Date of completion 12 16, 2023.

Implement Normalized LMS Filter for Noise Cancellation Using Raspberry Pi

Yuanpeng Wang [1]
[1]Electrical and Computer Engineering department, Western University, CA

ABSTRACT This paper reports a novel design method in MATLAB Simulink that uses the Normalized LMS algorithm with Raspberry Pi 4B for Noise Cancellation. It is found that the design method has a significant noise reduction effect for both simulation sources and Raspberry Pi real-time audio input.

INDEX TERMS Raspberry Pi 4B, MATLAB Simulink, MATLAB, Noise Cancellation, Normalized LMS Adaptive Filter, Audio Processing.

I. INTRODUCTION

The Normalized Least-Mean-Square (NLMS) algorithm has been extensively used for various practical applications. The Normalized Least-Mean-Square (NLMS) algorithm is the normalized version of the Least-Mean-Square (LMS) algorithm. When $u(n)$ is large, the LMS algorithm will have a gradient noise amplification problem. Therefore, we use NLMS with the tap-weight vector at adaption cycle n+1[1]. NLMS is comprehensively used in communications systems for echo cancellation. In audio communication and processing, echoes can be a significant problem, especially in long-distance or internet-based calls. The NLMS algorithm helps in identifying and canceling acoustic echoes from the loudspeaker, improving the overall clarity and quality of the voice transmission[2]. In active noise control, the NLMS algorithm is implemented to reduce noise. This is particularly useful in environments with variable noise characteristics, such as telecommunications or hearing aids, where it can adaptively filter out background noise while preserving the primary speech signal[3].

Raspberry Pi is a series of small single-board computers (SBCs) developed in the United Kingdom by the Raspberry Pi Foundation in association with Broadcom. The Raspberry Pi 4 Model B is a significant upgrade over its predecessors. It has an ARM v64 Cortex processor and standard 40-pin GPIO header This makes it ideal for various projects, from scientific research to home automation[4].

Simulink, developed by MathWorks, is a block diagram environment for modeling, simulating, and analyzing multidomain dynamical systems. It supports system-level design, simulation, automatic code generation, and continuous testing and verification of embedded systems. It provides an interactive graphical interface and a customizable set of block libraries that let you design, simulate, implement, and test a variety of time-varying systems, including control systems, signal processing, communications, and robotics. Simulink is integrated with MATLAB, allowing users to incorporate MATLAB algorithms into models and export simulation results for further analysis. It's widely used in education, engineering, and research for multidisciplinary projects[5].

II. System Description

A. Theoretical Description: LMS Algorithm and Normalized LMS Algorithm

We can represent the LMS algorithm as:

$$\boldsymbol{y(n) = \hat{w}^H(n)u(n)} \tag{1}$$

$$\boldsymbol{e(n) = d(n) - y(n)} \tag{2}$$

$$\boldsymbol{\hat{w}(n+1) = \hat{w}(n) + \mu u(n)e^*(n)} \tag{3}$$

Where u(n) is the input vector regressor, d(n) is the corresponding desired response, and $\boldsymbol{\hat{w}(n)}$ is an estimate of the unknown tap-weight vector, $\boldsymbol{w(n)}$, of the linear multiple regression model used to represent the environment from which u(n) and d(n) are jointly picked [1]. The signal flow of the LMS algorithm is shown in Fig. 1[1].

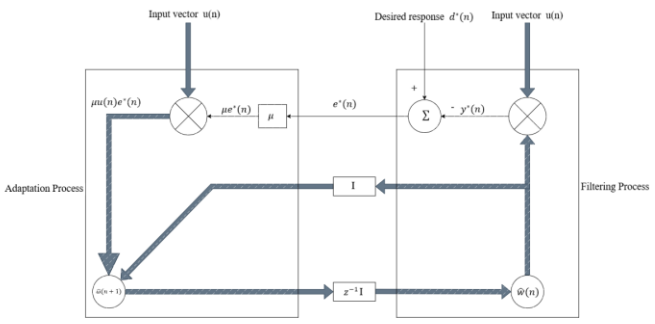


FIGURE 1. Signal-flow graph representation of the LMS algorithm, where I is the identity matrix and z^{-1} is the unit-time delay operator.

