Software Design Document

Project: DPM Final Project

Document Version Number: 2.0

Author: Zakaria Essadaoui

EDIT HISTORY:

[29/10/2018] Zakaria: Created the document + added flow chart

[30/10/2018] Zakaria: Added UMLs and class hierarchy+ comments

[05/11/2018] Zakaria: Added the Week 3 subsection on the document.

Tables of contents:

A. Week 2:

1. Overview:

Our software design will be mainly constructed from the subparts that were developed during the research phase of the project in the labs. We will be using all those pieces (with minor changes) in addition to new classes to perform the tasks required. Since the final project doesn't require searching, we will not be using that part from Lab5, we will only be using the color detection feature from this lab. We will also need the following parts: Odometer, Localization and Navigation.

2. Navigation:

This subsystem is used to go to a desired point in the map. It always interacts with odometer class to get its position on the grid to know by how much it should move. In our final project, this system will be used in a similar fashion in order to navigate to the tree and back to our starting corner.

In order to visualise this system, we included the UML diagram that we had from the Navigation Lab:

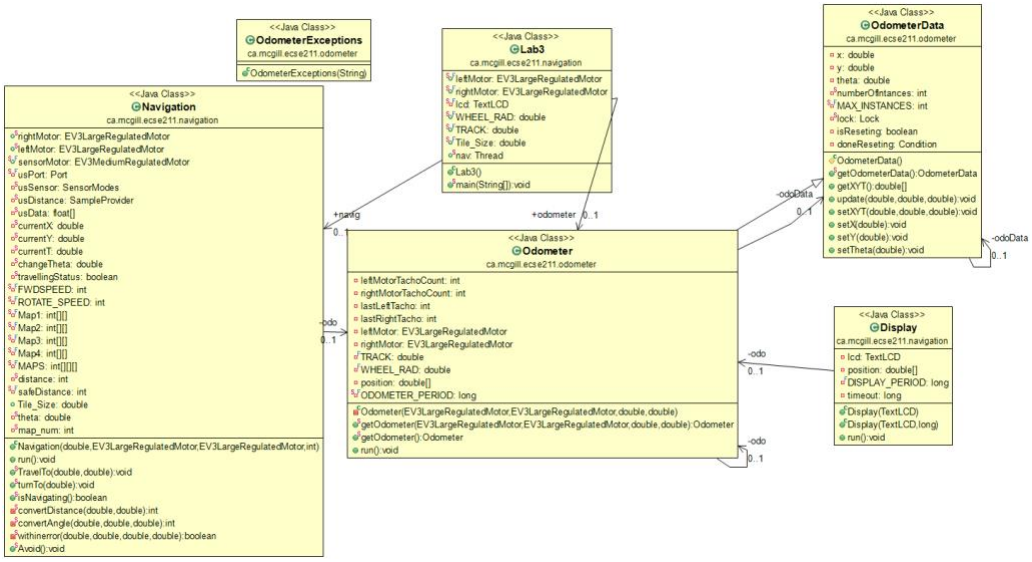

Figure 1: UML diagram from the Navigation Lab

3. Localization:

This system is used to perform the ultrasonic and light localization. In this lab, we had two ways of performing the ultrasonic localization (falling edge and rising edge). Since we don't use both ways and falling edge works better in our design, we decided to delete the rising edge part. For the light localization, we decided to change the design to a use two light sensor instead of one, we noticed that this gave us more precision. We also decided to add poller classes to have

cleaner code structure. By doing so, we would have three pollers for these tasks, “LeftLightPoller”, “RightLightPoller”, “UltrasonicPoller”.

We included the UML diagram from this lab in the following figure:

