Sign Language Interpreter

Using Leap Motion Sensor

Sanket R Bhimani

June 27, 2016

Contents

1 Tutorial Name:

Sign language Interpreter
Using Leap Motion Sensor

Objective:

Objective behind this project is to help deaf and dump people to communicate with normal people. So this system will try to interpreter their sign language into natural voice so normal people can understand what that person want to say.

Short Intro:

Here we are using leap sensor for detecting gesture and then we will convert in audio with mp3 module

2 Prerequisites:

Before going ahead it is better to learn these basic things

- Basic knowledge of Python
- Basic knowledge of javascript and HTML
- Basic knowledge of Linux
- Basic knowledge of NLTK
- Basic Electronics knowledge

3 Requirement

- Hardware Requirement:
  - Galileo Board
  - Leap Motion Sensor
  - microSD card of minimum size of 8 GB with card reader
  - speaker
  - MP3 Module
  - LAN cable or wifi module(for connecting board with PC)

- Software Requirement:
  - Python editor (vi is enough)
  - Leap sdk
  - LeapTrainer.js
  - NLTK for Python
  - Tornado Web-socket
  - Any web browser
  - Bitwise SSH Client
  - Wireshark(optional)

4 Theory and Description

This much things you must learn before starting this project:

- Leap motion library for Python
- LeapTrainer.js
- Python programming
- Websocket
- getting started with Galileo board
- interfacing with MP3 module

Leap motion Library:

We will see about python library. All class's name are given as human body part so it is very easy to understand. Main and useful classes are,

- Frame
- Hand
- Finger
- Pointable
- Arm

For getting data from leap sensor first we need to register listener class with controller object and for that first we need to create listener class. And register that listener class object with controller object you can find the basic code for getting basic frame data from given code examples "connection_with_leap.py"

Frame:

A frame is an object which can be generated from controller object. Leap sensor sends data through this frame object in listener class. From frame object, we can find all other objects. Means all row data comes with the frame. As per leap FPS on_frame(self,controller) method of listener class is called with a new frame. Each frame has its unique id and time stamp.

In on_frame method from controller object, we can get the current frame. It also supports serialization.

Hand:

Hands are the main entity tracked by the Leap Motion controller. The Hand class represents a physical hand detected by the Leap. A Hand object provides access to lists of its pointables as well as attributes describing the hand position, orientation, and movement.

From current frame object, we can get the current set of detected hand as an array. We can get hand from hands object by these properties,

```
frame = controller.frame()
hands = frame.hands

leftmost = hands.leftmost
rightmost = hands.rightmost
frontmost = hands.frontmost
```

Listing 1: getting frame and hand object

We can get these characteristics from hand object,

- isRight, isLeft — Whether the hand is a left or a right hand.
- palm_position — The center of the palm measured in millimeters from the Leap Motion origin.
- palm_velocity — The speed and movement direction of the palm in millimeters per second.

- palm_normal — A vector perpendicular to the plane formed by the palm of the hand. The vector points downward out of the palm.
- direction — A vector pointing from the center of the palm toward the fingers.
- grab_strength, pinch_strength — Describe the posture of the hand.
- stabilized_palm_position — The stabilized palm position of this Hand.
- Motion factors — Provide relative scale, rotation, and translation factors for movement between two frames.

We can get a pitch, yaw, and roll of hand by these properties,

```
pitch = hand.direction.pitch
yaw = hand.direction.yaw
roll = hand.palm_normal.roll
```

Listing 2: getting pitch yaw and roll from hand object

Figures & Pointables:

Fingers are Pointable objects that the Leap Motion software has classified as a finger. Get valid Finger objects from a Frame or a Hand object. This can be generated from frame object or hand object. As this way,

```
frame_pointables = frame.pointables
hand_pointables = hand.pointables
known_pointable = hand.pointable(known_id)
finger = Finger(pointable)
```

If pointable is detected as the particular figure then and then only it can be returned as finger object. But mostly this pointer is detected as a finger. We can get a bone object from finger object and get the position of all joints.

```
bone = finger.bone(Bone.TYPE_PROXIMAL)
```

inputs:

- 0 = TYPE_METACARPAL – The metacarpal bone.
- 1 = TYPE_PROXIMAL – The proximal phalanx; the closest bone segment to the hand.
- 2 = TYPE_INTERMEDIATE – The intermediate, or middle, phalanx.
- 3 = TYPE_DISTAL – The distal phalanx; the bone segment furthest from the hand.

```
1
2 Joint_position = finger.joint_position(0)
3
```

inputs:

- 0 = JOINT_MCP – The metacarpophalangeal joint, or knuckle, of the finger.
- 1 = JOINT_PIP – The proximal interphalangeal joint of the finger. This joint is the middle joint of a finger.
- 2 = JOINT_DIP – The distal interphalangeal joint of the finger. This joint is closest to the tip.
- 3 = JOINT_TIP – The tip of the finger.

We can get this type of properties from pointable object,

- id - A unique ID assigned to this Pointable object.
- Frame - The Frame associated with this Pointable object.
- Hand - The Hand associated with this finger.
- tip_position - The tip position in millimeters from the Leap Motion origin.
- tip_velocity - The rate of change of the tip position in millimeters/second.
- direction - The direction in which this finger is pointing.

- width - The estimated width of the finger in millimeters.
- length - The estimated length of the finger in millimeters.
- is_finger - Whether or not the Pointable is classified as a finger.
- stabilized_tip_position - The stabilized tip position of this Pointable.
- time_visible - The duration of time this Pointable has been visible to the Leap Motion Controller.

