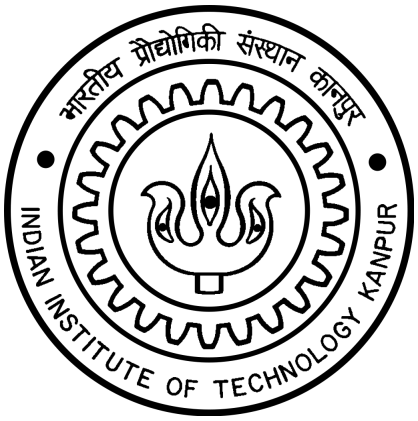
SUMMER'16 PROJECT
ROBOTICS CLUB
Science And Technology Council



WASP
Project Report

15-05-2016 To 26-06-2016

by

ABHIMANYU (150013)
HIMANSHU (150288)
SUNIT KR SINGH (150737)
TARUN AGARWAL (150762)
SUJAT ALI KHAN (150735)
SIRSENDU SARKAR (150720)

Contents

1 INTRODUCTION TO THE PROJECT **4**
- 1.1 Overall aim of the project . . . 4
- 1.2 Motivation . . . 4

2 HARDWARE OVERVIEW **5**
- 2.1 ESP8266 WiFi Module . . . 5
- 2.2 Access points:Routers . . . 5
- 2.3 Arduino ATMega2560 . . . 6
- 2.4 FTDI Programmer Basic . . . 6
- 2.5 Motor Driver . . . 6
- 2.6 Buck Convertor . . . 6
- 2.7 IMU . . . 6

3 SOFTWARE OVERVIEW **8**
- 3.1 Arduino IDE . . . 8
- 3.2 AT Firmware . . . 8
- 3.3 Magneto . . . 8
- 3.4 GNU Octave . . . 9

4 DESIGN OVERVIEW **10**
- 4.1 Design of Bot . . . 10
- 4.2 Circuit's Design . . . 11
- 4.3 Our Arena . . . 11

5 WiFi POSITIONING SYSTEM **13**
- 5.1 What is WiFi Positioning System? . . . 13
- 5.2 Free Space Path loss . . . 13
- 5.3 Data Collection . . . 14
  - 5.3.1 Aim . . . 14
  - 5.3.2 Reading1 . . . 14
  - 5.3.3 Reading2 . . . 16
  - 5.3.4 Reading3 . . . 17
- 5.4 Trilateration . . . 19
  - 5.4.1 Why Trilateration ? . . . 19
  - 5.4.2 Application in our project . . . 19

6 BOT CONTROL SYSTEM **20**
6.1 IMU Calibration 20
6.2 Kalman Filter 20

7 Conclusion **22**

1 INTRODUCTION TO THE PROJECT

1.1 Overall aim of the project

Aim of this project was to make a bot that would identify it's location in space with respect to the accessing points using ESP-8266 WI-FI module and the bot would then traverse to the required location in an obstruction free space as per the input location of the user using, method of trilateration.

1.2 Motivation

Global Positioning System is well known means of outdoor positioning. It fails to do so in indoor environment. It is important to acquire position in indoor environment because it can be used as navigation system in any library or colleges or it can be used to keep a track of labour and goods in underground construction site like tunnel. Nearly all modern buildings are equipped with Wi-Fi access points, indoor positioning using IEEE 802.11 standard has now become a realistic alternative .Moreover, recent smartphones are commonly equipped with Wi-Fi sensors, which make them adequate devices to implement such an indoor positioning system.

2 HARDWARE OVERVIEW

2.1 ESP8266 WiFi Module

The ESP8266 WiFi Module is a self contained SOC with integrated TCP/IP protocol stack that can give any microcontroller access to your WiFi network. The ESP8266 is capable of either hosting an application or offloading all Wi-Fi networking functions from another application processor. Each ESP8266 module comes pre-programmed with an AT command set firmware, meaning, you can simply hook this up to your Arduino device and get about as much WiFi-ability as a WiFi Shield offers. The ESP8266 module is an extremely cost effective board with a huge, and ever growing, community.We used ESP8266 as station point and access point(but we then moved from ESP8266 to routers as access point because esp signal get distorted easily also it get attenuated.Error in the case of esp is more so, it was advantageous for us to shifting to wireless routers.)

