International Journal of Multidisciplinary Engineering in Current Research - IJMEC
Volume 8, Issue 11, November-2023, http://ijmec.com/, ISSN: 2456-4265

REAL TIME AUDIO EQUALIZER USING DIGITAL SIGNAL PROCESSING TECHNIQUES

Dr. Bremiga Gopalan[1], Firdous Fatima[2], Vadithya Madhuri[3]

[1]Assistant Professor, Department of ECE, Bhoj Reddy Engineering College for Women, Hyderabad, India
[2,3]B.Tech Students, Department of ECE, Bhoj Reddy Engineering College for Women, Hyderabad, India

Abstract: This paper outlines the design of a high-precision graphic audio equalizer with digital filters in parallel, along with its implementation in MATLAB App Designer. The equalizer is comprised of bands separated with a one-third octave frequency ratio, and its frequency response is controlled by 63 filters. Furthermore, the application can process audio signals, in real time, recorded by microphone and from audio files. While processing, it displays an FFT plot of the output sound, also in real time, equipped with a knob by which the refreshing pace can be adjusted. The actual frequency response proved to match the desired one accurately, but the matching is computationally demanding for the computer. An even higher accuracy would entail a computational complexity beyond the power of ordinary computers, and was thus concluded to be inappropriate. As a result, the final application manages to provide most laptops with both high precision and proper functionality.

I. INTRODUCTION

Audio equalization regulates electrical signal frequencies using an equalizer. This tool reduces noise and reverb in auditoriums and music recordings. Many people associate equalizers with their music industry use.

Most sound recorders and reproducers use graphic equalizers, which are manually adjustable volume sliders that correlate to frequencies. The listener may shape sound using such a tool. Bass is emphasized by increasing a signal's loudness. Cutting the upper end of the spectrum lowers the amplitude of those frequencies, enhancing this impact. Reversing this method magnifies the treble.

The music business uses increasingly advanced and versatile equalizers to get the best sound quality. The manufacturer may pick an appropriate equalization setting by knowing the operating conditions. At a club, the mixer may wish to boost the bass, whereas at an auditorium lecture, the technicians may boost the high frequencies. There are two separate audio equalizer classifications. They may be divided into parametric and graphic equalizers or digital and analog ones. Digital and analog relate to the equalizer's signal type. The parametric-graphic division depends on how well the bands can be manipulated.

The parametric equalizer offers gain, center frequency, and bandwidth adjustments, whereas the graphic equalizer simply permits boosts. This gives the former capacity and flexibility over the latter [1]. However, the graphic equalizer is more user-friendly. Non-professionals find the parametric equalizer difficult. This is partially because simplicity and high-detail control are trade-offs.

Loudspeakers and their surroundings effect audio. Changing the surroundings, listener position, or speakers changes the system's frequency response. This idea inspires acoustic room correction research. A room's acoustics were often ignored until the 1950s [1]. Room correction techniques developed in the late 1960s, and massive filters were specifically made for big venues to improve acoustics. Many high-end home entertainment systems include room correction. These pricey systems are for serious audiophiles and professionals. This project intends to create a digital signal processor-based, automated audio equalization that adjusts for speaker and room aberrations in any situation. The technology generates test tones that a microphone at the listener records. Psychoacoustics is briefly discussed after characterizing the recorded tones to acquire a room and speaker response and creating a filter that normalizes an audio stream. Psychoacoustics, the science of sound perception, is applied in this project to determine the appropriate frequency response. A brief test was given to numerous various-aged volunteers who sampled music under different filter settings.

When building an equalizer, accuracy and computing speed are also important trade-offs. A high-order filter is needed to match the target response (desired frequency response) to the actual frequency response. The more complex the filter, the more calculations it requires. The filter changes when the goal response, or volume, changes at one or more frequency points. Only while doing so are these computations performed.

II. DESIGN

Initial Considerations

Several factors were considered while building this project's filter algorithm. The filter algorithm must be tuned for audio applications and testing hardware. Minimum requirements include CD-quality audio and a filter that covers the whole hearing spectrum. The filter working range was restricted to 30Hz to 22050Hz due to the test equipment's 30Hz frequency response.After reviewing the literature, the Audio Team decided a finite impulse response filter (FIR) would best suit the application. Table 1 lists the many advantages of FIR filters over IIR filters.