The normalized LMS algorithm is the same as the original LMS algorithm, as shown in Fig. 2[1]. The only difference between the LMS algorithm and the Normalized LMS is the weight controller is mechanized[1].

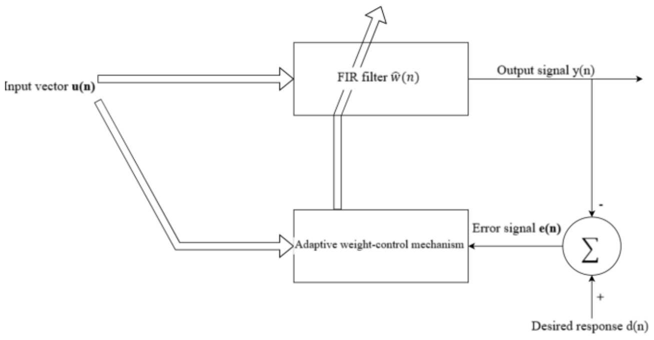


FIGURE 2. Block diagram of adaptive FIR filter.

we need to apply an adaptive echo canceller to reduce the acoustic echoes in telecommunication environments. The principle of the echo canceller can be described as an adaptively synthesized replica of the echo and subtracting it from the echo-corrupted signal[1]. Therefore, we can mitigate the echo effect in telecommunication. The structure of the Echo Canceller is shown in Fig. 3[1].

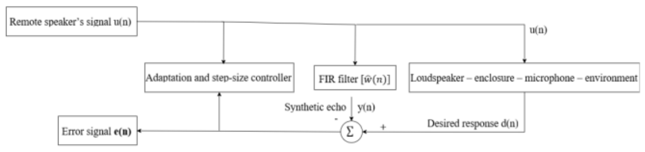

Remote speaker's signal u(n)
u(n)
Adaptation and step-size controller
FIR filter [$\hat{w}(n)$]
Loudspeaker – enclosure – microphone – environment
Synthetic echo
y(n)
Error signal **e(n)**
Σ
Desired response d(n)

FIGURE 3. The structure of Echo Canceller.

B. Simulation Description: Noise cancellation system[6] in MATLAB Simulink

The Noise Cancellation system uses the Normalized LMS filter module from Simulink Toolbox to filter noises from both real-time audio input and simulation sources. The noise cancellation system is shown in Fig. 4.

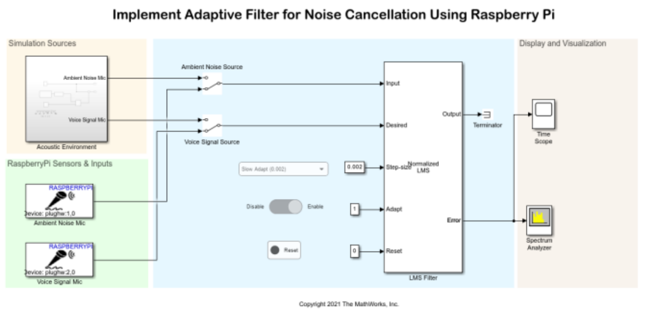


FIGURE 4. The noise cancellation system.

The system includes two input sources: simulation sources and real-time audio input sources received by Raspberry Pi two USB audio inputs at 8000Hz sampling frequency with 800 samples per frame. The device bit depth is the 16-bit integer with 1 channel. The simulation sources have a voice signal microphone output port, which outputs the summation of the sine wave and Gaussian white noise. Another port outputs the ambient noise to provide the correlation between the voice signal and the noise. The sine wave frequency is 860 Hz. And the sampling frequency is 8000Hz with 800 samples per frame. The structure of the source simulation system is shown in Fig. 5.

