SYSC 4805: Computer Systems Design Lab

Project Proposal

Team Spiro Disco Ball

Group #5

Daniel Tura	*100995028*
David Casciano	*101069255*
Aedyn Ladd	*101068855*
Tadhg McDonald-Jensen	*101047069*

TABLE OF CONTENTS

1 Charter

In seven weeks, our group plans to have fully developed a system capable of navigating and removing snow cover from a predetermined path.

1.1 Overall Deliverables

Our final deliverables will include a robot capable of navigating a user-defined path while avoiding damage to obstacles found in its dynamically changing environment. This will be done using a variety of sensors and actuators available to us in the coppeliaSim simulation program and a full description of how the system was built and corresponding vendors of the used components.

The robot will interface with coppeliaSim through a script developed using Python. The python script and robot model will be delivered with detailed instructions indicating how to run the system, as well as with a final report indicating the process in which the robot was built and insights into the design.

The development of the model will be broken down into a series of 28 unique activities distributed evenly among group members that will dictate the progression of our development cycle as we move through the term. At the end of the term, our findings will be presented to our colleagues, teaching assistants, and professor in a formal presentation.

2. Scope

The system we are intending to create will be described fully in this section. This will include a set of strict requirements that we will follow while designing and building our robot, as well as details describing how we intend to accomplish the tasks at hand. Along with this, we will also include a breakdown of how the system will be tested - taking into account factors such as the provided simulation environments and unit testing for code.

2.1 Requirements

This section defines requirements set out by the project outline, as well as requirements elicited by our group with respect to how the system should function. The table below describes the various requirements our project will meet. It is important to note that seeing as these requirements are elicited from the supporting project documentation that the owner of all requirements is the professor.

The risk column is filled based on the following definition:

- Event Risks: A supplier may go out of business during the project.
- Non-Event Risks: Ambiguity in one requirement
- Emergent Risks: being unknowable unknowns

Table 1: *Elicitation of project requirements as described by the 'Project Description'*

Requirement		Rationale	Type	Priority	Risk
A1	While idle, the size of the robot will be constrained to a space of 0.5 x 0.8 x 1 meters.	Size constraints are required to ensure the robot's environment properly mimics real-life conditions. This also ensures no extraneous parts are used	Functional Requirement	High	Event
A2	While in motion the size of the robot will never exceed a space of 1 x 0.8 x 1 meter.		Functional Requirement	High	Event
B	The system will be able to remove snow from an area not exceeding 144 sq. m, within an allotted time frame of 5 minutes, so long as the volume of snow does not far exceed the height of the robot.	This requirement specifies a realistic zone in which our robot is effectively able to operate. Any increase in the area will result in a decreased performance.	Functional Requirement	High	Event
C	The velocity of the robot will be limited to a maximum speed of 2 m/s. This will be calculated as a function of the angular velocity of the joints, and the circumference of the wheels.	Speed restraints are required to ensure the robot is able to maintain a sense of realism.	Quality-in-use Requirement	High	Event
D	Maximum torque will be limited to realistic values supporting the components used in the assembly of the robot.	Torque limit is set to ensure realism in the simulated environment	Quality-in-use Requirement	Medium	Non-Event
E	The robot will be controlled in CoppeliaSim through the use of the external API via a script written in Python.	Python is a simple programming language with a solid foundation. It allows for further development if desired.	Interface Requirement	Medium	Non-Event
F	The system in question will avoid obstacles presented to itself so long as obstacles are not intentionally attempting to actively disrupt operations at a speed faster than the system is able to operate.	Robot should never cause financial or bodily harm. We also specify that any harm caused by misuse of the robot is not captured by this requirement.	Human Factor Requirements	High	Event

G	A list of feasible components as well as their specifications, cost, and availability recorded from technology vendors will be maintained throughout the course of the project	This requirement serves to ensure the realism of the robot through its development life cycle. By maintaining a list of components we can ensure that the final product can be delivered.	Quality-in-use Requirement	Medium	Emergent
H	The system will have the ability to operate in whatever environment it is placed in so long as it follows requirements specified by B.	The system has a range of flexibility with respect to where it can operate.	Non-Functional Requirement	High	Event
I	A repository with all documentation will be maintained through the project lifecycle	This is to ensure transparency in the development of our project.	Process Requirement	Medium	Emergent

The requirements mentioned will be mentioned in further sections through the use of their tagging letter found in the first column of the above table. All requirements identified here will be delivered by the end date of the project.

2.2 Work Breakdown Structure

To satisfy the requirements listed above in Section 2.1, the group has composed twenty-eight tasks to create the Snow Removal Robot. These tasks are displayed in a hierarchical manner below, categorized into Project Management, Robot Modeling Hardware, Software, and Fail-Safe.