Figure 1: ESP8266 as Access Point

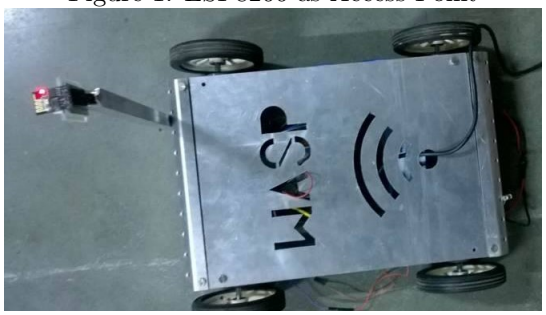
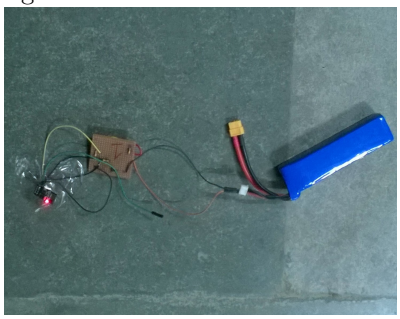


Figure 2: ESP8266 as Access Point



2.2 Access points:Routers

A wireless router is a device that performs the functions of a router and also includes the functions of a wireless access point. It is used to provide access to the Internet or a private computer network. It can function in a wired LAN (local area network), in a wireless-only LAN (WLAN), or in a mixed wired/wireless network, depending on the manufacturer and model.

2.3 Arduino ATMega2560

Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online.

2.4 FTDI Programmer Basic

The FTDI chips implement the USB protocol stack. The responsibility of this hardware is to tell your PC what it is (using some identification information) such that your computer can load the right driver for it, and also to manage the data transactions with the PC there-on after - look up USB endpoints for a better explanation of these processes. Once those drivers are loaded, this

would sort of specify a command set that your PC can use to query the chip. This hardware takes care of one side of the equation (communication with your PC). The other side of it would be some dedicated hardware to manage the UART protocol which includes logic, buffers and line drivers and the sorts. The command set that I mentioned earlier would be used to read from or write to the UART hardware. It should probably be mentioned that USB devices are polled by the PC, so in instances where you are using code which is event based, your PC is actually doing some polling to determine that new data has arrived - this may be different than a native serial port.

2.5 Motor Driver

The L298N H-bridge module can be used with motors that have a voltage of between 5 and 35V DC.H-Bridge's are typically used in controlling motors speed and direction, but can be used for other projects such as driving the brightness of certain lighting projects such as high powered LED arrays.

2.6 Buck Convertor

The Buck Converter we used in SMPS circuits where the DC output voltage needs to be lower than the DC input voltage. The DC input can be derived from rectified AC or from any DC supply. It is useful where electrical isolation is not needed between the switching circuit and the output, but where the input is from a rectified AC source, isolation between the AC source and the rectifier could be provided by a mains isolating transformer. The switching transistor between the input and output of the Buck Converter continually switches on and off at high frequency. To maintain a continuous output, the circuit uses the energy stored in the inductor L, during the on periods of the switching transistor, to continue supplying the load during the off periods. The circuit operation depends on what is sometimes also called a Flywheel Circuit. This is because the circuit acts rather like a mechanical flywheel that, given regularly spaced pulses of energy keeps spinning smoothly (outputting energy) at a steady rate.

2.7 IMU

An inertial measurement unit (IMU) is an electronic device that measures and reports a body's specific force, angular rate, and the magnetic field surrounding the body, using a combination of accelerometers and gyroscopes, along with magnetometers. IMUs are typically used to maneuver aircraft, including unmanned aerial vehicles (UAVs), among many others, and spacecraft,

including satellites and landers. Recent developments allow for the production of IMU-enabled GPS devices. An IMU allows a GPS receiver to work when GPS-signals are unavailable, such as in tunnels, inside buildings, or when electronic interference is present. A wireless IMU is known as a WIMU.

The IMU is the main component of inertial navigation systems used in aircraft, spacecraft, watercraft, drones, UAV and guided missiles among others. In this capacity, the data collected from the IMU's sensors allow a computer to track a craft's position, using a method known as dead reckoning.

3 SOFTWARE OVERVIEW

3.1 Arduino IDE

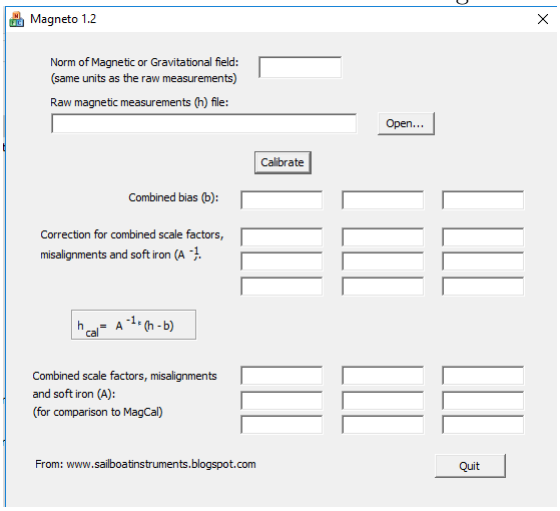
The Arduino project provides the Arduino Integrated Development Environment (IDE), which is a cross-platform application written in the programming language Java.It is designed to introduce programming to artists and other newcomers unfamiliar with software development. The Arduino IDE supports the languages C and C++ using special rules to organize code. The Arduino IDE supplies a software library called Wiring from the Wiring project, which provides many common input and output procedures. After compiling and linking with the GNU toolchain, also included with the IDE distribution, the Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal coding that is loaded into the Arduino board by a loader program in the board's firmware. Arduino IDE 1.6.9 was extensively used in this project.

