System Verification and Validation Plan for LiDart

Team 10
Jonathan Casella
Kareem Elmokattaf
Michaela Schnull
Neeraj Ahluwalia

November 2, 2022

1 Revision History

Date	Version	Authors	Notes
02/Nov/2022	1.0	Michaela Schnull Jonathan Casella Kareem Elmokattaf Neeraj Ahluwalia	Initial Release

Contents

List of Tables

2 Symbols, Abbreviations and Acronyms

symbol	description
CAD	Computer Aided Design
DRC	Design Rule Check
ERC	Electrical Rule Check
GUI	Graphical User Interface
HA	Hazard Analysis
LiDAR	Light Imaging, Detection, and Ranging
MG	Module Guide
MIS	Module Interface Specification
SLAM	Simultaneous Localization and Mapping
SRS	System Requirements Specification
V&V	Verification and Validation
T	Test
TA	Teaching Assistant

This document presents a verification and validation plan for the LiDart system. It will be used to establish verification and testing procedures and create a plan to determine if LiDart meets it's goals as defined in the Problem Statement and Goals document.

The remainder of this document is structured as follows:

- **Section 3** provides background information about the LiDartsystem which will be subject to verification and validation activities. It also outlines the objectives of this plan and lists relevant documents.
- **Section 4** defines the verification and validation team roles and responsibilities. It presents verification methodologies and tools that will be used.
- **Sections 5** and **6** define what will be tested, and provide specific test cases. Furthermore, traceability matrices are provided to link test cases to requirements.

3 General Information

3.1 Summary

LiDart is a low cost, simple to use, 3D scanning robot. The LiDart system uses of low cost LiDAR sensors, consumer grade web cams, and inexpensive location markers. The user interfaces with the robot through GUI that allows them to remotely drive the robot and perform 3D scans.

3.2 Objectives

The purpose of this document is to create a plan to demonstrate that LiDart satisfies requirements as specified in the SRS and meets the goals of the project. This includes following objectives:

- Validate that the LiDart system meets its goals
- Create a plan to asses if the LiDart system is in conformance with both functional and non-functional requirements as specified in the SRS
- Identify the methodologies, tools, and equipment that will be used to perform V&V activities
- Create test cases to execute the V&V plan

3.3 Relevant Documentation

Project documentation such as the SRS and design documents are referred to throughout the V&V plan. Relevant documents are included below for reference.

- System Requirements Specification (SRS) [1]
- Problem Statement and Goals [2]
- Module Interface Specification (MIS) [3]
- Module Guide (MG) [4]
- Hazard Analysis (HA) [5]

4 Plan

This section introduces methods and techniques used to verify design requirements and provides a high-level plan to validate the LiDart system.

4.1 Verification and Validation Team

Table 1 summarizes the roles and responsibilities of the verification and validation team.

Table 1: Team Member Roles

Team Member	**Roles and Responsibilities**
Jonathan Casella	- Software testing lead - Implementation of automated software testing methods - Design review lead
Michaela Schnull	- Verification of project documents including the SRS and V&V plan - Reporting of verification and validation activities - Printed circuit board verification
Kareem Elmokattaf	- Responsible for the design and execution of testing procedures - Maintenance of records documenting results from testing activities - Code verification
Neeraj Ahluwalia	- Preparation and maintenance of testing equipment - Hardware verification
Independent Reviewers (i.e. Teaching Assistant)	- Quality assurance and independent review

4.2 SRS Verification Plan

The SRS will be verified to ensure that requirements are complete, unambiguous, and meet the goals of the LiDart system. The following methods shall be used to verify the SRS:

- Verify that the SRS follows the SRS checklist [6]
- Review from all team members using GitHub pull request reviews
- Independent review from the TA and classmates

4.3 Design Verification Plan

The design of the LiDart system will be verified throughout development to ensure the system meets specifications and functions as intended. The following methods shall be used to verify the design:

- Hold internal design reviews before the proof of concept demo and prior to manufacturing of the system for Revision 0 and Revision 1 project phases
- Perform a failure modes and effects analysis

- Independent review from teaching assistants and classmates
- Verify that the design documentation follows the MIS [7] and MG [8] checklists

4.4 Verification and Validation Plan Verification Plan

The V&V plan will be verified to ensure that the plan to verify and validate the LiDart system is complete and feasible. The following methods shall be used to verify the V&V plan:

- Independent review from the TA and classmates
- Review from all team members using GitHub pull request reviews
- Verify that the V&V plan follows the V&V plan checklist [9]

4.5 Implementation Verification Plan

The implementation verification plan will be used to ensure the LiDart system meets all requirements as specified in the SRS. Verification methods that will be used include review, analysis, demonstration, and testing.