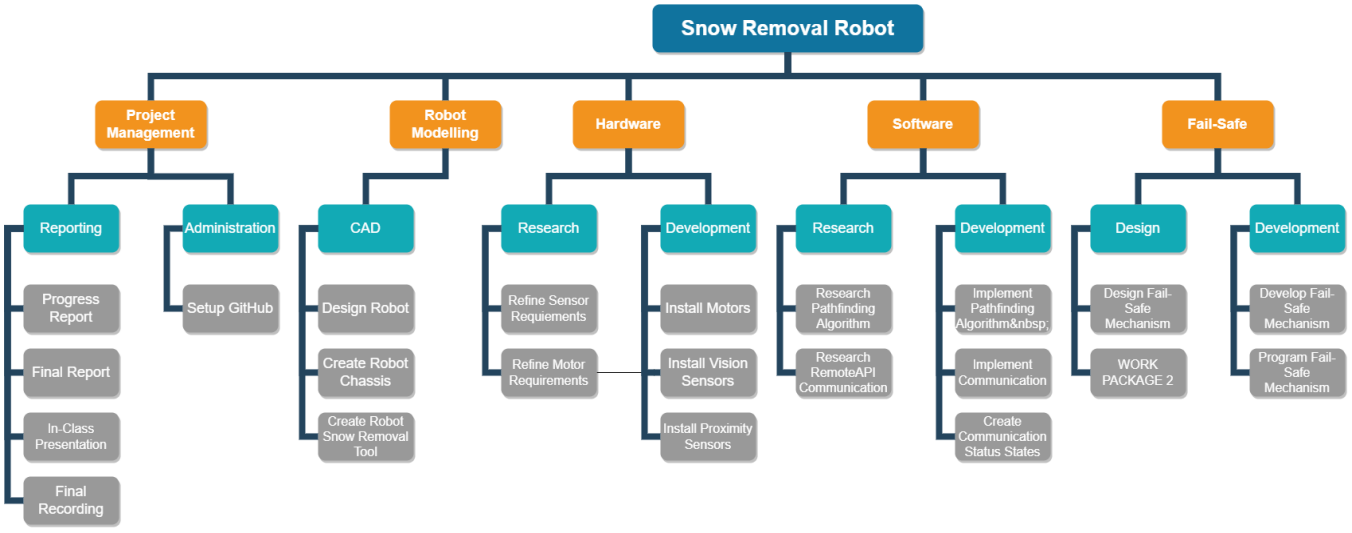


Figure 1: *A work breakdown structure of the activities set to be completed in the project*

Given Figure 1 on the previous page, Table 2 below comprises the tasks titles, descriptions, and requirements that it satisfies.

***Table 2**: A lookup table of task information*

Activity	Description	Satisfying Requirements
Draw Schematic	Create a rough draft of the robot with ideas of implementation	A1 and A2
Setup Github	Setup Github Repository, which includes the code base and Kanban boards	I
Refine Sensor Requirements	Identifying available sensors from real vendors	G and D
Refine Motor Requirements	Identifying available motors from real vendors	G, D and, C
Create Robot Body	Design robot in the coppeliaSim environment	A1, A2, and E
Research RemoteAPI Communication	Research available functions to be used through the RemoteAPI. These functions must be compatible with Python	I and E
Research Pathfinding Algorithm	Research pathfinding options to best clear the snow.	I and E
Design Fail-Safe Mode	Design a failsafe mode, which will allow the robot to fail gracefully in the case of a malfunction.	F, and H
Create Robot Snow Removal Tool	Implement a method for the robot to clear snow most effectively via CAD software.	A1, A2, and B
Install Motors	Implement physical components in CoppeliaSim and logical components in Python	G, A1, A2, C, and D
Install Vision Sensors	Implement physical components in CoppeliaSim and logical components in Python	G, A1, and A2
Install Proximity Sensors	Implement physical components in CoppeliaSim and logical components in Python	G, A1, and A2
Implement Pathfinding Algorithm	Program the algorithm in Python. This algorithm will communicate via RemoteAPI	I and E
Implement Communication via RemoteAPI	Program connection for bidirectional communication through RemoteAPI	I and E
Unit test sensors/vision	Create and perform tests to verify the correct function of the sensors	H

Unit Test Motors	Create and perform tests to verify the proper operation of the motors	H
Integration Testing	Test the complete robot with the test environment to identify and correct bugs	H

2.3 Testing

In order to ensure that the robot functions correctly and meets all the requirements listed in section 2.1 of this document, we will run a variety of tests on the system. The testing can be broken down as follows:

- *Unit Testing*
    - This will involve testing all the different components of the robot to ensure that on their own they are working properly. (ie. testing wheels and joints to ensure they have a proper range of motion)
    - This will also include testing the scripts we plan on using to ensure that they meet all necessary cases.
- *Integration Testing*
    - This will involve running the robot through our own environments to ensure that proper communication amongst parts is maintained throughout the simulation.
    - This involves activities such as moving around obstacles and collecting and removing snow from an area.
- *Environment Testing*
    - The final stage of testing will involve running the system in the training environments and ensuring all requirements with regards to time and completeness of the given task are completed successfully.

Once all the testing described above is complete, we will be able to say with certainty that our robot will be able to function correctly in any future testing environments with some level of success.

3. Schedule

The project is divided into thirty-three activities spread to ten weeks. The network diagram and Gantt chart identify activities that are float and critical (blocking). The work is fairly distributed equally to each member and we are confident that the project will be delivered to stakeholders by week ten.

3.1 Schedule Network Diagram:

The figure below shows the order and time analysis of each deliverable. Critical path analysis looks for the earliest and latest points at which tasks can begin and end. The digits on top of each box represent the est/lst and the arrows represent the steps of the tasks.