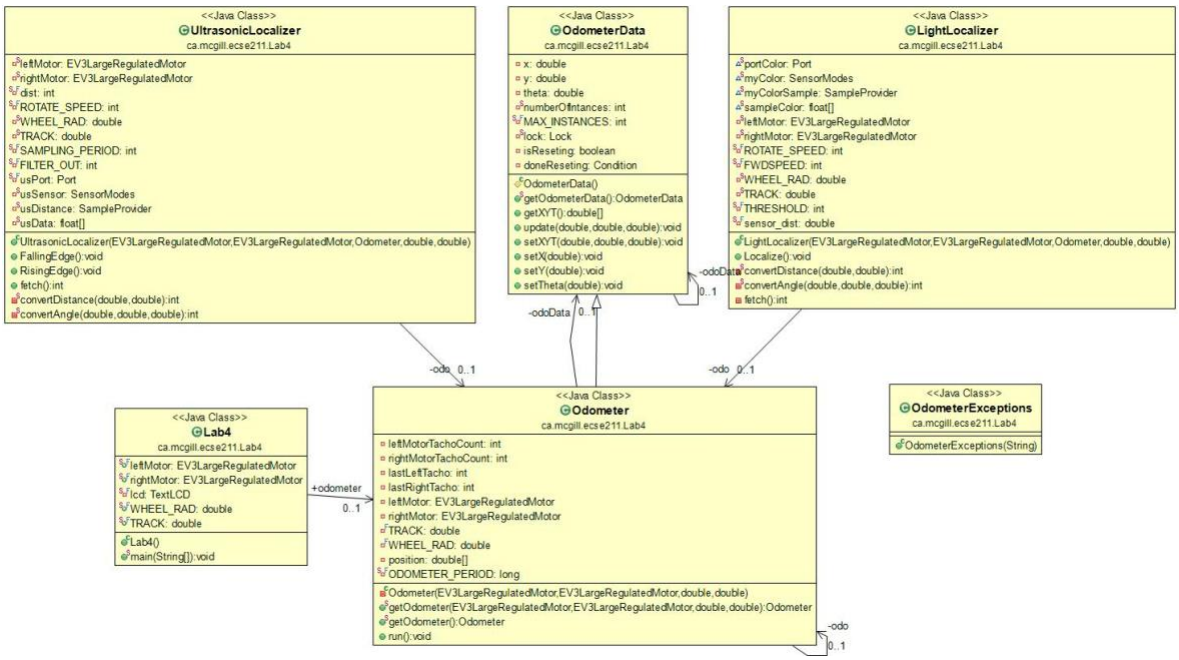

Figure 2: UML diagram from the localization lab

4. Search and Localize:

The main part that will be used from Lab 5 isthe color detection. Basically, we will create an instance of this class and call the detect method to get the ring color. We can see all the fields and classes from this class in the following figure:

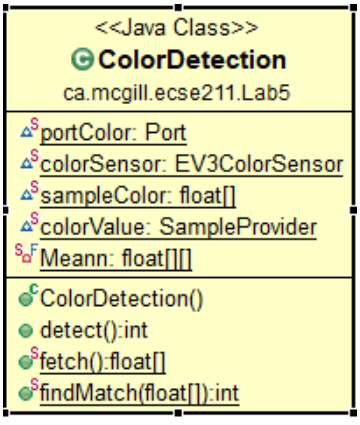

Figure 3: Summary of the Color Detection class

5. Project design:

For now, our software is still simple since we are still writing the initial code and building an effective prototype and thus we don't know what kind of issues we would have to fix with software.