- **Review**: Review can be used when meeting a requirement is evident to a trained observer. Review of engineering drawings, code, or the physical device may be used. Techniques include code walk-throughs, code inspection, and drawing reviews.
- **Analysis**: Analysis can be used to verify design requirements where physical testing is not necessary, for example through mathematical and computer modeling. Analysis of data obtained through testing may be used to verify requirements. This verification method must be conducted by qualified individuals.
- **Demonstration**: Demonstration can be used to show that the system functions as intended. Unlike testing, demonstration does not require further analysis to determine if the system meets a requirement.
- **Testing**: Testing can be used to verify the behaviour of the system. Testing is conducted in a controlled environment with defined inputs and outputs. Test results must be analyzed to determine if tests pass or fail. Techniques that will be used include unit testing, automated testing, regression testing, and integration testing.

Section 5 specifies system test cases and Section 6 specifies unit test cases.

4.6 Automated Testing and Verification Tools

4.6.1 Rust

Software for the LiDart project will be written in Rust, which natively includes testing facilities. The testing facilities built into Rust are explained in chapter 5 of the rustc book [10].

4.6.2 Rustfmt

The primary linter for Rust is Rustfmt. Rustfmt will be used to ensure a consistent programming style across all contributors.

4.6.3 Autodesk Inventor

LiDart's CAD suite of choice, Autodesk Inventor, will be used to ensure no mechanical conflicts exist in the design prior to manufacturing and assembly. This will be done by first modeling the LiDart robot in Autodesk Inventor then using tools within Inventor to detect any collisions or interpenetrating components.

4.6.4 EAGLE

Printed circuit board design will be done in EAGLE, which has built-in error checking tools. ERC (Electrical Rule Check) will be used to test schematics for electrical errors. DRC (Design Rule Check) will be used to ensure there are no board layout errors.

4.7 System Validation Plan

This section lays out a series of validation exercises that will be performed before the system is deemed suitable for general users. The following exercises are designed to test the 3 major goals set out in the problem statement.

4.7.1 Accurate 3D Scanning

To validate the accuracy of the 3D scans a test environment will be constructed and scanned. Said test environment will have objects with known shapes at predetermined positions. The resulting 3D scan will be compared against the expected values for the test environment. If the point cloud matches the expected result within some tolerance this validation exercise will be deemed complete.

4.7.2 Low Cost Hardware

To validate that the goal of low cost, easily accessable hardware was met two exercises will be performed.

The first exercise will be to compare LiDart's price to other options in the market with equivalent features and comparable build quality. LiDart's price should fall below the median price of the existing solutions.

The second exercise will be to evaluate the accessibility of the hardware used by LiDart. During this exercise a list of alternative hardware components will be created for all major components of the LiDart system (e.g. motor controller, motors, cameras). This exercise will be deemed complete when all major components have at least one alternate part listed.

4.7.3 Ease of Use

To validate LiDart's ease of use a focus group will be used. The focus group will consist of subjects with no prior knowledge of the system. Each member of the focus group will be provided with the user manual then asked to scan an object. The time required to complete the scan and the percentage of successful subjects will be used to determine the success of this exercise.

Usability survey questions

5 System Test Description

5.1 Tests for Functional Requirements

The following test cases present verification methods for functional requirements of the LiDART system.

5.1.1 Movement Controller

T1: Verification Method: Testing
Initial State: Nothing has been sent to the robot to output
Input: Arrow key movement (example: right arrow key)
Output: Robot moves in the direction specified (example: to the right)
How test will be performed: The robot will be given a series of inputs from the keyboard such as Right,Left and forward. The robot will then be observed and make sure that it follows the motions specified. This will verify that the robot is taking the input from the user as specified

T2: Verification Method: Demonstration
Initial State: Robot powered-on, connected, and stationary
Input: Arrow key movement (example: right arrow key)
Output: Robot moves in the direction specified (example: to the right)
How test will be performed: The robot will be given a series of inputs from the keyboard such as Right,Left and forward. The robot will then be observed and make sure that it follows the motions specified. This will verify that the robot is taking the input from the user as specified

T3: Verification Method: Demonstration
Initial State: Robot is operating normally
Input: Emergency stop button is pressed
Output: Robot should completely shut down
How test will be performed: Inputs will be sent to the robot and the robot should not action any of them. So the inputs can be movement (left, right and forward) or it can be to scan the object.

