Implementation of Pre-Emphasis Analog Filter on Raspberry Pi Using Zero-Order Hold Discretization as a Pre-Processing to Humanoids' Commands Recognition

Jans Hendry[1,*], Budi Sumanto[1], Yoga Mileniandi[1], Putri Mawaring Wening[1]
[1]Department of Electrical Engineering and Informatics, Universitas Gadjah Mada; budi.sumanto@ugm.ac.id, yogamileniandi@mail.ugm.ac.id, putrimawaringweningw@mail.ugm.ac.id
[*]Correspondence: jans.hendry@ugm.ac.id

Intisari – Dalam teknologi yang menggunakan pengolahan sinyal wicara, tapis* pre-emphasis *sering digunakan sebelum proses ekstraksi ciri. Tapis ini dapat memperbesar amplitudo dari sinyal pada frekuensi tinggi sambil tetap mempertahankan bagian sinyal pada frekuensi rendah. Dalam penelitian ini, kami menerapkan tapis* pre-emphasis *analog yang dikembangkan dari tapi pelolos frekuensi tinggi. Lalu,* filter *tersebut didiskretkan menggunakan metode* zero-order hold *(ZOH). Kedua* filter *tersebut memiliki* pole *yang menunjukkan kestabilan. Nilai* pole *dari tapis diskret selalu di titik nol, sedangkan nilai* zero *-0.9375. Hasil rekonstruksi dengan menggunakan tapis* de-emphasis *menunjukkan nilai* mean-squared error *nol yang menyimpulkan bahwa strategi menggunakan metode* frame-by-frame *untuk implementasi tapis* pre-emphasis *sangat dianjurkan.

Kata kunci* – pre-emphasis, de-emphasis, mean-squared error, *tapis, wicara

Abstract – In speech processing technology, the pre-emphasis filter is often used before feature extraction. This filter can emphasize amplitude with high frequencies while preserving the low frequencies side. In this research, we implemented a complete analog pre-emphasis filter from a conventional analog high-pass filter. Then the analog pre-emphasis filter is discretized using zero-order hold (ZOH). The analog and discrete forms have been shown stable according to pole location. The pole of the discrete form is always at the origin, while zero is -0.9375. The reconstruction result, with the help of the de-emphasis filter, shows no discrepancies over the original, hence the mean-squared error is zero which establishes the conclusion that the frame-by-frame method applies to the pre-emphasis filter is lossless and preferable.

Keywords – pre-emphasis, de-emphasis, mean-squared error, filter, speech

I. INTRODUCTION

Mainly, speech recognition is one of the features that humanoid robots are supposed to have. It aims to understand the meaning of words spoken to them, for instance, commands, questions, or other information. Generally, speech recognition comprises three stages namely pre-processing, processing, and recognition. In the pre-processing stage, standardization (normalization and dc-removal) and filtering can be the most inevitable process. They highly influence the quality of input for feature extraction that can be used further for artificial intelligence methods like machine learning. Certain well-known feature extraction methods use filtered speech signals as their input. This filter is called the pre-emphasis filter. It is merely a high-pass filter (HPF) that can be built in the analog form by using capacitor, inductor, and resistor components. Another filter so-called de-emphasis filter is a reciprocal of the aforementioned filter. It is a low-pass filter (LPF) that can reconstruct original speech from pre-processed speech. This filter also can be built using available passive components as well as the pre-emphasis filter.

There are several novel research still uses this technique as part of their main method. They are various research fields that embed speech processing in their research to produce a signal for actuators or the rest of the system. The research utilizes mel frequency cepstral coefficients (MFCC) and support vector machine (SVM) to control five degree-of-freedom (DoF) robot arms based on speech command has been conducted [1]. By using 12 cepstrum features of each speech, the system can recognize correctly 70% of test data. A similar feature extraction method has also been used to create a home automation system [2]. They implemented the design system into raspberry pi and NodeMCU and proved that the concept can work. A noisy speech recognition system that allegedly can help autism therapy has also been developed. This system is planned to be planted into a helper robot that uses MFCC which incorporates the aforementioned filter [3]. Implementation of mobile robot control based on speech recognition in an embedded device is feasible especially when the module and programming software has been provided for instance, by using NI MyRIO-1900 [4]. Research that focuses on a study of the effect of environmental disturbances on a robot has also been conducted. The system used MFCC which originally used the pre-emphasis filter before calculation. The result said that maximum accuracy in the appropriate conditions can reach 93.51% [5]. The pre-emphasis filter can be used as an independent tool to help voice data transmission. This filter can protect high frequencies from random noise in the air. When received, the signals are reconstructed back to voice data using the de-emphasis filter [6]. Hence, it shows that the role of this filter is significant as it provides a different representation of original signals to be used for feature extraction or data protection.

