UNIVERSITÄT DUISBURG-ESSEN
FAKULTÄT FÜR WIRTSCHAFTSWISSENSCHAFTEN
INSTITUT FÜR INFORMATIK UND WIRTSCHAFTSINFORMATIK
LEHRSTUHL FÜR PERVASIVE COMPUTING

**Masterarbeit**

# Exploiting Knowledge of Room Occupation for the Scheduling of Navigation Tasks of a Fleet of Robots in Office Environments

Xuanjiao Zhu
Matrikelnummer: 3038674

Networked Embedded Systems Group
Institut für Informatik und Wirtschaftsinformatik
Fakultät für Wirtschaftswissenschaften
Universität Duisburg-Essen

5. Oktober 2020

**Erstprüfer:** Prof. Dr. Pedro José Marrón
**Zweitprüfer:** Prof. Dr. Gregor Schiele
**Zeitraum:** 1.May 2020 - 1.October 2020

# Contents

## Abstract

The function of the abstract is to summarize, in one or two paragraphs, the major aspects of the entire bachelor or master thesis. It is usually written after writing most of the chapters.

It should include the following:

- Definition of the problem (the question(s) that you want to answer) and its purpose (Introduction).
- Methods used and experiments designed to solve it. Try to describe it basically, without covering too many details.
- Quantitative results or conclusions. Talk about the final results in a general way and how they can solve the problem (how they answer the question(s)).

Even if the Title can be a reference of the work's meaning, the Abstract should help the reader to understand in a quick view, the full meaning of the work. The abstract length should be around 300 words.

Abstracts are protected under copyright law just as any other form of written speech is protected. However, publishers of scientific articles invariably make abstracts publicly available, even when the article itself is protected by a toll barrier. For example, articles in the biomedical literature are available publicly from MEDLINE which is accessible through PubMed. It is a common misconception that the abstracts in MEDLINE provide sufficient information for medical practitioners, students, scholars and patients[citation needed]. The abstract can convey the main results and conclusions of a scientific article but the full text article must be consulted for details of the methodology, the full experimental results, and a critical discussion of the interpretations and conclusions. Consulting the abstract alone is inadequate for scholarship and may lead to inappropriate medical decisions[2].

An abstract[IGM97, **?**, MAdR02, Sal89] allows one to sift through copious amounts of papers for ones in which the researcher can have more confidence that they will be relevant to his research. Once papers are chosen based on the abstract, they must be read carefully to be evaluated for relevance. It is commonly surmised that one must not base reference citations on the abstract alone, but the entire merits of a paper.

# Chapter 1

# Introduction

[You should answer the question: What is the problem?]

This paragraph should establish the context of the reported work. To do that, authors discuss over related literature (with citations[1]) and summarize the knowledge of the author in the investigated problem.

ToDo: how to make citations

An introduction should answer (most of) the following questions:

- What is the problem that I want to solve?
- Why is it a relevant question?
- What is known before the study?
- How can the study improve the current solutions?

To write it, use if possible active voice:

- We are going to watch a film tonight (Active voice).
- A film is going to be watched by us tonight (Passive voice).

The use of the first person is accepted.

## 1.1 Motivation

A good introduction usually starts presenting a general view of the topic and continues focusing on the problem studied. Begin it clarifying the subject area of interest and establishing the context (remember to support it with related bibliography).

[1] To cite a work in latex

## 1.2 Problem definition

Additionally, focuses the text on the relevant points of your investigation and problems that you want to solve, relating them with the first part.

## 1.3 Thesis/Diplom/Bachelor/Master Structure

Present your work to the reader giving a brief overview of what is going to cover every chapter. Write only general concepts, no more than one or two sentences per chapter should be necessary.

# Chapter 2

# Materials and Methods

This section is to clarify the pre-existing tools, defining what was developed in this field until now, and why this tool was used instead of others.

The general structure is the following:

- Definition of the specific tool(s) studied (robots, sensor nodes, smart-phones). When relevant, pre-existing experiments.
- Definition of the context of use (indoor/outdoor, humans/animals/robots, with/without connection).
- Definition of used protocols (How the data are collected, when, etc.)

# Chapter 3

# Approach

In our system there are multiple robots that must handle various tasks. For example, visiting given rooms. To tackle this problem, a communication efficient task scheduling system is designed. This system allocate task according to system resources, including environment factors, robot status and task specifications. Once this information is attained, the task scheduling system assign robot a set of task.

- *Robot.* Each robot is responsible for moving in 2-dimensional physical space as well as gathering measurement result from sensors. It has a rechargeable battery, and its level drops as robot moves and rotates.
- *Tasks.* Each task requires one or more robots to traverse a path in the workspace and carry out certain actions[IGS17].
- *Environment.* In this project, all robots are considered moving in an office environment that contains a corridor along the central x-axis and 16 rooms located around the corridor. The environment factors, such as room locations and occupancy possibilities help task allocation.

## 3.1 Architecture Design

The architecture of the system consist of several parts: centralized pool, robot controller, navigation stack, charging station and system environment(Figure 3.1).

- *Centralized Pool.* A centralized pool consist of several modules: multi-robot task allocation module, map information, database, execution and monitoring. The database contains most of the environment information(Figure 4.2). The multi-robot task allocation module allocate tasks to robots once requested.
- *Robot Controller.* A robot controller contains several modules: local task queue, execution and robot action. The execution module receives commands from centralized pool and decides when and which task the robot should perform.

- *Navigation stack.* The move_base node provides a ROS interface for configuring, running, and interacting with the navigation stack on a robot. It makes robot move to desired positions using the navigation stack. Its advantages include optionally performing recovery behaviors when the robot perceives itself as stuck[mov].

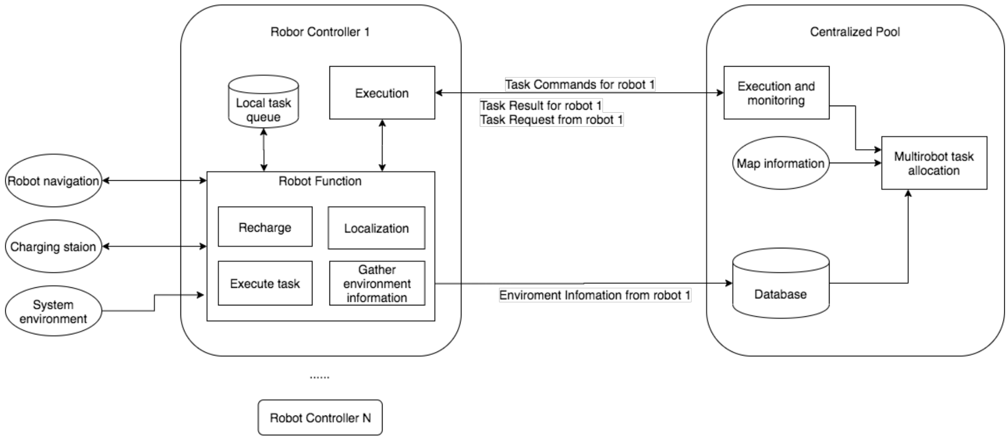


Figure 3.1: System Architecture

## 3.2 System Environment

As is discussed in the previous section, the system environment is an office environment. The SLAM (Simultaneous Localization and Mapping) is a technique to draw a map by estimating current location in an arbitrary space [sla]. Map (Figure 3.2) is created by SLAM.

There are following important areas and coordinates:

- *Rooms.* The rectangle areas (Figure 3.2) are used to represent Rooms. Each rectangle has its upper and lower limit in x and y coordinates. The system uses those limits to determine a position in which room.

- *Doors.* The positions of doors (Figure 3.3) are stored in database. There are used by a ROS door simulator node, which broadcasts positions and door status periodically. The broadcast messages are received and filtered by robots.

- *Charging Stations.* The positions of charging stations are used by ROS charging station nodes. For details please refer to Section 4.3.3.

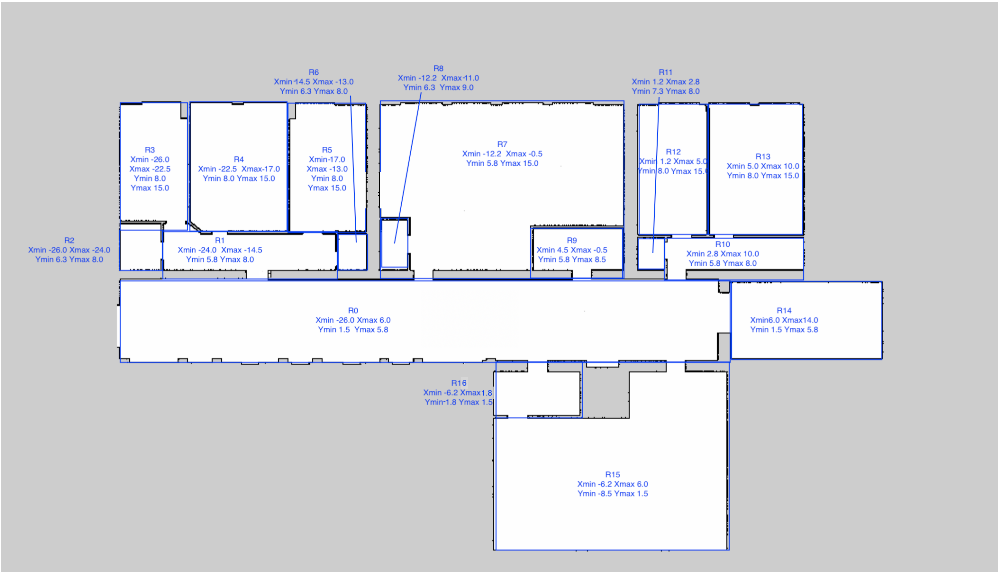


Figure 3.2: Room division

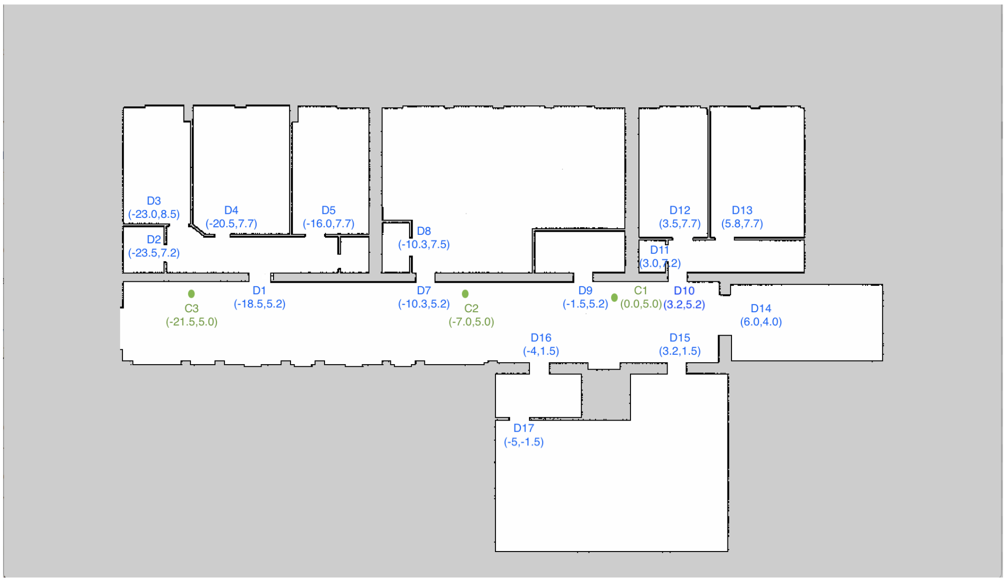


Figure 3.3: Doors and Charging Stations

## 3.3 Task Allocation

### 3.3.1 Task Specifications

On one hand, recharging is necessary for robots to work long hours. On the other hand, a robot should gather environment information as much as possible, which centralized pool would learn from and make better decision. Therefore, three types of task are defined. Table (Table 3.1 list task specifications. In addition, each task has a “start time” when the robot receives the task and a “finish time”, which robot the sends result) A task can be classified into small task and large task. One robot can carry either small task, which make robot move to one position, or large task, which consist of multiple small task.

