Thetis Software

Project Thetis, Embry-Riddle Aeronautical University

Software Requirement Specification

V 1.0

Introduction

This software is being designed to control Project Thetis's robot and sand sorting methods. The robot is to filter microplastics from beach sand, disposing of them in a housing unit coded "The Doghouse." The software will be run from a computer on this robot in real time, operating without the need for human interaction. It will only be out-of-the box compatible with the robot Thetis is to design.

General Description

Driving, sand sorting, identifying people and objects, and maneuvering in a coordinated fashion are all general goals of this robot, with the responsibility of such goals falling in part or in whole to the software on the robot. To drive, coordinate sand sorting, maneuver to avoid objects, and in general perform operations, software shall be developed and deployed to control the robot.

Requirements

To properly document the expected behaviour of the robot, requirements shall define the desired traits. These will vary from overarching functionality to specific software interface requirements, and shall be testable either in software or hardware environments, with atomic test results.

Functionality

Dictating what the robot shall do in particular cases is vital in software design, so the following requirements will determine what shall be programmed, and what exceptions will need to be handled.

Battery Level

1. The robot must be able to detect battery level and report levels to the user.
2. The robot shall pause sand sorting operations when the battery is low.
3. The robot shall return to the Doghouse when low battery is detected.
4. The robot shall determine if it can return the current battery life, and report such to the software.

Sand Sorting

5. The robot shall control sand sorting mechanisms.
6. The robot must be able to detect fill volume of microplastics and report levels to the software.
7. The robot must be able to pause beach cleanup when the fill volume is saturated.
8. The robot shall only dispose of microplastics when at the Doghouse.
9. The robot shall return to the Doghouse when software detects high microplastic levels.

Drive Operations

10. The software shall coordinate the drive motors in a coordinated fashion, allowing for driving operations.
11. The robot shall be able to avoid collisions with docks, people, and wildlife.
12. The robot shall avoid turtle nests that have been marked with markings.
13. The robot shall be able to avoid the ocean (~10m).

14. The robot shall obtain knowledge of tides and avoid the water levels.
15. The robot shall obtain knowledge of king tides and avoid sand sorting during such events.
16. The robot shall detect and relay its position to the DogHouse.

Usability

Alongside functionality, usability is important. It determines how the software will interact with the user, meaning it is the only thing that the user will interface with once deployed.

Terminal Execution

1. The user shall be able to run the program through the terminal.
2. The user shall be able to select drive operations and control basic robot functions through the terminal.
3. The user shall be able to end all other instances of the software through the terminal.

GUI Execution

1. The user shall be able to control the robot through a JavaFX or similar alternative GUI.
2. The user shall be able to select and execute drive operations through the GUI.
3. The user shall be able to see camera and LIDAR object recognition and video feed through the GUI.
4. The user shall be able to control the robot via a joystick or keyboard through the GUI.

Interfaces

Determining how the software shall interface with the hardware, as well as other components of the software is vital to streamline the coding process. By defining these interfaces, they can be reused across software, and will allow for simpler software development.

Hardware Interfaces

1. The software shall control the drive motors .
2. The software shall be able to control the sand sorting servos and motors.
3. The software shall be able to read images/data from the Lidar and/or cameras.
4. The robot shall shut off in the event of an overvoltage event.
5. The robot shall shut off or return to the doghouse in the even of a collision.
6. The robot must be able to detect any physical issues (Broken tracks, clogged scoop), and act accordingly, alerting the user if present.

Software Interfaces

1. The robot shall be able to communicate with the “Doghouse.”
2. The robot shall be able to display information pertinent to its hardware and be controlled via a screen.
3. The robot shall be able to detect voltage levels and enact proper protocol.