Recalling the significant role of this filter, this research aims to implement the pre-emphasis and the de-emphasis filter in the real-time process by using the Raspberry Pi 3

Model B. Both pre-emphasis and de-emphasis filters will be drawn from the analog form to give a comprehensive understanding. Added to that, a discretization method is used which is a zero-order hold. Raspberry Pi is one of the popular tiny computers utilized in robotic projects that can be incorporated with a microcontroller to form a complete (humanoid) robot. The de-emphasis is not necessarily needed in some cases, but it will be used to measure the error before and after filtering. In some research, the processing relies on a commonly used computer like a laptop or desktop, then it connects to a microcontroller to control the robots. In our research, this machine is proposed to be replaced by Raspberry Pi. This paper is organized as follows: 1) the motivations and aims are explained in the Introduction, 2) the methods of this research are explained in the Methodology, 3) the results and discussions are described in the Result and Discussion, 4) the Conclusion.

II. METHODOLOGY

Speech signals belong to non-stationary signals. These signals can be seen as stationary signals when they are divided into blocks around 10 to 30 ms, or quasi-stationary signals. Commonly, frequencies of human speech are below 4 kHz. Subject to the Nyquist theorem, the common sampling frequency used is 8 kHz to avoid aliasing. The pre-emphasis filter can be used as part of an algorithm or stand-alone. When used as part of an algorithm, it is mainly included in the features extraction phase, for example, MFCC and linear predictive coding (LPC). Many speech processing methods, tend to produce better patterns for low frequencies than high frequencies. Hence, it is common to emphasize high frequencies before the further process [7]. The pre-emphasis filter is a type of finite impulse response (FIR) where poles are always at the origin and zeros can be anywhere inside the unit circle. This is the reason the pre-emphasis filter and its reciprocal are always stable.

A. Analog Form

The realization of the pre-emphasis filter is a high-pass filter (HPF) while the realization of its reciprocal is a low-pass filter (LPF). This filter in analog form can be made of passive components. The ideal LPF and HPF responses are illustrated in Figure 1(a)-(b). These ideal filter responses depict the target of any designed LPF or HPF. Nonetheless, they are difficult to reach. Hence, all designed filters are intended to yield approximation values.

The pre-emphasis filter is a first-order filter that contains exactly one energy storage component for instance a capacitor. Even if the pre-emphasis filter resembles an HPF, it is slightly different in function. An ordinary HPF passes higher frequencies after 3 dB frequency while blocking lower frequencies. It does not meet the criteria of a filter needed in speech processing where all frequencies should be kept as they are with some part of them amplified to emphasize their existence. The ordinary HPF as shown in Figure 2(a) blocks all the lower frequencies below the cut-off frequency. To maintain the lower frequencies, a

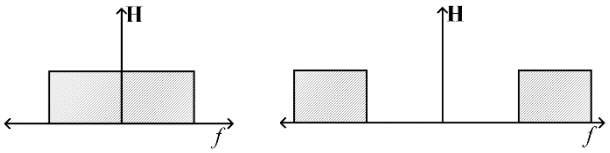


Figure 1. Ideal filter (a) low-pass filter (b) high-pass filter

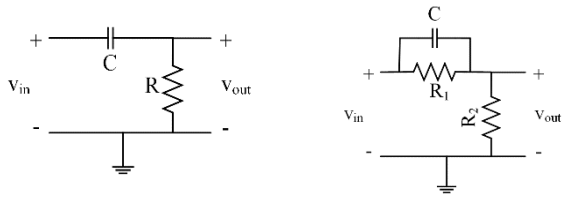


Figure 2. (a) Ordinary HPF, (b) pre-emphasis filter

resistor R_1 parallels with the capacitor C should be put on as shown in Figure 2(b).