Arm:

There is also an arm object which gives data related elbow_position, arm.wrist_position, width, direction, the basis of all three axises.
Arm object is generated from hand object.

```
hand = frame.hands.frontmost
arm = hand.arm
```

attributes:

- basis:
  Basis vectors specify the orientation of an arm.
  - x_basis. Perpendicular to the longitudinal axis of the arm; exits leterally through the sides of the wrist.
  - y_basis or up vector. Perpendicular to the longitudinal axis of the arm; exits the top and bottom of the arm. More positive in the upward direction.
  - z_basis. Aligned with the longitudinal axis of the arm. More positive toward the elbow.

```
basis = arm.basis
x_basis = basis.x_basis
y_basis = basis.y_basis
z_basis = basis.z_basis
```

```
5 center=arm.elbow_position+(arm.wrist_position-arm.
    elbow_position*.05
6 arm_transform = Leap.Matrix(x_basis, y_basis, z_basis,
    center)
```

- direction - The normalized direction of the arm from elbow to wrist.
- elbow_position - The position of the elbow.
- width - The average width of the arm.
- wrist_position - The position of the wrist.

Here we will use only some properties like,

- In hand,
  - direction
  - palm_position
  - stabilized_palm_position
- In pointable,
  - direction
  - tip_position
  - stabilized_tip_position

LeapTrainer.js:

This API is used gesture and pose learning and recognition framework for leap motion sensor. This is javascript based API, so using that framework we can detect gestures and poses in a web browser, and then using websocket it will go to python program. So here is the overview of this API, **Uses:**

```
<script src="http://js.leapmotion.com/0.2.0/leap.min.js"></
    script>
<script src="/path/to/leaptrainer.min.js"></script>
```

You need to include these two files for using this framework. The first file is for connection with leap sensor and second for this framework.

Then we need to create leap sensor's object and trainer object

```
var leapController = new Leap.Controller();
var trainer = new LeapTrainer.Controller({controller:
    leapController});
leapController.connect();
```

Training the system: We can train the system by adding new gesture or pose to system, and we can add new gesture or pose by calling this method,

```
trainer.create('Halt');
```

here in place of 'Halt' give any name related to your gesture or pose. And we can do something when this gesture is recognized like,

```
trainer.on('Halt', function() { console.log('Stop right there!')
    ; });
```

Importing and Exporting from LeapTrainer:

This feature is most useful in real application. Because normally wat happened whatever data we create like learning gestures or poses, are temporary means once you reload the script all objects will be reinitialized so all saved gestures or poses data will be lost. So by using this feature we can save that data. This framework exports data in the form of JSON and also accepts imports in the form of JSON.

```
var savedGesture = trainer.toJSON('Halt');
//Exporting to JSON

trainer.fromJSON(savedData);
//Importing from JSON
```

This can be used for saving and fetching gesture or pose data into or from file
Options: Here are some options for changing the setting of LeapTrainer,

- **controller:** An instance of Leap.Controller class from the Leap Motion Javascript API. This will be created with default settings if not passed as an option.
- **pauseOnWindowBlur:**If this variable is TRUE, then the LeapTrainer Controller will pause when the browser window loses focus, and resume when it regains focus (default: FALSE)
- **minRecordingVelocity:**The minimum velocity a frame needs to be measured at in order to trigger gesture recording. Frames with a velocity below this speed will cause gesture recording to stop. Frame velocity is measured as the fastest moving hand or fingertip in view (default: 300)
- **maxRecordingVelocity:**The maximum velocity a frame can measure at and still trigger pose recording, or above which to pose recording will be stopped (default: 30)
- **minGestureFrames:** The minimum number of frames that can contain a recognizable gesture (default: 5)
- **minPoseFrames:**The minimum number of frames that need to hit as recordable before pose recording is actually triggered. This higher this number, the longer a pose needs to be held in position before recognition will be attempted. (default: 75)

- **minPoseFrames:**The minimum number of frames that need to hit as recordable before pose recording is actually triggered. This higher this number, the longer a pose needs to be held in position before recognition will be attempted. (default: 75)
- **minPoseFrames:**The minimum number of frames that need to hit as recordable before pose recording is actually triggered. This higher this number, the longer a pose needs to be held in position before recognition will be attempted. (default: 75)
- **hitThreshold:**The return value of the recognition function above which a gesture is considered recognized. Raise this to make gesture recognition more strict (default: 0.7)
- **trainingCountdown:**The number of seconds after startTraining is called that training begins. This number of training-countdown events will be emitted. (default: 3)
- **trainingGestures:**The number of training gestures required to be performed in training mode (default: 1)
- **convolutionFactor:**The factor by which training samples will be convolved over a gaussian distribution in order to expand the input training data. Set this to zero to disable convolution (default: 0)
- **downtime:**The number of milliseconds after a gesture is identified before another gesture recording cycle can begin. This is useful, for example, to avoid a 'Swipe Left' gesture firing when a user moves his or her hand quickly back to center directly after performing a 'Swipe Right' (default: 1000)

Options can be passed to the LeapTrainer.Controller constructor like so:

```
1 new LeapTrainer.Controller({minRecordingVelocity: 100, downtime:
      100});
```

Events:

The LeapTrainer controller will emit events to listening components during training and recognition. The framework events are:

- **gesture-created:** Fired when a new gesture is added to a LeapTrainer controller object - either by a call to the create()function or by importing a saved gesture via the fromJSON() function. Carries two parameters: gestureName andtrainingSkipped. The latter will be true if this gesture was created by a call to the create() function in which the second parameter was true.
- **training-countdown:** Fired a configurable number of times, once per second, after the startTraining function is called, before the training started event fires and actual training begins
- **training-started:** Fired when training begins on a gesture - carries a single parameter, gestureName
- **training-complete:** Fired when training completes on a gesture - carries three parameters, gestureName,trainingGestures, and isPose. The second parameter is the array of encoded gestures recorded during training. The final parameter indicates whether a pose or a gesture has been learned.
- **training-gesture-saved:**: Fired during training when a new training gesture is recorded - carries two parameters,gestureName, and trainingGestures. The latter is the array of encoded gestures recorded so far during training, the last entry of which will be the gesture just recorded.
- **started-recording:**Fired when the recordableFrame function returns TRUE and causes gesture recordingFired when the recordableFrame function returns FALSE and recording was active, causing gesture recording to stop
- **gesture-detected:** Fired when any gesture is detected, regardless of whether it is recognized or not. This event fires before recognition are attempted and carries two parameters, gesture and frameCount. The first is the recorded encoded gesture, the second is the number of frames in the gesture.
- **gesture-recognized:** Fired when a known gesture is recognized - carries two parameters, hit and gestureName. The fit parameter is a value between 0.0 and 1.0 indicating how closely the recognized gesture was matched to the training data, with 1.0 being a 100% match.