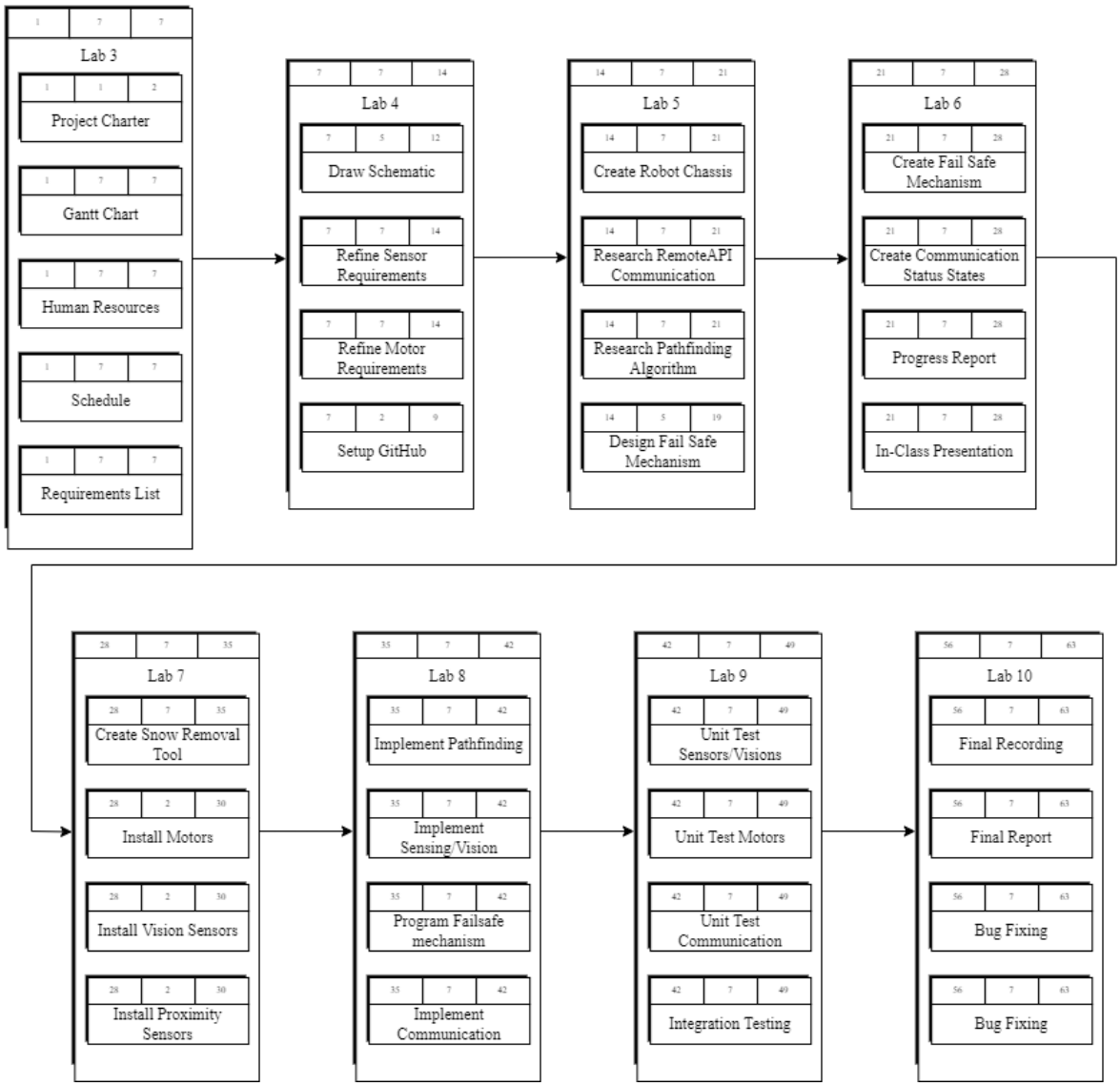
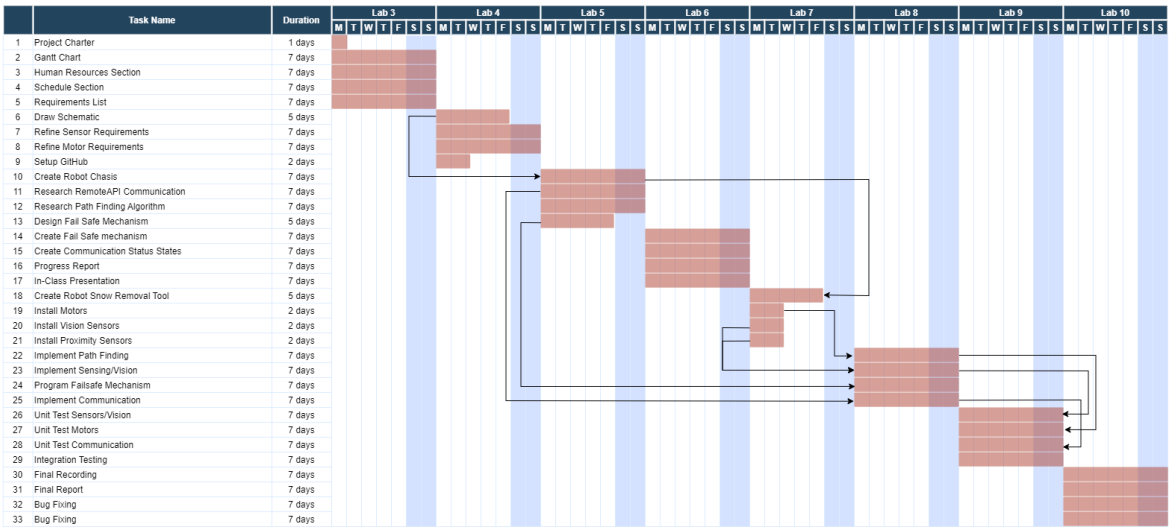


Figure 2: *Schedule diagram of the project deliverables*

3.2 Gantt Chart:

The Gantt chart is composed of three components. Activities are listed under the task name column, the number of days planned to complete the tasks are listed under the duration column, and the horizontal bar under labs is used to demonstrate tasks concurrency. In Addition, the order (prerequisites) of tasks are represented by using the arrows leaving and pointing to the horizontal bars.



