



DSP Experiment - 4

Speech recognition with temporal cues

Harshavardhan Alimi

18EC10021


Aim :-

- ❖ Implementation of Shannon's paper on speech recognition with temporal cues on a given speech signal.
- ❖ To observe the voice signals with frequencies in the range from 90 Hz to 5760 Hz.

Results :-

[Link to the processed audio files](#)

Discussion :-

- ❑ Nearly perfect speech recognition was observed under conditions of greatly reduced spectral information.
 - ❑ Temporal envelopes of speech were extracted from broad frequency bands and were used to modulate noises of the same bandwidths.
 - ❑ This manipulation preserved temporal envelope cues in each band but restricted the listener to severely degraded information on the distribution of spectral energy.
 - ❑ The identification of consonants, vowels, and words in simple sentences improved markedly as the number of bands increased; high speech recognition performance was obtained with only eight bands of modulated noise.
 - ❑ The signal is almost similar in the case for 16,128,512 bands of modulated noise, i.e., it gets saturated at 16 bands of frequencies.
- 

- ❑ Higher the number of bands of frequencies higher is the noise distortion in the processed audio file(evident from 512 and 1024 bands of frequencies i.e., the file with 1024 bands of frequencies is barely audible due to noise).
- ❑ After the process of passing the envelope of the extracted signal through a low pass filter, the signal is amplified by 50 times to make the signal audible clearly(audibility decreases due to adding noise and creating more bands of frequencies).
- ❑ The audio files can also be clearly processed with increasing the butterworth filter order(lowpass and bandpass filter) because this will decrease the stop band attenuation even more,making the signals in side filter bands not to intersect.

Matlab codes :-

Function 1:- filtered_env [link](#)

```
function y = filtered_env(x,f1,f2,Ts,s)

    [b,a]=butter(2,[f1*Ts f2*Ts],'bandpass'); %% creating a 4th order bandpass butterworth filter
    filtered_output=filter(b,a,x); %% filtering the main signal:x
    sout=filter(b,a,s); %% filtering the noise : s
    env=abs(hilbert(filtered_output)); %% hilbert transform
    [l_b,l_a]=butter(2,240*Ts,'low'); %% 4th order low pass butterworth filter
    y=filter(l_b,l_a,env); %% filtering
    y=y.*sout; %% multiplying with noise.
    y=y.*50; %% increasing amplitude

end
```

Function 2:- read_audio [link](#)

```
function read_audio(x,bands,Ts,Fs,s)

    spacing=logspace(log10(90),log10(5760),bands+1); %% dividing the bandwidth in logarithmic scale
    y=filtered_env(x,spacing(1),spacing(2),Ts,s); %% for first filter bank
    for i=2:bands
        y=y+filtered_env(x,spacing(i),spacing(i+1),Ts,s); %% to get the summation of all filter banks
    end
    audiowrite("audio_file_bands:"+bands+".wav",y,Fs); %% writing the audio signal into file

end
```

Function 3:- main function [link](#)

```
clear;

[x,Fs]=audioread("fivewo.wav"); %% reading the audio file
```

```
N=length(x);
Ts=1/Fs;
t=0:Ts:(N-1)*Ts;
bands=[1 2 3 4 8 16 128 512 1024]; %%no.of filter banks
z=zeros(size(x));
s=awgn(z,40); %% creating a white gaussian noise
%% extracting the audio for different no.of filter banks
for j=1:length(bands)
    read_audio(x,bands(j),Ts,Fs,s);
end
```