

Performing Arts and IOT

Rupanagudi Rakshit

March 2019

Contents

1	Introduction	1
2	State of the art	2
2.1	Why this Project?	2
2.2	Initiation	2
3	Device Understanding	3
3.1	Sensors	3
3.2	Micro-controller	5
3.3	Sloeben Eclipse IDE	5
4	Phase 1	6
4.1	Pattern Acquisition	6
4.2	Software Architecture	6
4.3	Hardware Architecture	7
4.4	MQTT Protocol	8
5	Phase 2	10
5.1	Pre Pattern Analysis	10
5.1.1	Multiple Wearable Devices	10
5.1.2	Technicality of the multi-devices	11
5.1.3	Issues faced and Solutions	13

1 Introduction

Dance is an art. It is the form of expressing ones feelings symbolically using mental and physical aspects. Technically portraying an art like Dance is the recent ideas which has evolved out to be a challenging tasks from the past decade. On the other hand, technologies like IOT(Internet of Things) is in a huge demand, wherein the world is looking to integrate IOT with everything. The bridge between Art and technology is IOT. The Dance IOT is a project where the dance arm movements are analysed and mapped onto several functional platforms(may be control of the auditorium's lighting system, volume system etc..),

so that the performance is enhanced and makes the audiences awestruck. This is more constrained to bring up new era in the performing arts like dance, theatre plays, magic shows and circus as well. The whole idea is to please the audiences with the innovative technologies that lead to great performances.

2 State of the art

2.1 Why this Project?

Performing Arts in an aspect of entertainment is growing in all the dimensions. As per the survey conducted by Quentin Fottrell, a US based surveyor, it is observed that people spend most of their time in getting entertained, both through live performances and through internet and still want for entertainment. People now a days expect entertainment in new manners, totally updated to the interest of current generation and surely expect a WOW factor in it. Creating such thing in an innovative manner requires the proper usage of technology.

Collaborating technologies like IOT with Performing Arts is both challenging as well as interesting. So, using technology to further add up to the growth of already growing performing arts is the task .

2.2 Initiation

Before going to the existing methods and stuff like that, the project must be initiated and analysed. This project is a collaboration of art and technology. The main motto lies in using IOT as the combining tool. There would be possibly four main domains in this project, categorised on the basis of modular functionality in achieving it. The following are the functionaries:

1. Pattern Acquisition
2. Pattern Analysis
3. Performance analysis

Pattern Acquisition:

Pattern acquisition is the process of reading data in the form of samples. There are various methods in doing so. Taking data in the forms of samples manually, or even computerised data acquisition. We are doing it as combination of both Manual-computerised data acquisition. The result of using this method would be discussed later in the Performance section.

Pattern Analysis:

Pattern Analysis is the process of understanding the read data. Here in our project there is not only a need for understanding data but also the need for differentiating data. Differentiating here refers to the process of the micro-controller understanding which sample is required for the task and which is not.

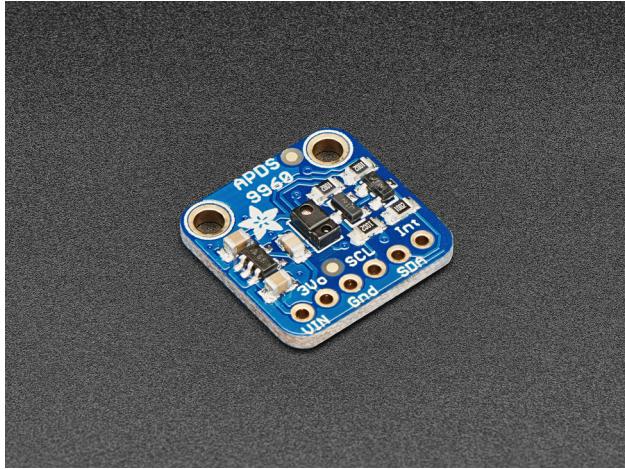


Figure 1: APDS9960

The usage of Hidden Markov Models(HMM) [1] found to be the most accurate models in this kind of usages and applications.This should also be determined only when the performance is fully studied.

Figuring all the possible ways, this can be further extended to find out a mathematical mapping relation between the sensed data and the assigned specific task .The process of designing such a system requires the perfect understanding of the components and their outcomes.

