Carleton University

Final Report

Title Page

<u>Computer Systems Design Lab</u>

Group Member 1: Kevin Belanger (101121709) L1

Group Member 2: Ezra Pierce (100991590) L1

Group Member 3: Mohammad Issa (101065045) L1

Group Member 4: Haoyu Xu (101088272) L1

Course: SYSC 4805

Team: Little Boy Blue

Date Submitted: April 12th, 2022

Table of Contents

Summary

The purpose of this project is to propose a robot built using CoppeliaSim that clears off the snow in an enclosed area. This can be done using various tools learned in the previous labs using CoppeliaSim. This proposal plans activities to accomplish within a 7 week period. Work with a team based environment to come up with a plan that is efficient and effective to develop the robot. CoppeliaSim is a robot simulator that is used for fast algorithm development, factory automation simulations, and robotics related education. The proposal will cover these topics: the project objective, deliverables, requirements, and testing. Also, it will consist of diagrams and tables showing the activities to be completed by each team member.

The objective of this project is to design and implement a simulated robot that removes snow from a defined area without hitting any and all obstacles. The deliverables of this project are based on the objective that is needed to be done by the end of each week. The end result is a working robot that satisfies the project objectives accompanied by a report detailing the information related to the project and the work that went into it. The project requirements consist of a list of parameters that the robot should have that would deem the project valid. Testing includes the tests that will take place each week in order to make sure the activities completed work as promised.

The work breakdown structure consists of a detailed timeline in which it divides the workload among the team members. Each member should have an activity to complete each week that follows the guidelines set by the proposal. Schedule network diagram is a diagram that connects all the activities set out to be completed by each team member with respect to their timeslot. Gantt chart: is similar to the schedule network diagram in terms of showing the activities and when they should be completed with respect to time. However, it is more visual and is easy to follow on a weekly basis. Responsibility Assignment Matrix: is a matrix that shows each member responsible for completing an activity as well as approving another member’s activity. In conclusion, the proposal will help the team to stay on top of tasks as well as allow for a more agile work environment.

1. Project Charter

1.1 Project Objective

Our goal is to design and implement a simulated robot that removes snow from a defined area without hitting any and all obstacles.

1.2 Project Deliverables

In terms of deliverables for the project, we expect to complete a project proposal that summarizes our goals and outlines the breakdown of milestones that will be achieved during each week and by which members of the group. After a few weeks have passed and progress has been made on the project, we will create a progress report to update the observers of the project on where we will currently be in terms of the previously set milestones and overall project completion. Nearing the end of the term when the project is complete, we will have a finalized version of the snow plowing robot we will have created in CoppeliaSim as an exported model which will be accompanied by the necessary script files that allow the robot to function and meet the outlined requirements of the project. This will be the complete work and it will be consolidated into a final report that will encapsulate the work completed throughout the term and provide the project observers with the results and struggles of the model and any other important information relating to the process of developing the final product.

1.3 The Overall Architecture

As highlighted in the project document, we have designed a robot body that has a few different major sections when it comes to the functionality and efficiency of the robot. These components include the body, wheels, motors, joints, arm, plow, and sensors. To give an understanding of how these components works together to serve the desired purpose, we can first look at the dynamic pieces: the body consists of a rectangular prism shaped cuboid attached to four wheels and a motor for each wheel. These components combined allow the robot to move. Next is the plow, it has four long and thin plates connected with moving joints to allow it to extend once set in motion. This plow section is then connected to the body by an arm attached to a tall cuboid sitting on the center of the body. This arm also has a joint where it connects to the center piece which allows the plow and arm to move up and down. These moving parts give the robot the ability to reach its maximum stationary size limitations while extending further once moving to take advantage of a larger plow as allowed by the moving size restrictions. Our goal is to create this robot while keeping in mind that we want it to act in an **event triggered** manner. This will allow the robot to continue working on removing the snow until it encounters an obstacle or the edge of the border (significant events). As for the testing scenarios implemented for the robot, all scenarios have a success and fail criteria. Currently the robot passes the first four tests, where it is able to achieve the success criteria mentioned in section 2.3 regarding testing. First, the attachments are connected to the robot body and perform as intended during simulation. Then, the python connection over RemoteAPI is successful during simulation. Third, the robot components are created explicitly not to exceed the robot dimension requirements. Finally, the robot is able to move and turn properly through the python APIs provided by CoppeliaSim. This architecture is visualized in the following figure:

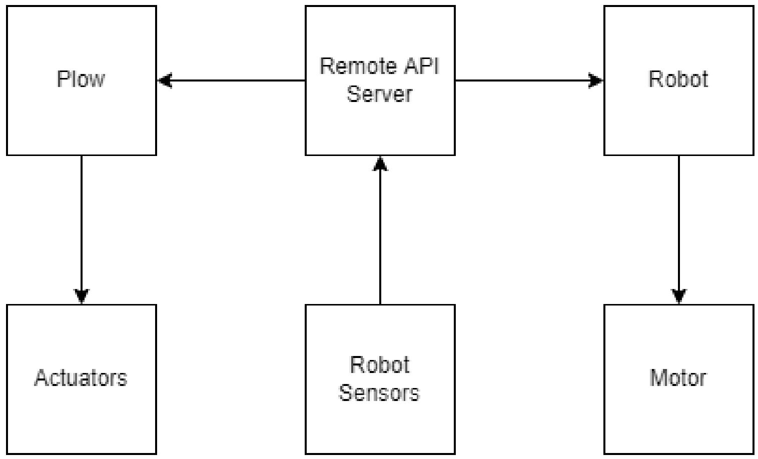


Figure 1: Architectural Diagram

The remote API server controls both the plow and the robot based on data collected by the sensors attached to the robot body. There are five actuators connected to the plow, which will extend the plow and move the plow up and down. There are four motors connected to the robot to control the movement of the robot.

1.4 State Chart of the Overall System

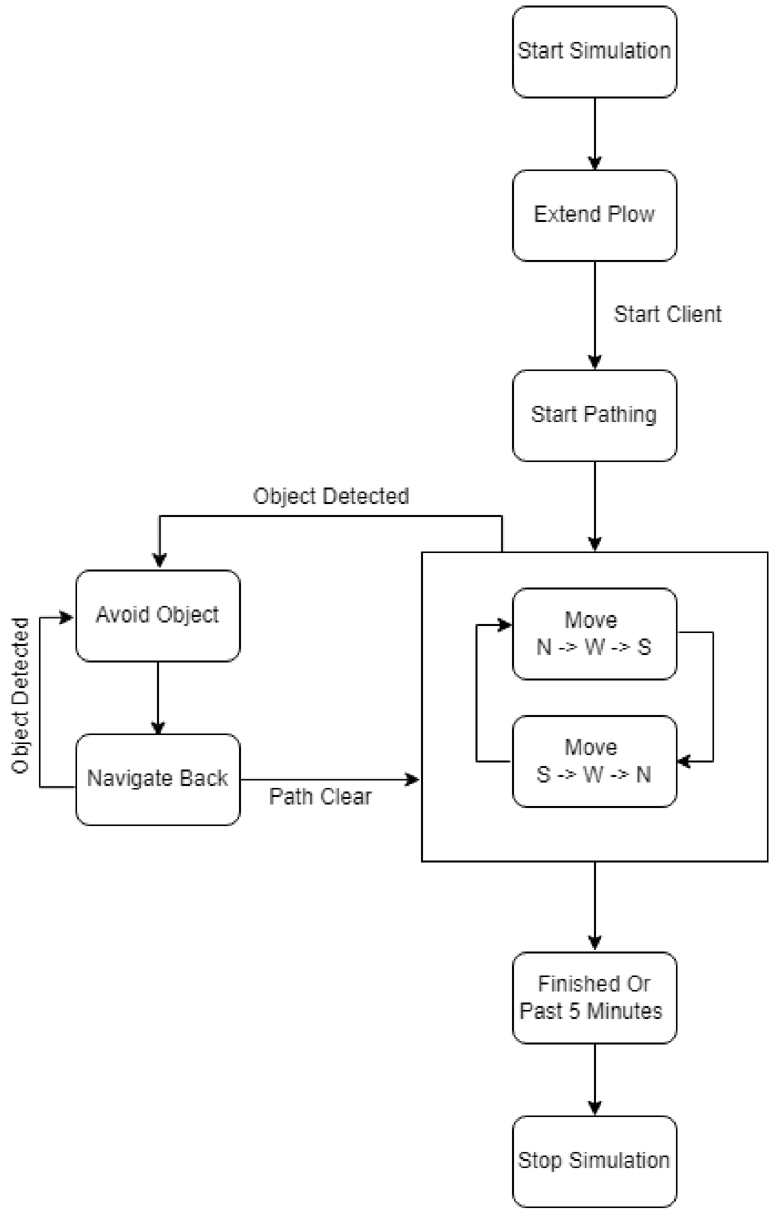
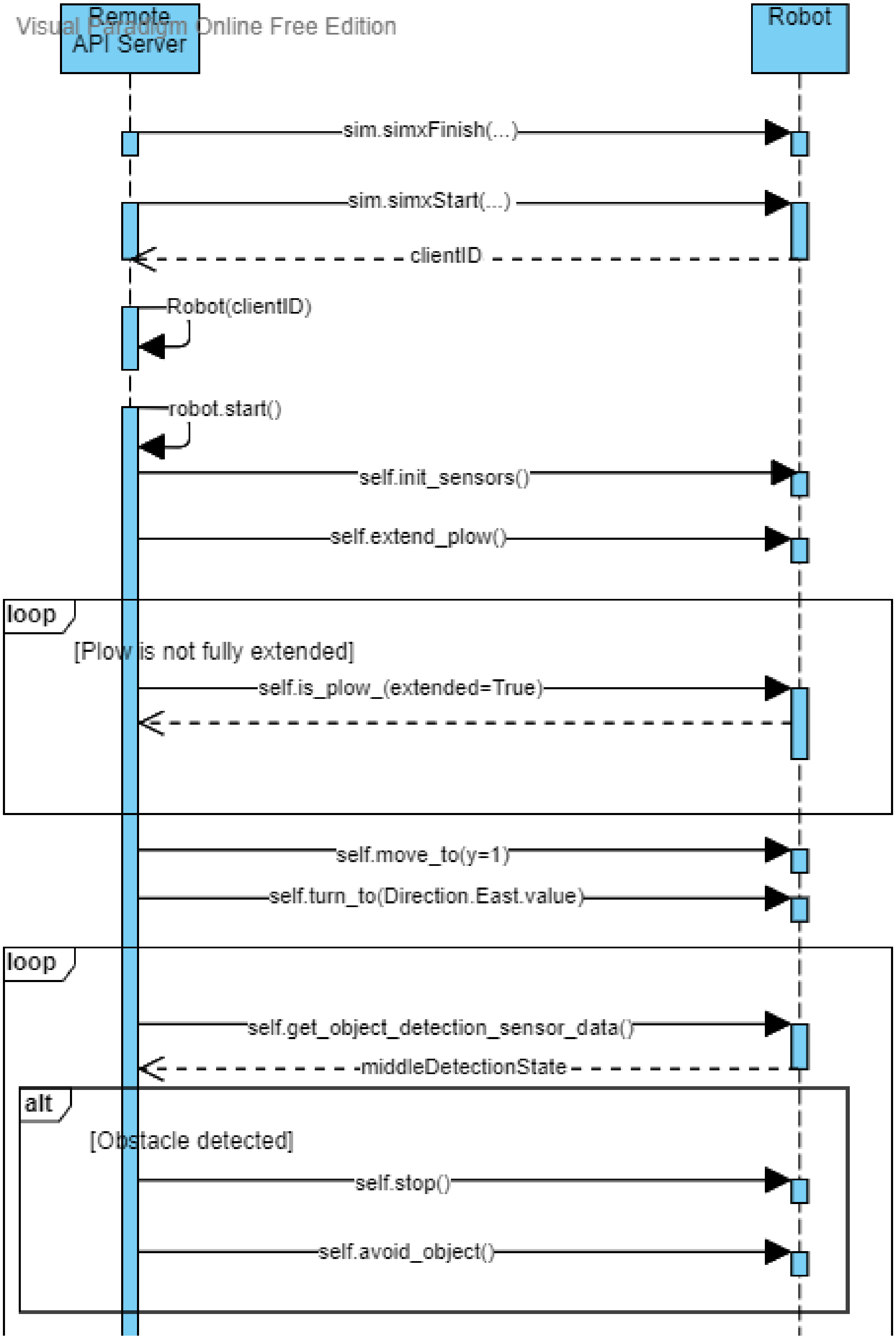