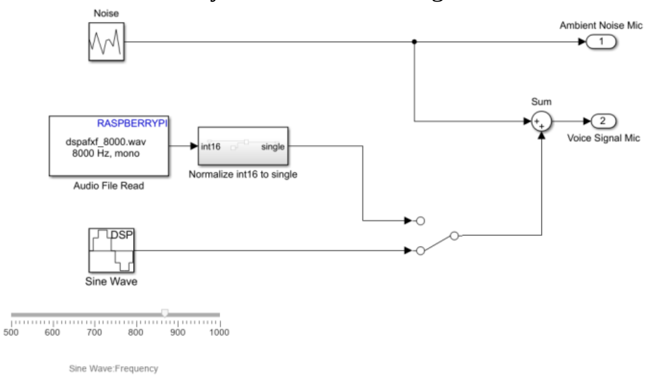
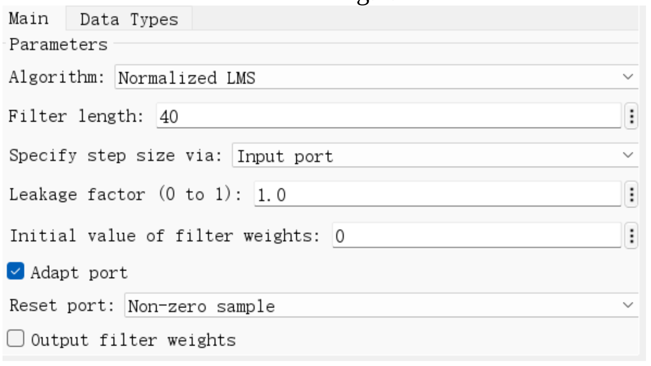


FIGURE 5. The structure of the source simulation system.

The noise cancellation has two switches to change the input sources. The normalized LMS receives the input signal and continuously adjusts its coefficients to minimize the between the predicted output and the actual desired signal in its iterative processing. The specifications for the normalized LMS are shown in Fig.6.


Main | Data Types

Parameters

Algorithm: Normalized LMS

Filter length: 40

Specify step size via: Input port

Leakage factor (0 to 1): 1.0

Initial value of filter weights: 0

☑ Adapt port

Reset port: Non-zero sample

☐ Output filter weights

FIGURE 6. The specifications of the normalized LMS filter.

The noise cancellation has the oscilloscope and spectrum analyzer to observe the processed signal in the time domain and the frequency, respectively.

C. Simulation Description: The comparison of noise cancellation system[6] in MATLAB Simulink

In order to observe the effect of the normalized LMS filter. A comparison group, without using the normalized LMS filter is applied to prove the validity of the experiment. The comparison group maintains the same specifications as the experiment group for other modules. The comparison of the noise cancellation system is shown in Fig.7.

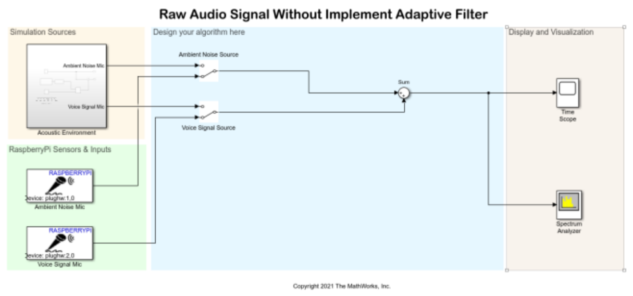


FIGURE 7. The comparison of the noise cancellation system.

D. Physical model Description: Firmware devices

The Raspberry Pi needs two USB audio microphones to receive the real-time audio signal and transmit the signal to the Simulink and a noise maker, which can create constant noise for two USB microphones to provide the correlation condition for the normalized LMS filter to adapt its iterative coefficients. One USB is used to receive the audio signal from the mobile phone mp3 player and ambient noise mainly from the noise maker (laptop cooling fan noise), which should be close to the audio signal input. Another USB is used to receive the ambient noise mainly from the noise maker.

It is required to connect both the host PC and the Raspberry Pi in the same LAN (local area network) to conduct the communication process. The Raspberry Pi used for this project is shown in Fig.8.

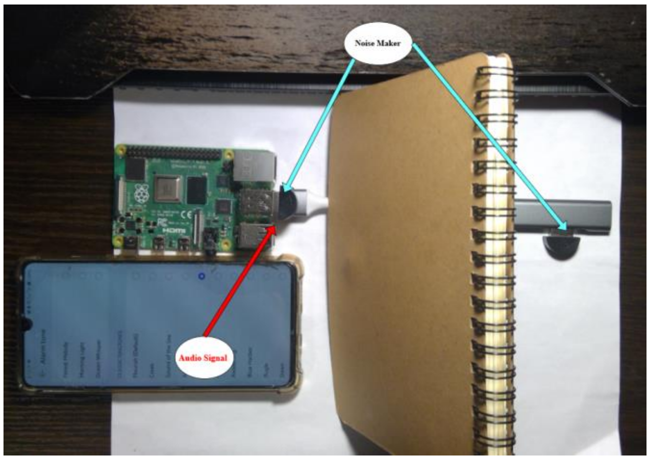


FIGURE 8. The Raspberry Pi with two USB audio inputs.

