Low Pass Filter, High Pass Filter and Delay using the MATLAB GUI

Shabuktagin Photon Khan, *Student Member, IEEE*

***Abstract*—In this paper, the MATLAB graphical user interface was used to load an audio data for signal processing. GUI user have the control to set the value of passband frequency in Hz for both low pass and high filter. Besides the filters, there is a delay option through which the user has the control over the amplitude as well as the delay time. The plot option allows the users to see the frequency response of the filters.**

***Index Terms*—Low pass filter, high pass filter, delay, MATLAB, Graphical user interface, frequency response, magnitude response, phase response, pole-zero plot, delay**

I. INTRODUCTION

USING MATLAB graphical user interface gives the freedom for the users to apply signal processing on an audio in a relaxed way. The graphical user interface will have filters and delay option which were built using the algorithm. The user have the option to load songs of '.*wav' or '.*mp3' format. The GUI will have the two separate sliders for delay magnitude and delay timing. Low pass filter as well as the high pass filter will have their separate sliders to control the passband frequency. Also, the user will have the access to see the magnitude response, phase response and the pole zero plot for both filters. In addition, the user can compare the low pass filter plots and the high pass filter plots. The user can see the discrete impulse response as well as unit step response. The paper goes through the summary of the algorithm behind it.

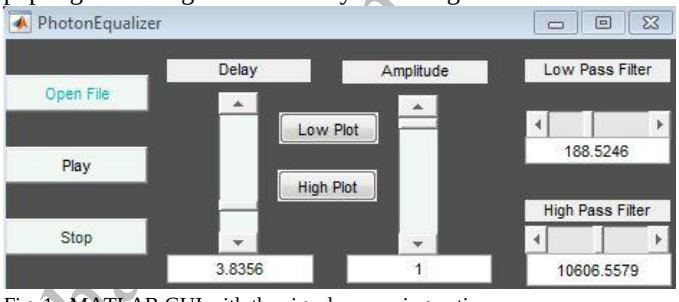

Fig. 1. MATLAB GUI with the signal processing options

Manuscript received December 3, 2015. The associate editor coordinating the review of this letter and approving it for publication was Dr. Xinrong Li. This was work has been primarily supported by Dr. Xinrong Li under the undergraduate program of Electrical Engineering department in part of Final Project of Digital Signal Processing course. This paper was presented as the final project of the digital signal processing course.

Shabuktagin Photon Khan is with the Electrical Engineering Department, University of North Texas, Denton, TX 76203 USA, (e-mail: shabuktaginkhan@my.unt.edu).

.

II. FLOW CHART

Before writing down the algorithm proper flow chart and pseudo code were made so that during the algorithm coding follows the proper logic and hierarchy order.

