Queen’s Robotic Warehouse

Presented to: Brian Frank and ELEC 299 TA’s

Proposal Report

ELEC 299 Mechatronics Project

Electrical and Computer Engineering

Faculty of Applied Science

Queen’s University

Prepared by Team 18, Section 203

Nick Totman, 20072196

Elizabeth McAvoy, 20046926

Eric Zhu , 20055416

Harleen Nijjar, 20055731

March 4th, 2019

# Statement of Originality

Our signatures below attest that this submission is our original work.

Following professional engineering practice, we bear the burden of proof for original work. We have read the Policy on Academic Integrity posted on the Faculty of Engineering and Applied Science web site (<http://engineering.queensu.ca/policy/Honesty.html>) and confirm that this work is in accordance with the Policy.

|  |  |
| --- | --- |
| **Individual 1:** |  |
| Signature: | Date: Mar 4, 2019 |
| Name: Eric Zhu | ID #: 20055416 |
| **Individual 2:** |  |
| Signature: | Date: Mar 4, 2019 |
| Name: Nick Totman | ID #: 20072196 |
| **Individual 3:** |  |
| Signature: | Date: Mar 4, 2019 |
| Name: Harleen Nijjar | ID #: 20055731 |
| **Individual 4:** |  |
| Signature: | Date: Mar 4, 2019 |
| Name: Elizabeth McAvoy | ID #: 20046926 |

Table of Contents

[Statement of Originality i](#_Toc2609096)

[List of Figures ii](#_Toc2609097)

[List of Tables ii](#_Toc2609098)

[1.0 Functional Requirements 1](#_Toc2609099)

[2.0 Hardware 2](#_Toc2609100)

[3.0 Software 3](#_Toc2609101)

[4.0 Collaboration 4](#_Toc2609102)

[5.0 Project Plan 5](#_Toc2609103)

[References 6](#_Toc2609104)

[Appendix A 7](#_Toc2609105)

# List of Figures

[Figure 1: Arduino Duemilanove Board Source: Adapted from [1] 2](#_Toc2609106)

[Figure 2 Robot Instruction Flowchart 3](#_Toc2609107)

# List of Tables

[Table 1: Pin Layout for Arduino Duemilanove 2](#_Toc2609109)

[Table 2: Project Timeline 5](#_Toc2609110)

# 1.0 Functional Requirements

For the upcoming Mechatronics 299 challenge, team 18 will build a robot which can autonomously retrieve objects in a set area and deposit the item to a set location. The robot will identify its starting location by using an IR sensor to read a value given by the IR beacon. Depending on the value, it will give one of the three preset paths to the robot to execute for the challenge. If this fails, it will use the forward IR sensor to detect the position of the other robots, and thus itself.

To displace the robot, it should be able to drive forward using the motors and linear encoders to follow a square grid made from black tape by using the bottom three photo sensors. While the robot is traveling along a path, it must be able to complete a 90 degree turn in either direction when necessary and self-correct its alignment if it were to start curving away from the black line.

For retrieving the object, the robot will use the front IR sensor to search for the object, and once it has been found, it will slow down its velocity while approaching it. When both bumpers contact the wall, it will notify the robot to stop and grab the object with the grippers. Using the pressure sensor on the gripper's pads, the grippers will close and until the pressure sensor reaches a set value which indicates that the ball has been gripped.

Lastly, the robot will trace back its path and return the object into the goal bin. Once the robot reaches the goal bin, it will tilt the grippers down and release the ball. The robot will return to the grid, restart and execute its search again.

# 2.0 Hardware



Figure : Arduino Duemilanove Board  
Source: Adapted from [1]

Table : Pin Layout for Arduino Duemilanove

|  |  |  |
| --- | --- | --- |
| **Sensor** | **Digital Pins** | **Analog Pins** |
| Bumpers | 2, 3 |  |
| Motors | 4, 5, 6, 7 |  |
| Grip, Tilt, Pan Actuators | 9, 10, 11 |  |
| Linear Encoders | 12, 13 |  |
| IR Reciever |  | 0 |
| Grip Sensor |  | 1 |
| Front IR Sensor |  | 2 |
| Front Photosensors |  | 3, 4, 5 |
| Bluetooth Module | Connected across the extra Pins (RESET through Vin) | |
| \*Digital Pin 8 is unused | | |

# 3.0 Software

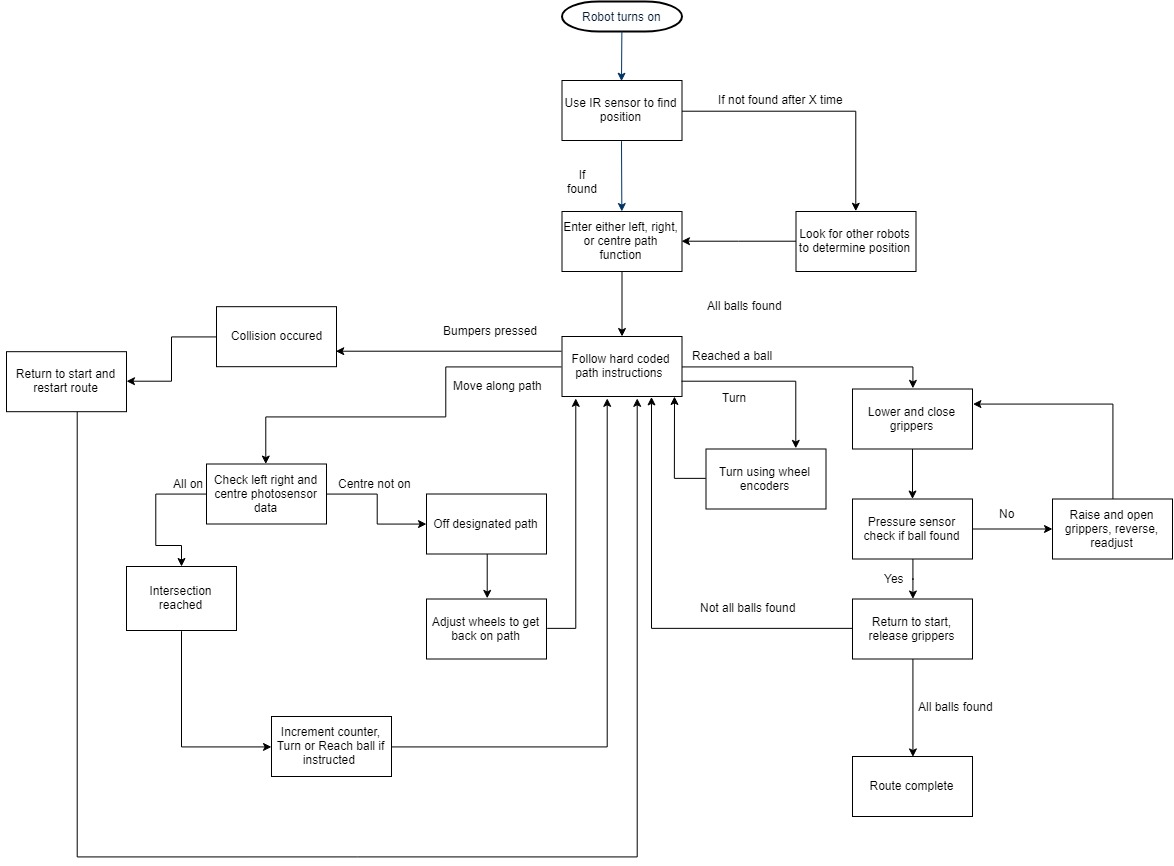


Figure Robot Instruction Flowchart

\*Pseudocode located in the Appendix

# 4.0 Collaboration

A high performing team requires a good communication between all team members. For teams 16, 17 and 18, all three teams will communicate mainly through Facebook Messenger to discuss any issues and updates for the duration of the project. Nick Totman will be the liaison for group 18 who will directly be in contact with other liaisons to discuss bigger milestones and completed progress. To improve team understanding, team meetings will be set in place between the completion date to discuss any questions and problems to be solved.

To ensure that all team members understand and can contribute for the coding process, the code will be placed in a GitHub repository. Commits will need to be accepted from all 3 teams before being pushed and merged onto the master code. This can allow for edits and alterations from all teams.

To minimize collision between robots, each group will be given a path to complete, such as the left side, the right side, or the middle. Depending on the specific position the robot, the robot will have to execute a case and follow a specific path. The code for all robots will be the same. This method results in all robots being interchangeable for specific task to accomplish and they can start in any position.

# 5.0 Project Plan

Table : Project Timeline

|  |  |  |
| --- | --- | --- |
| **Completion Date** | **Key Task** | **Assigned Members** |
| 04/03/2019 | Develop a strategy for the competition on how the robots should perform to complete the task. | Everyone |
| 10/03/2019 | Code IR sensors to determine the position of the robot based on the ASCII Character received, or on the positions of the other robots. | Eric |
| 10/03/2019 | Code the robot to move forward to the three top most balls after position is determined, while maintaining alignment. | Harleen |
| 10/03/2019 | Code the gripper to be able to collect a ball, sense when the ball is gripped, be able to carry it and release it. | Eric, Harleen |
| 10/03/2019 | Develop an optimized solution for the robots to be able to collect the remaining balls while minimizing time and collisions. | Everyone |
| 10/03/2019 | Code the robot according to the case assigned to each team to complete the remaining tasks. | Nick, Beth |
| 11/03/2019 | Have a collective group meeting with the other 2 teams to discuss any issues, seek help from the other teams and also to ensure that everyone is on the same page. | Everyone |
| 18/03/2019 | Complete coding the robot so that it is fully operation and be able to complete the designated case. Assign any ongoing issues to be resolved immediately. | Everyone |
| 23/03/2019 | Final test of robots. | Everyone |

# References

|  |  |
| --- | --- |
| [1] | "Arduino Duemilanove," Arduino, [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoardDuemilanove. [Accessed 4 March 2019]. |

# Appendix A

**Pseudocode:**

**Define constants**

ASCII Characters

Pin Definitions

**Setup:**

Read IR into int

If this doesnt work after ‘x’ time,

turn and look for other robots, determine position based upon other positions

End setup

**Loop:**

Call Route

Break

End loop

**Functions Definitions:**

**(Route)** (ASCII Character) //Routes are Hardcoded and predetermined to be fast with no collisions

if L

Follow predetermine L path

call forward, turn, correct, pick up, etc as necessary

else if C or R

Follow Corresponding Paths, similar to L

end if

**(Forward)** (number of intersections)

turn motors on

if all 3 IR sensors on //count intersections passes

count 1 intersection

end if

if count == number of intersections

stop

end if

**(Check Position)**

Turn left 90 degrees

Measure distance using front IR sensor

if object close

there is a robot on the left

end if

repeat for right side

return position ASCII Character

**(turn)** (direction, degrees)

convert degrees to number of Linear Encoder changes

adjust motor directions and speed to follow direction

count changes in Linear Encoders while count < degrees

if count > degrees

Stop

end if

**(correct)** //corrects if not straight

if left IR sensor on

turn right (small amount)

else if right IR sensor on

turn left small amount

end if

**(Pick up)**

move forward until IR Sensor good distance away

if bumpers activated

back up

end if

move actuators to ball

close grip

if grip sensor active

stop closing

end if