	Task Name	Duration
1	Project Charter	1 days
2	Gantt Chart	7 days
3	Human Resources Section	7 days
4	Schedule Section	7 days
5	Requirements List	7 days
6	Draw Schematic	5 days
7	Refine Sensor Requirements	7 days
8	Refine Motor Requirements	7 days
9	Setup GitHub	2 days
10	Create Robot Chasis	7 days
11	Research RemoteAPI Communication	7 days
12	Research Path Finding Algorithm	7 days
13	Design Fail Safe Mechanism	5 days
14	Create Fail Safe mechanism	7 days
15	Create Communication Status States	7 days
16	Progress Report	7 days
17	In-Class Presentation	7 days
18	Create Robot Snow Removal Tool	5 days
19	Install Motors	2 days
20	Install Vision Sensors	2 days
21	Install Proximity Sensors	2 days
22	Implement Path Finding	7 days
23	Implement Sensing/Vision	7 days
24	Program Failsafe Mechanism	7 days
25	Implement Communication	7 days
26	Unit Test Sensors/Vision	7 days
27	Unit Test Motors	7 days
28	Unit Test Communication	7 days
29	Integration Testing	7 days
30	Final Recording	7 days
31	Final Report	7 days
32	Bug Fixing	7 days
33	Bug Fixing	7 days

Figure 3: *Gantt chart of activities duration and order*

4. Human Resources:

Each member plays a different role to complete the project. For each deliverable, one member is responsible to implement the task and another member approves. The responsibility matrix table illustrates the accountability of members divided by activities.

4.1 Responsibility Matrix

Deliverables, names and responsibility assignments are components of the responsibility matrix. Deliverables are the list of tasks that are to be performed. Responsibility assignments are represented as R(responsible) and A (Approve).

Table 3*: Responsibility Matrix of team members*

Deliverables	Daniel	Aedyn	Tadhg	David
Project Statement		R		A
Gantt Chart		A		R
Human Resources	R		A	
Schedule	A		R	
Requirements List		R		A

Draw Schematic		R	A	
Refine Sensor Requirement	R			A
Refine Motor Requirement	A		R	
Setup Github		A		R
Create Robot		R		A
Research RemoteAPI Communication	R	A		
Research Path Finding Algorithm	A		R	
Design Fail Safe			A	R
Create Communication States	R			A
Progress Report Task		A	R	
Create Fail-Safe Mechanism	A			R
In-Class Presentation		R	A	
Create Robot Snow Removal Tool		R		A
Install Motors	A			R
Install Vision Sensors	R		A	
Install Proximity Sensors		A	R	
Implement Path Finding		R	A	
Implement Sensing/Vision			R	A
Implement Failsafe	A			R
Implement communication	R		A	
Unit test Sensors/Vision	A		R	

Unit test Motors	R			A
Unit test communication		A		R
Integration Testing		R		
Final Recording	R		A	
Final Report	A		R	
Bug Fixes		R		A
Bug Fixes		A		R

5. Architecture

When designing our robot, we started with several rough ideas on what our overall design should be based on the task at hand. While brainstorming several ideas came to light: a robot consisting of several smaller entities that would all work simultaneously to plow a larger area at once, a robot with a large storage system that instead of directing plowing the snow would store it and then dump snow in a designated area, we even considered a robot with a set of omni-directional wheels allowing it to have a full 360 degrees of motion at a whim. We decided that for this specific project - simple was better - and that our robot needed three main components to effectively meet all the requirements specified in the course outline:

1) The robot needs a way to navigate from location A to location B.
2) The robot needs a set of sensors to interact with and manage the state of its environment
3) The robot needs a plow that is able to move in various directions as situations evolve over time.

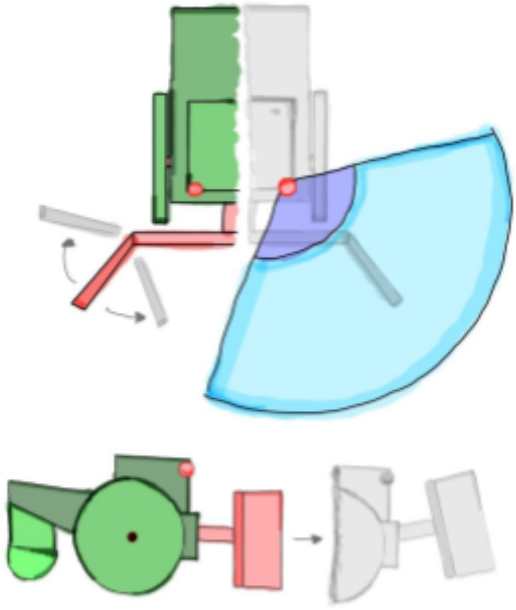


Figure 4: Initial Architectural Concept

The navigation system is a simple set of 2 wheels used to control where the robot is able to go, along with a castor wheel which acts as a third point of contact for additional stability. The robot also makes use of a series of proximity sensors allowing it to react to its environment. The proximity sensors are housed on the head of the robot and provide a large range of vision on the area directly in front of and to the left and right of the robot. Along with this we also have a series of vision sensors on the bottom of the robot that allow our system to identify when it has left (or is bordering on) the designated “clean

up" zone. As identified above, most of the sensors are housed in a large box at the head of the robot. This is to provide a sense of the space required to house the components required to allow our robot to function, including the ability to add additional sensors as we require in the future if needed without worrying about the space required to do so. The head is raised above the level of the plow to ensure that the proximity sensors are able to monitor the surrounding environment without being blocked.

The proximity sensors work on a premise similar to that of tomography. They are attached to a swivel, which allows them to freely rotate around the head of the robot to produce projections of the environment around it based on the distance of objects located around it in 1 degree segments. Using basic trigonometry we could map a projection of the robots available space into a two dimensional array as a function of the object's distance provided by the sensor, and the current angle of the sensor.

Lastly - and most important to our robot - is the mechanism used to move snow across the map, a plow. The plow we designed is fairly simple and consists of 4 main components: An arm, a body, and 2 separate wings. Using the motor connected to the main arm, the entire plow is able to move freely up and down as situations permit it to do so. The center body of the plow is fixed to this arm and will always be facing forward, to maximize the space our robot is able to operate in. The center body is flanked with left and right wings that are able to move with around 220 degrees of freedom. This allows for more snow to be moved in a larger area, as well as provide the ability to micromanage the areas in which our robot is plowing and ensure no snowballs are lost while plowing.