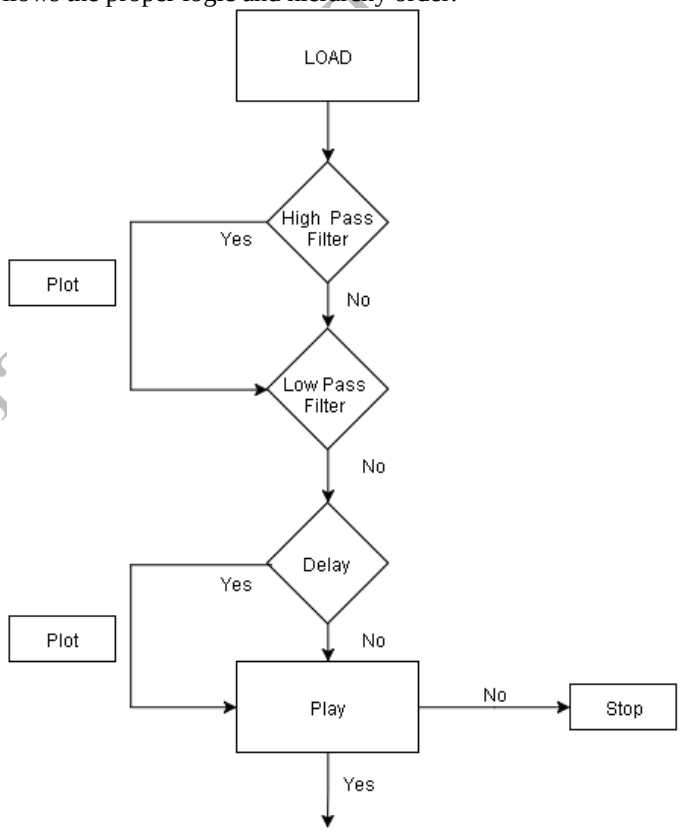

Fig. 2. Flow chart of the MATLAB algorithm

III. GRAPHICAL USER INTERFACE

A. Filters

Both the low pass filter and the High pass filter were made using the function called "designfilt" in MATLAB. All the parameters including stopband, filter order, design method, passband weight, stopband weight and the sample rate were set. The user has the option to control the passband. For the low pass finite impulse response filter, the stopband frequency was set to 600Hz. Therefore, the users have the option to set the passband from 0 to 500Hz. The least square was used for the design method. The passband weight was set to 5 and stopband weight was set to 0.1. The sampling rate was set to twice the sample frequency.