Figure 2: State Chart of the system

There are 4 states in total. One initial state, two recurring states, and one final state. The initial state follows the pathing algorithm designed which is going North then East to start the pathing algorithm. Then the two recurring states come into play where it starts off as North to West to South, then South to West to North. This way the whole map can be covered in terms of the paths taken to remove all the snow. The final state is either when the time elapsed is 5 minutes or if the direction is West and the orthogonal sensor detected a black line. This means that the robot turned left to continue the pathing algorithm and encountered the left side of the perimeter, resulting in the finished state.

1.5 Sequence Diagram



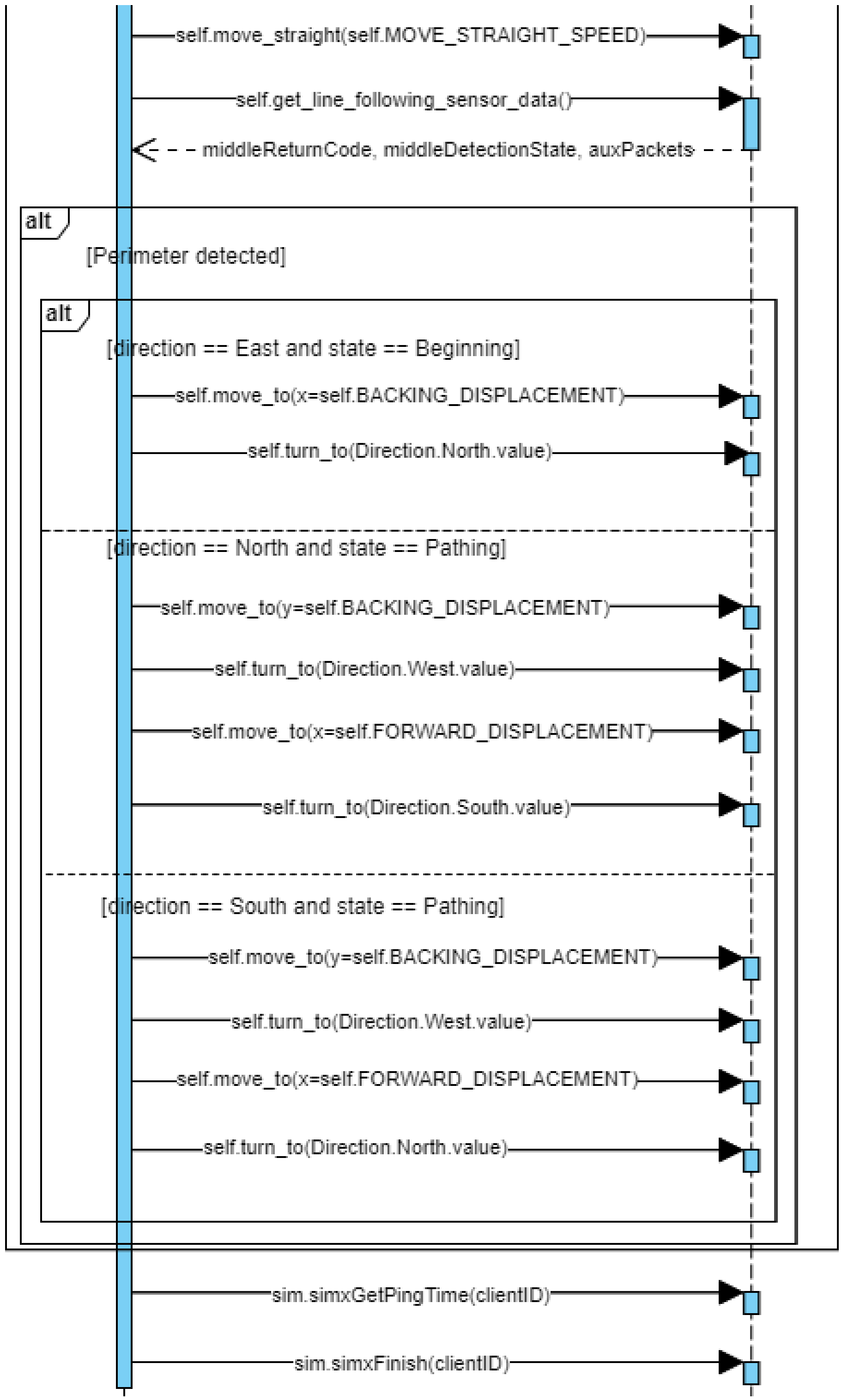


Figure 3: Sequence Diagram for the system

Steps:

1. **sim.simxFinish(...)** to close all opened connections
2. **sim.simxStart(...)** to connect to CoppeliaSim
3. **Robot(clientID)** to initialize the Robot control class
4. **robot.start()** to start to control the robot
5. **self.init_sensors()** to initialize all sensors attached to the robot
6. **self.extend_plow()** to fully extend the plow
7. **self.is_plow_(extended=True)** to check if the plow is fully extended in a loop
8. **self.move_to(y=1)** to move 1 meter forward in north direction
9. **self.turn_to(Direction.East.value)** to turn to east direction
10. To control the robot in a loop:
    1. **self.get_object_detection_sensor_data()** to get proximity sensors data
    2. If obstacle is detected:
        1. **self.stop()** to stop the robot
        2. **self.avoid_object()** to take a path that avoids the obstacle
    3. **self.move_straight(self.MOVE_STRAIGHT_SPEED)** to move at a certain speed.
    4. **self.get_line_following_sensor_data()** to get orthogonal sensor data
    5. If the perimeter is detected:
        1. If current direction is east and the state is beginning:
            1. **self.move_to(x=self.BACKING_DISPLACEMENT)** to back up a certain displacement in x (east/west) direction
            2. **self.turn_to(Direction.North.value)** to turn north direction
        2. If current direction is north and the state is pathing:
            1. **self.move_to(y=self.BACKING_DISPLACEMENT)** to back up a certain displacement in y (north/south) direction
            2. **self.turn_to(Direction.West.value)** to turn west direction
            3. **self.move_to(x=self.FORWARD_DISPLACEMENT)** to move forward a certain displacement
            4. **self.turn_to(Direction.South.value)** to turn south direction
        3. If current direction is south and the state is pathing:
            1. **self.move_to(y=self.BACKING_DISPLACEMENT)** to back up a certain displacement in y (north/south) direction
            2. **self.turn_to(Direction.West.value)** to turn west direction
            3. **self.move_to(x=self.FORWARD_DISPLACEMENT)** to move forward a certain displacement
            4. **self.turn_to(Direction.North.value)** to turn north direction
11. **sim.simxGetPingTime(clientID)** to ensure the last command has arrived before the remote API server ends
12. **sim.simxFinish(clientID)** to close the connection to CoppeliaSim

