part of the robot, therefore, it always receives the command from robot-server interaction module and then control the physical robot corresponding to these commands.

3.6 Robot Motion Modelling

This project is aimed to study the multi-mobile robot control framework. Therefore, the robots are low fidelity because it is not necessary to consider the physical or mechanical part of robots to study the multi-mobile control framework. But, it's still necessary to consider some physical constraint such as the velocity of robots in a different situation because it might affect the system performance significantly. For example, robots need to slow down if it needs to make a turn, it cannot exceed a certain velocity due to the hardware limitation. For simplicity, we assume all the robot motion is linear motion. The list below shows the information that required to capture the effect of robots' motion:

- These exists a maximum acceleration for each robot, a_{max}.
- These exists a maximum deceleration for each robot, d_{max}.
- These exists a maximum velocity for each robot, v_{max}.
- The velocity of a robot should be zero before it makes a turn.
- If a robot is not allowed to move to next grids, it should stop at the last grid it can reach.

Due to the discrete-event simulation is used, we skip the transition state when a robot is travelling from one grid to another grid but focus only on the event when the robot reaches the next grid. Thus, robots are considered jumping from one grid to another after certain amount of time t, and the velocity model of the robot is then used to calculate what the exact value of t. Besides, during the interval time t, both current grid and next grid is blocked. This situation is because the moving robot occupied part of the current grid and next grid. Since one grid can only occupy one robot, then current grid should be blocked until the robot completely move to next grid.

The velocity model is described as follows. We assume robots always try to move as fast as possible, and it always used maximum acceleration to increase its velocity and maximum deceleration to decrease its velocity. The objective for robots is to minimize the time needed for a robot to move from start vertex g_0 to end vertex g_n. Apart from that, the robot needs to stop at g_n, that means it should slow down at certain position in its moving path, let say g_i. Also, if possible, the robot needs to increase its speed in order to reduce the time needed for it to reach g_n, but note that the maximum velocity is limited by v_{max}.

Now, we can reformulate this situation in another way. Let t_0 denotes the time when the robot reaches g_0, t_1 is the time when the robot reaches maximum velocity but if the robot never reaches maximum velocity then t_1 is infinity, t_2 is the time when the robot starts to slow down its velocity, t_f is the time when the robot reaches g_n. Besides, we let v_0 denotes the initial velocity when the robot reaches g_0, and let s_g denotes the distance from one grid to another. Figure 3.3 shows how to get the value of t_1 and t_2 and Figure 3.4 shows the algorithm used to calculate the time needed for a robot from one to another grid.

After that, we can use the method shown in Figure 3.3 to know the value of t_1, t_2. Every time the reserved grids of a robot change, such as a grid is released or success to request a new grid, we can recalculate the t_1 and t_2 to make sure the robot can move in maximum speed subject to it able to stop at the last reserved grid or stop at the reserved grid before it makes a turn. On the other, it is necessary to know what is the transition time for a robot moves from one grid to another and this is the objective to design the algorithm shown in Figure 3.4.

Let:

1. Assume velocity of robots are not limited.
2. t': time spent in acceleration.
3. t'': time spent in deceleration.
4. Total time needed: $t = t' + t''$.
5. Total distance move: s.

Robots required to decrease their velocity to 0 once they reach last grid in their path.

Thus, $t = t' + \frac{a_{max}t\prime + v_0}{d_{max}}$, and

$$s = \frac{1}{2}(v_0 t' + a_{max}t'^2) + \frac{1}{2}(a_{max}(t - t'')^2)$$

$$= \frac{1}{2}\left(v_0 t' + a_{max}t'^2\right) + \frac{1}{2}\left(a_{max}\left(\frac{a_{max}t\prime + v_0}{d_{max}}\right)^2\right)$$

Value of s is known, thus, by solving the equation above, we can get the value of t' *(always take the positive and real number)*, and therefore both t and t'' can be solved. However, this calculation is not always correct due to the first assumption. In fact, robots are limited by speed $\mathrm{v_{max}}$. Therefore, we need to validate whether the robot exceeds the speed $\mathrm{v_{max}}$.

A robot does not exceed the speed v_{max} if $a_{max}t' + v_0 \leq v_{max}$. If true, we can use the previous solution but if it is false, we need to introduce a new variable t''' which indicate the total time the robot is remaining its maximum speed. Then, the t', t'', t''' are calculated by

$$t' = (v_{max} - v_0)/a_{max}$$

$$t'' = v_{max}/d_{max}$$

$$s' = s - \frac{1}{2}(v_0 + v_{max})t' - \frac{1}{2}(\mathrm{v_{max}})t''$$

$$t''' = s/v_{max}$$

Figure 3.3

Time needed for robots to acceleration / deceleration / remain maximum speed.

INPUT: $\mathbf{a_{max}}, \mathbf{d_{max}}, v_{now}, v_{max}, t_{now}, t_1, t_2, s$

OUTPUT: Time needed for robot to reach next grid; New velocity.

CALC_TIME_NEEDED ()

```
IF t_now < t_1
    result = Max ( Solve t : s = v_now t + (1/2) a_max t^2 ) # Ignore Complex Number.
    IF t_now + result > t_1
        t' = t_now + result − t_1
        v' = v_now + a_max t'
        s' = s − (1/2)(v_now + v')t'
        t, v_new = CALC_TIME_NEEDED(a_max, d_max, v', v_max, t_1, t_1, t_2, s')
        RETURN t + t', v_new
    ELSE
        v_new = v_now + a_max t'
        RETURN result, v_new
    END IF
END IF
IF t_1 ≤ t_now < t_2
    result = s/v_max
    IF t_now + result > t_2
        t' = t_now + result − t_1
        s' = s − v_max t'
        t, v_new = CALC_TIME_NEEDED(a_max, d_max, v_max, v_max, t_2, t_1, t_2, s')
        RETURN t + t', v_new
    ELSE
        RETURN result, v_now
    END IF
END IF
IF t_now ≥ t_2
    result = Max ( Solve t : s = v_now t − (1/2) d_max t^2 )
    v_new = v_now − d_max t'
    RETURN result, v_new
```

Figure 3.4

Algorithm to calculate time needed for a robot to move from one to another grid.

3.7 Multi-Mobile Robot System Design

3.7.1 Type of Multi-Mobile Robot System

To avoid any ambiguous, this section clarifies the definition of centralized and decentralized systems that be used from this section onward.

In this report, a system is called a centralized system only if all the decision-making process is made by a centre unit, and all the agents are only an execution unit of the system. On the other hand, a system is called a decentralized system only if all the agents made the decision by its own and there is not any centre agent that gathers and maintain the agents' information. Besides, any multi-robot system that does not belong to either one of them is called a hybrid robot system. In a hybrid robot system, the decision-making process can be made either by robots or servers, this depends on the specific implementation.

3.7.2 Proposed Framework

This project proposed a hybrid multi-robot control framework with high modification flexibility to fit the actual situation. The proposal framework in this project able to reduce the robot dependency on the centre server at the same time without losing much potential benefit of global information. The main idea of the proposed framework is simple, which is the centre server is mainly working as a task provider, resources manager and information provider and most of the decision making is made by robots itself. Besides, if the server has available computation resources, it can help the robots to make a better decision.

On the other hand, it's worth to mention that the proposed framework consider the parcel sorting environment as a grid world. The grid world design can simplify the algorithm and provide stability to the multi-mobile robot system because it can be a simple robot localization approach by installing RFID on each grid (Abdelgawad, 2014). Also, it's widely used in most of the multi-mobile robot system in a known environment. For instance, the Kiva system that implemented by Amazon warehouse (Guizzo, 2008) and the multi-mobile robot parcel sorting solution implemented by LiBiao Robotics company.

Next, the proposed framework views all the grids as mutual exclusive resources so that it can only be occupied by one robot. Therefore, if a robot requires to access a grid, the robot needs to request permission from the centre server. Once a grid is reserved for the robot, the grid is locked and no other robots able to access it until the grid is released by its owner. To increase the velocity of robots, each robot can reserve a certain number of grids, and this will be further discussed in Chapter 4. Besides, if there have multiple robots reserve the same grid, the centre server always assign these grids resources based on first come first serve principle.

3.7.3 Flow of Proposed Framework

The sequences below show the normal flow of a robot to complete its task under the proposed framework by assuming the robot known it destination points:

1. Robot requests the environment state information from the centre server.
2. Robot plan it trajectories based on the latest environment state.
3. Robot requests certain number of future grids that it will reach.
4. Robot reached a new position and notify the centre server so that the centre server can release the used and unoccupied resources.
5. Repeat from step 1 until the robot reached its destination.

For step 3, the maximum number of grids reservation for each robot is a modifiable parameter. Besides, there are a few points to note here. All these points are not hard constraints, but they can simplify the complexity or improve the performance of multi-mobile robot systems. First, for step 2, the robot can choose to re-plan its trajectory if the latest environment state is changed and the change might bring a negative effect to the robot. For example, some area becomes congestion. This mechanism provides the robot with the flexibility to choose a better path based on the new environmental state. Second, the robot should re-plan the trajectory from the grid resources that have not been reserved by it so that the speed control module can always adjust its speed based on the reserved resources. Third, other robots should not be considered an obstacle when selecting a path because it introduces a longer path.

3.7.4 Role of Robots

In the proposed framework, the actions of robots are as follows:

a) Select a pickup point.
b) Plan its moving trajectories based on the current and destination position.
c) Communicate with the centre server.
d) Control its speed and motion to connect itself to the charge station.
e) Drop the parcel when it reaches the destination.
f) Stop and wait for the worker to assign a parcel to it.

This project considers actions *d, e, f* as low-level built-in functions of robots because they are not related to the high-level multi-mobile robot planning processes. In other words, the robots should be able to perform these actions without the involvement of the multi-mobile robot control framework.

3.7.5 Behaviour of Robots in the Proposed Framework

In a high-level view, the behaviour of robot can be simplified as follows:

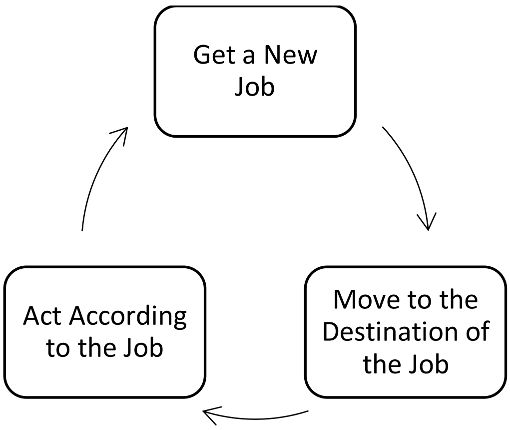


Figure 3.5 Behaviour of Robot / Job cycle

Figure 3.5 can also be viewed as a job cycle of robots. In the cycle, a robot starts to do something because it is assigned a job. In our case, the job can be to pick up a parcel, to drop a parcel, or to charge its battery. Next, the robot moves to its destination to complete the assigned job. After it reaches its destination, it acts according to the job, for instance, drops the parcel it is carrying to a drop-off point. Next, we investigate how a robot moves to its destination in our proposed framework.