For better understanding and also to uphold the terms of modularity, I have divided the project into three phases,they are:

1. Phase 1 - Pattern Acquisition
2. Phase 2 - Pattern Analysis
3. Phase 3 - Performance Analysis

3 Device Understanding

3.1 Sensors

Generally , there are sensors which can sense motion and detect its path of movement as well.Out from various available sensors we are using APDS9960 and LMS9DS1. They are the two most common sensing devices used in motion detection.They bring out a great user-friendly environment with standard accuracy.

1. APDS9960 :

includegraphics[]fig/apds9960.jpg

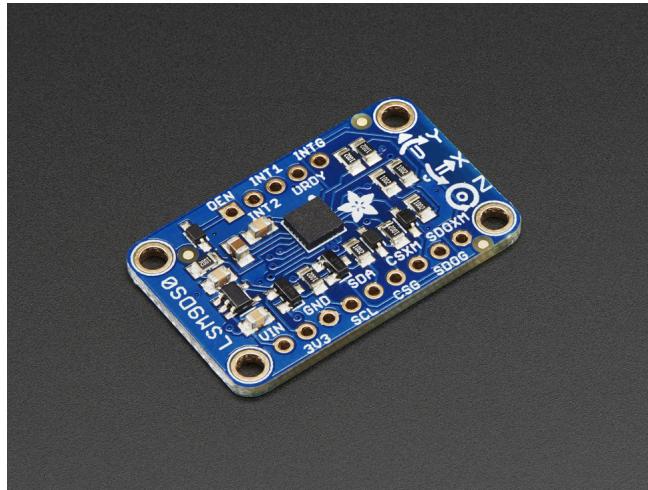


Figure 2: LMS9DS1

APDS9960¹ is a multi-functional sensor that mainly senses proximity,RGB, light and gesture of an object. It is the most popular sensor used in gesture detection. It is to be connected to a micro-controller for the action and control process of it. It generally returns the gesture detected(right, left, up, down) and even complex gestures such as zigzag,near,far,etc.. It also returns the values in Red, Green, Blue attributes of the object that it detects. When coming to the terms of proximity it returns the distance of the object detected with up to 8 bit resolution. As mentioned above it is a multi-functional sensor. It has six pins where VIN is for the input voltage, 3Vo is to power up the sensor, Gnd to ground the sensor so as the unwanted voltages are grounded, SCL is the serial clock, SDA is serial Data and, INT is the interrupt.

2. LMS9DS1:

includegraphics[scale=0.25]fig/LMS9DS1.jpg

LMS9DS1² is a motion sensing sensor. It is a combination of accelerometer, gyroscope and magnetometer. In depth it has a 3-axes accelerometer, 3-axes gyroscope and 3-axes magnetometer. So collectively its good for all the possible 9 degrees of freedom(DOF). It supports both I2C and SPI. It measures angular velocity, acceleration and magnetic heading and returns them to the micro-controller. It measure the above mentioned parameters in a three dimensional field resulting in 9 outcomes i.e angular velocity in x,y,z, and acceleration in x,y,z, and magnetic heading in x,y,z. Thus, this sensor outcome is more reliable and best to analyse.

¹(for specifications: <https://cdn-learn.adafruit.com/downloads/pdf/adafruit-apds9960-breakout.pdf>)

²(for details: <https://www.adafruit.com/product/2021>)

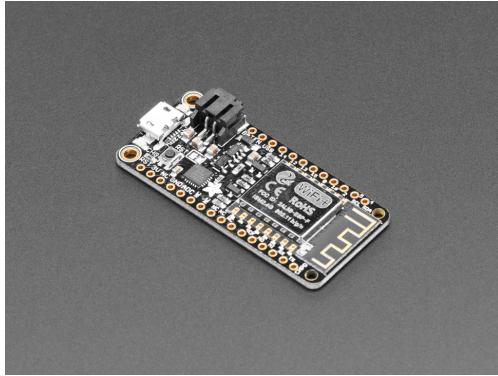


Figure 3: ESP8266

3.2 Micro-controller

A micro-controller is a user friendly electronic device which is generally used to perform a specific task. Here, task may be either one time or an infinite number of times, it is user controllable. User will have to code the required task in the form of an assembly language (or language with effective to the micro-controller, mostly a high level language). The micro-controller converts the code written into its understandable language and performs the task as per the code written by the user.