2. Scope

2.1 List of Requirements

Customer Requirements:

- The robot should be able to clear all snow spheres off of a 12m x 12m perimeter track
- The robot should clear the snow in less than 5 minutes
- The robot should not hit any obstacles while it is clearing the snow
- The robot must be smaller than 0.5m x 0.8m x 1.0m when parked
- The robot must be smaller than 1.0m x 0.8m x 1.0m at all times
- The speed of the robot should not exceed 2 m/s
- The simulation should be able to run in CoppeliaSim with additional Python scripts
- **The robot is event-triggered**

2.2 Work Breakdown Structure

Activity	Description
1	Robot Body Selection: Select and decide on an appropriate robot body to use
2	Robot Sensor Selection: Select and decide on appropriate sensors to use
3	Plow Controller Selection: Select and decide on appropriate actuators to use
4	Plow Design: Design the plow (Shape, size, operation)
5	Plow and Body integration: Integrate the plow on the robot body
6	Sensor and Body integration: Integrate the sensors on the robot body
7	Matching sensor specifications to datasheets from real sensors
8	Matching controller specifications to datasheets from real controller
9	Motor Control via Python: Control the motors using Python
10	Plow Control via Python: Control the plow using Python
11	Sensor Output to Python: Retrieve sensor data using Python
12	Controller Output to Python: Retrieve controller data using Python
13	Robot Plowing Behavior Initial Training: To extend and retract the plow
14	Robot Moving Behavior Initial Training: To move the robot in the perimeter

15	Robot Obstacle Avoidance Initial Training: To avoid the robot colliding the obstacles
16	Robot Perimeter Detection Initial Training: To detect the black boundary of the perimeter
17	Robot Behavior Training in training map 1: Train the robot in Map 1
18	Robot Behavior Training in training map 2: Train the robot in Map 2
19	Robot Behavior Training in training map 3: Train the robot in Map 3
20	Robot Behavior Improvement: To make the movement of the robot better
21	Improved Robot Behavior Training in training map 1: Train the updated robot in Map 1
22	Improved Robot Behavior Training in training map 2: Train updated the robot in Map 1
23	Improved Robot Behavior Training in training map 3: Train the updated robot in Map 1
24	Robot Behavior Adjustment: To adjust the movement of the robot based on the training results
25	Final Report Preparation: To prepare the final report
26	Robot Improvement: To make the robot look better and run better
27	Robot Testing: To test the overall robot behavior
28	Final Project Demonstration Preparation: To prepare the final demonstration

2.3 Testing

Test	Description	Success/Fail Criteria
1	Robot and Attachments test	Success: Attachments are connected to the robot body properly Fail: Attachments are not connected to the robot body properly (ex. They fall off when the simulation starts)
2	Robot and Python connection test	Success: Console prints “Connected Through RemoteAPI server” Fail: Connection was not established
3	Robot Dimension test	Success: The body parts are measured to satisfy the project requirements Fail: The body parts exceed the project requirement measurements (ex. During simulation mode the plow extends outside of the specified requirement)
4	Robot Movement test	Success: The robot can move straight, left, right, and back

		normally without exceeding the maximum speed. Fail: The robot cannot move as intended.
5	Plow shoveling test	Success: The plow is able to carry the snow outside of the perimeter Fail: The plow doesn’t interact with the snowballs properly
6	Obstacle avoidance test	Success: The robot successfully avoids the obstacles ahead Fail: The robot is not able to detect obstacles appropriately and avoid them
7	Map testing	Success: The robot can pass all test maps and clear all snow spheres off. Fail: The robot cannot clear all snow spheres off on all test maps.

