ELECTRONIC WORKSHOP-II
FINAL REPORT

"SOUND SOURCE LOCALIZATION BASED CAMERA MOVEMENT"

PROJECT GUIDED BY :
DR. JAYANTHI SIVASWAMY
DR. KAVITA VEMURI

PROJECT SUBMITTED BY :
MANAN BHANDARI

AIM & DESCRIPTION :

- To design a voice source localization based camera movement control system in auditoriums/stage.

- Main part is the sound source localisation which needs to be done in the most optimal way and within reasonable accuracy.

- Depending upon the feedback through the microphone array the degree of rotation (0-360 degree) of the camera i.e. its angle of orientation is controlled.

APPLICATIONS :

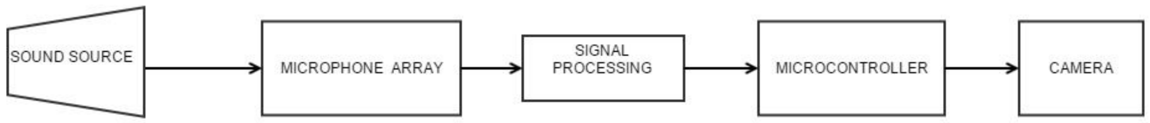
- This model can be installed in big classes of schools and colleges where the distance between the speaker and the last bench students is very much. By installing that, each member in the classroom will be able to see the speaker. Until now it is mostly being done manually. This is an attempt to automate it.

- This model can also be installed in the robot so that it can hear the voice of the instructor and move in that direction. Many other usages can also be found in the robot system for this model according to the code embedded in it.

- Can also be used in video surveillance and can be an important part in detecting crimes if already installed at the crime place.

WORKING PRINCIPLE :

The sound wave from the speaker will be detected by the microphones used in the circuit. These waves are analysed by us on the computer using softwares. Using the proper algorithm to know which microphone is closest to the speaker, we turn the servomotor, hence the camera in that direction.



GENERAL ALGORITHMS USED (for similar projects):

Reference papers:

- N-dimensional N-microphone sound source localization - Ali Pourmohammad and Seyed Mohammad Ahadi
- A Comparison of Algorithms of Sound Source Localization Based on Time Delay Estimation- Hasan Khaddour

---

<u>Some of the algorithms we studied:</u>

- **HRTF (Head related transfer function):**

Human beings are able to locate sound sources in three dimensions in range and direction, using only two ears. The location of a source is estimated by comparing difference cues received at both ears, among which are time differences of arrival and intensity differences. The sound source specifications are modified before entering the ear canal.

The head-related impulse response (HRIR) is the name given to the impulse response relating the source location and the ear location. Therefore, convolution of an arbitrary sound source with the HRIR leads to what would have been received at the ear canal.

The HRTF is the Fourier transform of HRIR and thus represents the filtering properties due to diffractions and reflections at the head, pinna, and torso. Hence, HRTF can be computed through comparing the original and modified signals.

Consider X c (k) being the Fourier transform of sound source signal at one ear (modified signal at either left or right ear) and Y c (k) to be that of the original signal. HRTF of that source can be found as

$$H_c(k) = \frac{Y_c(k)}{X_c(k)}$$

In detail:

$$|H_c(k)| = \frac{|Y_c(k)|}{|X_c(k)|}$$

$$\arg H_c(k) = \arg Y_c(k) - \arg X_c(k)$$

$$H_c(k) = |H_c(k)| e^{j \arg H_c(k)}.$$

- **ILD (Interaural level difference)**

This is used for finding the coordinates so that we can localise the sound source. We consider two microphones for localizing a sound source. Signal s(t) propagates through a generic free space with noise and no (or low degree of) reverberation.

According to the so-called inverse-square law, the signal received by the two microphones can be modeled as:

$$s_1(t) = \frac{s(t-T_1)}{d_1} + n_1(t)$$

and

$$s_2(t) = \frac{s(t-T_2)}{d_2} + n_2(t),$$

The relative time shift between the signals is important for TDE but can be ignored in ILD. The energy received by the two microphones can be obtained by integrating the square of the signal over this time interval.

$$E_1 = \int_o^w s_1^2(t)\,dt = \frac{1}{d_1^2}\int_o^w s^2(t)\,dt + \int_o^w n_1^2(t)\,dt$$

$$E_1 = \int_o^w s_2^2(t)\,dt = \frac{1}{d_2^2}\int_o^w s^2(t)\,dt + \int_o^w n_2^2(t)\,dt.$$

The received energy decreases in relation to the inverse of the square of the distance to the source.

$$E_1 \cdot d_1^2 = E_2 \cdot d_2^2 + \eta,$$

$$d_1 = \sqrt{(x_1-x_s)^2 + (y_1-y_s)^2}$$

$$d_2 = \sqrt{(x_2-x_s)^2 + (y_2-y_s)^2}.$$

- **TDE / TDOA (Time difference estimation):**

Correlation-based methods are the most widely used time delay estimation approaches. The autocorrelation function of s 1 (t) can be writ-
ten in time domain as:

$$R_{s_1 s_1}(\tau) = \int_{-\infty}^{+\infty} s_1(t) \cdot s_1(t-\tau)\, dt.$$

in frequency domain:

$$R_{s_1 s_1}(\tau) = \int_{-\infty}^{+\infty} S_1(f) \cdot S_2^{*}(f) e^{j2\pi f\tau} df.$$

If the time delay τ is zero, this function's value will be maximized and will be equal to the energy of s1(t).