The overall physical architecture of our system is rather simple, to show more personality and take ownership of our work we decided to add color and personality to the design, thus creating a system that resembles - very loosely - a frog. In the preceding section we will be discussing the internal architecture and processing mechanisms that our system makes use of.

5.1 Timer Triggered Framework

Given the requirements our system is required to be reactive to the state of the environment around it, a timer triggered framework was the most appropriate fit for our design. Every **X** seconds our system makes note of the environment it is in, this allows it to perform maneuvers and make adjustments as required. This was the optimal scenario for our robot to map out the region that it needs to plow, and proceed to clear it of snow. Along with this it also allows for a more constant decision making when approaching obstacles and an easier way for our robot to interact with the environment rather than react to it.

This relates to the real world as the act of driving and by extension plowing snow, is timer based. The expectation is that drivers maintain constant awareness of their environments by checking mirrors every 5-8 seconds and making small changes in anticipation of their environment, rather than in reaction to it.

6. System State Chart

The diagram depicted below provides readers with an explanation - in the simplest of terms - to how our system works as a whole, and how our design reacts and interacts with the various external stimuli. This mainly includes its ability to path find and remove snow from the specified region, as well as the avoidance of collisions with both stationary and dynamic obstacles in its environment.

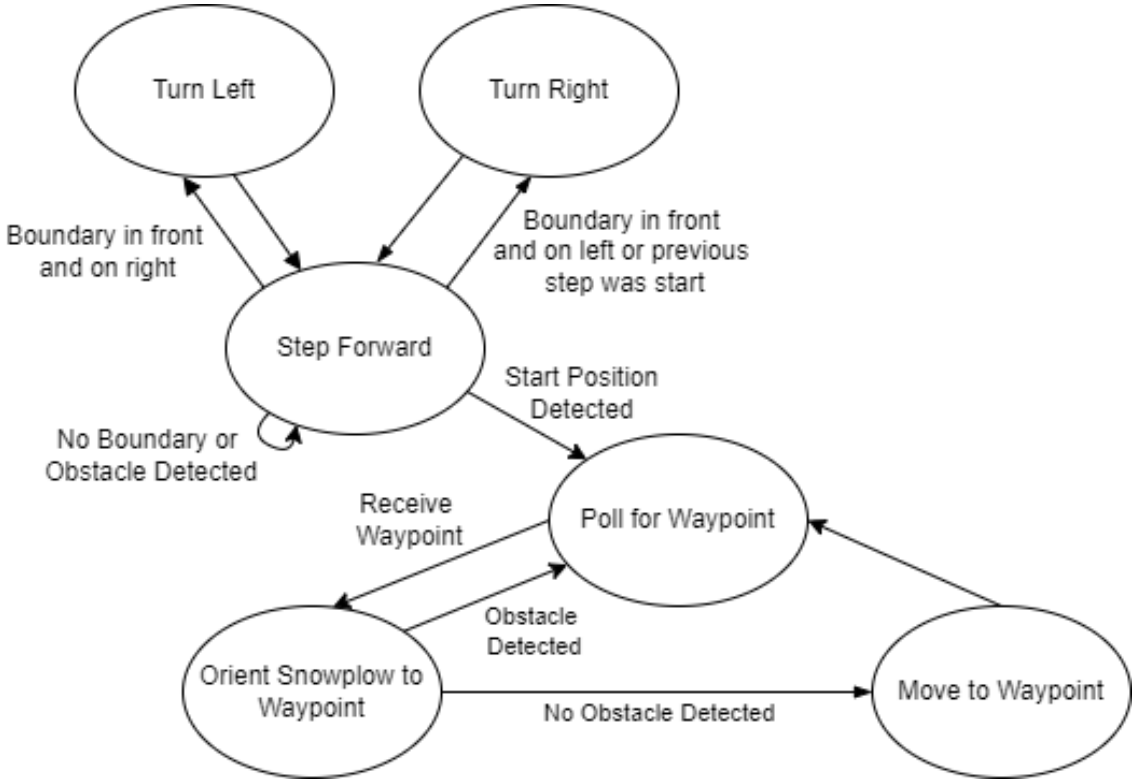


Figure 5: System State Chart Diagram

7. Sequence Diagram

On the following page is a sequence diagram dictating the communication flow of our system, as well as how various parts will interact and communicate with each other as based on the system state chart shown in figure 5 in the preceding section. Please note that due to the size of the sequence diagram and in the spirit of saving space, we are also including a larger version of it in Appendix A for the readers sake.