The following flowchart explains the biggest steps of the

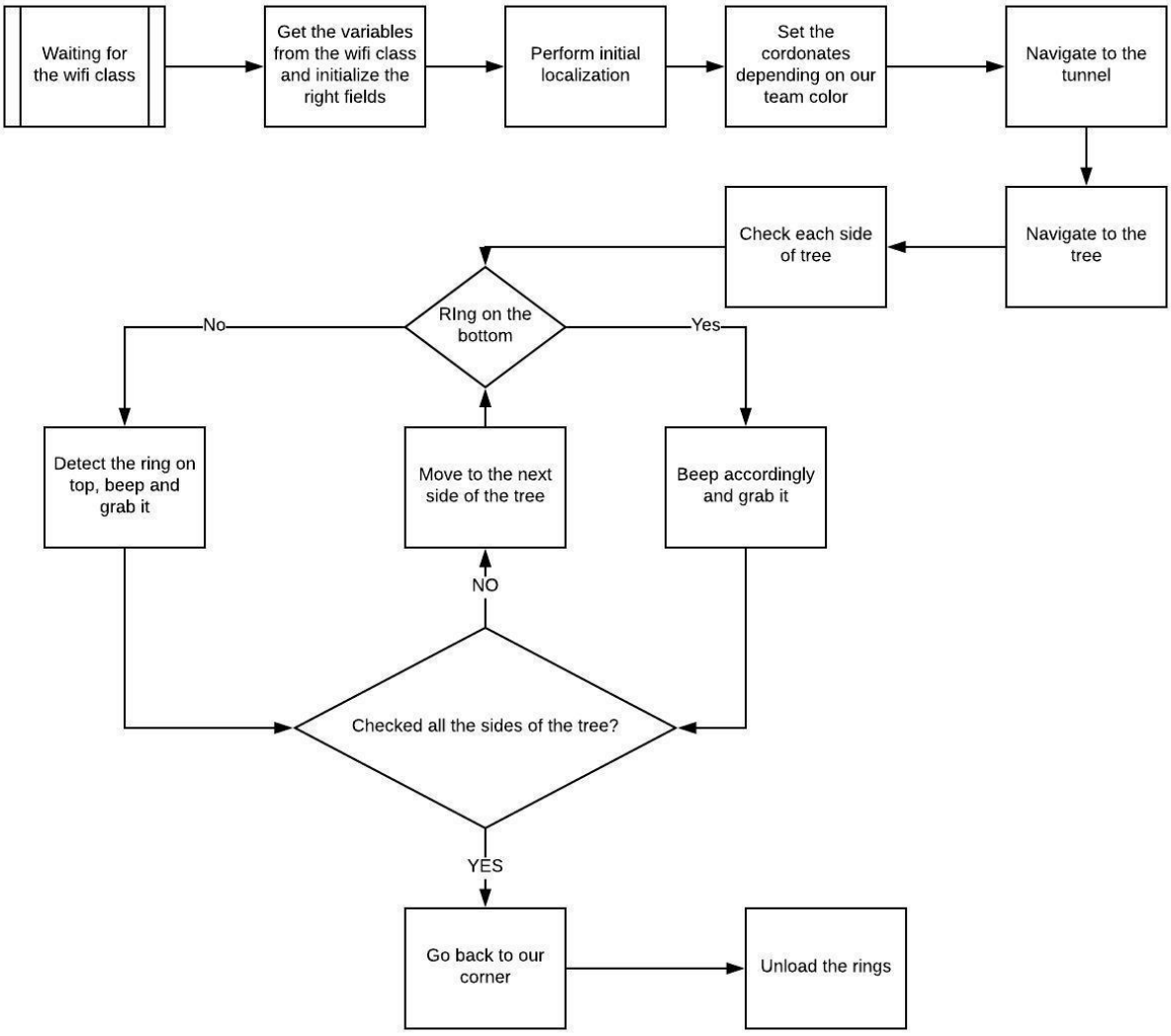

Figure 4: Preliminary flowchart for the project

We also included a preliminary class hierarchy that shows the basic interactions between the class. There are a few changes to note compared to the other labs:

- We are now using poller classes for the sensors instead of implementing them directly in the classes.

- We will be using two light pollers for light localization.
- The navigation class will call the light localization class from time to time

Preliminary class hierarchy:

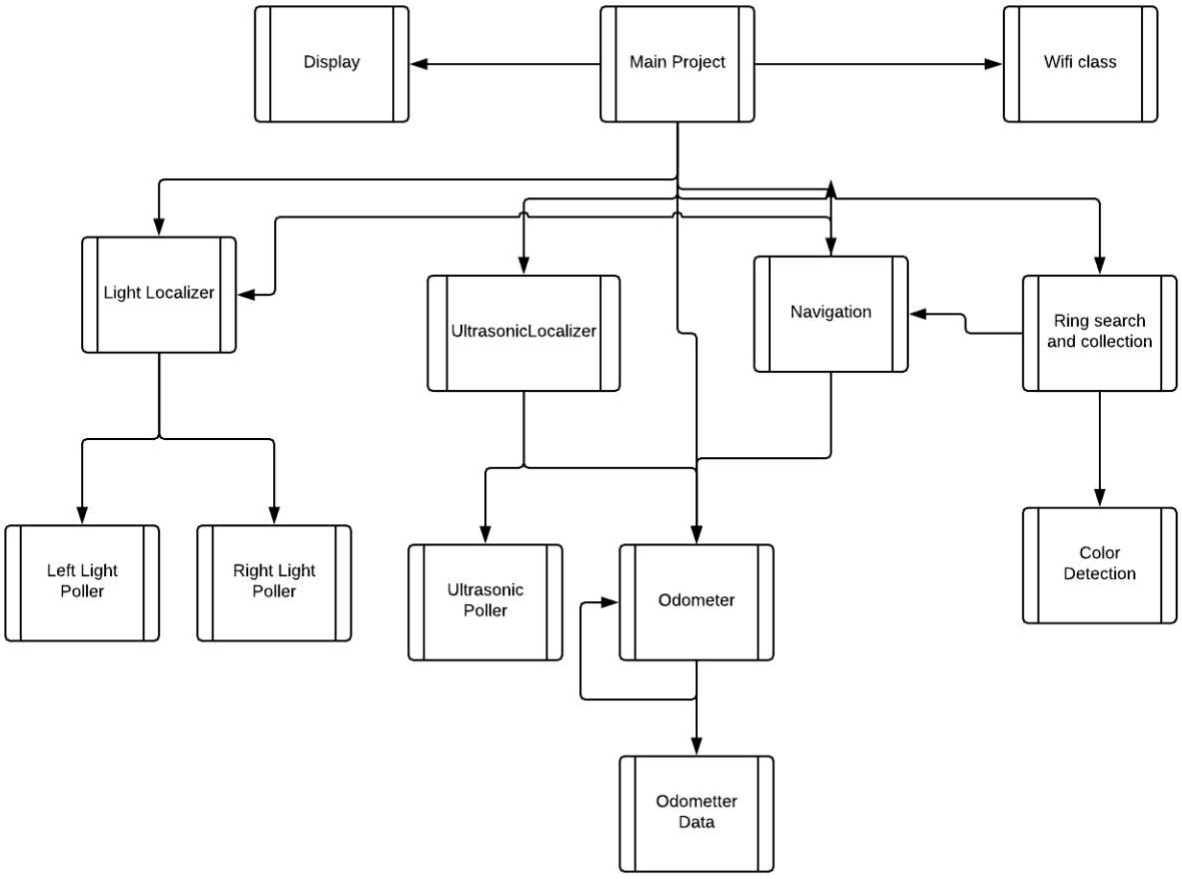

Figure 5: Preliminary class hierarchy for the project

6. Objectives for this week:

- Developing an accurate light localization routine that accounts for all the particular cases where the sensors are on the wrong line since we will be using it while travelling.
- Developing the first prototype of the ring search and collection class when the hardware will be available.
- Dealing with false positives from the color detection class coming from seeing the tree stand as being a yellow ring.

B. Week 3:

1. Overview and progress after week 2:

As was mentioned in the week 2 part, we focused this week on redesigning the software and our classes to get a better structure. We also created the first algorithm for grabbing the rings and ran some tests using it. Some issues with the color detection were discovered (mainly due to the distance of the sensor from the rings and will be dealt with). We will discuss our current software design and all the major changes.

2. Software design as of week 3:

A part of redesigning our software was the creation of two polling classes:

- One for the ultrasonic sensor which takes care of acquiring data from the sensor and also deals with false negative through our filter.
- The second one is for the light sensor, it takes care of the acquiring data from the sensor and detecting the lines using a differential filter.

We also developed the full algorithm for the game play (depending on which team we are); However, some changes must be made to that algorithm since the tree will be put in an intersection instead of the center of a tile.

We also started working on a new algorithm for line detection using the light poller class. This algorithm is still under developed and we are also still discussing whether we should detect black lines using the ratio of readings or the difference of the readings.

The algorithm used for grabbing rings was developed under the tester package. This preliminary code worked well and it was possible to retrieve a ring using. It is not yet implemented in the RingGrabbing class; However, this will be done shortly after the hardware is finalized.

A preliminary API has also been developed and have been generated using Doxygen. It can be accessed in the following folder: Week 3\ API_week3\html. Please select the file named: "_final_project_8java" to access. You will then be able to navigate through all the classes.