3. Schedule

3.1 Schedule Network Diagram

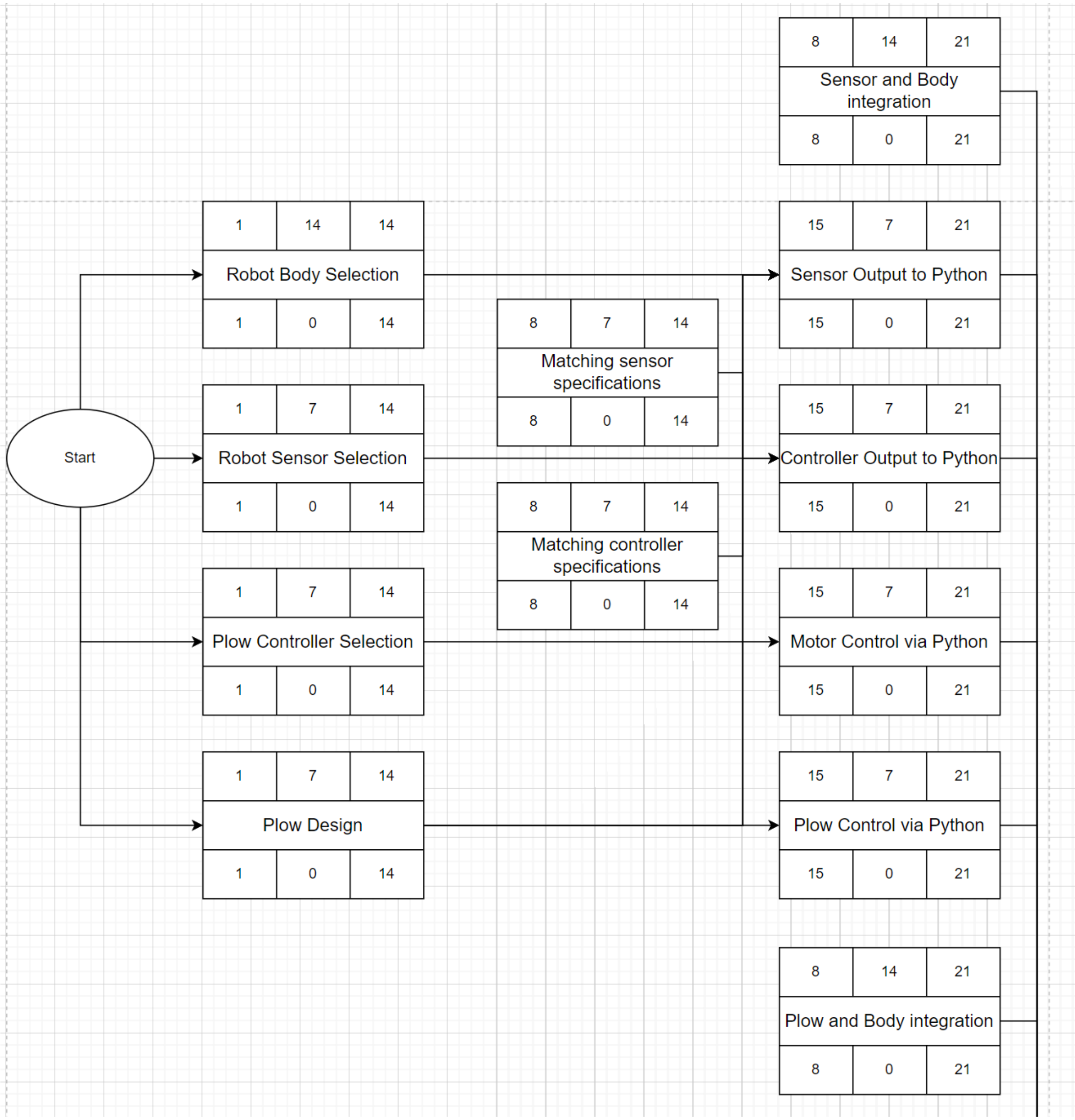


Figure 4: Part 1 of the Schedule Network Diagram

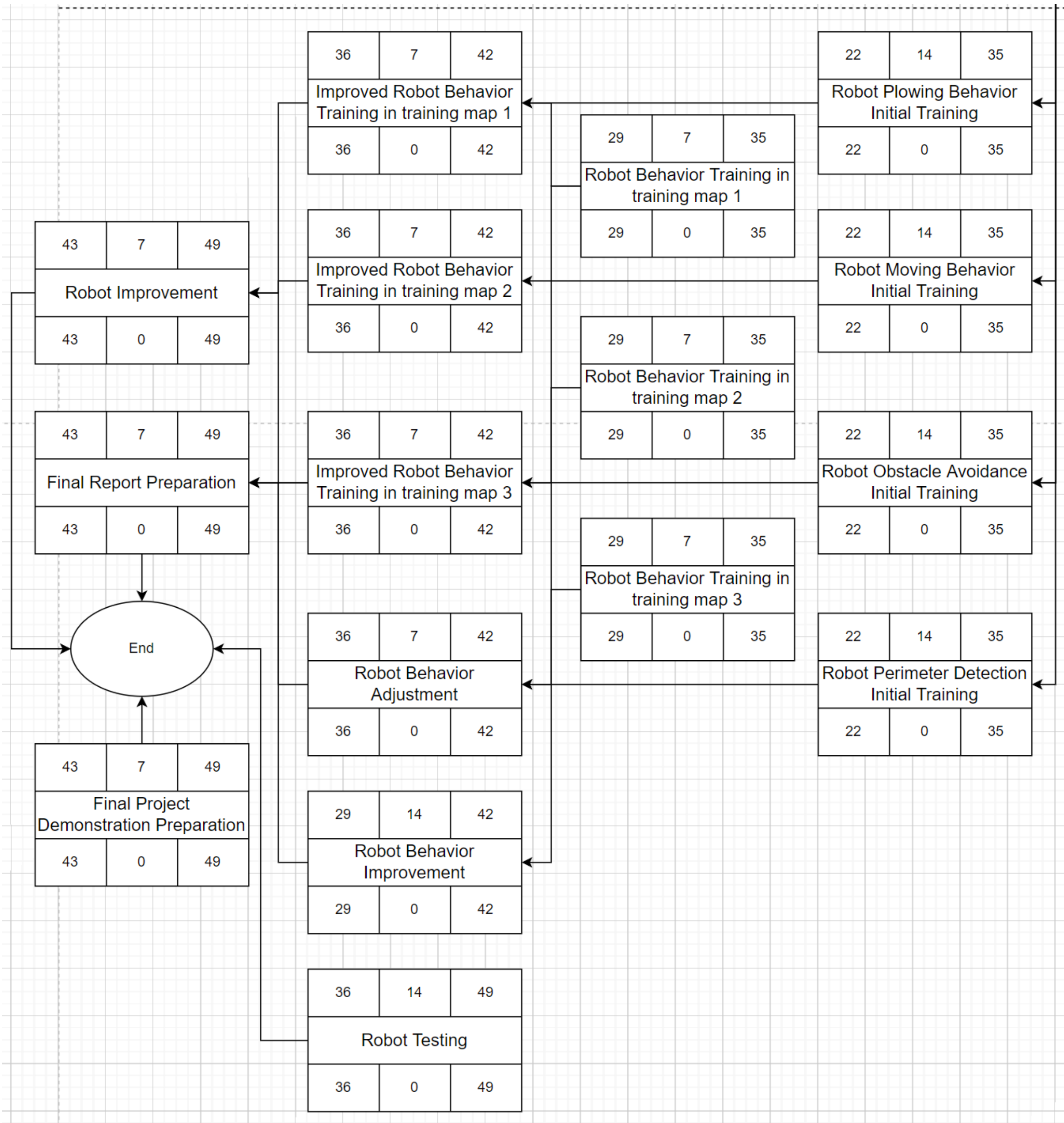


Figure 5: Part 2 of the Schedule Network Diagram

3.2 Gantt Chart

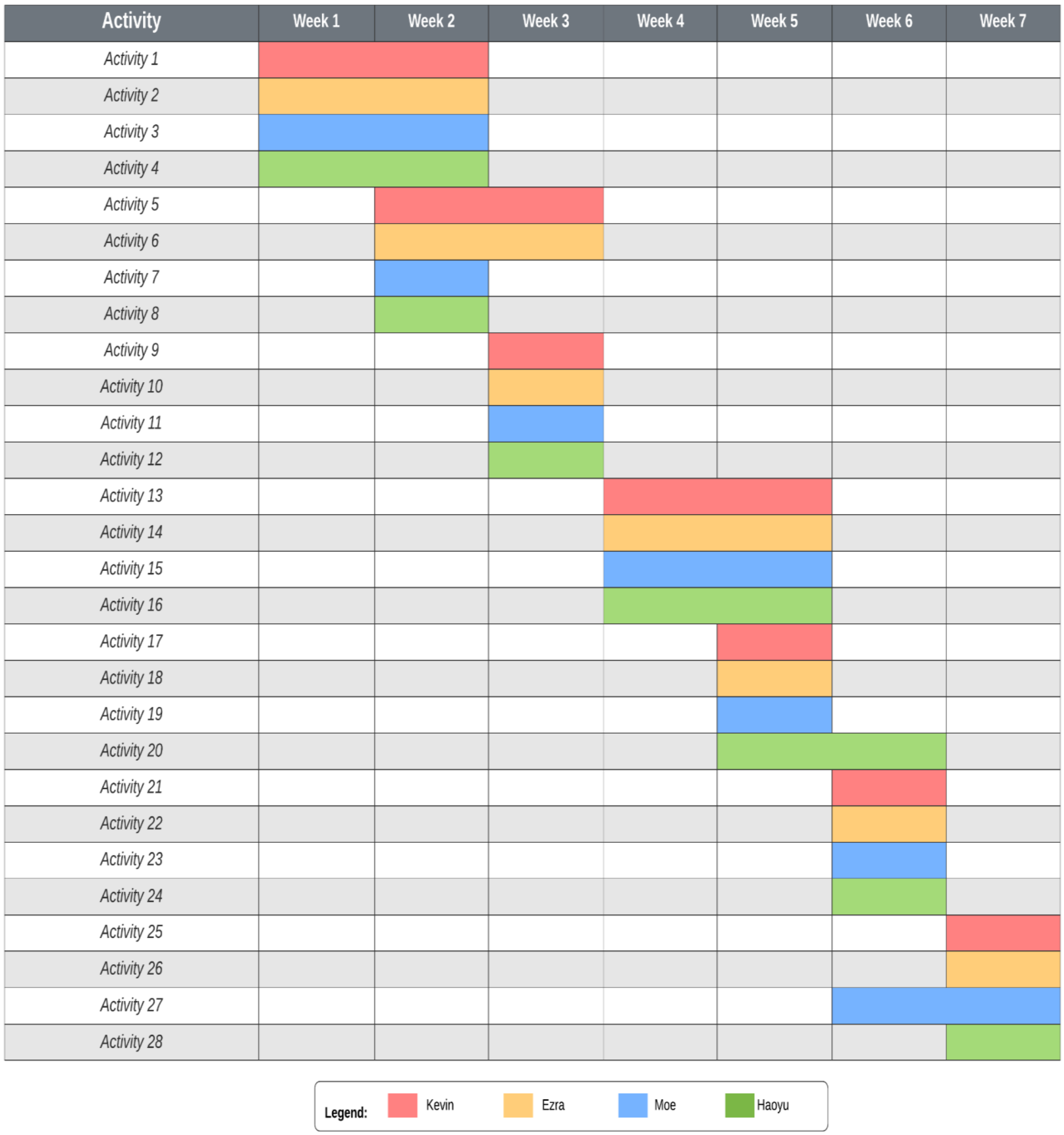


Figure 6: Gantt Chart of the system

4. Human Resources

4.1 Responsibility Assignment Matrix

*R - Responsible (Work on), Q - Quality Reviewer

Activity	Kevin	Ezra	Moe	Haoyu
1	R	Q		
2		R	Q	
3			R	Q
4	Q			R
5	R	Q		
6		R	Q	
7			R	Q
8	Q			R
9	R	Q		
10		R	Q	
11			R	Q
12	Q			R
13	R	Q		
14		R	Q	
15			R	Q
16	Q			R
17	R	Q		
18		R	Q	
19			R	Q
20	Q			R
21	R	Q		
22		R	Q	

23			R	Q
24	Q			R
25	R	Q		
26		R	Q	
27			R	Q
28	Q			R

5. Project Budget

5.1 Budget at Completion (BAC)

To determine the planned value of our project, we use the following table:

Component	Quantity	Price	Total
Orthogonal Sensor	3 units	$3.50/unit	$10.50
Proximity Sensor	3 units	$15.87/unit	$47.61
Plow Actuators	5 units	$129.99/unit	$649.95
Laser Sensor	2 units	$17.76/unit	$35.52
Labor	112 hours	$24.5/hour	$2744.00
Total			$3487.58

5.2 Planned Value (PV)

To determine the planned value of our project, we use the following table:

Completed Activities	Planned Activities	Planned Value
28	28	$3487.58

6. Team Member Contributions

6.1 Team Member 1 – Kevin Belanger

Over the course of the term, I contributed to the project by helping in design choices of the robot body and implementing the requirements from the project document. I also took on a team leader role by coordinating extra meetings for the group outside of the normal times as well as formatting and submitting all the documents for the project.

6.2 Team Member 2 – Ezra Pierce

During this project I led the sensor selection and placement process. Including sourcing real parts from distributors to match the CoppeliaSim sensors. I also implemented the code to read from the sensors during simulation and integrated the sensors into the custom robot body.

6.3 Team Member 3 – Mohammad Issa

In this project, I designed the plow controller and the forklift like mechanism. Then implementing a pathing algorithm that is suitable for any map and satisfies the requirements of this project. Also, I worked on the object avoidance algorithm to help the robot abode obstacles and continue on the original route.

6.4 Team Member 4 – Haoyu Xu

In this project, I mainly designed the plow and the body of the robot along with the code to control the plow and the movement of the robot.

7. Working Code

The code is available at https://github.com/SYSC4805-Winter2022/sysc4805-project-group-9-l1-little-boy-blue
Explanation for each folder/file:

- robot: Final robot model and the code submitted on Brightspace
- robot_improved: The robot that was intended to be used but due to implementation issues and unsatisfied testing results, the robot and the code were obsoleted.
- video: The videos for testing the robot in three test maps
- week1: The progress we achieved in week 1.
- week2: The progress we achieved in week 2.
- week3: The progress we achieved in week 3.
- week4: The progress we achieved in week 4.
- week5: The progress we achieved in week 5.
- week6: The progress we achieved in week 6.
- .gitignore: The file that tells git to ignore some files
- Map2.ttt: Training map 2 with our robot model.
- Map3.ttt: Training map 3 with our robot model.
- Map4.ttt: Training map 4 with our robot model.
- SYSC 4805 - Final Report.pdf: The final report.
- SYSC 4805 - Progress Report.pdf: The progress report.
- Training_Map_1.ttt: Training map 1 **without** our robot model. The one with our robot model is under week6 folder.

