Software Requirements Specification
LiDart

Team 10
Jonathan Casella
Kareem Elmokattaf
Michaela Schnull
Neeraj Ahluwalia

October 5, 2022

Contents

Revision History

Date	Version	Authors	Notes
05\Oct\2022	1.0	Michaela Schnull Jonathan Casella Kareem Elmokattaf Neeraj Ahluwalia	Initial Release

1 Reference Material

This section records information for easy reference.

1.1 Table of Units

Throughout this document SI (Système International d'Unités) is employed as the unit system. In addition to the basic units, several derived units are used as described below. For each unit, the symbol is given followed by a description of the unit and the SI name.

symbol	unit	SI
m	length	metre
kg	mass	kilogram
s	time	second
W	power	watt ($\mathrm{W} = \mathrm{J\,s^{-1}}$)

1.2 Abbreviations and Acronyms

symbol	description
A	Assumption
CAD	Computer Aided Design
CMM	Coordinate Measuring Machine
FSM	Finite State Machine
GS	Goal Statement
GUI	Graphical User Interface
IM	Instance Model
LiDAR	Light Imaging, Detection, and Ranging
LC	Likely Change
NFR	Nonfunctional Requirement
R	Requirement
RAM	Random Access Memory
SRS	System Requirements Specification
VR	Virtual Reality

1.3 Terminology and Definitions

Term	Definition
Term	Definition

2 Introduction

Businesses and individuals are increasingly using 3D scanning technologies to collect 3D data for modeling and analysis. The 3D scanning market is rapidly growing, with a wide range of applications such as VR, rapid prototyping, reverse-engineering, and inspection technologies. Many current scanning technologies require that objects are brought to specialized scanning facilities, where fixed devices such as CMM machines and robotic arms are installed. Hand-held scanning devices are also available, however these technologies are expensive and require human operation. The cost and domain specific knowledge required are barriers to many users. There are limited solutions available for portable, remotely operated, and inexpensive 3D scanning solutions. A low cost, portable, and user-friendly scanning solution would make scanning accessible to a wider audience.

This document defines the scope of the LiDart project, specifies the scanning system, and defines requirements. The purpose of the SRS is discussed in detail in Section 2.2. Section 2.3 describes key project deliverables, as well as exclusions that are outside the scope of work. Readers of this document should be familiar with concepts outlined in Section **??**.

2.1 Problem Description and Goals

2.2 Purpose of Document

The purpose of this document is to provide specifications and requirements for the LiDart 3D scanning system. This document describes functional and non-functional requirements, undesired event handling, and start-up behavior. The functional behaviour of the system is modeled through system diagrams and instance models. The requirements defined in this document will drive design decisions and will be referenced throughout the design phase to ensure requirements are being met. The requirements will be a direct input to the verification and validation plan.

2.3 Scope of Requirements

LiDart aims to design and build a low cost 3D scanning robot, paired with a user application that displays a model of the scanned data. The robot is expected to operate indoors on flat surfaces in a controlled environment. A robot that is capable of operating outdoors or on rough terrain is not in the scope of this project. The user software application will allow users to view 3D data. However, tools for analyzing and modifying 3D models generated by the application will not be implemented. Software considerations such as licensing, user authentication, security, and data storage are also not within the scope of this project.

2.4 Organization of Document

The rest of this document provides detailed specifications and requirements for the LiDart 3D scanning system. The document is organized as follows:

Section 3: General System Description
: A general overview of the system is provided using a system context diagram. The interactions between the system, the environment,and its users are identified.

Section 4: Specific System Description
: The problem description and project goals are given, followed by system specifications including assumptions, theories, definitions, and instance models.

Section 5: Requirements
: This section defines the functional and non-functional requirements of the system.

Section 7: Likely Changes
: This section describes changes to system components that are likely to occur as a result of new features or changes in scope.

Section 8: Unlikely Changes
: This section describes the system requirements that are not likely to change.

Section 9: Development Plan
: A plan outlining the steps that will be taken to create the LiDart system is given.

Section 10: Values of Auxiliary Constants
: This section provides values for symbolic parameters used in this document.

3 General System Description

This section provides general information about the system. It identifies the interfaces between the system and its environment, describes the user characteristics and lists the system constraints.

3.1 System Context

Figure 1 is a system context diagram of the LiDart 3D scanning system.