- **gesture-unknown:** Fired when a gesture is detected that doesn't match any known gesture sufficiently to be considered a match. This event carries two parameters, bestHit and closestGestureName, identifying the closest known gesture.

These events can be registered as this way,

```
trainer.on('training-started', function(movementName) {
    console.log('Started training ' + movementName);
});
```

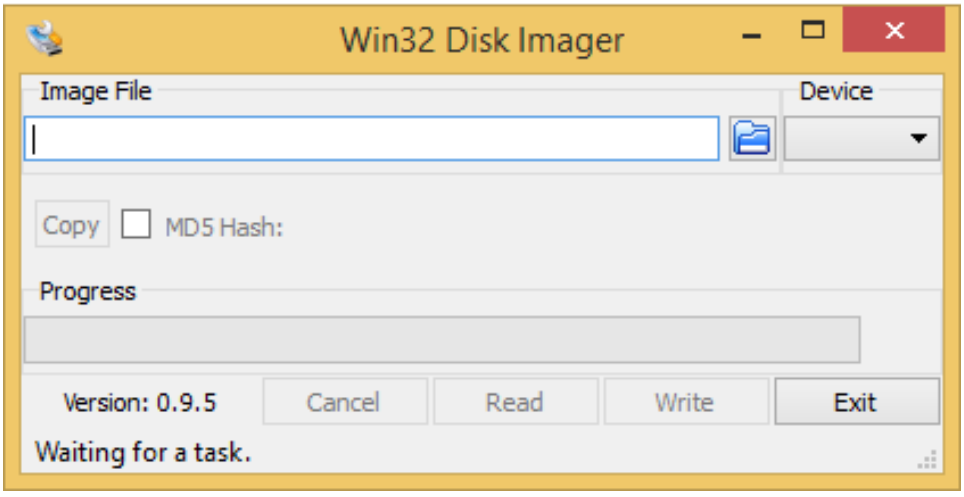
Installation of Linux image in Galileo board:

In Galileo board, we can install Linux, windows IOT or os x. So the interface with board become easy. Here we will use Linux image. For that, we need to write Linux image on the sd card. From sd card, Galileo board will boot.

So How to prepare sd card???

Step 1:
Download Linux image from this location.
`https://downloadmirror.intel.com/26028/eng/iot-devkit-prof-dev-image-galileo-201605`
`zip`
You will get one zip file extract it. You will find one '.direct'. You need to write this into sd card

Step 2:
Download this software from this location,
http://download.softpedia.com/dl/d5a643aad00816ee735372c2d530a1a2/57584b56/100173006/softw
0.9.5 – *binary.zip*
thiswillbehelpfulforwritinganimageinsdcard.



SelectImageFilewhichyouhavedownloadedearlierin'ImageFile'tab('.direct'file)
Selectsdcardin'Devicetab'

Nowclickon'Write'button,itwilltakesometime.Andwaituntilwriteprocesscompletes

Step 3:

NowinsertsdcardingalileoboardandconnectLANcablewithyourPCandgalileoboard.Nowconne

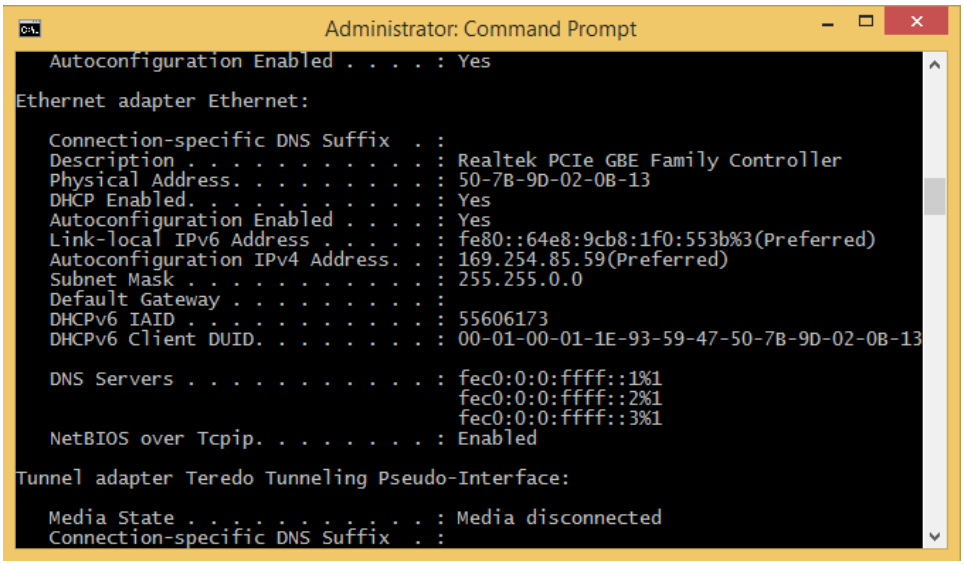
now the board will automatically boot. There will be blinking LED near sd card port. Wait until it stops lighting. Because it shows IO related sd card. So, when it become stop, IO related sd card is also stopped so the booting process is completed. Now you need to find the IP address of your board. For that, you can use any network packet tracer software like Wireshark, ettercap or any other.