- Small Task. A small task contains only one goal. When a robot receives a small task, it moves to the only goal.
- Large Task. A large task contains several small tasks, and each small tasks contains a goal. Those small tasks form a dependency chain. When a robot gets a large task, it moves continuously to those goals. The centralized pool can query “small execute tasks” and use them to form a “large execute tasks”.

| Task Specifications / Task type | Target | Dependency | Priority | Generator | Task Size |
|---|---|---|---|---|---|
| Gather environment infomation task | Door | No | 1 | Centralized pool | Small task |
| Execute task | Any point | Dependent on “execute task” | 2,3,4 | User | Small task (in database) or large task (in centralized pool) |
| Charging Task | Charging station | No | 5 | Centralized Pool | Small task |

Table 3.1: Task Specifications

| Task ID | Task Type | Start Time | Target ID | Robot ID | Priority | Status | dependency | Finish Time | Description |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Charging task | 2020-06-01 9:00:00 | 18 | 1 | 5 | RanToCompletion | 0 | 2020-06-01 9:00:20 | Succeeded |
| 2 | Execute task | 2020-06-01 9:00:50 | 22 | 2 | 2 | RanToCompletion | 0 | 2020-06-01 9:01:20 | Succeeded |
| 3 | Gather environment information task | 2020-06-01 9:02:00 | 2 | 2 | 1 | Running | 0 | 2020-06-01 9:02:40 | Succeeded |

Table 3.2: Task Table in Database

- *Target.* Targets include doors, points and charging stations. When a robot run a “gather environment information task”, it moves to the front of a door and interact with a sensor in the door position without entering the door. When robot run an “execute task”, the robot moves to a given point ether in corridor or in the room. When robot run a “charging task”, the robot moves to a charging station and interact with this charging station.

### 3.3.2 Execute Task Allocation

With robot status such as positions and available battery provided by robot, the multi-robot task allocation module in the architecture should perform multi-robot task allocation. When the centralized pool receives a task request (Table 4.2) from robot, the multi-robot task allocation module in the architecture. The implementation of task allocation is shown in Figure 4.3.

1. When the battery of robot belows 10%, the centralized pool will create a charging task to the charging staion with the lowest cost.
2. When the battery of robot aboves 10%, the centralized pool will load all "execute tasks" in database and combine small tasks with dependencies into large tasks(Section 3.3.1), finally calculate task costs and select one large task with the lowest cost.
3. If there are no suitable tasks, a "gather environment task" to the door with the lowest cost is generated.
4. The output of the task allocation includes: task ID, goal coodinates, timestamp and selected robot ID. The task is sent to the selected robot, and the robot performs the tasks.

**Decision variables** To select an "execute task", the following decision variables are considered.

- *Task Priority.* Task Priority. Task priority is an important factor that describes task emergency level. The "charging task" has the highest priority of 5. The "gather environment information" task has the lowest priority of 1. The "execute task" is determined by users but must be in the range of [2,4].
- *Product of Door Open Possibility.* Because of the limitation of simulation, the door open possibilities are used to represent room occupancies. A door open possibility is based on the statistic of door measurement results in a specific time period of a working day. The doors on trajectory (doors that the robot may pass through) can be obtained from the map information module. An example of "measurement result" table is shown in Table 5.8, an example of "open possibility" table is shown in Table 3.3.
- *Waiting Time.* The waiting time is the difference between the current simulation time and start time of the first task to be executed. $T_{waiting} = T_{first\_task} - T_{now}$
- *Battery Consumption.* The Battery Consumption is related to robot trajectory. For a Large "execute task" that contains n small task, Equation 3.1 can be used to calculate battery consumption. The centralized pool will send the task with the lowest cost to this robot.

Equation 3.2 shows how these decition variables are used to calculate the cost of a large "execute task".

ToDo: Database Table

| Door id | Day Of Week | Start Time | End Time | Initialized Open Possibility | Open Possibility Statistic |
|---|---|---|---|---|---|
| 1 | 2 | 10:00:00 | 10:59:59 | 0.80 | 0.80 |
| 1 | 2 | 11:00:00 | 11:59:59 | 0.20 | 0.18 |
| 2 | 2 | 10:00:00 | 10:59:59 | 0.80 | 0.82 |
| ... | ... | ... | ... | ... | ... |

Table 3.3: Door Open Possibility

B:Battery consumption
W: Weight
m: Number of waypoint
n: Number of task

$$
\begin{aligned}
&B_{\text{large task}} = \sum_{\text{task}_1}^{\text{task}_n} B_{\text{trajectory}} \\
&= \sum_{t=\text{task}_1}^{\text{task}_n} \sum_{\text{waypoint}_1}^{\text{waypoint}_m} [W_{\text{position}} \times \text{position variation} + W_{\text{angle}} \times \text{angle variation}] \\
&= \sum_{t=\text{task}_1}^{\text{task}_n} \sum_{p=\text{waypoint}_1}^{\text{waypoint}_m} [W_{\text{position}} \times \sqrt{(x_p - x_{p-1})^2 + (y_p - y_{p-1})^2} \\
&+ W_{\text{angle}} \times 2 \times \arccos(w_p)]
\end{aligned} \tag{3.1}
$$

W: Weight
n: Number of doors

$$
\begin{aligned}
&\text{Cost}_{\text{Large execute task}} = \frac{W_{\text{battery}} \times \text{Battery consumtion}}{n} + W_{\text{waiting}} \times \text{waiting time} \\
&+ W_{\text{possibility}} \times \prod_{i=1}^{n} \text{Door open possibility} + W_{\text{priority}} \times \text{Priority}
\end{aligned} \tag{3.2}
$$

### 3.3.3 Environment Task Allocation

As is discussed in section 3.3.2, once there are no suitable tasks in centralized pool, the task allocation module should create a“gather environment information task”, in order to gather more measurement result and further more improve the accuracy of “open possibilities” table. To create a “gather environment information task”, which door should the requesting robot visit must be considered. The following decision factors help task allocation module to select the door.

#### Decision variables

- *Door Last Update Time.* The latest timestamp when the door is measured.
- *Battery Consumption.* Similar to “execute task“ allocation, the battery consumption is related to the trajectory from robot to the door. Equation 3.1 can be used to calculate battery consumption.
- *Whether door is used.* If another robot is going to this door, the value is 0, otherwise the value is 1.

W: Weight

n: Number of doors on trajectory

$$\text{Cost}_{\text{door}} = \frac{W_{\text{battery}} \times \text{Battery consumtion}}{n} + W_{\text{time}} \times (T_{\text{last update}} - T_{\text{now}}) \tag{3.3}$$

$$+ W_{\text{possibility}} \times \prod_{i=1}^{n} \text{Door open possibility}$$

### 3.3.4 Charging Task Allocation

As is discussed in section 3.3.2, once a robot sends task request to the centralized pool, the centralized pool should figure out whether this robot need charging, if yes it should create a “charging task” for requesting robot. Since there are multiple charging station in the system environment (Figure 3.3), the centralized pool selects a charging station for this robot using the following decision variables.

#### Decision variables

- *Remain Time.* It describes how long will a charging station be free.
- *Battery Consumption.* Similar to “execute task” allocation, the battery consumption is related to the trajectory from robot to the charging station. Equation 3.1 can be used to calculate battery consumption.

In conclusion, equation 3.4 can be used to calculate the cost of a charging station. The centralized pool will generate a "charging task" for requesting robot to charging station with the lowest cost.

$$\text{W: Weight}$$

$$\text{Cost}_{\text{charging station}} = \frac{W_{\text{battery}} \times \text{battery consumtion}}{n} + W_{\text{time}} \times T_{\text{remain}} \tag{3.4}$$

# Chapter 4

# Implementation

## 4.1 Communication Protocols

Centralized pool, robots, charging stations and sensors need to share information with each other. To improve the communication efficiency, communication protocols are designed.

### 4.1.1 Message about Measurement

When a robot passes by a door, it should receive messages from a sensor. In this project, we use a ROS node ”sensor simulator” to simulate door sensors (Section 4.3.4) to publish instant measurement result (Table 4.1).

| Door ID | Position | Timestamp | Measurement Result |
|---|---|---|---|
| 1 | (-18.5,5.2) | 2020-06-01 9:00:02 | Door opened |

Table 4.1: Measurement Message Format and Example

### 4.1.2 Message about Task

There are some basic requirements for communication between robot and centralized pool: firstly, robot should initiate the communication once it has finished all task in task queue and get free. Secondly, robot should forward sensor data to centralized pool while processing a task. These Communication protocols save unnecessary communication cost by avoiding keep tracking the current position, availability and states of all robots (Figure 4.1). Four types of message are defined: (1)Task request message(Table4.2); (2) Task goal messages(Table 4.3); (3) Task feedback message (Table 4.4); (4) Task result message (Table 4.5).