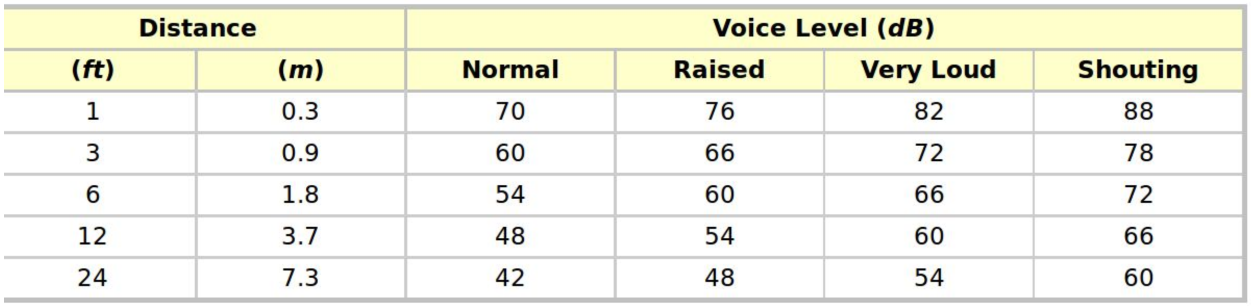
If s2(t) is considered to be the delayed version of s1(t), this function features a peak at the point equal to the time delay. This delay can be expressed as:

$$\tau_{12} = \operatorname{argmax}_{\tau} R_{s_1 s_2}(\tau).$$

OUR APPROACH TOWARDS THE PROJECT :

SELECTION OF MICROPHONES:

The normal voice of humans is 70dB. For every doubling of the distance from the noise source the sound pressure level is reduced with 6 decibels.
Higher sensitivity can sense more accurately from a greater distance.
Higher the voltage produced at a given pressure level more is the sensitivity.



Distance		Voice Level (*dB*)			
(*ft*)	(*m*)	Normal	Raised	Very Loud	Shouting
1	0.3	70	76	82	88
3	0.9	60	66	72	78
6	1.8	54	60	66	72
12	3.7	48	54	60	66
24	7.3	42	48	54	60

Different sensitivities of microphones:

i) Dynamic (59 dB)

ii) Condenser (55 dB)
iii) Electret (44 dB)
iiv) MEMS (23 to 26 dB)

We have used condenser Microphone as of now for the scaled down model. But for the implementation on a large scale as in big classrooms like H105, it is best to use MEMS.

For a basic version we have used a unidirectional mike. Due to time constraints we were not able to work with omnidirectional mikes.

CIRCUIT:

The circuit basically consists of three sub-circuits : amplifier, dc blocking and filtering.

Amplifier circuit:

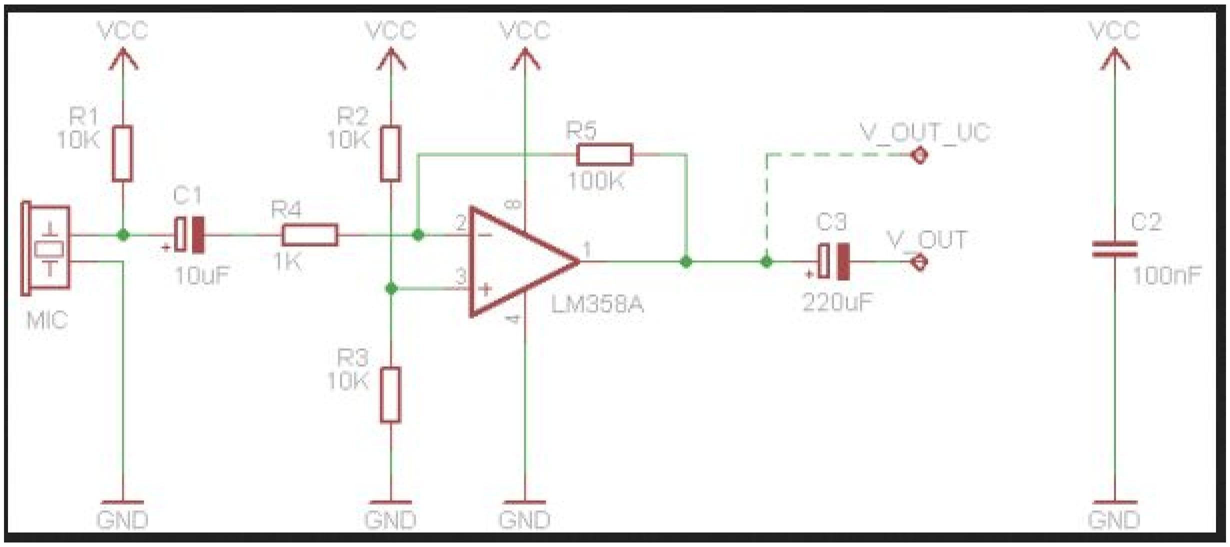
Amplifies sufficiently the audio signal which is received by the microphone. The earlier signal was in millivolts and not significant enough to be observed.