I am using Wireshark.

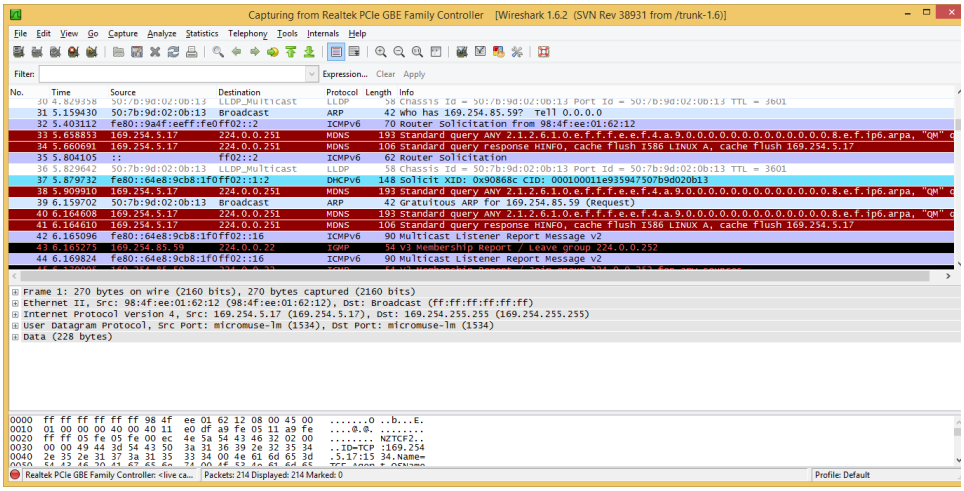
Now open Wireshark select ethernet as a network adapter. It will start tracing all packet data on ethernet. But here on an ethernet network, there is only two devices are there your PC and board. So there is only two IP address data. (ignore IPs with 255, 251, 224 and more than 200 value in last two part like 192.168.0.255 or 255.255.255.0) so find your PC IP address using this command

```
ipconfig (for windows)}
ifconfig (for Linux)}
```

Now another IP showing on Wireshark is your board IP.

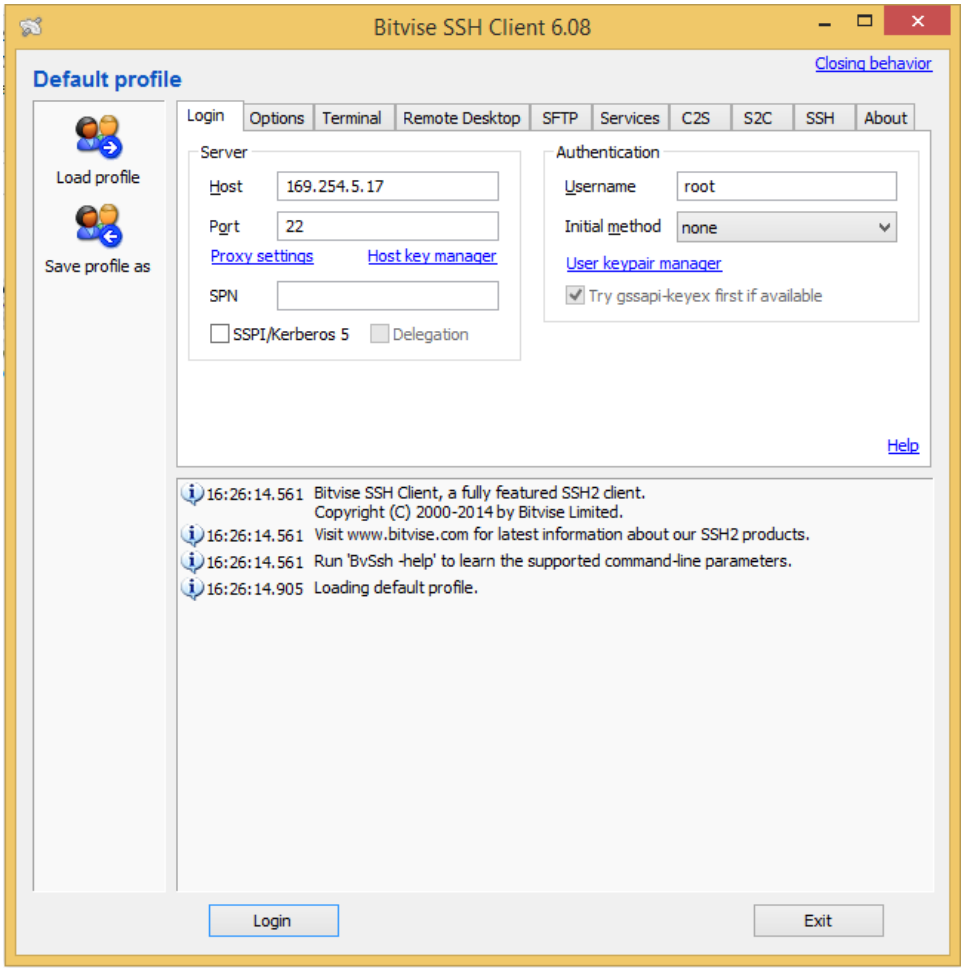


Your Wireshark windows:

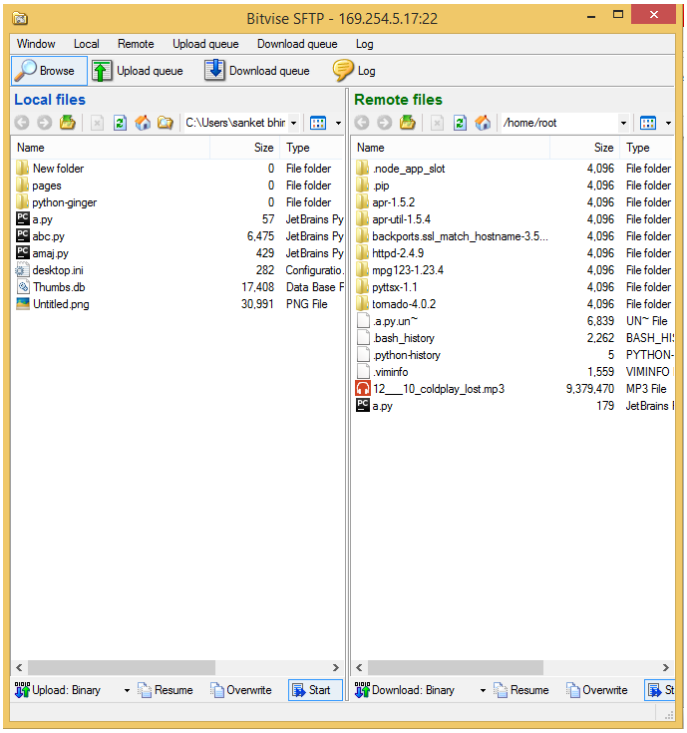

Capturing from Realtek PCIe GBE Family Controller [Wireshark 1.6.2 (SVN Rev 38931 from /trunk-1.6)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: Expression... Clear Apply
No. Time Source Destination Protocol Length Info
31 5.159430 50:7b:9d:02:0b:13 Broadcast ARP 42 Who has 169.254.85.59? Tell 0.0.0.0
32 5.403112 fe80::9a4f:eeff:fe0 ff02::2 ICMPv6 70 Router Solicitation from 98:4f:ee:01:62:12
33 5.658853 169.254.5.17 224.0.0.251 MDNS 193 Standard query ANY 2.1.2.6.1.0.e.f.f.f.e.e.f.4.a.9.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f.ip6.arpa, "QM"
34 5.660691 169.254.5.17 224.0.0.251 MDNS 106 Standard query response HINFO, cache flush I586 LINUX A, cache flush 169.254.5.17
35 5.804105 :: ff02::2 ICMPv6 62 Router Solicitation
36 5.829642 50:7b:9d:02:0b:13 LLDP_Multicast LLDP 58 Chassis Id = 50:7b:9d:02:0b:13 Port Id = 50:7b:9d:02:0b:13 TTL = 3601
37 5.879732 fe80::64e8:9cb8:1f0 ff02::1:2 DHCPv6 148 Solicit XID: 0x90868c CID: 00010001e93594750 7b9d020b13
38 5.909910 169.254.5.17 224.0.0.251 MDNS 193 Standard query ANY 2.1.2.6.1.0.e.f.f.f.e.e.f.4.a.9.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f.ip6.arpa, "QM"
39 6.159702 50:7b:9d:02:0b:13 Broadcast ARP 42 Gratuitous ARP for 169.254.85.59 (Request)
40 6.164608 169.254.5.17 224.0.0.251 MDNS 193 Standard query ANY 2.1.2.6.1.0.e.f.f.f.e.e.f.4.a.9.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f.ip6.arpa, "QM"
41 6.164610 169.254.5.17 224.0.0.251 MDNS 106 Standard query response HINFO, cache flush I586 LINUX A, cache flush 169.254.5.17
42 6.165096 fe80::64e8:9cb8:1f0 ff02::16 ICMPv6 90 Multicast Listener Report Message v2
43 6.165275 169.254.85.59 224.0.0.22 IGMP 54 V3 Membership Report / Leave group 224.0.0.252
44 6.169824 fe80::64e8:9cb8:1f0 ff02::16 ICMPv6 90 Multicast Listener Report Message v2
Frame 1: 270 bytes on wire (2160 bits), 270 bytes captured (2160 bits)
Ethernet II, Src: 98:4f:ee:01:62:12 (98:4f:ee:01:62:12), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Internet Protocol Version 4, Src: 169.254.5.17 (169.254.5.17), Dst: 169.254.255.255 (169.254.255.255)
User Datagram Protocol, Src Port: micromuse-lm (1534), Dst Port: micromuse-lm (1534)
Data (228 bytes)
0000 ff ff ff ff ff ff 98 4f ee 01 62 12 08 00 45 00 ....O ..b...E.
0010 01 00 00 00 40 00 40 11 e0 df a9 fe 05 11 a9 fe ....@.@. ....
0020 ff ff 05 fe 05 fe 00 ec 4e 5a 54 43 46 32 02 00 .... NZTCF2..
0030 00 00 49 44 3d 54 43 50 3a 31 36 39 2e 32 35 34 ..ID=TCP :169.254
0040 2e 35 2e 31 37 3a 31 35 33 34 00 4e 61 6d 65 3d .5.17:15 34.Name=
Realtek PCIe GBE Family Controller: <live ca...
Packets: 214 Displayed: 214 Marked: 0
Profile: Default

Step 3:

Now, from any SSH client make connection with board and use. :?)
(I'm using Bitvise SSH client)
Enter IP address and username: root there is no password.



Click on login, and wait. Now two windows will open one for terminal and another for file transfer. Your board is ready! :?)