III. Simulation and Implementation results

A. Simulation results comparison

The output signal from the simulation sources input, without using the normalized LMS filter in the time domain and the frequency domain are shown in Fig.9, and Fig.10, respectively.

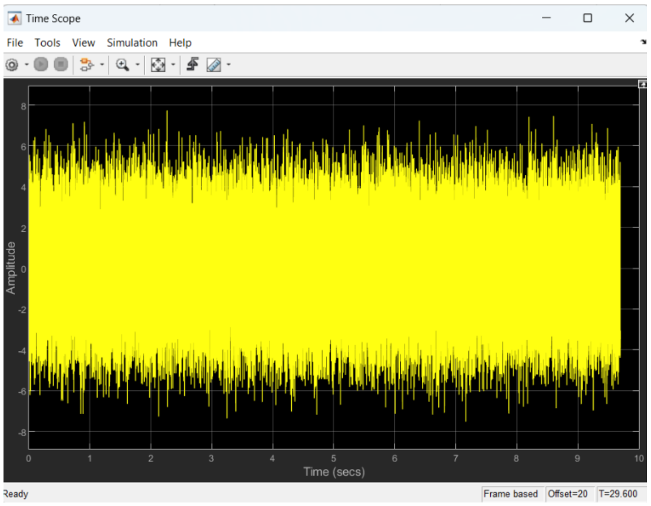


FIGURE 9. The output signal from the simulation sources, without using the normalized LMS filter in the time domain

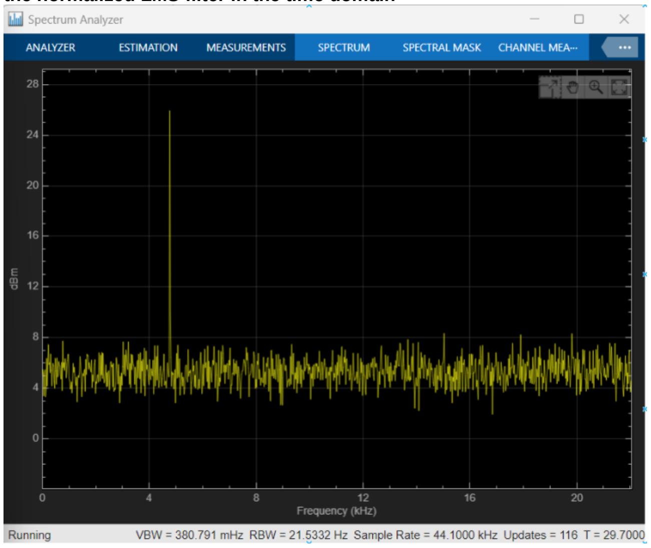


FIGURE 10. The output signal from the simulation sources, without using the normalized LMS filter in the frequency domain

The initial output signal from the simulation sources input with the normalized LMS filter in the time domain and the output from the simulation sources input with the normalized LMS filter in the time domain after some time are shown in Fig.11, and Fig.12, respectively.

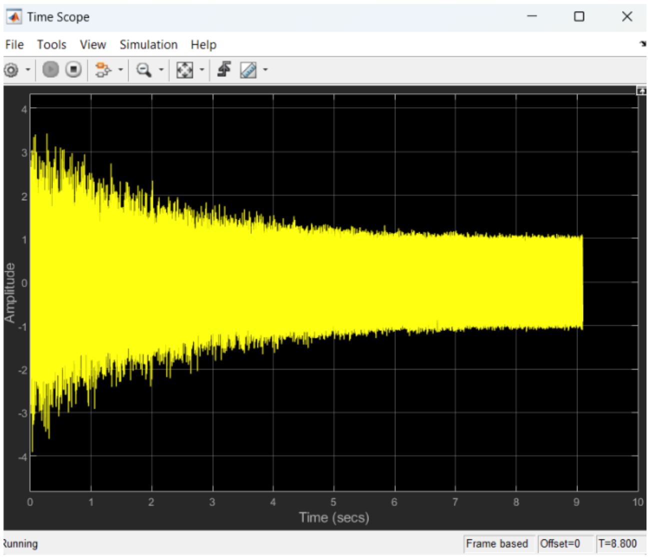


FIGURE 11. The initial output signal from the simulation sources with normalized LMS filter in the time domain.