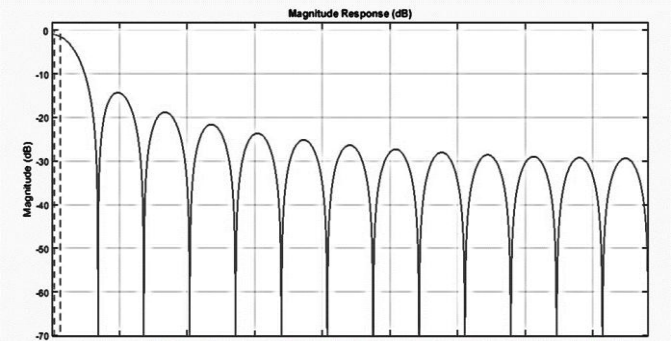

Fig. 3. Magnitude Response of the low pass filter at a certain passband

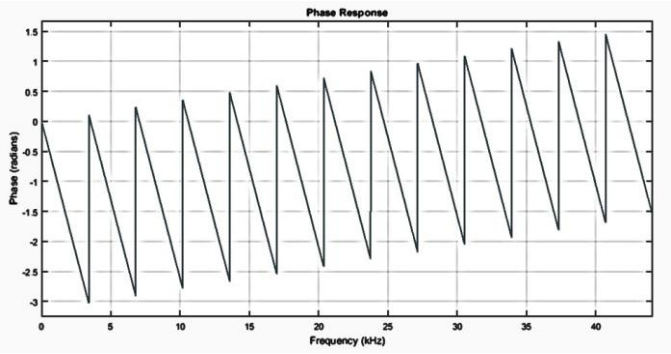

Fig. 4. Phase Response of the low pass filter at a certain passband

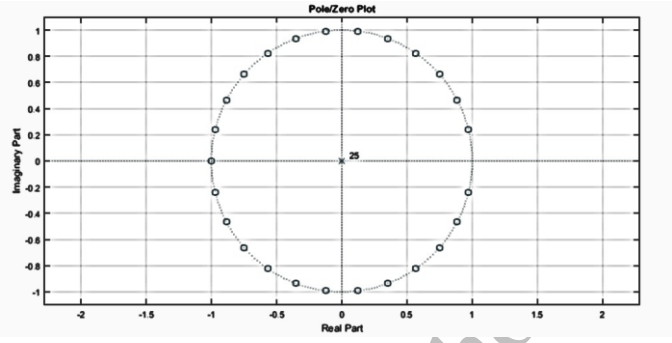

Fig. 5. Pole-Zero Plot of the low pass filter at a certain passband

"Fvtool" function was used to get the plots for magnitude response, phase response and the pole-zero plot. This function is better than using "freqz" function because it has more accessibility to watch the magnitude response and phase response together. It also shows the possible discrete unit impulse response. Similarly for the high pass filter, several plots were achieved using the "fvtool" and the parameters were also set. For the finite impulse response high pass filter, the filter order was 25 and stop band is set to 600. Therefore, the user have the access to set the pass band from 700Hz to 20000Hz. The passband weight was set to 5 and the stopband weight was set to 0.1. The sample rate is twice the sample frequency. The low pass frequency response plot can only be seen when the low pass frequency have the frequency more than 0 and the song needs to be played. Otherwise, it would give error. Similarly the frequency for the high pass filter needs to be more than 700Hz to make the filter work as well as for the plot. Even though the initial value for the high pass filter is set to 700Hz, the MATLAB act as if there is no filter at all. When the value of the high pass filter slider is more than 700Hz the audio file gets filtered. For the low pass filter there is 25 zeroes and there is one pole zero cancellation. For high pass filter there are 3 pole zero cancellation. They have zeroes in both inside and outside unity.

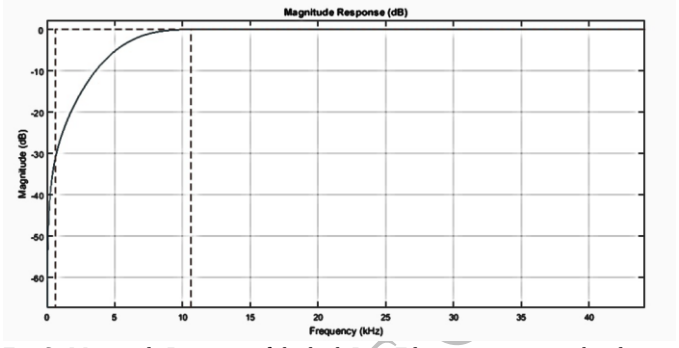

Fig. 6. Magnitude Response of the high Pass Filter at a certain passband

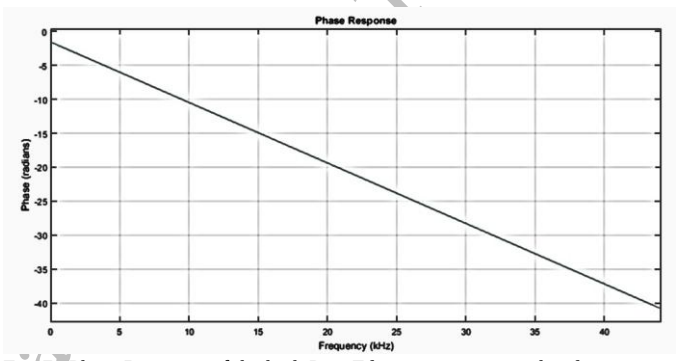

Fig. 7. Phase Response of the high Pass Filter at a certain passband

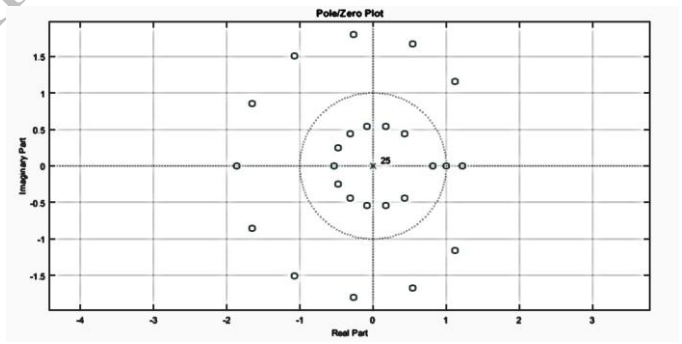

Fig. 8. Pole-zero Plot of the high Pass Filter at a certain passband

B. Delay

For the delay, there are two sliders. One is for the amplitude and other is for the time delay control. The amplitude slider is responsible for the volume of the delay with respect to the original audio file. However, the amplitude slider is just an arbitrary value with no units in it. Same case can be drawn for the time delay slider, the value mentioned in the slider is all arbitrary. This delay is stereo and it was done by adding delay on both sides using the algorithm as shown in [1]

$$x2(n,1) = x(n,1) + val1 * x(n-N,2); \quad (1)$$
$$x2(n,2) = x(n,2) + val1 * x(n-N,1);$$

IV. Finite Impulse Response Filter

For a causal discrete-time FIR filter of order N, each value of the output sequence is a weighted sum of the most recent input values:

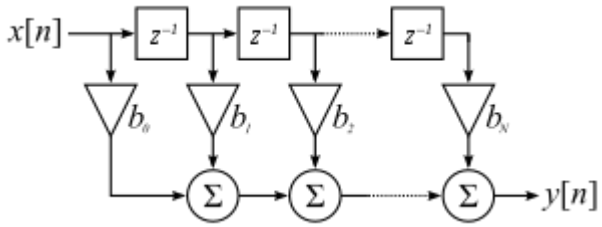

Fig. 9. A direct form discrete-time FIR filter of order N.

$$y[n] = b_0x[n] + b_1x[n-1] + \cdots + b_Nx[n-N]$$
$$\sum_{i=0}^{N} b_i.x[n-i] \quad (2)$$

The basic methods for designing FIR filters are window function, frequency sampling and Equiripple design or algorithmic design

$$H(z) = \sum_{n=0}^{N-1} h(n)z^{-n} \quad (3)$$

For linear phase

$$h(n) = h(N-1-n) \quad (4)$$

A. Equiripple and Least Square Method

The equiripple designs achieves optimality by distributing the deviation from the ideal response uniformly. Therefore, this reduces the maximum ripple or the deviation. The overall ripple in terms of energy is large. For the audio signal processing that is not desirable. Therefore the least square method provides the alternate solution of minimizing the energy at the stopband frequency. The attenuation for the stopband equirriple is lot greater than the least square method. The difference in the attenuation of the stopband frequency between equiripple and least square method is shown in Fig. 11.

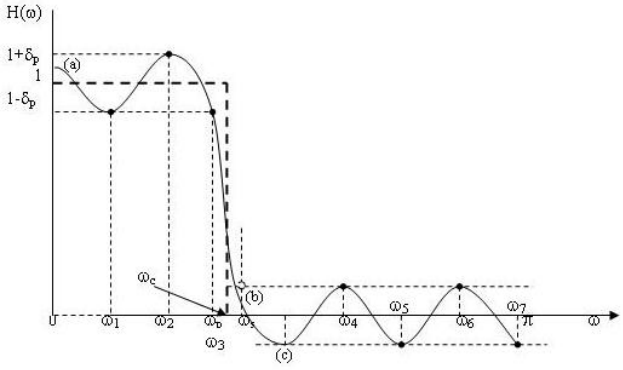

Fig. 10. Equiripple low pass filter

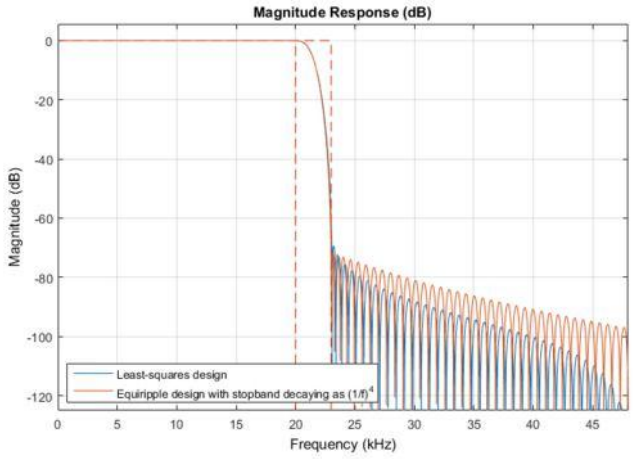

Fig. 11. Equiripple low pass filter vs least square low pass filter

Most of the FIR filters are not recursive and they are linear phase filters. The strategy to make the high pass filter is at first to make a filter in analog and then convert the idea from continuous to discrete domain for low pass filter. Using the low the pass filters concept the high pass filters are made.