T4: Verification Method: Demonstration
Initial State: Robot is stationary
Input: From the control, there will be input specified to move the robot forward, backwards, left and right
Output: Robot should move in all specified directions
How test will be performed: Inputs will be sent to the robot and the robot should operate accordingly. The robot should move forward, backward, and rotate left and right.

T5: Verification Method: Demonstration
Initial state: Robot is stationary
Input: No inputs are given to the robot
Output: Robot does not move or do any functions

How test will be performed: Robot will be left without any inputs given to it and its behaviour will be monitored/observed

T6: Verification Method: Testing
Initial state: Robot is stationary
Input: Through a remote device connected over the internet, there will be commands given to the robot (example: Move left and Scan)
Output: Robot should move left and then start scanning
How test will be performed: Robot will be remotely connected to a control source and the robot will then be given commands from the control source and the actions of the robot will be observed

T7: Verification Method: Testing
Initial state: Robot will be idle (TO BE REVIEWED)
Input: Through the internet connection established, parameters will be sent to the robot (example: robot status)
Output: Robot will respond over wifi connection with the robot status
How test will be performed: Over a wifi connection a control will send certain paramters to the robot and the expected output should be received from the robot

T8: Verification Method: Demonstration
Initial state: Robot is stationary
Input: Through a remote device connected over the internet, there will be commands given to the robot (example: Move left and Scan)
Output: Robot should move left and then start scanning
How test will be performed: Robot will be remotely connected to a control source and the robot will then be given commands from the control source and the actions of the robot will be observed

T9: Verification Method: Initial state: Robot will be idle (TO BE REVIEWED)
Input: Through the internet connection established, parameters will be sent to the robot (example: robot status)
Output: Robot will respond over WiFi connection with the robot status
How test will be performed: Over a WiFi connection a control will send certain parameters to the robot and the expected output should be received from the robot

T10: Verification Method:
Initial state: Robot is currently operating
Input: Robot switch will be flicked to go from "On" to "Off"
Output: Robot should power down and not have any power
How test will be performed: Robot will be powered on working. After confirming that the robot is powered on and performing actions, the robot's switch will be flicked to off and the robot should turn off

T11: Verification Method: Initial state: Robot battery is low
Input: Robot batter will be connected to a battery charger
Output: Robot battery percentage should start increasing
How test will be performed: Robot will have a low battery. The battery will then be connected to a charger and tester will observe as the battery percentage increases on the GUI.

5.1.2 State Estimation and Scanning

T12: Verification Method:
Initial state: Robot is operating without having scanned anything
Input: Robot scan
Output: Output from the robot to the remote connection will be the scans coming from the LiDAR sensor
How test will be performed: Robot will be placed close to an object and instructed to scan the object. The tester will then observer the output from the robot on the remote connection.

T13: Verification Method: Testing
Initial state: Robot uncalibrated
Input: Landmarks in the surrounding and the instruction to clibrate the position of the robot
Output: Coordinate of the robot in comparison to the landmarks
How test will be performed: Landmarks will be placed a specific distance away from the robot at a specific orientation. The robot will be asked to calibrate. The tester will verify the measurements from the robot and verify that they are correct

T14: Verification Method:
Initial state: Robot is not calibrated
Input: Landmarks in the surrounding and the instruction to calibrate the position of the robot
Output: Coordinate of the robot in comparison to the landmarks
How test will be performed: Landmarks will be placed a specific distance away from the robot at a specific orientation. The robot will be asked to calibrate. The tester will verify the measurements from the robot and verify that they are correct

T15: Verification Method:
Initial state: Robot has scanned an object
Input: User identifies that they are done scanning and what to see the results
Output: Robot will send the output file to the remote connection
How test will be performed: (TO BE REVIEWED) Robot will have files pre-installed and then asked to communicate them in the same that they will communicate the results of the scan. Those files will the be verified to be correct and not corrupted

T16: Verification Method:
Initial state: Robot moving with the camera installed
Input: User moving the robot around

Output: Live feed from the camera onto the remote connection
How test will be performed: Tester will place different objects infront of the robot and watch them change on the live feed on the GUI

T17: Verification Method:
Initial state: Robot scanning the object
Input: User continuing the scan of an object
Output: Current information on the scan. The current 3D scan model of the scan
How test will be performed: Tester will observe as the scan is happening and look at the output on the GUI from the robot. The tester will be able to confirm that the scan is coming as the robot is scanning

T18: Verification Method:
Initial state: Robot is ready to scan
Input: User asking for the current state of the robot
Output: Robot outputs that it is ready to scan
How test will be performed: Robot will be put in different states (Idle, ready to scan, etc..) and then asked to output the state to the GUI. The tester will be able to confirm that the output is matching what is expected

5.2 Tests for Nonfunctional Requirements

The following test cases present verification methods for non-functional requirements of the LiDART system.