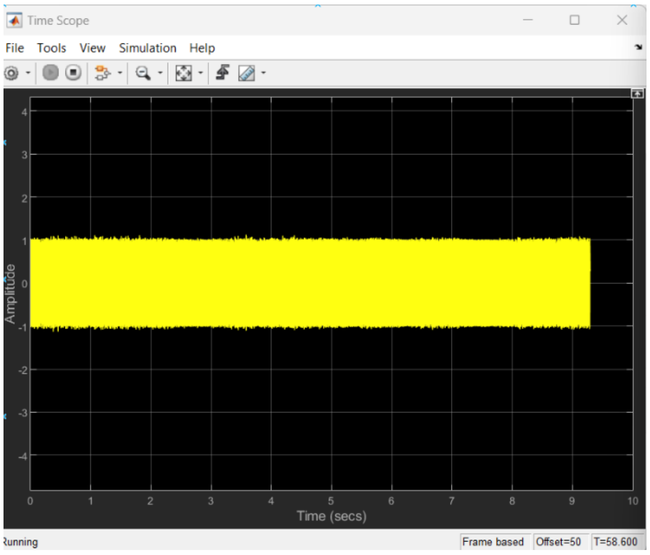


FIGURE 12. The output signal from the simulation sources with normalized LMS filter in the time domain after some time.

The initial output signal from the simulation sources input with the normalized LMS filter in the frequency domain and the output from the simulation sources input with the normalized LMS filter in the frequency domain after some time are shown in Fig.13, and Fig.14, respectively.

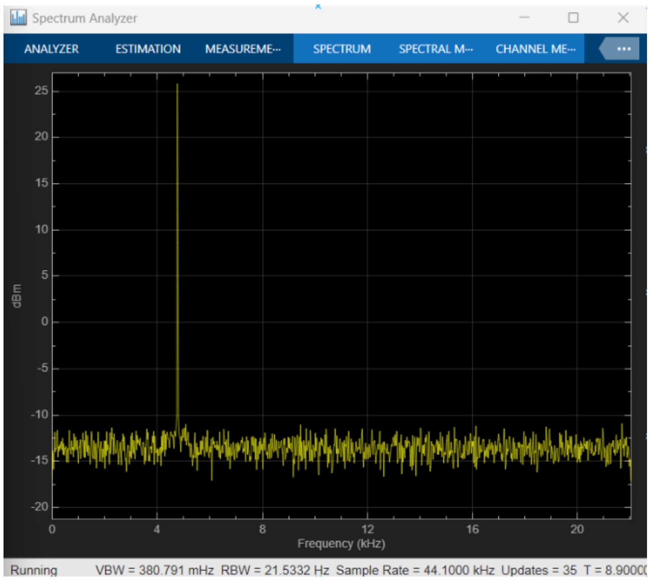


FIGURE 13. The initial output signal from the simulation sources with normalized LMS filter in the frequency domain.

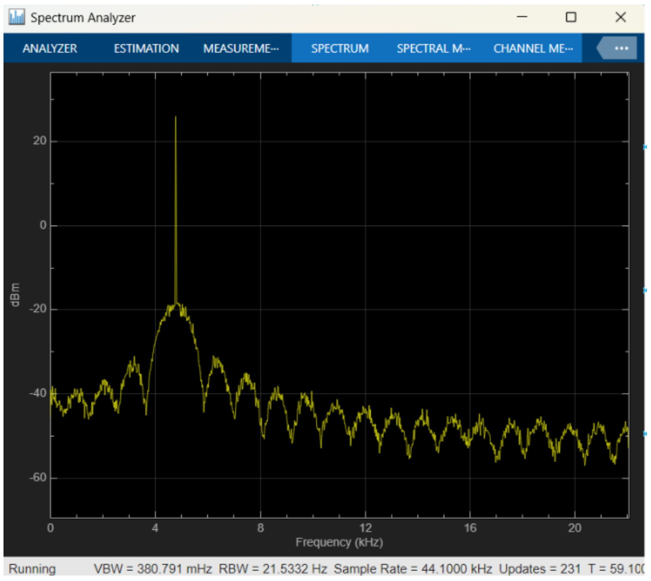


FIGURE 14. The output signal from the simulation sources with normalized LMS filter in the frequency domain after some time.