Filter structure

The transfer function of the resulting filter is given by

$$H(z) = c_0 + \sum_{k=1}^{K} \frac{b_{k,0} + b_{k,1}z^{-1}}{1 + a_{k,1}z^{-1} + a_{k,2}z^{-2}}$$

where K decides the number of second order filters arranged in parallel and c0 represents the direct path gain. This is illustrated in a block diagram in figure where X(z) denotes the input signal and Y (z) denotes the filtered output signal.

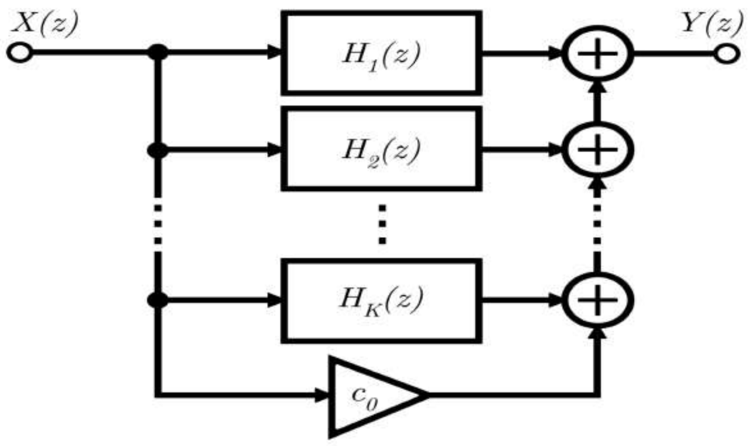


Figure 2: Illustration of the parallel filter structure

Derivation of the denominators

To begin filter design, the pole frequencies must be determined. Logarithmically, these frequencies were uniformly distributed. It seems fair to choose pole radii jpkj such that two nearby filters' transfer functions cross at 3dB drop [6]. This is obtained by

$$\theta_k = \frac{2\pi f_k}{f_s}, \qquad k = 1, 2, ..., K$$

$$|p_k| = e^{\frac{-\Delta\theta_k}{2}}$$

$$\Delta\theta_k = \frac{\theta_{k+1} - \theta_{k-1}}{2}, \qquad k = 2, 3, \dots, K-1$$

with the necessary exceptions for the two filters at the upper and lower edge:

$$\Delta\theta_1 = \theta_2 - \theta_1,$$
$$\Delta\theta_K = \theta_K - \theta_{K-1}.$$

The coefficients in the denominator of each transfer function are given by

Derivation of the numerators

When the parameters have been set, the problem reduces to a linear system which is a matrix representation of equation

$$\mathbf{h} = \mathbf{M}\mathbf{b},$$

where M is a matrix comprised of the denominators of the filter sections and their delayed counterparts, given by

$$\mathbf{M} = \begin{pmatrix} \frac{1}{den(1,1)} & \frac{e^{-j\omega_1}}{den(1,1)} & \cdots & \frac{1}{den(1,K)} & \frac{e^{-j\omega_1}}{den(1,K)} & 1 \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ \frac{1}{den(N,1)} & \frac{e^{-j\omega_N}}{den(N,1)} & \cdots & \frac{1}{den(N,K)} & \frac{e^{-j\omega_N}}{den(N,K)} & 1 \end{pmatrix}$$

where den(n; k) denote the denominators of the filter sections: 1+ak;1e-j!n +ak;2e-j2!n. The last column is filled with ones due to multiplication by the direct path gain. b is a column vector with the free numerator coefficients and the direct path gain c0. Hence, as the numerators is multiplied by their corresponding denominators, h becomes the vector with the resulting frequency response.

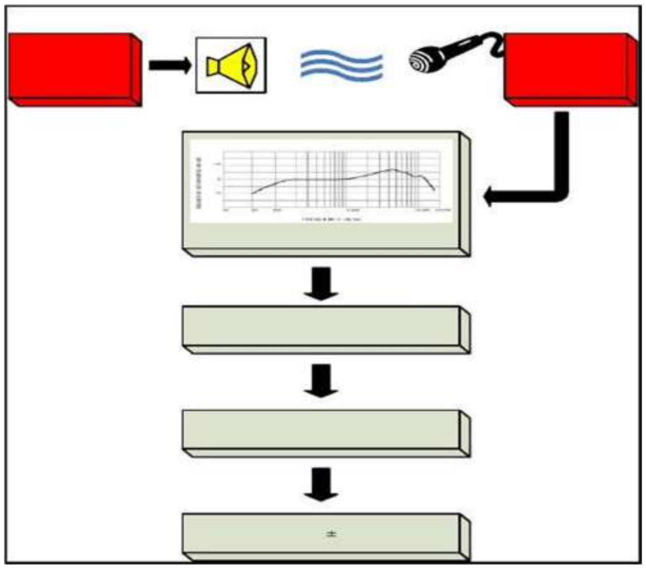


Figure 2. The steps of filter calibration/creation.