Assume that $\mathbf{H}_\infty\mathbf{(S)}$ is a no-load transfer function in the Laplace form of the ratio between v_{out} and v_{in}. The impedance between C and R_1 is expressed by (1).

$$Z_1 = R_1/(R_1Cs + 1) \tag{1}$$

Then,

$$\mathbf{H}_\infty(\mathbf{S}) = \frac{v_{out}}{v_{in}} = \frac{R_2}{R_2 + Z_1} \tag{2}$$

by substituting Z_1 into (2), it yields

$$\mathbf{H}_\infty(\mathbf{S}) = \frac{R_1R_2Cs + R_2}{R_1R_2Cs + (R_1 + R_2)} \tag{3}$$

by dividing the numerator and denominator in (3) with R_1R_2C, it leaves the unit constant for the higher order of the transfer function as expressed in (4).

$$\mathbf{H}_\infty(\mathbf{S}) = \frac{s + (1/R_1C)}{s + \frac{1}{C}(\frac{1}{R_1} + \frac{1}{R_2})} \tag{4}$$

The stability of the transfer function can also be examined by finding the pole of the transfer function. According to (4), there is only one pole, denoted with sp, with a value in (5).

$$s_p = -\frac{1}{\frac{1}{C}(\frac{1}{R_1} + \frac{1}{R_2})} \tag{5}$$

as can be seen that no matter what the values of the capacitor and resistors, the pole will always be on the left side of the S-complex plane. It concludes that the analog passive pre-emphasis filter is always stable. Meanwhile, the zero value from (4), denoted with s_0, is shown in (6) which also lies on the left side of the S-complex plane.

$$s_0 = -\frac{1}{R_1 C} \tag{6}$$

B. Discrete Form

To implement the analog passive filter in a microcontroller or processor, they should be converted to a discrete-time transfer function. In this research, the zero-order hold (ZOH) method is used. A continuous-time system, denoted with $\mathbf{H}_\infty\mathbf{(S)}$, can be discretized by using (7).

$$\mathbf{H}_\infty(\mathbf{Z}) = (1 - z^{-1})\mathrm{Z}\left\{L^{-1}\left[\frac{\mathbf{H}\infty(\mathbf{S})}{s}\right]\right\} \tag{7}$$

Equation (7) tells that the continuous-time transfer function should be divided by *s and* then inversed using the Laplace inverse transform. Then, it is followed by Z-transform before being multiplied by $1 - z^{-1}$.

C. Mean Square Error

Let *x[n]* and $\hat{x}[n]$ denote the original and the reconstructed speech signal, consecutively. Then, the error of the signal before and after filtering is calculated using mean squared error as expressed by (8). Here, *N* denotes the length of the speech signal.

$$mse = \frac{1}{N}\sum_{n=1}^{N}(x[n] - \hat{x}[n])^2 \tag{8}$$

III. RESULT AND DISCUSSION

The pre-emphasis filter is a first-order FIR filter. Hence, the components value should be selected carefully as such it contains pole only in origin and zeros inside the unit circle. This requirement ensures that the pre-emphasis filter is always stable. The optimal passive components value that meets the requirement is shown in Table 1. The range of capacitor that can be selected is broader than the resistor. The value displayed in Table 1 is the minimum value that the filter should have. By using selected component values, the analog transfer function in (4) becomes

$$\mathbf{H}_\infty(\mathbf{S}) = \frac{s + 6.667 * 10^5}{s + 1.067 * 10^7} \tag{9}$$

As can be seen that pole lies on the right-side S-complex plane, s_p = *-1.067*10^7* as well as the zero which is s_0 = *-6.667*10^5* as also shown by Figure 3 where the pole is marked with X and zero is marked with O.