DC blocking:

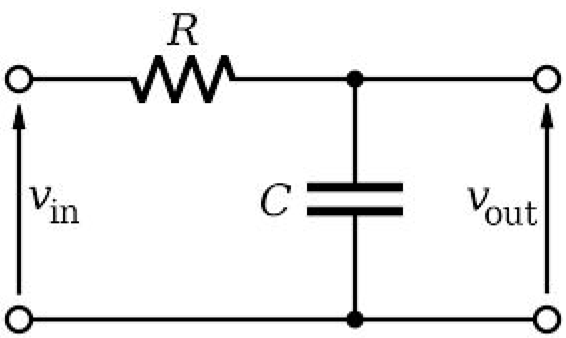
We observed that in real-time analysis, the DC component of the signal continuously dropped and therefore the threshold voltages for each mike would change every time. Therefore, the DC Blocking circuit is implemented after the amplifier circuit. Simply adding capacitor in series solved this problem.

Low pass filter:

We also observed that the audio signal obtained had too much noise with the actual signal. Low pass filter is implemented to remove the high frequency noise from the signal to get a high frequency output.



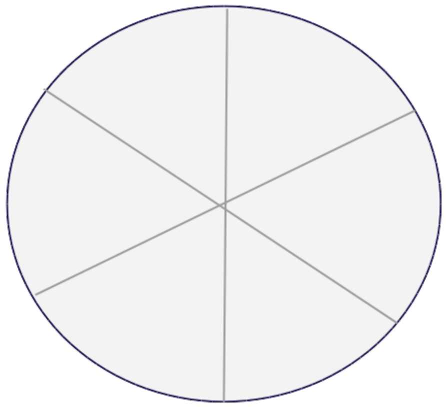
Amplifier Circuit



Simple low pass filter circuit

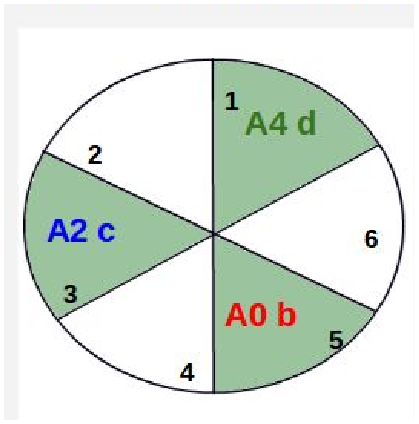
MICROPHONE ARRAY ARRANGEMENT SELECTION :

Our target was to make a 360 degree localisation model. (a basic model)
Linear wouldn't have worked that well. As we needed symmetry in all the directions. So a polygonal arrangement would be best. Our camera's field of view was found out to be 30 degrees. So we settled on a hexagonal array.



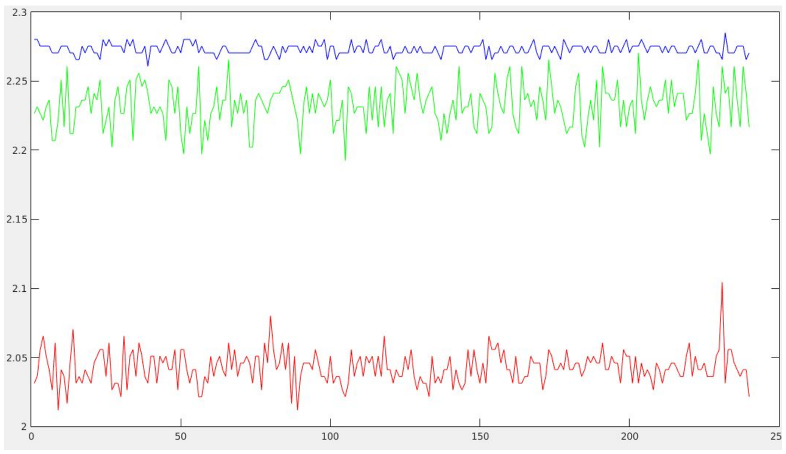
SIGNAL PROCESSING:

- Arduino support for matlab is being used for the signal processing part.
- Real time plotting of signals tested and compared between 3 mikes.
- Energy detected by each of the three mikes has been calculated and this part used to determine the sector.
  As because of the microphones constraints, only three of the six mikes were working. Therefore, we took alternate sectors for the approach. We have marked each sector by A0 (red : region 5), A2 (blue : region 3), A4 (green : region 1) .