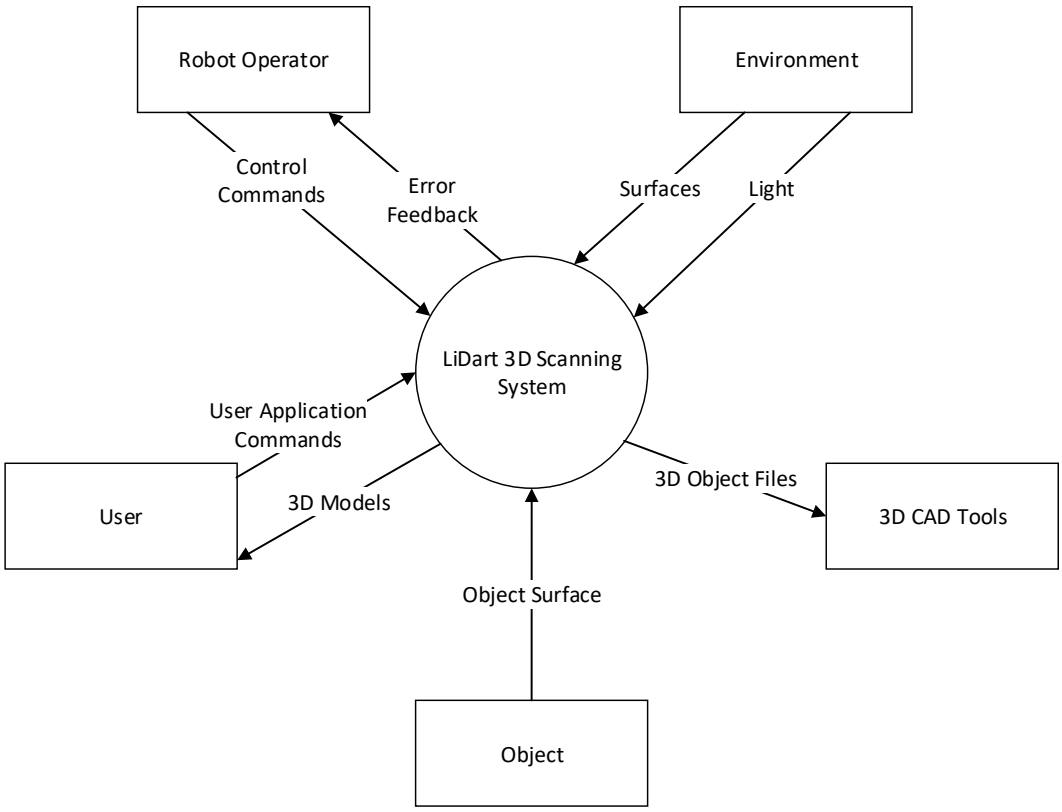

Figure 1: System Context Diagram

- Robot Operator:
  - The operator remotely controls the robot by sending commands to the robot. It is the operator's responsibility to instruct the robot to move and scan the desired object.
- Environment:
  - Information extracted from the surrounding environment is an input to the scanning system.
- User:
  - The user is responsible for providing input commands to the user application.
- Object:
  - The desired object that is being scanned is an input to the system.

- 3D CAD Tools:
  - External 3D CAD tools are able to import 3D object files that are outputted by the LiDart system. Further analysis and modification of 3D data can be carried out external CAD applications.
- LiDart 3D Scanning System:
  - The system will provide the robot operator with alerts about errors that occur during scanning operations.
  - The system is responsible for executing commands provided by the operator.
  - The system will check that the data it receives from the surrounding environment is valid.
  - The user application will respond to commands provided by the user.
  - The system outputs 3D Object Files that are supported by external CAD tools.
  - The user application will provide the user with view-able 3D models.

3.2 User Characteristics

The LiDart 3D scanning system does not require any domain specific knowledge. Individuals and business using the system are expected to have the following prior knowledge:

- An understanding of how to install applications on personal computers
- An understanding of 3D data file types and CAD programs
- Basic computer skills, such as how to navigate through application menus

3.3 Constraints

The following constraints are imposed on the LiDart system:

C1 The total cost of the system must be less than $750.

C2 The project must be completed by April 2022.

4 Specific System Description

The solution that LiDart proposes to the specified problem is a two wheeled mobile robot that can be remotely driven to take 3D scans of its environment.