Table 1. Selected components value

Type	Symbols	Value
Resistor	R_1	15 kΩ
Resistor	R_2	1 kΩ
Capacitor	C	0.1 nF

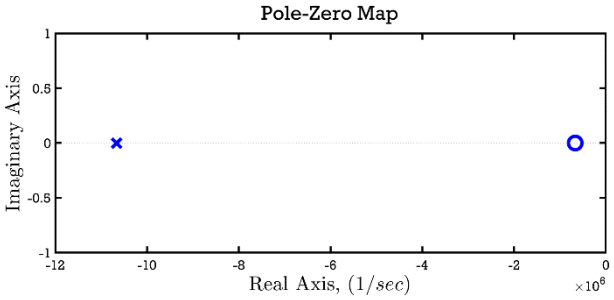
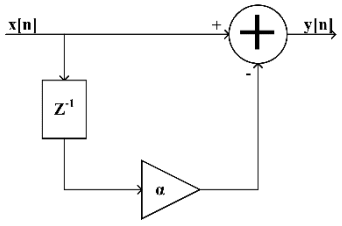


Figure 3. Pole-zero of analog pre-emphasis filter

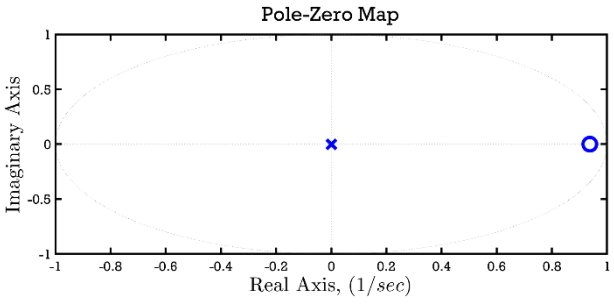
The analog transfer function is discretized by using (7) and yields (10). This form shows that pole exists only at the origin, while zero equals 0.9375 which lies exactly inside the unit circle. The time-domain form of this filter is a Z-transform inverse form that produces a causal filter. In this domain, the inversed form can be drawn using a direct form of FIR filter as shown in Figure 4(a). Along with that, the pole and zero plots are expressed in Figure 4(b) to show how they are nicely placed inside the unit circle.

$$\mathbf{H}_\infty(\mathbf{Z}) = \frac{z - 0.9375}{z} \tag{10}$$

The pre-emphasis filter in the form of (10) is programmed into Raspberry Pi Model 3 B as a digital filter for speech data input. The specification of this minicomputer is shown in Table 2 with the installation in Figure 5. In this experiment, a *frame-by-frame* method is applied. With this method, there is a chance that discontinuity between each consecutive frame will happen. This discontinuity is audible like a *click* or *nasal* sound. To avoid this, internal *history* data has been used. This internal *history* is yielded after each filtering of each frame; hence they can be called at any time to be incorporated into the next filtering.



(a)



(b)

Figure 4. (a) Direct form of first-order FIR filter, (b) pole-zero of discrete pre-emphasis filter

Table 2. Raspberry Pi 3 Model B specifications

Item	Remark
Processor	Quad Core Broadcom BCM2837 64bit ARMv7
Speed	1.25 GHz
Output Current	Up to 2.5 A

Figure 5. Raspberry Pi and microphone installation

The microphone attached to Raspberry Pi is recognized as *USB-Audio-UACDemoV1.0* with device ID '1,0', the number of channels equals 1, the bit depth is a 16-bit integer, and the sampling frequency is 48 kHz. Even though there is a discrepancy between the available device's sampling frequency and discretization (= 8 kHz), we can still use the designed discrete pre-emphasis. As can be seen in (11), changing the sampling frequency to 48 kHz does not change the pole much.

$$\mathbf{H}_{\infty}(\mathbf{Z}) = \frac{z - 0.9375}{z - 3.091 * 10^{-97}} \qquad (11)$$

The comparison of magnitudes response between analog and discrete pre-emphasis filters is shown in Figure 6. The higher frequencies in the discrete filter are amplified, while in the analog (continuous) form they are neither attenuated nor amplified. It can also be observed that lower frequencies have attenuation factors less than 25 dB indicating they are passed by the resistor R_1.

When tested with a real-time process using Raspberry Pi with spoken word is *test123*, the original speech, pre-emphasis result, and de-emphasis result (reconstructed) along with their spectrum frequencies are displayed in Figure 7.

