Yelda Gungor
Koray Cimen
25.1.2016

MIDDLE EAST TECHNICAL UNIVERSITY

ELECTRICAL AND ELECTRONICS ENGINEERING

EE430-Digital Signal Processing

TERM PROJECT-2016 Phase II

Koray Cimen-1877265

Yelda Gungor-1936947

Section-1

Yelda Gungor
Koray Cimen
25.1.2016

1. Project Description

Phase II of the project will develop a module, which consists of four tasks such as; Real Time Spectrum, Equalizer and Filtering, Special Audio Effects and Interfering Tone Removal. Also module includes the changes that are done since the Phase I of the project. In Phase I, Data Acquisition and Spectrum parts of the project were done and requirements were complied. They will be mentioned in this phase as well. One part of the Phase II, Real Time Spectrum allows anyone to look at signals in frequency domain rather than in the time domain. The requirement of this part is that the variables must be adjustable by the user. Equalizer and Filtering part includes creating a user interface and a system being able to amplify, suppress the specified frequency bands by using linear phase filters. In addition to these, a filter nonlinear phase filter design is used for 250-500 Hz. Another part is Special Audio Effects, which includes reverberation, synthetic stereo and changing the speed of the sound. Last part is the Interfering Tone Removal. First, this system will add a pure cosine signal to recorded audio. Then system will remove the noise and will be playing the filtered audio and the noisy one. At the end, system can even estimate the frequency of the noise.

2. Data Acquisition

Data acquisition is a process of capturing analog audio signal from environment by taking certain amount of samples per second. This part requires completing two tasks to be successful. First of them is being able to capture an audio playback from any device. Second is processing the mp3 files in computer by turning the file into proper format such as ".wav" format.

To record an audio playback:

We can create audio recorder object with specific parameters, which are sampling frequency, bits per sample and channel number. Then, MATLAB will be able to record audio for a desired duration. Related functions are as follows:

audiorecorder: creates a recorder object with desired parameters

recordblocking: records audio signal into created object for desired duration

getaudiodata: take all the signal values recorded in the object and writes all of them into an array as double values

To process mp3 files:

MATLAB is able to pick any mp3 file in the computer and convert it to .wav format. Related functions used are as follows

uigetfile(): open file directory to browse file

audioread: convert any music file into '.wav' format(available in MATLAB 2015a)

The user interface of the system is changed since the Phase I and is given in Figure 1.

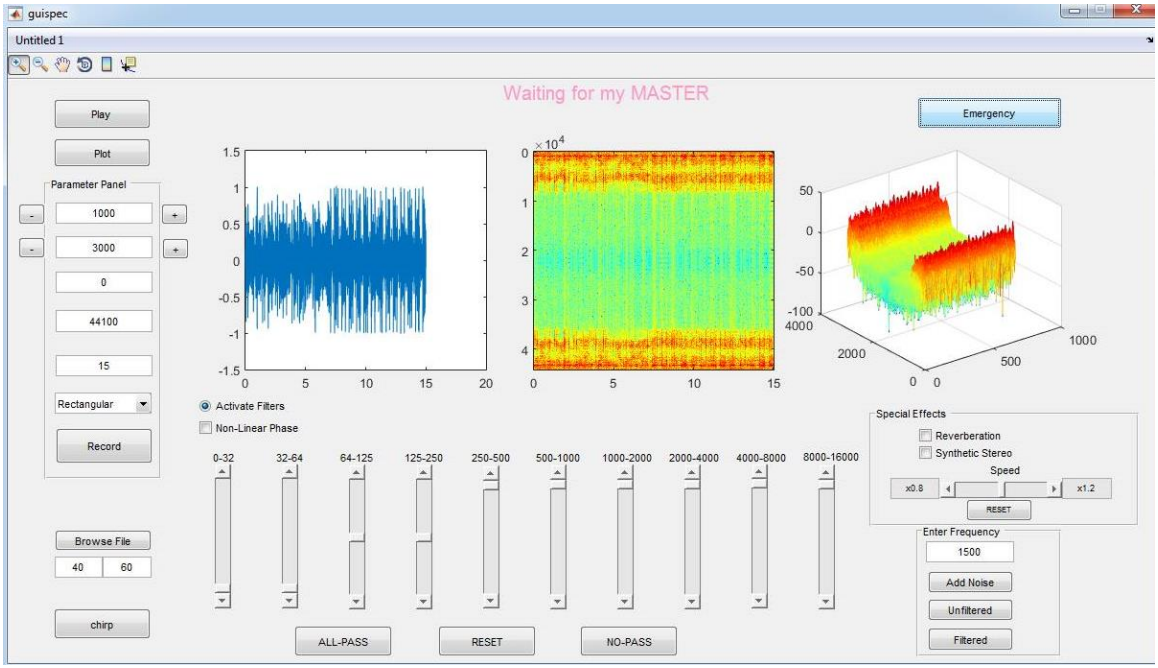

Figure 1

Record(push button): Records the audio signal.

Sampling Frequency(text input): Adjustable number of samples per second in the audio signal by user.

Time(text input): The desired duration of recording.(unit is seconds)

Browse(push button): The desired file could be selected from Browse button.

Start-Stop(text input): Determines the time interval of the browsed sound file to be processed.

Counter Region(static text): Displays the status of system; "Recording" or "Done".

3. Spectrogram

3.1. Description

As mentioned in Phase I, spectrogram is a visual representation of the spectrum of frequencies in a sound as they vary with time. It is a process of dividing the signal into chunks, taking the DFT of all chunks and representing it in a 2D or 3D plot. Horizontal axis represents time; vertical axis represents frequency and the colors represent the amplitude of the corresponding frequency. Adjustable parameters are 'window size', 'overlap size', 'DFT size' and 'window function'.

- **Window Size:** Window size is the number of samples in a chunk.
- **Overlap Size:** Overlap size is the number of samples that are same in consecutive chunks.
- **DFT Size:** Increasing the DFT size requires more calculations. However, frequency resolution of a chunk is much higher because higher number of DFT takes more samples from the DTFT of a discrete time signal.
- **Window Function:** Window function is very important parameter for spectrogram. Windowing a signal develop non-zero and non-sense frequency values appearing in the spectrum. These are called side lobes. In order to smooth or reduce these side lobes, windowing functions can be very beneficial. None of the window types has advantageous over the others. All window functions are used for different purposes in practical applications. In the system, we have six types as; Rectangular, Hamming, Blackman, Triangular, Gaussian, Tapered Cosine.

3.2. Algorithm

For spectrogram, STFT (Short Time Fourier Transform) of the signal should be calculated. Related algorithm for spectrogram and data acquisition is given below.

Pseudo Code

- open file directory and save the directory;
- write all data in to matrix y by using audioread;
- calculate the number of chunks n;

for k=1:1:n

- circularly shift input signal by wsize-overlap;
- take the values up to wsizeth element;
- write it to kth column of matrix 'main';

end

- select window type;
- apply window to main matrix;
- take the DFT of main matrix;
- convert DFT magnitude into dB
- discard the conjugate symmetrical part;
- display as image;

3.3. Results and Discussions

In Phase I, the results were shown. Here all effects will be displayed again but some conditions will be discarded. Even though our interface is changed, results will be displayed in old interface.

- Effect of Window Size: Window size is changed while other parameters remain same. Overlap Size=0, Sampling Frequency=10000, DFT size=10000, Window Type=Hamming. Results are shown in Figure 2&3.