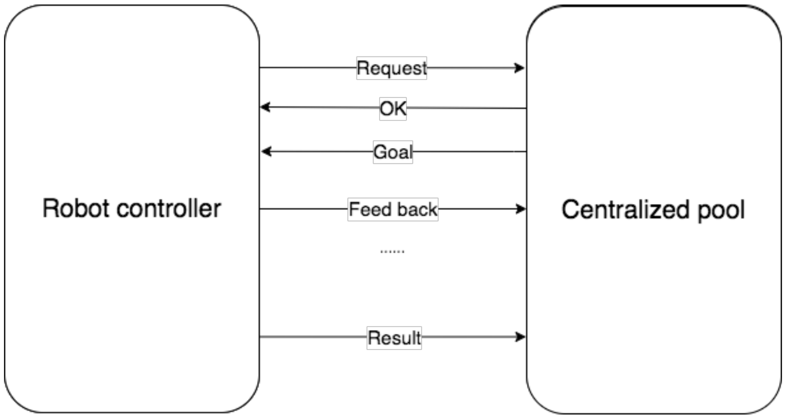


Figure 4.1: Communication between Robot and Centralized Pool

| Battery Level | Position | Robot ID |
|---|---|---|
| 93 | (2,4) | 1 |

Table 4.2: Request Message Format and Example

### 4.1.3 Message about Charging

When a robot arrives charging station's position, it sends a message to Charging station(Figure 4.6). The details of charging station is discussed in Section 4.3.3.

## 4.2 Database

The centralized pool keep environment information in database to make decisions. The structure of database is shown in Figure 4.2.

| Task ID-[] | Task type | Target ID | Goal[] |
|---|---|---|---|
| 1 | Gather Environment Info | 9 | (-1.5,5.2) 2020-06-01 9:00:00 |
| [3,4] | Execute task | 21, 22 | (-24.0,12.0), 2020-06-01 9:02:00 (-21.0,12.0) 2020-06-01 9:02:00 |
| 5 | Charging | 17 | (0.0,5.0), 2020-06-01 9:04:00 |

Table 4.3: Action Goal Message Format and Example

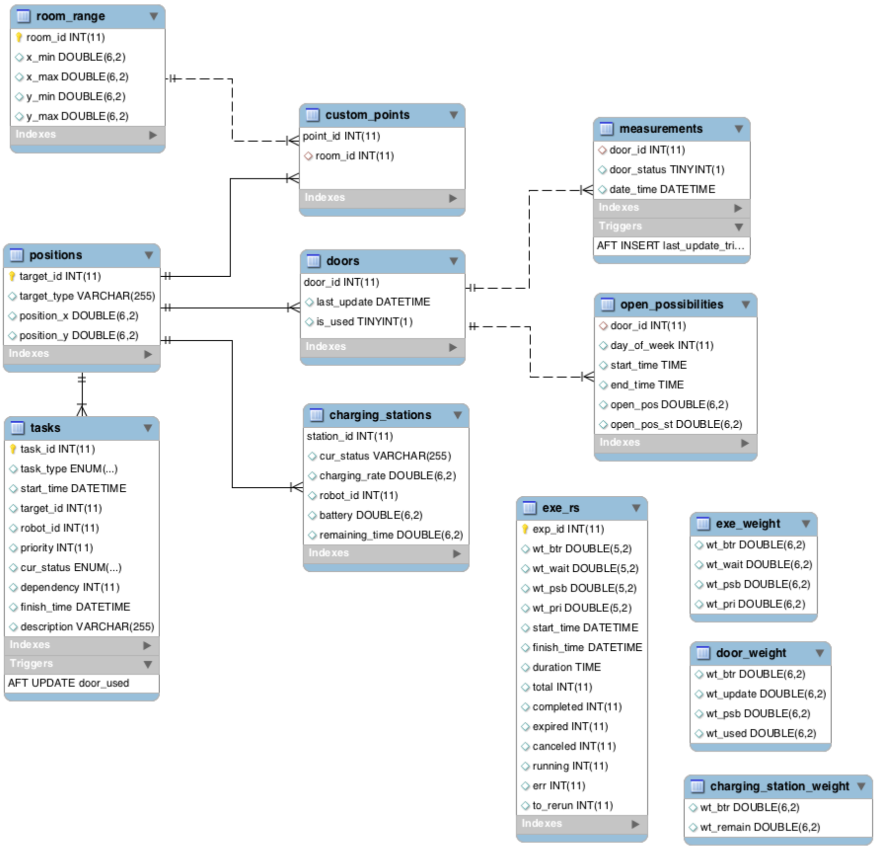


Figure 4.2: Database Entity Relationship Diagram

| Robot ID | Door ID | Measurement time | Measurement result |
|---|---|---|---|
| 1 | 3 | 2020-06-01 9:00:03 | Door open |

Table 4.4: Action Feedback Message Format and Example

| Task ID | Task type | Result |
|---|---|---|
| 1 | Gather Environment Info | Success |

Table 4.5: Action Result Message Format and Example

## 4.3 Procedure

As stated in the Chapter 3, the goal of task scheduling is finishing all tasks as soon as possible while keep the cost as low as possible. The task assignment and execution has at two level. [IGS17] the task and the path planner solves a planning problem. It takes and occupancy grid, a specific robot and a set of task specifications, and generates trajectories for each task. According to those trajectories and task specifications, the task with the lowest cost is assigned to robot. At the dynamic level, after each robot receive a task, it runs a navigation stack to execute this task stepwise. Each robot computes a local trajectory but takes into account dynamic obstacles. The process of the robot task allocation system is as follows.

### 4.3.1 Centralized Pool

**Task Allocation** The task allocation algorithm is discussed in Section 3.3.2. The process of task allocation is shown in Figure 4.3.

**Handle Task Feedback.** When the centralized pool receives a task feedback (Table 4.4) that contains a new measurement result from robot, it will add a record in measurement table and update “open possibilities” table in database.

| Robot ID | Battery Level |
|---|---|
| 1 | 93 |

Table 4.6: Message to Charging Station

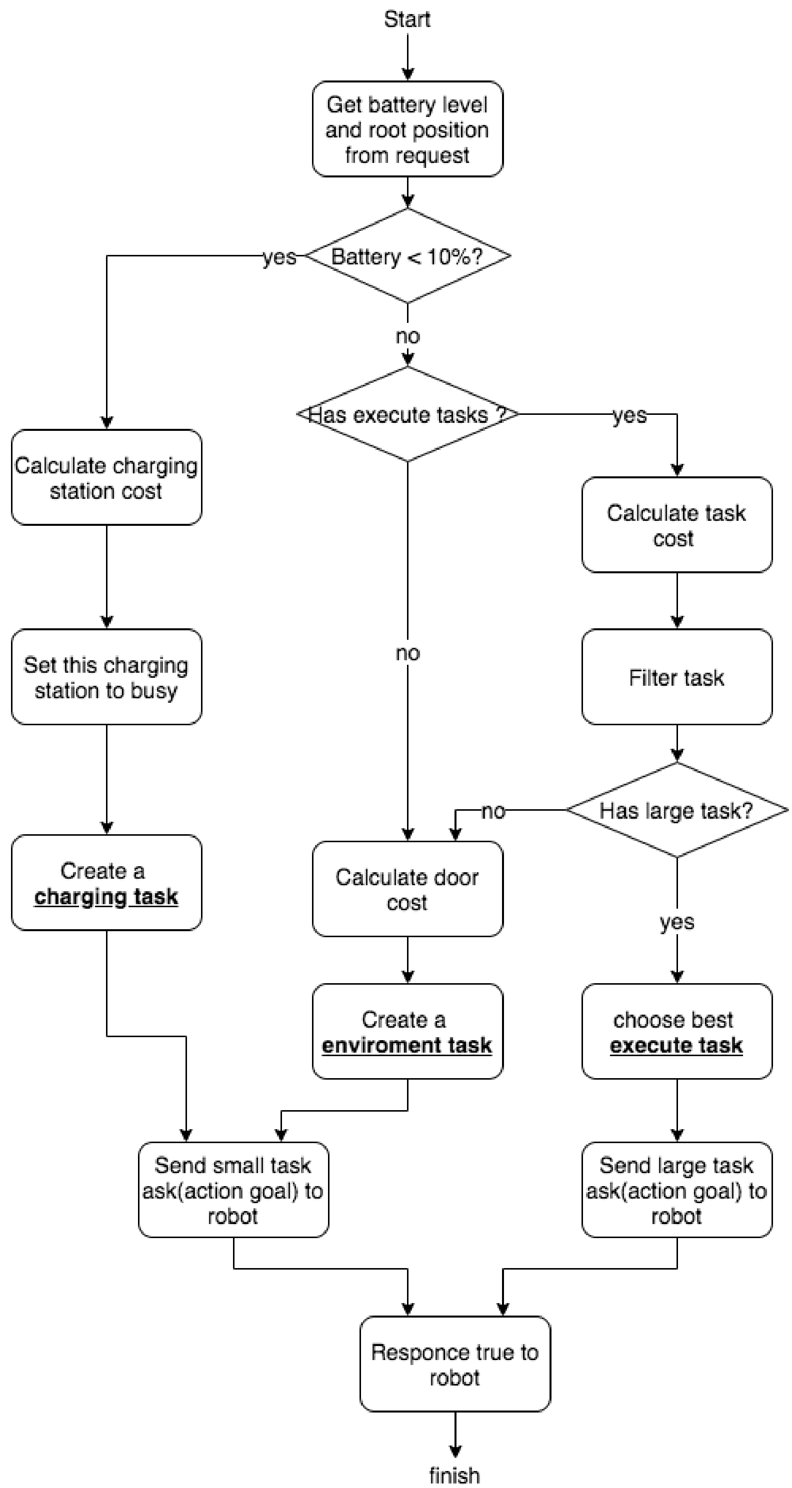


Figure 4.3: Centralized Pool Task Allocation

| Table Name | Example | Explaination |
|---|---|---|
| measurements | Table 5.8 | measurement result with timestamp |
| doors | Table 5.9 | door information |
| exe_rs | Table 5.2 | "execute task" experiment result |
| env_rs | Table 5.10 | "gather enviroment information task" experiment result |
| open_possibilities | Table 3.3 | open possibilities table for different time slots and weekdays |
| ... | ... | ... |

Table 4.7: Tables in Database

**Handle Task Result.** When the centralized pool receives a task result (Table 4.5), it updates "tasks" table. The failed "execute tasks" will be reused whileothers will be marked as "Cancel" or "Error"(Figure 4.4).

