Enjoy the Silence: Noise Control with Smartphones

Taesik Gong, Jun Hyuk Chang, Joon-Gyum Kim, Soowon Kang, Donghwi Kim, and Sung-Ju Lee
School of Computing, KAIST, Korea
{cathena913,brianjjh,kjkpoi,sw.kang,dhkim09a,profsj}@kaist.ac.kr

***Abstract*—While a certain sound serves a purpose to someone, the same sound could be noise to others. Specifically, an alarm sound could be necessary for someone to wake up and start the day, but it could be an unwanted sound for people sharing the same room who need not wake up as early. Noise cancellation is useful in this scenario, but most existing techniques require costly equipments (e.g., high quality speakers or microphones) or devices that are uncomfortable to wear during sleep. We thus explore the possibility of using only commodity smartphones to achieve active noise control and present Virtual Earplugs. As most people own a smartphone that includes a microphone and speakers, and has an alarm clock feature, we believe our system could be easily deployed and used in practice. We highlight the technical challenges in realizing our vision and demonstrate the feasibility of our approach using our preliminary prototype. Our results indicate that Virtual Earplugs reduces the alarm sound by up to 19 dB in only 2.1 seconds of processing delay.**

I. Introduction

Many people live in an environment where they share a room with other people during sleep. It could be a couple sharing a bed, roommates in a dorm room, or parents with an infant. There are occasions where not all members in the room need to wake at the same time. When the alarm buzzes for the earliest riser, other people in the room also hear the alarm and unnecessarily (and annoyingly) wake up. For them, the alarm sound is noise.

Noise cancellation (or control) is a technology that dates back to 1936 [1]. Traditional noise cancellation approaches include Passive Noise Control (PNC) [2] that attenuates the noise by sound-isolating materials, such as earplugs, sound barrier, etc. Another approach is Active Noise Control (ANC) [3] that cancels the noise by generating the opposite sound (the same magnitude but the inverted phase) of the noise. The most widely used ANC algorithms are Least Mean Square (LMS) [4] and Filtered-x LMS (FxLMS) [5]. These methods have been applied in noise-cancelling headsets [6], snoring noise reduction [7], etc. Recent noise cancelling products combine ANC and PNC to maximize the noise cancellation effect [8], [9].

Despite the efforts on noise cancellation, applying existing methods to cancel the unwanted alarm sound has several practical difficulties. First, most available noise cancelling products come in a wearable form [8]–[10]. These wearable devices require additional cost, and are uncomfortable to wear during sleep. Second, adopting traditional ANC algorithms (e.g., LMS and FxLMS) requires low audio latency (sub-millisecond) [11], thus must utilize specialized hardware.

In order to build an unwanted alarm cancellation system that could be widely deployed, we ask the question, *can we control the noise by using only commodity smartphones?* To this end, we present Virtual Earplugs. Unlike existing solutions, Virtual Earplugs require no wearable devices. Virtual Earplugs leverage the ANC principles to cancel the alarm sound and mitigate the audio latency problem in smartphones. Virtual Earplugs cancel the alarm sound in the following steps: in the offline stage, (i) the *alarmer* sends the time of the alarm to the *canceller* when the user sets the alarm before going to bed, and then (ii) the canceller analyzes the acoustic environment by generating the alarm sound. During the online stage, when the alarm sound is generated by the alarmer, (iii) the canceller finds the nearest volume and (iv) the canceller generates the optimal cancelling sound.

We built the Virtual Earplugs prototype using commodity smartphones (Nexus 5X and Samsung Galaxy S7), and evaluated the performance and feasibility: how much decibel (dB) is reduced from the original alarm sound. We measured the *cancelled zone* (i.e., reduced dB around the alarm canceller) created by Virtual Earplugs in three different environments. Our main contributions are summarized as follows:

- We design Virtual Earplugs, that reduce the dB of the unwanted alarm sound using only commodity smartphones without wearable devices.
- We identify practical challenges in realizing Virtual Earplugs when utilizing ANC in smartphones, such as the audio latency problem.
- We implement Virtual Earplugs to examine the feasibility and evaluate the performance in various acoustic environments. We measure the cancellation processing delay, the size of the cancelled area, and the amount of noise reduction in those areas.