Comparing Fig.9 to Fig. 12, we can observe the effect of implementing a normalized LMS filter for noise reduction in the time domain, as the noisy sine wave has been filtering out the artifact. Therefore, it retains the original amplitude of the initial sine wave.

Comparing Fig.10 to Fig. 14, we can observe the effect of implementing a normalized LMS filter for noise reduction in the frequency domain. Since the noise frequency has a significant reduction after applying the normalized LMS filter.

Comparing Fig. 11 to Fig. 12, we can view the transient state of the normalized LMS filter and the initialization of the adaptation process of the normalized

LMS filter. After the normalized LMS adjusted its coefficient, we can observe the steady state of the normalized LMS output.

Comparing Fig. 13 to Fig. 14, we can observe the process of the normalized LMS filter reducing the magnitude of the noisy artifacts.

B. Practical Application results comparison

The output signal from the Raspberry Pi audio input, without using the normalized LMS filter in the time domain is shown in Fig. 15. The output signal from the Raspberry Pi audio input, without using the normalized LMS filter in the frequency domain is shown in Fig. 16.

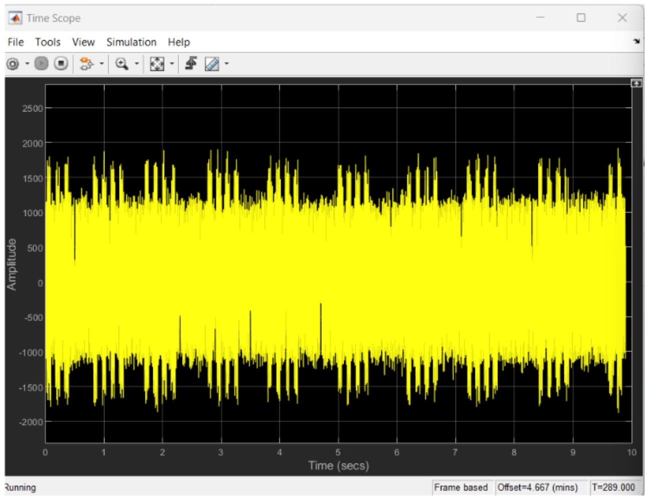
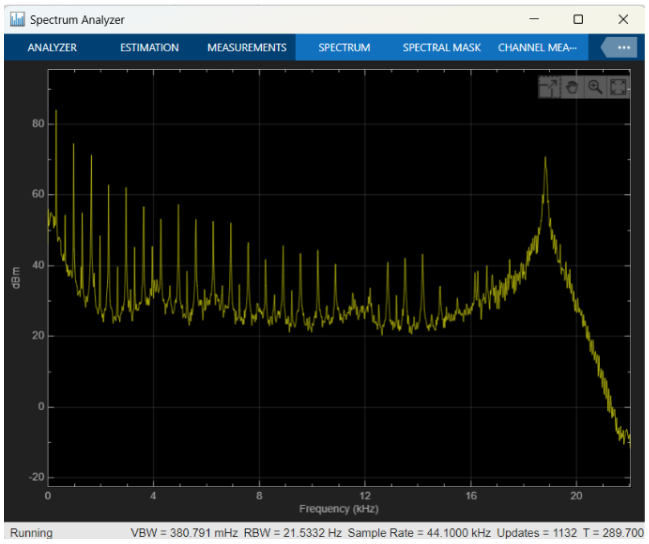


FIGURE 15. **The audio signal output from the Raspberry Pi's input, without using a normalized LMS filter in the time domain.**



VBW = 380.791 mHz RBW = 21.5332 Hz Sample Rate = 44.1000 kHz Updates = 1132 T = 289.700

FIGURE 16. **The audio signal output from the Raspberry Pi's input, without using a normalized LMS filter in the frequency domain.**