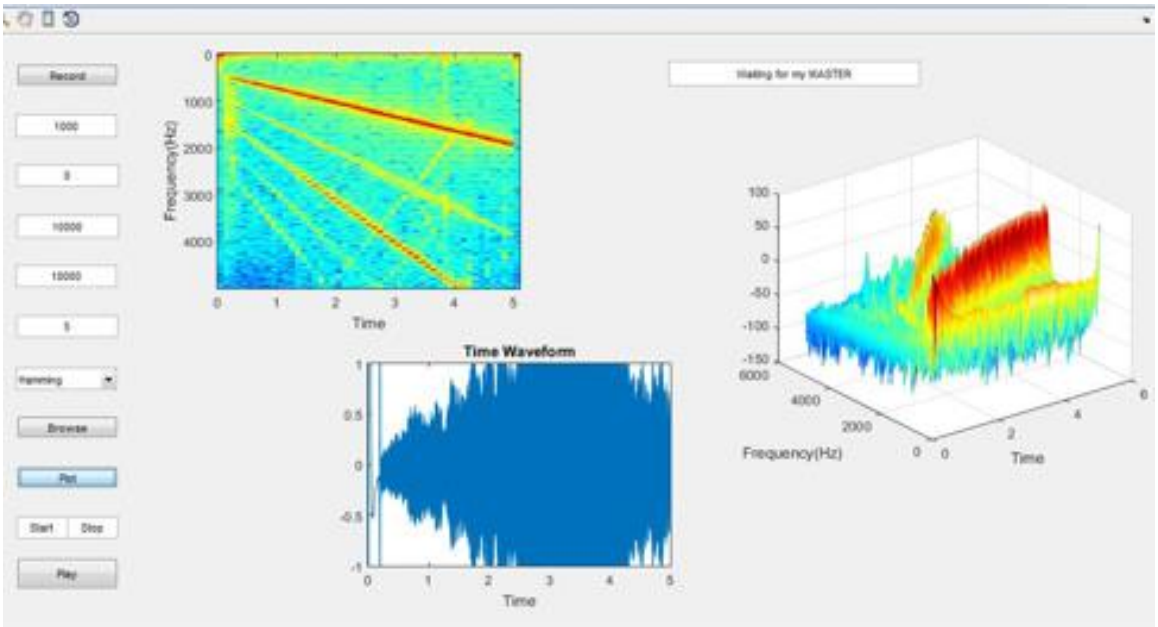

Figure 2-Window Size=1000

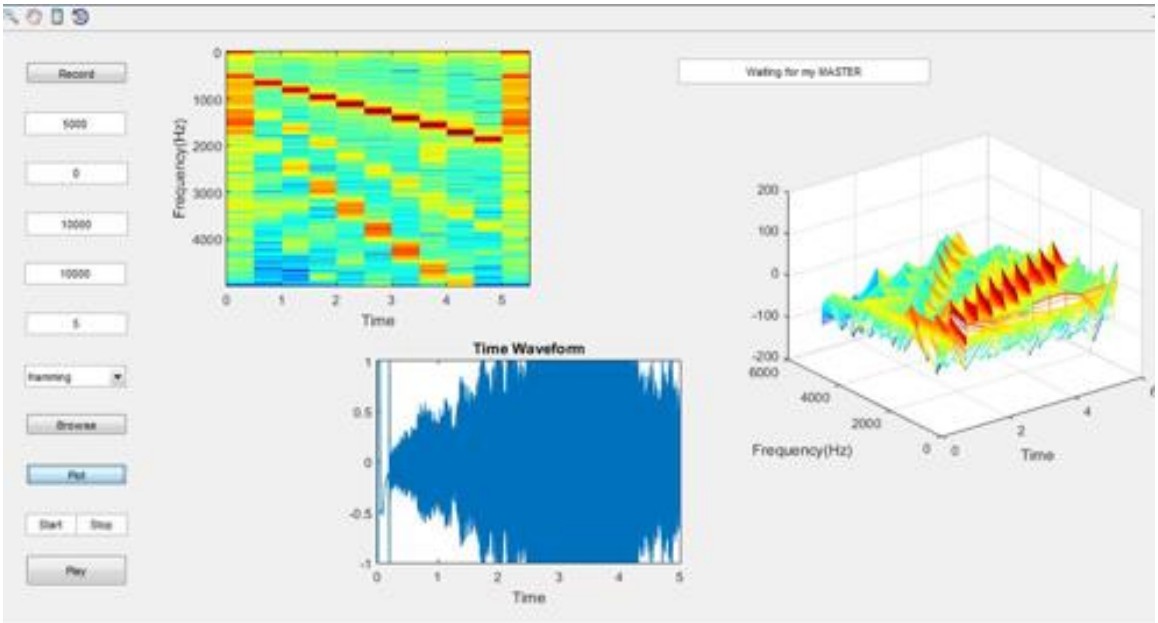

Figure 3- Window Size=5000

As window size gets larger, frequency resolution of spectrogram increases with the cost of decrease in time resolution. In other words, frequency content of chunks can be observed with better resolution, but change in frequency content of consecutive chunks can be observed with poor resolution.

Yelda Gungor
Koray Cimen
25.1.2016

- Effect of Overlap: Overlap size is changed while other parameters remain same. Window size=1000, Sampling Frequency=10000Hz, DFT Size=10000, Window Type=Hamming.

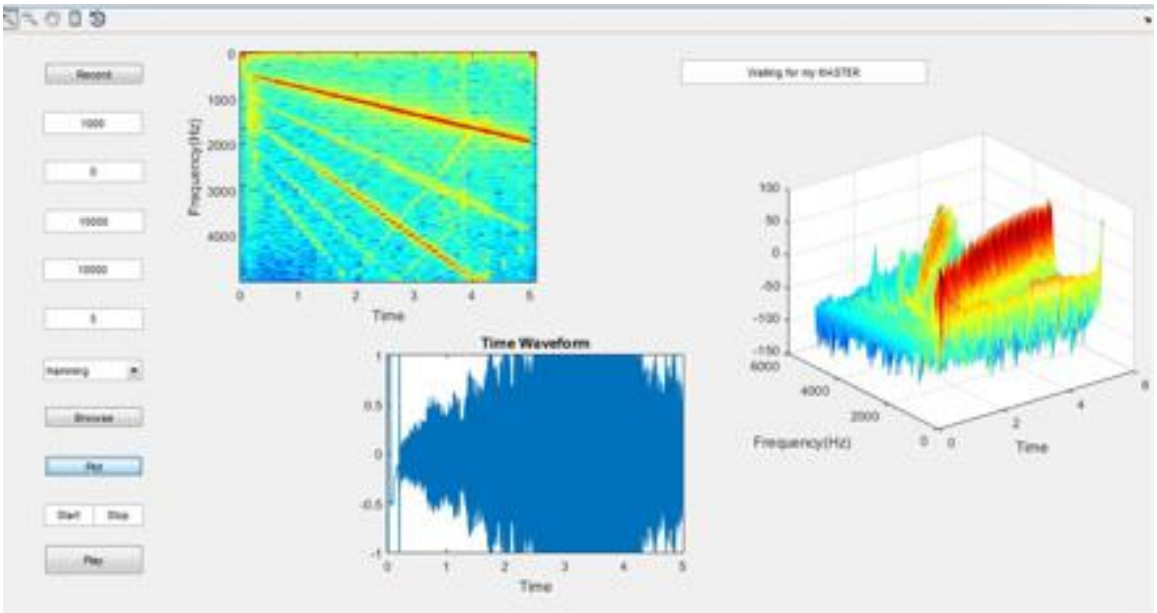

Figure 4- Overlap Size=0

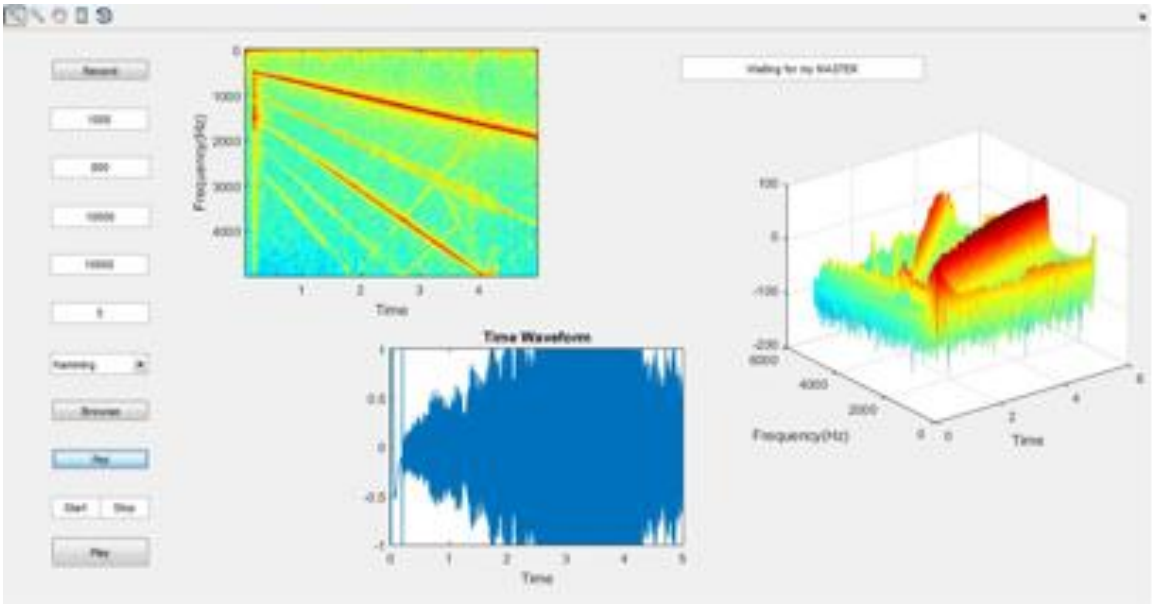

Figure 5-Overlap Size=800

As the overlap size increases, difference between time slices will get smaller and similarity between consecutive chunks will increase. Increasing the overlap size also increase the number of required calculations.

- Effect of Window Type: Window Type is changed while other parameters remain same. Window size=1000, Sampling Frequency=10000Hz, DFT Size=10000, Overlap size=0.