3. UML diagram:

The following figure shows the UML diagram of our design:

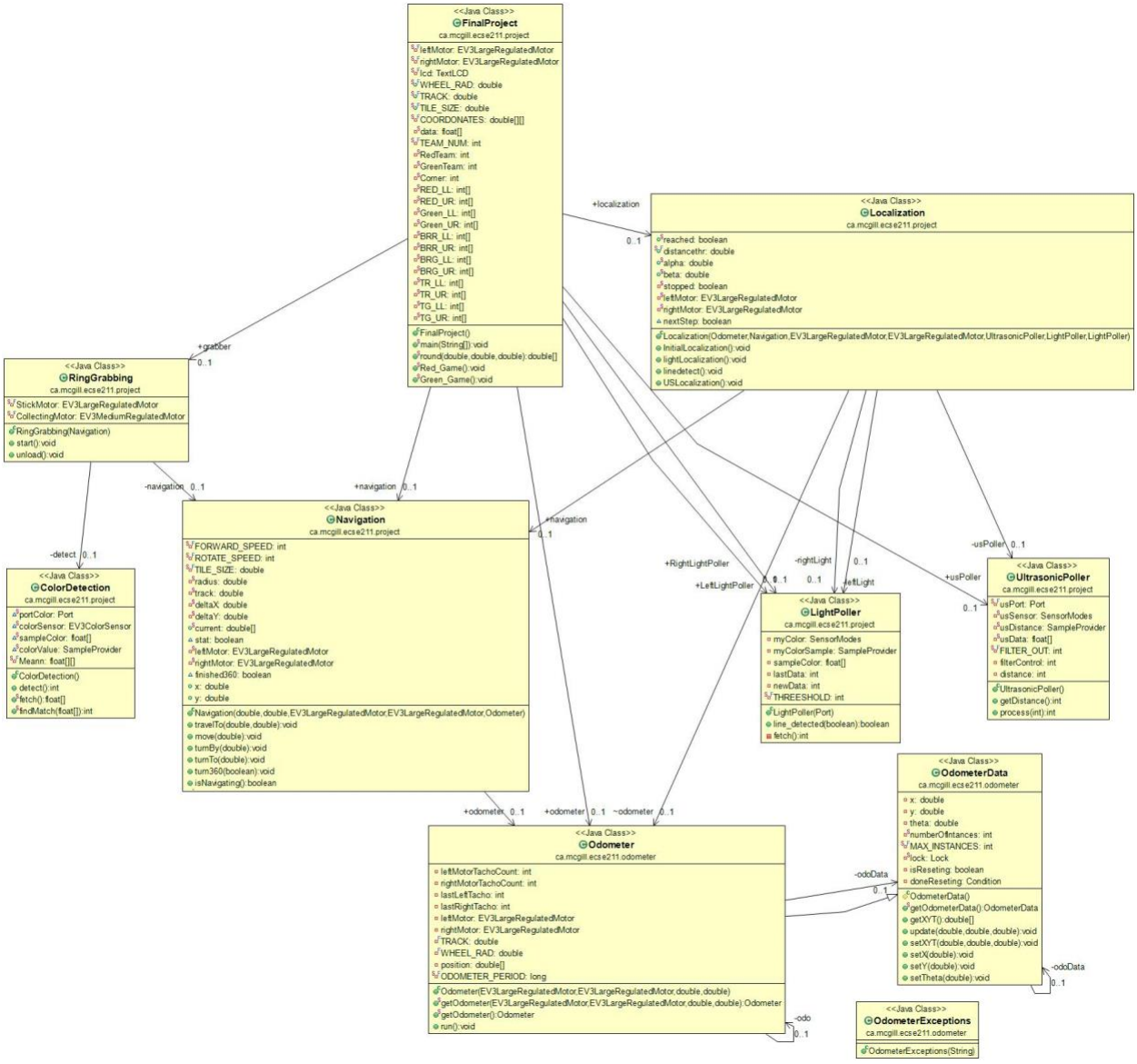

Figure 6: UML diagram of our project as of week 3

This structure is very similar to the class hierarchy proposed in week 2. The Wifi and Display classes will be added later on. The figure above clearly shows all the interactions between the different classes and class hierarchy. One thing to note here is that up to now only the odometer class is running as its own thread. All the other classes are called from the main method. However, this might change if we decide to use the light pollers to detect and correct the odometer while navigation. This design decision will be made after the line detection algorithm is finalized and that more intensive tests are ran on it to know its limitations.

4. Objective for week 4:

For week 4, our main focus will be to prepare a reliable software for the beta demo. To do so, we will be doing the following tasks:

- Finalizing the line detection method as well as a localization algorithm that would be used when navigating.
- Working with the hardware leader to develop a design that will detect the rings without any false readings.
- Building a faster localization algorithm that must finish under 30 seconds.
- Making changes to our initial gameplay algorithm to take into account the new changes in the setup of the competition that have been introduced in the version 2.0 of the project document.