In our proposed framework, the robot cannot move without reserving the grids it intends to move to. Thus, the robot-server interaction module keeps requesting the grids it will move to. If a grid is requested and reserved by the robot, the robot-server interaction module sends the grid information to a queue in the motion control module which the queue indicates where should the robot move to. The robot-server interaction module might make a new plan due to some reason, for example, it failed to reserve a grid after a certain period. In this situation, the motion control module has no idea about what is happening because it only controls the robot based on the reserved grids that store in a queue. Also, once the robot leaves a grid, it will send a notice to the robot-server interaction module so that it can release the unoccupied grid. On the other hand, one the robot reaches its destination, the robot act according to its mission. After that, the mission is completed, thus repeat the loop in Figure 3.5.

3.7.6 Pickup Point Selection Strategy

In the proposed framework, the pickup point selection strategy is made by both server and robot. The server mainly determines which pickup points should be removed from the list of candidate pickup points, while the robots select a pickup point suitable for them and these pickup points should exist in the candidate pickup points list. After that, the server doesn’t care which pickup point the robot go to.

This provides flexibility to design the pickup point selection strategy. The server might remove some pickup points that might overload, or it can simply treat all the active pickup points as candidate pickup points while removing inactive pickup points. This type of job is suitable for the server because it maintains global information.

In addition, the robots decide which pickup points it should go to. For instance, they might select the pickup points near to them based on the Manhattan distance or Euclidean distance. The algorithm used in this project is to select the recommended pickup point by using the Manhattan distance. Next, the pickup point is used as a reference pickup point for heuristic function in A search algorithm. Therefore, even a reference pickup point is selected, the robot is not necessary to move the pickup point. The actual pickup points the robot chooses is determined by which is the first pickup point it searched in A search algorithm. This is because the actual distance for a robot moves from its current position is not proportional to the Manhattan distance due to the cost of moving to each grid can be different, and thus the selected path might not always base on minimum distance but minimum cost.

In addition, if an area nearby a selected pickup point become congestion, these mechanisms allow the robot to switch to another pickup point without notice to the server and thus reduce the server workload. Also, this mechanism able to reduce the chance of deadlock to happen due to the congestion.

3.7.7 Path Planning Algorithm

The path planning algorithm used in this project is A search algorithm. A search algorithm is like the Dijkstra algorithm, but it introduces a heuristics function. As mentions in the previous chapter, A search is path planning algorithm popular in grid world environments because its efficiently. If its heuristics function never overestimates the cost of the path, then it always finds optimal solutions. In our proposed framework, the heuristic function used is Manhattan distance between the start vertex and end vertex. If all the cost from moving one grid to another is less than or equal to one, then the A search algorithm with Manhattan distance as heuristic function always provide the path with minimum cost. However, the cost for travelling from one to another grid might not always 1, also, it may not necessary to find an optimal path all the time because the sub-optimal path might good enough.

Therefore, we use a variable *greedy* to control the value of heuristics function used in the A search algorithm.

$$\text{heuristic cost} = \text{greedy} * \text{ManhattanDistance}(g_{\text{start}}, g_{\text{end}})$$

If we increase the value of *greedy* then the A search algorithm is more like a greedy search algorithm, and if we decrease the value of greedy, the A search algorithm is more like the Dijkstra algorithm. Thus, *greedy* is work as a controller of computation speed and optimality trade-off.

On the other hand, the A search algorithm here will accept a set of grids which these grids are blocked and thus not be searched. Thus, the algorithm shown in Figure

3.6 can be implemented, in fact, it is implementing in our robot pathfinding algorithm. Figure 3.7 shows the pseudocode of A search algorithm. Please note that when calculating the cost from moving one grid to another, the cost of turning is also calculated. If the previous moving direction is different to the next moving direction, an additional cost will be added. Therefore, the path planning algorithm will reduce the number of turns if possible.

Besides, a grid is considered failed to reserve if the robot waits more than a certain amount of time and the number of reserved grids of the robot does not exceed the allowable limit. Failure to reserve a grid might due to congestion of the area. In this situation, the path planning mechanism allows the robot to leave the congestion area by searching for another path.

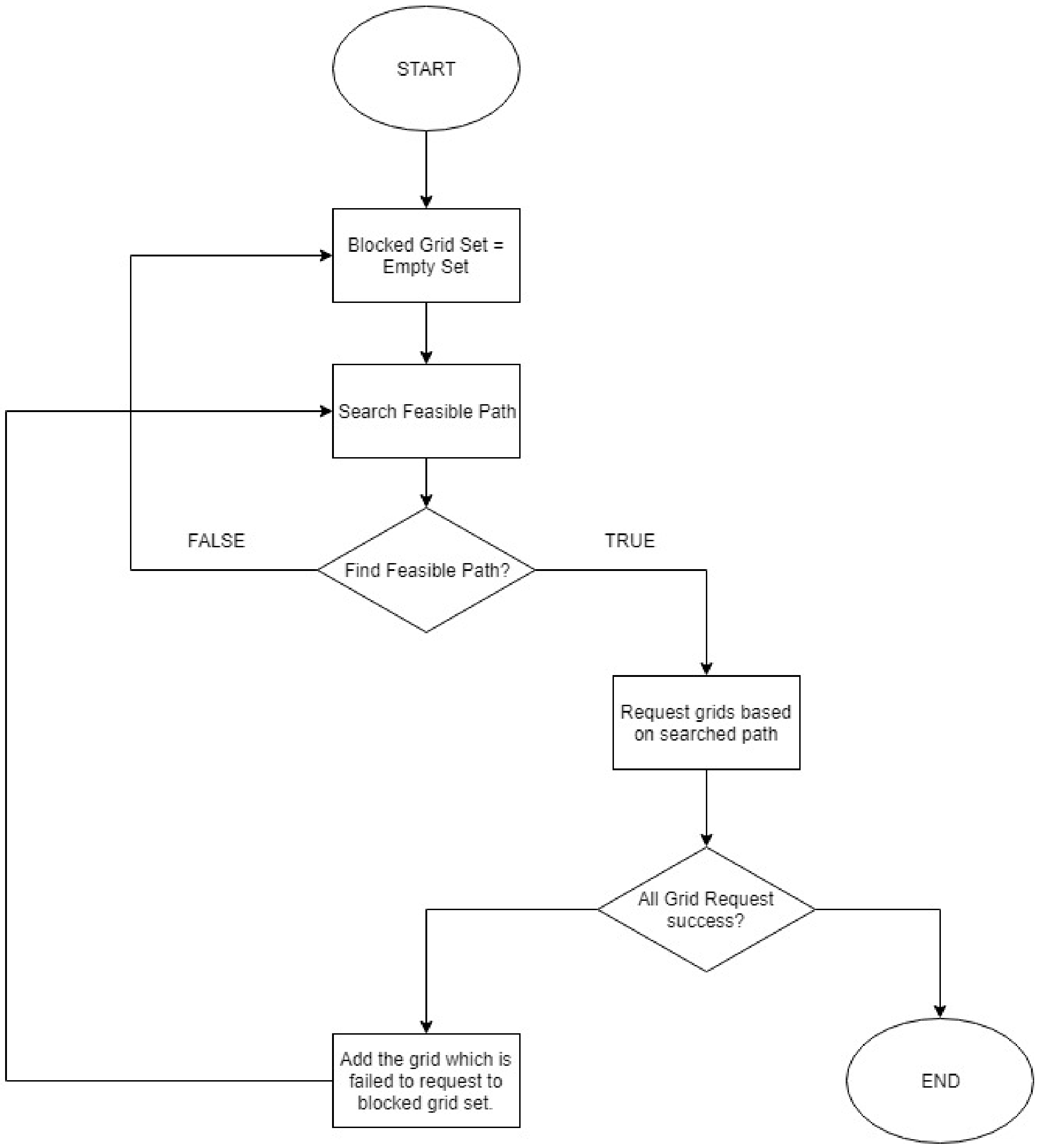


Figure 3.6 Path Finding Algorithm

INPUT: *StartVertex, EndVertices, BlockedVertices, Greedy*

OUTPUT: *Paths*

A Search ()

```
SET OpenList = [], ClosedList = [], Solutions = [], Nodes.Path = []
SET Nodes.Vertex = StartVertex
SET Nodes.RealCost = 0, Nodes.Cost = 0
Add Nodes INTO OpenList
LOOP UNTIL OpenList <= 0
        LET node = Node that has minimum cost in OpenList (Implement by Heap)
        POP node FROM OpenList
        ADD node.vertex INTO ClosedList
        FOR v IN adjacent vertex of node.Vertex AND
            v NOT IN BlockedVertices AND v NOT IN ClosedList
                APPEND v TO node.Path
                node.Vertex = v
                node.RealCost = node.RealCost + 1
                IF direction to v is different from previous
                        node.RealCost = node.RealCost + 1
                END IF
                node.Cost = node.RealCost + Heuristic*Greedy
                APPEND node TO OpenList
                IF node.Vertex IN EndVertices
                        ADD node INTO Solutions
                        RETURN Solutions
                END IF
        END LOOP
END LOOP
```

Figure 3.7 Pseudocode of A Search Algorithm

3.7.8 Charge Scheduler

Each time the robot completes the task, the robot needs to request the next task from the central server. Task types can either continue the sorting task or charge the battery and this is determined by the centre server.

A robot needs to charge if its battery level lower than a charge threshold. However, if there is not any available charge station, then the robot continues its task, that is pick up and drops a parcel. After that, it retries to request an available charge station.

Apart from that, if a charge station is assigned to a robot, then the charge station is dedicating to the robot within a certain amount of time. During the time, the charge station is not pre-emptive, that is even there have other robots have higher priority than the robot, they cannot interrupt the charging process of the robot. This prevents robots to waste their time moving to a charge station, but it charge process might be interrupted immediately. However, the reservation of a charge station cannot be too long because it might cause other robots that have a lower battery level to fail to request any charge station.

The charge station selection algorithm will first search these charge stations which is empty. If all the charge stations are not available, then the charge station selection algorithm will search from lowest priority (highest priority value) charge station and the searched priority value is not less than the robot's priority value + *search gap*, for each iteration, the priority value to search is decreased by search gap. The priority value of the charge station is determined by the robot's battery that is occupying the charge station while the priority value of the robot Is its battery level. For example, if a robot's battery level is 20, then it is its priority value. The search algorithm will first search for any unoccupied charge stations. If not exists, then search from priority value [95, 100], If still not exist, then search for priority value [90, 100], and so on. The minimum priority value for the charge station selection algorithm to search in this case is [25, 100]. Next The charge of the charge station A, B and C with priority values 26, 27, 24, respectively. The charge station A and B will be selected as candidate charge stations. The final selection is based on its Manhattan distance between the given robot.

