The Description of The Program

The target of the system

The system aims to take the input file and generate an echo of that audio. Then takes that echoed version and gets back the original version of the audio.

Program Flow

The program starts with reading the input audio file which consists of two channels represented in two columns we omit the second channel and just continue with only one channel in one column. Then we transpose the column into a row in order to perform the dot product later. Then we get the length of the input audio array.

Then we start designing the transfer system represented in impulse train. The impulse train is a set of impulses each has its magnitude and delayed a specific period and that's what we did.

Then in order to perform the convolution we have two approaches, either we directly convolve the input to the impulse train but that will be a headache when we want to deconvolve. So, we went with the other approach. We transform the signals into the frequency domain using Fourier Transform. So, the convolution turns to multiplication. So, we applied Fourier transform upon the two signals and then multiply the results to get the desired output in frequency domain. So, we applied Inverse Fourier Transform to turn back to the time domain.

Then in order to get the original signal back, we need to get the inverse of the impulse train. So, since we have the Fourier Transform of the impulse train, all we have to do is to just get the multiplication inverse H^{-1}. Then we multiply that inverse by the output signal to get the original input signal and remove the echo.

The System Equations

$x \equiv Input\ signal\ in\ time\ domain$

$X \equiv Input\ signal\ in\ freq\ domain$

$h \equiv echo\ system(Impulse\ train)$

$H \equiv echo\ system\ in\ freq\ domain$

$y \equiv the\ output\ of\ the\ echo\ in\ time\ domain$

$Y \equiv the\ output\ in\ freq\ domain$

Convolution in time domain

$$y[n] = x[n] \otimes h[n]$$

Convolution in frequency domain (applying the echo)

$$Y(\Omega) = X(\Omega) \times H(\Omega)$$

$$y[n] = IFFT(Y)$$

Deconvolution in time domain

$$h[n] \otimes h_i[n] = \delta[n]$$

$$y[n] \otimes h_i[n] = x[n]$$

Deconvolution in frequency domain (removing the echo)

$$H(\Omega) \times H_i(\Omega) = 1$$

$$Hi(\Omega) = \frac{1}{H(\Omega)}$$

$$X(\Omega) = Y(\Omega) \times H_i(\Omega)$$

$$x = IFFT(X)$$

The Code

```
%****getting the input audio****%
[x,FS] = audioread('Example.mp3');
%Since the audio is stereo, it's stored in two columns
to represents the two channels
%So, we take only one column i.e., channel and
transpose
x=transpose(x);
x=[x(1,:)];
%Get the length of the audio after transpose
xlen = length(x);
%Get the period of the audio in seconds to use it later
T = xlen/FS;

%****designing the echo system****%
delay1 = 0.2*FS;
mag1 = 0.8;
delay2 = 0.3*FS;
mag2 = 0.5;
delay3 = 0.5*FS;
mag3 = 0.2;

%Choose the strength of the echo and then get the
impulse train
prompt = ['Enter the strength\n 1. strong. 2.medium.
3.weak.\n'];
strength = input(prompt);
switch(strength)
    case(1)
```

```
        h = [1,zeros(1,delay1),mag1];
    case(2)
        h = [1,zeros(1,delay2),mag2];
    case(3)
        h = [1,zeros(1,delay3),mag3];
    otherwise
        prompt = ['invalid choice, choosing strong'];
        strength = 1;
        h = [1,zeros(1,delay1),mag1];
end

%Getting a multiple echo if desired
%h =[1,zeros(1,delay1),mag1,zeros(1,delay2),mag2,
zeros(1,delay3),mag3];



%****getting the echo****%
hlen = length(h);
%getting the length of the impulse train
Lout = xlen+hlen-1;
%Getting the Fourier transform of input and the impulse
train
X = fft(x,Lout);
H = fft(h,Lout);
%Apply the convolution in frequency domain
Y = X.*H;
%Getting the inverse transform of the output
y = ifft(Y);
%Play and save the audio output
sound(y,FS);
switch(strength)
    case(1)
        audiowrite('Strong_Echo.wav',y,FS);
    case(2)
        audiowrite('medium_Echo.wav',y,FS);
    case(3)
        audiowrite('weak_Echo.wav',y,FS);
end

%Pause the system for a period equals the audio file to
prevent interference
pause(T);

%****getting the deEcho*****%
%Getting the inverse of the transfer function in
frequency domain
```

```
H_inv = 1./H;
%Getting the deconvolution in frequency domain to
restore the input
W = Y.*H_inv;
%Getting the inverse Fourier transform of the output
w = ifft(W);
%Play and save the audio output
sound(w,FS);
switch(strength)
    case(1)
        audiowrite('Strong_deEcho.wav',w,FS);
    case(2)
        audiowrite('medium_deEcho.wav',y,FS);
    case(3)
        audiowrite('weak_deEcho.wav',y,FS);
end
```