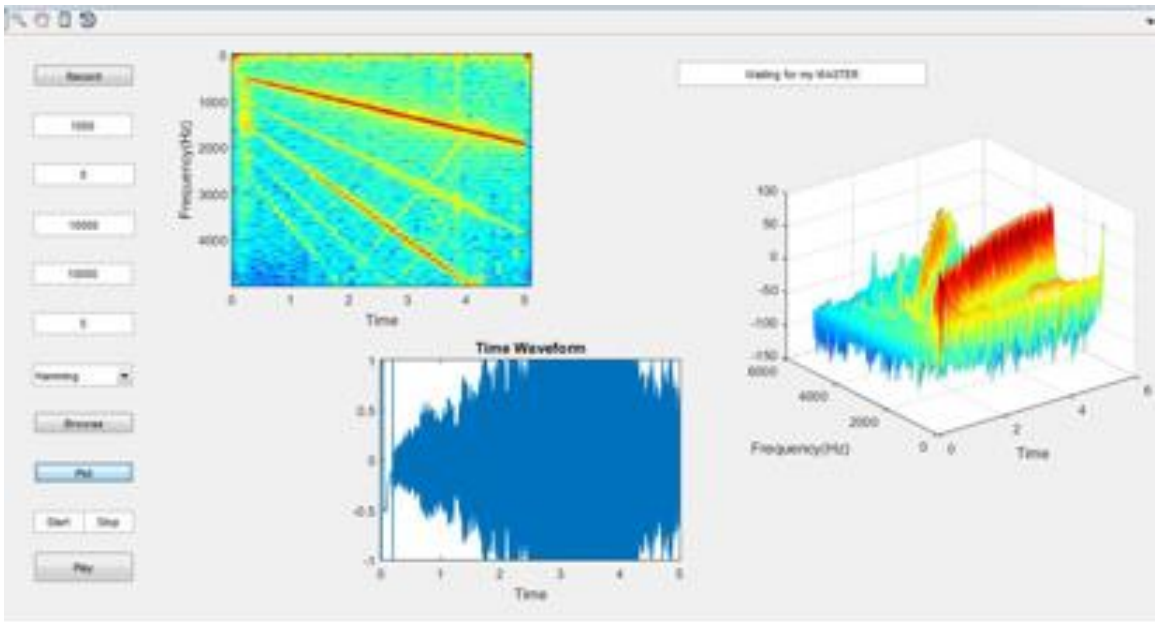

Figure 6-Window Type=Hamming

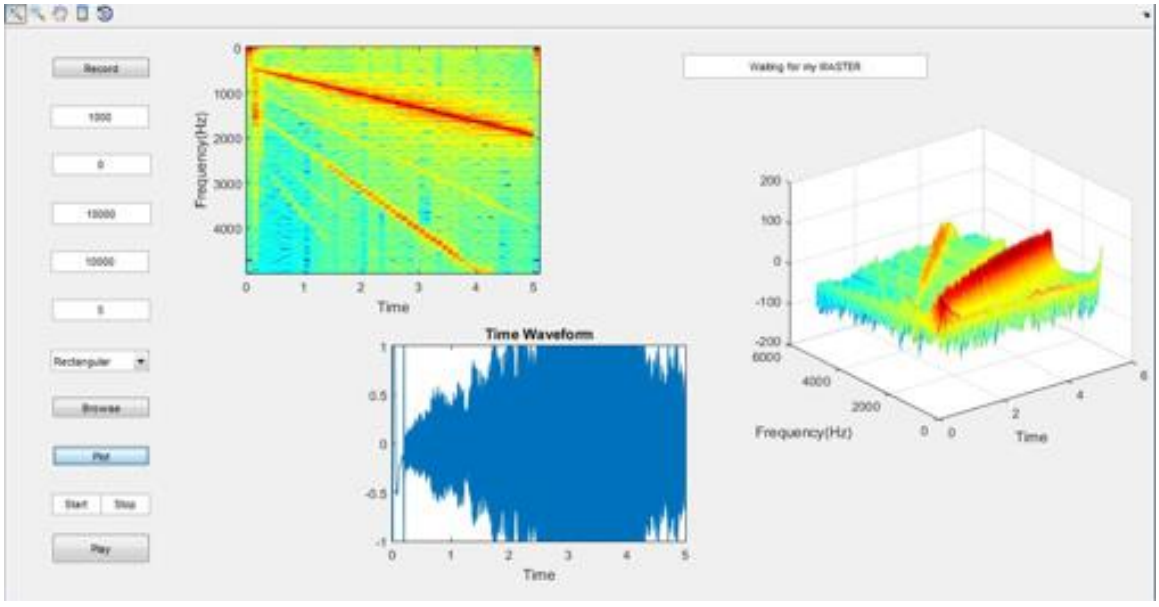

Figure 7-Window Type=Rectamgular

Rectangular window gives very accurate results when observing the difference in power densities of different frequencies away from each other. However, it has the highest amount of spectral leakage among the all windowing functions. Hamming windows is

another type of window, which has lower side lobes in the spectrum, but it has low peak value of main lobe compared to rectangular window.

- Effect of Sampling Frequency: Sampling Frequency is changed while other parameters remain same. Window size=1000, Window Type=Hamming, DFT Size=10000, Overlap size=0. Effect of sampling frequency is to be able to capture higher frequencies with higher sampling rates. With 10kHz sampling frequency, one cannot detect or calculate the frequencies higher than 5kHz because maximum frequency in DTFT is π. As it can be seen from spectrograms below, with 20kHz sampling frequency, maximum frequency we can detect is 10khz while 5kHz in 10kHz sampling rate.

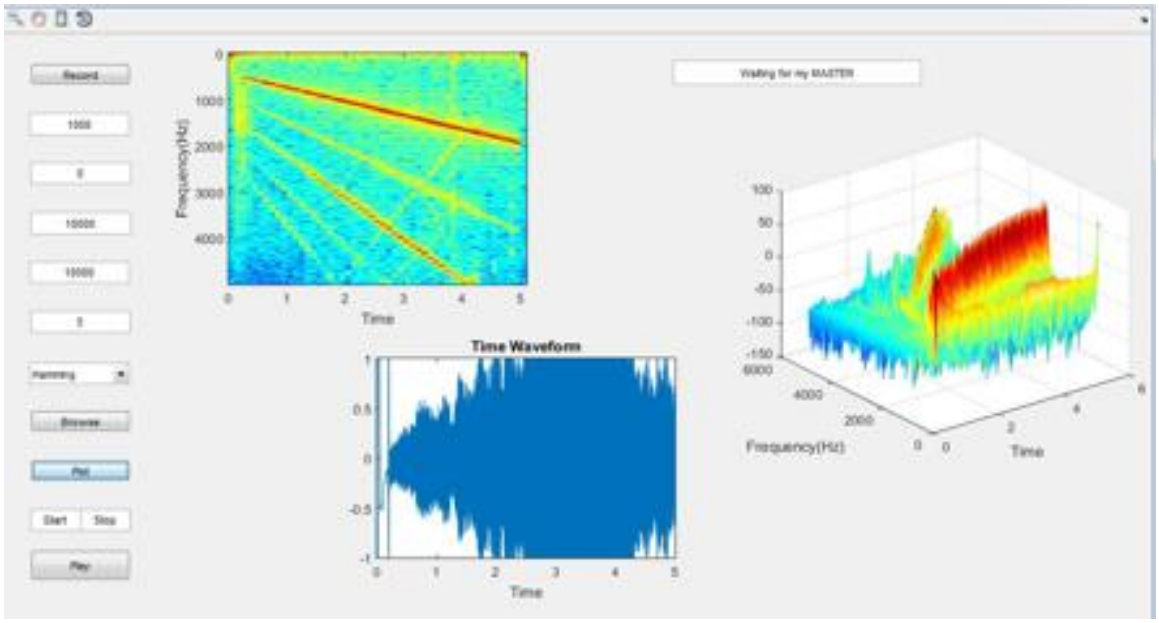

Figure 8-Sampling Rate=10000 Hz

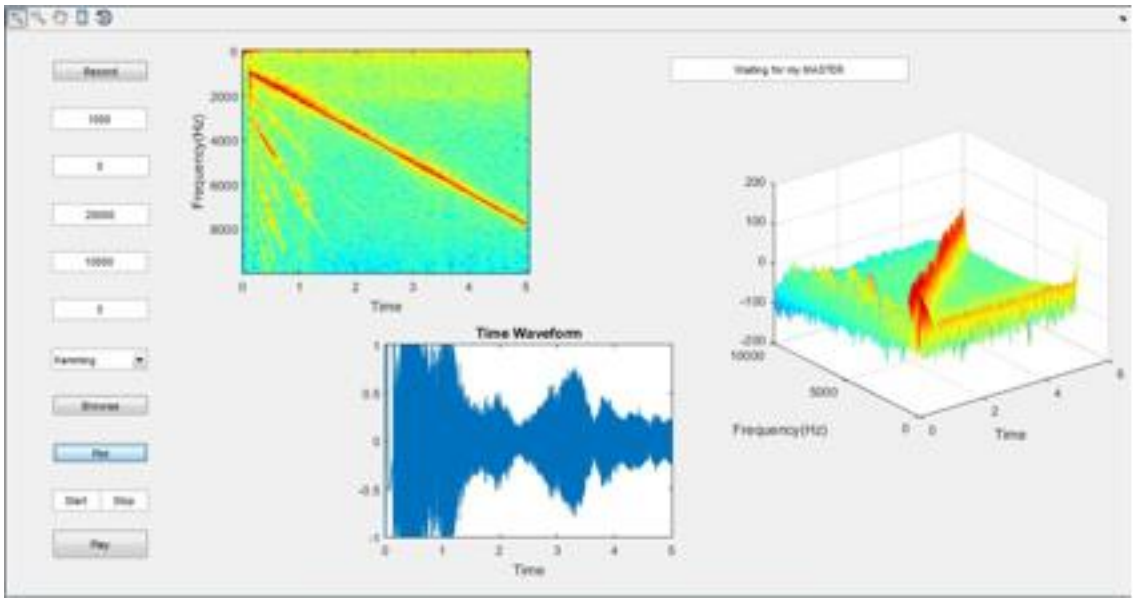

Figure 9-Sampling Frequency=20000Hz

Besides the recording, system has also the ability to process mp3 file. For this purpose, we used 'Michael Jackson-Billie Jean.mp3' and displayed the spectrogram and time waveform for 10 seconds. Window size=1000, Overlap size=0, Sampling frequency=44100, Window type=hamming shown in Figure 10.