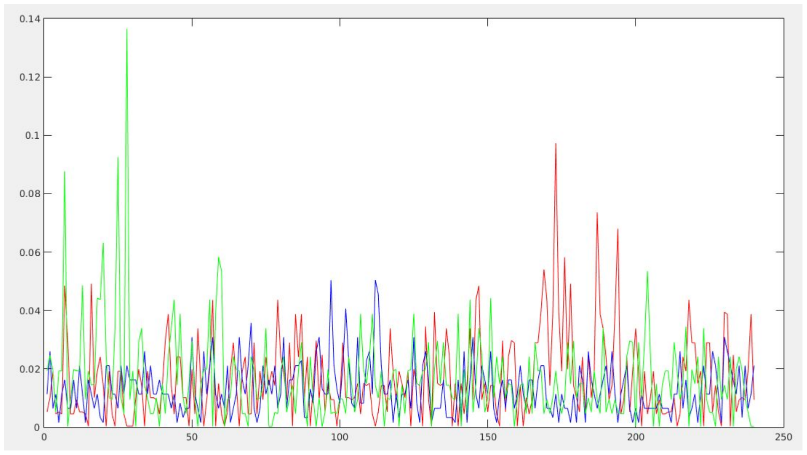


Following steps were followed:

1) The baseline for all the microphones was different. So it needed to be brought at the same level. And also for comparison, only the magnitude is required.



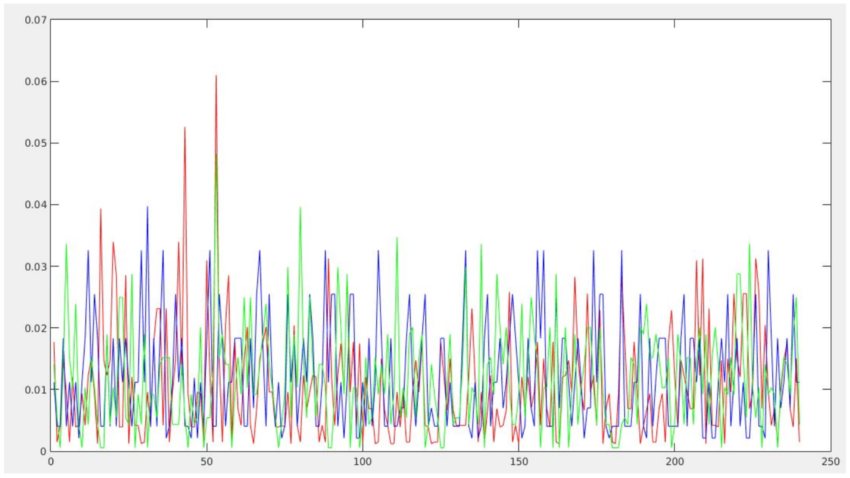
After correction:



Code for base correction:

```
clear b;
clear c;
clear d;
baseb=0;
basec=0;
based=0;

 scaleb=1;
 scalec=1;
 scaled=1;
 avgneb=0;
 avgnec=0;
 avgned=0;
% b red
% c blue
% d green
for i=1:120
b(i)=(readVoltage(a, 'A0'));
c(i)=(readVoltage(a, 'A2'));
d(i)=(readVoltage(a, 'A4'));
if(b(i)<0)
        b(i)=-b(i);
end
if(c(i)<0)
        c(i)=-c(i);
end
        if(d(i)<0)
        d(i)=-d(i);
        end
plot([1:i], b(1:i), 'r', [1:i], c(1:i), 'b',[1:i],d(1:i),'g');
drawnow
end
baseb=mean(b);
basec=mean(c);
based=mean(d);
```

2) Scaling of the microphones because of different sensitivities :



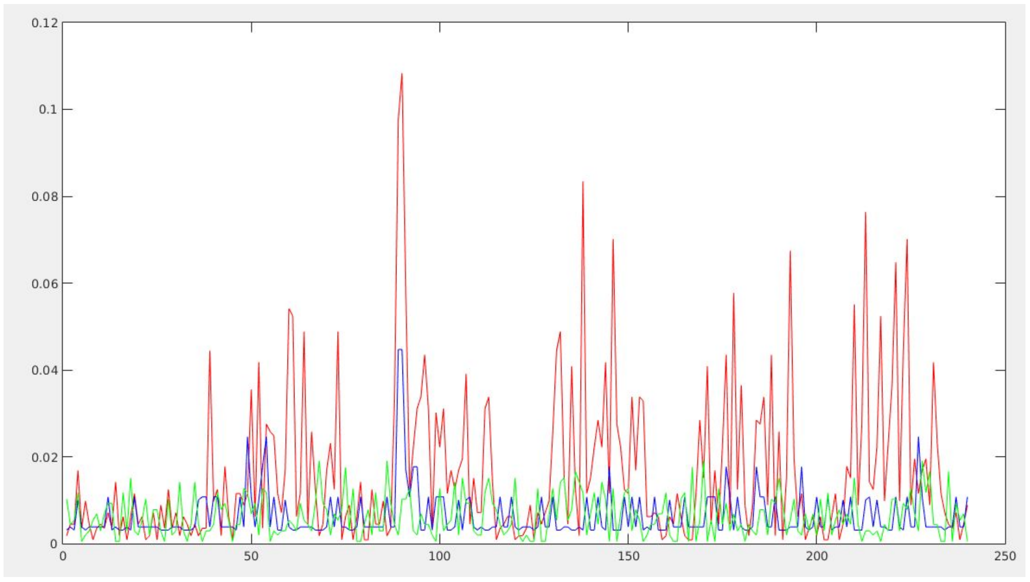
3) Finding the average noise energy and then removing it. The noise sensitivity of the mikes was different. Since we had to do a further 60 degree localisation, it was important to remove it. Average of 6 samples was computed after appropriate scaling.