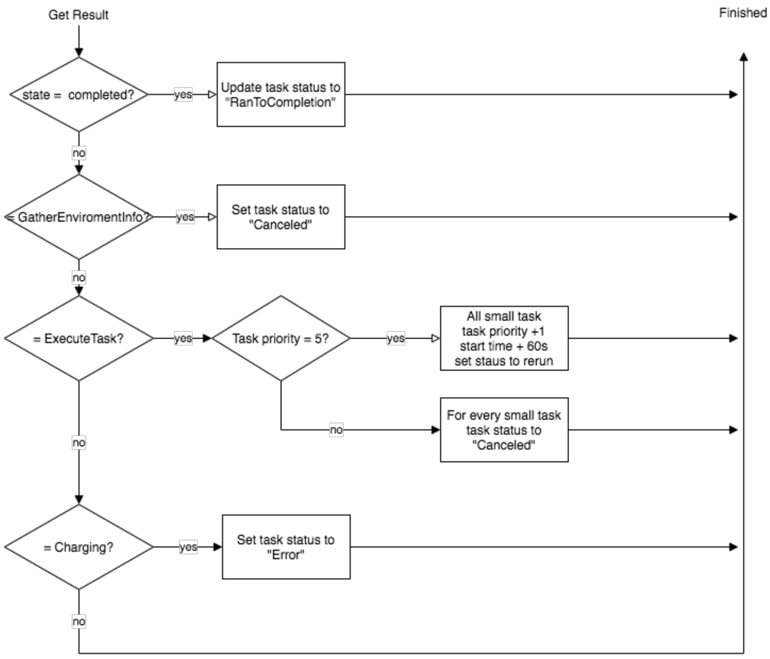


Figure 4.4: Centralized Pool Handle Task Result

### 4.3.2 Robot

**Robot Process Tasks** When the task queue(Figure 3.1) in a robot is empty, the robot requests a new task. If the robot gets a "charging task", it will move to the position of charging staion(Figure 3.3) and interact with charging station node (Section 4.3.3).

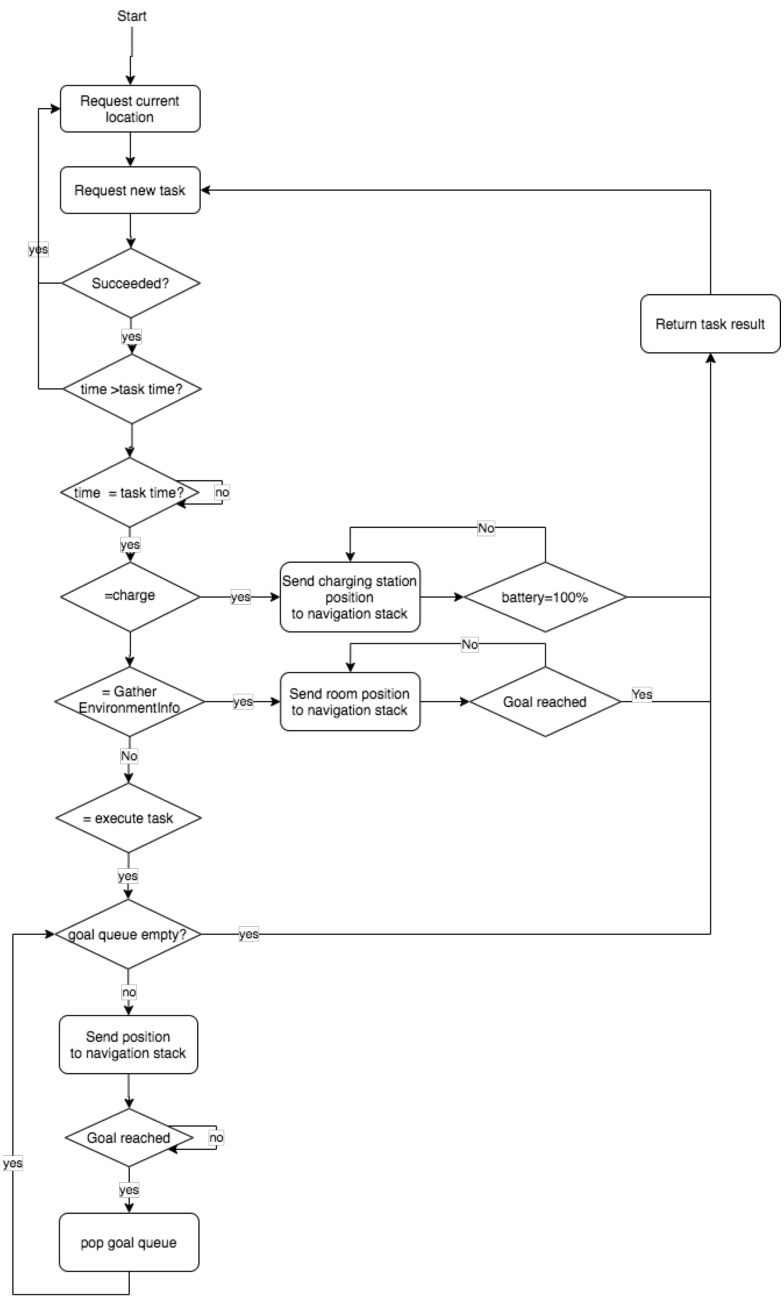


Figure 4.5: Robot Process Task

When a robot gets an "execute task" which is a large task, it will move to all goals in order. When a robot gets a "gather environment information" task, it will move to the door's position. During task processing, the timer checks periodically the status of navigation stack. If any errors occurs, the robot send a "failed" result with description to the centralized pool. When all tasks are complted without error, the robot will send "Succedded" result to the centralized pool.

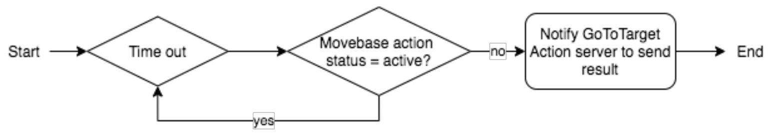


Figure 4.6: Robot Timer

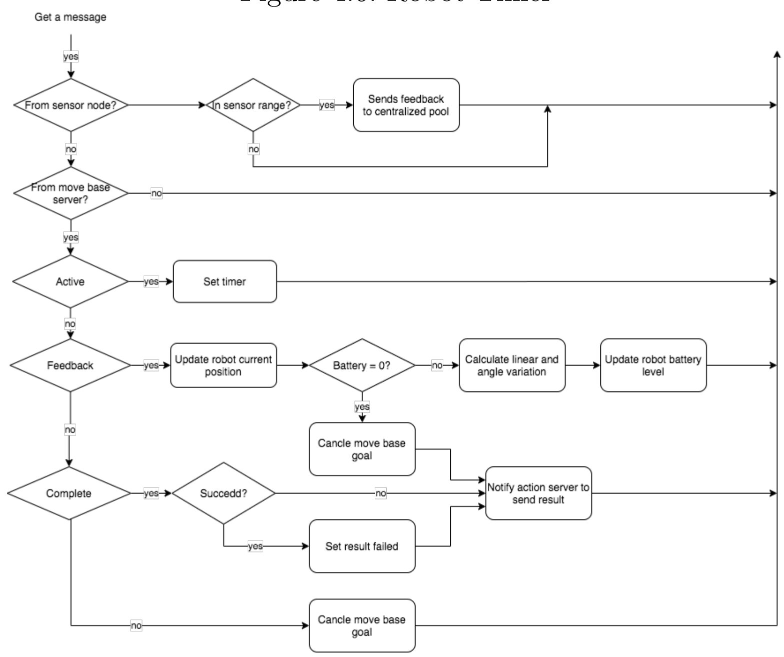


Figure 4.7: Robot Handle Message

**Robot Handle Messages** While a robot is processing a task, it listens to door sensors and forwards measurement result to the centralized pool. Besides messages from sensor, it also receives messages from "move_base" node. The details of robot message handling is shown in Figure 4.7.

### 4.3.3 Charging Station

The charging station consists of a charging station node and "charging station" table in database (Table 4.2).

A charging station has four states : "Fre", "Charging" and "Charging finishe". Its initial state is "Free". When a robot arrives the position of charging station, it will start interacting with charging station node (Figure 4.9). Once the charging station receives robot information, its state will be changed to "Charging" and its "battery level" will be increased and its "remaining time" will be decreased(Figure 4.8). Once finishing charging, its status will be set to "Charging finished". When robot leaves charging station, its status will be set to "Free".