Bitvise SFTP - 169.254.5.17:22
Window Local Remote Upload queue Download queue Log
Browse Upload queue Download queue Log
Local files
C:\Users\sanket bhir
Name Size Type
New folder 0 File folder
pages 0 File folder
python-ginger 0 File folder
a.py 57 JetBrains Py
abc.py 6,475 JetBrains Py
amaj.py 429 JetBrains Py
desktop.ini 282 Configuratio
Thumbs.db 17,408 Data Base F
Untitled.png 30,991 PNG File
Remote files
/home/root
Name Size Type
.node_app_slot 4,096 File folder
.pip 4,096 File folder
apr-1.5.2 4,096 File folder
apr-util-1.5.4 4,096 File folder
backports.ssl_match_hostname-3.5... 4,096 File folder
httpd-2.4.9 4,096 File folder
mpg123-1.23.4 4,096 File folder
pyttsx-1.1 4,096 File folder
tornado-4.0.2 4,096 File folder
a.py.un~ 6,839 UN~ File
.bash_history 2,262 BASH_HI
.python-history 5 PYTHON-
.viminfo 1,559 VIMINFO
12___10_coldplay_lost.mp3 9,379,470 MP3 File
a.py 179 JetBrains
Upload: Binary Resume Overwrite Start
Download: Binary Resume Overwrite


Bitvise xterm - 169.254.5.17:22
root@galileo:~#

Interfacing MP3 module:

MP3 module is used to play audio files stored in separate memory card attached with module. Control of this module is done the hex code sent from any controller via UART port. Here we will send code from Galileo board from Rx Tx pin to the board, there are some specified method to generating hex code which you can find on it's data sheet. We will se about that in next topic

5 Implementation:

- Alphabet training and recognizing with leap motion
- Make natural sentence from words using NLTK
- Make simple player for MP3 module
- Other Experiment for gesture recognizing.
- First version of this system with LeapTrainer.js
- second and final version of this system

Alphabet training and recognizing with leap motion:

Here we want to make a programme which records or say learns pose hand and then it will recognize. Means there will be two mode learning and recognizing. In Learning mode we need to perform pose so this pose will be saved for future reference and need to provide one alphabet for that pose. and In recognizing mode we need to perform specific pose so programme will recognize that pose and give related alphabet

Here we have used pointables x direction, pointable z direction, pointable x stabilized tip position, pointable z stabilized tip position, stabilized palm position of x axis and z axis.

Here we want to implement a programme which is independent of hand's position means we can perform action at any position in range of sensor. so we have take data relative palm center so we have used stabilized palm position but here only x and z axis is useful because we don't need to care about height from sensor.

Here we will find radius distance from palm center to every tip of finger so we need to find mean position of tip's x and z co-ordinates related to palm center

We are also taking direction vectored so we can increase range of detection of radius distance

Then sum interface related programming. whole programme is in python language you can find at github `https://github.com/sanketbhimani/`

eYSIP-2016-Sign-Language-Interpreter-Leap-Motion-Sensor-/blob/master/alphabet_using_leap.py

Silly mistake I made and also solve!

While copping object I was directly write like a = b. so programme was not working properly and only large performed pose goes to every object. I was not understanding what was happening. what was happening, when I write a = b it just transfer a pointer no new memory is allocated so when I change value for one object all values will change because they all are pointing to same memory location. then i used a = copy.deepcopy(b) then it will allocate new memory for that object. and it is working fine!!!

Make natural sentence from words using NLTK:

In sign language, normally whole sentence is not actioned mean little grammar thing like "is" are not much important. because action just want to show the understanding about what he want to tell. they not try to build a complete sentence with correct grammar.

So, this programme will help to create a proper natural sentence with correct grammar. like if you do input like "WHAT NAME YOU", means you want to ask a name someone this should be like "WHAT IS YOU NAME?" so this type of conversation is done by this programme

For that I have use NLTK library, NLTK have lot of data to compare and detect what is this word means this is verb, pronoun, noun etc. Let take one example, "make a note" here note is noun but "note it down" here note is verb so for different cases same word can perform different role, so NLTK helps you to detect the role of words, it also helps you to organize sentence and word at smart way.

After detecting the role of this words, it's our job to make a proper sentence. This programme will put them in proper sequence and add necessary words like "is". then it will correct this sentence grammar. like if the input is like "WHAT NAME YOU" our programme generate new sentence like "WHAT IS YOU NAME" then after correcting grammar this will be like "WHAT IS YOUR NAME?" For correcting grammar I have use online API, it's ginger grammar. So you need to just send raw sentence and as a res ponce you will get proper correct sentence with some extra information as a JSON data you need to fetch correct sentence. So you need to learn basic NLTK library and sending and receiving HTTP request and response using python.

Make simple player for MP3 module:

This is a simple programme designed to test and get learn an interfacing with MP3 module. This programme is basically mp3 player which have option of next, previous, play/pause, volume up and volume down.

This programme is in command line interface. And it generates appropriate hex code for perform certain task and send it to mp3 module through UART (Rx and Tx pin available on board). This includes of generating oh check bits and all.

Other Experiment for Gesture recognizing:

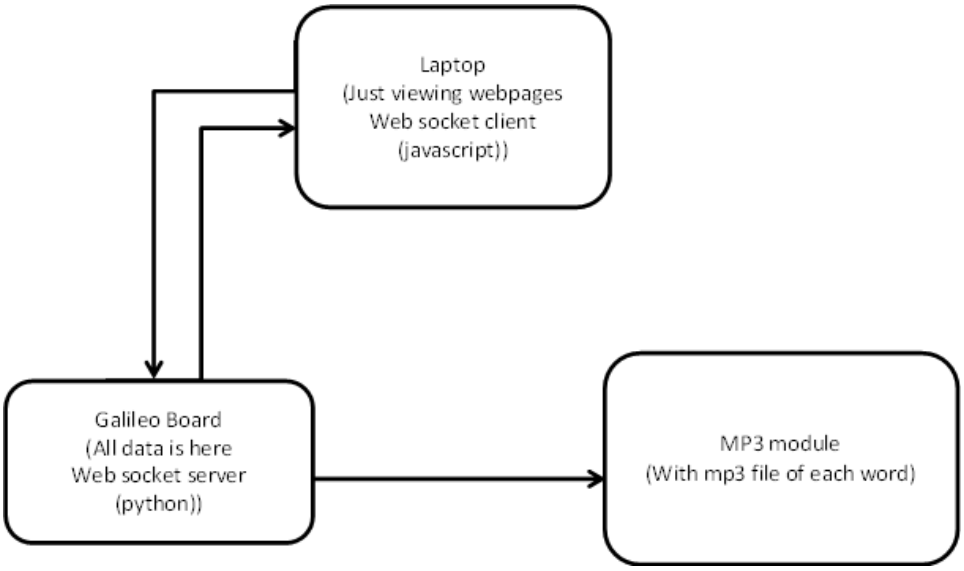
Now, we need to do gesture recognizing. means in first alphabet recognising programme we have just recognize just pose but now we need to recognize whole motion. For that i was searching for works done on this topic. From leap developer community I found two useful repository first one is $1 unistock gesture recognizing library. If know about $1 and $n algorithm of university of Sweden. This algorithm compares two pattern and gives percentages matches. So what i found that some one have made like using only one pointable we need to perform action and system will recognize that action. but here we can not use standard sign language because here only liner actions can be performed. And it with too much errors. Means it does mistakes so much while recognizing.

