

ELE494-08

Autonomous Robotic Systems

Project CTE Document 2

Yousif Khaireddin

63618

Dr. Shayok Mukhopadhyay

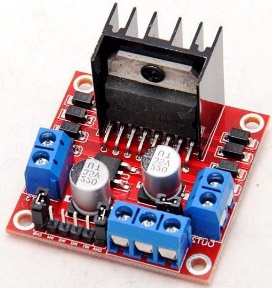
**Updated Goal Statement**

Throughout our project, we aim to build a robot that will survey an area to identify the point of maximum light intensity while simultaneously wirelessly communicating without our phones to send back real-time information regarding its position and light intensity at that area. Should time allow, we hope to also incorporate obstacle avoidance without prematurely hard-coding the location of these obstacles.

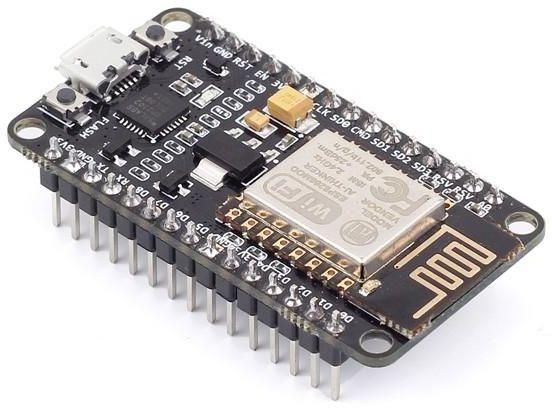
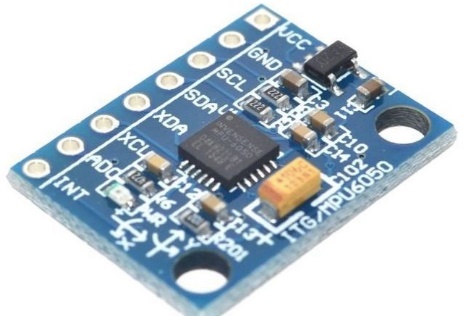
**Objective**

This robot will be placed in a pre-defined area which it should survey to obtain the location of maximum light intensity. We hope to accomplish the above through continuously measuring the position of the robot using two encoders and an accelerometer and feeding this information into a Kalman filter to obtain accurate positioning and localization. During the execution of the continuous loop described above, the robot should also measure the light intensity as it moves. This whole process will occur while the robot wirelessly feeds this information back to us enabling us to plot its movement and progress in real-time along with the light intensity values at these different positions. Once done with surveying the area, the robot should go to the point of maximum light intensity and stay there.

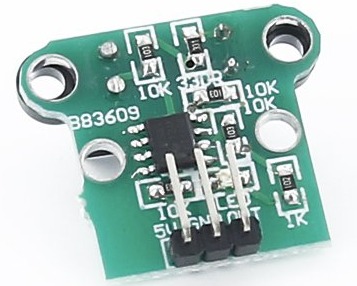
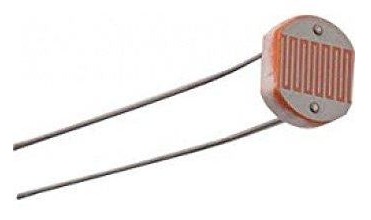
**Hardware Specifications**

* *

*Robot Chassis, Wheels, & Motors L298N Dual H-Bridge motor controllers*

* *

*ESP8266 based NODEMCU microcontroller MPU6050 3 axis accelerometer/gyroscope*

* *

*HC-020K Speed Measuring Module Photoresistor LDR CDS 5mm*

**

*Huawei 6700 mAH power bank*

**Plan of Action**

**Preliminary Tests**

Once we have everything required, we hope to start by building the code to run each of the components separately. This will be done through powering each thing alone and watching for its expected operation.

**Robot Assembly**

Once we have ensured the different systems work separately, we can then mount everything onto the chassis and test that the robot can actually move. We will supply some PWM signal and watch for the robot’s motion.

**Sensor Data Acquisition Functionality**

Next, we will test he ability of the encoders to capture the robots wheel speed by running the robot for a set number of wheel rotations and ensuring that the encoder counter is relatively accurate.

Next, we will test the accelerometer to ensure that it is also producing a reasonably understandable and logical output with changes in the robot’s acceleration. To do this, we could increase the PWM signal over time and see that that the acceleration output is also increasing.