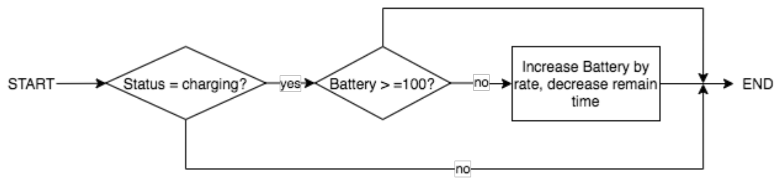


Figure 4.8: Charging Station Scheduled Charging Event in Database

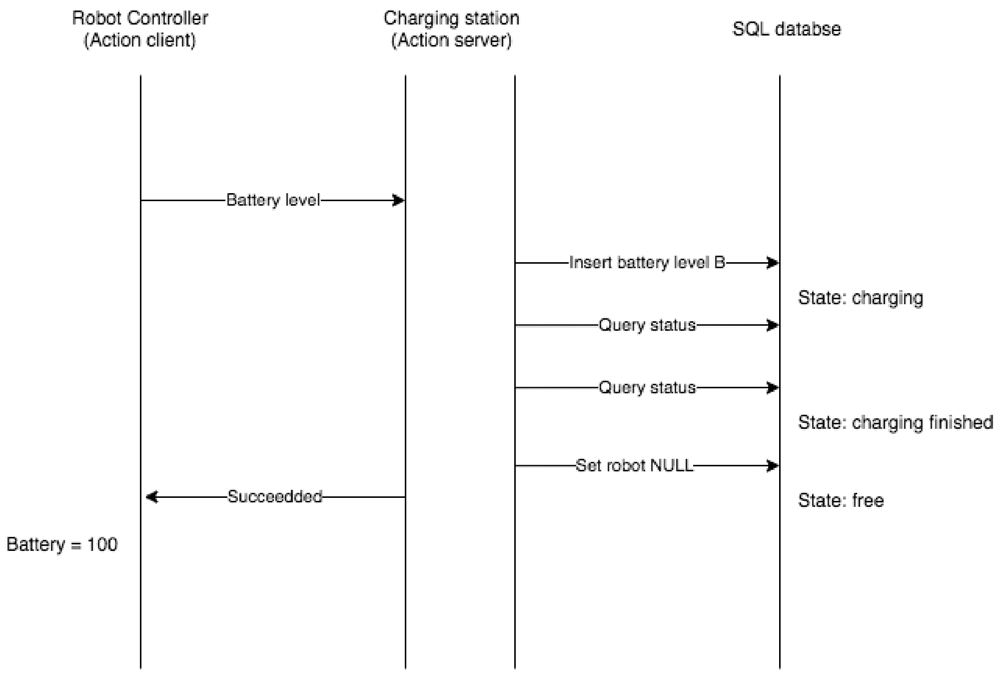


Figure 4.9: Charging Station Message

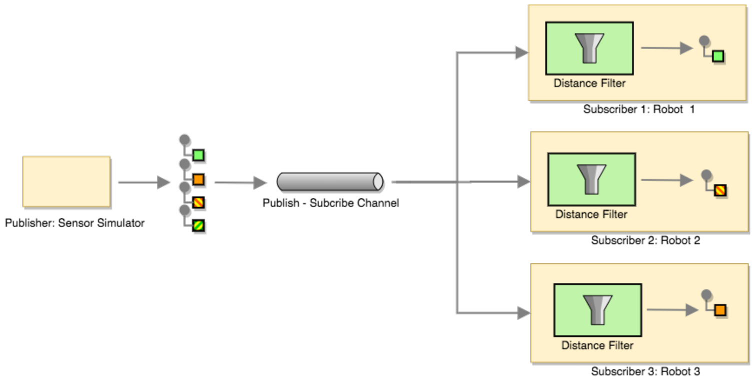


Figure 4.10: Sensor Simulator

### 4.3.4 Sensor Simulator

**Sensor Message Publish** In this project, a sensor simulator node is used to publish measurement results. The door simulator publish one messages for each door periodically to a ROS topic "sensor data". The message contains door id, sensor position and door status(Table 4.1.1). The door status are created according to open possibilities table (Table 3.3). For example, on monday (day of week is 2) at simulation time "10:30:00" (on timeslot 10:00:00-10:59:59), in 80% possibility a "door open" is generated and in 20% a "door closed" is generated (Initialized Open Possibility is 0.80).

**Sensor Message Subscribe** The process of sensor simulation is shown in Figure 4.3.4. The robots (robot controller nodes) subscribe the same ROS topic "sensor data". Every time the sensor simulator sends a message, all robots will receive this message at the same time. Their distance filters filter sensor messages with position outside the comminication range and keep sensor messages within the communication range. With this process, the sensor simulatior sends instant measurement result to robots within communication range. However, if the system is applied to the real world, instead of sending instant measurement result, the real world sensor could send a record with history measurement.

# Chapter 5

# Evaluation

In the system, cost functions are importent because an accurate cost function can improve the efficiency of task allocation. Followings are goals of experiments:

- Evaluate the need of decision variables.
- Find the best weight combinations in cost functions.
- Evaluate the hypothesis that the more robots, the faster to complete a task set, the higher the completion rate.

**Significant physical features** All simulation experiments were conducted using a multirobot simulation enviroment(Figure 5.1) Gazebo. As discussed in section 3.2, the map (Figure 5.2) was greated by SLAM [sla]. There were important coordinates: D1-D17 were doors; C1-C3 were charging stations; P1-P10 were points used as goal positions in "execute task" experiments. In addition, the experiment used simulation time. The simulation time was set in Gazebo world configuration file. When experiments started, the simulation time was configured as "2020-06-01 9:00:00".

## 5.1 Execute Task Evaluation Experiment

### 5.1.1 Experiment: Impact of Decision Variables

**Experiment Introduction** This set of experiments evaluated the need of four decision variables: battery consumption, waiting time, product of door open possibility and priority (Section 3.3.2).

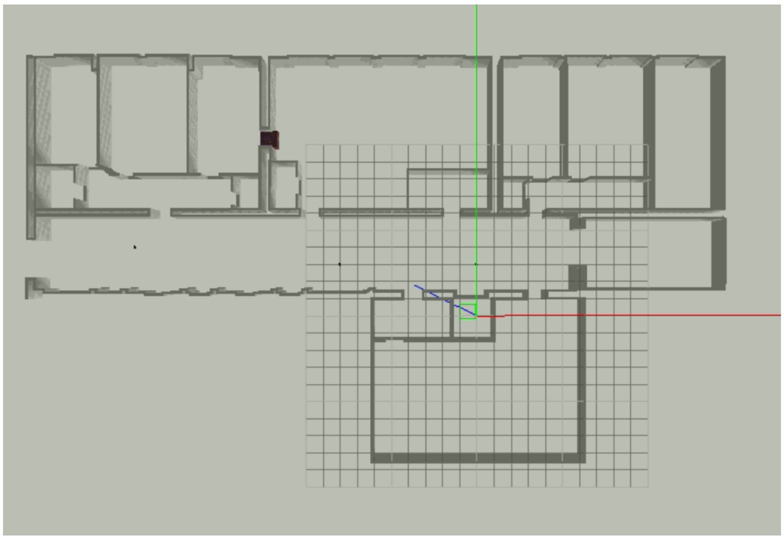

Figure 5.1: Gazebo Simulation

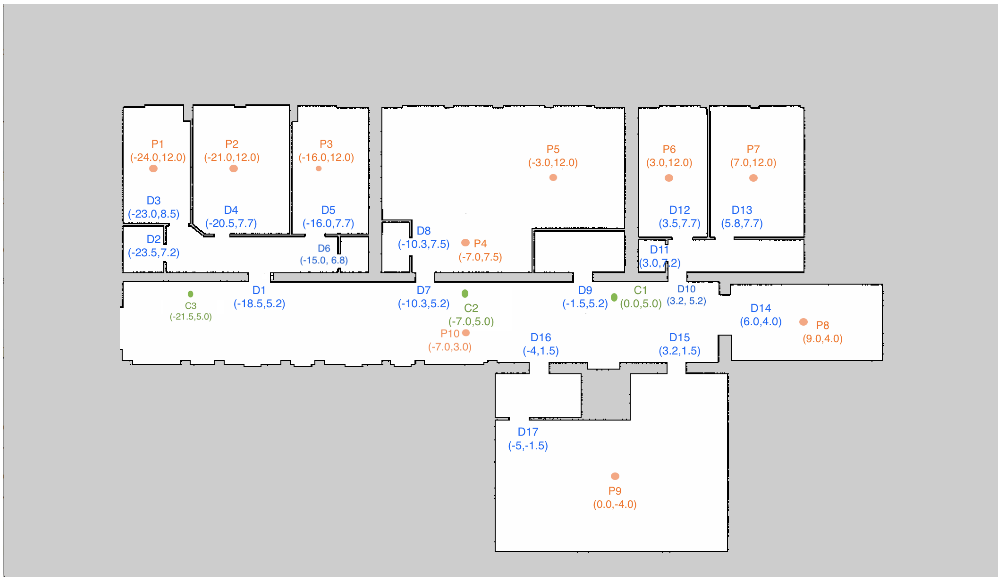


Figure 5.2: Experiment Map

| Task ID | Task Name | Start Time | Target ID | Robot ID | Priority | Task Status | Dependency | Finish Time | Description |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Charging | NULL | 18 | 1 | 5 | Succedded | 0 | 2020-06-02 11:03:12 | Succeeded |
| 2 | Charging | NULL | 19 | 2 | 5 | Succedded | 0 | 2020-06-02 11:03:12 | Succeeded |
| 3 | Charging | NULL | 20 | 3 | 5 | Succedded | 0 | 2020-06-02 11:03:11 | Succeeded |
| 4 | ExecuteTask | 2020-06-02 11:04:17 | 21 | 3 | 2 | Succedded | 0 | 2020-06-02 11:05:16 | Succeeded |
| 5 | ExecuteTask | 2020-06-02 11:05:22 | 22 | 1 | 2 | Succedded | 0 | 2020-06-02 11:07:29 | Succeeded |
| 6 | ExecuteTask | 2020-06-02 11:06:27 | 23 | 2 | 2 | Succedded | 0 | 2020-06-02 11:08:06 | Succeeded |
| 7 | ExecuteTask | 2020-06-02 11:07:32 | 24 | 3 | 2 | Succedded | 0 | 2020-06-02 11:09:25 | Succeeded |
| 8 | ExecuteTask | 2020-06-02 11:08:37 | 25 | 1 | 2 | Succedded | 0 | 2020-06-02 11:10:46 | Succeeded |
| 9 | ExecuteTask | 2020-06-02 11:09:42 | 26 | 2 | 2 | Succedded | 0 | 2020-06-02 11:12:42 | Succeeded |
| 10 | ExecuteTask | 2020-06-02 11:10:47 | 27 | 3 | 2 | Succedded | 0 | 2020-06-02 11:12:56 | Succeeded |
| 11 | ExecuteTask | 2020-06-02 11:11:52 | 28 | 1 | 2 | Succedded | 0 | 2020-06-02 11:14:13 | Succeeded |
| 12 | ExecuteTask | 2020-06-02 11:12:57 | 29 | 2 | 2 | Succedded | 0 | 2020-06-02 11:14:20 | Succeeded |
| 13 | ExecuteTask | 2020-06-02 11:14:02 | 30 | 3 | 2 | Succedded | 0 | 2020-06-02 11:15:36 | Succeeded |
| 14 | ExecuteTask | 2020-06-02 11:15:07 | 21 | 1 | 2 | Succedded | 0 | 2020-06-02 11:18:07 | Succeeded |
| 15 | ExecuteTask | 2020-06-02 11:16:12 | 22 | 2 | 2 | Succedded | 0 | 2020-06-02 11:18:49 | Succeeded |
| 16 | ExecuteTask | 2020-06-02 11:17:17 | 23 | 3 | 2 | Succedded | 0 | 2020-06-02 11:18:58 | Succeeded |
| 17 | ExecuteTask | 2020-06-02 11:18:22 | 24 | 1 | 2 | Succedded | 0 | 2020-06-02 11:20:14 | Succeeded |
| 18 | ExecuteTask | 2020-06-02 11:19:27 | 25 | 2 | 2 | Succedded | 0 | 2020-06-02 11:21:36 | Succeeded |
| 19 | Charging | NULL | 18 | 1 | 5 | Succedded | 0 | 2020-06-02 11:23:08 | Succeeded |
| 20 | Charging | NULL | 19 | 2 | 5 | Succedded | 0 | 2020-06-02 11:23:08 | Succeeded |
| 21 | Charging | NULL | 20 | 3 | 5 | Succedded | 0 | 2020-06-02 11:23:08 | Succeeded |
| 22 | ExecuteTask | 2020-06-02 11:24:13 | 21 | 3 | 2 | Succedded | 0 | 2020-06-02 11:25:12 | Succeeded |
| 23 | ExecuteTask | 2020-06-02 11:25:18 | 22 | 2 | 2 | Succedded | 0 | 2020-06-02 11:26:54 | Succeeded |
| 24 | ExecuteTask | 2020-06-02 11:26:23 | 23 | 1 | 2 | Succedded | 0 | 2020-06-02 11:28:35 | Succeeded |
| 25 | ExecuteTask | 2020-06-02 11:27:28 | 24 | 3 | 2 | Succedded | 0 | 2020-06-02 11:29:23 | Succeeded |
| 26 | ExecuteTask | 2020-06-02 11:28:33 | 25 | 2 | 2 | Succedded | 0 | 2020-06-02 11:30:41 | Succeeded |
| 27 | ExecuteTask | 2020-06-02 11:29:38 | 26 | 1 | 2 | Succedded | 0 | 2020-06-02 11:32:37 | Succeeded |
| 28 | ExecuteTask | 2020-06-02 11:30:43 | 27 | 3 | 2 | Succedded | 0 | 2020-06-02 11:32:54 | Succeeded |
| 29 | ExecuteTask | 2020-06-02 11:31:48 | 28 | 2 | 2 | Succedded | 0 | 2020-06-02 11:34:09 | Succeeded |
| 30 | ExecuteTask | 2020-06-02 11:32:53 | 29 | 1 | 2 | Succedded | 0 | 2020-06-02 11:34:15 | Succeeded |
| 31 | ExecuteTask | 2020-06-02 11:33:58 | 30 | 3 | 2 | Succedded | 0 | 2020-06-02 11:35:32 | Succeeded |
| 32 | ExecuteTask | 2020-06-02 11:35:03 | 21 | 2 | 2 | Succedded | 0 | 2020-06-02 11:38:03 | Succeeded |
| 33 | ExecuteTask | 2020-06-02 11:36:08 | 22 | 1 | 2 | Succedded | 0 | 2020-06-02 11:38:45 | Succeeded |
| 34 | ExecuteTask | 2020-06-02 11:37:13 | 23 | 3 | 2 | Succedded | 0 | 2020-06-02 11:38:54 | Succeeded |
| 35 | ExecuteTask | 2020-06-02 11:38:18 | 24 | 2 | 2 | Succedded | 0 | 2020-06-02 11:40:11 | Succeeded |
| 36 | ExecuteTask | 2020-06-02 11:39:23 | 25 | 1 | 2 | Succedded | 0 | 2020-06-02 11:41:40 | Succeeded |
| 37 | Charging | NULL | 18 | 1 | 5 | Succedded | 0 | 2020-06-02 11:43:45 | Succeeded |
| 38 | Charging | NULL | 19 | 2 | 5 | Succedded | 0 | 2020-06-02 11:43:45 | Succeeded |
| 39 | Charging | NULL | 20 | 3 | 5 | Succedded | 0 | 2020-06-02 11:43:45 | Succeeded |