In order to process an audio stream, the system does the following, which is also illustrated in Figure 5.1.

1. Buffer 4096 samples.2.
2. Apply the Hanning window.3.
3. FFT the buffer.4.
4. Multiply the signal spectra with that of the filter.5.
5. IFFT the filtered data.6.
6. Sum the overlapping section of this and the previous buffer that was worked on.7.
7. Store the current buffer to be summed with the overlapping section from the next buffer.8.
8. Play the summed section out to the receiver.

This entire process is performed twice every 93ms - once for each channel in the stereo audio stream.

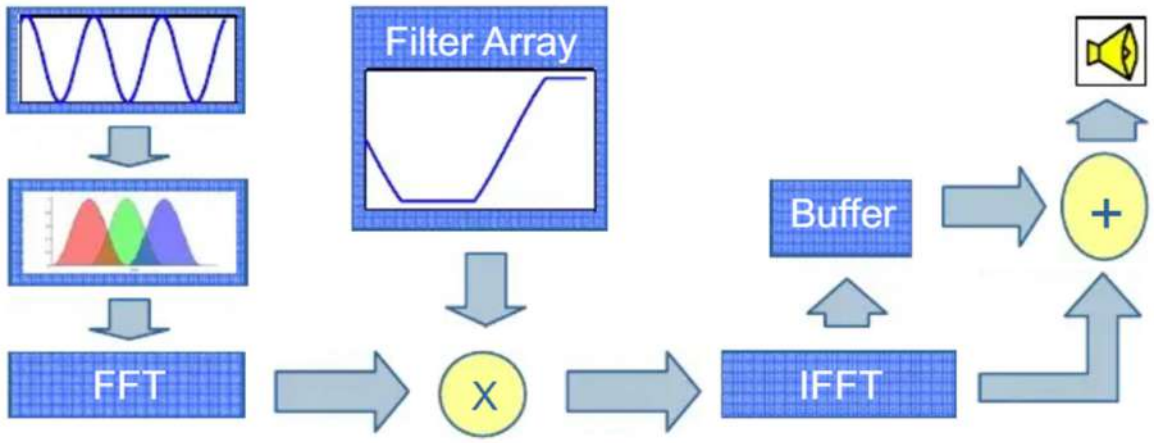


Figure 5. Process of applying filter to the audio stream

DOI: https://doi-ds.org/doilink/11.2023-75724699

III. IMPLEMENTATION

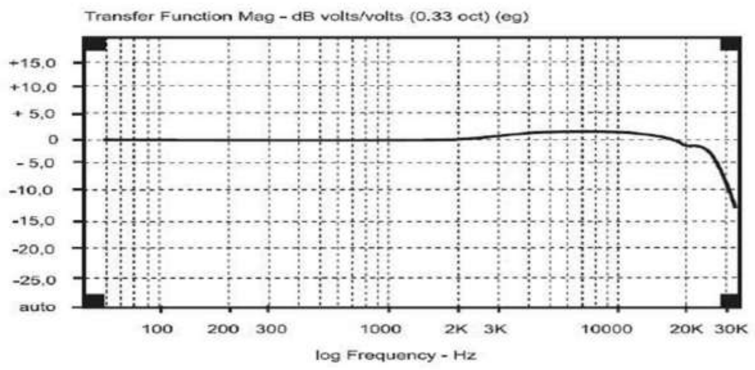
Implementation Overview

As the title indicates, this chapter describes equalizer implementation. MATLAB App Designer with Audio Toolbox 2.0 was used to write the code. The Digital Audio Equalizer needed a filter implementation with filter calibration, application, and control software on a Digital Signal Processor and a host PC graphical user interface for DSP interface. DSP was initially designed to function automatically without user involvement. Adding functionality to the design revealed the need for a user interface.