In the nutshell, this mechanism allows to select multiple candidate charge station and then select the charge station near to the robot that requires charge station. This reduces the overhead in moving to the charge station. Also, it is not a good design if a robot able to interrupts a robot that has similar battery power as itself. It might cause robots to keep interrupt others, but no one spent their time in charging their battery. That is why the *search gap* and *charge station reservation tim*e is introduced.

3.8 Sorting Centre Layout

The multi-mobile robot sorting centre consists of 4 important components, which is robots, charge stations, pickup points and drop-off points. Figure 3.8 shows a simulation of a multi-mobile robot sorting centre. The blue points are mobile robots, grey grids are the centre of drop-off points, purple grids are pickup points and the charge stations are light green colour square at top, left and right side. Apart from that, there have 506 drop-off points, 44 pickup points and 108 charge stations in the simulation environment. Besides, the environment is formed by 75×75 grids and each grid have size $0.6\text{m} \times 0.6\text{m}$ and has total area $2025m^2$.

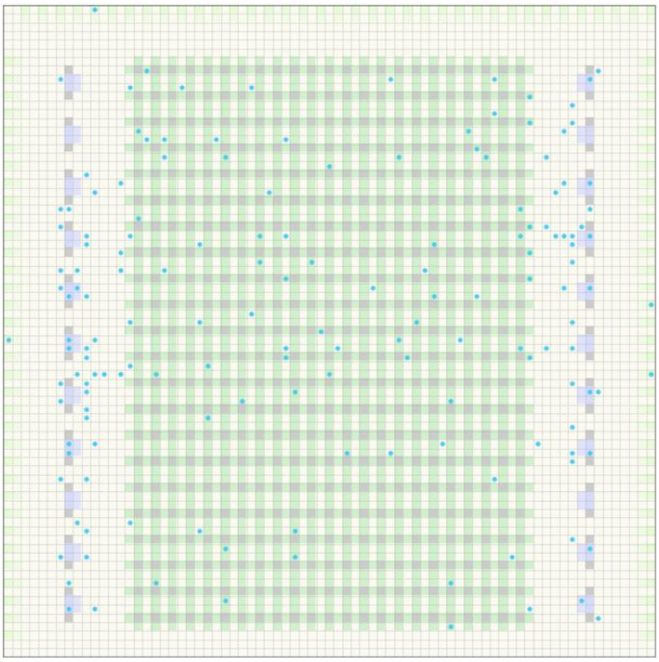


Figure 3.8 The Layout of The Simulation Environment

3.9 Robot Pathfinding Preference

The proposed framework provides high flexibility in designing robot path planning algorithms. However, due to the consideration of computation ability, A search algorithm might be a good choice even in a multi-mobile system, but it never considers the conditions of other agents. This makes the robots in the system become greedy, they trying to move in minimum cost even though it might hurt others and thus affect the overall performance. In this situation, we might want to provide some rule to reduce the chance of conflict to happen. For example, we might set some path to unidirectional to avoid head-to-head collisions.

By considering the grids environment as a directed graph, we can easily implement this by removing some of the edges in graphs. However, once an edge is removed, the edge will never be used by robots to establish a feasible path. Sometimes, it is unnecessary because this reduces the flexibility for the robot to select their path. Thus, instead of provides “rules”, we might provide our preference by controlling the cost of each edge. Thus, robots can break our preference if the robots think it is worth.

In this project, both methods are used to control the robot behaviour. The implementation will be discussed in Chapter 4 whereas the result of the implementation will be discussed in Chapter 5.

3.10 Assumptions

Due to time constraints, some assumptions were made in this project, so this project can focus on the multi-mobile robot control framework.

- The robots are required to slow down completely before making a turn.
- There always has perfect robot-server communication.
- The battery capacity of robots can long 8 hours for a robot in a moving state. In addition, it takes 90 minutes for a battery to go from empty to full charge.
- They have an unlimited parcel to sort in order to know the maximum throughput.

3.11 Parameters

Number of Pickup Points	***44***
Number of Drop-off points	***506***
Number of Charge Stations	***105***
Number of Grids	***75 x 75***
Maximum Waiting Time	***3200ms***
Grids Size	***0.6m x 0.6m***
Charge Threshold	***30***
Search Gap	***5***
Charge Station Reservation Time	***10 minutes***
Greedy for A search Algorithm	***1 / 4***
Time Spent in Picking Up a Parcel.	***800ms***
Time Spent in Dropping a Parcel	***800ms***
Maximum Number of Reservation Grids for Each Robot	***5***
Maximum Acceleration, Deceleration, Velocity of Robots	$\boldsymbol{3ms^{-2}, 3ms^{-2}, 3ms^{-1}}$

Table 3.1 Parameters Setting

3.12 Performance Indicators

It is important to have performance indicators to determine the efficiency of a multi-mobile robot system. The main performance indicator in this project is throughput / hour. However, only knowing throughput / hour is not enough to provide clues to investigate bottleneck and understand the system. Thus, other performance indicators that help us in understanding the system are average moving speed of robot, average moving distance per job cycle, average throughput per hour and per robot and average time spent in completing a job cycle. The performance indicators used in this project show in Table 3.2. Besides, Figure 3.9 shows the equations used to calculate these indicators.

Throughput / hour	The number of sorted parcels per hour in the simulation environment.
Average Robot Performance	The number of parcels sorted per robot per hour in the simulation environment.
Average Speed of Robots (m/s)	The average speed of robots in the simulation environment.
Average Moving Distance / Job Cycle	Average distance moves to complete a job cycle in the simulation environment.
Average Time spent / Job Cycle	Average time spent to complete a job cycle in the simulation environment.

Table 3.2 Performance Indicators

m = total number of robots.

p_i = the number of parcels sorted by robot i.

d_i = total moving distance of robot i in meter.

t = total running time (hour) of the sorting centre.

z_i = total number of job cycles of robot i.

a_i = total time spent (hour) in waiting for a job to complete of robot i.

For example, picking up a parcel, dropping parcel, and charging its battery.

Throughtput / Hour:

$$\frac{\sum_1^m p_i}{t}$$

Average Robots Performance:

$$\frac{\sum_1^m p_i}{t \times m}$$

Average Speed of Robots (m/s):

$$\frac{\sum_1^m d_i}{60 \times 60 \times (t \times m - \sum_1^m a_i)}$$

Average Moving Distance / Job Cycle:

$$\frac{\sum_1^m d_i}{\sum_1^m z_i}$$

Average Time Spent / Job Cycle:

$$\frac{\mathrm{t} \times \mathrm{m}}{\sum_1^m z_i}$$

Figure 3.9 Equations for Performance Indicators

Chapter 4 Design Consideration

4.1 Multi-Mobile Robot System

The benefit of the decentralized multi-mobile robot system is it provides scalability in term of the change of the number of robots. Also, it is robust to a complex and dynamic environment. For the centralized approach, the centre unit required more processing power as the number of robots growing and this might become a bottleneck to the scalability of the system.

Yet, theoretically, centralized multi-mobile robot system can provide equal or better performance compared to the decentralized multi-mobile robot system because it has more information to make a better decision, however, to utilize all the information to achieve optimal solutions might be difficult since most of the multi-mobile robot problems are NP-Hard such as multi-mobile robot path planning and multi-mobile robot task allocation. This shows that even the centralized approach is adopted, it is still difficult to guarantee the optimality of solutions.

There is no one size fit all solution. Thus, the best multi-robot system can be completely different in a different situation. For the multi-mobile robot sorting centre, it’s difficult to avoid the role of the centre agent because it is required to handle some business logic. For example, the robot is required to know where a parcel required to place to, and it is important to keep track where the parcels are going to and which parcels are sorted, and which are not. Also, sorting centre environments are usually known and controllable, so there is no reason to give up the potential performance acceleration from the centralized method. Thus, we can conclude that the multi-robot system in sorting centre is difficult to be a decentralized system.

However, if the multi-robot system is implemented in a centralized approach, the scalability of the system is scarified. Also, robots nowadays normally have some computation power, if the robots only follow the instructions of the central unit, it might waste the computation resources of robots. Thus, the robots can in charge of some decision-making process to reduce the server workload and thus increase the scalability.

In the nutshell, a good multi-robot control framework or system for sorting centre should be located somewhere between a centralized system or hybrid system and it must be flexible to various type of trade-off.

4.2 Lock Grids Strategies

Since the space of sorting centre is always limited, it's important to know how many spaces are required by a robot. The more the number of spaces required by a robot, the easier for congestion to happen. For the reactive paradigm, a robot occupies only one or two grids at a given time if the robot only moves in either horizontal or vertical. It occupies one grid if it is completely inside a grid whereas occupies two grids if it is moving from one grid to another. Because the robot in reactive paradigm solve conflict locally and make their decision locally, no future grids are required, thus two grids are the maximum occupied space for a robot. However, to make sure no collision occurs, the robots might require slowing down their speed because it's difficult to guarantee the next grid it moves to will not occupy by other robots.

The proposed framework is a hybrid approach so that the robots able to request some future grids they will move to and the server able to guarantee these grids will not be occupied by other, and thus increase robots' speed. But in this case, each robot requires more grids, this might affect the overall performance due to the congestion. For example, if a robot request all the grids they will reach in their current plan, its speed can be maximized, but other robots might be blocked because they might require the grids occupied the robot. Therefore, the selection of how many grids can be occupied by each robot in any given time becomes an important consideration.

Please note that all the discussion above is built on top of the adopted lock grid strategies which is once a grid is locked, it will not be assigned to others robot. This type of strategy is used because it might difficult to guarantee the expected time for a robot to reach and leave a given grid in a real system. To understand this, we now assume we have a perfect system, and we know the expected time interval for each robot to reach and leave each grid, and the expected time is always correct. In this situation, instead of reserving grids to a robot, we reserve grids to a robot only in a time interval.

However, this assumption might too optimistic. If robots might make some small mistake every move, the error of the expected time for a robot to reach future grids can accumulate and thus the error bound become bigger if the time difference is huge. To overcome this, the time interval for reserving grids need to be bigger. These cause robots unable to request a grid that is reserved by other robots, even though, in reality, they use the grid at different time. In this case, each robot needs to occupy more space. This will give us to abandon our original goal which is to reduce the space required by robots.

Therefore, the proposed lock grid strategy might be a simple and efficient strategy to reduce the space required by each robot and to increase the moving speed of the robot as long as we set a good maximum number of allowable reservation grids of robots.

4.3 Rules / Preference for Grid Environment

This section we not only discuss rules but only on the preference that stated in the previous Chapter. As mention before, it's difficult to make a globally optimal solution especially the path planning problem in a multi-robot environment. Thus, we might need to define some rules manually or control the cost of connected grids to inflect the path planning behaviour of robots to improve the overall performance. For simplicity, here, we consider the grid environment as a graph in computer science.