The rest of the paper is organized as follows. Section II provides the background knowledge in noise control and challenges of performing noise control with smartphones. Section III presents the design and algorithms of Virtual Earplugs, and Section IV details the implementation of the system. The system is evaluated in Section V. Section VII presents the related work, and Section VI discusses the limitations of the current system and future directions. Finally, Section VIII concludes the paper.

II. Background & Challenges

We overview the basic noise control methods and explain the principles of the appropriate method for smartphones. We then discuss the challenges in achieving noise cancellation using only smartphones.

978-1-5090-2991-4/17/$31.00 ©2017 IEEE

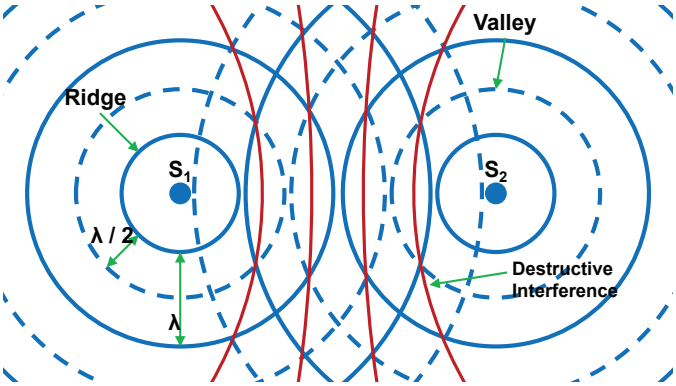


Fig. 1. An interference example when two identical acoustic waves meet.

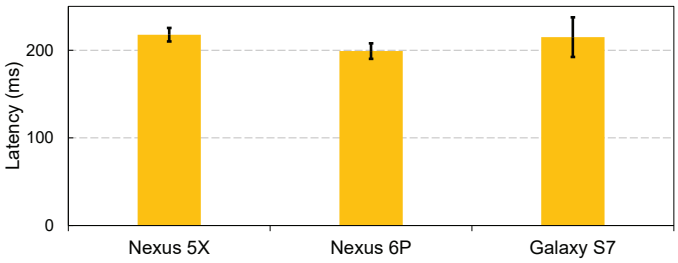


Fig. 2. Android audio round-trip latency measurement on three different models.

A. Background

Noise Cancellation: There are two main noise control methods: *passive noise control* (PNC) [2] and *active noise control* (ANC) [3]. PNC controls acoustic noise by isolating the noise source with noise-isolating materials, such as barriers, enclosures, sound-absorbing tiles, or a muffler to attenuate the unwanted noise. ANC on the other hand, controls acoustic noise by generating the exact opposite sound (i.e., the same amplitude but inverted phase) of the unwanted sound. While PNC requires aforementioned sound-isolating materials, ANC requires a microphone (mic) and a speaker to obtain the information of the unwanted sound and generate the opposite sound (cancelling sound). Given that most smartphones have built-in mic and speaker that we can leverage, ANC is our preferred method.

ANC Principles: When two identical acoustic waves meet, two types of interference occur: *constructive interference* and *destructive interference*, as illustrated in Figure 1. The two types of interference appear alternatively when the phase difference of the two sound sources is a multiple of π; the constructive interference occurs when the phase difference is an even multiple of π, while the destructive interference happens when the phase difference is an odd multiple of π. The original sound becomes louder near the constructive interference, and quieter near the destructive interference. ANC leverages *destructive interference* to cancel the unwanted sound. Note that the distance between the constructive and destructive interference depends on the wavelength of the sound source. The wavelength λ of the sound is determined by the frequency f of the sound:

$$\lambda = \frac{v}{f}$$

where v is the speed of sound (340 m/s). Higher frequency makes shorter distance between the two interference types and lower frequency makes longer distance. Thus, the silent zone created by the destructive interference is inversely proportional to the frequency of the sound. If the frequency of the noise is high, the silent zone is small; if low, the zone is large. ANC is usually applied to the sound source with the frequency lower than 500 Hz [3], so that the silent zone can cover the distance between the ears and the eardrums.

B. Challenges

We identify three key challenges the traditional ANC algorithms face in a smartphone environment.

