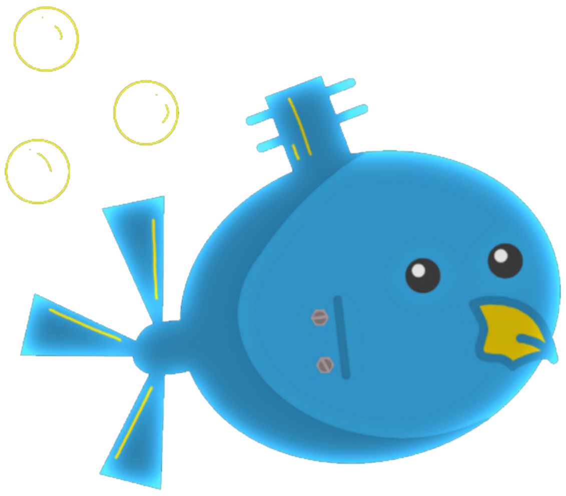
California State University, Los Angeles

Software Requirements Document



RoboSub

Members
Thomas Benson, David Camacho, Bailey Canham, Brandon Cao, Roberto Hernandez, Andrew Heusser, Hector Mora-Silva, Bart Rando, Victor Solis

Sunday 5th March, 2023

Table of Contents

Revision History

Version	Changes	Date
1.0	First release of Software Requirements Document.	9 December 2022
1.1	Style Changes.	5 March 2023

Table 1: Revision History

1 Introduction

The computer science senior design team for RoboSub will work in conjunction with another team of 10 consisting of undergraduates from the Mechanical and Electrical Department of California State University, Los Angeles. The goal of the two teams is to design, manufacture and program an AUV (Autonomous Underwater Vehicle) to compete in the annual RoboSub [2] competition hosted by RoboNation [1].

There are common requirements across previous RoboSub competitions; these requirements include: autonomous navigation, movement, object detection, object classification, and object manipulation.

This year, the RoboSub senior design team inherited the design of the previous year's senior design team. They started the mechanical, electrical and software design for the Autonomous Underwater Vehicle: Lanturn. The RoboSub Computer Science team will work with the Mechanical and Electrical Senior Design team to finish the work done by the previous year.

This document will give an overview to both users and developers of the expected behavior and software requirements for Lanturn.

1.1 Purpose

1.1.1 Purpose of the Document

This document will list the software requirements imposed by RoboNation in the 2023 RoboSub competition and any self-imposed requirements. Self-imposed requirements might include requirements for development, design improvements, system modularity, or quality of life improvements.

- Section 3 will give requirements for setting up an external interface
- Section 2 will give an overall description of the requirements
- Section 4 will explicitly define specific requirements
- Section 5 will define non-functional requirements
- Section 6 will define any other requirement that does not fit in any other section

1.1.2 Purpose of the Product

The product being developed is software for the Autonomous Underwater Vehicle, Lanturn. Lanturn is to compete in the 2023 RoboSub competition hosted by RoboNation. The RoboSub 2022 competition was hosted at The University of Maryland in their Olympic pool. Other years, the competition is hosted at the TRANSDEC center in San Diego. Since the competition for 2022 was a success in the University of Maryland, a similar environment is expected for the 2023 competition.

Lanturn should be able to autonomously navigate and execute tasks within a course. The layout of the course and a description of the tasks is published by RoboNation by the end of March of the year of the competition. For now, the software requirements and assumptions on what tasks Lanturn will need to do will be based on the 2022 Mission Handbook.

1.2 Intended Audience and Reading Suggestions

This software document is intended for users, developers, documentation writers and testers.

- Documentation Writers

  - The 2022 RoboSub Team Handbook
  - section 4 to understand the requirements of the system and what documentation might exist in other places
  - if they are responsible for document one software submodule, they can read their own section
  - section 5 for requirements that do not fit in other categories
- Developers
  - The RoboSub Software Design Document
  - The appropriate README in the git repository hosted on GitHub
  - section 4 to get a complete understanding of the complete system (even if they are only responsible for one software module)
  - section 5 to get an understanding of restrictions they should keep in mind while developing the software
- Users
  - The 2022 RoboSub Team Handbook
  - section 3 to understand what is required to interface with Lanturn
- Testers
  - The 2022 RoboSub Team Handbook
  - section 3 to understand what is required to interface with Lanturn
  - section 4 to understand the functionality that needs to be tested

1.3 Product Scope

The goal of the software for Lanturn is to autonomously accomplish tasks at the RoboSub competition.