3.2 AT Firmware

AT firmware is a program that runs on ESP-8266 and has built in command for various functions such as configuring modes of ESP, fetching RSSI values of WiFi signals, sending and receiving data from a server,etc.

3.3 Magneto

We used a software named Magneto 1.2 which was used to remove the error in magnetic field present due to other devices present there.



In the above we give norm of the field and the raw data in .txt format and the software itself give the combined bias (b) and an

$$A^{-}1$$

matrix. Using the formula given for h(cal), we get the final calibrated value of the data, either magnetic field value or acceleration value.

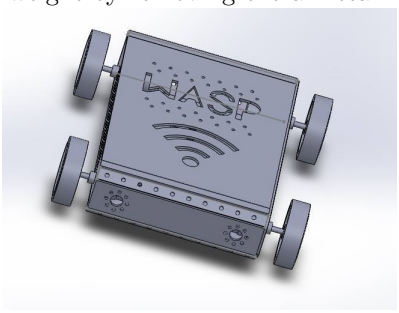
3.4 GNU Octave

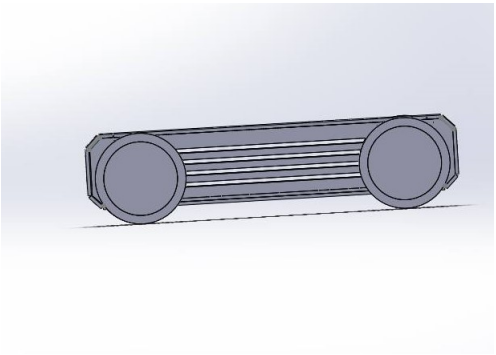
GNU Octave is a high-level interpreted language, primarily intended for numerical computations. It is quite similar to Matlab. It provides platform for data visualisation and manipulation. We used it for plotting our RSSI reading at the trial points to plot Distance vs RSSI fig. 4.4.a Log(Distance) vs RSSI fig. 4.4.b and for obtaing the best fit line for Log(Distance) vs RSSI fig.4.4.c.

