CONSTRAINTS DOCUMENT

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ICARUS

Version: 2.0

November 27th, 2017

Developed by Team 08

**Project**: ECSE-211 Design Project: Game of Capture the Flag

**Task**: The goal of this project is to construct an autonomous machine that can play a one-on-one version of the game Capture the Flag, performing tasks such as navigation, localization etc.

**Document Version Number**: 2.0

**Date**: November 27th, 2017

**Author**: Ezz Aboulezz

**Edit History**:

|  |  |
| --- | --- |
| Date | Purpose |
| Oct 16 | Added Hardware constraints |
| Oct 17 | Added Software constraints |
| Oct 17 | Added Environmental issues |
| Oct 17 | Modified Hardware constraints |
| Oct 18 | Edited and completed document |
| Oct 25 | Fixed author to one person |
| Nov 27 | Final edit and re-structure |

1. [TABLE OF CONTENTS](#_top)

[2. ENVIRONMENTAL ISSUES........................................................................ 3](#ENVIRONMENTAL)

[3. HARDWARE CONSTRAINTS...................................................................... 3](#HARDWARE)

[4. SOFTWARE CONSTRAINTS....................................................................... 4](#SOFTWARE)

[5. AVAILABILITY OF RESOURCES............................................................... 5](#AVAILABILITY)

[6. BUDGET......................................................................................................... 5](#BUDGET)

[7. GLOSSARY………………………………………………………………… 6](#GLOSSARY)

1. **ENVIRONMENTAL ISSUES**

[\*See the Requirements document; section 2.5 Operating Environment, for a broad description of the competition environment and objects present in the environment.](Requirements%20Document.docx)

The environment in which the robot competes should be taken into account since it can cause several issues. Non-uniformity of lines on the competition surface can hinder localization and navigation. Random dust collection on competition surface affects tire traction and may cause slippage. These errors must be accounted for in the navigation and localization implementations. Large windows that let in ambient light are subject to change with the time of the day or the weather. The color sensor must be calibrated to these conditions. Furthermore, using the provided rechargeable batteries instead of the non-reusable batteries to power the EV3 brick would help reduce the amount of waste and the release of harmful chemicals to the environment.

1. **HARDWARE CONSTRAINTS**

[\*See System Document, section 2.0 Hardware Available and Capabilities for all the available material in the LEGO Mindstorms kit.](System%20Document.docx)

Three EV3 Mindstorms kits are available to assemble the autonomous machine. This limits the amount of components to use. Each kit includes one EV3 Brick, four sensors, and four large motors. The EV3 Brick features a 32-bit microprocessor, a large matrix display, four input ports, four output ports, a Bluetooth communication link and a USB communication link. Since four AA batteries or the EV3 Rechargeable DC Battery power the brick, the performance of the connected sensors and motors will depend on the available battery power.

Also, since each brick has four inputs ports and four output ports, this results in a maximum of four sensors and four motors to be used per brick. Furthermore, the light sensor and ultrasonic sensor are only rated in a range of distances and do not return very accurate values, thus various filters and conditions must be developed to deal with the errors. The EV3 ultrasonic sensor can detect distances with an error of 1 cm and has a scope range of 30° according to the manufacturer. The color sensor can distinguish between eight different colors and can serve as a light sensor to detect light intensity. It can measure reflected light with a sample rate of 1kHz.

Slippage of the wheels can accumulate errors in localization and navigation. To prevent the wheels from slipping, lower speeds will be used, which in turn will increase the time to complete a certain task. In addition, having a balanced robot design reduces the possibility of slipping. Hence, the robot should not be too big and accurate free body diagrams should be developed.

Since our robot design is a single brick solution, only four motors and four sensors can be used. Two motors are used for the wheels, one motor is used for zipline crossing, and one motor will be used for a rotating ultrasonic sensor. Also, since we only have access to four input ports, the optimal solution would be to use two light sensors and one ultrasonic sensor. One light sensor is used for the localization while the other is used to detect the colors of the blocks. The ultrasonic sensor is used for localization and object detection.

Finally, the Lego kits contain components with fixed lengths and angles. This may limit the implementation of the sketched designs of the robot. Since the odometer is based on the size of the wheels used for our previous labs, we cannot use any other types of wheels for the movement of the robot.

[\*See the Hardware sketch document for further details on the hardware design with its pros and cons](../Design%20Documents/Hardware%20Document.docx)

1. **SOFTWARE CONSTRAINTS**

***[See System Document, Section 3.0 Software Available and Capabilities and Section 8.0 Tools for all software and tools available.](System%20Document.docx)***

Since we are using the leJOS virtual machine, we are limited to using the Java language and cannot really change the tools. Nevertheless, Java is one of the most used languages with strong documentation and many libraries, which makes it very complete and easy to use. The leJOS software is built exactly for robots and thus facilitates access to all motors, sensors as well as their data. This is crucial when it comes to testing and analyzing what the robot does. During the development of our code, we must keep in mind that it has to run on the Mindstorms brick, which adds specific constraints. More specifically, the brick is small and not very powerful. The more complicated and long our code is, the less precise the robot is going to be. We thus must keep everything simple and avoid unnecessary threads. The software capabilities are also limited by the team's coding competence.

Also, since several bricks are available, we can run more than one brick in parallel. Although this is feasible, it would be very difficult to connect the bricks together and to synchronize them. Since we have a tight time budget, there won’t enough time to implement a dual brick solution. Knowing that each brick has four motor ports and four sensor ports, the code will be restricted to a limited amount of ports based on the number of bricks used. Also, we are provided with the code that connects the robot to the competition server and retrieves the parameters before starting. Our code must be compatible with this provided tool. This means that our architecture must include the provided code and our methods will take in the information as inputs.

1. **AVAILABILITY OF RESOURCES**

[***See Capabilities Document; section 5.0 Availability for all team members’ midterm times and the times that they are available.***](Capabilities%20Document.docx)

We determined that the best time for a weekly meeting to discuss and evaluate our progress, as well as to prepare for the upcoming meeting with the professor, is Tuesday from 12:30 to 13:30. During this time everyone is available so we expect full attendance. Furthermore, in terms of management, meeting two days before each week's deliverable allows us to make any necessary changes to our work before the meeting with the prof and allows us to reconsider any decisions we have made for next week's agenda.

The key academic dates listed in the chart suggest that the last week of October and the first week of November will be particularly difficult as more than half of the team will be unavailable for two to three days. This means that in order for the project's necessary tasks, included in the critical path, not to be delayed we need to allocate more resources in the weeks prior and after. To do this the team manager decided to slightly overload the entire team's workload before and after the two weeks in question, such that we will be able to achieve the average required number of hours over the time interval in question. This in theory should prevent our progress in the critical path from stagnating, while allowing the team members enough time to fulfill other commitments.

1. **BUDGET**

Our total budget is estimated in the amount of work hours we need to come up with and construct a design that solves the problem. After discussing the topic with the team members, the team manager decided that an average of 9 hours of work every week for every member was reasonable. This is an estimated number that we got from the professor's advice since none of the team members have any experience in similar projects. Given the goal of 9 hours of work per member per week for 6.5 weeks we have a total of 351 hours of work for all 6 members combined. While nine hours of work is an ideal average, we expect to deviate form it during weeks of high academic workload, as discussed in section 5; Availability of Resources.

In addition, there is an Engineering Bill of Materials (EBOM) that includes all the materials that were used to build the robot, but were not provided in the Lego Mindstorm kits. The poster also costs 24$ to print in color at McGill’s Copi-EUS.

1. **GLOSSARY OF TERMS**

Eclipse: an integrated development environment (IDE) used for [computer programming](https://en.wikipedia.org/wiki/Computer_programming).

EV3 brick: Lego Mindstorms EV3 is the third generation robotics kit in [Lego](https://en.wikipedia.org/wiki/Lego)'s Mindstorms line.

LeJOS: leJOS is a firmware replacement for Lego Mindstorms programmable bricks. It includes a Java virtual machine, which allows Lego Mindstorms robots to be programmed Java.

Mindstorms: The Lego Mindstorms series of kits contain software and hardware to create customizable, programmable robots. They include an intelligent brick computer that controls the system, a set of modular sensors and motors, and [Lego](https://en.wikipedia.org/wiki/Lego) parts from the [Technic](https://en.wikipedia.org/wiki/Lego_Technic) line to create the mechanical systems.

Gantt chart: a chart in which a series of horizontal lines shows the amount of work done or production completed in certain periods of time in relation to the amount planned for those periods.

Critical Path: The longest sequence of dependent tasks in the system.