includegraphics[scale=0.8]fig/esp8266.jpg

In our project we are using ESP8266 as a micro-controller. An ESP8266 is a versatile micro-controller, it has an on-chip wifi module which gives the option to the user to connect to a cloud or server based on his need. This is similar to an arduino micro-controller but has more performance stability and advanced features than an arduino has. This is the main reasons for the usage of it in complex projects. Projects filled with real time robotics, real time analysis prefer ESP8266 over other micro-controllers that are available in the market.

3.3 Sloeber Eclipse IDE

Sloeber Eclipse IDE is an open source user-interface, generally a tool that is used to code and manage the micro-controllers. Sloeber is used in our project to manage the code that is used to perform our project. It has code files that are in cpp format and as well as python format. It is multi-disciplinary tool, since it can manage code of different formats. We also use sloeber to set a connection for the module and the server precisely speaking, setting a wifi connection.

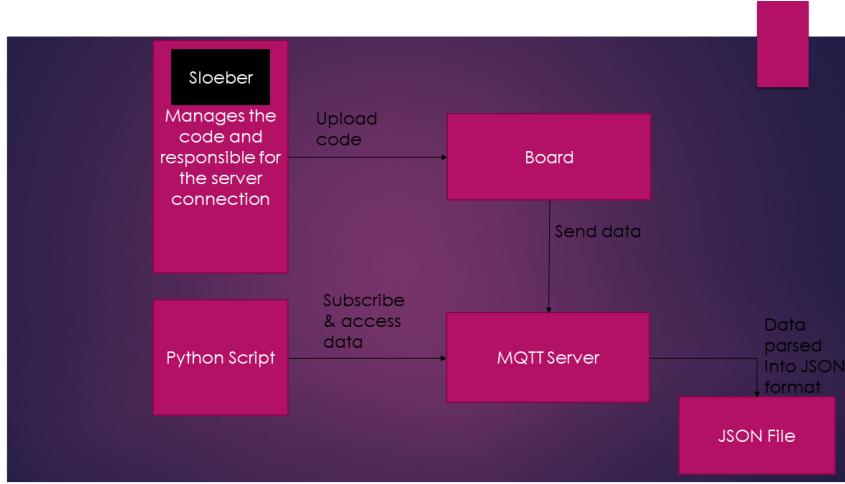


Figure 4: Software Architecture

4 Phase 1

4.1 Pattern Acquisition

Pattern acquisition is the process of collecting a large number of data sets relevant to the task to be performed. Pattern Acquisition is a result obtained by the sensors through control by the micro-controller.

4.2 Software Architecture

includegraphics[scale=0.45]fig/sad.png

From the figure above, it can be inferred that the total process of data acquisition involves multiple tasks. Considering each element represents a task, the element's description as below:

- Sloeber: Sloeber is a user-friendly software that is generally used to operate on with advanced, smart micro-controller devices. Generally this is analogous to an Arduino IDE, but as mentioned above it is used to program high end micro-controllers. Here, Sloeber is used to code the required lines that program the task required and manage the code. Set up the wifi connection. This is totally responsible in managing the action of the wearable device(Board). Uploading the code through sloeber to the board has to be done.
- Board: Board is a wearable device that comprises of the different sensors, and the micro-controller. This physical device is responsible for interacting with the server as it has been programmed using Sloeber. Data