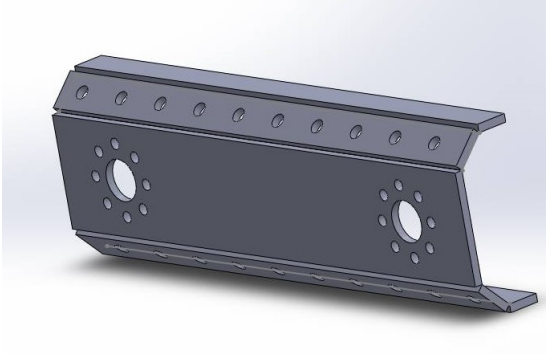
4 DESIGN OVERVIEW

4.1 Design of Bot

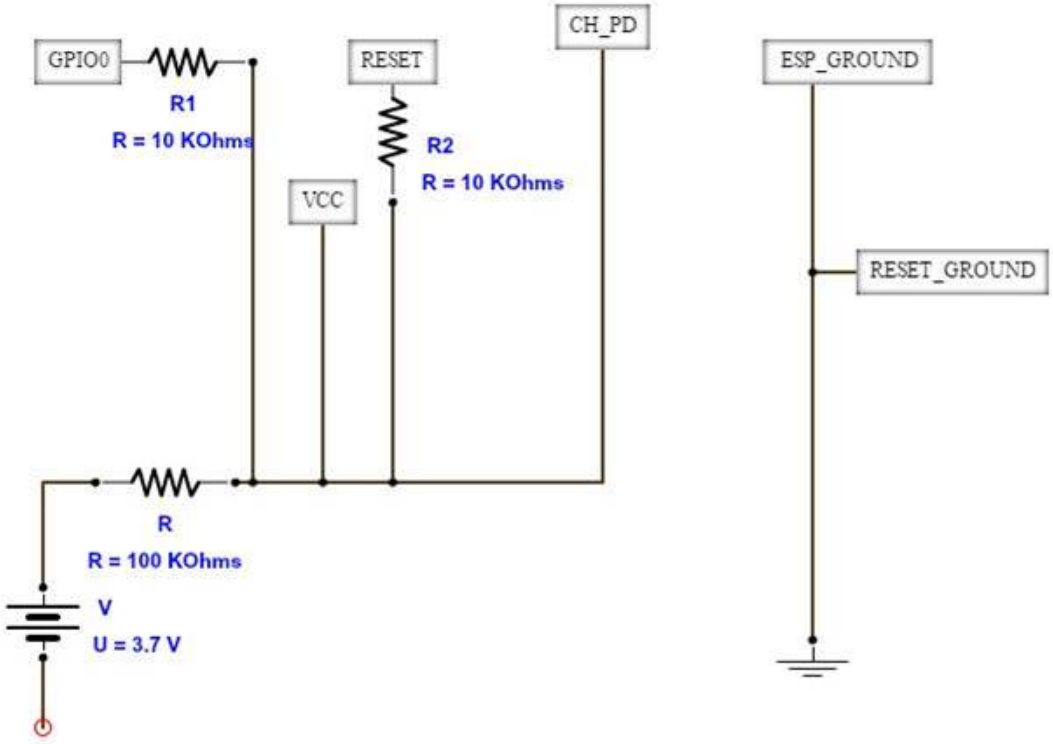
The bot has been designed on Solidworks (designing software). It consists of three steps, firstly to design every part of the bot separately on Solidworks which included top, bottom, sides, front, back, motors and wheels. This is important to even make designs of motors and wheels to make sure your bot finally gets proper ground clearance and proper space availability for fixing of motors. Along with it you need to make holes in every part for screws. Then secondly, we have to do assembly of all our parts in Solidworks which makes us ensure about the credibility of our design and to find mistakes if any. This step is tricky as any fault found in this step would lead to going to step 1 and redesign the faulty parts. When you are done with assembling then comes the third step i.e. to make a DXF file (an unfolded 2-D version of your 3-D parts) of your parts so that you can give your design for manufacturing (We used water jet cutting to make our parts). While designing parts we must also try to minimize the weight by removing extra metal from our bot which will improve the performance of our motors.







4.2 Circuit's Design



Circuit of ESP used as acess point Circuit of ESP used as station point

4.3 Our Arena

The arena for our bot testing was suitably chosen to be LHC foyer. In which we adopted the grid system. The entire arena was divided into 156 square grids each of dimension 55X55 square cm. Two(hypothetically) perpendicular axes were setup as x-axis and other as y-axis. Several points were marked as testing points whose readings of signal strength from four routers fixed on the outskirts of the arena were later fed to the Adruino.

fig. 4.3 LHC FOYER

5 WiFi POSITIONING SYSTEM

5.1 What is WiFi Positioning System?

Since GPS is inadequate due to various causes including multipath and signal blockage for positioning indoors so WiFi positioning was a better option. Positiong is done using wireless acess points based on the intesity of the received signal. Four of such wireless access points (routers) were used and placed as shown in Fig 4.3.
Localization technique used in our case was based on RSSI and lateration i.e. based on the signal strength received by the client (ESP8266-01) from the access points the distance is calcualated from each of the AP's using the relation between log of Distance vs Signal Strength [section- ??] . Trilateration (multilateration in our case) techniques is then used to calculate the estimated client device position relative to the known position of access points.

5.2 Free Space Path loss

Free Space Path Loss (FSPL) is the loss in signal strength of an electromagnetic wave (WiFi Signal in the present case). Free-space path loss is proportional to the square of the distance between the transmitter and receiver, and also proportional to the square of the frequency of the signal. [source- Wikipedia].

WiFi signal Strength is measured in dBm. FSPL in terms of dB is:

$$\begin{aligned} FSPL(dB) &= 10\log_{10}\frac{(4df\pi)^2}{c^2} \\ &= 20\log_{10}\frac{4df\pi}{c} \\ &= 20\log_{10}(d) + 20\log_{10}(f) + 20\log_{10}\frac{4\pi}{c} \\ &= 20\log_{10}(d) + 20\log_{10}(f) - 147.55 \end{aligned}$$

d- distance between emitter and receiver
f- frequency of wifi Signal (lies between 2400-2500 Hz)
c- speed of light

So, that was the theoretical part. Now, in order to get the relation between RSSI and distance for all of the four routers RSSI data from the routers was taken at 156 points in the arena and then subsequently plotted.

5.3 Data Collection

5.3.1 Aim

To get a relationship between RSSI(received signal strength) and Distance(between Access point and Client) we chose 156 points in our arena where we would take the RSSI values and subsequently plot them using Octave .