The robot will have all required sensors onboard, these sensors will include: - One or more cameras - A LiDAR module capable of measuring the distance to objects in a given plane relative to the sensor's orientation and position - Two encoders capable of measuring wheel rotations, one for each of the two wheels

4.1 Assumptions

It is assumed that the environment that LiDart will operate within will have the following properties.

- Not exposed to the elements (eg. Rain) The robot will not be weather proofed, as this increases cost and complexity. A weather proofed configuration could be designed at a later date by modifiying the current robot.

- Floor should be flat and suitable for driving on with plastic wheels (dry, hard, not slippery. eg. tile, hardwood, concrete) The robot's simple 2 wheeled drive train is not suited to all terrains. This drive train was chosen to minimize cost and complexity.

- Will remain static while the robot is active This ensures there are no artifacts in the final stitched 3D scan resulting from changes to the environment between individual 3D scans.

- Prepopulated with landmarks This choice was made to simplify the operation of the state estimation algorithm. This should reduce the need for the user to understand the inner workings of the state estimation process.

- No translucent or reflective objects Translucent and reflective objects can result in poorly behaved LiDAR measurements, and may impact the final stitched 3D scan.

- Is illuminated such that the onboard camera(s) can clearly view the surrounding The robot relies on cameras for state estimation, as such it must operate in a sufficiently lit environment.

4.2 Behaviour Overview

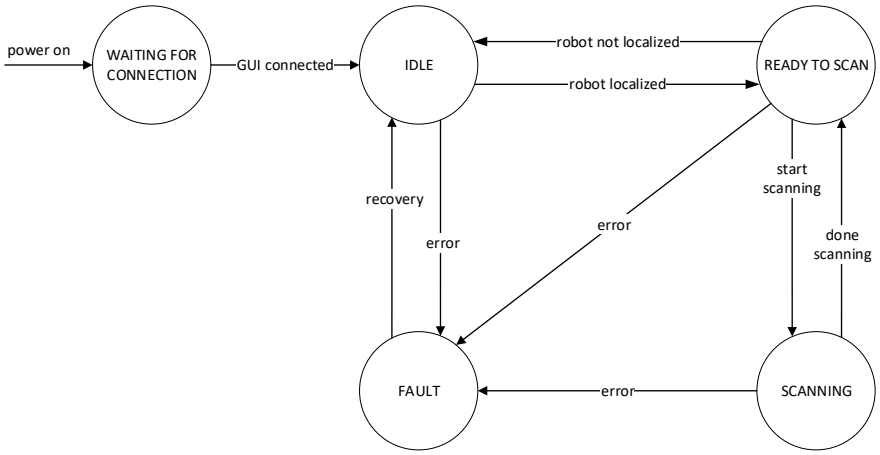

Figure 2: Behaviour Overview

Figure A2 shows the high level behaviour of the LiDart system.

Normal operation of the LiDart system will proceed as follows: 1. Robot is powered on 2. User connects to the robot via WiFi 3. User opens the GUI 4. User drives the robot to a desired location, the user can use the live video feed and LiDAR data preview to choose such a location 5. User initiates a scan 6. Steps 4 and 5 are repeated until the user has scanned all desired areas of the environment 7. User downloads the final stitched 3D scan

There are several events that fall outside of normal operations, those include:

Cancelling a scan - If the user has initiated a scan but then decides that the location is not ideal, the user can cancel the scan. This discards the information from that specific scan and allows the user to resume driving the robot.

Robot not localized - If the robot is not localized, it will not allow the user to initiate a scan. To be able to initiate a scan the user must first drive the robot to a location where the robot can localize itself.

Emergency stopping - If the robot begins misbehaving it can be powered off with an onboard emergency stop switch. This is required in the event that a hardware failure causes the robot to pose a safety risk or to stop responding to user commands.