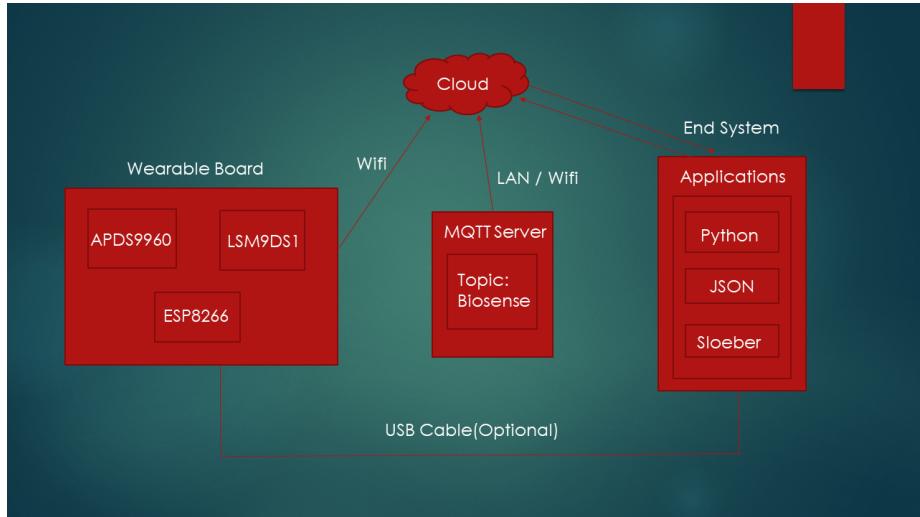


Figure 5: Hardware architecture

measured, is sent to the server from the board. Measuring data is by moving/rotating/positioning the board. Everything is counted as data. This is sent to the MQTT server.

- **MQTT server:** MQTT server, simply a server is a cloud that stores data in it collectively in the form of topics. Access is granted when the user/device is subscribed to the topic inside the server. As this project uses MQTT protocol for the storage of data, connection and access of stored data it is called as a MQTT server. MQTT server is clearly explained in the next section.
- **Python Script:** Python script here is precisely the code responsible for accessing and receiving the data that is stored in the server. Apart from the sloeber code; the code responsible for writing the sensed data on to the server, the python script must be run parallelly so that the data stored in the server can be retrieved or read back. The code is written such that the data read is appended into a JSON file. JSON file stores all the data coming from the server and keeps on appending the data into itself.

4.3 Hardware Architecture

From Figure 10 we can see the hardware architecture diagram. The action performed between the board and the server systematically is represented in the figure above.

When the code is uploaded onto the module (combined micro-controller connected with the servers), the program runs and the results are obtained on successful compilation. The results are sent to a server using MQTT protocol.

Once the server starts storing data, we can retrieve it back. This set of data received is called a pattern. Depending on the action or movement of the wearable module, the values change. So, in case of waving a hand, the action must be performed so that the server stores the action and the user can then successfully retrieve data. This action is one sample of a pattern. With varied acceleration and speed we need to record as many samples as required for a single pattern. The pattern is obtained in a JSON format. Java Script Object Notation popularly known as JSON is a flexible text format that can store data and communicate it with other devices. So, anything related to a cloud or server based data system a JSON format must be used and hence we are using it. Finally my Pattern is acquired in the form of JSON.

4.4 MQTT Protocol

As mentioned above, we have to use a platform for sending data, storing it onto the server (may be cloud also) and retrieve the data from the server, function it. This so called platform works on some set of rules called protocol. The protocol used here is MQTT.

Message Queuing Telemetry Transport, shortly known as MQTT³ is an International Standard Organisation (ISO) publish-subscribe based messaging protocol. It works on top TCP/IP protocol i.e. apparently works in the application layer of the TCP/IP protocol. It is a publish-subscribe messaging pattern, and thus requires a messaging broker. It contains no. of clients communicating with the server, often called a "broker". A client can be either a publisher of message or a subscriber who generally expects and accepts data or messages from the server. All clients can connect to the server through proper subscription.

In order to have a connection, the publisher need not have any information on the number of subscribers or location info about the subscribers. Similarly the subscriber also need not have to be configured with any data of the publisher. This turns out to be an advantage for the usage of this protocol.

MQTT relies on TCP/IP for data transmission. In case of UDP or Bluetooth MQTT-SN is used. For transmission of data, there must be a connection between the two entities i.e. the client and server. There is a separate set of data that controls the action of connection and other stuff related to it called as "Message Type". This is generally a control message. The minimal size of a control message would be 2 Bytes which can be maximum extended upto 256 Mega-bytes. There are 14 pre-defined Message Types. A few of them include commands to connect and disconnect a client from a broker, to publish data, to acknowledge receipt of data, and to supervise the connection between client and server.

Connect: Waits for a connection to be established with the server and creates a link between the nodes.