5.2.1 Accuracy Requirements

T19: NFR1

Verification Method: Testing
Initial State: Robot is fully assembled
Input: n/a
Output: The 3D scan must be within SCAN_TOL
How test will be performed: The object that was scanned will be measured. The measurements from the object will be compared to that of the 3D scan provided by the robot S

5.2.2 Performance Requirements

T20: NFR2

Verification Method: Testing
Initial State: Robot is fully assembled and is operating
Input: n/a

Output: The robot is moving at a speed of ROBOT_SPEED
How test will be performed: The robot will have to cover a specified distance. The time it takes to cover that distance will be measured. The speed can thus be calculated to determine the speed of the robot

T21: NFR3

Verification Method: Testing
Initial State: Robot is fully assembled and operating
Input: n/a
Output: Robot must run for RUN_TIME
How test will be performed: The robot will be fully charged, and then left to operate. The time from when it is turned on till the battery runs out will be measured

T22: NFR4

Verification Method: Testing
Initial State: Robot is fully assembled and operating
Input: n/a
Output: The system must be able to produce a 3D scan within SCANNING_TIME
How test will be performed: Once the robot starts scanning, the timer will be started until the single planar scan is completed. When it is completed the timer will be stopped

T23: NFR5

Verification Method: Testing
Initial State: Robot is fully assembled and operating
Input: n/a
Output: The system must be able to output the final 3D scan withing MA_PROCESS_T time
How test will be performed: Once the robot starts scanning the object, the timer will be started. Once the robot is fully done scanning and has produced the 3D scan file, the timer will be stopped

T24: NFR6

(*TO BE CHECKED*) Verification Method: Testing
Initial State: Robot is fully assembled and operating
Input: n/a
Output: The robot must respond within MAX_RESP_T amount of time
How test will be performed: A command will be sent to the robot at a specified time, the robot will respond with the time time. The time difference will be the response time

T25: NFR7

(*TO BE CHECKED*) Verification Method: Testing
Initial State: Robot is fully assembled
Input: n/a
Output: The live feed of the video must have a minimum resolution of MIN_VIDEO_RES
How test will be performed: The live feed from the robot can be stored on a Windows platform. The video can then be checked from its properties to verify the resolution

T26: NFR8

(*TO BE CHECKED*) Verification Method: Testing
Initial State: Robot is fully assembled and operating
Input: n/a
Output: The live feed displayed on the GUI must have a maximum delay of MAX_VIDEO_DELAY
How test will be performed: NO IDEA HOW THIS ONE IS TO BE DONE. CLOCK? MOVE SOMETHING AWAY FROM DISPLAY? IDK

5.2.3 Usability Requirements

T27: NFR10

Verification Method: Testing
Initial State: Robot is fully assembled
Input: n/a
Output: The robot must weigh less than ROBOT_MAX_WEIGHT.
How test will be performed: The robot shall be weighed on a scale. The weight of the robot and the accuracy of the measuring device must be recorded.

T28: NFR11

Verification Method: Testing
Initial State: Robot is fully assembled
Input: n/a
Output: The robot dimensions must be less than ROBOT_MAX_DIM
How test will be performed: The robot dimensions (length x width x height) shall be measured with a measuring tape. The robot dimensions and the accuracy of the measuring device must be recorded.

T29: NFR12

Verification Method: Review
Initial State: The GUI main page is open

Input: n/a
Output: n/a
How test will be performed: Each page will be navigated to from the main screen. The number of navigation levels will be recorded for each page. It must take less than MAX_NUM_LEVELS to navigate to any page on the application.

T30: NFR13

Verification Method: Review
Initial State: The GUI main page is open
Input: n/a
Output: n/a
How test will be performed: A qualified individual will review the application to ensure all text is consistent, has a minimum font size of MIN_FONT_SIZE, and is sans-serif.