Hardware Overview

This system uses two hardware parts. The filter software is implemented on a Texas Instruments DSP development board. Digital Signal Processors that serve this application are usually packaged in Ball Grid Arrays or other dense packaging, hence the commercial board was selected over in-house fabrication. Commercially made development boards must be obtained to make these components. Digital signal processor development software costs $1,000 or more as standalone software. A minimal “student” programming environment was supplied with the development board to keep deployment costs low. I bought a TIDSK6713 development board. This board has an audio codec, microphone, Line In, Line Out connectors, USB interface, software development tools, and other hardware for audio engineering. Table 3 lists significant DSP specs. Digital Signal Processor Key Features:225MHz

- 192kb SRAM
- 8 Instructions / cycle
- 1800 MIPS / 1350 MFLOPS
- Optimized for audio



The Behringer ECM8000 frequency response. Note that over the range of human hearing, the response does not deviate from the flatline by more than 3db

Filter Implementation

For two main reasons, the set-top Digital Signal Processor board runs all filter calibration, application, and control software: After calibration, the device should function without the host PC and apply and control filters quickly. C was used for all DSP board software. The interrupt service routine (ISR) and main route loop are DSP execution pathways. ISR outputs and buffers/records all audio. The primary route coordinates filter calibration and buffered audio filtering.

Interrupt Service Routine

The ISR is called automatically by the audio codec when new data is ready to be serviced (and when new data is requested to be played, although these happen concurrently). Given the sample rate of 44.1 kHz, the ISR occurs approximately every 23us. Below Figure shows the ISR flow diagram.