8. Results

8.1 Training Results

Below is an overview of the results we found on the listed maps:

Map	Total Snow	Cleared Snow	Time	Hits
Training_Map_1	366	72	02:10	0
Map2	460	15	00:45	0
Map3	492	55	01:20	1
Map4	488	80	02:36	0

8.2 Training Results

In this section, we take a look at different functionalities of the robot and multiple trials of their time to perform such actions. This is summarized in the following figures:

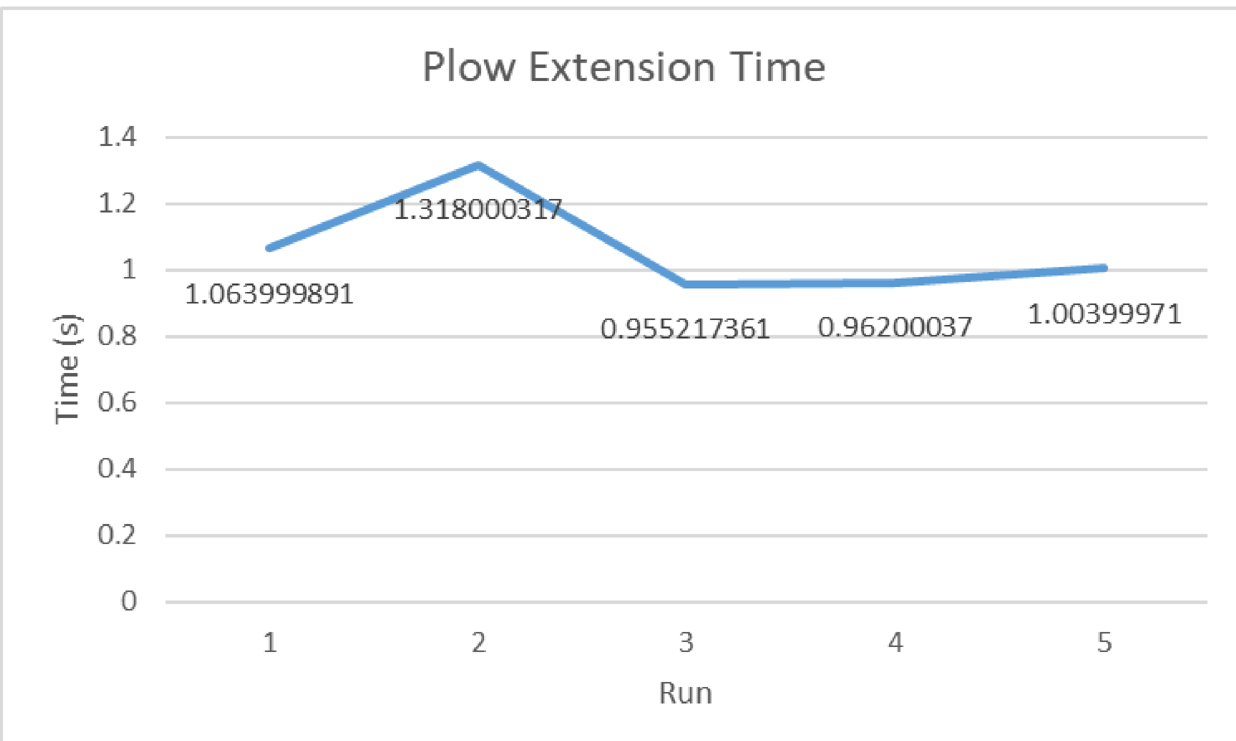


Figure 7: Time for the robot to extend the plow from a retracted state to an extended state

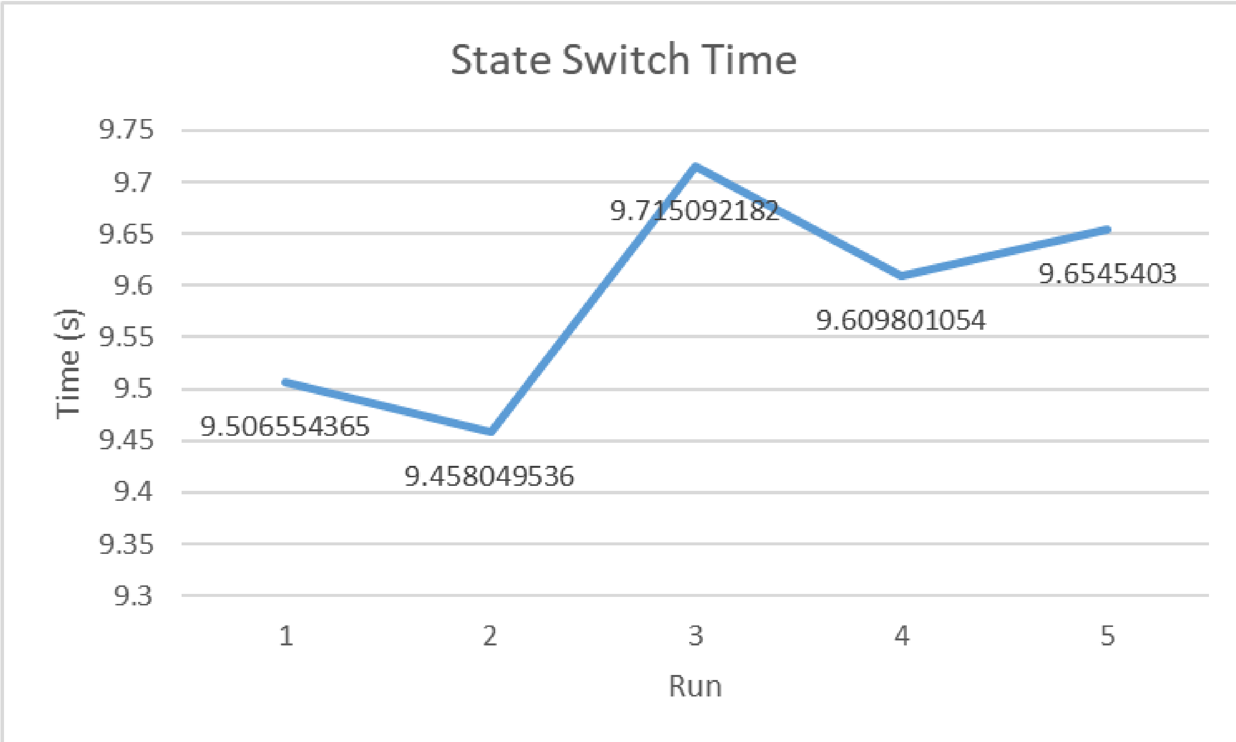


Figure 8: Time for the robot to switch from the initial state to the pathing state