Disconnect: Waits for the MQTT client to finish any work it must do, and for the TCP/IP session to disconnect.

³(<https://en.wikipedia.org/wiki/MQTT>)

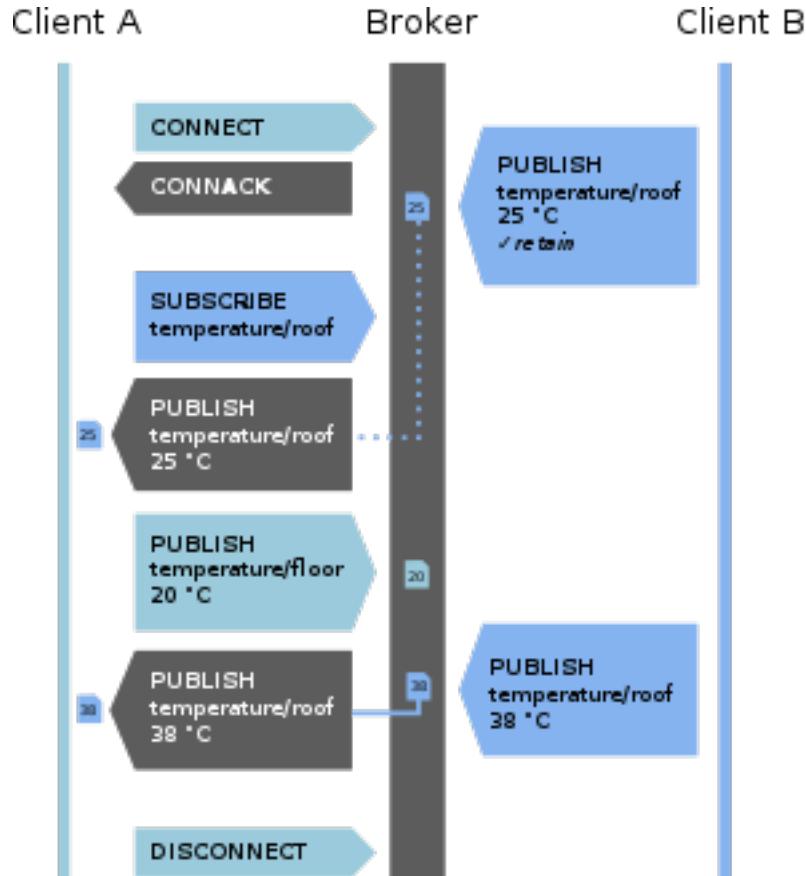


Figure 6: MQTT Protocol Handshaking

Publish Returns immediately to the application thread after passing the request to the MQTT client.

includegraphics[]fig/mqtt.png

MQTT just sends connection credentials in text format, there is no measure of security or authentication involved in it. TCP is the most reliable protocol used in today's world. As we are using TCP, it is responsible for the security and measure to protect integrity of the transferred information from interception and duplication. The connection alone is not sufficient, for a better communication one must know the Quality-of-Service(QOS) of the connection formed. The QOS is a measure of wellness i.e how good a connection is established and the how good the bandwidth is utilised. The quality of service measures are as follows: At most once - the message is sent only once and the client and broker take no additional steps to acknowledge delivery (fire and forget). At least once - the message is re-tried by the sender multiple times until acknowledgement is received (acknowledged delivery). Exactly once - the sender and receiver engage

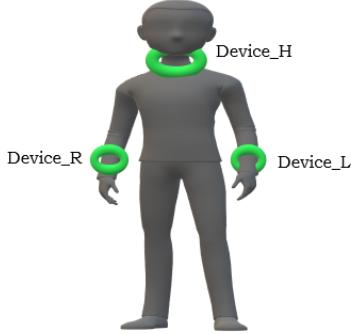


Figure 7: Person with 3 wearable devices

in a two-level handshake to ensure only one copy of the message is received (assured delivery).

5 Phase 2

5.1 Pre Pattern Analysis

5.1.1 Multiple Wearable Devices

Coming to the point; as discussed earlier, we expect this project to be a real time case, wherein the primitive essence lies in the data/pattern read or recorded accordingly. We can firmly tell that the data is the main concern here because it is quite obvious that data read is to be mapped on to, either the robot or a task. Say, there would be an error or misread of data i.e the pattern acquisition part failing partially/fully, the whole process of mimic or task would be an epic fail.