4.3.1 Configuration A

First, we look into the most naïve version of graph design, or we called configuration A, which is the graph has no logical restriction at all. That is, if two grids are physically connected, then in the graph, they are connected by an edge with the cost of one in bidirectional. However, for these grids are placed with a resource such as pickup points, drop points or charge stations, we still follow the rules of the resources and we consider it is a physical constraint. For example, any drop-off point has a centre that used to drop the parcel thus it’s impossible for a robot moves to the grids.

Obviously, the configuration A is troublesome, since most of the grids is allowing the robot moving in bi-direction, head-to-head collisions can happen easily unless the robots have a good path planning algorithm that utilizes all the information of other robots efficiently. This is clearly out of the capability of simple A search algorithm. Thus, the actions of robots can always like this:

- *A robot faced head-to-head collision.*
- *The robot waits for a certain amount of time.*
- *The robot failed to request the grids it needs to move to.*
- *The robot makes a new plan to avoid accessing the grid.*

If this situation occurs, the robots spent more time to complete its task, also, the moving distance is longer because it needs to make a new plan which is typically worse than the original path in term of cost.

4.3.2 Configuration B

Thus, we proposed the second configuration or called configuration B. In configuration, most of the neighbour grid is connected by only one direction edge. That is, it a grid A and grid B is physically connected, then if a robot is allowed to move from grid A to grid B then the robot is not allowed to move from grid B to grid A. In the graph, we can implement this by removing the edge from vertex B to vertex A. However, for consistency in converting the grid environment to graph, in our implementation, we set the cost of edge B to A to infinity.

This configuration solves the head-to-head collision; however, it introduces additional moving distance for a robot from one grid to another. Thus, to minimize this, two nearby grids can be connected to become a road. The allowable direction of the two grids is in the opposite direction. Therefore, the increases in the distance of the shortest path are small. Thus, the benefit of this design can easily cancel the negative impact it brought.

4.3.2 Configuration C

Even though the configuration B solves the reduce the chance for head-to-head to the collision, but it still has a room of improvement. Figure 4.1 shows the snapshot of the simulation environment that implementing configuration B and the red rectangle in Figure 4.1 shows the main activity area of robots in the environment. From the picture, we can notice that the area outside the red rectangle is rarely used. This is wasting resources. In here, we call the area outside the red rectangle as the outer ring.

To avoid the situation above, we control the moving cost to encourage the robots to utilize the area outside of the red rectangle. However, it is not easy to set up a suitable cost to control the robot's behaviour. For example, if we set the cost of the outer ring too low, most of the robots will utilize the outer ring, and thus the utilization of area inside the red rectangle becomes less and the congestion can happen in the outer ring. Also, when the robots use the outer ring to travel to its destination, the travel distance is higher.

On the other hand, even though changing the cost of a graph can affect the preference of path planning algorithm, but the effect can be cancelled by heuristic function. For instance, the heuristic function used in this project is Manhattan distance between the current vertex and the goal vertex. If the set the greedy of our path planning algorithm to a higher value, the robot tread to find the shortest distance path instead of the lowest cost path. Thus, it is important to consider this situation when designing the cost of the graph.

Also, to get better performance, the vertical path in the inner area is also decreased. The vertical path near the centre has a lower cost compared to the vertical path near to the pickup points. This is because the probability for a robot to pass through the vertical path is highest when the vertical path is near to pick up point and the probability for a robot to pass through a vertical path is lowest when the vertical path is at the centre. If robots prefer to utilize the centre vertical path compare to the vertical path near to the pickup points, the probability of the cross-collision can be reduced. Figure 4.2 shows the snapshot of the simulation environment that implementing configuration C.

We believe that there exists a better way to design the moving cost. We might find better moving cost set up by using some random search technique such as Genetic Algorithm, Particle Swarm Algorithm and even learn the cost by using machine learning algorithm such as reinforcement learning. However, this is out of the scope of this project and will not be further discussed in here.

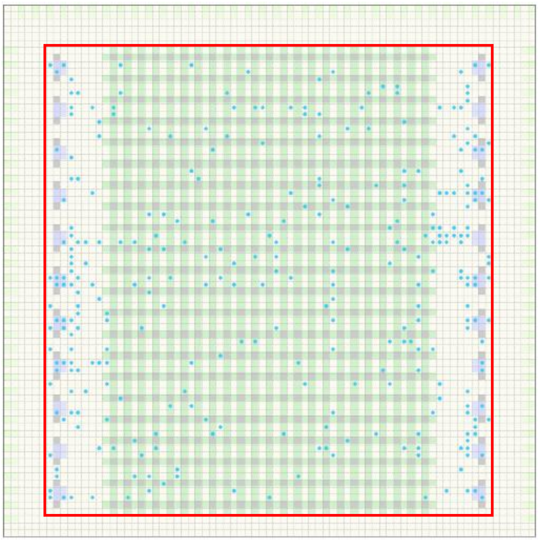


Figure 4.1 Simulation Snapshot - B

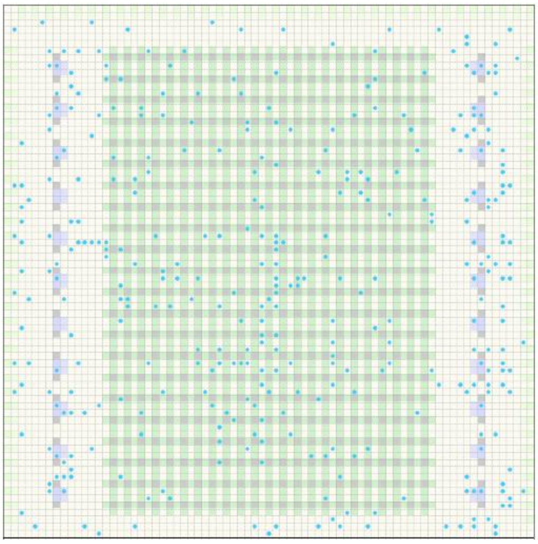


Figure 4.2 Simulation Snapshot - C

Chapter 5   Experiment Results & Analysis

5.1   Experiment Design

The objective of the experiment is to know the performance of the sorting centre that implementing the proposed framework. As mentioned in the previous chapter, the key indicator is the throughput per hour of the sorting centre in term of the number of parcels sorted. Apart from that, the indicators such as average robots' performance, average robots' moving speed, the average distance for completing a task cycle, and the average time spend in completing a task cycle is used to analyse the results. On the other hand, the number of robots used in the experiment is 10, 60, 110, 210, 260, 310, 360, 410 to investigate the performance of the proposed framework when the number of robots is different. Next, the experiment can be further split into three parts.

For the first part of the experiment, we investigate and analyse the different rules and preferences configuration discussed in Chapter 4. The running time for each configuration is 2 hours in simulation time (the time in the simulation environment). Besides, the number of robots used in this experiment is 10, 60, 110, 160, 210, 260, 310, 360 and 460. On the other hand, the rules and preferences used are Configuration A, Configuration B and Configuration C that mentioned in the previous Chapter. Next, to avoid unnecessary noise, this part of the experiment assumes the robots have an infinite battery.

For the second part of the experiment, only the best rules and preferences configuration in Part A is used. Next, the maximum allowable grids reservation numbers are set to the 2, 5, 8 respectively. The number of 2 is the minimum number of grids reservation that can be set to allow the proposed framework working normally. This is because a robot needs to reserve the grids where the robot currently occupied at and the grid it is travelling to. Besides, note that the default value of the maximum number of grids reservation of each robot is 5. Next, the simulation time for this part is 2 hours, and the battery of the robots is infinite.

After that, the third part of the experiment aims to investigate whether the performance of the sorting centre that implementing the best configuration of the proposed framework is stable, that is if congestion occurs, it should be resolved after some time and should not cause deadlock. Also, this part of the experiment is aiming to provide a more accurate simulation result for the overall throughput per hour and unlike others, this part of experiment considers the power consumption and the battery charging process of the robots. The simulation time in this part of the experiment is 10 hours.

5.2 Results Analysis – Part 1

5.2.1 Throughput and Average Robots Performance

Figure 5.1 shows the average robot performance and Figure 5.2 shows the throughput per hour of the sorting centre. The two indicators are discussed together because they are highly correlated. Figures 5.2 shows the throughput of all configurations growing with a curve shape when the number of robots is small, but throughput for configuration A and B drop sharply when the number of robots becomes huge. Apart from that, Figure 5.1 shows a similar situation. At first, the average performance of robots for each configuration decreases slowly as the increasing number of robots but for configuration A and configuration B, the average performance of robots drops dramatically start from some points.

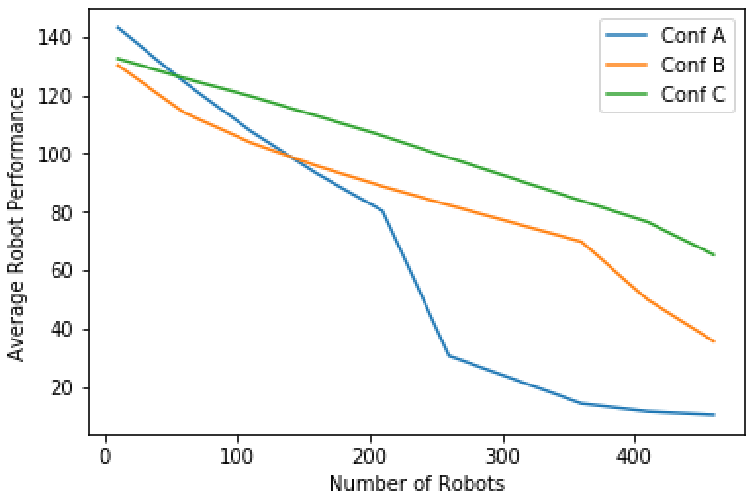


Figure 5.1 Average Robot Performance

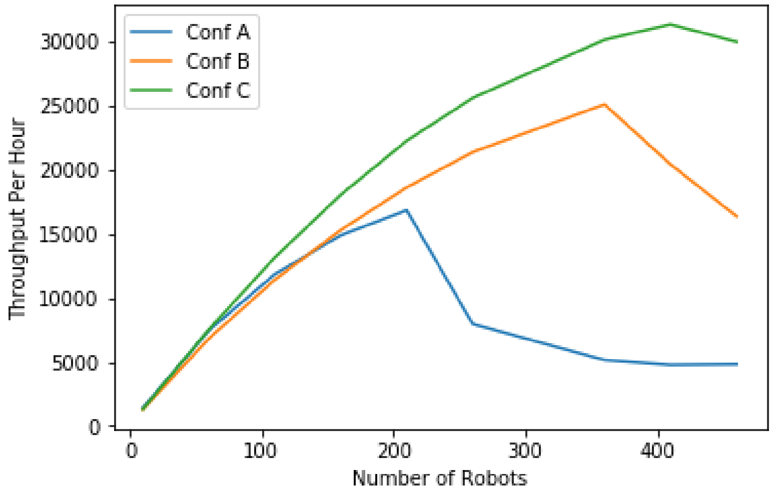


Figure 5.2 Throughput Per Hour

Obviously, configuration A and configuration B is unstable compared to configuration C. This is because the congestion situation in configuration A is more serious compared to B and the congestion situation of configuration B is more serious compared to configuration C. These results are predictable because the configuration B is trying to reduce head-to-head collisions while the configuration C did the same thing as configuration B but also trying to utilize as much space as possible to avoid congestion.