Table 5.1: Task Table

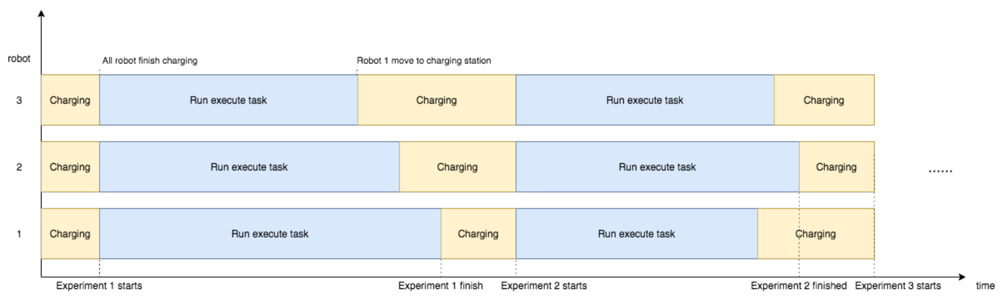


Figure 5.3: Execute Task Experiment Timeline

**Sampling design** Table 3.2 shows ”task table” in database. At this point, robots finished task 4-21 in experiment 1 and finished 21-39 in experiment 2. When simulation started, 3 robots ran charging task (Task 1-3) and moved to corresponding charging station and started charging (Figure 4.8). For example, robot 1 charged at charging staion 1, robot 2 charged at charging staion 2, robot 3 charged at charging staion 3. When robot fully charged, the first experiment started (Figure 5.3), and 15 “execute tasks” and 3 “charging tasks” were created (Task 4-21). Especially, in the same experiment, the task interal were the same (Table )3.2). In different experiment, the corresponding task had the same goal positions (Task 4 and 21, Task 4 and 22). When all robot finished tasks, the experiment would be finished and robots charged at previous charging station(Task 19-21). These rules ensured that robot not only started at their initial positions but also processed same task set and not shuted down because of power exhaustion.

**Analysis design** As is discussed in last paragraph, in each experiemnts, robots need to finish 15 “execute tasks” and 3 “charging tasks”. The experiment start time was when all robot finshed charging and started request task (Figure 5.3), and the experiment finish time was when the latest task is finished. The experiment duration was an important evaluation factor, which was the time difference between experiment start time and experiment finished time. There end state of “execute tasks” were evaluated. In an experiment result table, the “Succeeded” column counted the tasks ended with “Succeeded” state, the “Expired” column counted the tasks ended with “Succeeded” state, the “Failed” column counted the tasks ended with “Runing”, “Error”, “Canceled”, “To rerun” states. Table 5.2 is an example of experiment result.

**Experiment Result** The experiment result (Table 5.2.) shows that in experiment 1, all 15 task were succesfully finsihed with mininal experiment duraion. In experiment 2, 4, 5 mores than half of the task were expired. In experiment 3, all tasks were successfully finished but it took more time than experiment 1. The experiment 6-24 (Table 5.3 and Figure 5.3) evaluated each decision variable sepratelly.

**Experiment Analysis.** Comparing experiment 1-5, it was concluded that the cost function with multiple decision variables had better performance than a cost function with single decition variable. However, according to experiment 6-24, the “experiment duration” changed very little when only one desition variable changed others unchanged, therefore how each decision variable affect the task allocation is still unknown.

### 5.1.2 Experiment: Find the Best Weight Combinations

**Experiment Introduction** In addition to cost function, the task interval may affect experiment duration and experiment succeeded rate. Therefore, three set experiment

| Experiment | $W_{\text{battery consumption}}$ | $W_{\text{waiting time}}$ | $W_{\text{door open possibility}}$ | $W_{\text{priority}}$ | Experiment Duration | Total Task | Succedded Task | Expired Task | Failed Task |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1.00 | 1.00 | -1.00 | -1.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 2 | 1.00 | 0.00 | 0.00 | 0.00 | 00:17:23 | 15 | 7 | 8 | 0 |
| 3 | 0.00 | 1.00 | 0.00 | 0.00 | 00:18:56 | 15 | 15 | 0 | 0 |
| 4 | 0.00 | 0.00 | -1.00 | 0.00 | 00:18:41 | 15 | 5 | 10 | 0 |
| 5 | 0.00 | 0.00 | 0.00 | -1.00 | 00:17:21 | 15 | 3 | 12 | 0 |

Table 5.2: Runing execute task with single decision variable and multiple decision variables

| Experiment | $W_{\text{battery consumption}}$ | $W_{\text{waiting time}}$ | $W_{\text{door open possibility}}$ | $W_{\text{priority}}$ | Experiment Duration | Total Task | Succedded Task | Expired Task | Failed Task |
|---|---|---|---|---|---|---|---|---|---|
| 6 | 20 | 1 | -1 | -1 | 00:09:26 | 15 | 10 | 5 | 0 |
| 7 | 40 | 1 | -1 | -1 | 00:09:48 | 15 | 10 | 5 | 0 |
| 8 | 60 | 1 | -1 | -1 | 00:09:50 | 15 | 10 | 5 | 0 |
| 9 | 80 | 1 | -1 | -1 | 00:09:47 | 15 | 10 | 5 | 0 |
| 10 | 100 | 1 | -1 | -1 | 00:09:06 | 15 | 10 | 5 | 0 |
| 11 | 1 | 20 | -1 | -1 | 00:09:48 | 15 | 10 | 5 | 0 |
| 12 | 1 | 40 | -1 | -1 | 00:09:38 | 15 | 9 | 5 | 0 |
| 13 | 1 | 60 | -1 | -1 | 00:10:09 | 15 | 10 | 5 | 0 |
| 14 | 1 | 80 | -1 | -1 | 00:09:46 | 15 | 10 | 5 | 0 |
| 15 | 1 | 100 | -1 | -1 | 00:10:01 | 15 | 10 | 5 | 0 |
| 16 | 1 | 1 | -20 | -1 | 00:09:48 | 15 | 10 | 5 | 0 |
| 17 | 1 | 1 | -40 | -1 | 00:09:50 | 15 | 10 | 5 | 0 |
| 18 | 1 | 1 | -60 | -1 | 00:10:24 | 15 | 10 | 5 | 0 |
| 19 | 1 | 1 | -80 | -1 | 00:09:48 | 15 | 10 | 5 | 0 |
| 20 | 1 | 1 | -100 | -1 | 00:09:35 | 15 | 10 | 5 | 0 |
| 21 | 1 | 1 | -1 | -20 | 00:10:22 | 15 | 10 | 5 | 0 |
| 22 | 1 | 1 | -1 | -40 | 00:10:02 | 15 | 10 | 5 | 0 |
| 23 | 1 | 1 | -1 | -60 | 00:09:48 | 15 | 10 | 5 | 0 |
| 24 | 1 | 1 | -1 | -80 | 00:09:51 | 15 | 10 | 5 | 0 |

Table 5.3: only one desition variable changed others unchanged

are generated to find the best weight combinations.

**Sampling design** Table 3.2 shows "task table" in database. At this point, robots finished task 4-21 in experiment 1 and finished 21-39 in experiment 2. When simulation started, robots ran charging task (Task 1-3) and moved to corresponding charging station and started charging (Figure 4.8). For example, robot 1 charged at charging staion 1, robot 2 charged at charging staion 2, robot 3 charged at charging staion 3. When robot fully charged, the first experiment started (Figure 5.3), and 15 "execute tasks" and 3 "charging tasks" were created (Task 4-21). Especially, in the same experiment, the task interal were the same. In Table 3.2 the task interval was 65 seconds. In different experiment, the corresponding task had the same goal positions (Task 4 and 21, Task 4 and 22). When all robot finished tasks, the experiment would be finished and robots charged at previous charging station(Task 19-21). These rules ensured that robot not only started at their initial positions but also processed same task set and not shuted down because of power exhaustion.

**Analysis design** In each experiemnt, 3 robots need to finish 15 "execute tasks" and 3 "charging tasks". The experiment start time was when all robot finshed charging and started request task (Figure 5.3), and the experiment finish time was when the latest task is finished. The experiment duration was an important evaluation factor, which was the time difference between experiment start time and experiment finished time.

