SYSTEM DOCUMENT

Project: ECSE 211 Final Design Project – Team 6
Document Version Number: 4.0
Creation Date: 14/10/2017
Original Author: Alex Hale

Edit History:
[14/10/2017] Justin Tremblay: First draft.
[15/10/2017] Justin Tremblay: Filled in most of the parts relating to software.
[16/10/2017] Alex Hale: added TODO notes for sections we can't fill in yet; edited some writing; changed version system to WEEK.EDIT (e.g. it is currently week 1, edit 3 => 1.3)
[22/10/2017] Alex Hale: transferred to Word document for easier formatting; minor editing
[23/10/2017] Alex Hale: added electromechanical limitations
[30/10/2017] Alex Hale: updated state machine and class hierarchy diagrams
[4/11/2017] Alex Hale: updated all sections of the document; resolved TODOs; changed to present tense

2.0 SYSTEM MODEL

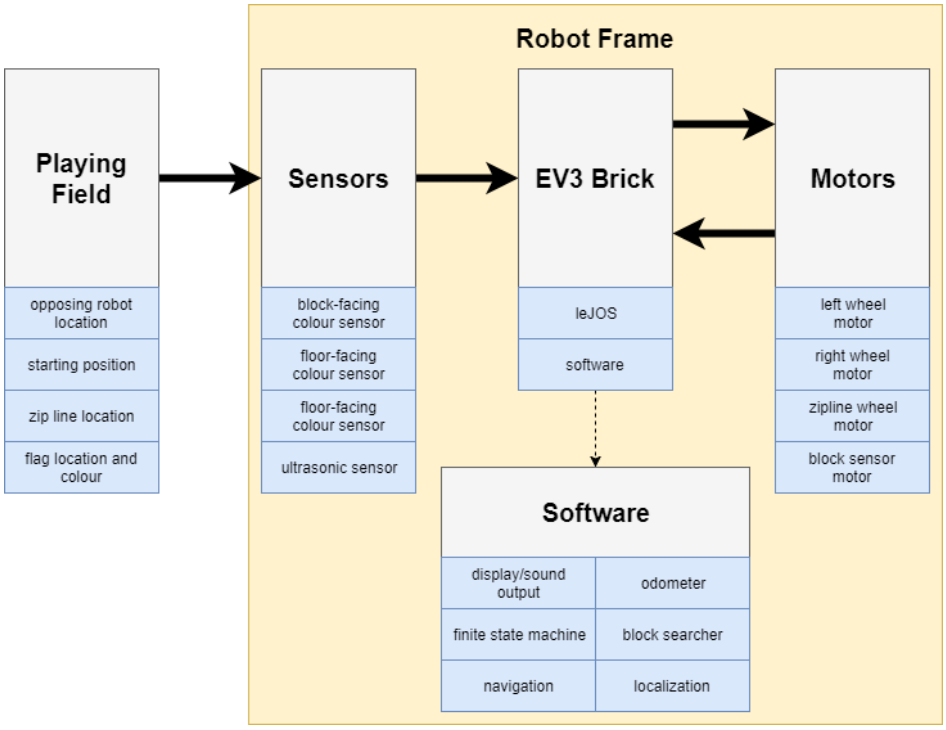

Figure 1 – Block Diagram of System Model

2.1 HARDWARE AVAILABILITY AND CAPABILITY

The hardware available for this project is three complete LEGO Mindstorms kits, each including one EV3 brick, five motors, one colour sensor, one ultrasonic sensor, two touch sensors, a gyroscopic sensor, and a wide variety of connecting pieces for construction. In addition, a custom 3D-printed wheel is provided to cross the zip line. Only one EV3 brick is used, while motors, sensors, and connecting pieces from various Mindstorms kits are used.

The capacity of the battery is limited, and as the battery depletes, the voltage supplied to the motors drops dramatically. The lower voltage affects the performance of the motors in an unpredictable fashion, so it is important to always test the robot with a charged battery.

The EV3 brick contains a low-power ARM processor, so computing speed is limited. System storage is provided by a micro SD card, which is fast enough that it does not bottleneck speed before the processor.

2.2 SOFTWARE AVAILABILITY AND CAPABILITY

This project is written in Java, making use of the leJOS library for the LEGO Mindstorms EV3 brick.