Code for noise removal and scaling:

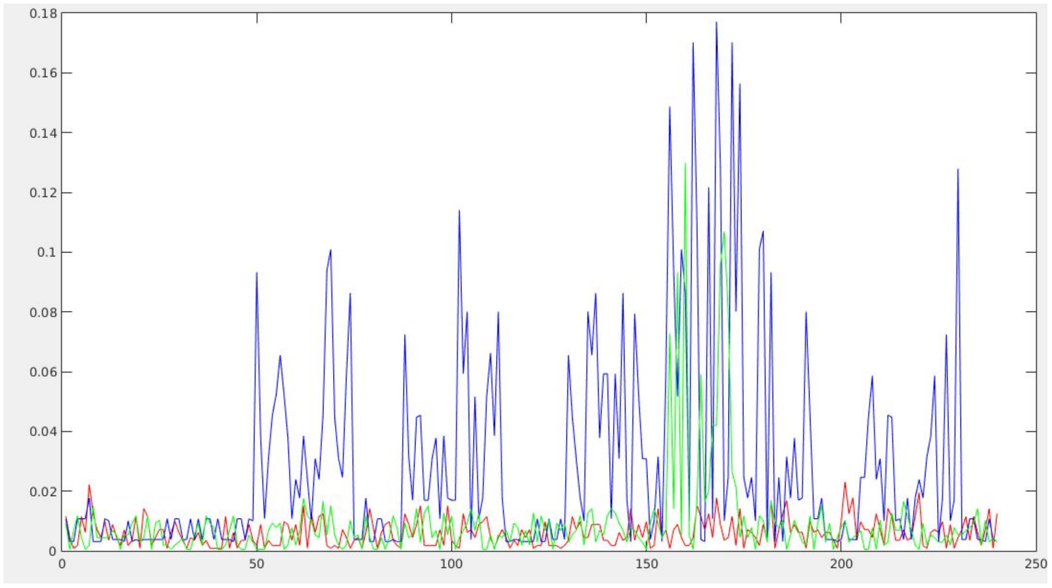
```
clear b;
clear c;
clear d;
Bn=zeros(1,6);
Cn=zeros(1,6);
Dn=zeros(1,6);
% scaleb=1.833;
% scalec=1.42;
% scaled=2;
 scaleb=1.7;
 scalec=1.4622;
 scaled=1.5;
% avgneb=0.0195;
% avgnec=0.0099;
% avgned=0.01155;
 avgneb=0;
 avgnec=0;
 avgned=0;
 j=1;
% b red
% c blue
% d green
```

```
for i=1:120
b(i)=(readVoltage(a, 'A0'))-baseb;
c(i)=(readVoltage(a, 'A2'))-basec;
d(i)=(readVoltage(a, 'A4'))-based;
if(b(i)<0)
        b(i)=-b(i);
end
b(i)=b(i)/scaleb;
if(c(i)<0)
        c(i)=-c(i);
c(i)=c(i)*scalec;
end
        if(d(i)<0)
        d(i)=-d(i);
        end
  d(i)=d(i)/scaled;
Bn(j)=Bn(j)+(b(i)*b(i));
Cn(j)=Cn(j)+(c(i)*c(i));
Dn(j)=Dn(j)+(d(i)*d(i));
plot([1:i], b(1:i), 'r', [1:i], c(1:i), 'b',[1:i],d(1:i),'g');
drawnow
x=mod(i,40);
if x==0
 j=j+1;
end
end
avgneb=mean(Bn);
avgnec=mean(Cn);
avgned=mean(Dn);
```