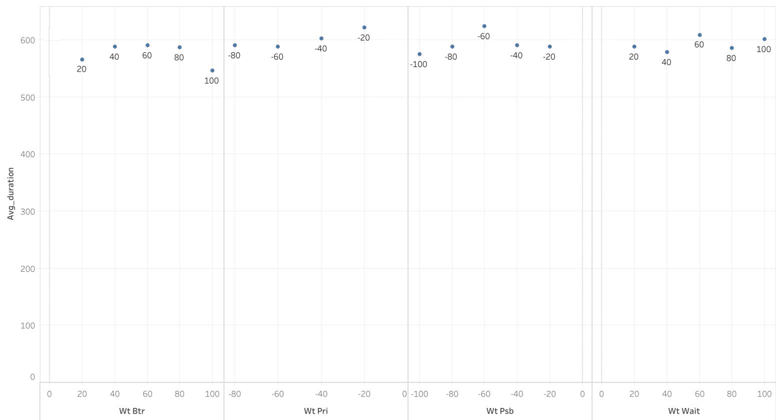


Figure 5.4: Only one desition variable changed others unchanged

There end state of "execute tasks" were evaluated. In an experiment result table, the "Succeeded" column counted the tasks ended with "Succeeded" state, the "Expired" column counted the tasks ended with "Succeeded" state, the "Failed" column counted the tasks ended with "Runing", "Error", "Canceled", "To rerun" states. Table 5.2 is an example of experiment result.

**Experiment Result** The first set of experiment uses "execute tasks" with 30 seconds interval. Its best weight combinations is shown in (Table 5.4). The second set of experiments used "execute tasks" with 45 seconds interval. Its best weight combinations is shown in (Table 5.4). The third set of experiments used "execute tasks" with 65 seconds interval. Its best weight combinations is shown in (Table 5.6).

**Analysis.** It was concluded that the longer the task interval, the higher the task completion rate.

| $w_{\text{battery consumption}}$ | $w_{\text{waiting time}}$ | $w_{\text{door open possibility}}$ | $w_{\text{priority}}$ | Experiment Duration | Total Task | Succedded Task | Expired Task | Failed Task |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | -1 | -3 | 00:09:05 | 15 | 10 | 5 | 0 |
| 1 | 1 | -30 | -1 | 00:09:05 | 15 | 10 | 5 | 0 |
| 1 | 20 | -20 | -1 | 00:09:05 | 15 | 10 | 5 | 0 |
| 1 | 25 | -1 | -1 | 00:09:05 | 15 | 10 | 5 | 0 |
| 5 | 1 | -5 | -5 | 00:09:05 | 15 | 10 | 5 | 0 |

Table 5.4: Best weight combinations with task interval 30s

| $W_{\text{battery consumption}}$ | $W_{\text{waiting time}}$ | $W_{\text{door open possibility}}$ | $W_{\text{priority}}$ | Experiment Duration | Total Task | Succedded Task | Expired Task | Failed Task |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | -1 | -10 | 00:11:35 | 15 | 12 | 3 | 0 |
| 1 | 50 | -50 | -50 | 00:11:35 | 15 | 12 | 3 | 0 |
| 25 | 20 | 0 | 0 | 00:11:35 | 15 | 12 | 3 | 0 |
| 35 | 30 | 0 | 0 | 00:11:35 | 15 | 12 | 3 | 0 |
| 40 | 10 | 0 | 0 | 00:11:35 | 15 | 12 | 3 | 0 |
| 45 | 1 | -1 | -1 | 00:11:35 | 15 | 12 | 3 | 0 |
| 50 | 50 | -50 | -1 | 00:11:35 | 15 | 12 | 3 | 0 |

Table 5.5: Best weight combinations with task interval 45s

| $W_{\text{battery consumption}}$ | $W_{\text{waiting time}}$ | $W_{\text{door open possibility}}$ | $W_{\text{priority}}$ | Experiment Duration | Total Task | Succedded Task | Expired Task | Failed Task |
|---|---|---|---|---|---|---|---|---|
| 10.00 | 1.00 | -1.00 | 0.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 1.00 | 1.00 | -1.00 | 0.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 1.00 | 1.00 | -1.00 | -1.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 1.00 | 5.00 | -1.00 | -1.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 1.00 | 5.00 | -1.00 | -5.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 1.00 | 10.00 | -10.00 | -1.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 1.00 | 5.00 | -5.00 | -5.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 1.00 | 5.00 | -10.00 | -5.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 1.00 | 5.00 | -1.00 | -1.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 1.00 | 20.00 | -1.00 | -1.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 1.00 | 1.00 | -1.00 | -30.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 1.00 | 1.00 | -1.00 | -35.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 10.00 | 1.00 | -1.00 | -5.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 10.00 | 10.00 | -1.00 | -1.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 10.00 | 1.00 | -1.00 | -1.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 10.00 | 1.00 | -10.00 | -1.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 10.00 | 5.00 | -5.00 | -5.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 10.00 | 1.00 | -10.00 | -5.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 10.00 | 10.00 | -10.00 | -10.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 10.00 | 1.00 | -10.00 | -1.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 25.00 | 1.00 | -1.00 | -25.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 30.00 | 30.00 | -1.00 | -1.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 45.00 | 1.00 | -1.00 | -1.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 45.00 | 1.00 | -45.00 | -1.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 5.00 | 10.00 | -10.00 | -1.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 5.00 | 5.00 | -1.00 | -1.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 5.00 | 10.00 | -5.00 | -1.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 5.00 | 5.00 | -10.00 | -1.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 5.00 | 10.00 | -10.00 | -1.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 5.00 | 5.00 | -1.00 | -5.00 | 00:18:23 | 15 | 15 | 0 | 0 |
| 5.00 | 10.00 | -5.00 | -10.00 | 00:18:23 | 15 | 15 | 0 | 0 |

Table 5.6: Best weight combinations with task interval 65s

### 5.1.3 Experiment: Impact of the Number of Robots

**Experiment Introduction** In this set of experiments, how the number of robots affected task allocation was evaluated. One of the best weight combination was selected: $W_{\text{battery}} = 10, W_{\text{waiting time}} = 1, W_{\text{possibility}} = -1, W_{\text{priority}} = -10$.

**Sampling design** When simulation started, robots ran charging task (Task 1-3 in Table 3.2) and moved to corresponding charging station and started charging (Figure 4.8). For example, robot 1 charged at charging staion 1, robot 2 charged at charging staion 2, robot 3 charged at charging staion 3. When robot fully charged, the first experiment started (Figure 5.3), and 15 "execute tasks" and 3 "charging tasks" were created (Task 4-21). Especially, in the same experiment, the task interal were the same. In Table 3.2 the task interval was 65 seconds. In different experiment, the corresponding task had the same goal positions (Task 4 and 21, Task 4 and 22). When all robot finished tasks, the experiment would be finished and robots charged at previous charging station(Task 19-21). These rules ensured that robot not only started at their initial positions but also processed same task set and not shuted down because of power exhaustion.

**Analysis design** As is discussed in last paragraph, in each experiemnts, robots need to finish 15 "execute tasks" and 3 "charging tasks". The experiment start time was when all robot finshed charging and started request task (Figure 5.3), and the experiment finish time was when the latest task is finished. The experiment duration was an important evaluation factor, which was the time difference between experiment start time and experiment finished time. There end state of "execute tasks" were evaluated. In an experiment result table, the "Succeeded" column counted the tasks ended with "Succeeded" state, the "Expired" column counted the tasks ended with "Succeeded" state, the "Failed" column counted the tasks ended with "Runing", "Error", "Canceled", "To rerun" states. Table 5.2 is an example of experiment result.

**Result** The experiment result (Table 5.7) showed that when one robot processed tasks, only 7 tasks were finished successfully; when two robots processed the same task set, 11 tasks were finished succesfully. Compared with experiment 1, experiment 2 tooks 102 seconds more but 2 more tasks were completed; when three robots processed the same task set, all 15 tasks were finished succesfully. Compared with experiment 2, the experiment duration is slightly increased but 4 more tasks were finished succesfully.

**Analysis** It was concluded that as the number of robots increased, the task completion rate increased. However, the relationship between the number of robots and the speed of completing the task set cannot be obtained.

| Experiment | Robot number | Experiment Duration | Total Task | Succedded Task | Expired Task | Failed Task |
|---|---|---|---|---|---|---|
| 1 | 1 | 00:16:56 | 15 | 7 | 8 | 0 |
| 2 | 1 | 00:16:52 | 15 | 7 | 8 | 0 |
| 3 | 1 | 00:17:02 | 15 | 7 | 8 | 0 |
| Average | 1 | 00:16:56 | 15 | 7 | 8 | 0 |
| 4 | 2 | 00:19:06 | 15 | 11 | 4 | 0 |
| 5 | 2 | 00:18:24 | 15 | 11 | 4 | 0 |
| 6 | 2 | 00:18:24 | 15 | 11 | 4 | 0 |
| Average | 2 | 00:18:38 | 15 | 11 | 4 | 0 |
| 7 | 3 | 00:18:23 | 15 | 15 | 0 | 0 |
| 8 | 3 | 00:18:55 | 15 | 15 | 0 | 0 |
| 9 | 3 | 00:18:57 | 15 | 15 | 0 | 0 |
| Average | 3 | 00:18:45 | 15 | 15 | 0 | 0 |

Table 5.7: Result 4: Different number of robot processing the same task set

## 5.2 Enviroment Task Evaluation Experiment

### 5.2.1 Experiment: Find the Best Weight Combinations

**Experiment Introduction** The Goal of this set of experiment was to find the best combinations in 30 weight combinations. In addition, the conditions $W_{\text{update}} = 0$ was not testable, because during the experiment centralized pool generate the same "gather enviroment information tasks", which let robots not moving and measuring their nearest door.

**Sampling Design** When simulation started, robots ran charging task (Task 1-3) at their charging station and started charging (Figure 4.8). For example, robot 1 charged at charging staion 1, robot 2 charged at charging staion 2, robot 3 charged at charging staion 3. When robot fully charged, the first experiment started (Figure 5.5). After that, the task allocation module in centralized pool created a "gather enviroment information task" to each robot requested a task. After a constant time T, experiments were finished and robots went to their charging stations. These rules ensured that 1) Each Robot always started at an initial position. 2) Robots would not shut down because of power exhaustion. 3) Robots spent the same time gathering enviroment information.