B. Network Structure for FIR Systems

FIR systems do not have any poles in their systems. One of the disadvantage of the FIR system is that it requires poles at unit circle to cancel out zeros. Since the FIR system is linear phase we can exploit the symmetry property of coefficients.

$$H(z) = \sum_{n=0}^{N-1} h(n)z^{-n} \quad (5)$$

If we consider N even and see that half of the numbers are identical.

$$\sum_{n=0}^{\frac{N}{2}-1} h(n)z^{-n} + \sum_{n=\frac{N}{2}}^{N-1} h(n)z^{-n} \quad (6)$$
$$n = N-1+r$$
$$\sum_{r=0}^{\frac{N}{2}-1} h(N-1-r)z^{-n}$$
$$\sum_{n=0}^{\frac{N}{2}-1} h(n)z^{-n} + \sum_{n=0}^{\frac{N}{2}-1} h(n)z^{-(N-1-n)}$$
$$\sum_{n=0}^{\frac{N}{2}-1} h(n)z^{-n} + \sum_{n=0}^{\frac{N}{2}-1} h(n)z^{-(N-1-n)}$$
$$\sum_{n=0}^{\frac{N}{2}-1} h(n)[z^{-n} + z^{-(N-1-n)}]$$

Therefore, we can use (6) to reduce the number of multipliers.

In general FIR filters are canonical. One of the feature of the FIR filters is that it requires no feedback which means relative quantization error occurs in each calculation. All most all the FIR filters are stable. FIR filters always have linear phase also known as group delay. One of the cons of the FIR filter is the amount of power required by the processors to compute the algorithm. Therefore, FIR system requires more taps to reach the same performance as infinite impulse response (IIR) which means it requires more memory. Taps are the delays or coefficient pairs. The number of taps is often represent as N. Even though more the power more the flexibility to fine tune the response of the filter. These are problems for the system that has lower memory for example

small microcontrollers. FIR is also sometimes limited in resolution in low frequency. The filter design is favorable for optimization based designs, arbitrary magnitude and phase response. The basic difference between IIR filter and FIR filter is shown in Fig. 12.

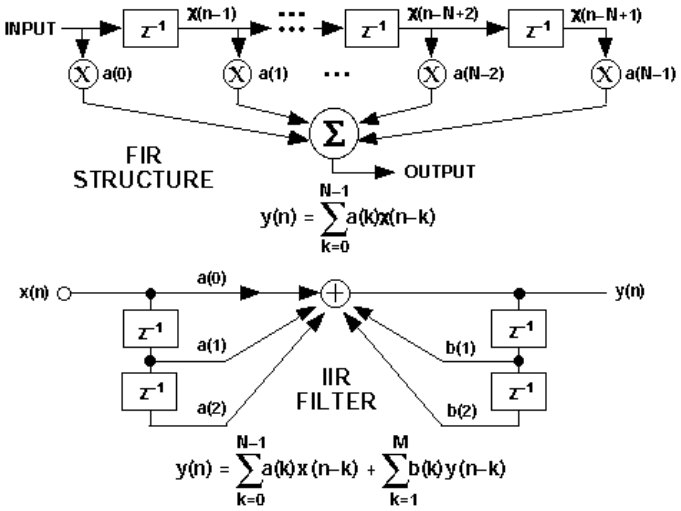

Fig. 12. Basic structure difference between FIR and IIR filter

V. Conclusion

The MATLAB graphical user interface provide an ease playground to work on the signal processing. For the time being, the GUI is not a real time signal processor. So, when the delay or the filter slider are moved the delayed or the filtered signal can be heard when the play button is pressed. Therefore, after changing the settings in the filters and delays the play button needs to be pressed. This project can be further improved to make the delay or filter be real time. Due to the time constraint, the project could not go to real time complexities. During this short duration of time, the idea of framing were achieved. In brief, to achieve real time, the strategy is to work with buffer and frames. The idea of frames was to save the audio into split of 0.1 frames in a multidimensional array and make the audio play frame by frame. Since, the playing of the frame by frame audio requires a loop, the delay or the filter function can be called, each and every time it passes the loop. The problem regarding playing in the loop is that there is some minor glitches when the audio plays and shifts from one from to another. It is due the algorithm execution time. This small glitches can be removed using several more lines of codes. Therefore, this small project in summary can load, play, and stop an audio. Also, it can add delays, low pass filter and high pass filter. MATLAB is one of the most popular computation program which allows use to use simple graphics capabilities. The plots are obtained very easily using handles and objects. Additional codes were required for making the sliders, the static and interactive textbox work dynamically. However, this paper doesn't discuss about those additional codes.

References

[1] A. V. Oppenheim and R. W. Schafer, Digital signal processing. Englewood Cliffs, N.J.: Prentice-Hall, 1975.