So, finally i decided to not to use this system. So I search another library named LeapTrainer.js. It was good and i start working on it.

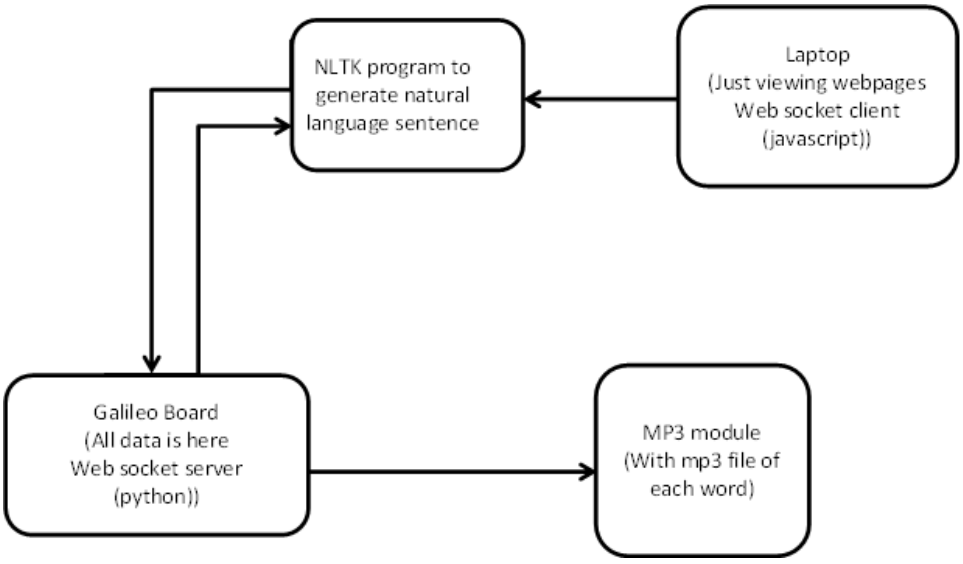
First version of this system with LeapTrainer.js:

This is based on web pages meas whole system was web based. it runs as website. So gesture training and recognising happens on browser. Means, here we need to train the system first (Loading of gesture data), Then system will recognize that gestures. But problem was that after loading data when we reload the page or reopen the page saved data ware destroyed. meas the data are not being permanently stored so we need to make some system for permanent saving of data.

As explined in theory there is a listener function for each events. So, when gesture-created event generated i store data of gestures in form of JSON using toJSON method. and in starting (means when we open the page) I load all gestures from JSON file. But here we want to make our system fully portable so JSON file is on Galileo board so the transfer of data is through Websocket. On board there is one python programme running which will create websocket server and our webpage be a websocket client. And all website data is also on board. we are just opening site from board to our laptop's browser. Here we need to connect Leap Motion Sensor on laptop side.



But here we also use that NLTK programme to convert sentence in natural language with correct grammar so we need to first pass the output of this web application to this programme. Now, this programme requires more processing power so it can not be run on Galileo board so we need to run it on laptop so we will pass the output to python script running on y laptop and then it sent to Galileo board. here also communication will be done through websocket.



There is also some problem like they made mistake many time to recognising gesture and its interface is also quite annoying. So we need to think something new.

second and final version of this system:

Now, I have tried everything available on Internet related gesture recognising but nothing was efficient. So, I have decided to make my own programme to recognising gestures. And worked! So I've create my own algorithm to recognising gestures. It's very similar to the programme of alphabet recognition.If you remember in that programme we are taking average of 300 frames and we are using that average value for future comparison so what was happening that small movement while recording frames are gone by taking average but now in this case we want to capture the movements so we can not average. But we can directly compare that much frame to that much frame. So if sufficient frame will match, then we can say that, this is that gesture.

whole programme interface is command-line. There will be two mode of programme, Train and Recognize. So, in train mode we need to perform gesture so system will remember as a future reference. And in recognize mode we need to perform this gesture and system will recognize that gesture and give word matched with it.