**Accuracy in Positioning**

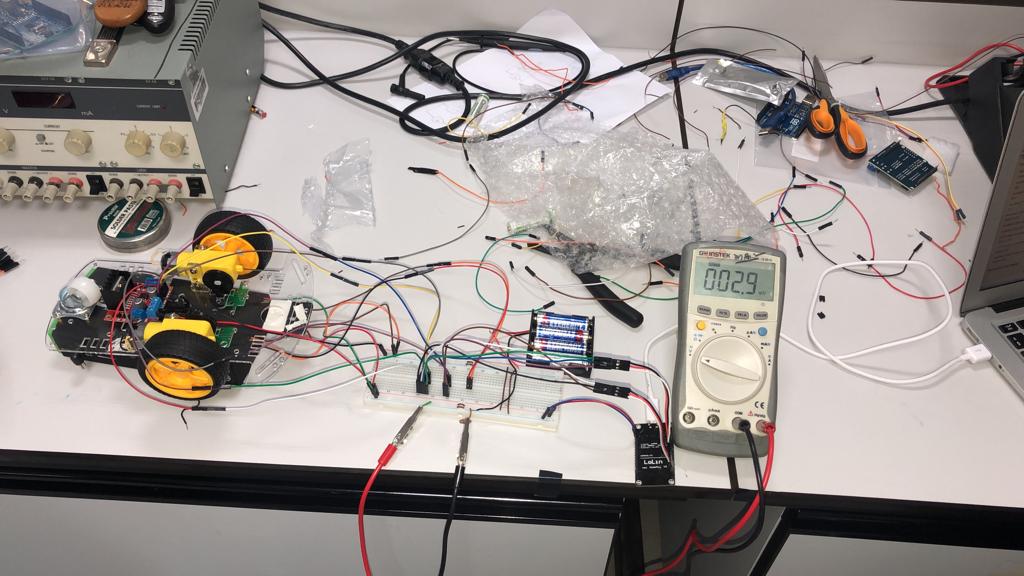
Once done, we will then move on to implementing the Kalman filter code and ensuring its operation. This will be tested by moving the robot 1m and seeing that its measured position with respect to its initial position is 1m (with reasonable allowed error).

**Real-Time Communication**

Then, we will move on to ensure the ability for the robot to communicate with us in real-time, as required by testing that the measured values are indeed passed over the network created and plotted as expected.

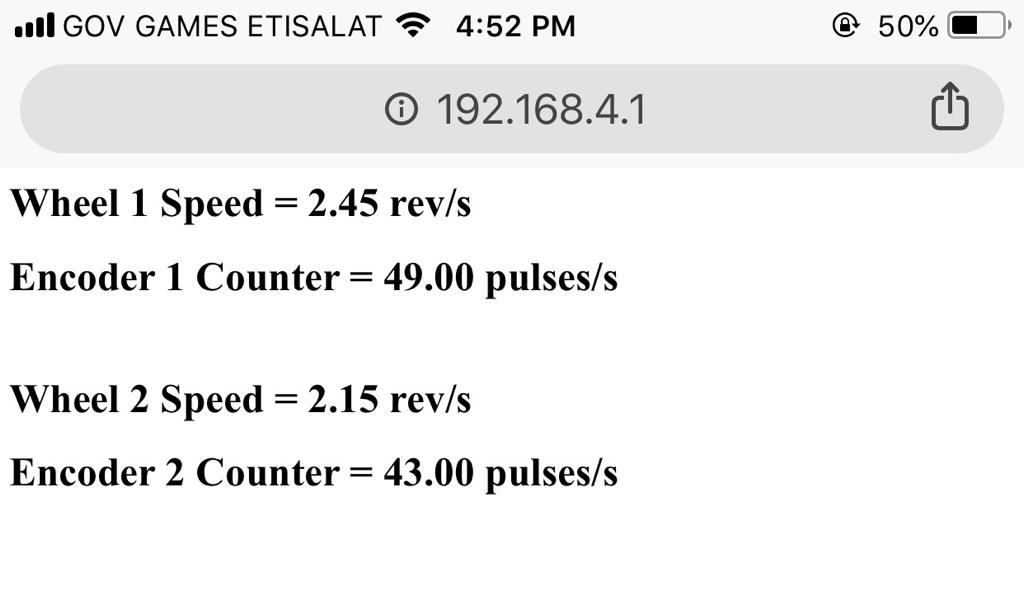
**Primitive Testing and Results**

Some initial testing of the system can be seen below. Here, we were in the process of testing that the different components do indeed work well together, before we mount everything onto the robot.



*Hardware Connections*

Through the testing done above, we were able to ensure that the encoders are indeed operating as expected. Their outputs were then fed into the access point to allow us to access them through our phones and see the following.