T31: NFR9

Verification Method: Testing
Initial State: The robot is powered-on and connected to the GUI through WiFi
Input: The robot is powered on and the GUI is open
Output: At least 9/10 users with no prior knowledge of the system are able to successfully drive the robot, perform a scan, and save the scan data
How test will be performed: A usability test shall be performed. Users will be given a user manual and instructed to drive the robot, perform a scan, and save the scanned data.

T32: NFR14, NFR12

Verification Method: Usability Survey
Initial State:
Input:
Output:
How test will be performed:

5.2.4 Error-Handling Requirements

T33: NFR15

Verification Method: Testing
Initial State: The robot is powered-on and connected to the GUI through WiFi
Input: User begins scanning and then attempts to move the robot
Output: Error message stating the robot cannot move until scanning is complete
How test will be performed:

5.2.5 Maintainability Requirements

T34: NFR17

Verification Method: Review
Initial State: n/a
Input: n/a
Output: n/a
How test will be performed:

T35: NFR18

Verification Method: Review
Initial State: n/a
Input: n/a
Output: n/a
How test will be performed: A qualified individual shall review drawings and the assembled robot to determine if all hardware and electronic components are easily accessible.

T36: NFR19

Verification Method: Review
Initial State: n/a
Input: n/a
Output: n/a
How test will be performed: A list of alternative hardware components will be created for all major components of the LiDart system (e.g. motor controller, motors, cameras). All major components should have at least one alternate part.

5.2.6 Portability Requirements

T37: NFR20, NFR21

Verification Method: Testing
Initial State: The GUI application is installed
Input: Open the application
Output: The application must run without crashing or performance issues
How test will be performed: The application will run on a computer with the specifications IntelCore i5 processor and 8 GB of RAM or equivalent.

5.2.7 Safety Requirements

T38: NFR22

Verification Method: Review
Initial State: The robot is fully assembled
Input: n/a
Output: n/a
How test will be performed: A qualified individual shall inspect the robot to ensure there is no exposed wiring.

5.2.8 Standards Requirements

T39: NFR25

Verification Method: Review
Initial State: The robot is fully assembled
Input: n/a
Output: n/a
How test will be performed: A qualified individual shall inspect the robot to ensure it is in conformance with CSA 22.1:21 (Canadian Electrical Code) [11] and CSA Z434 (Industrial robots and robot systems) [12]

5.3 Traceability Between Test Cases and Requirements

The following tables show the traceability between test cases and requirements from the SRS. Table 2 shows the functional requirements and Table 3 shows the non-functional requirements.

NEED TO UPDATE TABLES

Table 2: Functional Requirements Dependency Matrix

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14
R1	X													
R2		X												
R3	X		X											
R4				X										
R5	X				X	X								
R6					X	X								
R7							X							
R8								X						
R9									X					
R10										X				
R11											X			
R12												X		
R13													X	
R14														X

Table 3: Non-Functional Requirements Dependency Matrix

	T	T	T	T	T	T	T	T	T	T	T	T	T	T
NFR1														
NFR2														
NFR3														
NFR4														
NFR5														
NFR6														
NFR7														
NFR8														
NFR9														
NFR10														
NFR11														
NFR12														
NFR13														
NFR14														
NFR15														
NFR16														
NFR17														
NFR18														
NFR19														
NFR20														
NFR21														
NFR22														
NFR23														
NFR24														
NFR25														

6 Unit Test Description

6.1 Unit Testing Scope

6.2 Tests for Functional Requirements

6.2.1 Module 1

1. test-id1

   Type:

   Initial State:

Input:

Output:

Test Case Derivation:

How test will be performed:

2. test-id2

   Type:

   Initial State:

   Input:

   Output:

   Test Case Derivation:

   How test will be performed:

3. ...

6.2.2 Module 2

...

6.3 Tests for Nonfunctional Requirements

6.3.1 Module ?

1. test-id1

   Type:

   Initial State:

   Input/Condition:

   Output/Result:

   How test will be performed:

2. test-id2

   Type: Functional, Dynamic, Manual, Static etc.

   Initial State:

   Input:

   Output:

   How test will be performed:

6.3.2 Module ?

...

6.4 Traceability Between Test Cases and Modules

References

[1] N. Ahluwalia, J. Casella, K. Elmokattaf, and M. Schnull, "Software requirements specification," 2022.

[2] J. Casella, "Problem statement and goal," 2022.

[3] N. Ahluwalia, J. Casella, K. Elmokattaf, and M. Schnull, "Module interface specification," 2022.