[2] J. G. Proakis and D. G. Manolakis, *Digital signal processing*, 4th ed. Upper Saddle River, N.J.: Pearson Prentice Hall, 2007.

[3] "Documentation," *Designing Low Pass FIR Filters*. [Online]. Available at: http://www.mathworks.com/help/dsp/examples/designing-low-pass-fir-filters.html. [Accessed: 2015].

[4] "Documentation," *Design digital filters*. [Online]. Available at: http://www.mathworks.com/help/signal/ref/designfilt.html. [Accessed: 2015].

[5] "draw.io - free flow chart maker and diagrams online," *Flow Chart Maker & Online Diagram Software*. [Online]. Available at: https://www.draw.io/. [Accessed: 2015].

[6] "flute.wav," *Sound Processing in MATLAB*. [Online]. Available at: http://homepages.udayton.edu/~hardierc/ece203/sound.htm. [Accessed: 2015].

[7] "Digital Audio Equalizer - File Exchange - MATLAB Central," *Digital Audio Equalizer*. [Online]. Available at: http://www.mathworks.com/matlabcentral/fileexchange/23982-digital-audio-equalizer. [Accessed: 2015].

[8] [Online]. Available at: http://cnx.org/resources/3c3c08691e0e0bbe0c35b6e8ad5aed771ada9751/5.27.jpg. [Accessed: 2015].

[9] "Discrete-Time Signal Processing," *MIT OpenCourseWare*. [Online]. Available at: http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-341-discrete-time-signal-processing-fall-2005/. [Accessed: 2015].

[10] "Signals and Systems," *MIT OpenCourseWare*. [Online]. Available at: http://ocw.mit.edu/resources/res-6-007-signals-and-systems-spring-2011/. [Accessed: 2015].

[11] Okyere Attia, J., "Teaching AC circuit analysis with MATLAB," in Frontiers in Education Conference, 1995. Proceedings., 1995 , vol.1, no., pp.2c6.9-2c612 vol.1, 1-4 Nov 1995 doi: 10.1109/FIE.1995.483086

[12] "Matlab Signal Analysis - frame by frame analysis of a signal - silence removal audio example.avi," *YouTube*. [Online]. Available at: https://www.youtube.com/watch?v=wpxtsrpaloa. [Accessed: 2015].

[13] A. V. Oppenheim and A. S. Willsky, *Signals & systems*, 2nd ed. Upper Saddle River, N.J.: Prentice Hall, 1997.

[14] "Martin Finke's Blog," *Making Audio Plugins* -. [Online]. Available at: http://www.martin-finke.de/blog/tags/making_audio_plugins.html. [Accessed: 2015].

[15] "Wikipedia," *Wikipedia*. [Online]. Available at: https://en.wikipedia.org/wiki/finite_impulse_response. [Accessed: 2015].

[16] "FIR vs IIR filtering," *FIR vs IIR filtering*. [Online]. Available at: https://www.minidsp.com/applications/dsp-basics/fir-vs-iir-filtering. [Accessed: 2015].

[17] "Overview of FIR and IIR Filters," *YouTube*. [Online]. Available at: https://www.youtube.com/watch?v=9ynqbwkrss4. [Accessed: 2015].

[18] Azemi, A., "Utilizing MATLAB in undergraduate electric circuit's courses," in Frontiers in Education Conference, 1996. FIE '96. 26th Annual Conference., Proceedings of , vol.2, no., pp.599-602 vol.2, 6-9 Nov 1996

[19] "FIR Filter Basics," *FIR Filter Basics*. [Online]. Available at: http://dspguru.com/dsp/faqs/fir/basics. [Accessed: 2015].

[20] "Why Use DSP?" *Analog Devices: Analog Dialogue: Digital Signal Processing 101 An introductory course in DSP system design* [Online]. Available at http://www.analog.com/library/analogdialogue/archives/31-1/dsp.html. [Accessed: 2015].

[21] Green, R.A., "Getting a handle on MATLAB graphics," in Potentials, IEEE , vol.26, no.4, pp.31-37, July-Aug. 2007 doi: 10.1109/MP.2007.4280330