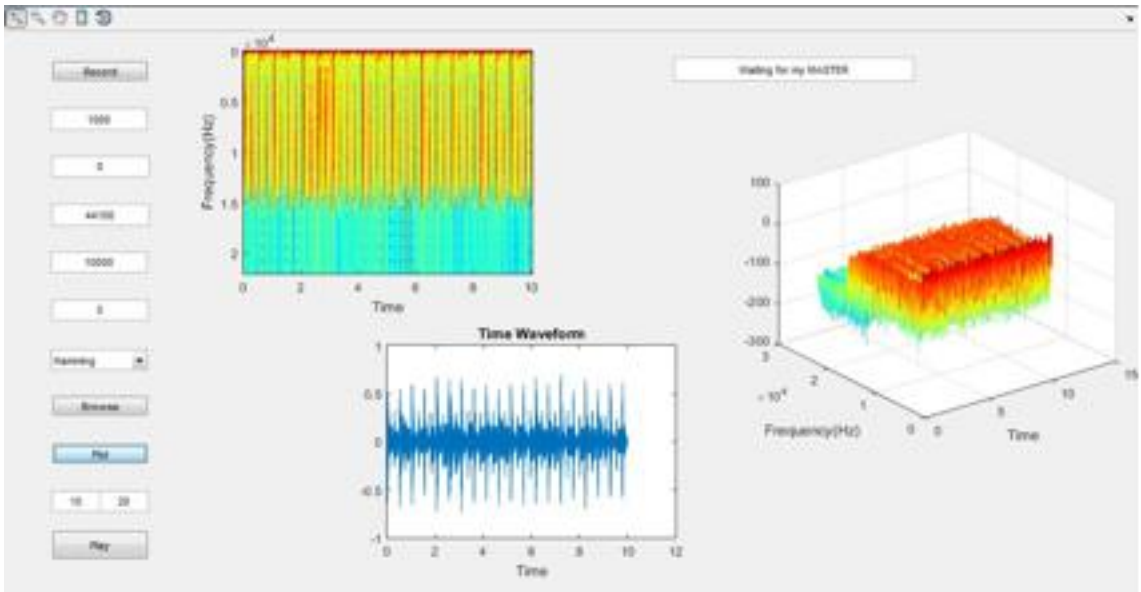

Figure 10

Yelda Gungor
Koray Cimen
25.1.2016

4. Real Time Spectrum

4.1. Description

As another feature, system is expected to get recorded audio in frames while simultaneously showing the spectrum (DFT) of the current frame. Frame size, DFT size and also window type will be parametric and adjustable for the user from the interface. The parameter places and related buttons are shown in Fig. 11 on the GUI.

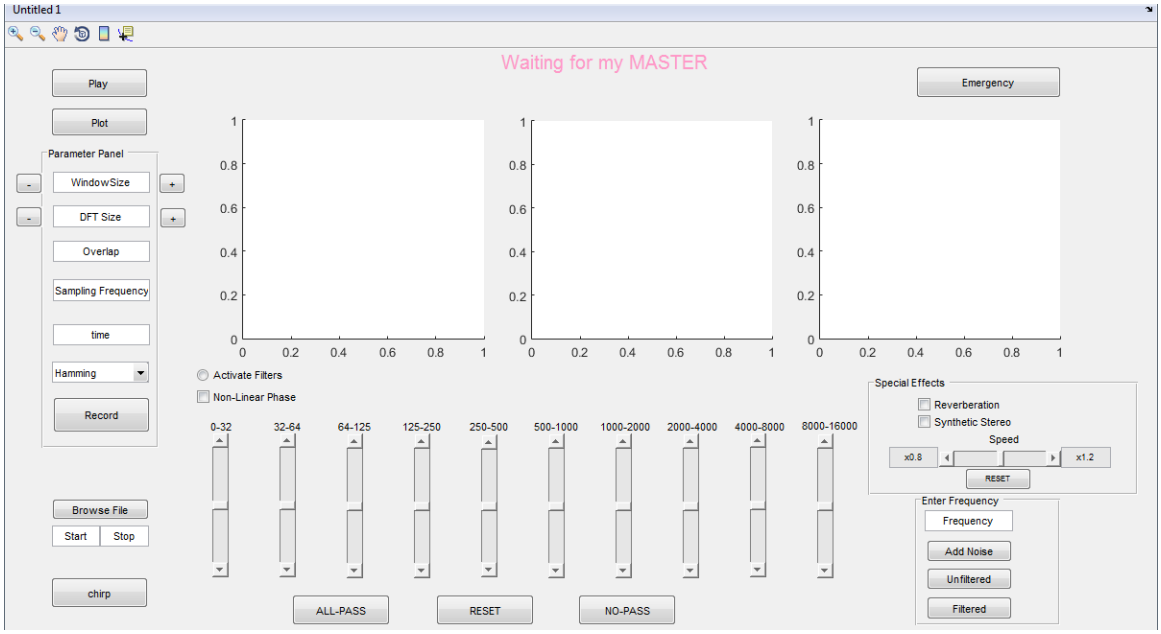

Figure 11

In the **"Parameter Panel"**;

- **Window Size (Text):** Desired window size can be adjusted. It can be changed from (+) and (-) buttons during the recording as well.
- **DFT Size (Text):** Size of DFT can be adjusted. It can be changed from (+) and (-) buttons during the recording as well. (Note: DFT size should be equal to or larger than the window size)
- **Sampling Frequency (Text):** Sampling frequency of the recording can be adjusted before the recording starts.
- **Time (Text):** Desired duration of the recording can be adjusted before the recording starts.

- **Window Function (Pop-up Menu):** Type of window can be adjusted from the window type pop-up menu. (Hamming window is chosen in the above figure)
- **Record (Push Button):** After setting all the parameters above, record button initiates the recording.

4.2. Algorithm

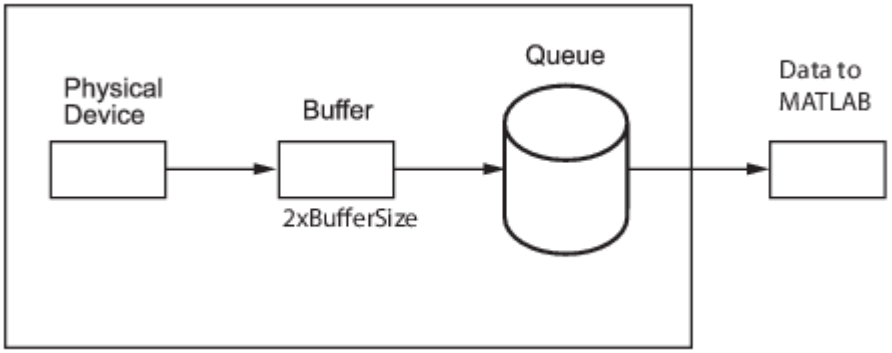

Figure 12-Stream Processing

In MATLAB, stream processing is done by using certain objects. Required objects for real time spectrum are as follows:

- **dsp.AudioRecorder:** Incoming analog audio data from environment is sampled in audio card of the computer and can be streamed to MATLAB frame by frame. Audio recorder object can adjust sampling frequency, number of samples per frame and also queue duration.
- **dsp.SpectrumAnalyzer:** While streaming data with dsp.AudioRecorder object, spectrum analyzer object can process the data by taking the DFT of the current frame and visualize it on a separate screen. Window size, DFT size and window type can be adjusted.

Pseudo Code
While (time< desired duration) ➢ Stream single frame; ➢ Delete the 2nd coloumn of the matrix for the sake of simplicity; ➢ Check the window length, DFT size and window type from parameter panel and update; ➢ Apply the window type; ➢ Send the proccessed data to spectrum analyzer object; **end**

4.3. Results

Results are satisfactory. Successful visualization of the frames while recording could be observed from the screen. Horizontal axis represents the frequencies and vertical axis represents the magnitude of the frequencies in dB unit.