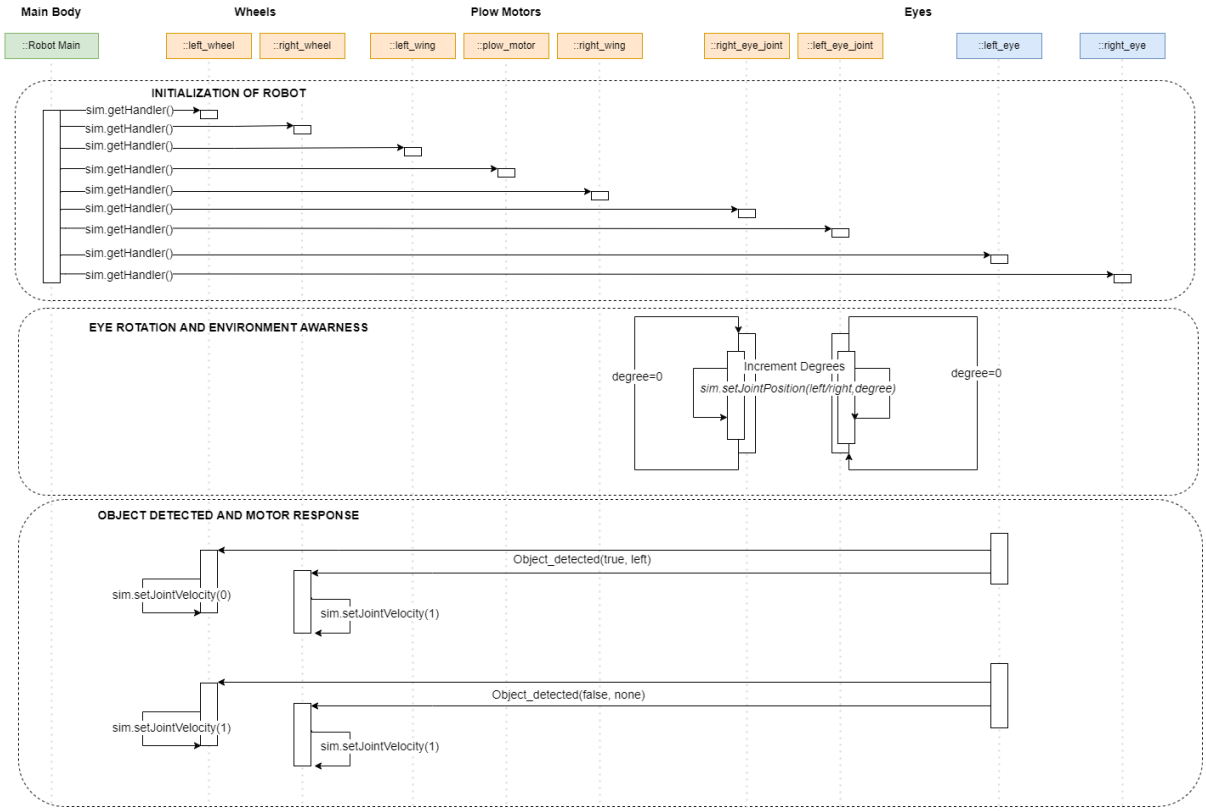


Figure 6: Sequence Diagram - see appendix for larger view

8. Project Budget

The following section explores the budget we have laid out for our project, including the time we have taken to design the product, as well as the cost associated with acquisition of the various necessary components and their vendors.

8.1 Planned Value

A proximity sensor will be used for the detection of both moving and static obstacles. For this, the group has chose the following requirements for the sensor:

- Detect object from a distance of a meter
- Detect both nonmetallic and metallic objects
- Wide beam angle
- Cost Effective

Due to this the team is using the CUSA-R75-18-2400-TH . This ultrasonic proximity sensor has a price of $5.08 per unit and satisfies the above requirements. The sensor has a sensing distance from 20 centimeters to 18 meters. As well the sensor has a ebam angle of 75 degrees.

Infrared sensors will be used for the line detection that is boundary sensing. The Sparkfun QRE1113 is a widely used infrared sensor within line following robots. The infrared sensor is $4.07 per unit. This sensor has a sensing distance of 1mm therefore will be placed at the bottom of the robot.

The motor the team has selected for the snow removal robot is the Sparkfun ROB-09238. This motor is widely used in mobile robots. As well this motor is $24.70 per unit. This motor was solected due to its sizing and availability.

8.2 Budget at Completion

Given the pricing of the above components the estimated budget at completion will be

Item	Price (CAD)	Quantity	Provider
ROB-09238	24.70	2	SparkFun
CUSA-R75-18-2400-TH	5.08	2	CUI Devices
QRE1113	4.07	4	Sparkfun
Total		75.84	

8.3 Cost Management Chart

Each group member planned to work on the project five hours a week. The labor rate is $30 Canadian dollar per hour totaling to $600 CAD each week for ten weeks . Based on the work schedule, we are on time however we completed some of the activities less than the budgeted hours.