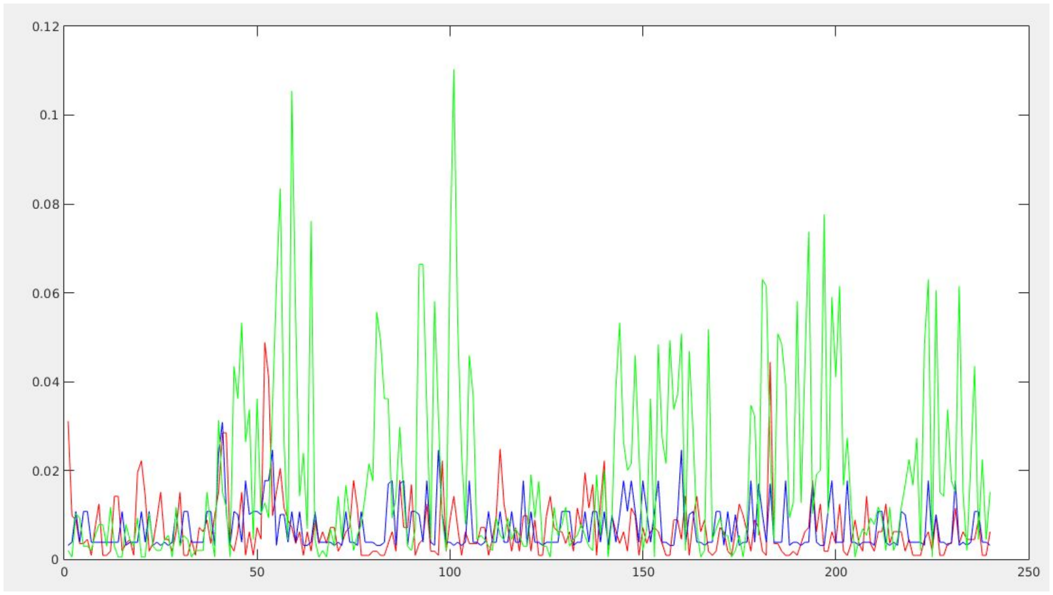
4) 120 degree localisation (for sector 1,3 and 5)
   Threshold values of the three microphones was found out for the 3 region.



Graph obtained for region 5 (red max)



Graph obtained for region 3 (blue max)



Graph obtained for region 1 (green max)

Main code:

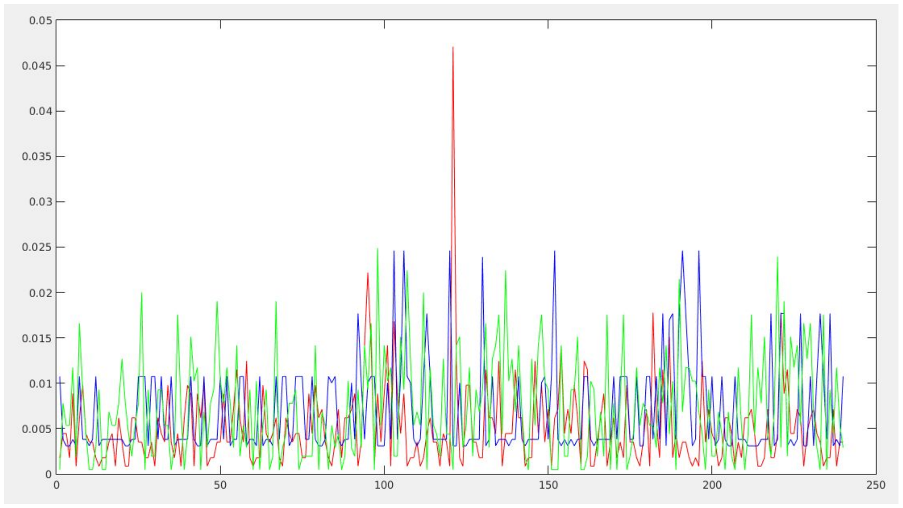
```
% servo_motor = servo(a, 'D4');
writePosition(servo_motor, 0);
clear b;
clear c;
clear d;
B=zeros(1,6);
C=zeros(1,6);
D=zeros(1,6);
j=1;
% b red
% c blue
% d green
for i=1:120
b(i)=(readVoltage(a, 'A0'))-baseb;
c(i)=(readVoltage(a, 'A2'))-basec;
d(i)=(readVoltage(a, 'A4'))-based;

if(b(i)<0)
        b(i)=-b(i);
end

b(i)=b(i)/scaleb;
```

```
if(c(i)<0)
            c(i)=-c(i);
end
c(i)=c(i)*scalec;
            if(d(i)<0)
            d(i)=-d(i);
            end
            d(i)=d(i)/scaled;
B(j)=B(j)+(b(i)*b(i));
C(j)=C(j)+(c(i)*c(i));
D(j)=D(j)+(d(i)*d(i));
plot([1:i], b(1:i), 'r', [1:i], c(1:i), 'b',[1:i],d(1:i),'g');
drawnow
x=mod(i,40);
if x==0
            B(j)=B(j)-avgneb;
            C(j)=C(j)-avgnec;
            D(j)=D(j)-avgned;
 j=j+1;
 flag=0;
 disp(B(j-1));
 disp(C(j-1));
 disp(D(j-1));
% %condition for 1 3 and 5
if B(j)>0.1
            ans=5;
            writePosition(servo_motor, 1);
            pause(2.4);
%  writeDigitalPin(a, 'D5', 1);
 end
   if D(j-1)>0.1
            disp(D);
            ans=3;
 %  writeDigitalPin(a, 'D2', 1);
  writePosition(servo_motor, 2/3);
   pause(2.4);
   flag=1;
   end
  if C(j-1)>0.1
            ans=1;
%   writeDigitalPin(a, 'D3', 1);
 writePosition(servo_motor, 1/3);
  pause(2.4);
  flag=1;
  end
  end
end
end
```