Audio Latency: Audio latency is the time taken for an audio signal to pass through a system. Since we build our system on Android, we consider audio latency in Android, mainly caused by the audio buffer. Android typically has three audio latency components: (i) audio output latency, (ii) audio input latency, and (iii) warm up latency [12]. Audio output latency is the time interval between the generation of an audio sample by a system and the actual playback through the speaker. Audio input latency is the time between receiving an audio sample through the mic and the available state of the audio in a system. Note that we refer round-trip latency as the sum of the audio input/output latency. Warm up latency is the time taken to wake up the audio pipeline when first playing the audio. These audio latency depends on smartphone models and Android versions, and thus is difficult to estimate [13].

We measured the round-trip audio latency using the Zoiper Audio Latency Benchmark application [14]. The app generates an audio sample, listens to that sample, and calculates the time between them. We used three different smartphone models (Nexus 5X, Nexus 6P and Galaxy S7) to evaluate the latency and variance, as shown in Figure 2. The Android version was 6.0.1 in all models. All three models show around 200 msec round-trip latency, with the largest variance of 9.4 msec. We can see that Android round-trip audio latency is quite high and unpredictable, even with the same smartphone model and OS. Popular ANC algorithms such as LMS [4] and FxLMS [5] are feedback based algorithms that mimic the desired filter to cancel the noise by calculating the least mean square error. Therefore, adopting those algorithms entails audio round-trip latency and warmup latency. However, these Android audio latency cannot be tolerable in ANC systems as these systems are highly sensitive to any latency. Traditional ANC systems are thus usually designed with special hardware with low latency (sub-millisecond) [11].

Multipath: Unlike other signal processing domains that must *recover* the original signal from multipath interference, sound cancellation requires all sound signal, including those propagated through multipath, must be *cancelled*. In other words,

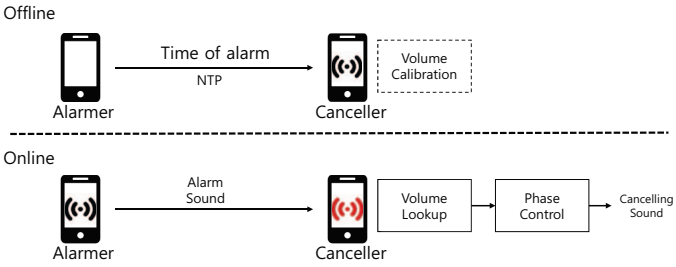


Fig. 3. Virtual Earplugs design overview.

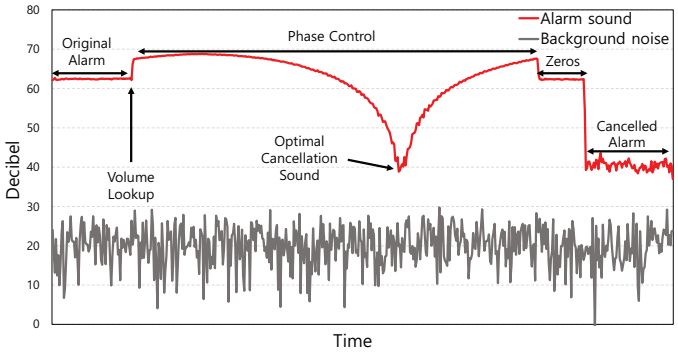


Fig. 4. Decibel variance at the canceller during the cancellation process.

an ANC system must not only cancel the unwanted sound propagated through the direct path, but also all the remaining sound propagated through multipath. Moreover, this multipath chaotically affects the cancellation pattern (Figure 1). Thus, typical room environment with multipath would be challenging for ANC systems.

No Wearables: Most available noise cancellation products in the market come with some form of wearable components [8]–[10], for both PNC and ANC. Noise cancellation is most effective with wearables, as PNC requires a noise-isolating material and ANC has a very small noise-cancelled zone. Wearable devices however require additional cost and are often uncomfortable to sleep with. Without wearable devices, ANC becomes even more challenging due to the distance between the canceller and the noise source, and also the unpredictable acoustic environment.