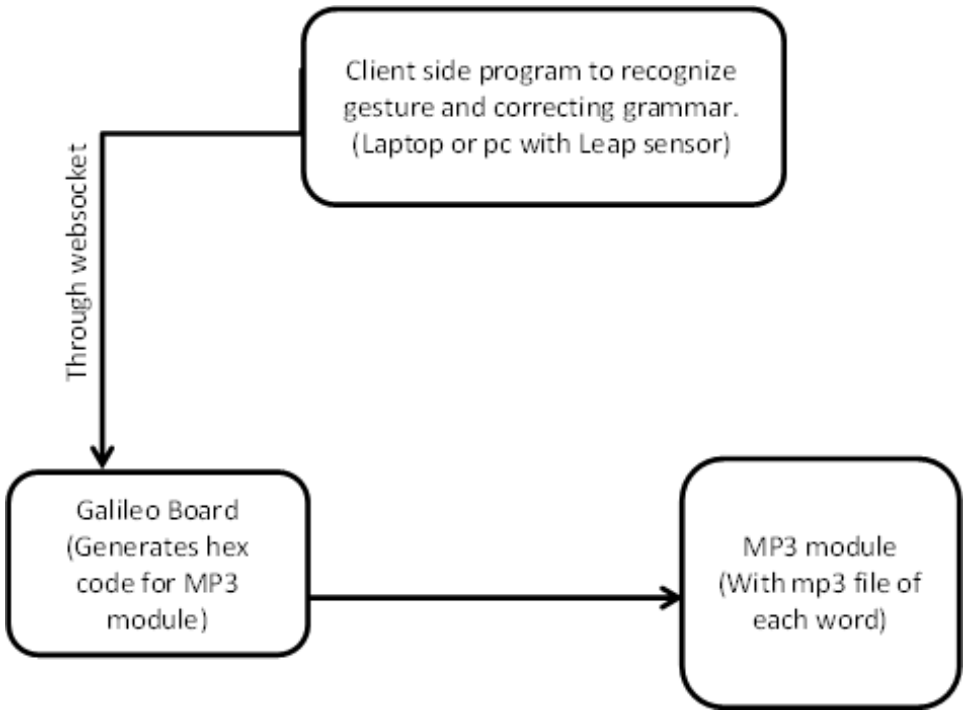
Here, in each mode there is a timer of 3 second so you can set a proper position and check whether your each hand is detected properly or not. So, capturing of frame will start after 3 second. For changing of options and all thing see programme on github. There is well commented code so you can change every option like change the timer second or say change no of frame captures.

Here, NLTK programme also merged with main programme so output is directly sent to board and from board to MP3 module, Here all gesture's data will be in laptop. And the communication with board and laptop will be through websocket.

Here whole system is in python. there is two programme one for server and one for client.

- Task performed by client side programme:
  - Recognizing and Training of gesture.
  - Storing and Loading data to and from JSON file stored on same laptop/pc.
  - Add necessary words and Correcting the position of words.
  - Correcting the grammar.
  - Sending sentence to board through websocket.

- Task performed by server side programme:
  - Receiving sentence sent from client side programme.
  - Davide it into each word and find file name for that word in MP3 module
  - And play appropriate file stored in mp3 module through sending proper hex code.



6 Exercise

Finding Error rate for Alphabet recognising programme

Here is the result of experiments:

- "Yes" means recognise
- "No" means not recognise

A		yes	yes	yes	yes	yes	yes	yes	yes
B		yes	yes	yes	yes	yes	yes	yes	yes
C		yes	no	no	yes	yes	yes	yes	yes
D		no	yes	yes	yes	yes	yes	yes	yes
E		yes	yes	yes	yes	yes	yes	yes	yes
F		no	yes	yes	yes	no	yes	yes	yes
G		no	no	yes	yes	yes	yes	yes	yes
H		yes	yes	yes	yes	yes	yes	yes	yes
I		yes	yes	yes	yes	yes	yes	yes	yes
J		no	yes	yes	yes	yes	yes	yes	no
K		yes	yes	yes	yes	no	yes	yes	yes
L		yes	yes	yes	yes	yes	yes	yes	yes
M		yes	yes	yes	yes	yes	yes	yes	yes
N		no	yes	yes	no	no	yes	yes	yes
O		yes	yes	no	yes	yes	yes	yes	yes
P		yes	no	yes	yes	yes	yes	yes	yes
Q		yes	yes	yes	yes	yes	yes	no	yes
R		no	yes	no	yes	no	no	no	yes
S		yes	yes	yes	yes	yes	yes	yes	yes
T		yes	yes	yes	yes	yes	yes	yes	yes
U		yes	no	yes	yes	yes	yes	yes	yes
V		yes	yes	yes	yes	yes	yes	yes	yes
W		yes	yes	yes	yes	yes	yes	yes	yes
X		yes	yes	yes	yes	yes	yes	yes	yes
Z		no	yes	yes	yes	yes	yes	yes	yes

total: 26*8 = 208
Incorrect recognition: 23
So rate is: 23/208 = 0.11057 :)

7 References

- For learning python: www.tutorialspoint.com/python/
- For Various doubt solution: www.stackoverflow.com/questions/tagged/python/
- For understanding leap library in python: www.developer.leapmotion.com/documentation/pyth
- For javascript gesture recognizing and learning system: www.github.com/roboleary/LeapTrainer
- For $1 algorithm with Leap motion sensor: https://github.com/liangzan/leap-demo
- http://depts.washington.edu/aimgroup/proj/dollar
- www.developer.leapmotion.com/libraries
- www.stackoverflow.com/questions/tagged/javascript
- www.stackoverflow.com/questions/tagged/jquery
- www.t4t5.github.io/sweetalert
- www.github.com/t4t5/sweetalert
- www.pypi.python.org/pypi/nltk
- www.nltk.org
- www.youtube.com/watch?v=FLZvOKSCkxY
- www.gingersoftware.com/grammarcheck
- www.github.com/zoncoen/python-ginger
- www.blog.livedoor.jp/xaicron/archives/54466736.html
- www.youtube.com/watch?v=gN3CGhFPF1s
- www.github.com/tornadoweb/tornado
- www.pypi.python.org/pypi/tornado
- www.gist.github.com/billroy/3761495
- www.stackoverflow.com/questions/2835559/parsing-values-from-a-json-file-in-python/
- www.stackoverflow.com/questions/29430333/simultaneously-reading-and-writing-to-json-file-in-python

- www.downloadmirror.intel.com/26028/eng/iot-devkit-prof-dev-image-galileo-20160525.zip
- www.software.intel.com/en-us/iot/hardware/galileo/downloads
- www.downloadmirror.intel.com/25384/eng/w_galileo_2015.0.010.exe
- www.youtube.com/watch?v=yrRMomesBKM
- https://github.com/intel-iot-devkit/mraa