5.3.2 Reading1

The setup for the first reading was as follows:
– Four ESP8266 with their separate Lippo batteries were placed at the ground level at the corners of the side of the arena.
–The client ESP was connected to the laptop through which command were sent to it, via FTDI programmer basic.(see the fig below)

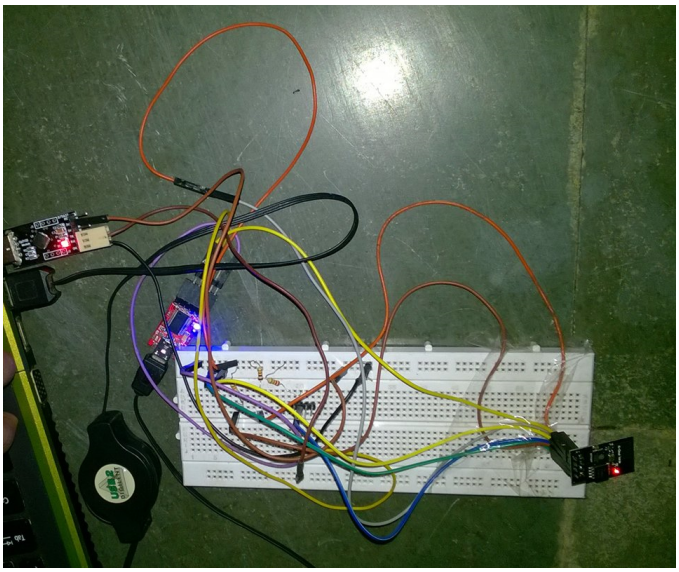


fig 5.3.2.a FTDI

Due to atteneuation of the wifi signal and the fluctutating signal strength from the ESP's, the reading was not up to the mark. The graphs that were plotted subsequently were as follows:

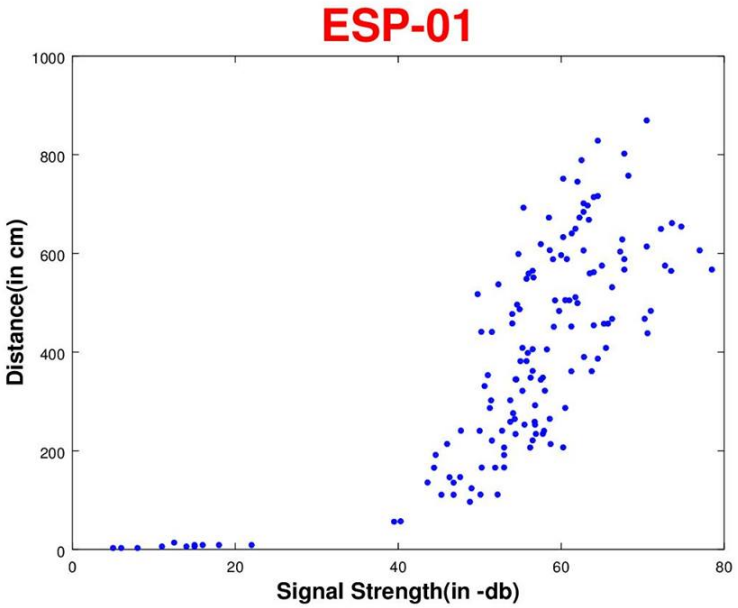


fig 5.3.2.b ESP-1.1

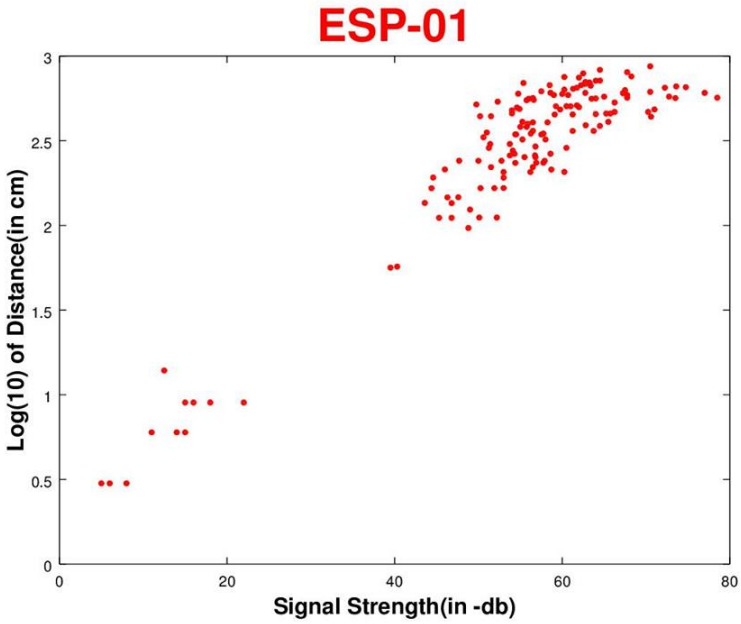


fig 5.3.2.c ESP-1.2

5.3.3 Reading2

Since the reading 1 was not what was expected theoreticaly so we took reading 2. The setup for the second reading was as follows:
– Four ESP8266 with two one lippo battery for two of them were placed at around 50cm above ground level again at the corners of the side of the arena using thermocol stands . – The setup for the client ESP was same as in Reading 1. Again due to fluctuating values of signal strength from the ESP's, following plots were obtained :