To accomplish the set goal, the software will have five modules:

1. Autonomy
   - Navigation
   - Decision making
2. Computer Vision
   - Object detection
   - Object classification
3. Controls
   - Sensor data acquisition
   - Actuator control
4. Mapping
   - Environment mapping
5. Localization

- Vehicle localization in map

1.4 Definitions, Acronyms, and Abbreviations

A list of Definition, Acronyms, and Abbreviations can be found at the end of this document in Appendix A: Glossary.

1.5 References

2 Overall Description

The requirements for the software developed for Lanturn will be largely derived from version 3 of the 2022 RoboSub Mission Handbook. The mission handbook states the logistics of the competition, the rules during the competition and the tasks to be done by any AUV participating.

Along with the requirements derived from the 2022 Mission Handbook, additional requirements will be added based on what was learned from previous years and from suggestions from the team's advisor(s).

As of the publish date of this document, the 2023 Mission Handbook has not been released. In later versions of this document, the requirements will be updated for every version released of the 2023 Mission Handbook.

2.1 System Analysis

The goal of the project is to develop software that will accomplish as many tasks as possible as quickly as possible with the sensors and actuators available on the submarine.

1. Problems
    - 100% autonomous operation
    - Submodule checks
    - Data accuracy
    - Testing Limitations
    - Self-diagnosis
    - Lanturn manufacturing time
2. Solutions
    - Behavior Trees
    - Global watchdog service
    - Local watchdog services
    - Sensor fusion
    - Simulation

2.2 Product Perspective

The RoboSub Organization states the following on their website: "The behaviors demonstrated by these experimental AUVs mimics those of real-world systems, currently deployed around the world for underwater exploration, seafloor mapping, and sonar localization, amongst many others.” [1]

Since Lanturn is to be completely autonomous, the system itself is completely isolated from all other systems, an exception is when it is tethered to an external computer for real-world testing and (re)configuration.

Concepts used by all or some modules in the AUV can be applied to computer networks, other robotics projects, surveillance systems, ROVs (remote operated vehicle), UAV (Unmanned aerial vehicles), video-game programs (programming behavior of NPCs), electronics projects, systems designed for underwater exploration (or general exploration of territory), systems designed for planetary exploration, cartography, localization, any system that will benefit from more accurate sensor data, agriculture (autonomous behavior) and other software systems.

2.3 Product Functions

Functionality of the software will be derived from the requirements of the 2022 RoboSub Mission Handbook. The software will be designed to accomplish the tasks listed in the mission handbook.

1. With Moxy (Coin Flip)
2. Choose Your Side (Gate)
3. Collecting (Bins)
4. Make the Grade (Buoys)
5. Survive the Shootout (Torpedoes)
6. Cash or Smash (Octagon)
7. Follow Orange Guide Markers

None of the functions (tasks) laid out in the Mission Handbook are required (except Choose Your Side), however, the more tasks the AUV can do, the more points will be scored for the mission. The assumption that will be made, for now, is that all tasks will at least be attempted.

Other functions that will be implemented are:

- Autonomous navigation
- Autonomous decision making
- Object detection
- Object classification
- Object relative position estimation
- Vehicle localization
- Vehicle state estimation
- Vehicle state control
- Vehicle state monitoring
- Vehicle state logging
- Environment mapping

The functionality listed above is all the general functionality that will enable Lanturn to accomplish any one of the competition tasks.

2.4 User Classes and Characteristics

User Class	Description
Swimmer	Will start Lanturn and monitor its state.
Tester	Will verify that Lanturn is working as expected.
Developer	Will develop the software for Lanturn.

Table 2: User Classes and Characteristics

2.5 Operating Environment

The hardware that will be used for the development of Lanturn is the following:

TX2 Module with Development Kit Carrier Board

- Ubuntu 20.04
- NVIDIA JetPack 4.5
- NVIDIA CUDA 10.2
- NVIDIA cuDNN 8.0
- NVIDIA TensorRT 7.1
- ROS2 Foxy Fitzroy
- OpenCV 4.5.1
- Python 3.8.10+
- C++ 17

The software will be developed on a computer running Ubuntu 20.04 and using the ROS2 framework.

Teensy 4.1

- PlatformIO
- Arduino Framework
- Micro-ros-plarformio
- C++ 17

2.6 Design and Implementation Constraints

All software (except interfaces with sensors and actuators) must be implemented in Ubuntu 20.04, as that is the only operating system that is supported by the Nvidia TX2 module. By extension, any software designed to be run in the TX2 module will also have to be a ROS2 package.

All software designed to interface with sensors or actuators must be compatible with communication protocols available for the Teensy 4.1 microcontroller.

All software designed must follow the limitations set by the policies listed in the Mission Handbook provided by RoboNation.

If, the ME/EE team for RoboSub cannot design or manufacture any of the actuators, that is, the claw, torpedoes, or dropper, then the software team will not be able to test or implement any software designed to execute any functionality requiring missing actuator(s).