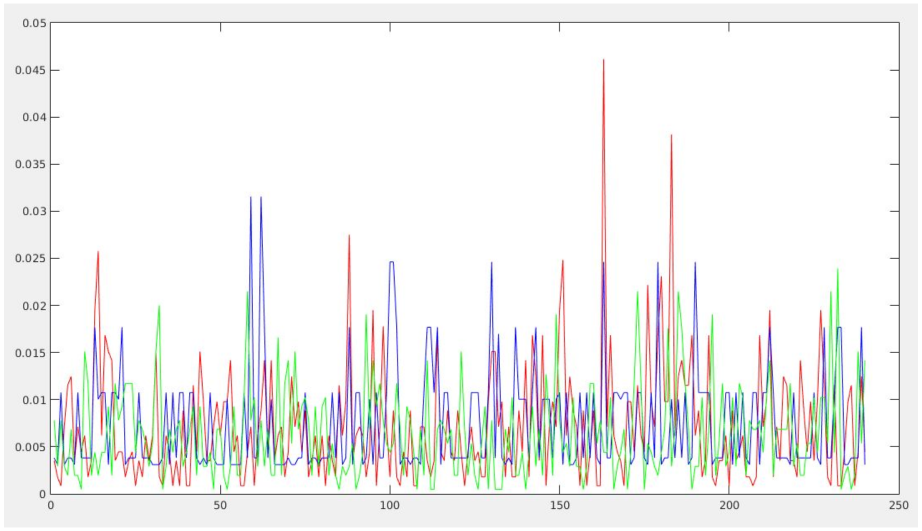
5) Further 60 degree localisation (for sector 2, 4 and 6)

Both adjacent mikes needn't detect at the same sensitivity. The noise threshold for the mikes is also different.While it is in between two adjacent mikes, it is also directly opposite to the third mike. So the third mike also detects signal.These two things caused a lot of problem in localisation of these 3 sectors. Sample values of energy was taken and the data observed to find patterns in measurement.

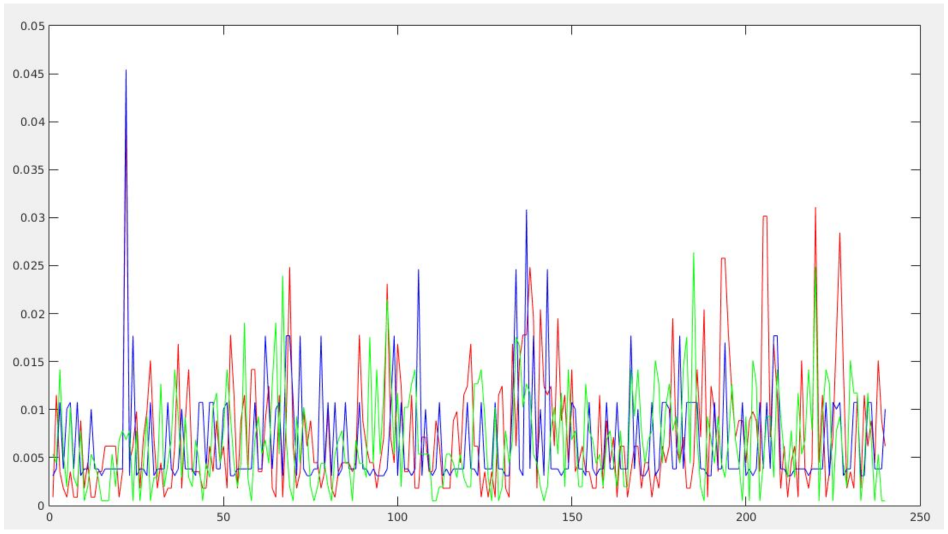
As can be seen from the graphs below not much clear difference was observed.



Graph obtained for region 2



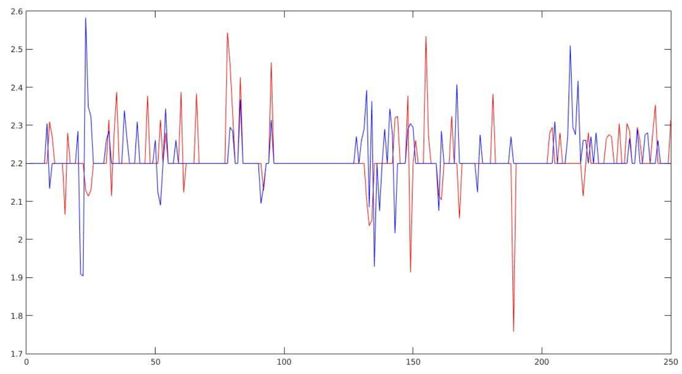
Graph obtained for region 4



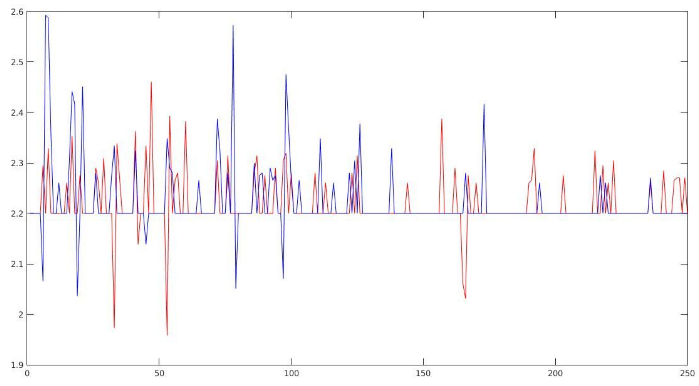
Graph obtained for region 6

DISTANCE ANALYSIS:

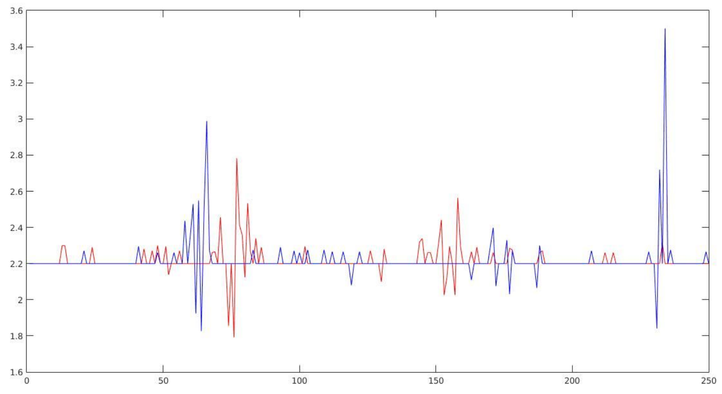
The spacing between the microphones is also going to play an important role in determining the signal interference each microphone gets. For optimal resource usage, this spacing should be minimised.



Least distance between microphones



Moderate distance between microphones



Maximum distance between microphones

When it’s the first two cases then there is lot of overlapping between the signals which makes the job tough to choose which microphone receives maximum signal.

In the third case, overlapping is reduced a lot and we are easily able to distinguish which mike to choose. Therefore the minimum distance between the microphones is chosen to be the distance in the third case.

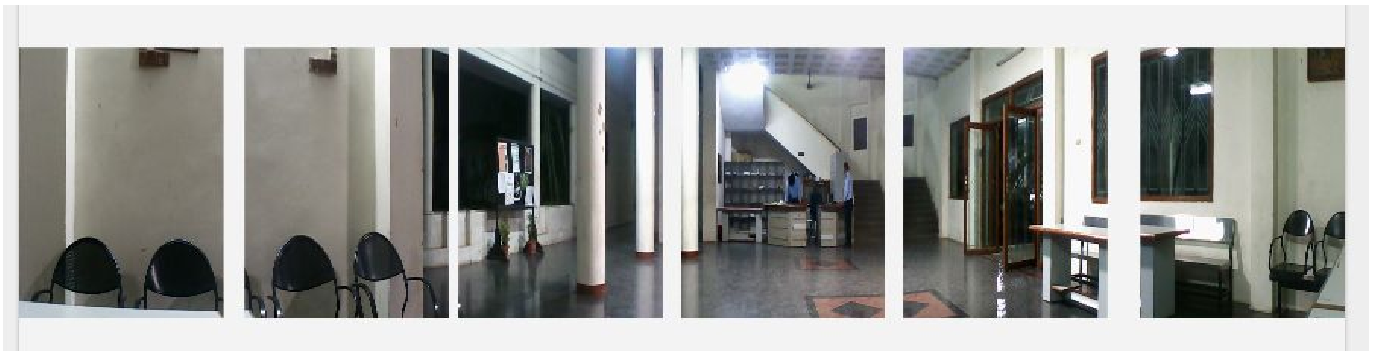
This while using more than two microphones for analysis would require making spatial comparisons.

Similarly we extended it to 3 microphones.

CAMERA, FIELD OF VIEW & SERVO :

- 180 degree servo available.
- Different steps of angles tested.
- The maximum step of rotation of the camera using which the entire 180 degree span was being covered was found to be 30 degree.
- This helped us determine that 3 microphones would be sufficient enough for 180 degree localisation.
- The plan was to use just amplitude calculation for determining the required sector. And then use advanced algorithm to further localise it within 30 degree range. But due to hardware limitations this wasn’t achieved.

The following sequences of images were obtained while using 30 degree steps:



CHALLENGES AND LIMITATIONS :

- The mikes available are not uniform so the resistances and capacitance values had to be kept on the basis of that and they detect the signal only when spoken closer to them. If microphones with high sensitivity are used, then the model can be further optimised and will work more precisely.

- Making of the model (hardware part) took a lot of time as we made 6 circuits on PCBs. Soldering them took a lot of time. It often happened that even if one wire got desoldered, we need to check the whole hardware every time.

- The servo motor is a 180 degree motor.

- Involving arduino with MATLAB was a big challenge. It took time but we did it.

- The software hangs too often. And is able to process only at a limited speed.The latency between arduino and matlab is very high. This was a major hindrance as the camera needs to be rotated fast enough in real time.