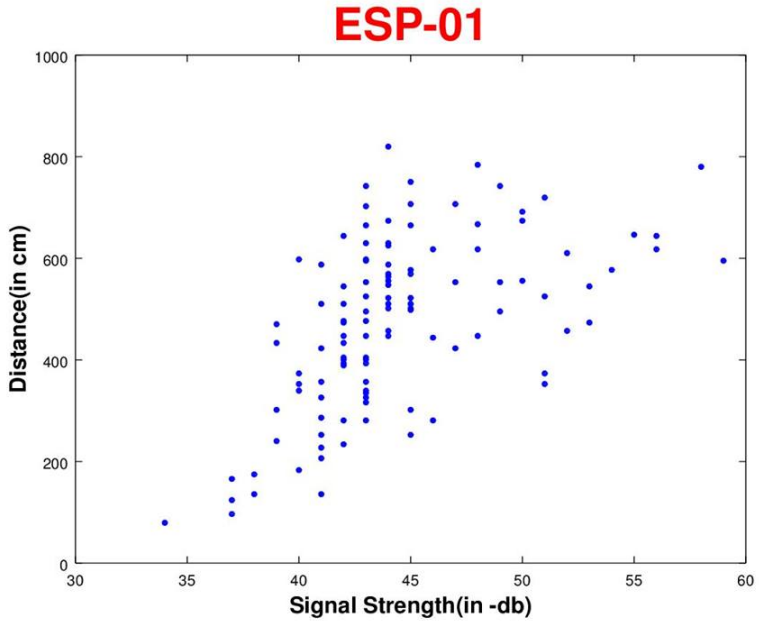


fig. 5.3.3.a ESP 2.1

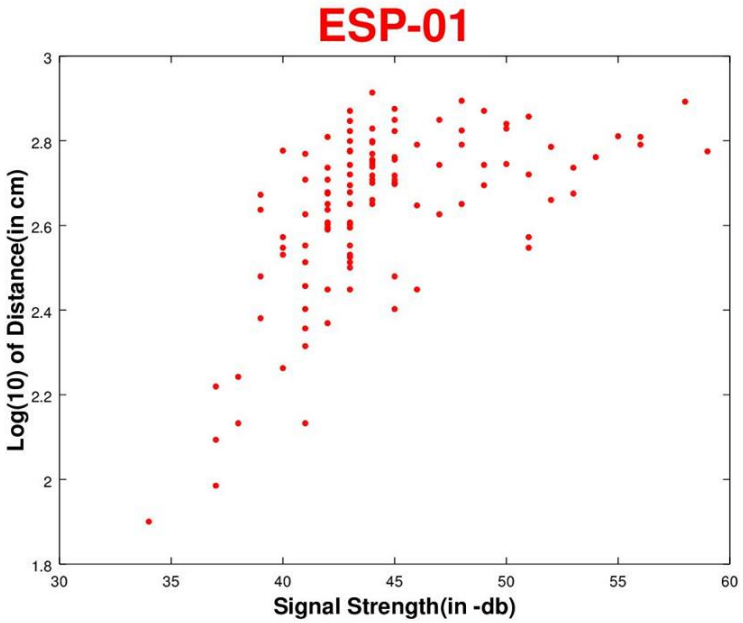


fig. 5.3.3.b ESP 2.2

5.3.4 Reading3

Since the reading 1 and 2 both were not what was expected theoreticaly so we decided to take reading 3. This time the setup had major changes as follows:
– Instead of using ESP8266 as access points, 4 D-Link WiFi routers were placed at the mid of the sides of the arena. This time reading were taken with client i.e. the ESP mounted on the bot, connected to the arduino which sent the commands to seek RSSI at intervals of 10 seconds (so that we can shift the bot to next position) and subsequently sent the RSSI values to the monitor. Before taking reading 3, it was decided to take the around 80 readings with a single WiFi router at equal distances of 20cm with ESP8266 as client. Surprisingly, the plot was very close to theoretical plot.(see the fig. below)