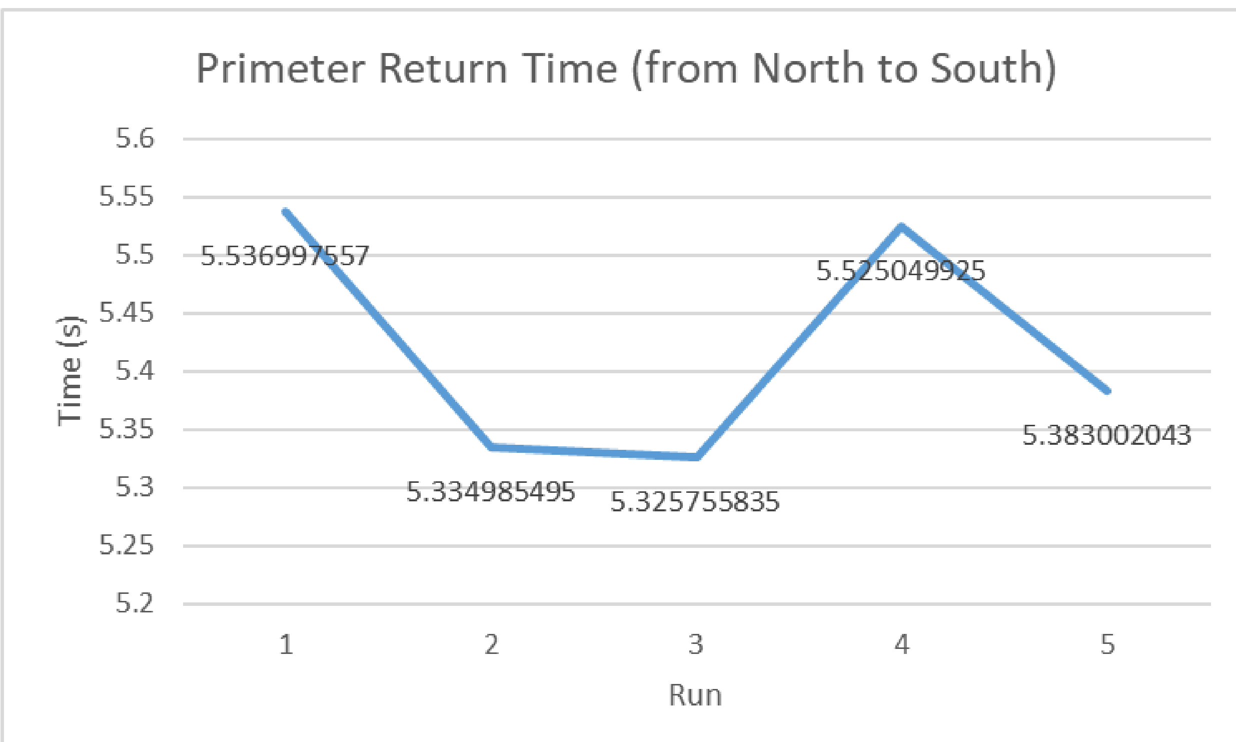


Figure 9: Time for the robot to return to the perimeter when its path is north

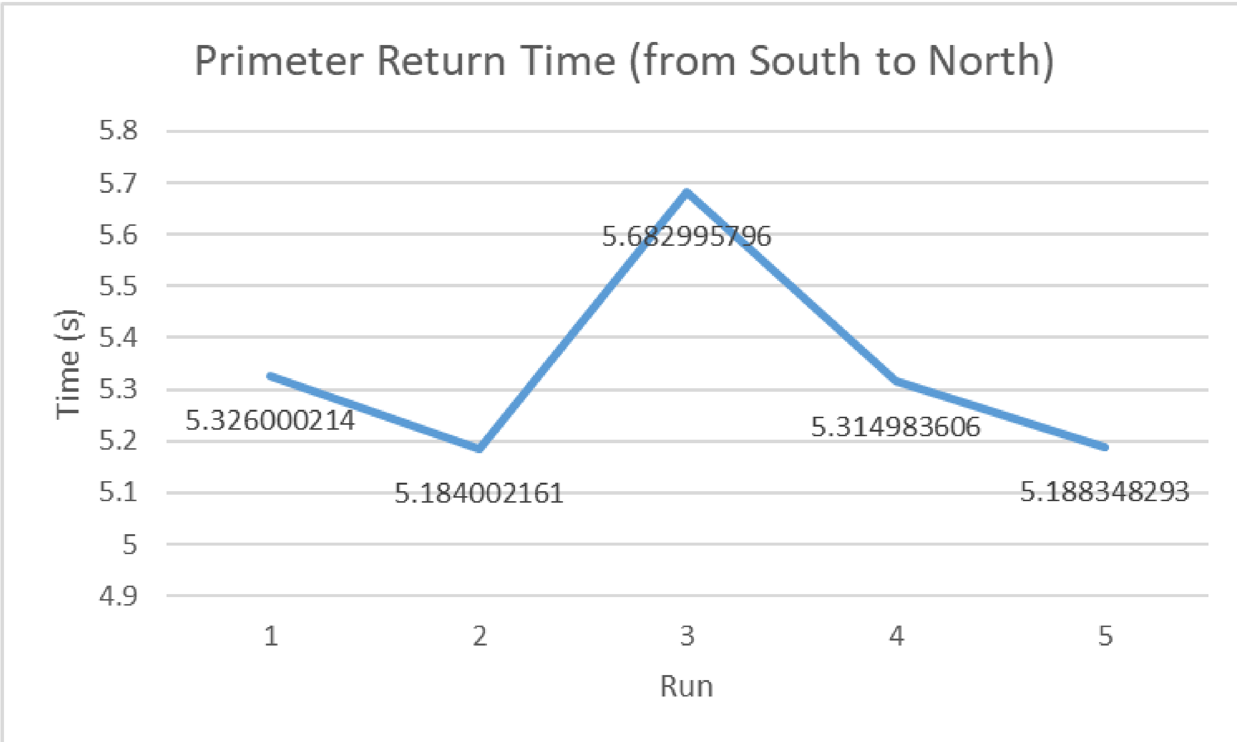


Figure 10: Time for the robot to return to the perimeter when its path is south

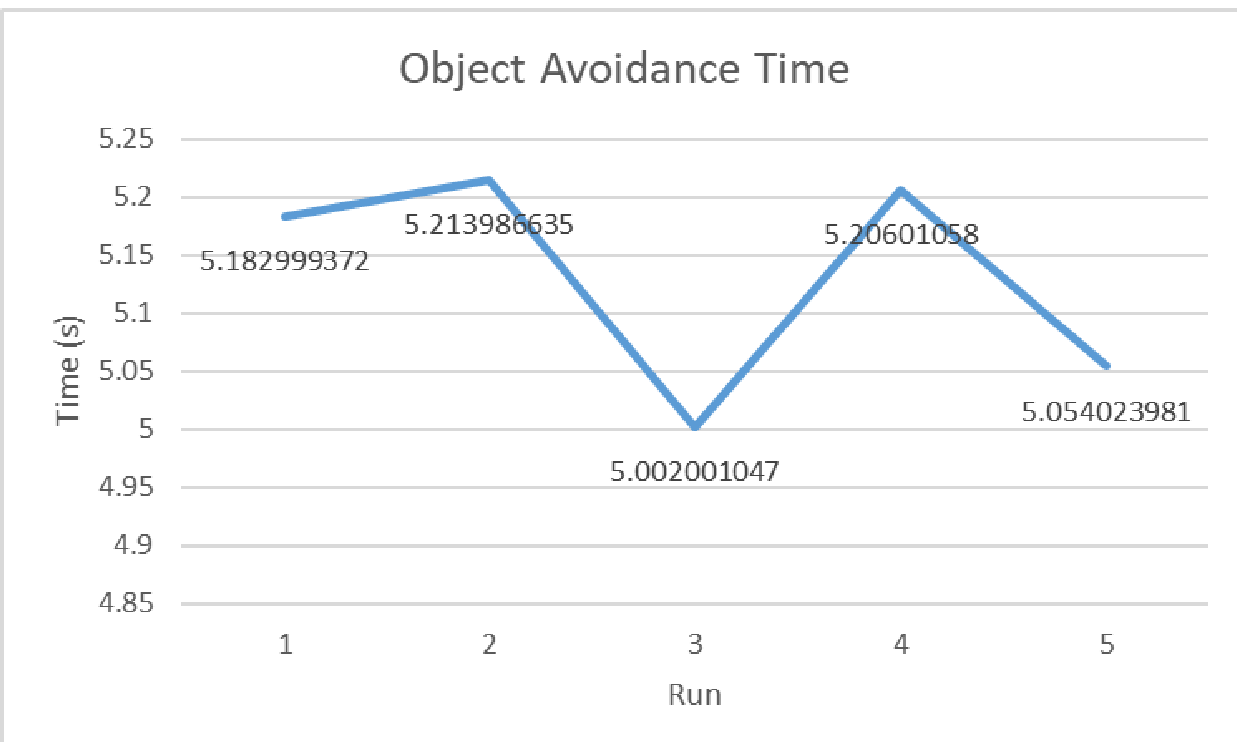


Figure 11: Time for the robot to avoid the first obstacle in Training_Map_1

Figure 12: Time for the robot to finish Training_Map_1

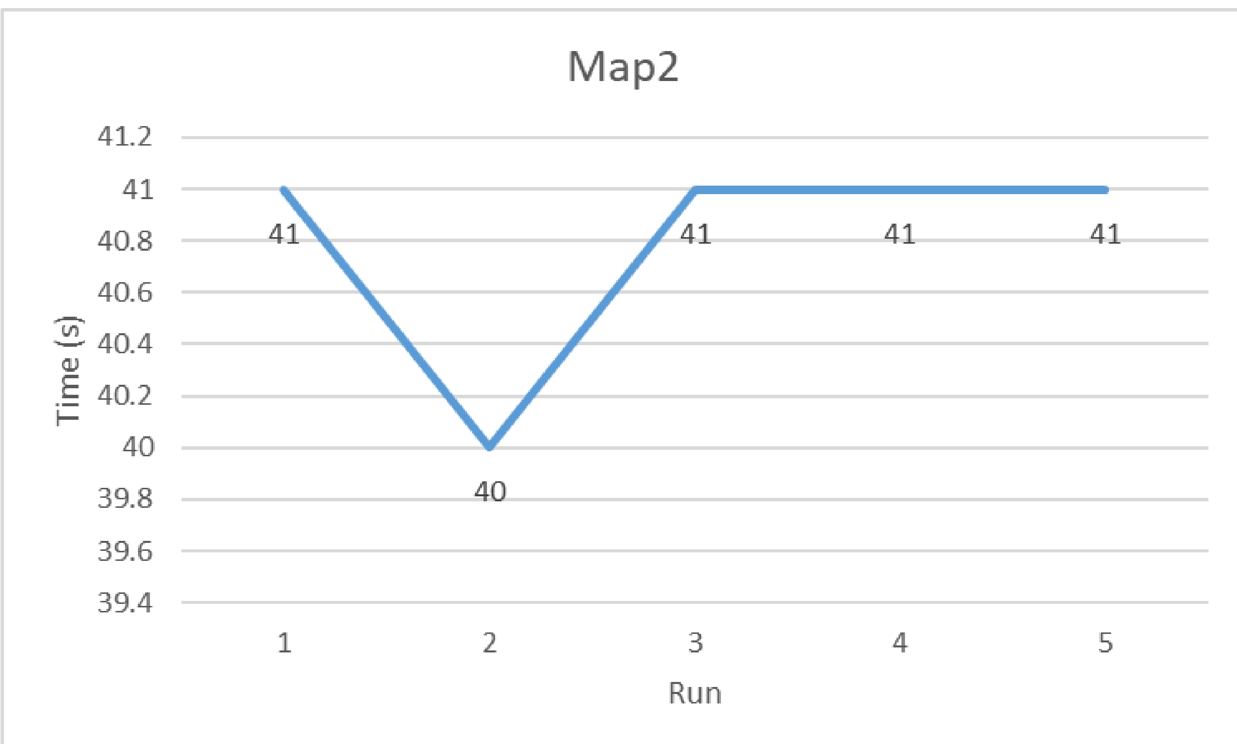


Figure 13: Time for the robot to finish Map 2

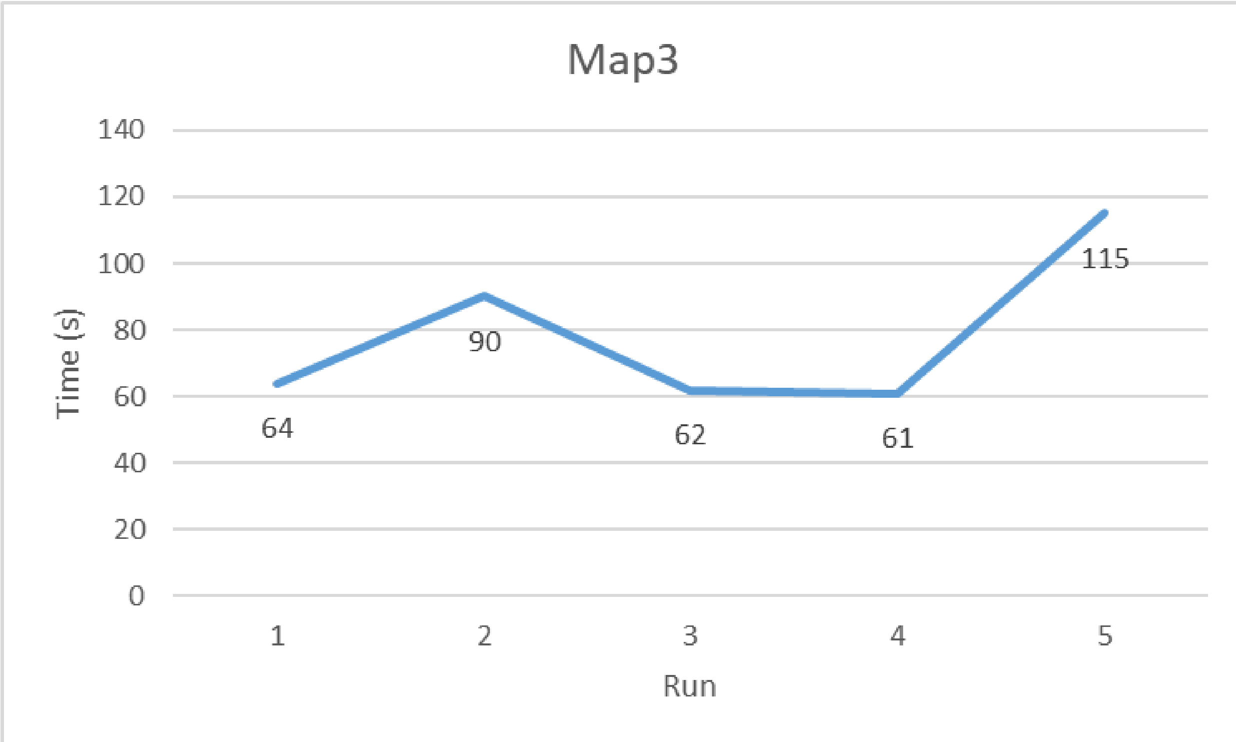


Figure 14: Time for the robot to finish Map 3

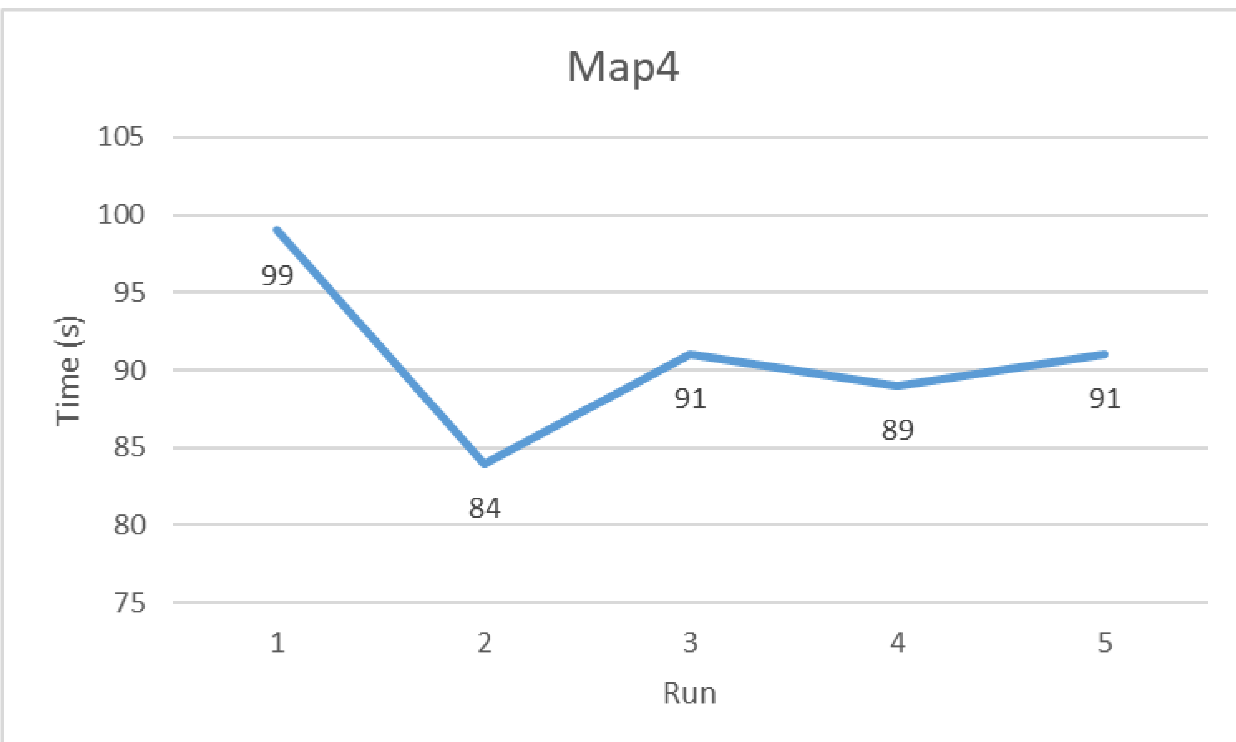


Figure 15: Time for the robot to finish Map 4

9. References

[1] “CH101-00ABR,” *DigiKey*. [Online]. Available: https://www.digikey.ca/en/products/detail/CH101-00ABR/1428-CH101-00ABRCT-ND/13174319?WT.z_cid=ref_netcomponents_dkc_buynow&utm_source=netcomponents&utm_medium=aggregator&utm_campaign=buynow. [Accessed: 01-Mar-2022].

[2] AI8Z, M. #619676, M. #665168, Plexi, M. #673009, and Skye, “SparkFun Redbot sensor - line follower,” *SEN-11769 - SparkFun Electronics*. [Online]. Available: https://www.sparkfun.com/products/11769. [Accessed: 01-Mar-2022].

[3] ServoCity. 2022. 4" Stroke 115 lb Thrust Heavy Duty Linear Actuator. [online] Available at: <https://www.servocity.com/4-stroke-115-lb-thrust-heavy-duty-linear-actuator/> [Accessed 1 March 2022].

[4] ServoCity. 2022. 10" Stroke 115 lb Thrust Heavy Duty Linear Actuator. [online] Available at: <https://www.servocity.com/10-stroke-115-lb-thrust-heavy-duty-linear-actuator/> [Accessed 1 March 2022].

[5] Coppeliarobotics.com. 2022. *Remote API functions (Python)*. [online] Available at: <https://www.coppeliarobotics.com/helpFiles/en/remoteApiFunctionsPython.htm> [Accessed 1 March 2022].

Appendix A: Project Proposal

Carleton University

Project Proposal

Title Page

<u>Computer Systems Design Lab</u>

-

Kevin Belanger (101121709) L1

Ezra Pierce (100991590) L1

Mohammad Issa (101065045) L1

Haoyu Xu (101088272) L1

Course: SYSC 4805

Team: Little Boy Blue

Date Submitted: February 4th, 2022

Table of Contents

Summary