2.7 User Documentation

Each sub-module of the system will come with documentation in the form of a README.md on the project's GitHub repository.

The structure of the README's will be as follows:

1. Overview: An overview of the role played by the submodule
2. Dependencies: A list of dependencies required to develop and run the sub-module
3. Installation: Instructions on how to install any software needed for the submodule
4. Setup: Instructions on how to setup the environment for either development or deployment
5. Usage: Instructions on how to use the submodule on the submarine and on the field
6. Problems and Fixes: Common issues encountered and, either, how to fix or work around the issue

2.8 Assumptions and Dependencies

The assumption while developing the software for this system is that the rules for the 2023 RoboSub competition will largely remain the same as the previous year.

2.9 Apportionment of Requirements

New software requirements may arise from the release of the 2023 RoboSub Mission Handbook, and it is the responsibility of the team to adjust the software system to any new requirement.

3 External Interface Requirements

This software does not currently have any custom interfaces, but there are interfaces provided by ROS2 which can be used to configure or control the submarine in real time.

3.1 User Interfaces

Currently, there are not any graphical user interfaces designed for interfacing with Lanturn. However, ROS provides a general way of interfacing and interpreting data from a ROS system.

RQT

RQT is a software framework for creating graphical user interfaces with the running ROS system. It behaves like a module within the ROS system and comes with implementation of common functionality in the form of plugins.

Above is a sample configuration of several plugins that might be useful to have when interpreting data from Lanturn's ROS system.

Of note is the drop-down menus at the top left. The drop-down menu shown when selecting “Plugins” gives a list of plugins to open. Once a plugin is selected, it will open in the RQT window and then can be dragged around to reconfigure the layout of the plugins.

More information can be found on their website http://wiki.ros.org/rqt.

RVIZ

Rviz is a 3D visualization tool for ROS that can be used to visualize aspects of a robot model. It provides different panels that can help you visualize position, orientation and camera data in the running ROS system.

More information can be found on their website http://wiki.ros.org/rviz.

3.2 Hardware Interfaces

Fathon-X

Although Lanturn is meant to be entirely autonomous, it is beneficial to be able to connect to the ROS system while testing new components for fast testing and real-time development.

Lanturn will have a Fathom-X, an interface board sold by Blue Robotics, onboard that can be used to establish a connection to an external computer. To establish the connection, two of the Fathom-X boards are required: one onboard Lanturn and the other connected to the external computer through an Ethernet port.

Fathon-X Features

- 80 Mbps Ethernet over two wires (per our own bandwidth testing)
- 300m+ tether length capability
- Plug-and-play with no setup involved
- Onboard switching power supply with 7-28V input range
- USB Mini-B connector for powering directly from a computer on the topside
- Indicator LEDs for power, link, and data
- Included 6" Ethernet cable for connection to onboard computer

More on the Blue Robotics website: https://bluerobotics.com/store/comm-control-power/tether-interface/fathom-x-tether-interface-board-set-copy/

Fathom ROV Tether The Fathom ROV Tether is another product from Blue Robotics and can be connected to one of the penetrators on Lanturn to host the connection between the two Fathom-X Boards.

3.3 Software Interfaces

The external computer that is connected through the Tether should be running on the same software as Lanturn.

- Ubuntu 20.04
  - ROS2 Foxy (Middleware)
  - OpenSSH

3.4 Communications Interfaces

The external computer will use OpenSSH to establish an SSH connection with Lanturn and will use the same Message Passing interfaces as the ROS system in Lanturn (i.e., ROS CLI tools and ROS APIs).

4 Requirements Specification

This section contains all the necessary software requirements with enough detail to allow designers to accurately design the software to satisfy those requirements, and to allow testers of the software to verify that all requirements have been satisfied.

4.1 Functional Requirements

There will be six modules that will be implemented; each with a different set of related requirements.

The six modules that will be implemented will be:

1. Autonomy
2. Computer Vision
3. Controls
4. Localization
5. Mapping
6. Watchdog

These are the following responsibilities of each sub-module:

1. Autonomy
    - (a) The autonomy module shall manage time in a run through the course
    - (b) The autonomy module shall send current state to system logs
    - (c) The autonomy module shall read and interpret mapping data
    - (d) The autonomy module shall read and interpret localization data
    - (e) The autonomy module shall read and filter computer vision data
    - (f) The autonomy module shall send command to control claw
    - (g) The autonomy module shall send command to shoot torpedoes
    - (h) The autonomy module shall send command to release dropper
    - (i) The autonomy module shall send a heartbeat to the watchdog service
    - (j) The autonomy module shall know the task being executed
    - (k) The autonomy module shall autonomously navigate map
    - (l) The autonomy module shall position submarine in a desired position
    - (m) The autonomy module shall orient submarine in a desired orientation
    - (n) The autonomy module shall be able to center with objects
2. Computer Vision