We have discussed earlier that our technical team has readily built a wearable device, which includes all the modules as discussed in chapter 2, and performs accordingly as discussed in chapter 4. The complexity begins here, now we have decided to include multiple wearable devices(say 3), in order to make it more realistic.

We can give an example, consider an artist using both his hands and face(precisely head) for portraying a dance move or an expression in the dialogue of a play, if each of this is symbolising something either collectively or individually, we can treat this as the indication for us to make either the robot mimic as the artist or perform a specified task.

Consider me in a dance performance, where I'm widening my both hands and waving them to and fro towards each other, it symbolises something, may be happiness. Now the act actually is responsible for the data recording process(since we have the devices worn). If I have only one device worn, the data

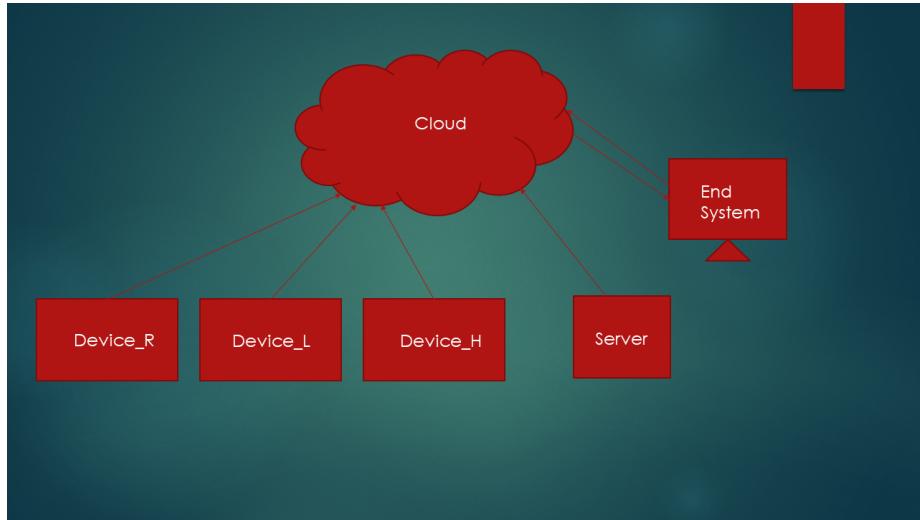


Figure 8: Multiple devices Server Access

from the alternate hand or head cannot be read, which is incomplete and does not meet the project's real time views. Considering this, we can understand the importance of multiple devices usage. Hence, we would now start the usage of multiple devices.

5.1.2 Technicality of the multi-devices

Now the technicality here, focuses on the configuration and connectivity of the multiple devices. We being human beings having our brains can manually identify and separate the devices based on some aspects. But the issue arises, when it comes to the server accession.

The device uses MQTT protocol to access the server as mentioned above in the chapter 4. So, according to the MQTT protocol, the server grants access to data if subscribed to the topic correctly irrespective of the number of clients. Similarly the client(i.e the wearable device) also need not require any information about the publisher. This would not be a problem for server.

Once the data is received from the server, for the so called process of analysis, it would be difficult for us to understand which data has come from which device. In order to eliminate such confusions, we configure each device with a different board id.

Now, each of the three devices are assigned with separate board ids, readily access the topic Biosense in the server, for publishing the data. It is now a process where you configure the devices by uploading the code onto them. Uploading code to each device, connecting to the prescribed wifi network(this is also inclusive in the code) then powering them with battery, makes the device ready to work.