*Display of Results*

From the above, as seen as though our scenario isn’t overly complex, we expect that the Kalman filter will indeed rectify the errors in positioning that will occur if we just integrated these speeds to obtain a measure of position. It is important to note that although the same PWM signal was applied to both H-Bridge inputs, the second wheel is (according to our encoder alone) spinning 0.3 rev/sec slower than the other one. This means that some calibration and balancing will be required to ensure that both wheels are equally responsive and similar in performance.

Next, we also expect that the wireless transfer of results to be possible since we were able to obtain the above results. Currently, these values are updated whenever this page is refreshed, so actually taking this data to create real-time plots might present some issues.

**Team Member Roles**

Regarding our project, both members will equally contribute to almost all parts to fulfill the required objects.

I will be responsible for developing the circuitry and Hardware connections.   
The circuit in figure 1, for instance, was built by me.

Nasir, on the other hand will be responsible for preparing the code for displaying the results in real-time. This can be seen in figure 2.

Next, we will both be responsible for implementing the Kalman filter into our system and developing the algorithm that the robot should follow to ultimately reach the area of maximum light intensity.

Some of the code we have already written regarding these tasks, along with which member wrote that specific set of code, can be seen in the appendix.

**Appendix**

Some of the code we have already developed can be seen below:

Here are the initializations (written by both)

1. #include <ESP8266WiFi.h>
2. #include <WiFiClient.h>
3. #include <ESP8266WebServer.h>
5. //Pin Definitions
7. #define ENCODER1 5 //[D1]
8. #define ENCODER2 12 //[D6]
10. #define EN1  4  //[D2] 44 ON BREADBOARD
11. #define IN1  3  //[rx]
12. #define IN2  1 //[tx]
14. #define EN2  14 //[D5] 35 on BREADBOARD
15. #define IN3  16 //[D0] 43 ON BREADBOARD
16. #define IN4  13 //[D7] 42 ON BREADBOARD

19. **const** **char**\* ssid = "Robot";
21. **float** count1;
22. **float** count2;
23. **float** rev1;
24. **float** rev2;
25. **float** rev1\_f;
26. **float** rev2\_f;
27. String message;
29. ESP8266WebServer server(80);

Server Operation (written by Nasir)

1. **void** handleRoot(){
2. message = "<h1>Wheel 1 Speed = ";
3. message += String(rev1\_f);
4. message += " rev/s";
5. message +=  "</h1>";
7. message += "<h1>Encoder 1 Counter = ";
8. message += String(count1);
9. message += " pulses/s";
10. message +=  "</h1>";
11. message +=  "<br>";
13. message += "<h1>Wheel 2 Speed = ";
14. message += String(rev2\_f);
15. message += " rev/s";
16. message +=  "</h1>";
18. message += "<h1>Encoder 2 Counter = ";
19. message += String(count2);
20. message += " pulses/s";
21. message +=  "</h1>";
22. message +=  "<br>";
24. server.send(200, "text/html",  message);
25. }

Microcontroller Setup (written by both)

1. **void** setup() {
2. delay(1000);
4. //Defining PIN directions
6. pinMode(EN1, OUTPUT);
7. pinMode(IN1, OUTPUT);
8. pinMode(IN2, OUTPUT);
9. pinMode(EN2, OUTPUT);
10. pinMode(IN3, OUTPUT);
11. pinMode(IN4, OUTPUT);
13. delay(1000);
14. WiFi.softAP(ssid);
16. IPAddress myIP = WiFi.softAPIP();
18. analogWrite(EN1, 512);
19. analogWrite(EN2, 512);
20. digitalWrite(IN1, HIGH);
21. digitalWrite(IN2, LOW);
22. digitalWrite(IN3, LOW);
23. digitalWrite(IN4, HIGH);
25. server.on("/", handleRoot);
26. server.begin();
28. pinMode(ENCODER1, INPUT);
29. pinMode(ENCODER2, INPUT);
30. attachInterrupt(digitalPinToInterrupt(ENCODER1), High\_Callback, RISING);
31. attachInterrupt(digitalPinToInterrupt(ENCODER2), Low\_Callback, RISING);
32. }

Encoder Counter (written by me)

1. **void** loop(){
2. rev1 = 0;
3. rev2 = 0;
4. **for**(**int** j=1; j<11;j++){
5. count1 = 0;
6. count2 = 0;
7. delay(100);
9. rev1 += count1 / 20; //number of revolutions
10. rev2 += count2 / 20; //number of revolutions
12. }
13. rev1\_f = rev1;
14. rev2\_f = rev2;
15. server.handleClient();
16. delay(100);
17. }
19. **void** High\_Callback(){
20. count1 += 1;
21. }
22. **void** Low\_Callback(){
23. count2 += 1;
24. }