[4] N. Ahluwalia, J. Casella, K. Elmokattaf, and M. Schnull, "Module guide," 2022.

[5] M. Schnull and K. Elmokattaf, "Hazard analysis," 2022.

[6] S. Smith, "SRS and CA checklist," 2022.

[7] S. Smith, "MIS checklist," 2022.

[8] S. Smith, "MIS checklist," 2022.

[9] S. Smith, "System verification and validation plan checklist," 2022.

[10] Rust Programming Language, *Testing.* https://doc.rust-lang.org/rustc/tests/index.html.

[11] CSA Group, "CSA C22.1:21 Canadian electrical code, part I (25th edition), safety standard for electrical installations," tech. rep., 2021.

[12] CSA Group, "CAN/CSA-Z434-14 Industrial robots and robot systems (adopted ISO 10218-1:2011, second edition, 2011-07-01, with Canadian deviations and ISO 10218-2:2011, first edition, 2011-07-01, with Canadian deviations)," tech. rep., 2019.

7 Appendix

7.1 Symbolic Parameters

The definition of the test cases will call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance.

Constant	**Description**	**Value**
ROBOT_SPEED	Maximum speed of robot in navigation mode.	0.5m/s
SCAN_TOL	Maximum deviation between sufficient scans.	+/- 1cm
RUN_TIME	Maximum time robot can run before needing to be recharged.	3hrs
SCANNING_SPACE	Unit used for defining robot scan speed	1 square meter
SCANNING_TIME	Time elapsed to complete scan of a specified scanning space.	2mins
MAX_PROCESS_T	Time needed for the robot to process raw scanned data.	10min
MAX_RESP_T	Time needed for robot to respond to navigational commands given by user.	1s
MIN_VIDEO_RES	Minimum resolution of video output feed in GUI.	1280x720p
MAX_VIDEO_DELAY	Max time between change in camera and output video feed in GUI.	1s
ROBOT_MAX_WEIGHT	Maximum weight of the fully assembled robot.	50kg
ROBOT_MAX_DIM	Maximum size of the fully assembled robot.	1m x 1m x 3m (LxWxH)
MAX_NAV_LEVELS	Maximum number of layers required to access any function in GUI.	4
MIN_FONT_SIZE	The minimum font size used in the GUI for accessibility purposes.	14pt
EXECUTE_TASK_TIME	Time it would take an average user to execute 9/10 tasks within the GUI.	10s
REPLACE_TIME	Time it would take an average user to replace a damaged component.	1hr

7.2 Usability Survey Questions

1. On a scale of 1-10, how easy is it to navigate the user application?
2. On a scale of 1-10, how helpful are the error messages that are displayed?
3. On a scale of 1-10, how would you rate the usability of the application?
4. On a scale of 1-10, how easy is it to read text on the application?
5. Were you able to successfully use the system by following the instructions in the user manual?

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Lifelong Learning. Please answer the following questions:

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Lifelong Learning. Please answer the following questions:

1. What knowledge and skills will the team collectively need to acquire to successfully complete the verification and validation of your project? Examples of possible knowledge and skills include dynamic testing knowledge, static testing knowledge, specific tool usage etc. You should look to identify at least one item for each team member.

2. There are a number of skills that will be needed for the verification and validation of the project. This includes creation of testing plans, accurately measuring distance to verify localization, precision phone movement,

3. For each of the knowledge areas and skills identified in the previous question, what are at least two approaches to acquiring the knowledge or mastering the skill? Of the identified approaches, which will each team member pursue, and why did they make this choice?

4. Michaela will pursue the creation of the testing plans as she has the most experience with designing a comprehensive set of tests that will cover all possible options. She will obtain knowledge from reviewing previous projects and researching existing tests of similar robots.

5. Jonathan will pursue the testing of the software as he would know the software's function and potential issues. He will execute Michaela's plan and obtain knowledge in the areas of software testing from online resources, and from the development of the code and its potential problems.

6. Neeraj will pursue the testing of the mechanical aspects of the robot as during his construction of the parts, will gain a sense of the required trials. He will obtain knowledge from mechanical engineering textbooks as well as online from researching existing robotic mechanical malfunctioning.

7. Kareem will pursue the testing of the electrical system as it pertains to his knowledge around the electrical engineering in this project. Kareem will be vaidating the electrical system against common faults and will expose it to a variety of conditions. He will obtain the knowledge required by investigating online sources and contacting professors at McMaster for guidance on robot electrical issues.