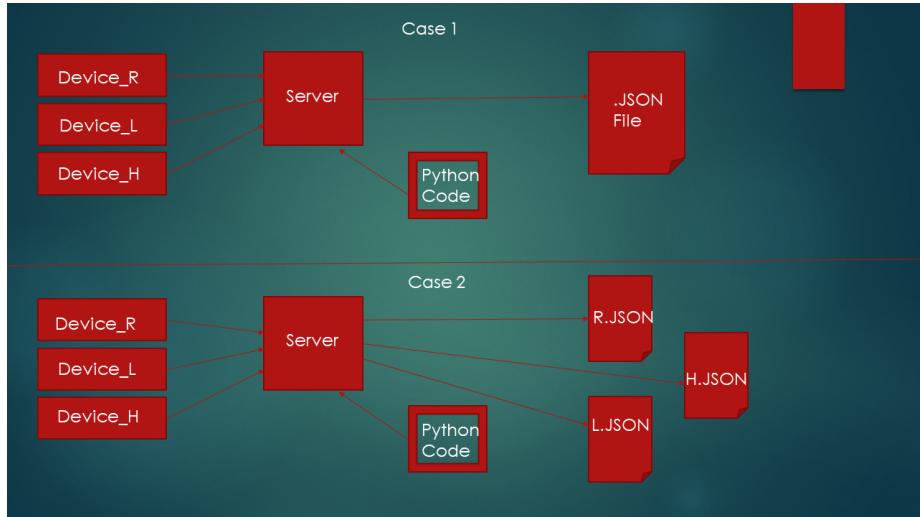


Figure 9: Cases of dumping read data into JSON format

Caution: If the network of transfer i.e wifi is changed(generally the wifi changes with the place or location changes made for the devices), all three devices must be reconfigured(i.e the code must be uploaded once again) individually and follow the rest of the steps as mentioned above exactly then same manner.

Once the devices are ready to act, the code responsible for reading data from the server must be run so that the data published by the devices onto the server can be read back and dumped into some readable file(here we read it in JSON format so, we dump into a JSON file).

According to me, here, reading the data from server can be done in two possible ways:(as shown in figure) i) All the data read irrespective of the device's board-id into a single file ii)Separation of data based on board-id and dumping into three separate files.

Considering the facts, that the project to be realistic, it is impossible for us to surely say that the data can be published and received in an exactly particular way. It all depends on the sequence of action of the performer. So, Case 2 wouldn't be the best solution for this issue. Which indeed results the case 1 to be the optimal one.

Now, the data read from the server is dumped into a single file. The data is in JSON format.The below picture represents the data read that is dumped into a JSON file. One can find the three devices data that is random to see, but actually based on the movement of the user.

From the figure, we can notice that the data is random(like how it is recorded) and there are different board ids for different devices. Say Sensor-Test-R for Device R, Sensor-Test-L for Device L and Sensor-Test-H for Device H respectively.

The JSON file obtained after the dump, is unclear and physically untidy,

Figure 10: JSON File after dump

unorganised to watch. For analysis, this would not be fruitful. So using a CSV file would be a better solution for this.

A CSV(Comma Separated Value) is a file which contains organised data in the form of tables. Data inside data types such as list, tuple, dictionary in python can be converted into a CSV format. This makes the data more clear and easy to analyse.

5.1.3 Issues faced and Solutions

Coming to the accession of the devices to the server, the devices use MQTT protocol as discussed in chapter 4. This is pure advantage that any number of devices can access the server. Every device has the same importance. If at all something matters the device accession to server that would either be physical issue of the device such as battery connection, aging, quality of connectivity to the local area network. All the devices can publish data with almost same frequency.

In our earlier trials in succeeding this, we faced issues of data being non uniform, asynchronous, delay in either of the devices when compared to the other. What we could figure out from this was only that the data was non uniform, asynchronous because it has to be non uniform. Clearly putting it, the user may first raise his right hand so R data is read, then his head moved so H data read, then his right hand again so R again, then moves his left so L. Here it is quite observable that its all in the users hand how he recorded you cannot expect data to be in RLH,RLH,RLH,RLH format. The data would be non uniform say, RRHLRHLHLHLHLHLHLHLHLLHRR.

Coming to the issue of time lapse/delay, the devices are configured such

that they have equal priority in publishing onto the server. The server is such that it provide equal bandwidth to the publishers and the clients. Then we understood that the device's aging, quality of connectivity the local area network also mattered to the time delay.

References

- [1] Holger Junker, Oliver Amft, Paul Lukowicz, and Gerhard Tröster. Gesture spotting with body-worn inertial sensors to detect user activities. *Pattern Recognition*, 41(6):2010–2024, 2008.