Java is a high-level object-oriented programming language with features like multi-threading and garbage collection. To its advantage, it is easy to use and multi-platform. Its disadvantages are that, since it runs on a virtual machine, it has a lot of overhead code, making the software difficult to optimize. This is an especially grievous disadvantage on the EV3 platform, which has very limited resources. Java's memory management model is also a big disadvantage: its use of garbage collection adds a lot of overhead, whereas a language like C uses pointers and manual memory management. Finally, when Java is compiled, it is converted to class files that are then compiled at run time, which often produces suboptimal code.

3.0 COMPATIBILITY

3.1 HARDWARE COMPATIBILITY

The physical robot design is limited to the items available in the LEGO EV3 Mindstorms kit. This kit has hundreds of pieces, but parts can only be connected in a limited number of ways, limiting the design possibilities. It is also possible to create custom parts using a 3D printer. However, this option would take a significant amount of time to create, and the provided pieces are deemed adequate by the hardware team.

3.2 SOFTWARE COMPATIBILITY

Each piece of software developed during the Research and Development phase was written in Java, meaning that the code developed by each laboratory group can be adapted to work together. The development tools in use, namely Eclipse and GitHub, are multi-platform, accommodating all the team members regardless of which operating system they use.

4.0 REUSABILITY

4.1 HARDWARE REUSABILITY

Not much of the hardware design is carried forward from the labs, because the robots in the labs made many assumptions. For example, the labs required the robot to have only one or two sensors and two or three motors equipped, whereas the project robot has four of each. The project robot has an arm to grab the zipline, and is larger and more stable to accommodate more sensors and motors than the laboratory robot. The only significant design element carried forward from the labs is the concept of a three-wheeled robot.

4.2 SOFTWARE REUSABILITY

Each piece of software developed during the Research and Development phase was written in Java, meaning that the code developed by each laboratory group can be adapted to work together. Software tools such as odometry, navigation, and localization are essential to this project. These tools are taken from the three laboratory code bases, the best elements of each are selected, and the combined product is included in the final project code base.

5.0 STRUCTURES

5.1 HARDWARE STRUCTURE

See Hardware Design Document, Section 6 – Chosen Mechanical Design

5.2 SOFTWARE STRUCTURE

See Software Design Document

6.0 METHODOLOGIES

6.1 HADWARE METHODOLOGY

The robot design is as simple and robust as possible. The robot's frame must support enough motors and sensors to complete the required tasks of the project (see Requirements Document for task details), but any extra functionality is not necessary and might jeopardize the performance of the necessary parts. Since the project requires the robot to traverse a zip line and to be on the playing field with another robot, the robot must be strong enough to take a small impact and continue pursuing its goal. Versions of the robot are tracked with version numbers listed in the Hardware Design Document, ensuring the documentation contains a record of the form of the robot at every point of the project.

6.2 SOFTWARE METHODOLOGY

Software is developed iteratively: each piece of code is written and tested, then the process is repeated until the necessary performance is achieved. Any issues are documented and assigned to a member of the software team. Whenever possible, each task is assigned to only one team member, ensuring that no resources are being wasted by multiple members working on the same task. Finally, the software development is documented with an API, a class hierarchy, and a state diagram, ensuring that everybody outside the software team can understand how the robot functions. Copies of these documents are created for each weekly meeting, ensuring that the documentation contains a record of the software's state on a week-to-week basis.

7.0 TOOLS

- GitHub
    - Source code control software. Allows the software team to collaborate on the code base, track issues, and view previous versions of the software.
- Eclipse
    - Integrated development environment. Provides plugins and interfaces for the software team to work on the leJOS project, create API documentation, and write code.
- Google Drive
    - Document storage in the cloud. Allows all team members to contribute to documentation, store documents that all members can see, and keep a weekly document record.
- Slack

    - Communication platform. Allows conversation about the project to be separated into multiple channels, ensuring that messages about specific aspects of the project do not get lost, as they would be in one long conversation.
- leJOS
    - Firmware for LEGO Mindstorms EV3 bricks. Used by the software team to code for the brick in Java, rather than the native graphical interface system provided by LEGO.
- LEGO Mindstorms
    - Kits of hardware and software made for the creation of customizable, programmable robots.

8.0 GLOSSARY OF TERMS

None required