As mentioned previously, the configuration A is the most naïve version which in this configuration, only the minimum rules and no preferences are provided. Thus, the greedy behaviour of robots can easily cause the head-to-head collision, and thus need to take some time in resolving the conflict. During this time, other robots might come and complicate the situation. Finally, the congestion becomes unsolvable and thus the throughput drops significantly.

On the other hand, although the configuration B avoid most of the head-to-head collision, the robots still spending time in cross-junctions. When the number of robots becomes huge, congestion can also happen. In spite of the configuration C also facing the cross-junctions but the situation is not that serious as configuration A and B because the configuration C is encouraging the robots to use the outer ring in the sorting centre, and this reduces the chance of conflict by utilizing more space and forcing the robots to move in the same direction. This design is similar to the highway in road design.

5.2.2 Average Moving Speed

Figure 5.3 shows the average moving speed of the robots, it shows that when the robots numbers are small, that is 10, then the robots in configuration B have the slowest average moving speed which is 1.754 m/s, whereas the robots in configuration C have the highest average moving speed which is 1.926 m/s and this moving speed is similar to the robots in configuration A which is 1.917 m/s. Next, the robot's average moving speed for configuration B surpass the configuration A when the number of robots near to 150, while for the robots in configuration C, their moving speed is far higher than others when the number of robots is huge.

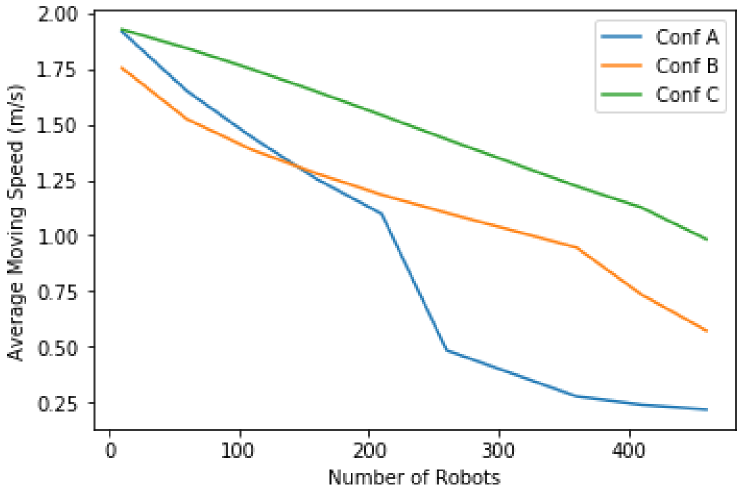


Figure 5.3 Average Moving Speed

Apart from that, the average moving speed of robots in configuration A drop dramatically when the number of robots after the number of robots exceed 210. The situation is also like configuration B when the number of robots exceeds 360. Unlike others, the average moving speed for robots in configuration C is relatively like a straight line with an approximate gradient of -0.002 m/s per robot. But, when the number of robots reaches 410, the approximate gradient drops to -0.003 m/s per robot. Although the gradient only drops slightly, it is reflecting the proposed framework and the predefined preferences and rules unable to solve the congestion situation efficiently. By comparing Figure 5.2 and Figure 5.3, it is easy to note that once the gradient in Figure 5.3 drop, the overall throughput also drops.

This result shows an intuitive situation, which is when the number of robots increases, the average speed of robots decreases. This is because the robots need to wait for others to avoid the collision. Therefore, it is important to select a good path which minimizes the possibility that the selected path will affect other robots. To achieve this, some advance but normally higher complexity algorithms might be used, but this project shows that this can be achieved by using a simple A search algorithm, and by modifying the moving cost from a grid to another to control the robots' path planning behavior.

5.2.3 Average Moving Distance / Job Cycle

The average moving speed of robots does not mean all because the robots might have a slower speed but moving in a short distance and thus complete its tasks in a shorter time. Therefore, the average distance for robots to complete a job cycle was investigated, this information is shown in Figure 5.4.

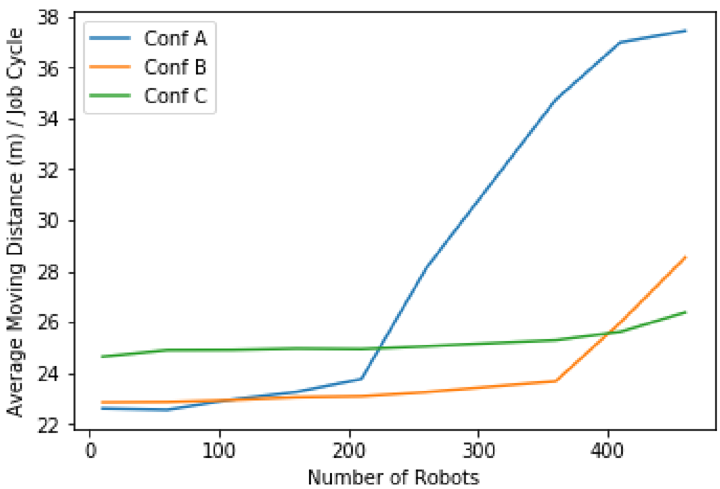


Figure 5.4 Average Moving Distance

Figure 5.4 shows that when the number of robots is small, the average moving distance for robots to complete a task cycle in configuration A is shortest and this distance is similar to robots in configuration B. This situation is because the robots in configuration A always searches the shortest path. As long as the robots not blocked by others, and thus it doesn't require to change its path to resolve the conflict, the selected paths are always optimal in term of distance. For Configuration B, the moving distance is only slightly higher than Configuration A although most of the grids are restricted to unidirectional. This is because in most of the time, two opposite directional paths are near to each other in our design, thus the increase of the distance is limited.

Apart from that, the average moving distance for robots to complete a task cycle in configuration C is far longer than others. The reason for this situation is obvious because the robots in configuration C utilize the outer ring of the sorting centre and this introduces additional distance. But, thanks to that, the average moving distance for configuration C is growing slowest as the increasing the number of robots, also, it is most stable amongst all other configurations. Thus, it's easy to infer that the average moving distance for configuration B and C will finally higher than the average moving distance in configuration A. In fact, this situation happens is far faster than expected, since the performance of configuration B and C is not stable. That is when the number of robots increases, the robots in these configurations spent a lot of effort to overcome the conflicts but the effect is not obvious, thus their average moving distance is higher than configuration C when the number of robots reaches 210 and 360 respectively.

5.2.4 Average Time Spent / Job Cycle

By combining the information from Figure 5.3 and Figure 5.4, the information in Figure 5.5 is obtained. Figure 5.5 shows the average time spent on robots to complete a job cycle. The results for both 3 configurations are very close at first, but the gap becomes huge as the number of robots increase.

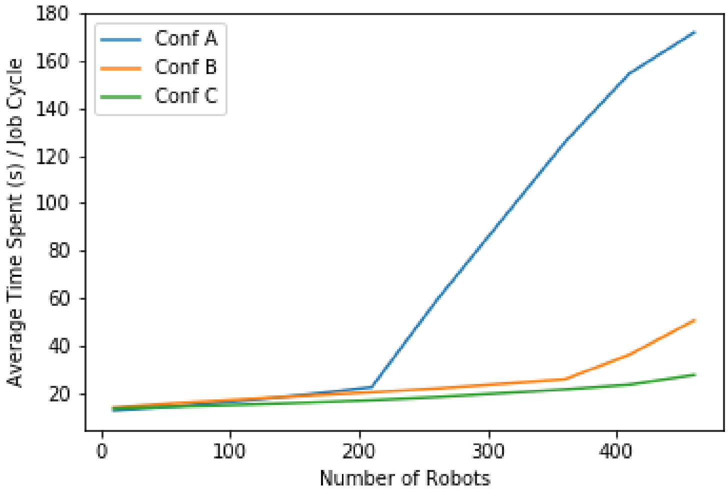


Figure 5.5 Average Time Spent

Configuration A performs the best at first, but the Configuration C surpass Configuration A when the number of robots around 60. Apart from that, it is worth to mention that the performance of Configuration B is always between Configuration C and Configuration A. That is, when the number of robots is large, it is more stable compared to Configuration A but worse than Configuration C. Also, when the number is a robot is small, it performs better than Configuration C but worse than Configuration A . It sounds not that bad, but this situation reveals that there is not any reason to adopt Configuration B because there always has a better selection in different cases.

5.3 Results Analysis – Part 2

This part of the experiment study how the maximum allowable reservation grids for each robot affect the performance of the proposed framework. Figure 5.6 shows that when the maximum allowable reservation grids for each robot is 2, the overall throughput per hour is worst. This is because the speed of the robot is restricted to make sure it able to stop in a short time to avoid collision with other robots. In this situation, the robots have not idea about others robot's plan and thus robot always prepare to solve the local conflict it might face. Thus, this is similar to a simple distributed approach.

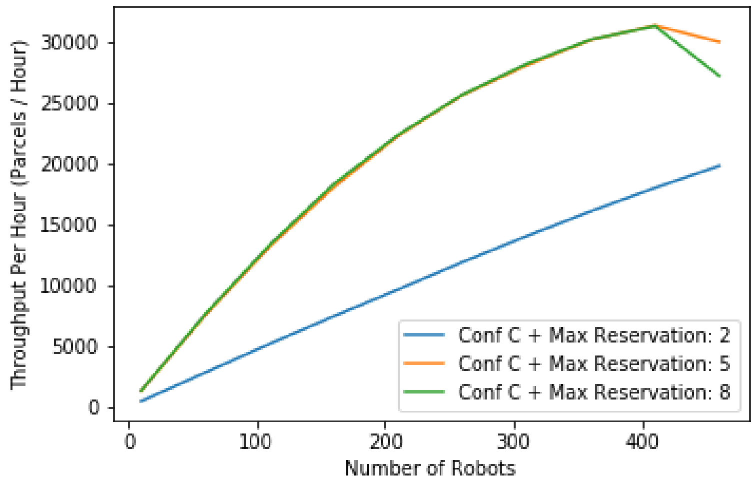


Figure 5.6 Throughput Per Hour - Part 2

On the other hand, when the maximum reservation number are 5 and 8, their performance looks similar. But in fact, the value of 8 is slightly better than 5 before the number of robots reaches 410. Figure 5.7 shows the exact value of the average throughput per hour for the different values of maximum allowable grids reservation for each robot. However, once the number of robots reached around 410, the value of

5 surpasses the value of 8. This situation is because the proposed framework allows the robots to switch their path if the robots fail to reserve a grid after a certain amount of time. The waiting time in this project is fixed to 3200 milliseconds. Also, the location of a robot plays no role in here, that is, after waiting for a grid for 3200 milliseconds, the robots will give up requesting the grid even the robots is still far from the grid.