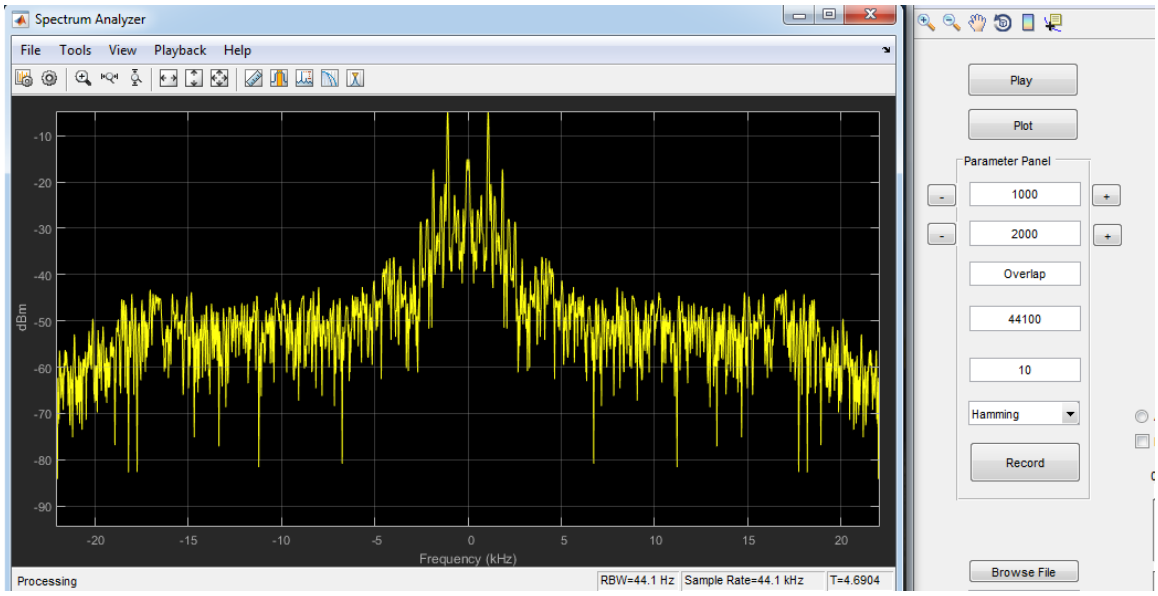

Figure 13-DFT of a Single Frame

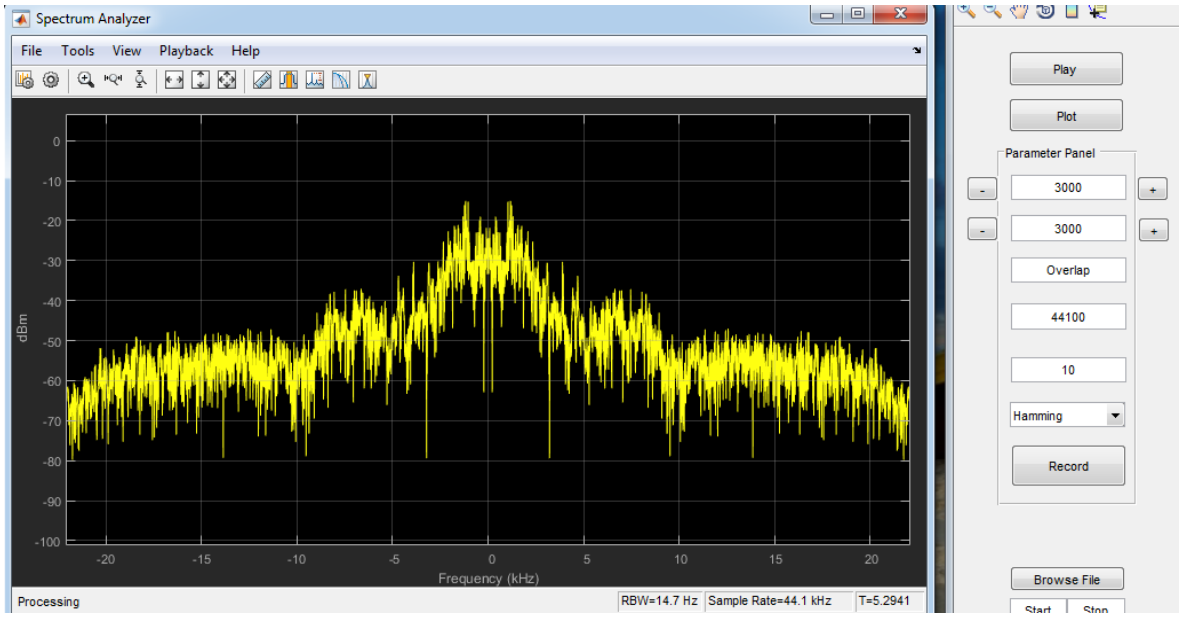

Figure 14-DFT of a Single Frame

After adjusting the parameters from the panel, DFT of the current frame can be seen from the figure above. While processing, by using (+) and (-) buttons, corresponding parameters can be changed as well as the window type.

Effect of window size can be seen from the above figures by looking at their densities simply. Window size of 3000 samples requires more computation time, but shows the spectrum content of a longer frame.

5. Equalizer and Filtering

5.1.Description

a) By the help of interface, system is able to suppress or amplify the frequencies specified as follows:

- 0-32 Hz
- 32-64 Hz
- 64-125 Hz
- 125-250 Hz
- 250-500 Hz
- 500-1000 Hz
- 1000-2000 Hz

- 2000-4000 Hz
- 4000-8000 Hz
- 8000-16000 Hz

Input and output waveforms and spectrogram should be displayed. Besides, all the frequency selective filters are linear phase filters. MATLAB has its own function to produce coefficients of a desired digital filter whose specifications are defined. For this part, linear phase FIR filters are used.

b) For the 250-500Hz filter, applied filter should be non-linear phase. Outputs of linear and non-linear phase filters will be compared.

5.2. Algorithm

a) Equalizer with Linear Phase Filters

MATLAB has some objects to stream audio data, process and visualize. Required important objects and functions are as follows:

- **dsp.SignalSource:** Acquire single frame with a specific length from an array. Samples per frame can be adjusted. For the sake of simplicity and since it is sufficient to process a data, 1 second frame is chosen.
- **dsp. AudioPlayer:** Send a data frame to physical device (speaker of the computer or ear phones). Sample rate and channel mapping can be adjusted.
- **fir1:** Produce the nominator coefficients of a linear phase FIR filter. Note that FIR filters do not have any denominator coefficients except 1 for z^{o}.

Recorded audio data has already been written in an array in the previous parts. That array is then processed frame-by-frame and sent to speaker with certain parameters. Related algorithm is as follows:

Pseudo Code

- Filter coefficients of all frequencies are calculated with fir1 function(order of filters=600);

  While (length(streamed data)<length(whole recorded data))

  - Stream single frame;
  - Get all weights of frequencies from the sliders on the interface;
  - Multiply the filter coefficients with their weights and sum;
  - Apply the filter;
  - Send the proccessed data to audio player object;
  - Plot the original data;
  - Plot the filtered data;

  end

- Display the spectrograms of the whole data(both unfiltered and filtered);

Note that spectrograms are not displayed in real time because of timing problems. MATLAB is a powerful program, but calculating spectrograms and visualizing them in real time causes serious timing problems and significant latency for the filters have been observed. Therefore, we decided to show the spectrograms at the end of the loop.

b) Equalizer with Non-Linear Phase Filters

IIR filters have non-linear phase response in general. MATLAB has the ability to produce the filter coefficients of desired IIR filters. In our project, we use **cheby1** filter to produce a non-linear phase filter.

Yelda Gungor
Koray Cimen
25.1.2016

Pseudo Code
➢ Filter coefficients of all frequencie are calculated with cheby1 function(order of filter=3); **While (length(streamed data)<length(whole recorded data))** ➢ Stream single frame; ➢ Apply the filter; ➢ Send the proccessed data to audio player object; **end**

5.3. Results

Magnitude responses of the linear phase and non-linear phase filters are shown in Fig. 15 and Fig. 16.