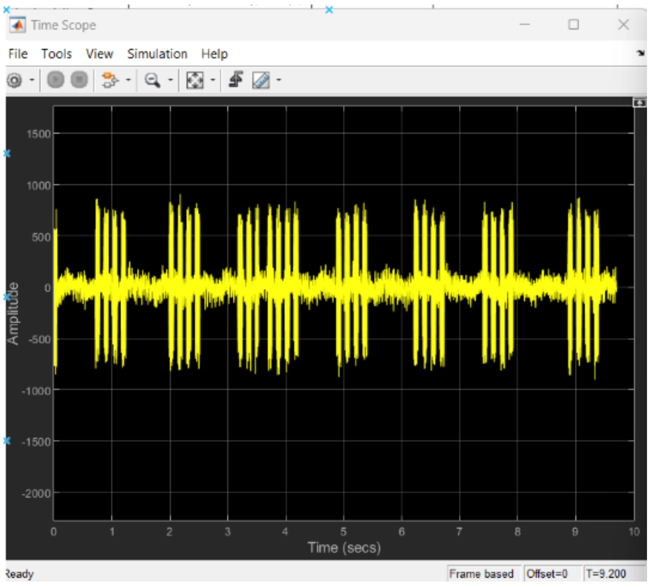


FIGURE 17. **The initial output signal from the Raspberry Pi audio input with normalized LMS filter in the time domain.**

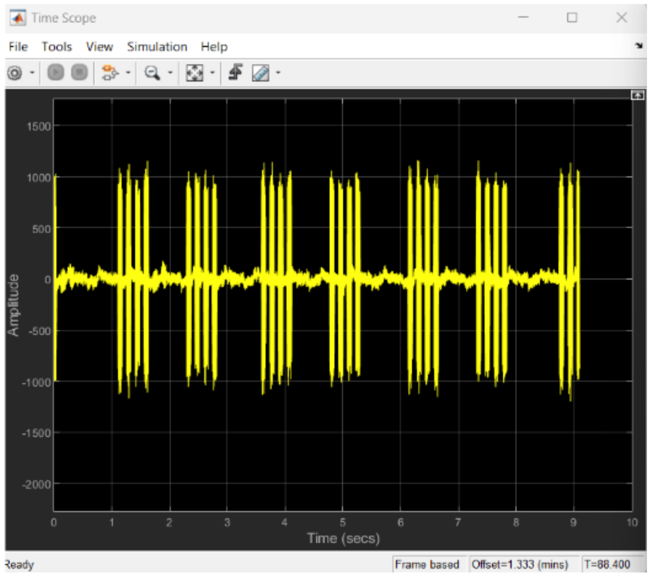
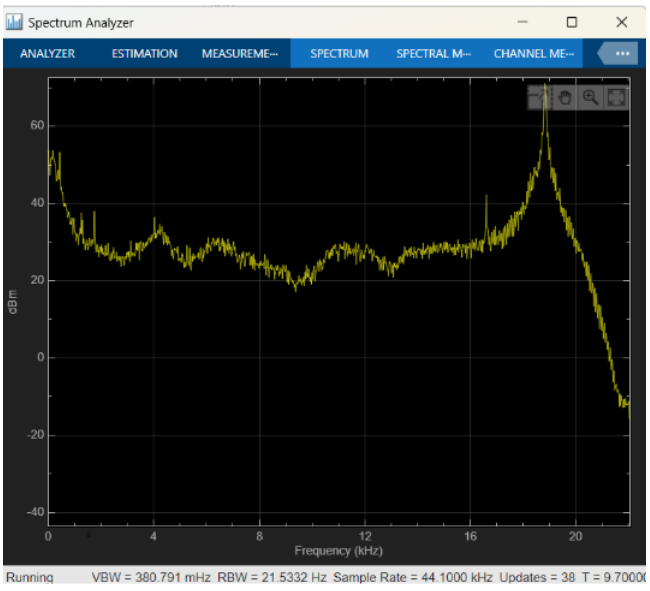
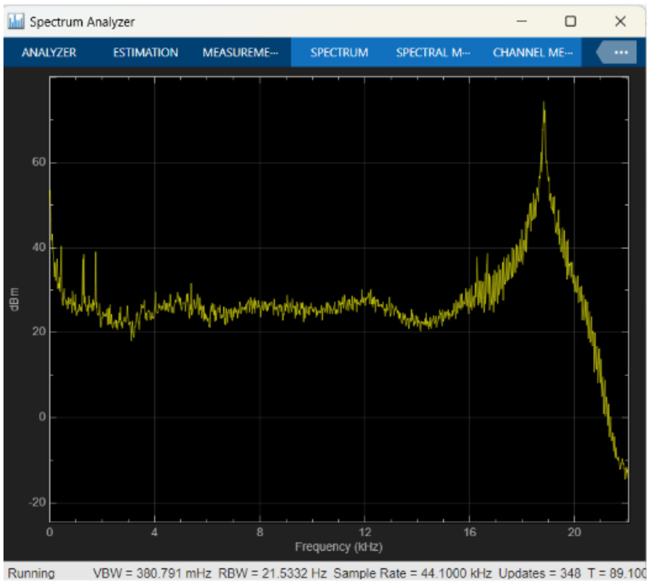