The objective of this project is to design, build, and simulate a robot that clears snow from an enclosed area with random obstacles it must avoid. This will be done using CoppeliaSim with the various tools learned in the previous labs. The purpose of this proposal is to plan and divide different activities to accomplish the goals of the project within the given 7-week period as a team. Working in a team-based environment will allow us to come up with a plan that is efficient and effective to develop the robot by the deadline.

CoppeliaSim is a robot simulator that is used for fast algorithm development, factory automation simulations, and robotics related education which allow us to complete our project. Our goal, by the time the project is completed, is to have designed and implemented a simulated robot that removes snow from a defined area without hitting any obstacles. The deliverables of the project are outlined further in this proposal and are based on the objectives set by the team by completing smaller components each week leading up to the culmination of these components in the final report and demonstration where information of the work that was done will be detailed.

The project requirements are listed below in the proposal which consists of a list of specifications that the robot should meet that would deem the project valid in the eyes of the customer. The work that will be done as a team is then outlined in terms of activities which are also assigned numbers in a general order that they will be completed in. Each member of the team will have an activity to complete each week to stay on track to the overall progress of the project. Each of these activities will be tested through the list of tests further in the proposal to ensure the project and its components function on their own and as a whole.

The activities previously mentioned are condensed into a schedule network diagram that connects all the activities set to be completed by each team member and the order in which they will be worked on. Following this is a Gantt chart which is a visual representation of the schedule network diagram, however it shows what activity will be worked on, by who, and which week. Lastly, there is a responsibility matrix formed to ensure every activity has at minimum one team member working on it and a different team member reviewing the quality of their work.

In conclusion, the proposal will help the team to stay on top of tasks by scheduling and assigning work while also allowing for a more agile work environment.