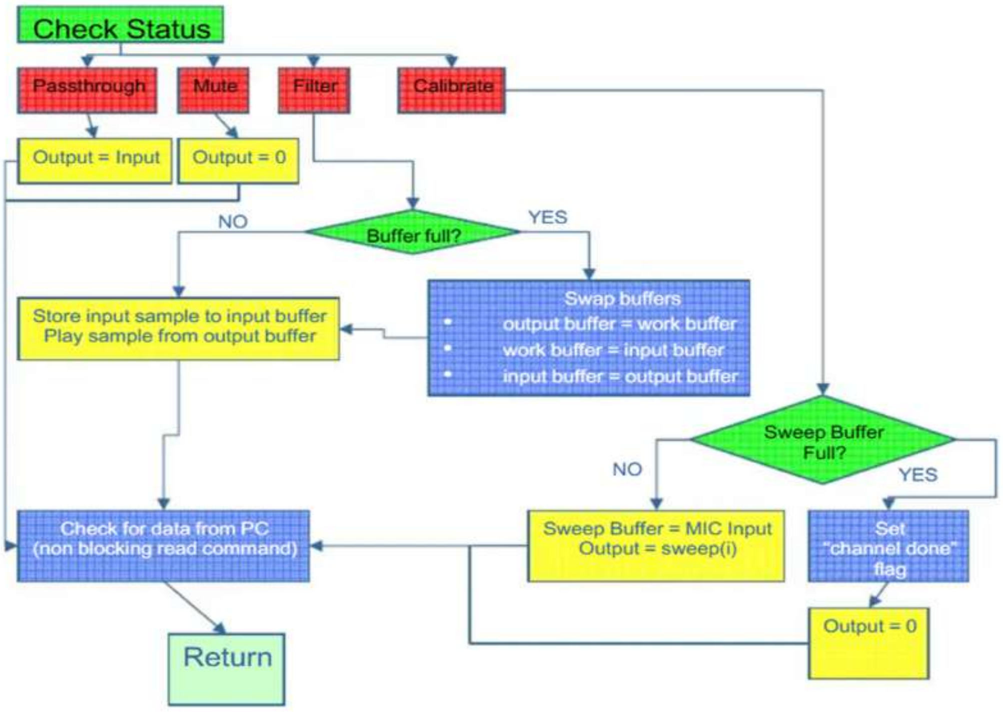


Figure Flow diagram of the Interrupt Service Routine on the DSP board

We examine board status flags at each ISR input. This flag selects one of four board output modes: Passthrough, Mute, Filter, or Calibrate. The ISR sends the audio codec's current input sample to the receiver in passthrough mode. Nothing is filtered. ISR sends nothing to the receiver in quiet mode. Nothing is filtered. In Filter mode, the ISR controls three data buffers for input, output, and filtering. Input audio samples are stored in input buffers and output samples are played to the receiver at each ISR entry. The ISR switches buffers when the input buffer is full. The primary route of execution filters the work buffer after swapping the input buffer. For playback, the filtered work buffer is swapped into the output buffer. Finally, the faulty output buffer is recycled to the input buffer for fresh data. The ISR plays a logarithmic sweep sample (30Hz to 22050Hz) in calibrate mode. This sweep has 32768 samples (743ms). After playing the sweep, the ISR saves a microphone input sample in a huge buffer for further processing. After the sweep, the ISR flags “channel done”. The primary route of execution uses this flag to facilitate the following calibration step, explained below. A read command is delivered to request a new mode flag from the user interface before the ISR returns to the main route. The DSP's real-time operating system schedules the non-blocking command to handle incoming data. The mode flag is changed if data exists.

Main Path of Execution

Figure 8 depicts the primary execution route simplified. The primary execution route loops forever, monitoring for filter mode modifications or ISR filter requests. At program start, numerous initialization operations are done, including initializing all filter arrays to all pass filters, building the FFT routine's digit reverse table, activating the ISR and other global interrupts, RTDX/JTAG, and initial status mode. The software enters an indefinite loop after setup and branches at one of four status modes.

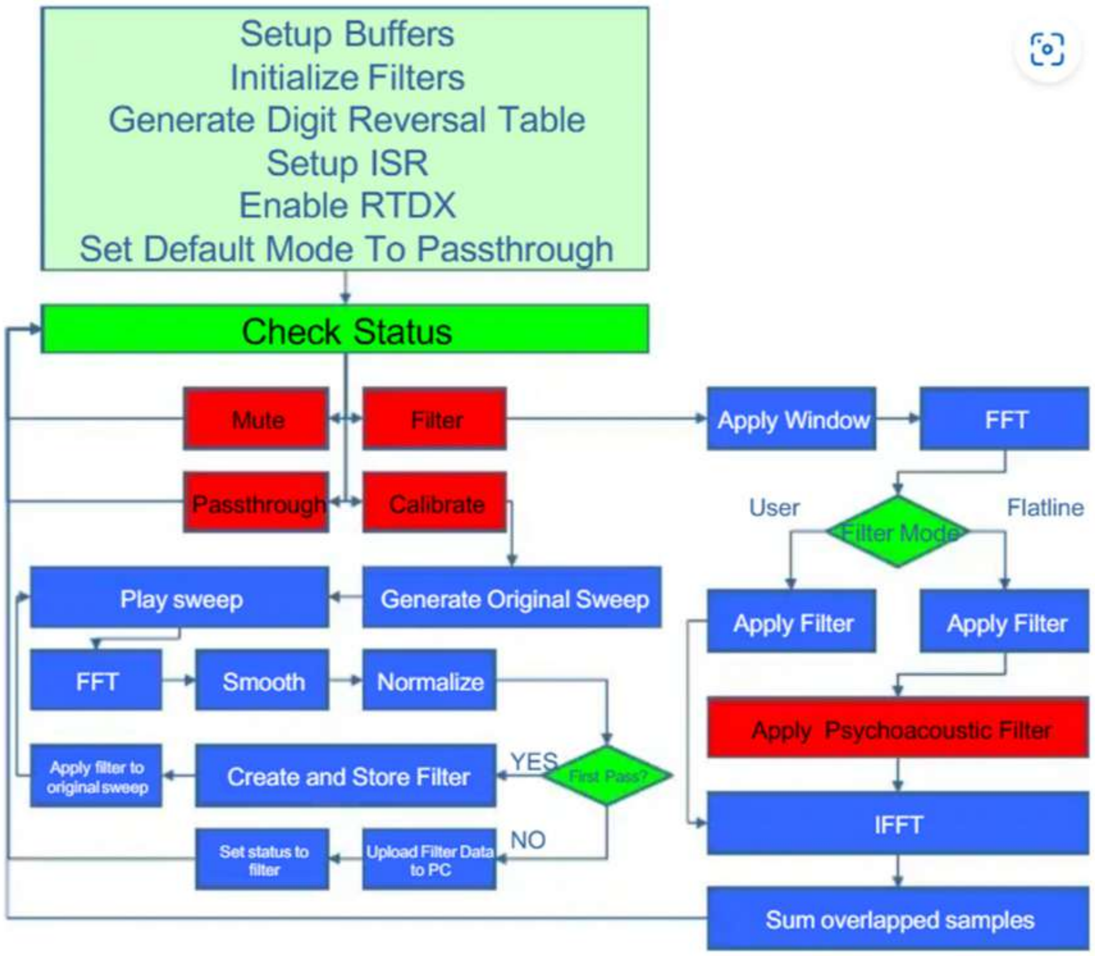
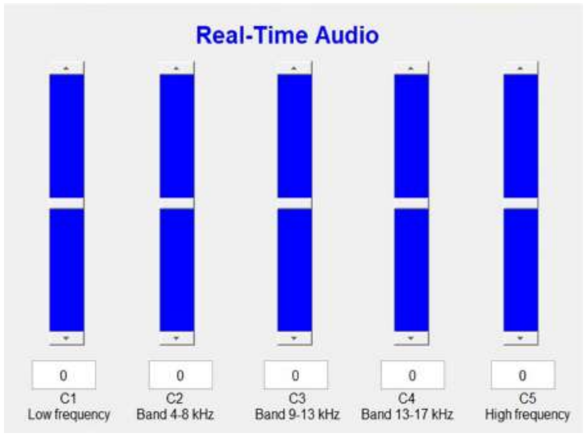