FIGURE 18. **The output signal from the Raspberry Pi audio input with normalized LMS filter in the time domain after some time.**



Running VBW = 380.791 mHz RBW = 21.5332 Hz Sample Rate = 44.1000 kHz Updates = 38 T = 9.7000

FIGURE 19. **The initial output signal from the Raspberry Pi audio input with normalized LMS filter in the frequency domain.**



Running VBW = 380.791 mHz RBW = 21.5332 Hz Sample Rate = 44.1000 kHz Updates = 348 T = 89.100

FIGURE 20. **The output signal from the Raspberry Pi audio input with normalized LMS filter in the frequency domain after some time.**

Comparing Fig.15 to Fig. 18, we can observe the effect of implementing a normalized LMS filter for noise reduction in the time domain. Since the fan noise has been identified and significantly filtered by the normalized LMS. The amplitude of the filtered alarm impulse has some reduction, as the overlapping noise has been filtered out.

Comparing Fig.16 to Fig. 20, we can observe the effect of implementing a normalized LMS filter for noise reduction in the frequency domain. Since the noisy impulse has a significant reduction after applying the normalized LMS filter.

Comparing Fig. 17 to Fig. 18, we can view the initialization and the adaptation process of the normalized LMS filter. After the normalized LMS adjusted its coefficient, we can observe the amplitude has gained a boost and the noisy component has a significant reduction in the output of the normalized LMS filtered in the time domain.

Comparing Fig. 19 to Fig. 20, we can observe the process of the normalized LMS filter reducing the magnitude of the noisy artifacts.

IV. Conclusion

This work has demonstrated that the implementation of normalized LMS for noise cancellation using Raspberry Pi is an applicable approach for designing real-time audio processing in practical usages. The simulation and the experiment are conducted to validate the effectiveness and efficiency of this system. According to the results, both simulated sources and audio signals have significant noise reduction, which can be observed in the both time and frequency domain. The results show that the system is capable of significantly reducing noise while preserving the quality of the original audio signal. This makes it a viable solution for various applications, such as communication systems, hearing aids, and noise-cancellation systems.

The use of Raspberry Pi also makes the solution more cost-effective and accessible, opening up possibilities for broader application in both commercial and educational settings. Future work may explore the integration of more advanced algorithms and the application of this system in more complex noise environments, further enhancing its utility in the field of audio processing technology.

Acknowledgment

The author expresses deep gratitude to Dr. V. Parsa for providing extensive personal and professional guidance.

REFERENCES

[1] Simon Haykin, Adaptive Filter Theory, Prentice-Hall, Inc., 1996.

[2] Paleologu, C., Ciochină, S., Benesty, J. et al. An overview on optimized NLMS algorithms for acoustic echo cancellation. EURASIP J. Adv. Signal Process. 2015, 97 (2015). https://doi.org/10.1186/s13634-015-0283-1

[3] Desai, N., Pandya, U., & Desai, C. Active Noise Control Using LMS & NLMS Algorithm. Chhotubhai Gopalbhai Patel Institute of Technology, Bardoli, India.

[4] Raspberry Pi Ltd. raspberry-pi-4-product-brief, December 2023.

[5] The MathWorks, Inc., "Get Started with Simulink," in Help Center: Simulation and Model-Based Design.