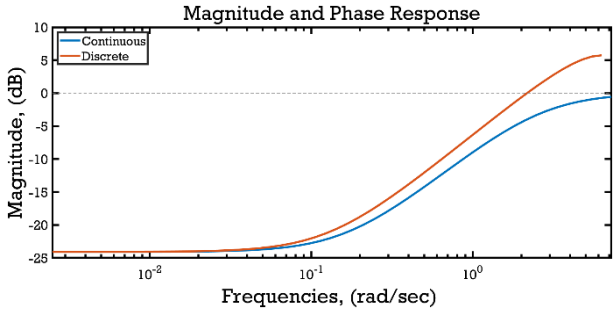


Figure 6. the Magnitude response comparison of analog and discrete filter

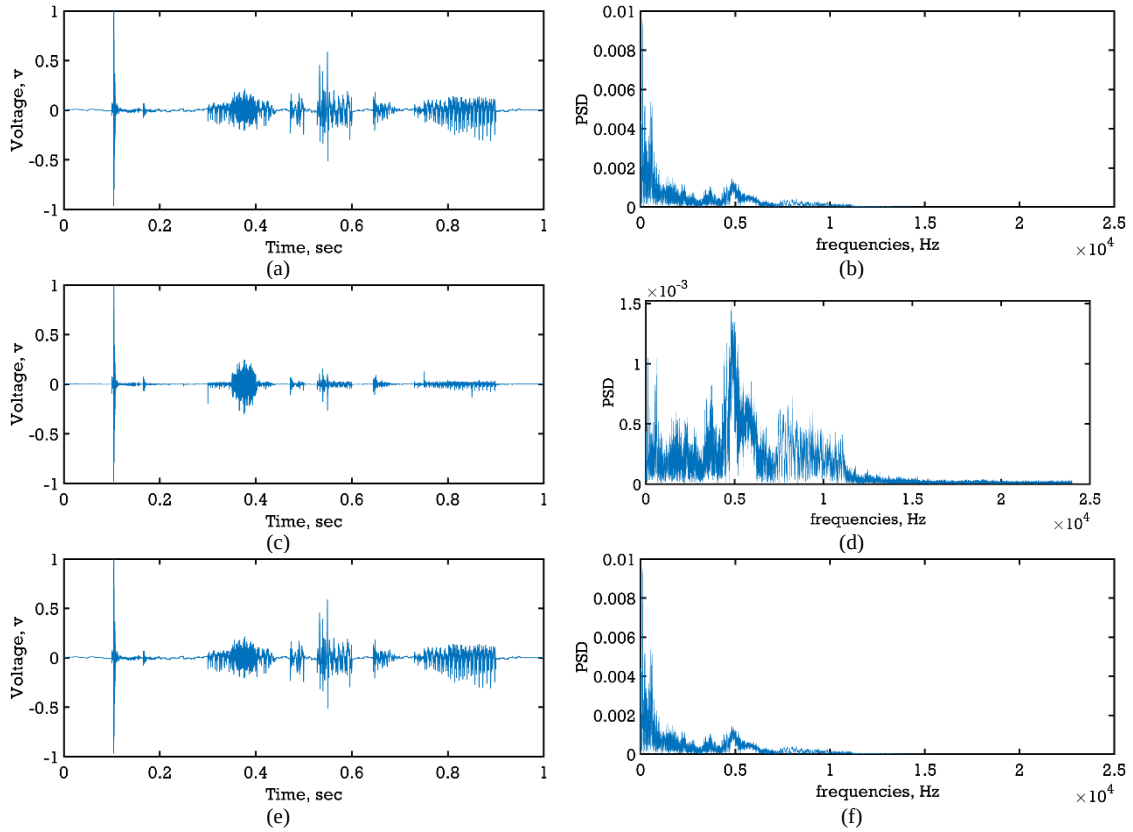


Figure 7. (a) Original speech, (b) pre-emphasis speech, (c) de-emphasis speech, (e) and (f) their corresponding spectrum frequencies