4.3 Functional Decomposition

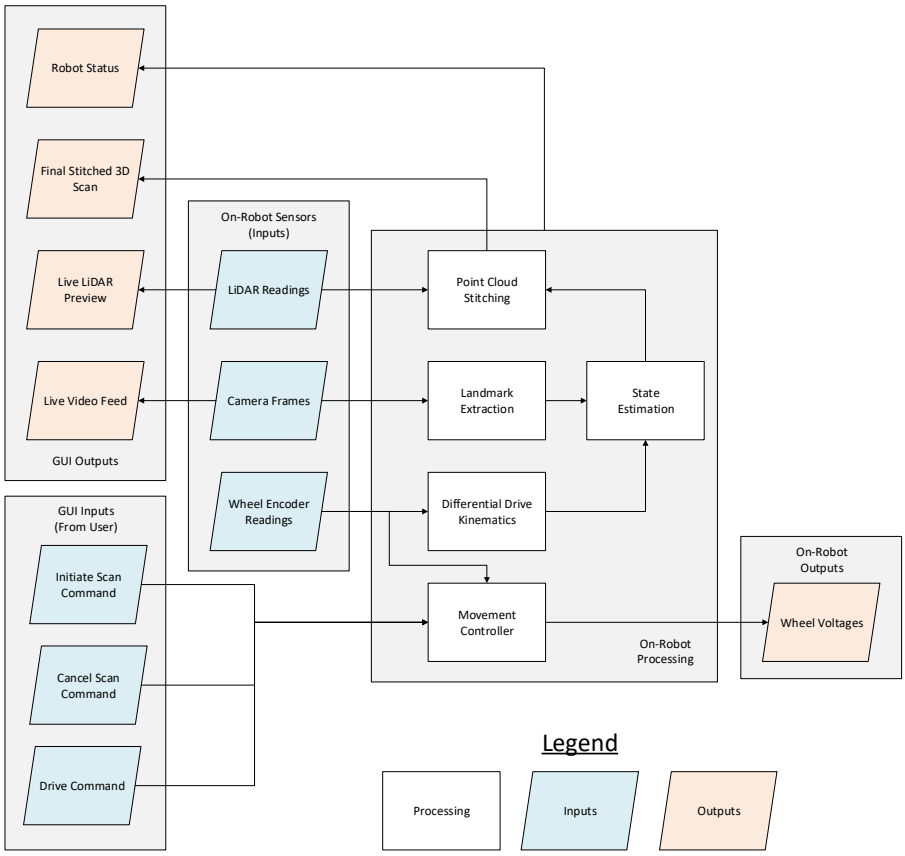

Figure 3: Functional Decomposition

Figure A3 shows the high level data flow throughout the LiDart system.

The system receives data from the user through the GUI alongside inputs from the various sensors onboard the robot (monitored variables). These inputs include: - Initiate scan command - Cancel scan command - Drive command - LiDAR readings - Camera frames - Wheel encoder readings

The system will use the listed inputs to produce or drive the following outputs (controlled variables): - Final stitched 3D scan - Live LiDAR preview (figure ??, look here https://www.slamtec.com/en/Lidar/A1) - Live video feed of the robot's environment - Wheel voltages - Robot status (eg. Fault occured, Ready to scan)

4.4 Subsystem Descriptions

4.4.1 Landmark Extraction

Given a frame from the onboard camera, the landmark extraction subsystem will find all landmarks in the frame and extract information about them. This information will include an ID which can be used to correlate subsequent detections of the same landmark. It will also contain some information about the position, and possibly orientation, of the landmark relative to the camera. The exact form of this information will vary based on the choice of landmark type. Two example landmark types are listed below.

Colored Spheres Uniquely colored spheres could be used as landmarks. They would be easy to detect in a frame, uniquely identifiable based on their color, and would provide information about the direction of the landmark relative to the camera. Given enough landmarks in view these directions could be used to estimate the camera's pose. The size of the sphere could also be used to estimate distance from the camera to the landmark.

Although colored spheres are a very simple example of a possible landmark, they have several drawbacks. One drawback is that for them to remain unique, each sphere would need to be a distinct color from everything else in the environment, this quickly becomes difficult and would quickly increase the rate of landmark misdetections.

Another drawback is the sort of position data provided by using colored spheres, you only get direction and distance, no information about orientation. This increases the number of landmarks required to be in view for an accurate estimation of camera pose.

AprilTags (https://april.eecs.umich.edu/software/apriltag) AprilTags are a 2D barcode system (similar in concept to QR codes) designed at the University of Michigan for easy and robust pose estimation.

Each AprilTag encodes a unique number which can be used to correlate subsequent detections. AprilTags are also robust to misdetection, as their barcode encodes information such that errors can be detected and rejected. Finally, AprilTags provide pose information for each landmark, which greatly reduces the number of landmarks required to be in view for an accurate estimation of camera pose.