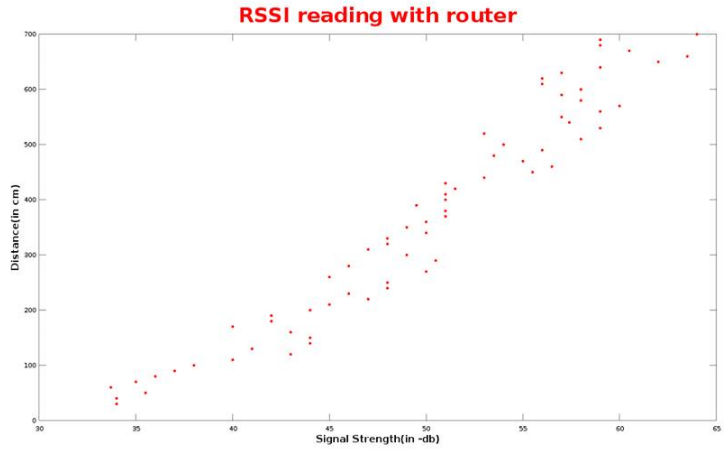


fig. 5.3.4.a RSSI READINGS WITH ROUTER.

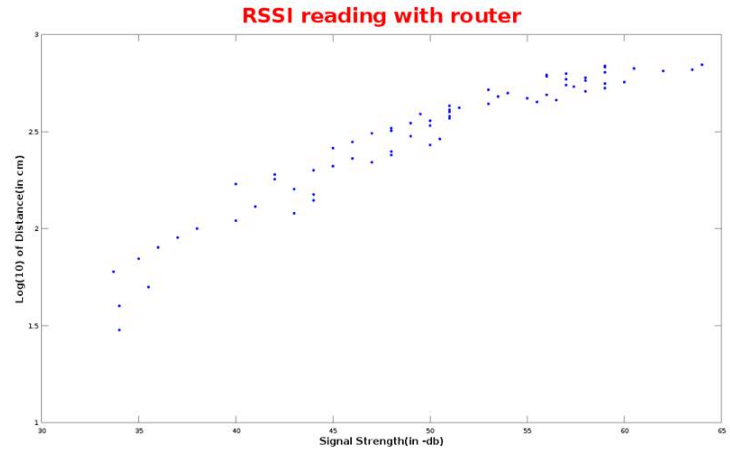


fig. 5.3.4.b LOGARITHM OF RSSI READINGS WITH ROUTER.

And the final plot was as follws:

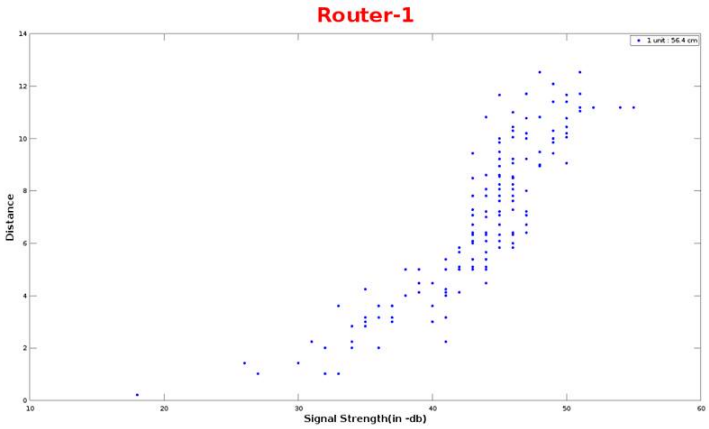


fig. 5.3.4.c WASP FINAL READING

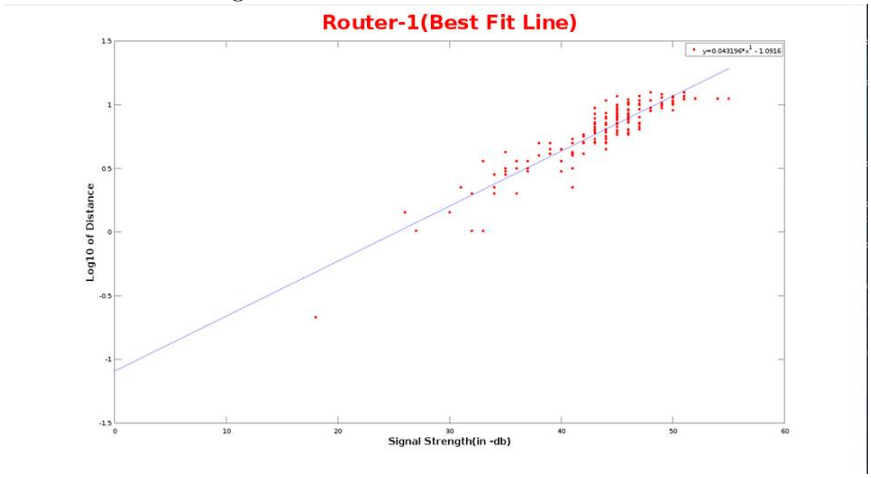


fig. 5.3.4.d BEST FIT LINE

5.4 Trilateration

5.4.1 Why Trilateration ?

In geometry, trilateration is the process of determining absolute or relative locations of points by measurement of distances, using the geometry of circles, spheres or triangles.