1. Project Charter

1.1 Project Objective

Our goal is to design and implement a simulated robot that removes snow from a defined area without hitting any and all obstacles.

1.2 Project Deliverables

In terms of deliverables for the project, we expect to complete a project proposal that summarizes our goals and outlines the breakdown of milestones that will be achieved during each week and by which members of the group. After a few weeks have passed and progress has been made on the project, we will create a progress report to update the observers of the project on where we will currently be in terms of the previously set milestones and overall project completion. Nearing the end of the term when the project is complete, we will have a finalized version of the snow plowing robot we will have created in CoppeliaSim as an exported model which will be accompanied by the necessary script files that allow the robot to function and meet the outlined requirements of the project. This will be the complete work and it will be consolidated into a final report that will encapsulate the work completed throughout the term and provide the project observers with the results and struggles of the model and any other important information relating to the process of developing the final product.

2. Scope

2.1 List of Customer Requirements

The following is a list of all the requirements of the customer listed in the project description:

- The robot should be able to clear all snow spheres off of a 12m x 12m perimeter track
- The robot should clear the snow in less than 5 minutes
- The robot should not hit any obstacles while it is clearing the snow
- The robot must be smaller than 0.5m x 0.8m x 1.0m when parked
- The robot must be smaller than 1.0m x 0.8m x 1.0m at all times
- The speed of the robot should not exceed 2 m/s
- The simulation should be able to run in CoppeliaSim with additional Python scripts

2.2 Work Breakdown Structure

The following table is a list of all the activities that will performed over the course of the project:

Activity	Description
1	Robot Body Selection
2	Robot Sensor Selection
3	Plow Controller Selection
4	Plow Design
5	Plow and Body integration
6	Sensor and Body integration
7	Matching sensor specifications to datasheets from real sensors
8	Matching controller specifications to datasheets from real controller
9	Motor Control via Python
10	Plow Control via Python
11	Sensor Output to Python
12	Controller Output to Python
13	Robot Plowing Behavior Initial Training
14	Robot Moving Behavior Initial Training
15	Robot Obstacle Avoidance Initial Training
16	Robot Perimeter Detection Initial Training

17	Robot Behavior Training in training map 1
18	Robot Behavior Training in training map 2
19	Robot Behavior Training in training map 3
20	Robot Behavior Improvement
21	Improved Robot Behavior Training in training map 1
22	Improved Robot Behavior Training in training map 2
23	Improved Robot Behavior Training in training map 3
24	Robot Behavior Adjustment
25	Final Report Preparation
26	Robot Improvement
27	Robot Testing
28	Final Project Demonstration Preparation

Table 1: Work Breakdown Structure

2.3 Testing

The following figure lists the tests that will be performed at different stages of the project when necessary:

Test	Description
1	Robot and Attachments test
2	Robot and Python connection test
3	Robot Dimension test
4	Robot Speed test
5	Plow shoveling test
6	Obstacle avoidance test
7	Integration testing

Table 2: List of tests

3. Schedule

3.1 Schedule Network Diagram

Below is a figure that outlines the overall sequence of each activity previously mentioned for the project:

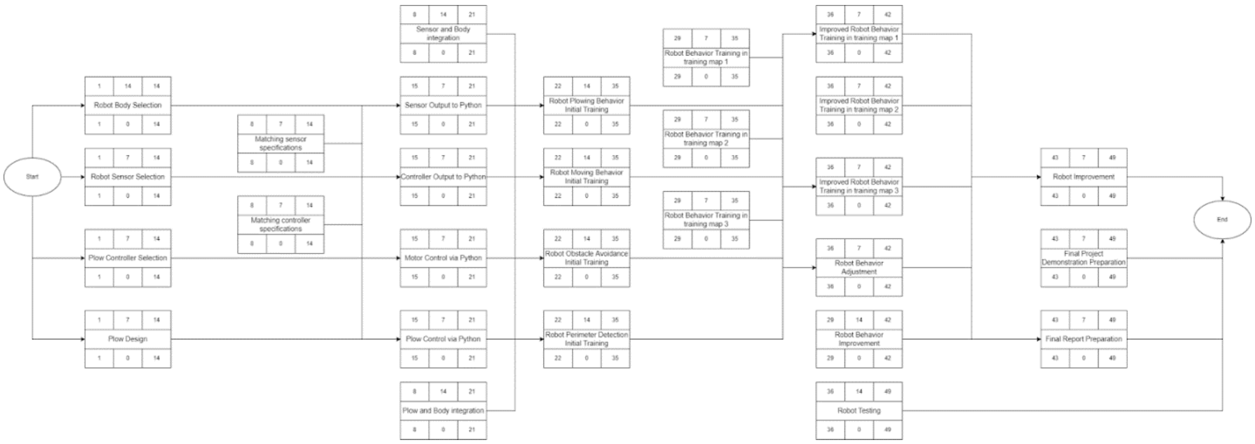


Figure 1: Schedule Network Diagram

3.2 Gantt Chart

Below is a figure that outlines the overall timeline, with its assigned team member, of each activity previously mentioned for the project:

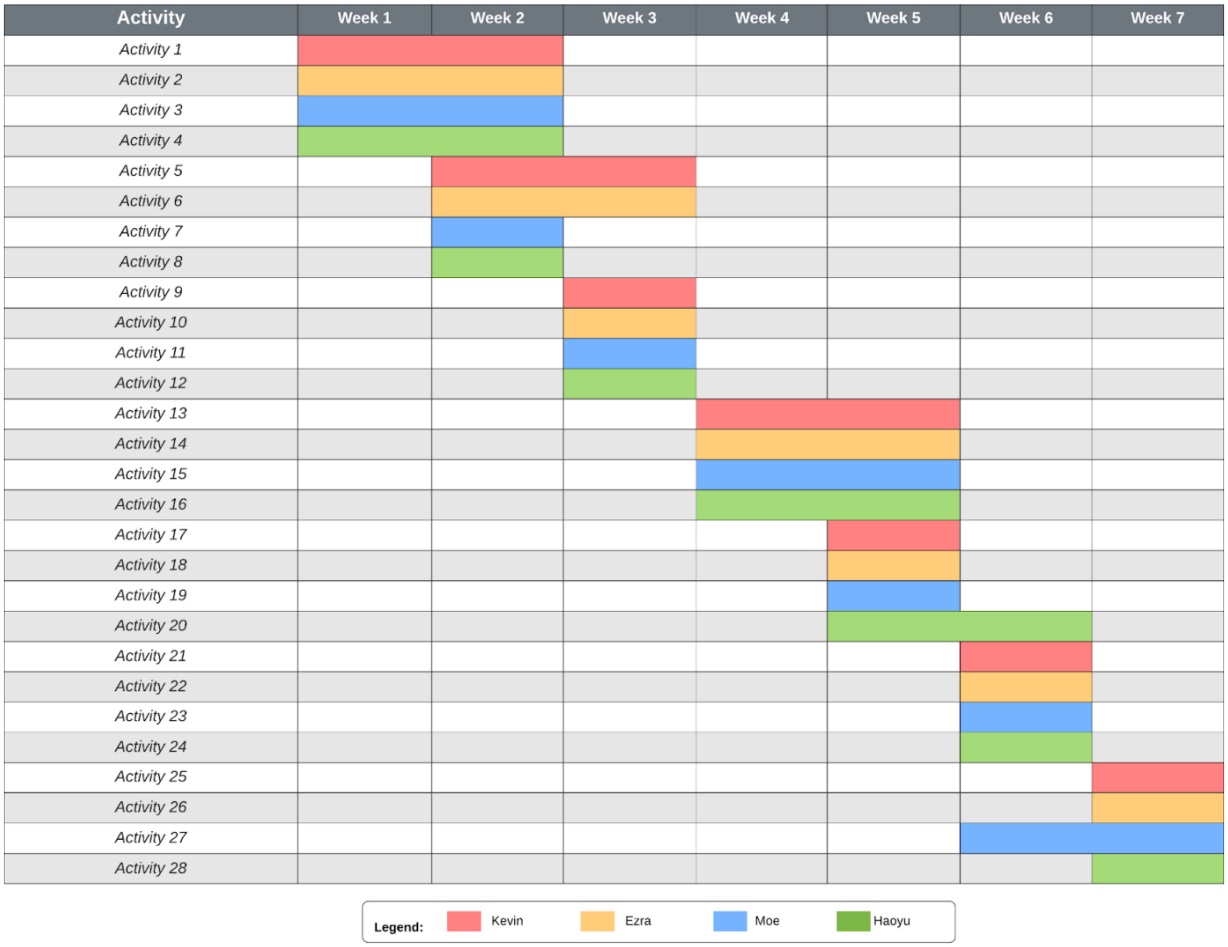


Figure 2: Gantt Chart outlining the timeline for the activities

4. Human Resources

4.1 Responsibility Assignment Matrix

The following table shows who will be working on each activity and who will reviewing this work:

*R - Responsible (Work on), Q - Quality Reviewer

Activity	Kevin	Ezra	Moe	Haoyu
1	R	Q		
2		R	Q	
3			R	Q
4	Q			R
5	R	Q		
6		R	Q	
7			R	Q
8	Q			R
9	R	Q		
10		R	Q	
11			R	Q
12	Q			R
13	R	Q		
14		R	Q	
15			R	Q
16	Q			R
17	R	Q		
18		R	Q	
19			R	Q
20	Q			R
21	R	Q		
22		R	Q	

23			R	Q
24	Q			R
25	R	Q		
26		R	Q	
27			R	Q
28	Q			R

Table 3: Responsibility Assignment Matrix