4.4.2 Differential Drive Kinematics

The differential drive kinematics subsystem will provide an estimate of the pose of the robot using readings from the wheel encoders. This estimate will accumulate error over time due to being a purely feed-forward estimator, but it can be used to suplement a more complex state estimator (see section 'State Estimation').

Figure ???

The LiDart robot will use a 2 wheel differential drive system, the derivation for the kinematice of such a system is as follows.

Constants: r - wheel radius b - the distance between the centers of the wheels

Variables: V - velocity vector of the robot VL - velocity vector of the left wheel VR - velocity vector of the right wheel omega - angular velocity of the robot wL - velocity of the

left wheel in the direction that it rotates wR - velocity of the right wheel in the direction that it rotates Vx - x component of the robot velocity Vy - y component of the robot velocity

Rolling without slipping: wL = r*omegaL wR = r*omegaR

Kinematics: VR = V + (omega zhat)x(b/2)xhat VR = V + (omega)(b/2)yhat wR*yhat = Vx*xhat + Vy*yhat + (omega)(b/2)yhat

wR = Vy + (omega)(b/2) ¡- y component 0 = Vx ¡- x component

Result: omegaR = (Vy + omega*b/2) / r omegaL = (Vy - omega*b/2) / r

4.4.3 Movement Controller

The movement controller subsystem will receive commands that the user issues via the GUI and control the drive train as necessary.

While the robot is in scanning mode this subsystem will attempt to keep the robot stationary, and as such will ignore any drive commands. While out of scanning mode this subsystem will control the drive train according to any received drive commands.

The movement controller will make use of a closed loop velocity controller to control the wheel speeds of the drive train. In this system the velocity of the wheels will be the measured variables, and the voltage given to the wheel motors will be the actuated/controlled variables.

An example closed loop controller that is suitable for this use case is a PID controller, the discrete time equation for which is as follows.

4.4.4 State Estimation

The state estimation subsystem will receive data from the landmark extraction and differential drive kinematics subsystems and use that data to estimate the pose of the robot alongside the poses of all the landmarks it has seen.

This state estimation can be done using various different algorithms, one example of such algorithm is the particle filter.

4.4.5 Point Cloud Stitching

The point cloud stitching subsystem will take the raw LiDAR readings along with the pose estimation from the state estimation subsystem and use that information to add data to the current 3D scan.

It will do this by first transforming the raw LiDAR readings into the reference frame of the environment, as the raw readings will initially be in the LiDAR sensor's local reference frame. This will be done using the pose estimate provided by the state estimation subsystem.

It will then take these transformed points and stitch them into the current 3D scan. There are many algorithms for point cloud stitching, two examples of such algorithms are Iterative Closest Point and Moving Least Squares.

5 Requirements

This section provides the functional requirements, the business tasks that the software is expected to complete, and the nonfunctional requirements, the qualities that the software is expected to exhibit.

5.1 Functional Requirements

R1: The GUI shall take input from the user through a keyboard and standard pointing device such as a mouse.

R2: There shall be a emergency stop mounted on the robot, which immediately powers down the the robot when pressed.

R3: The robot shall be able to move forward and backwards, and turn in both directions.

R4: The robot shall be stationary if no commands are given.

R5: The robot shall be able to be operated remotely.

R6: The robot shall be able to be connected to over a wireless network.
Rationale: A wireless connection is required for remote operation.

R7: The robot shall be able to be connected to over WiFi.

Rationale: WiFi is a commonly available wireless connection technology, it will be familiar to most users

R8: The robot shall have a power switch.

R9: The robot shall have a means of being charged.

R10: The system shall stitch multiple LiDAR readings into a large point cloud.

R11: The system shall be able to perform state estimation calculations based on landmarks in the surrounding environment.
Rationale: The estimated state of the robot will be used to stitch the multiple LiDAR readings as specified in A10

R12: The robot shall verify closed-loop calculations. (needs to be more specific)
Rationale:

R13: The GUI shall output the scanned data to a 3D model file type.
Rationale: The exported files should be compatible with existing CAD applications.

R14: The GUI shall display live video feed of the environment surrounding the robot. Rationale: This video feed can be used by the remote operator.

R15: The GUI shall display a real-time visualization of the raw scanned data.
Rationale: A real-time representation of the data can be used by the robot operator to ensure the scanner is functioning properly.