The benefit of this mechanism is the robots has a higher chance to leave the congestion area and therefore, the moving speed of the robots can be higher. When the value of allowable reservation grids for each robot is higher, the easier for a robot to notice that a region might in congestion. Thus, the robots can switch path earlier. However, the cost of this is the new path almost always having a longer distance than the original path. This situation can be observed in Figure 5.8 which is shows the average distance for robots to complete a job cycle when the allowable reservation grids is different. Also, when the robots are allowed to reserve more grids, the easier for the congestion to happen.

	10	60	110	160	210	260	310	360	410	460
Conf C + Max Reservation 2	475.0	2834.0	5186.0	7458.5	9656.5	11902.5	14029.5	16074.5	18003.0	19799.0
Conf C + Max Reservation 5	1324.5	7546.0	13160.5	18054.5	22267.5	25611.0	28067.5	30157.5	31351.0	30004.0
Conf C + Max Reservation 8	1347.5	7648.5	13319.5	18320.5	22360.5	25659.5	28195.0	30190.5	31309.0	27209.5

Figure 5.7 Exact Values for Throughput Per Hour - Part 2

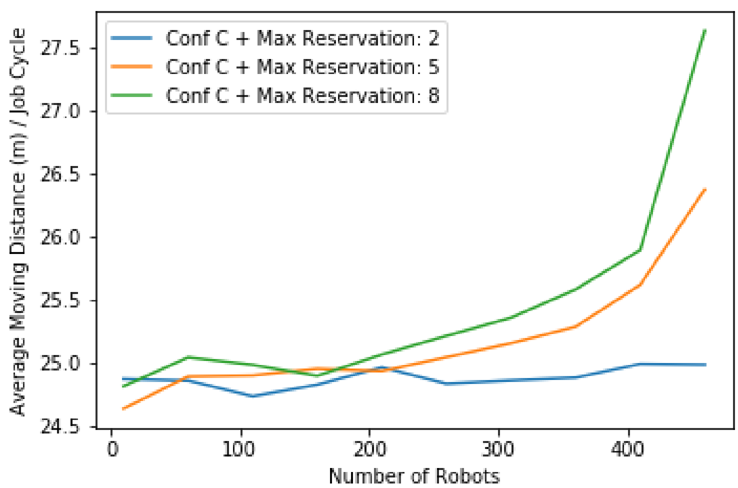


Figure 5.8 Average Moving Distance - Part 2

For the first point, the effect of additional distance can be cancelled if the moving speed of the robots increases, and this is why the performance of value 8 is slightly better than the 5 before the number of robots reaches 410. This Figure 5.9 and

Figure 5.10 shows this phenomenon. Figure 5.9 shows the average speed of the robots while Figure 5.10 shows the average time spent to complete a job cycle by a robot. Please note that for Figure 5.9, the green line is on the top of the orange line until the number of robots reached around 410, and thus the higher speed cancels the effect of the additional distance in this situation.

For the second point, it is a significant issue because this might affect what is the best performance can be achieved in the proposed framework. This is because congestion always is the main barrier that stops the sorting centre to perform better by increasing the number of robots. This situation can also be observed from Figure 5.5 when the number of robots is 460, the performance for 8 allowable reservation grids is dropped more sharply than 5 allowable reservation grids.

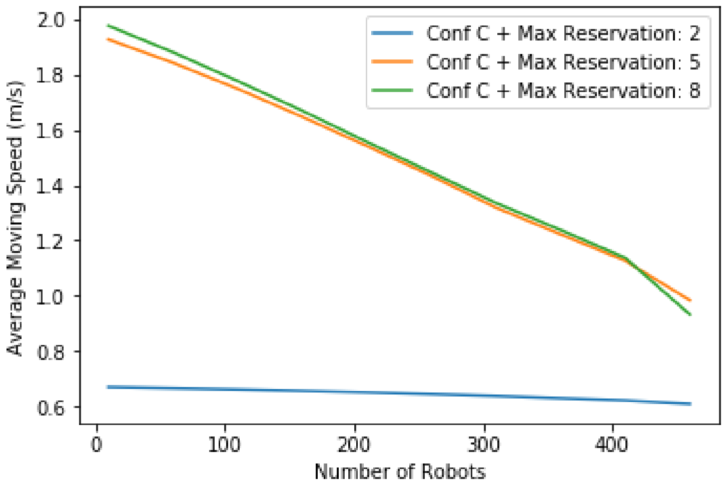


Figure 5.9 Average Moving Speed - Part 2

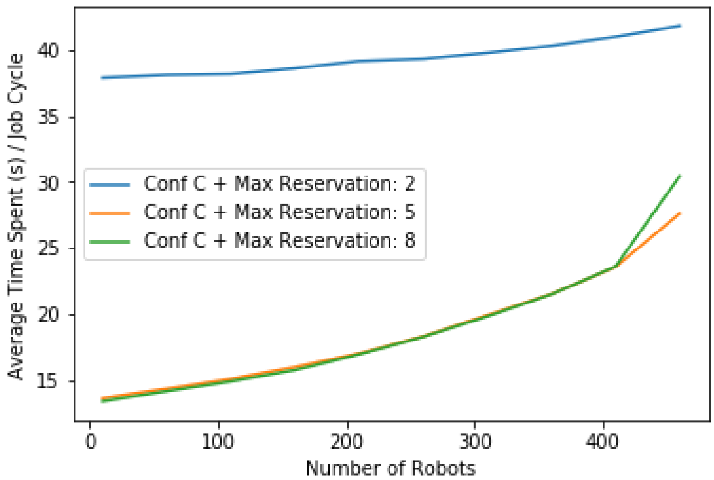


Figure 5.10 Average Time Spent - Part 2

5.4 Results Analysis – Part 3

The experiment in this part is running for 10 hours of simulation time and two different settings are simulated. The first setting is assuming the robots have an infinite amount of battery while another setup is not. Thus, for the second setup, the robots need to charge their battery before they run out of battery. In this situation, the robots need to move to charge stations to charge their battery. Since all the battery stations are placed at the edge of the environments, the charging robots reduce the condition of congestion. Figure 5.12 shows a snapshot of the simulation environment that implements the second setup, some of the robots is charging at the charge stations.

As a result, Figure 5.11 shows that when the robots are assumed to have an infinite battery, the performance drop when the number of robots reaches 460. However, in a normal situation, the throughput per hour still increasing even the number of robots reach 460. This is because the maximum throughput that can be achieved is probably depending on the congestion condition.

The maximum throughput per hour for the first situation reach 31402 at its peak while for the second situation is 31339. The two values are similar, and it is likely near to the maximum performance in the current framework setting. Also, this part of the experiment also shows the proposed framework, as well as the setting of the framework, is stable in the long run.

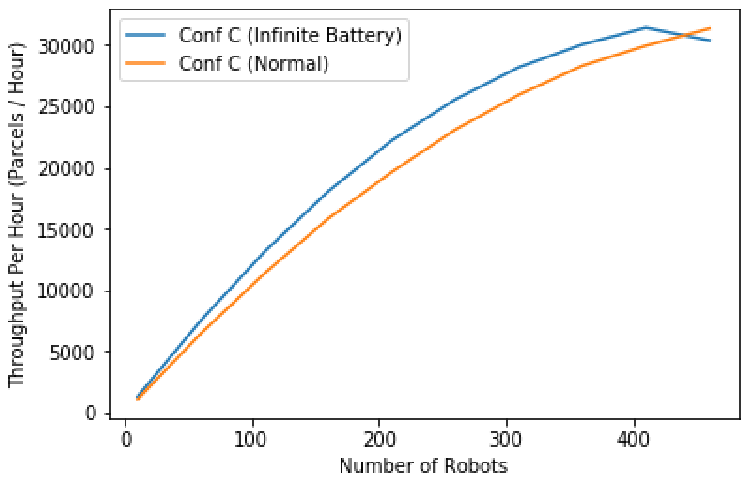


Figure 5.11 Throughput Per Hour - Part 3

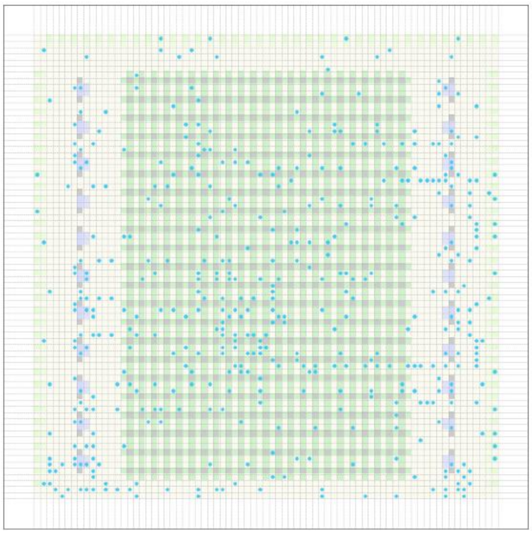


Figure 5.12 Snapshot of Simulation Environment for Second Setup

5.5 Discussion on Congestion Condition & Cost Configuration

Even the congestion condition is mentioned frequently but the reason why the congestion occurs and what is the potential solutions is still not yet discussed. Undeniable, the first and the main reason for the congestion to happen is because the number of robots is too large, but space is limited. But, Figures 5.12 shows that there still has much empty space. This is because it doesn't show that reserved grids, thus Figures 5.13 shows the reserved grids by using pink points.

Now, the empty space decrease obviously. In this case, the conflict can always happen if the robots just move in random directions they like. Although most of the path is set to unidirectional the cross-conflict is still unavoidable. Thus, it is better to force the robots to move in the same direction so that the cross-conflict can be avoided. Clearly, this is impractical, but it is still possible to encourage the robots to move in the same directions. For example, encourage the robots to move vertically by using the outer ring but this might introduce additional distance and cause the drop of performance. Also, even the outer ring can face a congestion condition if the number of robots increases. On the other hand, the configuration C also encourage robots to utilize the centre of the centre region, this is distributing the robots to a different area. In fact, that has other reason for doing this and this was discussed in Chapter 4.

By using those tricks, the performance of Configuration C is outperforming other configuration, but it is still difficult to have a clear understanding of why the thing works. Also, it is not that easy to select a good value of cost to control the robots' path planning behaviour. The information that wants to introduce in here is there might have a better way to configure the cost and the best configuration that figured out in this report might still have a room of improvement. For example, when the number of robots is large, the congestion is most probably happening in the centre of the inner space. Thus, if the robots can be distributed more efficiently, the performance of the sorting centre might be further boosting up. Unfortunately, how to do this efficiently is still not clear at the moment. If the solution for this is figured out, then it might not only able to apply at the static cost but also at the dynamic cost based on different situations to control the robots' behaviour. Thus, the congestion can be resolved more efficiently.

