**System Document**

**Project:** ECSE211 Design Project – Capture the Flag

**Task:** Construct an autonomous robot that can play one-on-one version of the game Capture the Flag

**Document Version Number:** Version 1.1

**Date:** November 28, 2017

**Authors:** Priscilla

**Edit History:**

- Created: 15/10/2017

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Edited By:** | **Section** | **Reason** |
| Oct. 15, 2017 | Ismail | Hardware, Compatibility, Reusability | Added these sections |
| Oct. 15, 2017 | Younes | System Model, Software | Added these sections |
| Oct. 15, 2017 | Mahad | Methodologies, Tools | Added these sections |
| Oct. 19, 2017 | Priscilla | Author | Changed to one author |
| Nov. 25, 2017 | Priscilla | Tools | Organized and updated tools used |
| Nov. 28, 2017 | Priscilla | Table of Contents | Added links |

1. **Table Of Contents**

[2.0 System Model………………………………………………………………………………… 2](#System_Model)

[3.0 Hardware Available and Capabilities……………………………….............................. 3](#Hardware)

[4.0 Software Available and Capabilities……………………………………………………….. 3](#Software)

[5.0 Compatibility………………………………………………………………………………….. 4](#Compatibility)

[6.0 Reusability…………………………………………………………………………………….. 4](#Reusability)

[7.0 Structures………………………………………………………………………………………](#Structures) 4

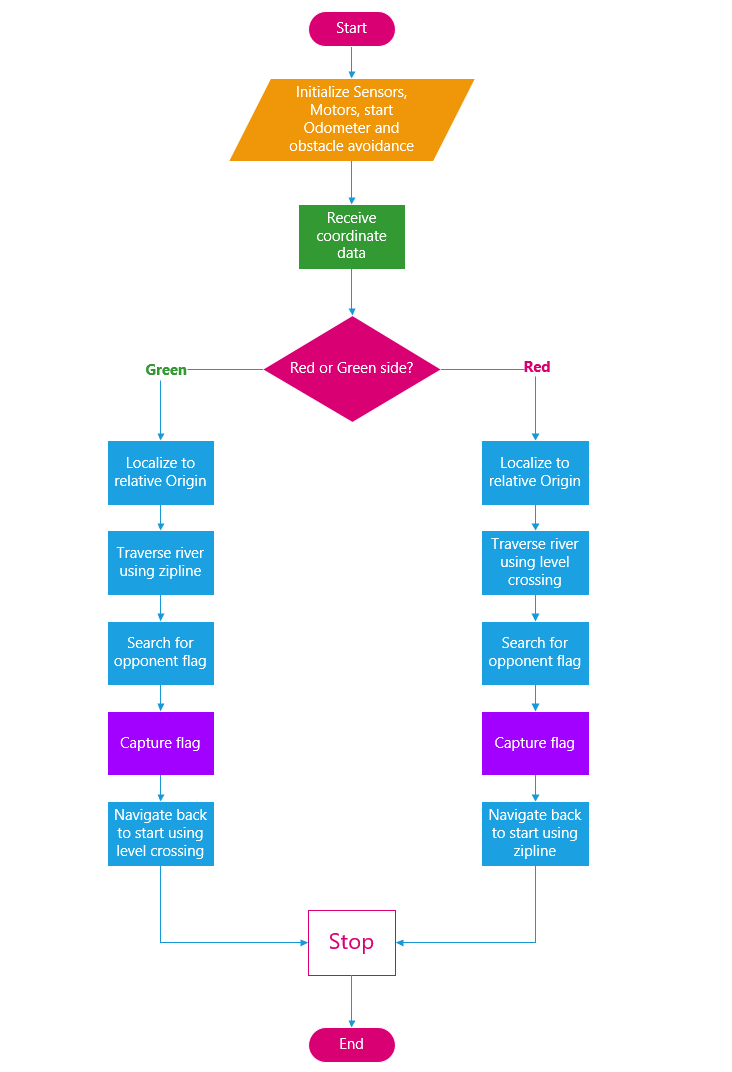
[8.0 Methodologies………………………………………………………………………………… 5](#Methodologies)