R16: The GUI shall display the current state of the system, as depicted in Figure 4.
Rationale: The states can be used by the robot operator to troubleshoot operational issues.

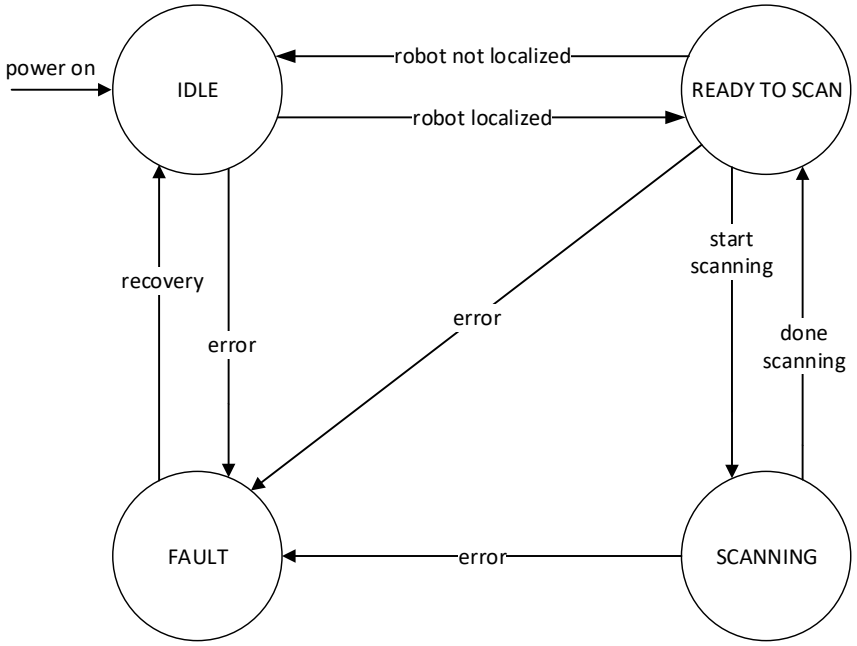

Figure 4: Robot FSM

5.2 Nonfunctional Requirements

5.2.1 Accuracy Requirements

NFR1: The accuracy of the scans shall be within SCAN_TOL. This level of accuracy meets the needs of parties interested in low-cost, easily accessible scanning.

5.2.2 Performance Requirements

NFR2: The robot shall be able to move at a maximum speed of ROBOT_SPEED.

NFR3: The robot shall be able to run for RUN_TIME of time without being charged.

NFR4: The system must be able to scan 1 square meter within SCANNING_TIME of time.

NFR5: The system shall require a maximum of MAX_PROCESSING_TIME to process raw data.

NFR6: The robot shall require a maximum of MAX_RESPONSE_TIME to respond to user input commands.

NFR7: The video feed shall have a minimum resolution of MIN_VIDEO_RESOLUTION.

NFR8: The video feed displayed on the GUI shall have a maximum delay of MAX_VIDEO_DELAY.

5.2.3 Usability Requirements

NFR9: The robot shall be easy to operate with very little knowledge. (needs to be measureable)

NFR10: The robot shall weigh less than ROBOT_MAX_WEIGHT.

NFR11: The robot shall be able to be stored within the dimensions ROBOT_MAX_DIMENSIONS.

NFR12: The GUI shall be easy to navigate. The number of levels of navigation shall not exceed MAX_NUM_LEVELS.

NFR13: The font style and size shall be consistent throughout the GUI. Fonts must be sans-serif and have a minimum font size of MIN_FONT_SIZE.

NFR14: The GUI shall be intuitive to use. Users must be able to execute desired tasks within EXECUTE_TIME seconds of accessing the user interface, for at least every 9/10 tasks.

5.2.4 Error-Handling Requirements

NFR15: The system shall display informative messages that alert the user of errors that have occurred. The messages may provide suggested steps to resolve the error.

NFR16: The system shall log error information. Error information should be specific and descriptive.

5.2.5 Maintainability Requirements

NFR17: Robot parts should be easily replaceable. It should take a maximum of REPLACE_TIME to replace any part on the robot.

NFR18: All hardware and electronic components shall be easily accessible. No special tools must be required to access hardware.
Rationale: This will facilitate activities such as debugging and reprogramming the robot.

NFR19: Standard, off-the-shelf components shall be used where possible.