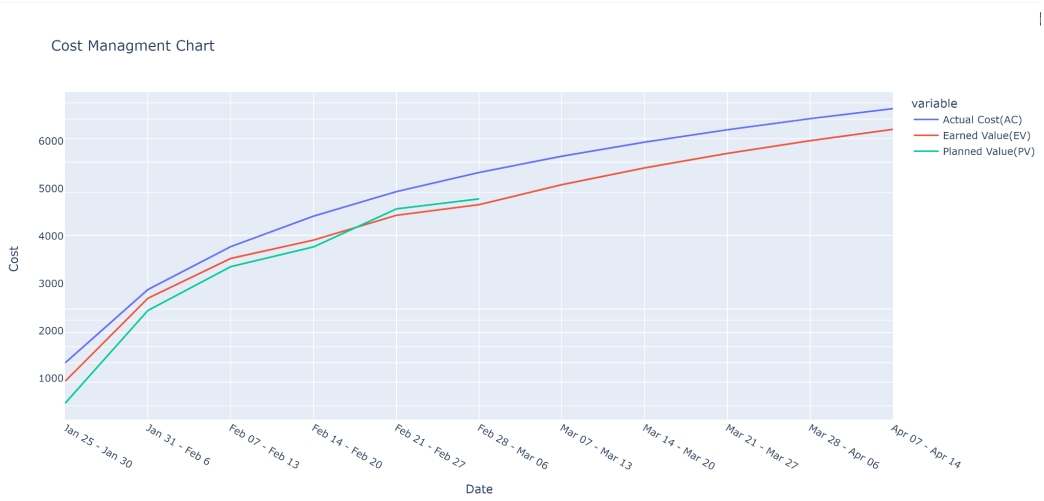
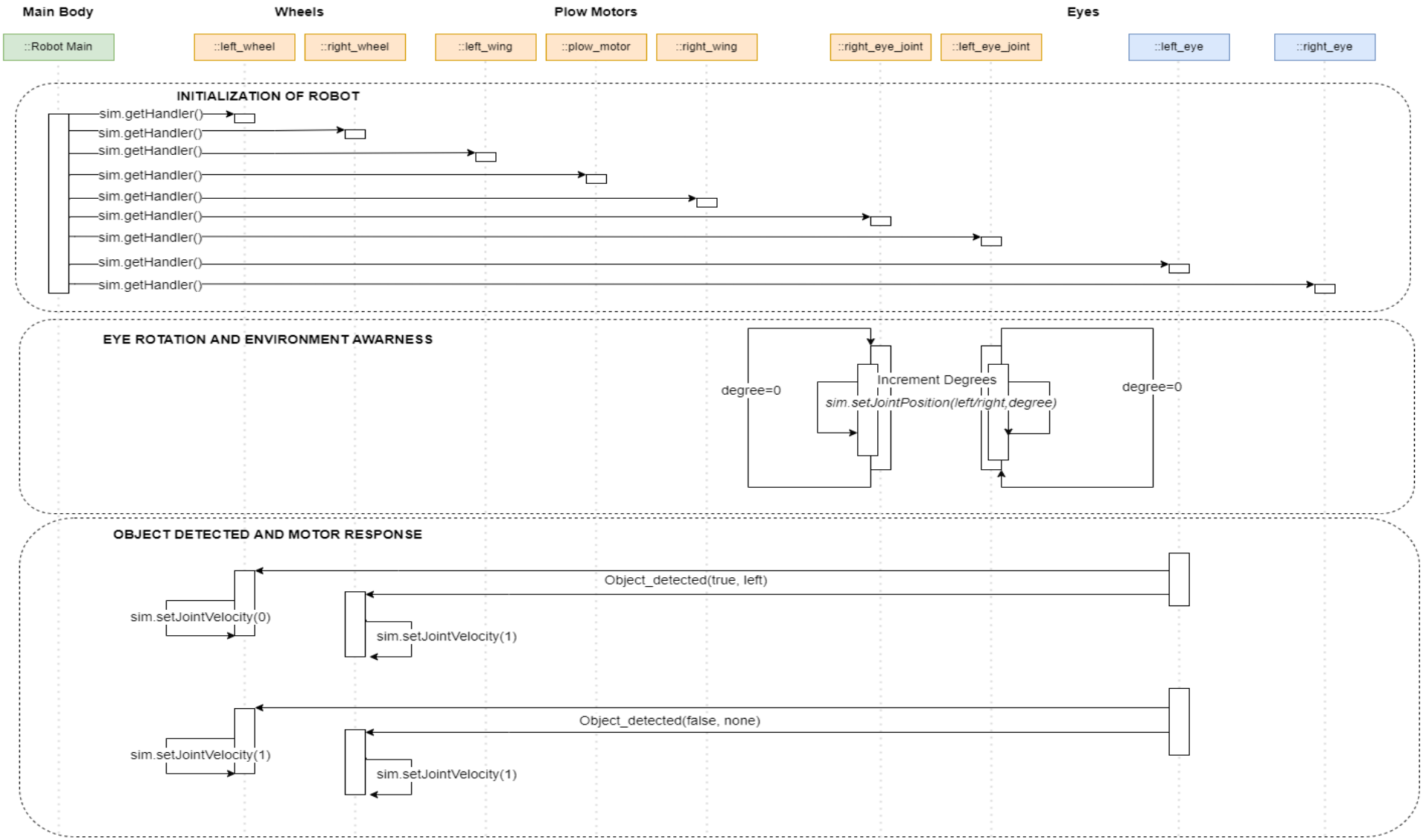


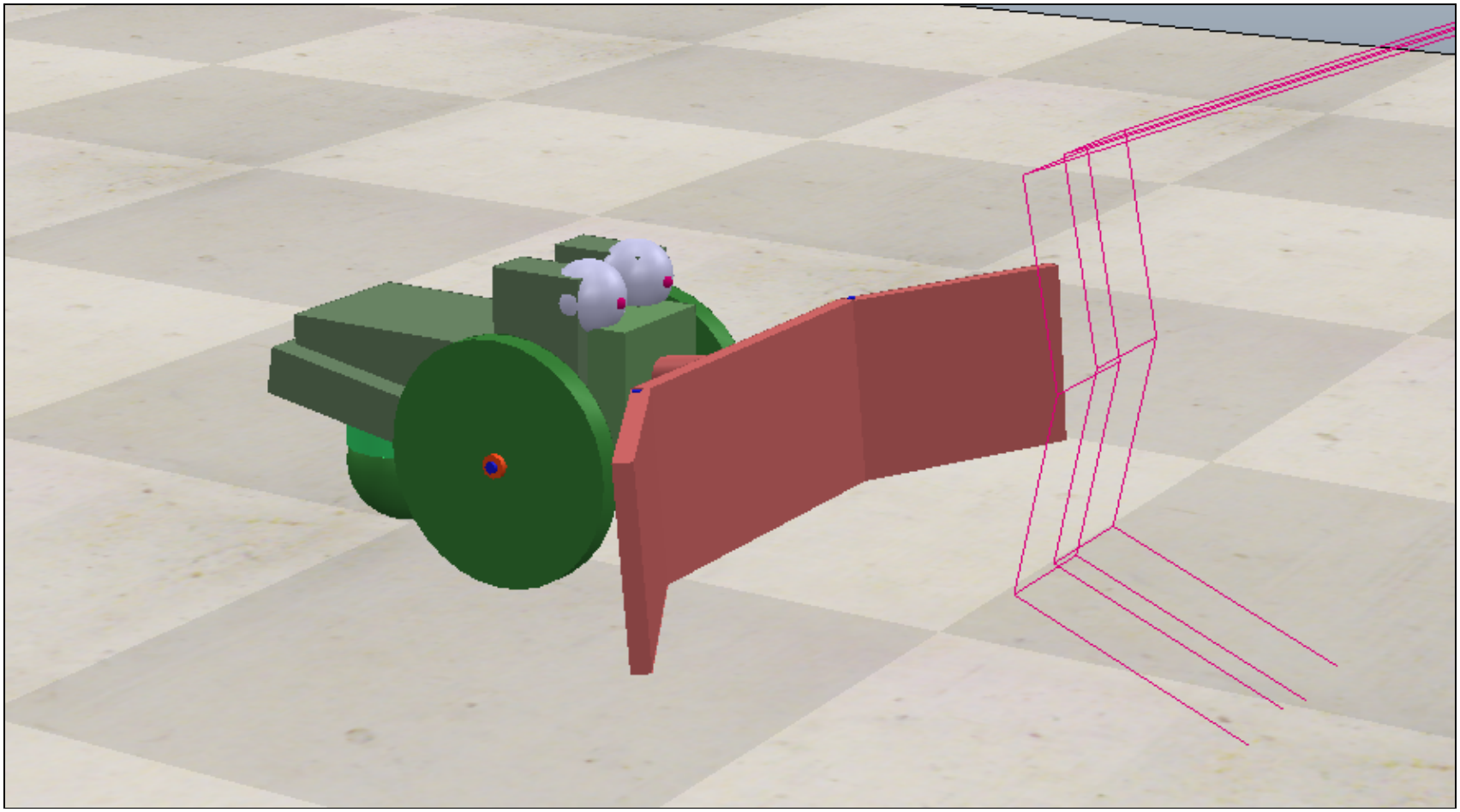
Figure 7: Cost Management Chart - see appendix for larger view