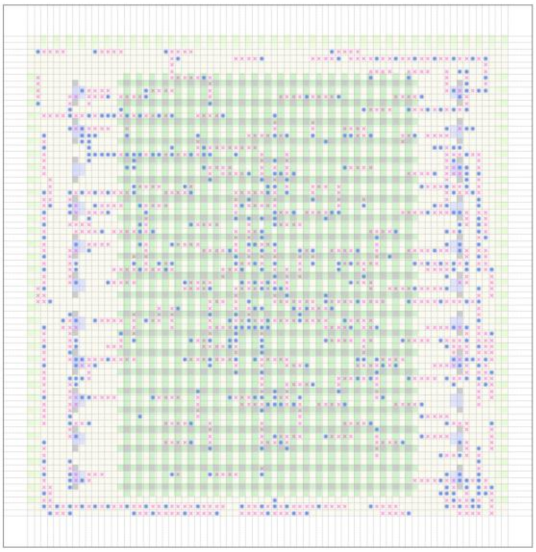


Figure 5.13 Snapshot of Simulation Environment – Show Reserved Grids

5.6 Summary

The experiment shows that it is important to control the robots' path planning behaviour to achieve better results and this can be achieved by control the moving cost for the interconnected grids. Next, reduce the chance of conflict to happen such as head-to-head collision and cross-collision can improve the performance efficiently. Besides, when the number of robots is small, the non-restriction map setup can perform better but when the number of robots is huge, Configuration C might be a good choice. Apart from that, most of the time, congestion is the bottleneck that stopping us to boost the average throughput per hour by increasing the number of robots. Thus, it is important to avoid congestion and one of the methods is to utilize as much space as possible. For example, Configuration C encourages the robots to utilize the outer ring and thus utilize more space compared to other configuration.

Besides, the experiment also shows the number of allowable reservation grids for each robot can also affect the average throughput of the sorting centre. If the number of robots is small, the number of allowable reservation grids for each robot can be set to higher to achieve better performance but the increase in performance is limited. On the contrary, if the number of robots is large, a smaller number of allowable reservation grids can be set to avoid congestion. But, if the value is set too small, the speed of robots reduces because they need to make sure it can stop in a short time to avoid collisions. Lastly, the proposed framework and the best framework setting and configuration able to achieve 31339 throughputs per hour when the number of robots is 460 and this probably near to the best results that can be obtained by current setup.

Chapter 6 Conclusion

6.1 Conclusion

This project proposed an efficient multi-mobile robot control framework for parcel sorting in the sorting centre. The situation of the step of automation in the logistics world is introduced in Chapter 1. Next, Chapter 2 includes literature reviews about the topics or sub-topics related to this project. For example, the type multi-robot systems and the simulation design approaches were studied. After that, the whole proposed framework was designed and discussed in Chapter 3. The proposed framework is a hybrid multi-mobile robot control system that aims to provide flexibility, extendibility and simplicity. In Chapter 3, the environment of the multi-mobile sorting centre was also been modelling to develop a simulator to verify the performance of the proposed framework. The simulator built is discrete-event simulator which using time-advance mechanism. Besides, Chapter 3 also proposed that the robot's path planning behaviour can be controlled by modifying the moving cost for one grid to another. Thus, a layout and three types of moving cost setup were designed. In Chapter 4, some design considerations were discussed to study what is the factor that affects the design of the proposed framework as well as the design of the cost configuration. In Chapter 5, an experiment that consists of 3 parts was designed and conducted to study the performance of the proposed framework and understanding what is the factors that affect the sorting centre performance. Based on the result of Chapter 5, the proposed framework and setup able to sort 31339 parcels per hour in a sorting centre with area $2025m^2$ and the number of robots is 460. Chapter 5 also discussed the barrier if the sorting centre performance which is the congestion condition. The potential solution and direction were also proposed in the Chapter.

6.2 Future Work

In this project, a multi-mobile robot control framework for parcel sorting in the sorting centre is built. But this is far from the end of the story. The stage was built, the future is belonging to the actors. It's easy to note that most of the algorithms and strategies in this project are relatively simple. For example, the path planning algorithm is a traditional A search algorithm which it never utilizes global information. Also, the pickup point selection strategy is greedy to the shortest distance which is simple but might not the best choice in most of the time. Besides, although the by modifying the move cost, the robots path planning behaviour can be affected so that the path planning behaviour is more suitable for the multi-robot environment but this only applicable for static cost in this project.

In short, the current algorithms and strategies are based on local information. This is ignoring most of the global information that might help in doing a better decision. Please noted that it is possible for the proposed framework to assign the job of global information processing to the server, and let the robots make their decision based on the processed information, but this type of algorithms is still not be developed in this project.

On the other hand, this project proposed that path planning behaviour can be controlled by modifying the cost, but how to design the cost is still mainly empirical. This is providing a few possible future directions, for example, the robots might learn the cost by using learning algorithms, or this problem can be converted to a search problem and thus solve the problem by using stochastic algorithms. Apart from that, since the cost can affect the robots' path planning behaviour, this technique might able be used to resolve the congestion by modifying the cost based on the particular situation. That is, a dynamic cost might be used in the future.

Apart from that, the layout of the sorting centre might also be improved. In this project, only one layout is investigated. But in fact, the layout might not the best, especially in a different situation, for example, the target of a sorting centre is to sort the parcel into 200 destinations instead of 500 or more. In this situation, multiple drop points can be used for a single destination and thus reduce the moving distance for the robots to sort the parcels if the location of the drop-off points is well-designed. Besides, the mapping between the drop off-point location and the destinations where the parcels need to sort to can be improved. This is because of some destination might be used more frequently compared to others.

REFERENCES

Abdelgawad, A. (2014). *Localization System for Indoor Robot using RFID.*

Andersson, T., & Jang, H. (2018). *Goods to Person (G2P) Robotics.* Styleintelligence.

Army of robots sort out 200,000 packages a day (2017). [Motion Picture]. Retrieved 3 23, 2019, from https://www.youtube.com/watch?v=XoQI-kLRrXA

Barer, M., Sharon, G., Stern, R., & Felner, A. (2014). Suboptimal Variants of the Conflict-Based Search Algorithm for the Multi-Agent Pathfinding Problem. *Proceedings of the Seventh Annual Symposium on Combinatorial Search.*

Bonkenburg, T. (2016). *Robotics in Logistics.* DHL Trend Research.

Cannon Hill Logistics. (2017, June 9). *What's the Difference Between Warehouses and Distribution Centers?* Retrieved from http://cannonhill.net/whats-difference-warehouses-distribution-centers/

Cap, M. (2016). Centralized and Decentralized Algorithms for Multi-Robot Trajectory Coordination.

Constantiono, N., Dotolim, Fanti, M., & Mangini, A. (2012). A model for supply management of agile manufacturing supply chains. *International Journal of Production Economics*, 451-457.

E.Hart, P., & J.Nilsson, N. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions of Systems Science and Cybernetics*.

Frazelle, E. (2002). World-Class Warehousing and Material Handling. *McGraw-Hill Companies, Inc.*

General, O. o. (2018). *Autonomous Mobile Robots and the Postal Service.* RARC Report.

Guizzo, E. (2008). Kiva Systems: Three Engineers, Hundreds of Robots, One Warehouse. *IEEE Spectrum*. Retrieved 3 21, 2019, from https://spectrum.ieee.org/robotics/robotics-software/three-engineers-hundreds-of-robots-one-warehouse

Husain, S., Kennedy, S., & Krebs, D. (2016). *Robotics in Logistics.* VDC Research.

Law, A. M. (2007). *Simulation Modeling Analysis Fourth Edition.*

LiBiao, R. (2019, 2 19). *浙江立标机器人*. Retrieved from http://www.libiaorobot.com/

Liu, Y., Wang, L., Huang, H., Liu, M., & Xu, C.-z. (2017). A Novel Swarm Robot Simulation Platform for Warehousing Logistics. *IEEE International Conference on Robotics and Biomimetics*.

Ma, H., Gong, Y., Song, J., Wu, X., & Cui, Y. (2015). A Task-Grouped Approach for the Multi-Robot Allocation of Warehouse System. *International Conference on Computer Science and Mechanical Automation.*

Mauro, F. (2017). *Towards the Design of An Effective and Robust Multi-Robot Parcel Sorting System.*

Pinkam, N., Bonnet, F., & Chong, N. Y. (2016). Robot Collaboration in Warehouse. *International Conference on Control, Automation and Systems.*

Sharon, G., Stern, R., Felner, A., & R.Sturtevant, N. (2014). Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence*, 40-66.

Singhal, A., Kejriwal, N., Pallav, P., Choudhury, S., Sinha, R., & Kumar, S. (2017). Managing a Fleet of Autonomous Mobile Robots (AMR) using Cloud Robotics Platform.

Stanek, O. (2012). Centralized Multirobot System.

Trigui, S., Koubaa, A., Cheikhrouhou, O., Youssef, H., Bennaceur, H., Sriti, M.-F., & Javed, Y. (2014). A Distributed Market-Based Algorithm for the Multi-Robot

Assignment Problem. *3rd Internaational Workshop on Cooperative Robots and Sensor Networks*.

Tsang, K. F., Che, S. H., Zhang, C., & Song, J. (2017). Warehouse Multi-Robot Automation System.

Wang, S., & Chen, W. (2002). Queue Coordination Strategy Based on Local Sensing in Distributed Multiple Mobile Robot Systems.

Wessman, I., & Bärring, M. (2014). Analysing the Current State of a Warehouse.