   (a) The Computer Vision shall send a heartbeat to the watchdog service
   (b) The Computer Vision shall publish raw images from front camera
   (c) The Computer Vision shall publish raw images from bottom camera
   (d) The Computer Vision shall receive images from front camera
   (e) The Computer Vision shall receive images from bottom camera
   (f) The Computer Vision shall detect and classify all task objects
   (g) The Computer Vision shall publish bounding boxes of objects
   (h) The Computer Vision shall provide distance from objects
   (i) The Computer Vision shall calculate angle of incidence of objects

3. Control
   (a) The Controls shall send a heartbeat to the watchdog service
   (b) The Controls shall read and publish data from Bar30 barometer
   (c) The Controls shall read and publish data from Sonar
   (d) The Controls shall implement a PID Library
   (e) The Controls shall generate PWM values that will move submarine to a desired position and/or orientation
   (f) The Controls shall output PWM values to thrusters
   (g) The Controls shall clench and release a mechanical claw on command
   (h) The Controls shall shoot torpedoes on command
   (i) The Controls shall release ball from dropper on command

4. Mapping
   (a) The mapping module shall send a heartbeat to the watchdog service
   (b) The mapping module shall subscribe to all computer vision data
   (c) The mapping module shall subscribe to Sonar data
   (d) The mapping module shall implement a Kalman Filter
   (e) The mapping module shall generate a map of environment
   (f) The mapping module shall position all task objects in map
   (g) The mapping module shall publish map

5. Localization
   (a) The localization module shall send a heartbeat to the watchdog service
   (b) The localization module shall subscribe to IMU data topic
   (c) The localization module shall subscribe to Barometer data topic
   (d) The localization module shall subscribe to both camera topics
   (e) The localization module shall subscribe to DVL data topic
   (f) The localization module shall subscribe to map topic

(g) The localization module shall position and orient submarine inside the map

(h) The localization module shall publish localization data

6. Watchdog

   (a) The watchdog module shall subscribe to all heartbeats from all modules

   (b) The watchdog module shall shutdown submarine if major components fail

   (c) The watchdog module may have fallback module configurations

4.2 External Interface Requirements

There are no external interface requirements to interact with Lanturn, aside from those listed in section 3. However, it will benefit the user to be familiar with the messages that are passed between the ROS modules in Lanturn.

Logical Data Requirements

4.3 Logical Data Requirements

Not applicable.

4.4 Design Constraints

The following are the design constraints for Lanturn:

- Sensors
  - Camera: Front and bottom
    - 1920x1080 resolution
    - 20-30 FPS
    - 80 degree vertical field of view
    - 120 degree horizontal field of view
  - Barometer
  - Doppler Velocity Log (DVL)
  - Sonar
  - Inertia Measurement Unit (IMU)
- Actuators
  - Thrusters
  - Claw
  - Torpedoes
  - Dropper
  - Transducer

Component	Descriptions
GPU	256-core NVIDIA Pascal GPU architecture wtih 256 NVIDIA CUDA cores
CPU	Quad-core ARM Cortex-A57 MPCore processor with 64-bit architecture
Memory	8GB 64-bit LPDDR4 memory
Storage	32GB eMMC 5.1 flash storage
Power	7.5/15 Watts power consumption

Table 3: Specifications of the TX2 module

TX2 Module Specifications

TX2 Development Kit Carrier Board Specifications

The Jetson TX2 Developer Kit carrier board (P2597) includes [10]:

- Micro-USB to USB A cable
- (2x) WLAN/Bluetooth antenna

Teensy 4.1

The pjrc website [7] which develops the Teensy 4.1 boards provides the following specifications:

- ARM Cortex-M7 at 600 MHz
- Float point math unit, 64 & 32 bits
- 7936K Flash, 1024K RAM (512K tightly coupled), 4K EEPROM (emulated)
- QSPI memory expansion, locations for 2 extra RAM or Flash chips
- USB device 480 Mbit/sec & USB host 480 Mbit/sec
- 55 digital input/output pins, 35 PWM output pins
- 18 analog input pins
- 8 serial, 3 SPI, 3 I2C ports
- Ethernet 10/100 Mbit with DP83825 PHY

5 Other Non-Functional Requirements

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Software Quality Attributes

5.5 Business Rules

6 Appendices

6.1 Appendix A: Glossary

6.2 Appendix B: Analysis Models

6.3 Appendix C: To Be Determined

References

[1] RoboNation. Robonation. https://robonation.org/, Dec 2022. Accessed: 2022-12-09.