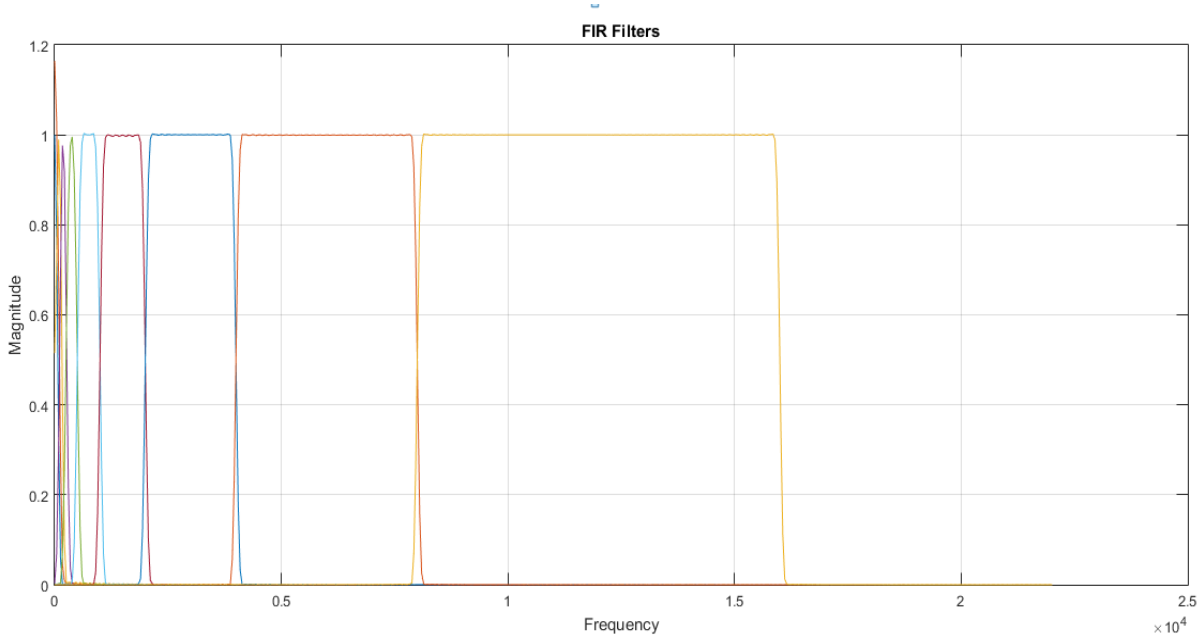

Figure 15

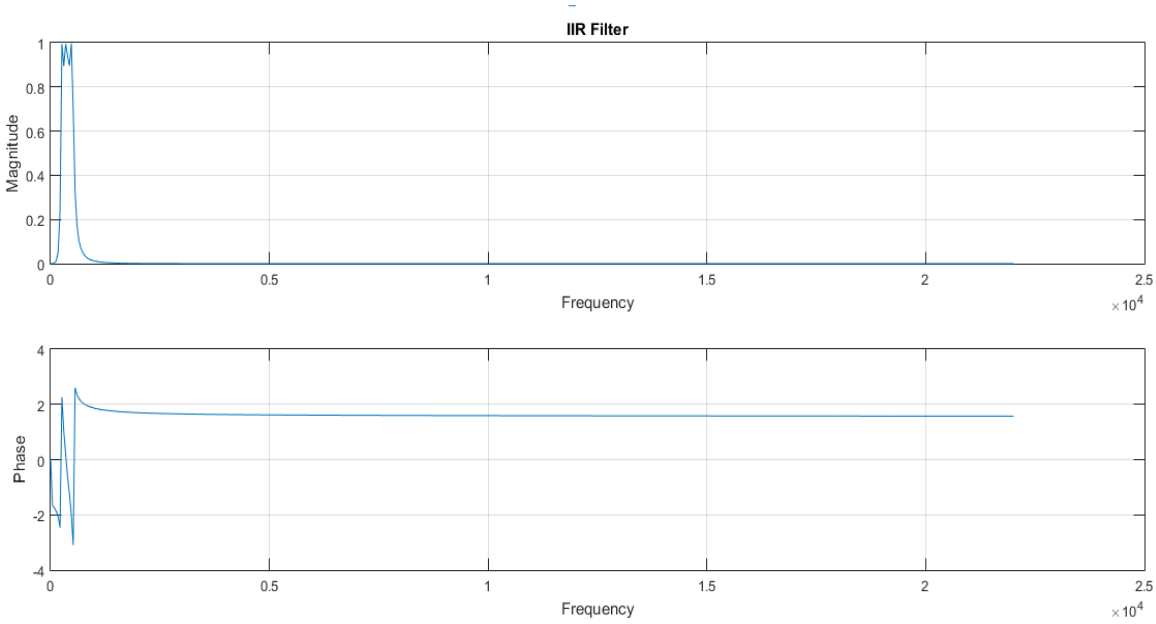

Figure 16

Non-filtered and filtered (FIR filter is used) time waveforms of a single frame while playing the audio can be shown in Fig. 17. Position of the sliders that determine the weights of the filters can also be seen.

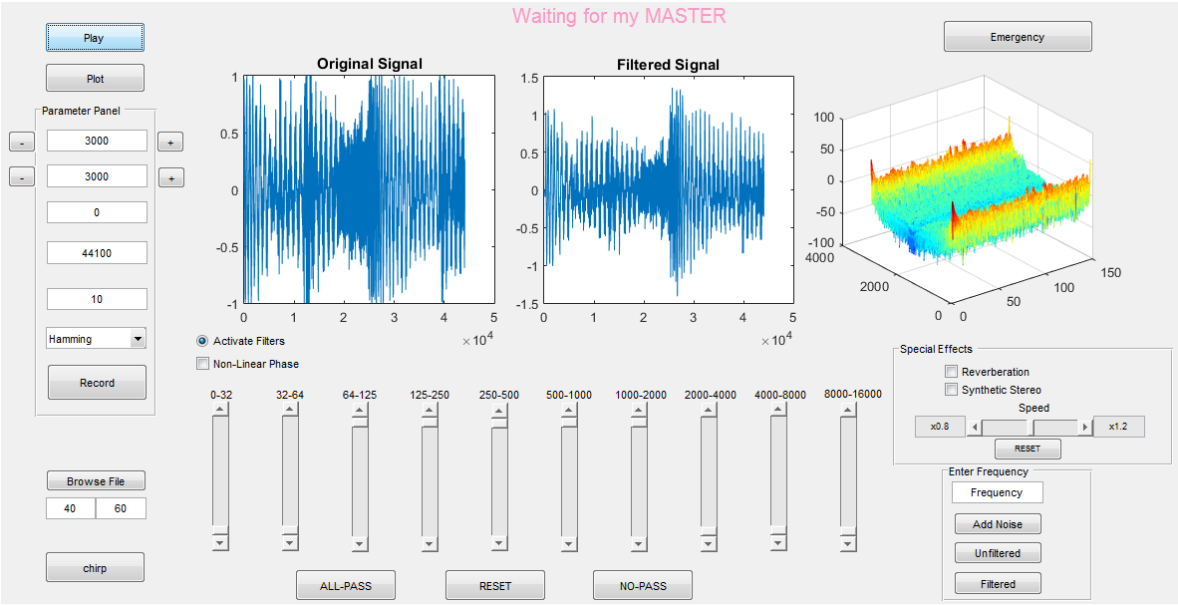

Figure 17

Yelda Gungor
Koray Cimen
25.1.2016

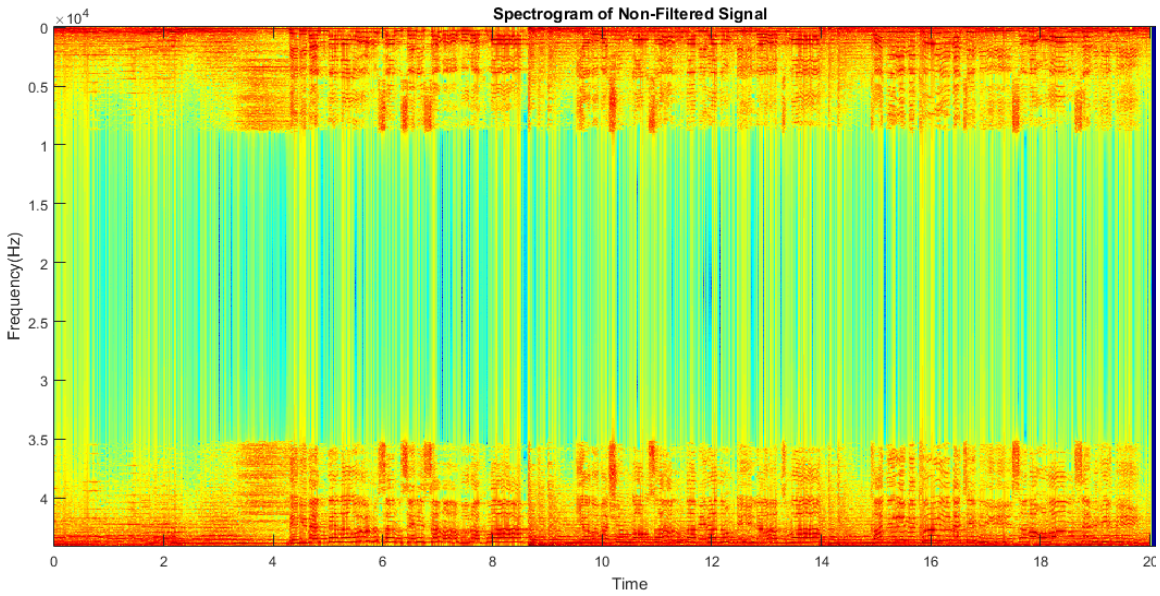

Figure 18

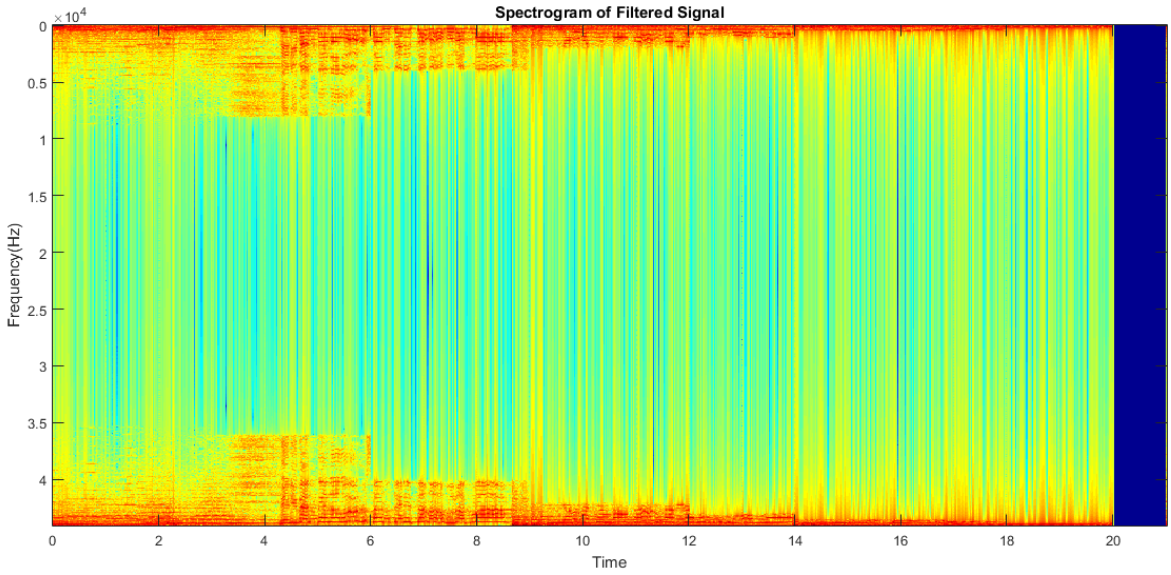

Figure 19

As it can be seen, as time goes, each slider has been consecutively pulled down to suppress their corresponding frequencies. At the end of the song, all filters suppress their bandwidths and no frequency is able to survive giving pure blue color map indicating the lowest dB values.