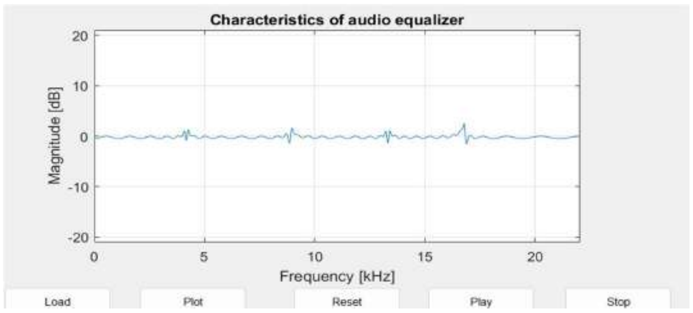
Figure Main path of execution flow diagram

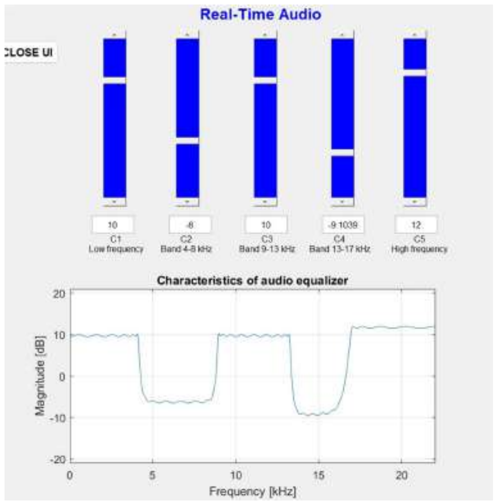
For both the mute and passthrough status modes, nothing is done within the main path, and a simple busy wait occurs. If the status mode is set to filter, the main path first checks if the work buffer (described in 3.3.1) has been flagged for filtering. If so, the filter operation occurs as detailed in section 2.4. After filtering, the block flag is cleared and a busy wait occurs until either the status mode changes or a new work buffer is prepared. If the status mode is set to calibrate, the main path coordinates calibration with the ISR. First, by generating a logarithmic sweep, setting output to the left channel only, and clearing the “channel done” flag, the main path waits for the ISR to play and record the sweep on the left channel and set the “channel done” flag. Once set, the main path sets the channel output to the right channel and clears the flag again. After the ISR sets the flag, both the left and right channel responses can be calculated from the recordings as discussed. After filter creation, the original sweep is played and recorded again on the left and right channels through the filter. These new recordings can be used to evaluate the effectiveness of the filter and are uploaded to the PC for examination by the user. After uploading to the PC, the status mode is changed to filter.

IV. SIMULATION & RESULTS

OUTPUT







DOI: https://doi-ds.org/doilink/11.2023-75724699

Audio signals are analyzed and manipulated to create frequency responses. FIR and IIR filters are two algorithms you may try. FFT is another option for spectrum analysis. For real-time applications, latency and processing efficiency matter.