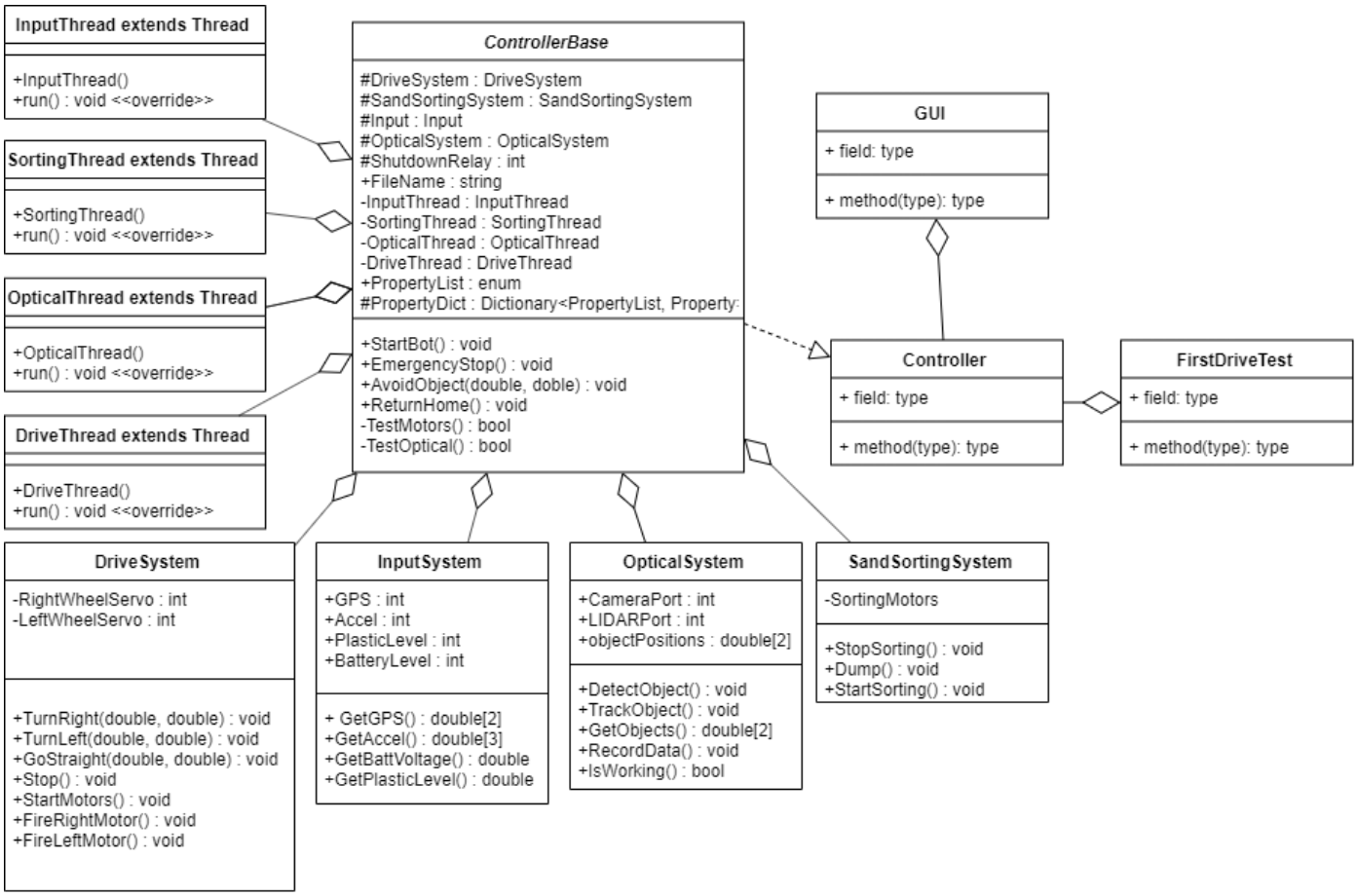
4. The robot shall be able to stop all processes in the event of a manual override.

More Software Interface Requirements

To adequately render camera and LIDAR objects, it is required that this software be a multi-threaded application, leveraging a multi-core CPU such as the one on the RPI that will be utilized. In order to prevent cross threading, four main systems have been defined: inputs, sand sorting, optical tracking and detecting, drive functions.

Each system will operate independently from all the others, receiving commands from the ControllerBase. This does mean that invokes will be necessary; however, the performance gain is worth it.

The following UML dictates the overarching organization that is to be used in the software, allowing code reuse and easy modification:



Performance

1. The software shall search for objects at a minimum of 1 Hz.
2. The software shall track current objects at a minimum of 20 Hz.
3. The software shall never pause execution of the main thread, nor shall the main thread be utilized for any computation.
4. The software shall make the GUI on a separate thread than the Controller.
5. The software shall handle all foreseeable errors and exceptions without terminating.
6. The software shall not print unnecessarily to the terminal.

Constraints

1. The robot must be able to be restricted in operation to a predefined area (geofence).
2. The robot will be able to limit its speed to a predefined value.
3. The robot's sand collecting method must be restricted to a certain depth.

Conclusion

In order to provide functionality to Thetis's beach cleaning robot, software shall be built to control it. This software will operate within the constraints listed above, providing it with simple interfaces and concrete requirements.