[9.0 Tools…………………………………………………………………………………………… 5](#Tools)

[10.0 Glossary of Terms…………………………………………………………………………..](#Glossary_of_Terms) 6

[11.0 References………………………………………………………………………………...... 6](#References)

1. **System Model**



1. **Hardware Available and Capabilities**

There are three times the hardware resources in the R&D labs that are available for this project. Therefore, there are three EV3 Bricks, three of each sensor (color, ultrasonic, touch) and three Lego kits. The provided Lego hardware components are limited in amount and in shape. They allow for a moderate scope for structure design, as the Lego parts are fixed and cannot be molded to give any shape wanted. Thus, the minimum amount of parts required starts to increase which creates difficulty when trying to minimize weight.

When running the motors and sensors, the EV3 can only exchange and process discrete values of data due to the limitations of its input and output devices. This means that data being exchanged can only be done in discrete values as the sensor and motor systems can only run on discrete values. This limits the scope of how the robot can function as opposed to using devices that can exchange and process continuous data types. Since real world signals are analog, the digital models may not be very accurate in their representations.

Another electrical limitation would be the power consumption as only six lithium ion batteries can be attached at a time. This limits how much power can be supplied to the EV3 to carry out the entire task. If the power falls below a certain threshold, the robot might malfunction or stop midway through its task. Also, the display is limited to a 178x128 pixel Monochrome LCD, which means there is a limit to how much data we can display at a time. There is also the limitation of color.

In terms of the processor, the EV3 has a single core of clock speed 300 MHz and RAM of 64 MB. This means that it can, at a maximum, execute roughly 300 million cycles per second. This greatly limits how fast the code is executed as opposed to processors using dual cores with clock speed of 1 GHz or above. The RAM being 64 MB also creates a problem. This means LeJOS must run and store data temporarily within a limit of 64 MB, creating a constraint on how much data of the sensor can be stored temporarily.

1. **Software Available and Capabilities**

The provided tools for the software systems are Eclipse, the LeJOS plugin and the EV3 Console (which is used for testing to read the information). These tools are semi-imposed as using them is a requirement of the project and hardware. Using other tools or technologies would require a fresh start since the R&D Labs were completed using the aforementioned tools. Furthermore, the EV3 platform itself has limitations. This platform is where the hardware and software meet; where the software instructs the hardware on what to do. The EV3 brick has only 0.06 GB RAM and a single core processor with just 0.3 GHz clock speed. These properties of the EV3 brick have a direct impact on the software performance. To provide a better picture of the limitations that arise from these properties, the brick can be compared to a $50 modern smartphone. A smartphone that costs around this much has approximately 1 GB RAM, and a 1.2 GHz quad-core processor, which allows multithreading in hardware [1].

The language used to program the EV3 brick is Java. Java is a powerful object-oriented language, however it has been around since the 1990’s and is showing its age. The Java 7 lacks many functional features found in other languages, making certain operations more complex. This is especially true for thread handling. This may cause slower execution than that if the code was written in a more modern language. Also, the Java Virtual Machine has an overhead ratio that can be as bad as 4:1 for complex objects (like the Odometer) according to IBM [2]. This means if the robot were efficiently programmed in Simple C or Assembly, it could be as much as four times faster. Also, we use the LeJOS libraries to program the EV3 brick. LeJOS is not always well documented. Some parts are vague or completely undocumented. This means some functions must be found using trial and error which is costly for the budget.

1. **Compatibility**

For the software aspect, all systems to be used will be specified by the client, so no third party systems will be used. An exception is any LeJOS Java Libraries deemed helpful. In the labs, each pair of team members has developed software for the essential mission tasks. These tasks include obstacle avoidance, odometry, color detection, line detection, navigation, and localization. These multiple implementations allow us to leverage the best ideas in our code base for the project. On the other hand, they must be made compatible (e.g. units of measurement, constants related to hardware). In addition, these implementations were under the constraint of one light sensor, one ultrasonic sensor and one EV3 brick and required a significantly smaller distance and time to be running. Therefore, the errors in the previous implementations must be accounted for or they will accumulate in this larger-scale project.