V. CONCLUSION

This article shows a 31-band visual equalization with an easy-to-use interface. Download MathWorks equalizer for free [8]. Free 31-band equalizers with high precision are uncommon. Volume slider presets included in the program. Users may instantly change the track's genre frequency response with one touch. Its high frequency resolution and accuracy make the equalizer perfect for testing. Adjust a slider and listen to the result. You may also modify the sliders. The equalization improves any genre or style with its manually adjustable frequency magnitude responsiveness. Graphic equalizers with parallel filters are discussed here. The total of filters determines frequency response, while frequency-specific sliders govern magnitude response. Parametric equalizers provide manual control of additional parameters but need expertise. Changing frequency response is faster but harder than graphic ones. Instead of parallel filters, cascaded filters provide the frequency response. Different equalization and filter topologies have strengths and downsides. This paper equalizes to attract more buyers. For instance, simplicity has been prioritized above detail control and target reaction time.

A multipurpose audio equalizer controls electrical signals. One component of music reproduction may be advanced. Not just music reproduction, the utility corrects telephone line frequency responses [1]. Read this whole paragraph to comprehend equalizer creation. Summarizing the process structure is possible. The equalizer is filter-based. The structure was created by matrixing all filter section denominators. Then, a vector of numerators was multiplied by the denominator matrix to achieve the desired frequency response. Interpolated slider discrete function frequency response is needed. Sliders control sound and filters. Humans interpret "audio" as sound. Equalizers affect sound. This virtue clarifies signal frequency and strength. Normal hearing allows frequency discrimination. High frequency hurts, low deep. A trained ear can detect even a little tone variation. We want equalizers with high frequency resolution and the ability to experiment with settings and frequency response since we are sensitive to frequencies. This audio adjuster features 31 bands, which is a lot. The only variable in audio equalizers is volume, which measures sound strength. In bass mode, low frequencies are louder and vice versa. Setting weakens high tones, not decreases them.

REFERENCES

[1] Vesa V□alim□aki, Joshua D. Reiss, All About Audio Equalization: Solutions and Frontier, Department of Signal Processing and Acoustics, Aalto University, May 2016.

[2] Ste_ Knorn, Signals and Systems, Signals and Systems at department of engineering sciences at Uppsala University, 2018.

[3] Julius O. Smith III, Introduction to Digital Filters with Audio Applications, Center for Computer Research in Music and Acoustics (CCRMA), September 2007 edition.

[4] Alan V. Oppenheim and Ronald W. Schafer, Discrete-Time Signal Processing - second edition, Prentice-Hall, 1999.

[5] Sample- and Frame-Based Concepts, https://www.mathworks.com/help/dsp/ug/sampleand-frame-based-concepts.html. Accessed: 2019-05-07.

[6] Ganvi Ranjitha , Sheguri Kushma, Boyala Bhavani, Classification And Detection Of Hyperspectral Images, International Journal of Multidisciplinary Engineering in Current Research - IJMEC Volume 8, Issue 4, April-2023, http://ijmec.com/, ISSN: 2456-4265.

[7] Mohammed Riyaan, Mohammed Saif, Syed Mohtashim Ahmed, Khutaija Abid, Employing Machine Learning Algorithm To Decipher And Forecast Subjective Well Being, International Journal of Multidisciplinary Engineering in Current Research - IJMEC Volume 8, Issue 4, April-2023, http://ijmec.com/, ISSN: 2456-4265.

[8] Chandra Singh, Ch. Nivas, G. Pranay Kumar, V. Manoj Kumar, A Hybrid Deep Transfer Learning Model With Machine Learning Methods For Face Mask Detection In The Era Of The Covid-19 Pandemic, International Journal of Multidisciplinary Engineering in Current Research - IJMEC Volume 8, Issue 4, April-2023, http://ijmec.com/, ISSN: 2456-4265.

[9] J. Ramo, V. Valimaki, and B. Bank. High-Precision Parallel Graphic Equalizer. IEEE/ACM Transactions on Audio, Speech and Language Processing, Vol. 22, No. 12, pp. 1894-1904, December 2014. DOI: 10.1109/TASLP.2014.2354241

[10] Balazs Bank, Direct design of parallel second-order filters for instrument body modeling, Laboratory of Acoustics and Audio Signal Processing at Helsinki University of Technology, 2007.

[11] Johannes Langelaar (2019). One third octave graphic equalizer (https://www.mathworks.com/matlabcentral/_leexchange/71618-onethird-octave-graphic-equalizer), MATLAB Central File Exchange. Retrieved May 23, 2019.

DOI: https://doi-ds.org/doilink/11.2023-75724699