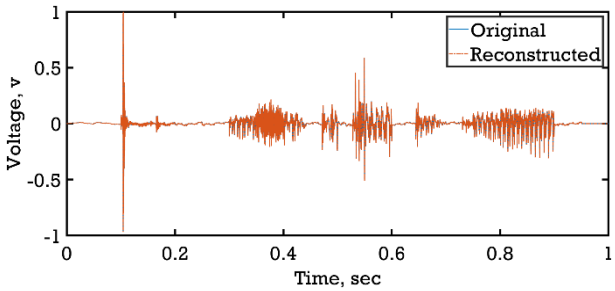


Figure 8. Original vs. reconstructed speech signal

As can be seen that from the figure that the original speech signal mostly occupies low frequencies bar, and it can be verified from its spectrum frequencies in Figure 7(a) to (b). Then, the pre-emphasis filter lifted the amplitude of the high frequencies side of the signal and slightly lowered the low frequencies amplitude as can be seen in Figure 7(c)-(d). By listening to the result through Raspberry Pi using *Audio Jack 3.5mm* port, likely, the pre-emphasis filter has attenuated the whole sound. Then, the de-emphasis filter which is merely a low-pass filter with the same discrete coefficients is applied to help the reconstruction at the end of the process. As can be seen in Figure 7(e) to (f), the de-emphasis filter has successfully reconstructed the filtered signal that the discrepancies are hard to see.

The mean-squared error between the original and reconstructed signal is 0 which means the strategy that we designed for pre-emphasis which is based on the *frame-by-frame* method establishes a *lossless* filter. It means no information is lost hence the further feature extraction method can extract wanted features. Added to that, the absolute error between the original and reconstructed speech is shown in Figure 8 as visual proof.

IV. CONCLUSION

Two options can be chosen for pre-emphasis implementation either in analog or discrete form. The zero-order hold can be an optimal method for analog to discrete conversion. The effect of a sampling frequency selection is not major, hence arbitrary selection will work. According to the pole location, analog and discrete forms are always stable. In the discrete form, the pre-emphasis filter lifts up the amplitude of the high frequencies part and slightly lowers the amplitude of the low frequencies part. On the contrary, the high frequencies part will be kept as they are while slightly attenuating the low frequencies part in the analog pre-emphasis filter. The mean-squared error between the original and reconstructed signal is 0 which indicates that the *frame-by-frame* method we employ establishes a lossless filter that can help further feature extractions method to extract appropriate features. In the future, we are going to implement the whole of our research result into Raspberry Pi starting from speech separation and proceeding to speech command recognition.

ACKNOWLEDGEMENT

This is part of the main research sponsored by the Department of Electrical Engineering and Informatics and Vocational College of Universitas Gadjah Mada under a *competitive scheme* with contract number 92/UN1.SV/K/2021.

REFERENCES

[1] D. Anggraeni, W. S. M. Sanjaya, M. Y. S. Nurasyidiek, and M. Munawwaroh, “The implementation of speech recognition using mel-frequency cepstrum coefficients (MFCC) and support vector machine (SVM) method based on python to control robot arm,” in *IOP Conference Series: Materials Science and Engineering*, 2018, vol. 288, no. 1, p. 012042.

[2] A. S. Haq, M. Nasrun, C. Setianingsih, and M. A. Murti, “Speech recognition implementation using MFCC and DTW algorithm for home automation,” in *Proceeding of the Electrical Engineering Computer Science and Informatics*, 2020, vol. 7, no. 2, pp. 78–85.

[3] S. Attawibulkul, B. Kaewkamnerdpong, and Y. Miyanaga, “Noisy speech training in MFCC-based speech recognition with noise suppression toward robot assisted autism therapy,” in *2017 10th Biomedical Engineering International Conference (BMEiCON)*, 2017, pp. 1–5.

[4] N. Adnene, B. Sabri, and B. Mohammed, “Design and implementation of an automatic speech recognition based voice control system,” 2021.

[5] B. Birch, C. A. Griffiths, and A. Morgan, “Environmental effects on reliability and accuracy of MFCC based voice recognition for industrial human-robot-interaction,” *Proc Inst Mech Eng B J Eng Manuf*, vol. 235, no. 12, pp. 1939–1948, 2021.

[6] A. F. Isnawati and J. Hendry, “Implementasi Filter Pre-Emphasis untuk Transmisi Sinyal Audio pada Sistem Komunikasi FBMC-OQAM,” *Jurnal Nasional Teknik Elektro Dan Teknologi Informasi (JNTETI)*, vol. 8, no. 4, pp. 340–346, 2019.

[7] I. McLoughlin, *Applied speech and audio processing: with Matlab examples*. Cambridge University Press, 2009.