5.4.2 Application in our project

To increase accuracy of our trilateration we used 4 routers means 4 points with known respective distances from current position of our bot. There are many algorithm we can use to solve the equations . To solve these equation we used matrix multiplacation.

6 BOT CONTROL SYSTEM

6.1 IMU Calibration

We used GY-85 IMU for our project which has 9 degrees of freedom. IMU is a device which has 3 sensors accelerometer, magnetometer and gyroscope. Accelerometer of IMU works on the principle of pseudo forces. You may assume that ball is kept in a box each face of box is supplied with same voltage and when the ball touches any face voltage change is proportional to force it applies on that wall. GYROSCOPE works by calculating the magnetic feilds around IMU in X and Y direction. To get output from the IMU we used some parts of the code Code used for getting output from IMU . IMU gives us output in bits which is proportional to readings it has calculated. We then calculates actual reading by using senstivity and V-reference which is constant voltage accross each face of BOX with ball .
BUT the readings which we were getting initially were not good and uncalliberated on which we could not rely on. The acceleration and mainly magnetometer readings which we were getting from IMU were not reliable because as we were interchanging any two axes from X,Y,Z our reading did not interchange rather there was a significant difference beetween them. Hence we used software called "MAGNETO" for calliberation of IMU readings. we used bias and caliberating matrix to calliberate readings every time we take reading from IMU for both Acceleration and Magnetometer.

6.2 Kalman Filter

Kalman filter is bassically a ALGORITHM which decides which is the most trusted value using outputs from 2 or more sensors. Eack sensor sensor returns a value which accounts the actual reading with probability distibution around it. Kalman filter helps to get new mean and new probability distribution filtering 2 or more sensors. We used two ways to calculate the current position of our bot. 1. Wifi-positioning 2. IMU with the help of previous position These two methods have their respective mean value and certain praobability distribution.
Mean and covariance matrix via wifi-psitioning We programmed our ESP module such that it took 4 reading from each router we have placed in arena and then gives out the mean of position. For covariance matrix we devided arena into 8 parts . figure of arena Areas marked 1,2,3 and 4 are more accurate than 5,6,7 and 8. For covariance matrix we checked the position via wifi positioning at each integer coordinate of that region and find out covariances in each region using this formula -

$$covarience\ x = \sqrt{\sum (x_{true} - x_{wifi})^2}$$
$$covarience\ y = \sqrt{\sum (y_{true} - y_{wifi})^2}$$
$$covarience\ xy = \sqrt{\sum (xy_{true} - xy_{wifi})^2}$$

Mean and covariance matrix via IMU we code our bot such that it moves for approx. 5 seconds and then it takes reading from routers for wifi positioning. While moving it takes 4 reading in each loop and stores avg. in an array so that we can use them at the end of 5 sec for calculation.

The reason behind duing so was only to minimize looptime for next calculation because more is the loop time less is our accuracy. For calculating Distance via IMU we used this formaula simple kinematics equation for each time interval.

$s_f = s_i + \frac{1}{2} \times (acceleration) \times (looptime)^2 + u \times (looptime)$

hence, smaller the looptime higher is the accuracy.
Update matrix for IMU each time we calculate distance traveled using IMU we have update to covariance matrix for IMU as error in distance calculated by IMU passes for further calculations also. Hence updatation is needed. Detailed explanation on
http://www.bzarg.com/p/how-a-kalmanfilter-works-in-pictures/ and youtube link for understanding kalman filter OUR CODE LINK

7 Conclusion

Overall at the end of our project we were able to make a bot which could detect its positions using wifi positioning system. The accuracy of the positioning system was about 50-70 cm. We will be continuing the project further in our semester and following will be our future coarse of action:

1. Make the system wireless so that commands to the bot could be given through phone or laptop.
2. Improve the accuracy of positioning with kalman filters, PID and increasing the RSS reading frequency of the ESP8266 (we'll be using node lua for programming it this time).
3. Upgrade the positioning system so that it can navigate in the arena where obstacles are present too.
4. Create an android app so that the map of the arena (the obstacles,positions of routers,etc) and the instructions could be sent to the bot through the app.
5. Upgrade the bot so that it can pick and place objects from and to desired locations. Image processing would be used here for recognizing the objects.