1. **Reusability**

As mentioned previously in Section “[5.0 Compatibility](#Compatibility)” of this document, this project will build on everything done during the R&D labs. Naturally, the project setup will consist of threads that are very similar to the labs. For those software constructs, reusability examples include digital signal filtering and processing, which can be used for both the ultrasonic sensor and the color sensor. Many Odometer and Navigation methods can be reused throughout the project. In addition, certain algorithms and logic structures we learned from other courses (like COMP251 and ECSE321) could be useful here (e.g. Dijkstra’s algorithm).

In terms of hardware, the mechanical structure built for the final R&D lab can be used as a base model for the project. The existing structure will have the mechanism required to traverse the zip line and perform all the other major tasks (navigate, avoid obstacles). Although it will not be perfect for the task, it provides a starting point from which optimizations and additions can be made.

1. **Structures**

Software structures for this project are built to run on the Linux-based LeJOS and will therefore follow an Object Oriented approach. Java objects can easily represent information and actions as well as the relations between different classes and the program control flow. An effort will be made to make the software as optimized as possible but due to strict limitations imposed by the embedded system hardware constraints (see Section “[3.0 Hardware Available and Capabilities](#Hardware)” above). This means that certain Software Engineering best practices, which assume modern hardware, must be modified to fit the EV3. For the same reasons, we will keep multithreading to a minimum.

In terms of hardware, the main factor considered while making the structure was balance and weight minimization. For good balance, the robot’s center of gravity is lowered and its structure stabilized whether stationary or in motion. Center of gravity is lowered by using a wider base and a low height. The structure is stabilized using additional Lego parts to make sure the structure is sturdy. If that is not the case, the robot would not operate as expected by the software.

1. **Methodologies**

For the software aspect, the software to be implemented is modelled using a simple flowchart. Then in the implementation, the method used is the same as in the R&D labs: test-driven development. This means that the mission is broken up into smaller subtasks. For each subtask, a test is devised in order to ensure proper performance. After writing the preliminary version of the method, the test is repeated under different conditions and the results are evaluate. After that, algorithms and hardware are adjusted as needed ad then tested again. The client has indicated that in the past, many firms were not successful in integrating their final design close to the deadline. This suggests that continuous integration should be used to maintain one codebase at all times. This is the approach maintained by Google [3]. Continuous integration takes more time to setup initially, but saves time on debugging by finding and fixing problems early on.

1. **Tools**

**Software**

* Java – high-level object-oriented programming language used to construct software
* Eclipse – integrated development environment for Java programming to develop the software
* LeJOS – Java Virtual Machine for Java to be run on Lego Mindstorms
* Github – used to store and update software effectively

**Hardware**

* Lego Mindstorms – kit that contains the necessary means to create customizable and programmable robots
* Lego Digital Designer – computer program in which virtual models of Lego pieces can be designed

**Hardware**

* GanttProject – scheduling of tasks and visual for timeline and allocation of resources

**Communication Channels**

* Dropbox – document storage space used to present documentation to the clients
* Google Hangouts – instant messaging application where groups can communicate
* Google Drive – collaborative document sharing platform

1. **Glossary of Terms**

* Digital signal filtering and processing: converting continuous signals to digital
* Multithreading: a single set of code that can be used by different processors at different times of execution
* Test Driven Development: A software development process that relies on continuously testing software components while they are being developed
* R&D lab: the research and development labs that were conducted prior to starting the project

1. **References**

|  |  |
| --- | --- |
| [1] | GSMArena, "LG Tribute - Full phone specifications", 2017. Available: https://www.gsmarena.com/lg\_tribute-6812.php. [Accessed: 17-Oct-2017]. |
| [2] | IBM, "From Java code to Java heap", Ibm.com, 2012. Available: https://www.ibm.com/developerworks/java/library/j-codetoheap/index.html?ca=drs-. [Accessed: 17-Oct-2017]. |
| [3] | C. Metz, "Google Is 2 Billion Lines of Code—And It’s All in One Place", WIRED, 2015. Available: https://www.wired.com/2015/09/google-2-billion-lines-codeand-one-place/. [Accessed: 17-Oct-2017]. |