5.2.6 Portability Requirements

NFR20: The user application shall run on a Windows operating system.

NFR21: The user application must be able to run on a standard personal computer. For example, the application must be able to run on a system with an IntelCore i5 processor and 8 GB of RAM.

5.2.7 Safety Requirements

NFR22: There shall not be any exposed electrical components or wiring.

NFR23: The operator shall be able to stop the robot at any time.

NFR24: The robot shall be placed in a safe-state if communication with the operator is lost.

5.2.8 Standards Requirements

NFR25: The system shall be in conformace with the following standards:

- CSA 22.1:21, Canadian Electrical Code [**?**]
- CSA Z434, Industrial robots and robot systems [**?**]

5.3 Requirements Dependencies

6 Traceability Matrices and Graphs

The purpose of the traceability matrices is to provide easy references on what has to be additionally modified if a certain component is changed. Every time a component is changed, the items in the column of that component that are marked with an "X" may have to be modified as well. Table **??** shows the dependencies of theoretical models, general definitions, data definitions, and instance models with each other. Table **??** shows the dependencies of instance models, requirements, and data constraints on each other. Table 1 shows the dependencies of theoretical models, general definitions, data definitions, instance models, and likely changes on the assumptions.

	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??
T??	X																		
T??																			
T??																			
GD??		X																	
GD??			X	X	X	X													
DD??							X	X	X										
DD??			X	X						X									
DD??																			
DD??																			
IM??											X	X		X	X	X			X
IM??												X	X			X	X	X	
IM??														X					X
IM??													X					X	
LC??				X															
LC??								X											
LC??									X										
LC??											X								
LC??												X							
LC??															X				

Table 1: Traceability Matrix Showing the Connections Between Assumptions and Other Items

7 Likely Changes

LC1:

8 Unlikely Changes

LC2:

9 Development Plan

1.
2.
3.

10 Values of Auxiliary Constants

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_PROCESSING_TIME	Time needed for the robot to process raw scanned data.	10min
MAX_RESPONSE_TIME	Time needed for robot to respond to navigational commands given by user.	1s
MIN_VIDEO_RESOLUTION	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_DIMENSIONS	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

Appendix — Reflection

Throughout the LiDart project, the team will be collectively working on gaining knowledge as well as experience in order to get the project completed. Ranging from programming to electrical to mechanical knowledge each team member will be gaining a lot of knowledge in their respective domains depending on their responsibilities.

Jon and Kareem will need to understand how to get the proper communication between the robot and the machine for the LIDAR sensors. Throughout this process of figuring out the parameters, Jon will be able to expand on his knowledge on communications between the LIDAR sensors and the connected machine by working through the available documenation online. However, Kareem will be working on the controls of the robot by working through trial and error scenarios on the movement of the robot. Even though Kareem will be primarily working through trial and error, he will be using a lot of resources online to help learn how to properly figure out the routes of communication that are going to be used. Jon has picked reading through online documentation since he prefers gathering information from the original source and then applying his understandings. Kareem has picked working through trial and error as that will be the best approach to seeing how the different parameters will affect the robot and its controls. Reading through documentation would not be very effective when trying to get the right controls for the robot as it would be based on theoretical values which won't directly apply to the robot. Through theory, values can be used as a starting point.

While the software is being configured, Neeraj will be working on making the chassi for the robot. Making the chassi for LiDart will require a lot of knowledge based on weight distribution and stabilizing the different moving components. The robot will have a variety of components such as the LIDAR sensors, mounting components and motors to name a few. A lot of theory could be applied to designing the robot, but there will be a lot of trial and error with the process. That is why Neeraj is opting for using the trial and error method to figure out the best design for the robot.

Lastly, the electrical components will be a very critical domain of the robot. Getting the correct electrical equipment depending on the needs of LiDart will require a lot of work with the rest of the team. This is due to how LiDart needs to be configured. The required components such as the motor will require input from Kareem, Jon and Neeraj. The motor will depend on the weight of the chassi (Neeraj) alongside how the motor will be controlled (Kareem and Jon). The equipment picked will affect the mechanical design as the size of the equipment will be very critical to the design of LiDart. Michaela will be exploring online documentation to find the correct equipment that best matches what is required for the robot. Michaela figured that by searching online for the proper equipment she will be able to find the best matches for the requirements needed. This would also be the most effective strategy due to the limitation on the budget that has been given.