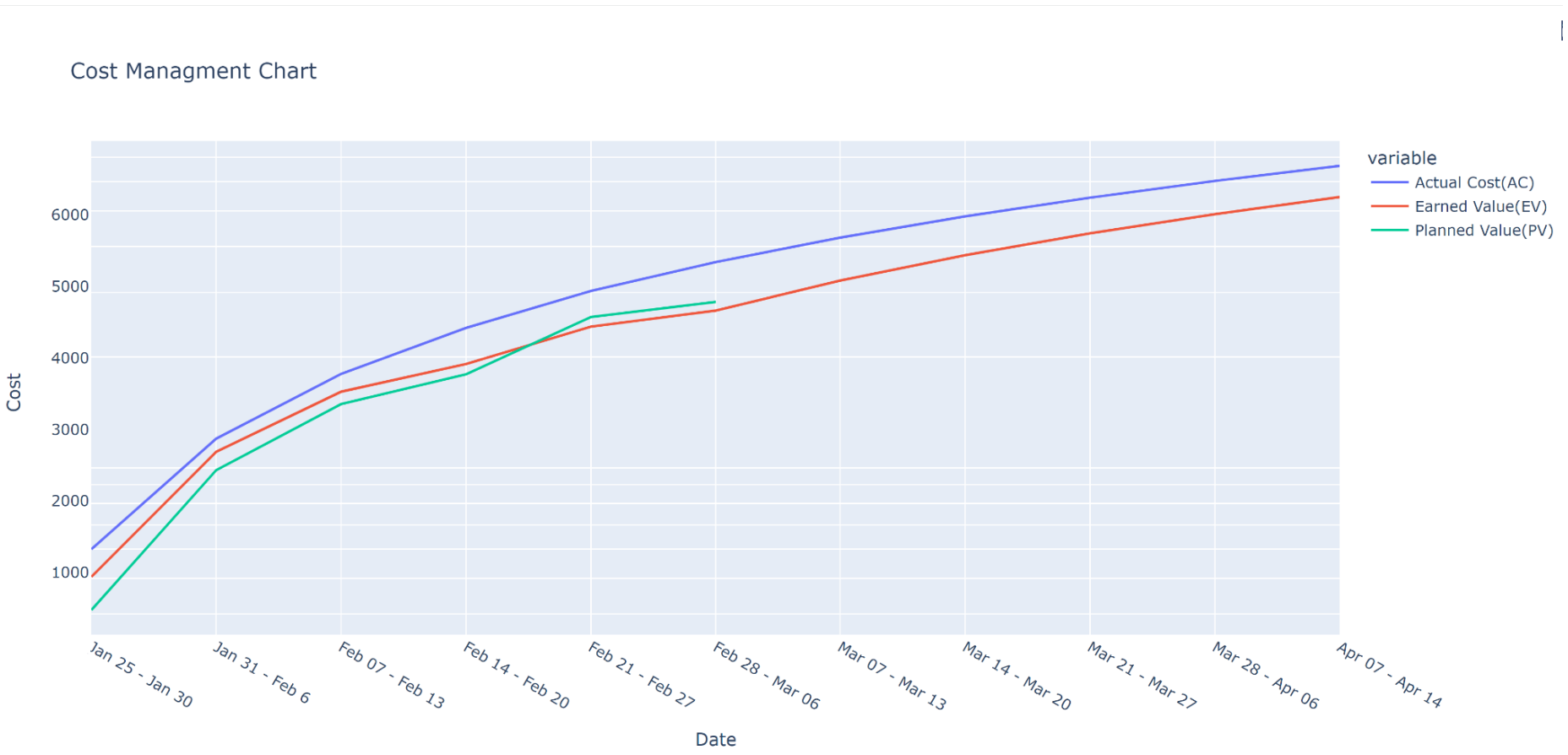
APPENDICES

APPENDIX A: Sequence Diagram



APPENDIX B: Current Project State




Cost Managment Chart
variable
Actual Cost(AC)
Earned Value(EV)
Planned Value(PV)
Cost
6000
5000
4000
3000
2000
1000
Jan 25 - Jan 30
Jan 31 - Feb 6
Feb 07 - Feb 13
Feb 14 - Feb 20
Feb 21 - Feb 27
Feb 28 - Mar 06
Mar 07 - Mar 13
Mar 14 - Mar 20
Mar 21 - Mar 27
Mar 28 - Apr 06
Apr 07 - Apr 14
Date