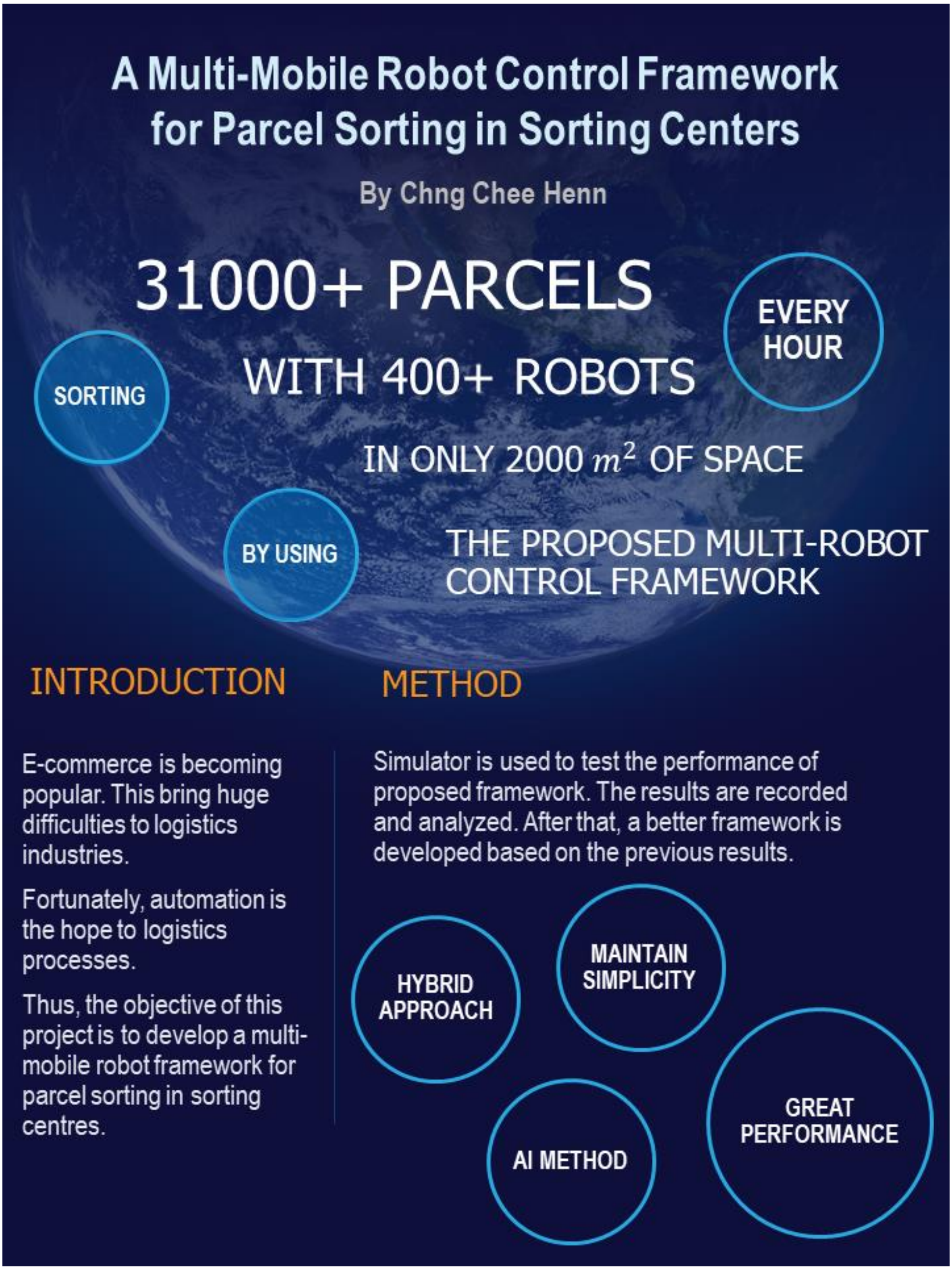
Why Gazebo? (2018). Retrieved August 14, 2018, from http://gazebosim.org/

Wurman, P. R., D'Andrea, R., & Mountz, M. (2008). Coordinating Hundreds of Cooperative, Autonomous Vehicles in Warehouses. *AI Magazine Volume 29 Number I*.

Yan, Z., Jouandeau, N., & Cherif, A. A. (2013). A Survey and Analysis of Multi-Robot Coordination.

Yu, J. (2016). Intractability of Optimal Multirobot Path Planning on Planar Graphs.

Zhang, Z., Guo, Q., Chen, J., & Yuan, P. (2018). Collision-Free Route Planning for Multiple AGVs in an Automated Warehouse Based on Collision Classification. *Special Section on Human-Centered Smart Systems and Technologies*.


A Multi-Mobile Robot Control Framework for Parcel Sorting in Sorting Centers
By Chng Chee Henn
31000+ PARCELS
EVERY HOUR
SORTING
WITH 400+ ROBOTS
IN ONLY 2000 m^2 OF SPACE
BY USING
THE PROPOSED MULTI-ROBOT CONTROL FRAMEWORK
INTRODUCTION
METHOD
E-commerce is becoming popular. This bring huge difficulties to logistics industries.
Fortunately, automation is the hope to logistics processes.
Thus, the objective of this project is to develop a multi-mobile robot framework for parcel sorting in sorting centres.
Simulator is used to test the performance of proposed framework. The results are recorded and analyzed. After that, a better framework is developed based on the previous results.
HYBRID APPROACH
MAINTAIN SIMPLICITY
AI METHOD
GREAT PERFORMANCE

A Multi-Mobile-Robot Control Framework for Parcel Sorting in Sorting Centres

ORIGINALITY REPORT

4%	1%	2%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Zheng Zhang, Qing Guo, Juan Chen, Peijiang Yuan. "Collision-Free Route Planning for Multiple AGVs in Automated Warehouse Based on Collision Classification", IEEE Access, 2018 Publication	<1%
2	Guni Sharon, Roni Stern, Ariel Felner, Nathan R. Sturtevant. "Conflict-based search for optimal multi-agent pathfinding", Artificial Intelligence, 2015 Publication	<1%
3	Submitted to Universiti Teknologi MARA Student Paper	<1%
4	Submitted to University of Pretoria Student Paper	<1%
5	Dong W. Kim, Ty A. Lasky, Steven A. Velinsky. "Autonomous multi-mobile robot system: simulation and implementation using fuzzy logic", International Journal of Control, Automation and Systems, 2013	<1%

Publication

6	Submitted to University of Warwick Student Paper	<1%
7	www.it.s-t.au.ac.th Internet Source	<1%
8	Submitted to Embry Riddle Aeronautical University Student Paper	<1%
9	Submitted to University of Glasgow Student Paper	<1%
10	Submitted to University of Newcastle upon Tyne Student Paper	<1%
11	repository.unika.ac.id Internet Source	<1%
12	Bo Sun, Jingwei Li, Guanci Yang. "Multi-robot Cooperation Strategy Based on Wireless Sensor Network", International Journal of Online Engineering (iJOE), 2018 Publication	<1%
13	Michael Filzmoser. "Simulation", Simulation of Automated Negotiation, 2010 Publication	<1%
14	Submitted to University of Technology Student Paper	<1%

Jayasree Chakraborty, Amit Konar, Uday K.

15	Chakraborty, L.C. Jain. "Distributed cooperative multi-robot path planning using differential evolution", 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), 2008 Publication	<1%
16	Submitted to Cranfield University Student Paper	<1%
17	www.elec.qmul.ac.uk Internet Source	<1%
18	Submitted to Hellenic Open University Student Paper	<1%
19	Submitted to Monash University Student Paper	<1%
20	Submitted to Universiti Teknologi Malaysia Student Paper	<1%
21	Submitted to University of Technology, Sydney Student Paper	<1%
22	Jau Jen, Mon Shie, Charlie Chen. "A Circular Hough Transform Hardware for Industrial Circle Detection Applications", 2006 1ST IEEE Conference on Industrial Electronics and Applications, 2006 Publication	<1%
23	Submitted to October University for Modern	<1%

Sciences and Arts (MSA)
Student Paper

24	"MEMS/NEMS", Springer Nature, 2006 Publication	<1%
25	Naai-Jung Shih, Ching-Yuan Lin, Chih-Hsiang Yang. "A virtual-reality-based feasibility study of evacuation time compared to the traditional calculation method", Fire Safety Journal, 2000 Publication	<1%
26	hdl.handle.net Internet Source	<1%
27	Zeyad Abd Algfoor, Mohd Shahrizal Sunar, Afnizanfaizal Abdullah. "A new weighted pathfinding algorithms to reduce the search time on grid maps", Expert Systems with Applications, 2017 Publication	<1%
28	www.scribd.com Internet Source	<1%
29	Hoon Sung Chwa, Andrii Shyshkalov, Jinkyu Lee, Kilho Lee, Hyoungbu Back, Insik Shin. "Adaptive trajectory coordination for scalable multiple robot control", 2011 IEEE International Conference on Service-Oriented Computing and Applications (SOCA), 2011 Publication	<1%

30	Submitted to The University of the South Pacific Student Paper	<1%
31	Lien-Wu Chen, Jun-Xian Liu. "Time-Efficient Indoor Navigation and Evacuation With Fastest Path Planning Based on Internet of Things Technologies", IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2019 Publication	<1%
32	Submitted to City University of Hong Kong Student Paper	<1%
33	Guo Shuai, Dai Leying, Ouyang YuPing. "The Research and Simulation on the Path Planning Based on the Improved Grid Model", 2009 International Conference on Artificial Intelligence and Computational Intelligence, 2009 Publication	<1%
34	Submitted to Heriot-Watt University Student Paper	<1%
35	Submitted to Sheffield Hallam University Student Paper	<1%
36	Submitted to Coventry University Student Paper	<1%
37	eprints.utm.my Internet Source	<1%
38	Submitted to University of Edinburgh	

Student Paper <1%

39 Bungartz, Hans-Joachim, and Michael Griebel. "Sparse grids", Acta Numerica, 2004. <1%
Publication

40 Heng Wei, Qiang Lv, Nanxun Duo, GuoSheng Wang, Bing Liang. "Consensus Algorithms Based Multi-Robot Formation Control under Noise and Time Delay Conditions", Applied Sciences, 2019 <1%
Publication

Exclude quotes	Off	Exclude matches	< 8 words
Exclude bibliography	On		

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	Ch'ng Chee Henn
ID Number(s)	15ACB03595
Programme / Course	CS
Title of Final Year Project	Developing a Cooking Gas Ordering and Delivery System Mobile Application

Similarity	**Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)**
Overall similarity index:________ % **Similarity by source** Internet Sources: ______________% Publications: ______________ % Student Papers: ______________ %	
Number of individual sources listed of more than 3% similarity: _________	

Parameters of originality required and limits approved by UTAR are as Follows:

(i) Overall similarity index is 20% and below, and

(ii) Matching of individual sources listed must be less than 3% each, and

(iii) Matching texts in continuous block must not exceed 8 words

Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

______________________________ Signature of Supervisor

______________________________ Signature of Co-Supervisor

Name: __________________________

Name: __________________________

Date: ___________________________

Date: ___________________________

UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	
Student Name	
Supervisor Name	

TICK (√)	DOCUMENT ITEMS Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
	Front Cover
	Signed Report Status Declaration Form
	Title Page
	Signed form of the Declaration of Originality
	Acknowledgement
	Abstract
	Table of Contents
	List of Figures (if applicable)
	List of Tables (if applicable)
	List of Symbols (if applicable)
	List of Abbreviations (if applicable)
	Chapters / Content
	Bibliography (or References)
	All references in bibliography are cited in the thesis, especially in the chapter of literature review
	Appendices (if applicable)
	Poster
	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report. ______________________ (Signature of Student) Date:	Supervisor verification. Report with incorrect format can get 5 mark (1 grade) reduction. ______________________ (Signature of Supervisor) Date: