University of Missouri – Kansas City

Designed in-part by:

Salim Al-harthi

SID: 16182850

Course Number:

ECE 403

Team Members:

[Benjie Merid](https://umkc.instructure.com/courses/28949/users/21958)

[Caitlyn Hetzel](https://umkc.instructure.com/courses/28949/users/17561)

[Jacob Lopez](https://umkc.instructure.com/courses/28949/users/17849)

School of Computing and Engineering

University of Missouri – Kansas City

5/5/2019

**Introduction:**

In this senior design project, we are tasked with designing and building a fully autonomous vehicle to navigate the NRC’s obstacle course that shown in Figure 1. Specifically, the Autonomous Vehicle Challenge (AVC)

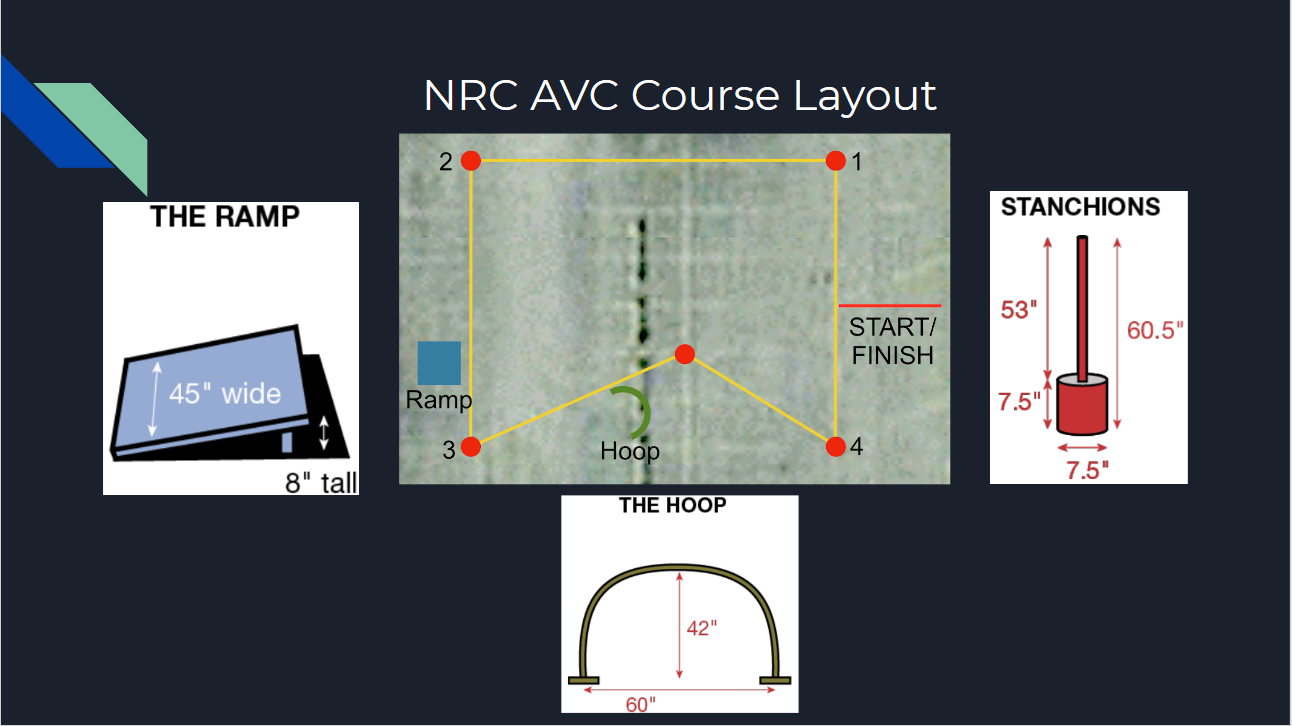


Figure 1 NRC AVC Course Layout

It was 2 semester project. We learned several new things and improved in what we know.

**Objective:**

The objective of this senior design is to build an autonomies vehicle to navigate an obstacle course. A successful run is one where the vehicle navigates around the 4 waypoints (red stanchions) and crosses the finish line in under 5 minutes. Additionally, bonus points are given for completing special tasks during a run which are clearing a hoop and a ramp.

**Equipment and cost:**

NCR Chassis ($200)

Adafruit 9-DOF Orientation IMU ($30)

Soldering boards ($5)

Switches ($4)

BTS7960 Motor driver ($16)

Power Charger ($24)

Spray Paint ($10)

Plywood ($20)

Arduino Mega (pre owned)

2x Arduino Nano (pre owned)

Infrared sensor (pre owned)

GPS (pre owned)

Raspberry pi 3 (pre owned)

Raspberry battery (pre owned)

2x 9v lithium batteries (pre owned)

5v power supply (pre owned)

Total cost $ 309

**Procedure:**

We started by analyzing the course and planning our design.

initial design:

our initial design which was a vehicle that navigate using a GPS and a magnetometer as a compass then detecting the hoop and the ramp using OpenCV.

Problems:

The problem with this design was that the magnetometer will provide false data if it was not parallel to the ground.

Another problem with this design is the GPS accuracy.

Second design:

The second design was a solution to the first design problems. We solved the magnetometer problem by using IMU which and adjust itself. In addition, we solved the GPS problem by detecting the stanchion in OpenCV to follow them while keeping track of our location on the GPS. But we had communication problems between the Arduino and the raspberry pi3.

Third and final design:

The third design solved the communication problems between the Arduino and the raspberry pi3. We used I2C and serial communication to transmit data.

**How does the project work?**

We first input GPS location that we desire to clear then the Arduino mega will calculate the distance and heading between its current position and the first GPS point. The vehicle will start moving to the fist pint will one of the Arduino Nano calculate the distance that being traveled. If the vehicle saw the stanchion it will be following the stanchion and keep calculating the distance. Once it traveled the desired distance it will check if it is within the threshold of the location if so, it will turn accordingly and update the GPS pint and the input location. If the vehicle so the ramp it will go to it and clear it.

**The following flow charts describe how our vehicle operates:**

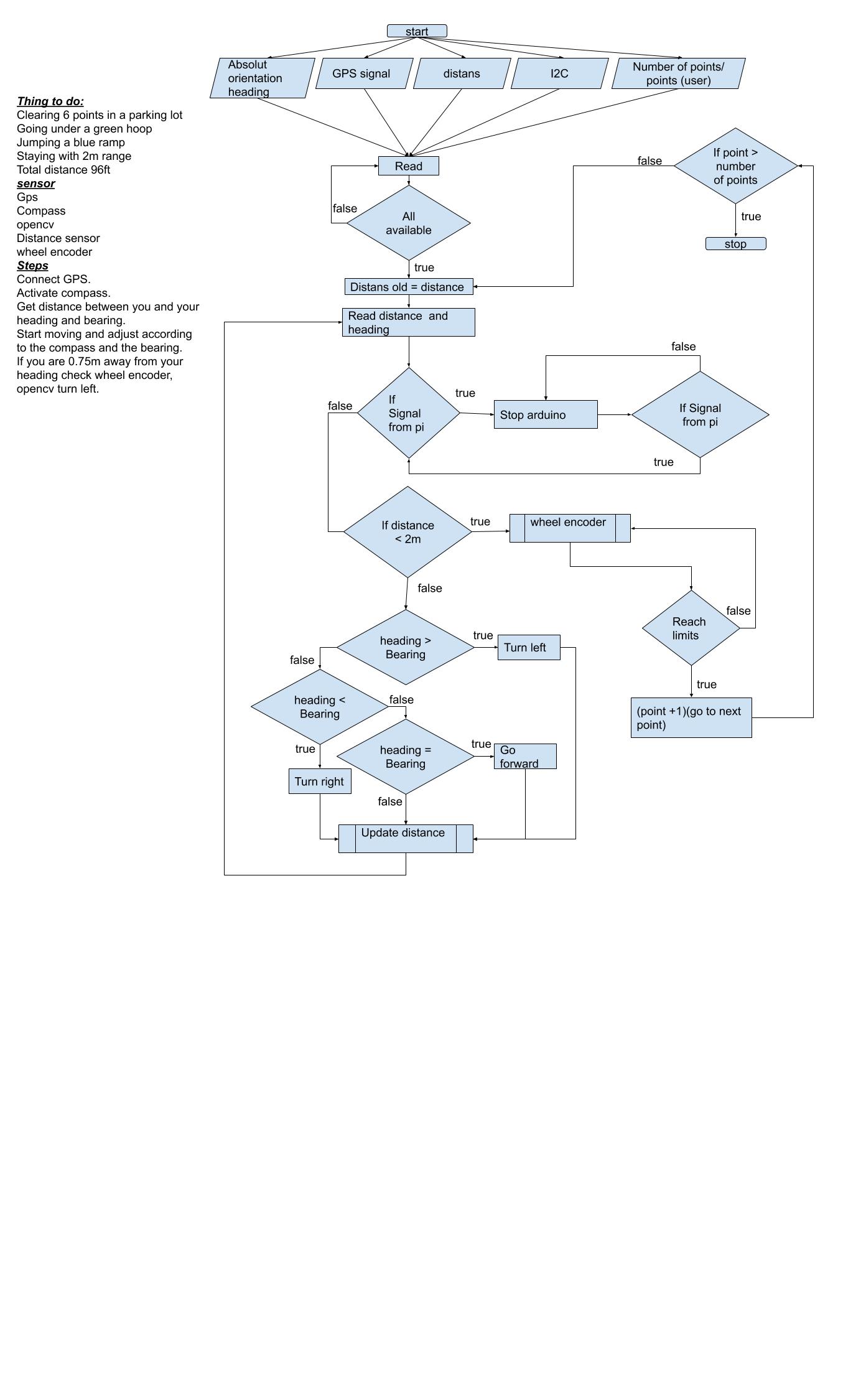
****

Figure 2 high level flow chart for the Arduino Maga code

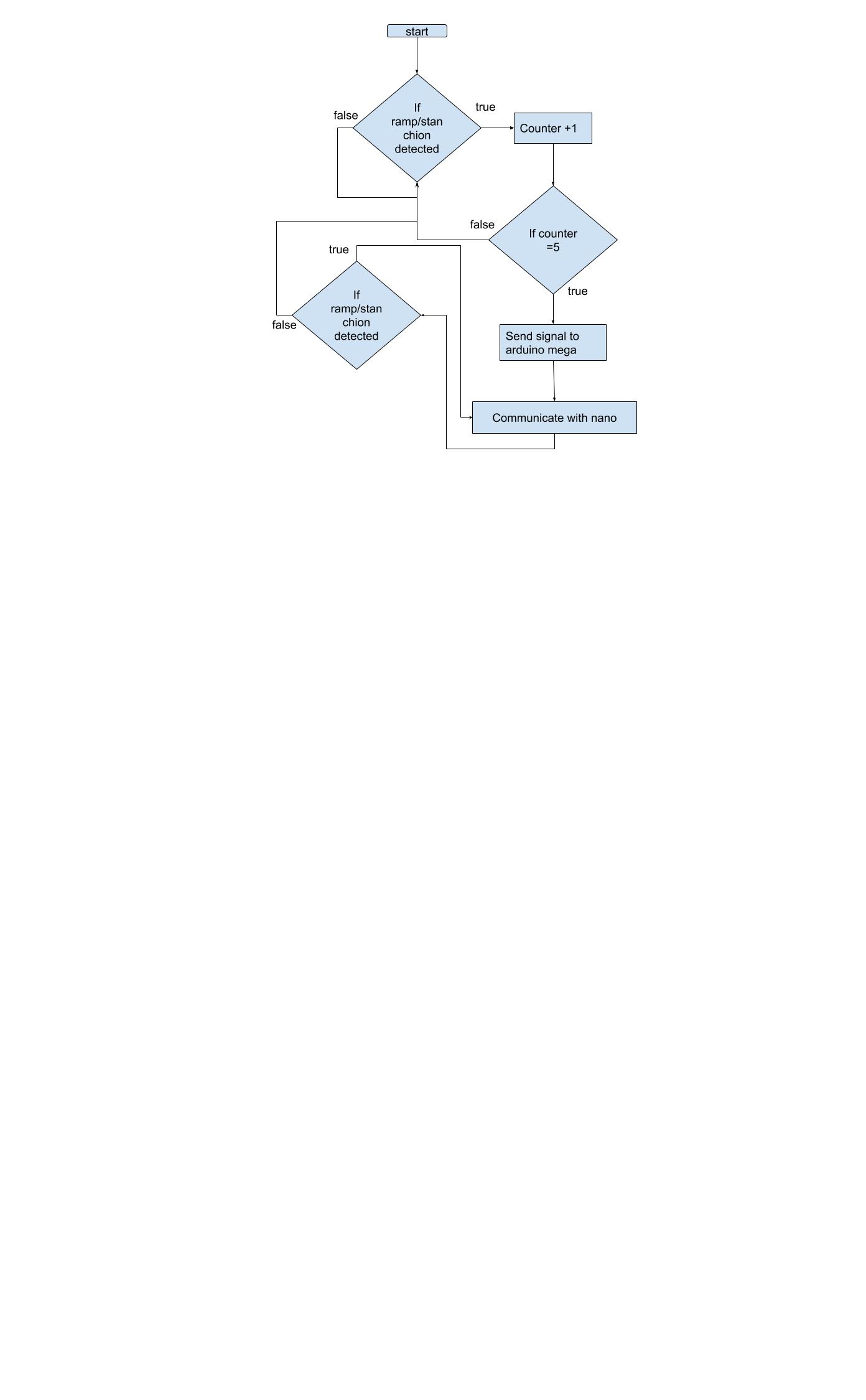
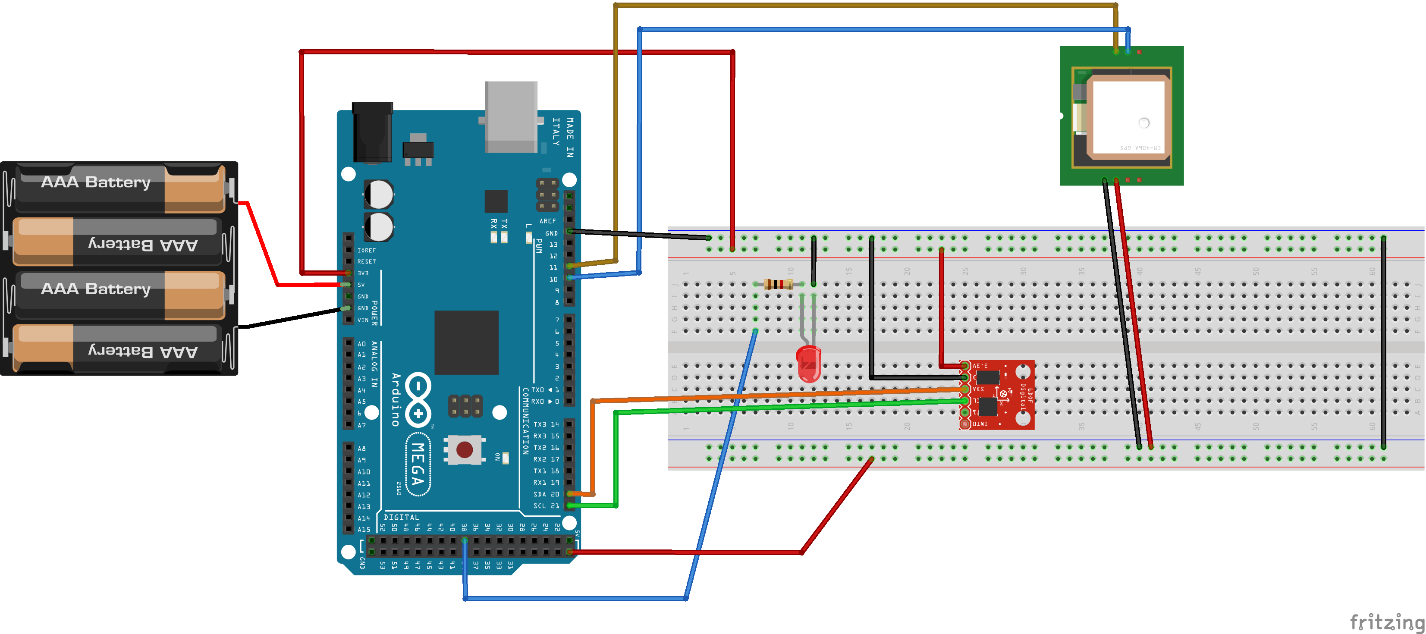


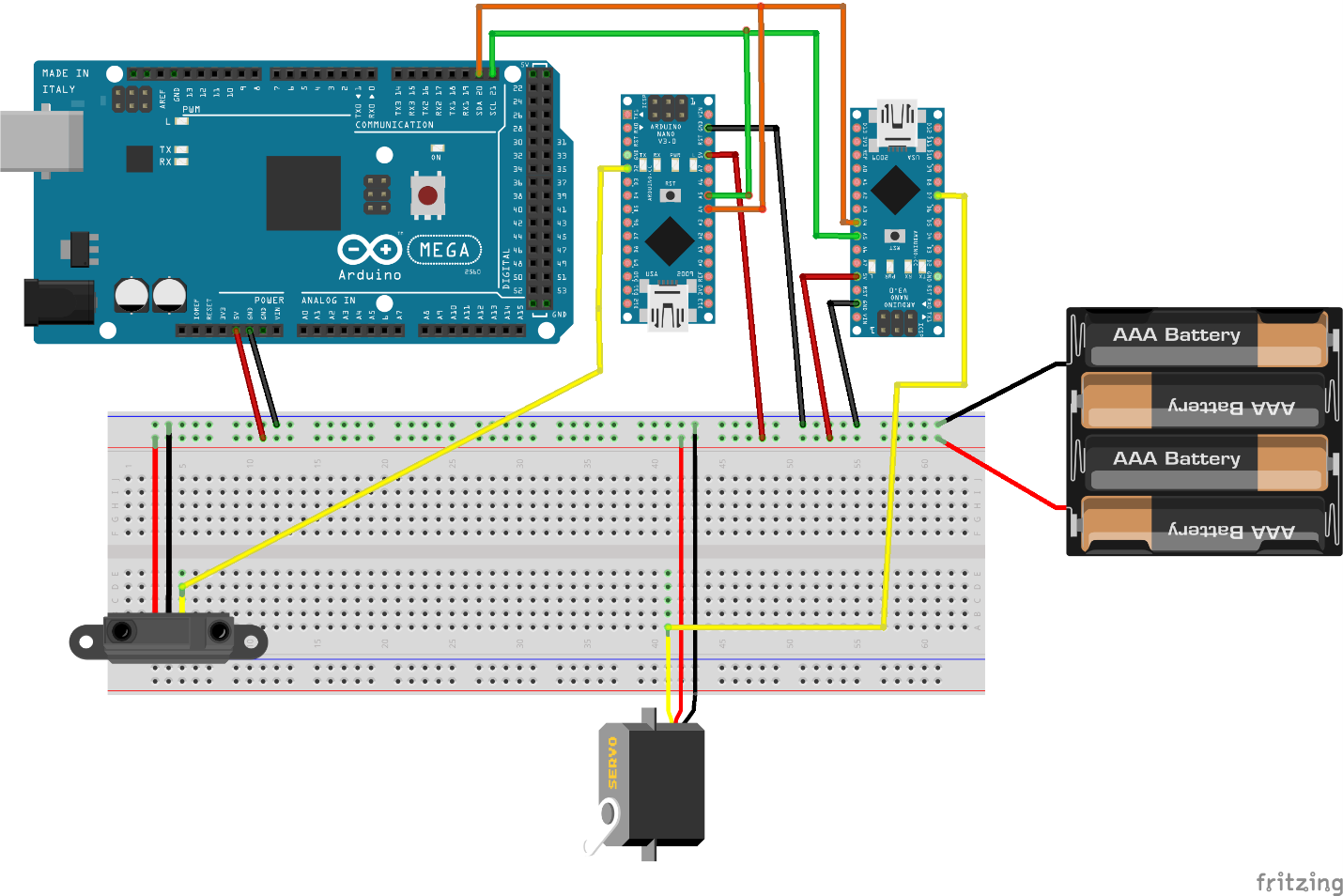
Figure 3 high level flow chart for the OpenCV (object detection) code

**The following figures show the wire connection:**



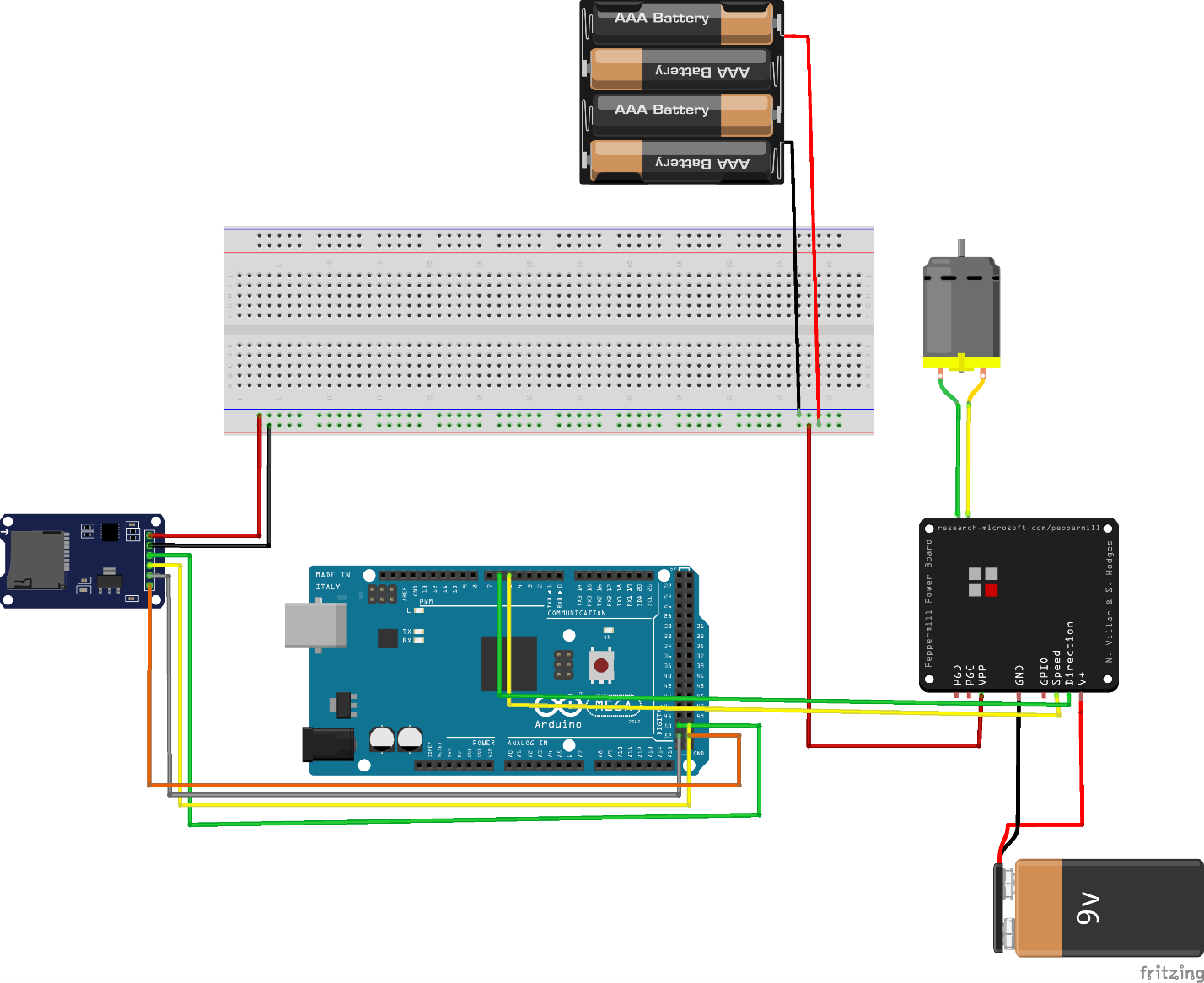
|  |  |
| --- | --- |
| Arduino | **IMU** |
| 21(SCL) | SCL |
| SDA (20) | SDA |
| 3.3V | 3.3V |
| GND | GND |

|  |  |
| --- | --- |
| Arduino | **GPS** |
| 11 | RX |
| 10 | TX |
| 5V | VCC |
| GND | GND |



|  |  |
| --- | --- |
| Arduino Nano | **servo** |
| 7 | Servo input |
| 5V | 5V |
| GND | GND |

|  |  |
| --- | --- |
| Arduino | Arduino Nano |
| SDA | A4 |
| SCL | A5 |
| 5V | 5V |
| GND | GND |

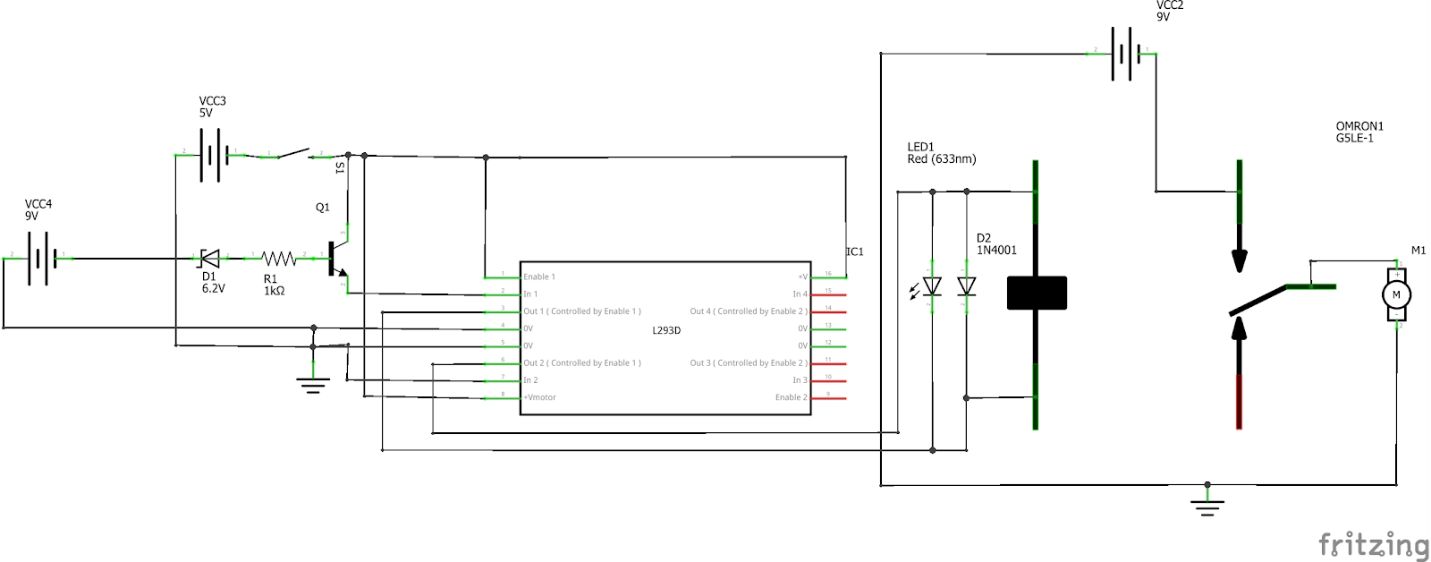


|  |  |
| --- | --- |
| Arduino | **Motor driver** |
| Motor black | w- |
| Motor red | W+ |
| Battery - | B- |
| Battery+ | B+ |
| 5v | VCC |
| GND | GND |
| 5 | RPWM |
| 6 | LPWM |
| 23 | R\_EN |
| 24 | L\_EN |

|  |  |
| --- | --- |
| Arduino | **SD** |
| 50 | MISO |
| 51 | MOSI |
| 52 | SCK |
| 53 | CS |
| 5V | VCC |

**Build from scratch component:**

This circuit is a switch to turn the motor on and off as well as isolating the main battery from the motor battery. In addition, it will turn the motor off if the battery level went below 10% to protect against malfunction due to undervoltage.



**Results:**

we successful completed a full loop with clearing the ramp but not the hoop

**Discussion of Results:**

we could not clear the hoop since we could not figure out the color range for the hoop. The hoop is to dark and in our camera, it looks like a black collard instead of dark green so the program will detect every other black circle.

****

Figure 4 picture of the project

**Reference:**

Contest Manual, 2019 National Robotic Challenge.