Sound effect of the nonlinear phase filter was the most challenging part and we actually could not have a successful result when comparing the linear and nonlinear phase filters. Only applying 250-500Hz filter could not create observable changes in the sound.

6. Special Audio Effects

By digitizing an analog audio signal and using DSP, audio signal can be manipulated in an infinite amount of ways that can sound different. In project description, it is required to develop special audio effects, which are Reverberation, Synthetic Stereo and changing the speed of the audio by an external factor. The panel that creates these special effects can be seen from Fig. 20.

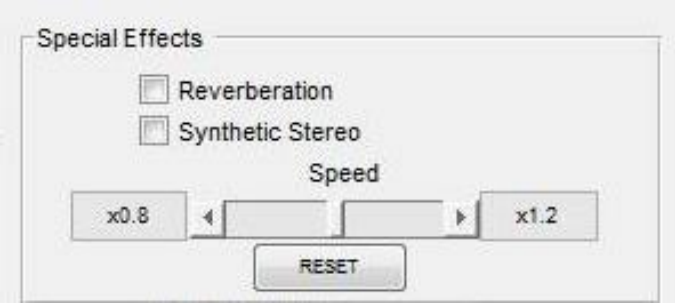

Figure 20

6.1.Reverberation

Direct sound is the sound wave that reaches the listener's ear directly from the source. As for the reflected sound, it travels longer path than direct sound to reach the listener's ear. Since the reflected sound might keep bouncing off many surfaces, a continuous stream is created and it becomes a single entity, which continues after the original sound decreases and disappears. This is called the reverberation, which can be realized by a single pole IIR Filter, whose transfer function is shown in Fig. 21.

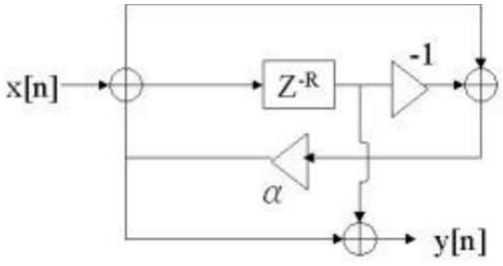

Figure 21

$$H(z) = \frac{\alpha + z^{-R}}{1 + \alpha z^{-R}}, \quad |\alpha| < 1$$

In our case, we take α as 0.8 and R as 3000.

Reverberation checkbox in Fig. 20 is able to add reverberation effect at any time during the playing and vice versa.

6.2. Synthetic Stereo

Synthetic stereo is simply achieved by creating a delayed version of the audio and adding to itself. Block diagram of the process can be seen from Fig 22.

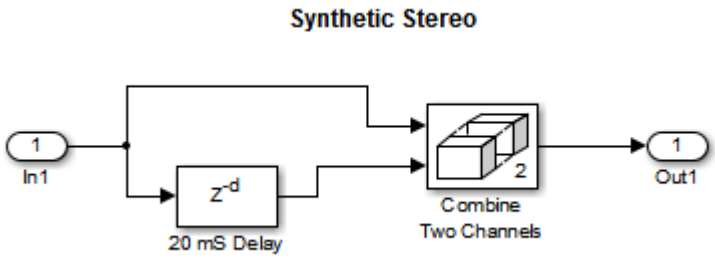

Figure 22

$$H(z) = 1 + z^{-d}$$

$$Our\ selection\ is\ d = 5000$$

Similar to reverberation, synthetic stereo checkbox in Fig. 20 is able to add stereo effect at any time during the playing and vice versa.

6.3. Changing the Speed of the Audio

Changing the speed of an audio is simply sending more or less samples to the physical device in one second. This effect is achieved by changing the property of the dsp.AudioPlayer object when desired. A slider can be seen from the Fig. 20 with the title 'speed'. Minimum speed is x0.8 of the original speed while maximum speed is x1.2 of the original speed.

7. Interfering Tone Removal

7.1.Description

It is usual in practice that recorded sounds are contaminated by tones coming from other devices or environment. This is called interference and annoying or frustrating in practical

life. Solutions of removing interference is complicated by the fact that there are several kinds of interference and each of them requires different approaches. Therefore a successful filter is necessary for removing the noise. Interface and Buttons for this system is given in Fig. 23.

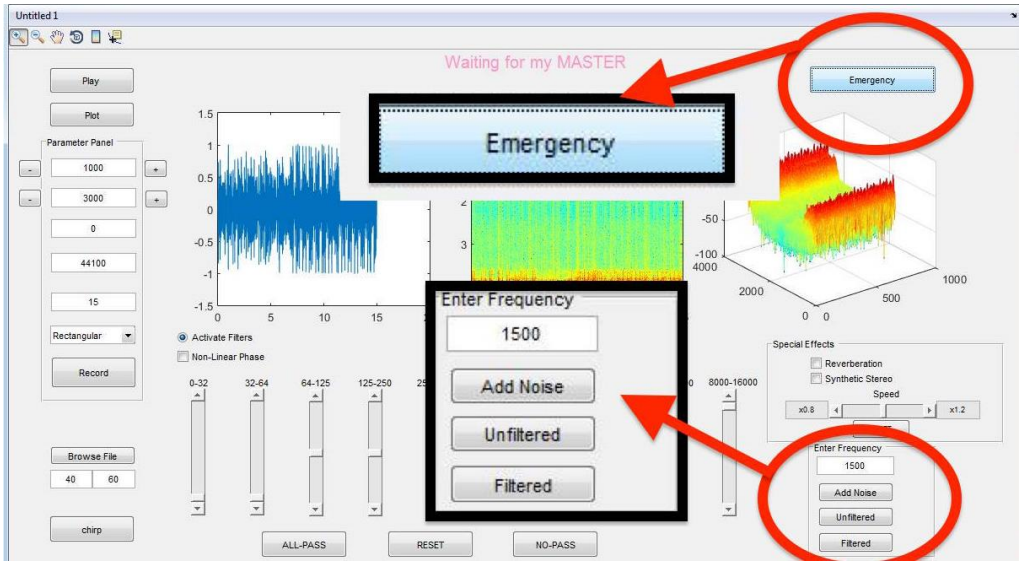

Figure 23

Enter Frequency: The panel where noise interfering and clearing can be done

Add Noise: Desired frequency can be added to original data with this button.

Unfiltered: This button will play the noisy data.

Filtered: This button will play the filtered, meaning the audio that is free from the noise.

Emergency: This extra button is built in case of an emergency. This button will stop the audio immediately.

7.2.Algorithm

Solution to the clearing of the noise (pure cosine) requires notch filtering. Frequency of the interfered noise signal should be estimated as well. We made use of the spectrogram of the noisy signal. When a pure cosine signal is added to the audio, it creates an almost single colored horizontal line at the frequency position of the noise signal. Therefore, by calculating the mean value of each row in spectrogram, we could actually estimate the

frequency at which noise exists. However, depending on the amplitude of the cosine signal, its mean value can go lower and it can be harder to estimate the true frequency. Therefore, by looking at the row in which mean value is high and also variance is low may be more convenient because pure cosine signal creates an unchanged amplitude value in the spectrogram. In our project, we actually just calculated the mean value and pick the highest one and it turned out to be satisfactory.

Pseudo Code
➢ Calculate the spectrogram matrix of noisy signal; ➢ Calculate the mean value of each row; ➢ Pick the highest one and extract the frequency in Hertz; ➢ Create numerator coefficients of notch filter who will suppress the above noise(FIR filter of order=1500 because no concern of timing); ➢ Apply filter; ➢ Listen;

7.3. Results

Results are satisfactory. Noise is added with a frequency of 1500 Hz. Successful hearing of filtered and unfiltered audio is achieved. Their spectrograms can also be seen in Fig.24 and Fig.25.