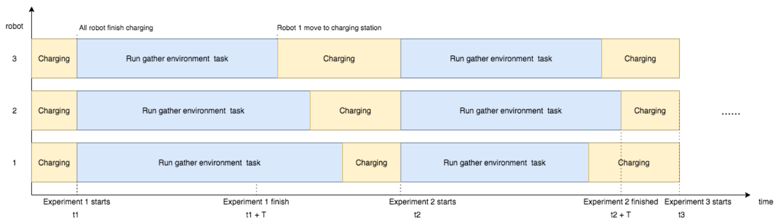


Figure 5.5: Enviroment Task Experiment Timeline

**Analysis design** The experiment start time was when all robot finshed charging and started request task (Figure 5.5). The experiment duraion is a constant value T. The experiment finish time $T_{\text{finish time}} = T_{\text{start time}} + T$. Some important factors are evaluated.

- *Minial last update.* The "minial last update" factor is the time difference between experiment start time and minial value in "last update" column in door table (Table 5.9) when an experiment finished. For example, "mininal last update" factor in experiment 1 (Table 5.10) is "00:02:57". It means that the door in the worst case not be measured since 2 minites 57 seconds after experiment start.

- *Average Update Interal* The "Average update Interal" means the average interval of door update. For example, "Average Update interal" factor in experiment 1 (Table 5.10) is "00:01:00". It means that on average, every door is updated every minute.
- *Succedded task.* The "Number of task" factor means the number of succeeded "gather enviroment information" task.

**Experiemnt Result** The experiment result was shown in Table 5.10.

**Expariment Analysis** As shown in experiment result, the "Minial Last Update" value is much less than experiment duraion. It means some doors were not timely updated information. One possible reason is that the velocity of robot is small (about 0.2 per second), the simulation time was not long enough to let three robot pass to all doors. Another possible reason is that the robot's route is partially duplicated. For example, when system started, robot 1 was in charging station 1 and got a task to door 3, while robot 2 was in charging station 2 and got a task to door 4. As shown in Figure 3.3, these two route are partially duplicated, both of them pass through door 1 and entered room 1 (Figure 3.2). The idealy solution was to give both task to one robot.

| Door ID | Door Status | Date Time |
|---|---|---|
| 9 | 1 | 2020-06-01 09:05:39 |
| 1 | 0 | 2020-06-01 09:05:49 |
| 7 | 1 | 2020-06-01 09:05:49 |
| 9 | 1 | 2020-06-01 09:05:49 |
| 1 | 1 | 2020-06-01 09:05:59 |
| 7 | 1 | 2020-06-01 09:05:59 |
| 9 | 1 | 2020-06-01 09:05:59 |
| 7 | 1 | 2020-06-01 09:06:09 |
| 16 | 0 | 2020-06-01 09:06:29 |
| 7 | 1 | 2020-06-01 09:06:39 |
| 16 | 1 | 2020-06-01 09:06:39 |
| 9 | 1 | 2020-06-01 09:06:49 |
| 8 | 0 | 2020-06-01 09:06:59 |
| ... | ... | ... |

Table 5.8: Measurement Result Table

| Door ID | Last Update | Is Used |
|---|---|---|
| 1 | 2020-06-01 15:15:26 | 0 |
| 2 | 2020-06-01 15:15:06 | 0 |
| 3 | 2020-06-01 15:12:36 | 0 |
| 4 | 2020-06-01 15:15:16 | 0 |
| 5 | 2020-06-01 15:11:46 | 0 |
| 6 | 2020-06-01 15:11:36 | 1 |
| 7 | 2020-06-01 15:14:26 | 0 |
| ... | ... | ... |

Table 5.9: Doors Table

| Experiment | $W_{\text{battery consumption}}$ | $W_{\text{update}}$ | $W_{\text{possibility}}$ | Experiment Duration | Minial Last Update | Average Update Interal | Succedded Task |
|---|---|---|---|---|---|---|---|
| 1 | 1.00 | -1.00 | -1.00 | 00:10:00 | 00:02:57 | 00:01:00 | 19 |
| 2 | 1.00 | -1.00 | 0.00 | 00:10:00 | 00:03:46 | 00:01:08 | 28 |
| 3 | 0.00 | -1.00 | -1.00 | 00:10:00 | 00:01:52 | 00:00:50 | 15 |
| 4 | 0.00 | -1.00 | 0.00 | 00:10:00 | 00:01:11 | 00:00:47 | 14 |
| 5 | 1.00 | -1.00 | -5.00 | 00:10:00 | 00:03:13 | 00:00:50 | 20 |
| 6 | 1.00 | -1.00 | 10.00 | 00:10:00 | 00:04:06 | 00:01:15 | 15 |
| 7 | 1.00 | -5.00 | -1.00 | 00:10:00 | 00:03:20 | 00:01:02 | 21 |
| 8 | 1.00 | -5.00 | -5.00 | 00:10:00 | 00:02:52 | 00:00:58 | 21 |
| 9 | 1.00 | -5.00 | -10.00 | 00:10:00 | 00:01:31 | 00:00:57 | 22 |
| 10 | 1.00 | -10.00 | -1.00 | 00:10:00 | 00:03:04 | 00:01:21 | 22 |
| 11 | 1.00 | -10.00 | -5.00 | 00:10:00 | 00:03:55 | 00:01:30 | 22 |
| 12 | 1.00 | -10.00 | -10.00 | 00:10:00 | 00:03:22 | 00:01:12 | 21 |
| 13 | 5.00 | -1.00 | -1.00 | 00:10:00 | 00:02:54 | 00:00:59 | 27 |
| 14 | 5.00 | -1.00 | -5.00 | 00:10:00 | 00:01:46 | 00:00:56 | 21 |
| 15 | 5.00 | -1.00 | -10.00 | 00:10:00 | 00:04:05 | 00:01:14 | 23 |
| 16 | 5.00 | -5.00 | -1.00 | 00:10:00 | 00:03:08 | 00:00:55 | 26 |
| 17 | 5.00 | -5.00 | -5.00 | 00:10:00 | 00:02:58 | 00:00:54 | 24 |
| 18 | 5.00 | -5.00 | -10.00 | 00:10:00 | 00:04:48 | 00:01:17 | 23 |
| 19 | 5.00 | -10.00 | -1.00 | 00:10:00 | 00:03:52 | 00:01:01 | 31 |
| 20 | 5.00 | -10.00 | -5.00 | 00:10:00 | 00:02:06 | 00:00:47 | 20 |
| 21 | 5.00 | -10.00 | -10.00 | 00:10:00 | 00:02:18 | 00:00:48 | 19 |
| 22 | 10.00 | -1.00 | -1.00 | 00:10:00 | 00:03:53 | 00:01:10 | 29 |
| 23 | 10.00 | -1.00 | -5.00 | 00:10:00 | 00:03:01 | 00:01:06 | 26 |
| 24 | 10.00 | -1.00 | -10.00 | 00:10:00 | 00:04:31 | 00:01:04 | 23 |
| 25 | 10.00 | -5.00 | -1.00 | 00:10:00 | 00:02:58 | 00:01:08 | 25 |
| 26 | 10.00 | -5.00 | -5.00 | 00:10:00 | 00:03:37 | 00:01:08 | 26 |
| 27 | 10.00 | -5.00 | -10.00 | 00:10:00 | 00:02:22 | 00:00:55 | 20 |
| 28 | 10.00 | -10.00 | -1.00 | 00:10:00 | 00:04:39 | 00:01:07 | 29 |
| 29 | 10.00 | -10.00 | -5.00 | 00:10:00 | 00:02:40 | 00:00:55 | 23 |
| 30 | 10.00 | -10.00 | -10.00 | 00:10:00 | 00:04:03 | 00:01:07 | 30 |

Table 5.10: Enviroment Experiment Result Table

# Chapter 6

# Discussion

The meaning of this paragraph is to interpret the results of the performed work. It will always connect the introduction, the postulated hypothesis and the results of the thesis/bachelor/master.

It should answer the following questions:

- Could your results answer your initial questions?
- Did your results agree with your initial hypothesis?
- Did you close your problem, or there are still things to be solved? If yes, what will you do to solve them?

write about limitations on sensor simulator: sensor send a table of measuement results.

# Chapter 7

# Acknowledgements

(This part is optional, and it could be completely excluded by deleting
**\include {content/chapters/chapter7}**
from the Firstname_Lastname_Diplom_Master_arbeit.tex file)

This paragraph could mention people or institutions that supported you to some extent with your work or friends and relatives that supported you during your study period.

# Bibliography


[IGM97] Ikeda, Mitsuru, Shogo Go and Riichiro Mizoguchi: *Opportunistic Group Formation*. In *Proceedings of the Conference on Artificial Intelligence in Education (AI-ED)*, pages 167–174, Amsterdam, 1997. IOS Press.

[IGS17] Ivan Gavran, Rupak Majumdar and Indranil Saha: *Antlab: a Multi-Robot Task Server*. ACM Transactions on Embedded Computing Systems, 9:39–58, 2017.

[MAdR02] Middleton, S. E., H. Alani and D. C. de Roure: *Exploiting Synergy between Ontologies and Recommender Systems*. In *Proceedings of the Semantic Web Workshop at the WWW Conference*, Honolulu, HI, 2002.

[mov] *Move Base Node*. `http://wiki.ros.org/move_base`. accessed 2020-09-25.

[Sal89] Salton, Gerard: *Automatic text processing — the transformation, analysis, and retrieval of information by computer*. Addison-Wesley series in computer science. Addison-Wesley, 1989.

[sla] *SLAM*. `https://emanual.robotis.com/docs/en/platform/turtlebot3/slam/#ros-1-slam`. accessed 2020-09-28.

**Versicherung an Eides Statt**

*German*

Hiermit versichere ich, dass ich die vorliegende Bachelor(Master)arbeit selbständig - mit Ausnahme der Anleitung durch die Betreuer - verfasst, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt, sowie Zitate kenntlich gemacht habe. Ich versichere dass ich alle entsprechenden Angaben nach bestem Wissen und Gewissen vorgenommen habe, dass sie der Wahrheit entspechen und dass ich nichts verschwiegen habe. Mir ist bekannt, dass eine falsche Versicherung an Eides Statt nach §156 und nach §163 Abs. 1 des Strafgesetzbuches mit Freiheitsstrafe oder Geldstrafe bestraft wird.

*English*

I hereby declare that I have written this Bachelor (Master) thesis - except for the guidance of the supervisor - independently, using no other than the specified sources and resources, and that all quotations have been indicated. I declare that I have reported to the best of my knowledge all the relevant information, that it is true and that I concealed nothing. I am aware that a false declaration will be punished according to §156 and §163 par. 1 of the criminal code with a prison sentence or a monetary penalty.

Essen, October 5, 2020
(Ort, Datum)

Xuanjiao Zhu