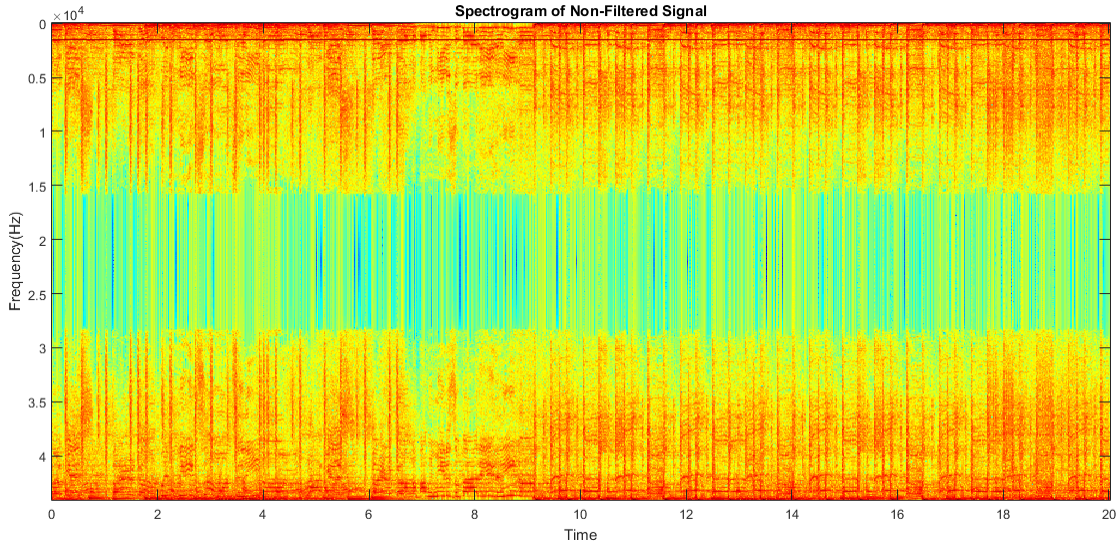

Figure 24

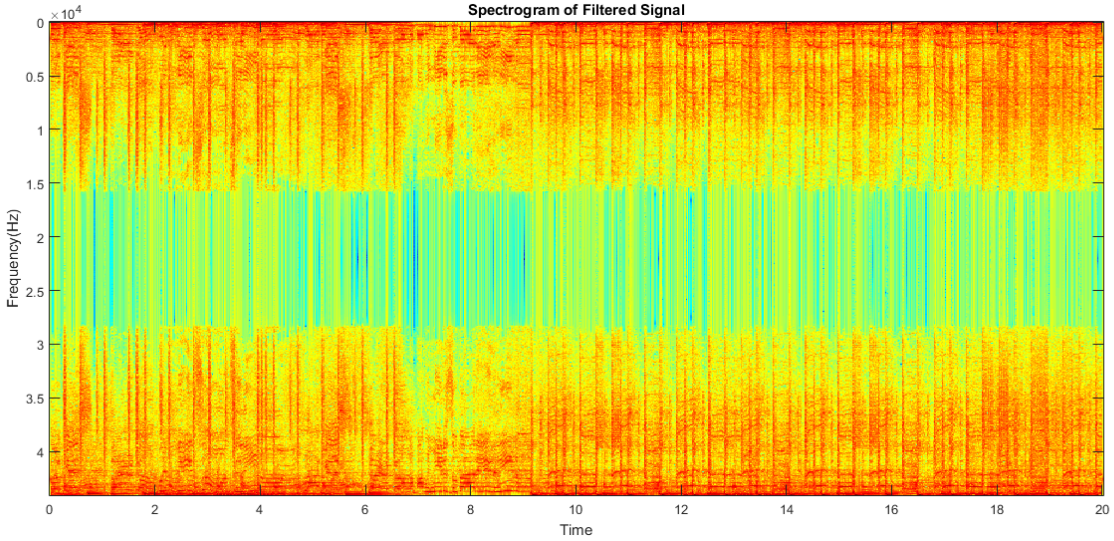

Figure 25

As it can be seen, in the spectrogram of noisy signal, there is a red horizontal line at 1500Hz. Frequency of the noise has successfully estimated and a notch filter is created whose cut off frequencies are [f-50,f+50] where f is estimated noise frequency. Notch interval is chosen as 50+50=100 in order not to filter out the original content of the song too much while suppressing the noise. Note that system is also able to suppress any frequency other than 1500Hz. This specific frequency is chosen just to demonstrate. Magnitude response of the notch filter can be seen from Fig. 26.

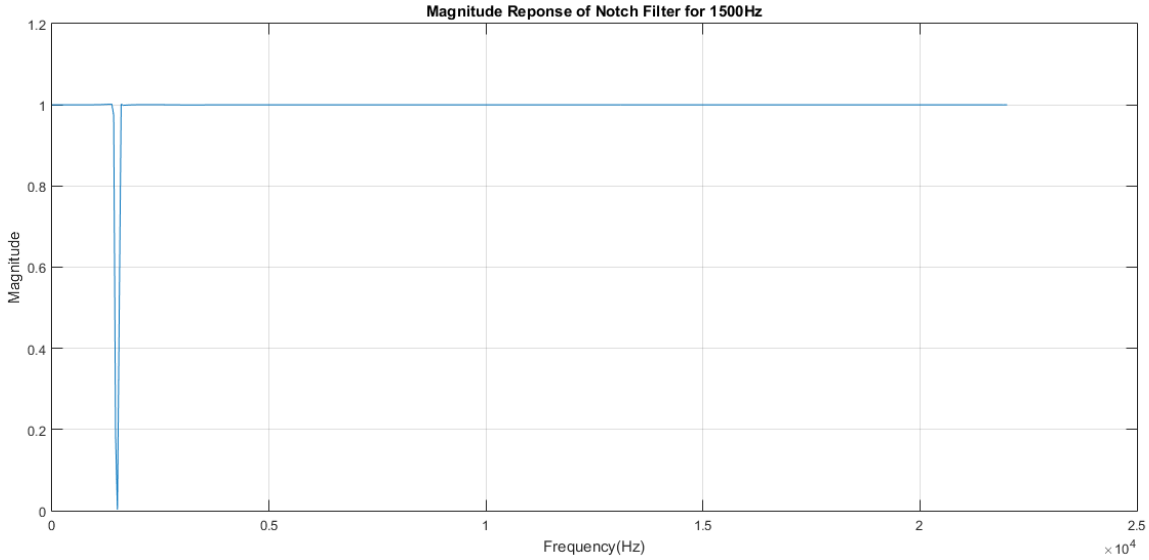

Figure 26

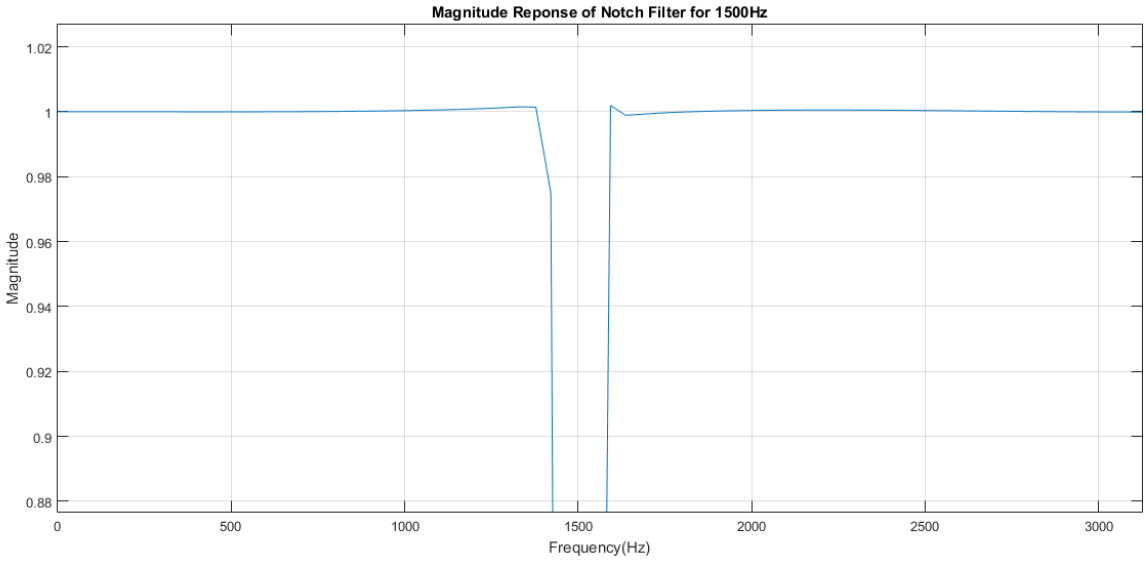

Figure 27

8. Conclusion

All in all, this project has been beneficial for many aspects. We had a chance to use our knowledge in a practical application. This project provided us to chance to learn many things about MATLAB, speech processing and use of various types of filters. We believed that we have been successful and get the desired outputs in general. Our requirement compliance can also be shown below.

Part	Requirement	Compliance
DATA ACQUISITION	System should capture an audio (or voice) playback from another device (e.g. mobile phone) by the help of a microphone connected to the user computer.	✓
	Analog-to-digital conversion sampling rate of this process should be adjusted from a user interface.	✓
	The captured audio input should be played on the speaker of the computer.	✓
	The system should also be able to process mp3 music files.	✓
SPECTROGRAM	System should determine and display the spectrogram of an audio file which is previously captured by a microphone.	✓
	System should be able to playback and plot the time waveform of the data, while showing the spectrogram.	✓
	The variables (such as window size, DFT size, overlap size,window type etc.) related to spectrogram should be parametric and therefore adjustable by the user.	✓

Yelda Gungor
Koray Cimen
25.1.2016

REAL TIME SPECTRUM	The system should be able to get recorded audio in frames while simultaneosly showing the DFT of the current frame in real time.	✓
	The variables such as frame size, DFT size, etc. should be parametric and therefore adjustable by the user. The window function (rect, hamming, etc.) should also be adjustable.	✓
EQUALIZER AND FILTERING	By the help of a user interface, system should be able to amplify or suppress the specific frequency bands of input data.	✓
	The input and output time waveforms, as well as spectrograms should be displayed.	✓
	Comparison of nonlinear and linear phase filter for 250-500Hz bandwidth.	—
SPECIAL EFFECTS	Reverberation	✓
	Synthetic stereo	✓
	Change (increase/decrease) the speed of the audio by an external factor.	✓

INTERFERING TONE REMOVAL	The system should be able to add a pure cosine signal to the recorded audio. The frequency of the cosine signal should be specified by user.	✓
	Design a filter that will suppress (or filter out) the generated noise.	✓
	The system should be able to play the noisy and the filtered audio.	✓
	Moreover the spectrograms of those signals can be shown on the screen.	✓
	The final filtered audio is expected to sound similar to the original one.	✓
	Propose and realize a method that will estimate the frequency of the above noise.	✓