III. Virtual Earplugs

A. Design Overview

Our goal is to cancel unwanted alarm sound using smartphones and without any wearable device, by leveraging the ANC principles. Virtual Earplugs target a scenario where the *alarmer* generates an alarm sound and the *canceller* creates the optimal cancelling sound by analyzing the dB of the original sound through its mic. The one who must wake up has the alarmer nearby while the one who does not want to wake up at that time has the canceller nearby.

As discussed in Section II-B, directly adopting traditional ANC algorithms is impractical due to the Android audio latency. Moreover, the required iteration of the traditional algorithms depends on the prerequisite parameters (e.g., the step size or initial value of LMS), whose optimal values are dependent on the environment [15]. Instead of using the traditional algorithms, we thus design our own cancelling algorithm that (i) does not highly depend on the Android audio latency and (ii) finds the cancelling sound in a short time.

When the alarm sound is sampled and digitized through a mic in the system, there always exist a digitally optimal cancelling sound. By leveraging this fact in our design, the canceller generates the optimal cancelling sound by searching for all candidates. We present a novel sound cancellation algorithm that generates the opposite sound (i.e., the same amplitude but inverted phase) in two steps: *volume control* and *phase control*. The *volume control* process is further divided into *volume calibration* and *volume lookup*.

Figure 3 illustrates the design of Virtual Earplugs, with the alarmer and the canceller. During the offline phase (before going to bed), the alarmer notifies the time of the alarm to the canceller using network time protocol (NTP) [16], and the canceller stores the decibel values of the alarm sound through *volume calibration*. During the online phase, the alarmer starts generating the alarm sound. The canceller at the same time, finds the nearest volume of the alarm at the canceller through *volume lookup*, and generates the anti-phase sound of the original alarm sound through *phase control*. Figure 4 shows the decibel variance from the viewpoint of the canceller during the online cancelling process. First, the canceller equalizes the volume of the cancelling sound with the alarm sound by *volume lookup*. Second, the canceller finds the anti-phase through *phase control* process. Finally, the canceller reduces the decibel of the alarm sound by generating a cancelling sound.

B. Alarm Sound Selection

As discussed in Section II, to achieve alarm cancellation using smartphones, the alarm sound should be at lower than 500 Hz. Otherwise, we cannot leverage ANC and additional devices such as (real) earplugs are needed to utilize PNC. The alarm sound should also fulfill its main responsibility of waking up a human. We consider two facts: (i) around 520 Hz is the most appropriate frequency of an alarm sound [17] and (ii) the frequency of the alarm should be lower than 500Hz for cancellation, and select 500 Hz sine wave as the alarm sound in our system.

C. Volume Control

In order to successfully cancel the alarm sound, firstly, a canceller must generate a sound that (i) has the same decibel as the alarm sound and (ii) has the anti-phase of the alarm sound. Unlike the phase difference, sound pressure level (decibel) is not sensitive to latency in a given environment, unless the *acoustic environment* (an environment from the viewpoint of acoustics, usually affected by surrounding objects) changes. As sound pressure is not sensitive to latency, a canceller smartphone conducts *volume calibration* once before the alarm

begins (e.g., volume calibration could be done before going to bed). Note that once the canceller conducts volume calibration, there is no need for further calibration under the same acoustic environment.

Algorithm 1 Volume Control

```
procedure VOLUME CALIBRATION
    N ← total number of search
    step_size ← 1/N
    for i ∈ {0, ..., N − 1} do
        vol_i ← i * step_size
        Play audio sample with vol_i
    Wait until the end of playbacks
    for i ∈ {0, ..., N − 1} do
        dB_i ← dB of the i^th audio sample
        Store the paired value: {vol_i, dB_i}
procedure VOLUME LOOKUP
    dB_o ← measured dB of the alarm sound
    N ← total number of search
    min ← ∞
    best_vol ← ∞
    for i ∈ {0, ..., N − 1} do
        Load the paired value: {vol_i, dB_i}
        if |dB_o − dB_i| < min then
            min ← |dB_o − dB_i|
            best_vol ← vol_i
    return best_vol
```

Algorithm 1 summarizes the volume calibration process. The canceller generates the cancelling sound with increasing volumes. The canceller then measures the dB values of the sounds at the canceller's mic and stores all dB data as paired values with the volume. Android volume control methods (e.g, *MediaPlayer.setVolume(x)*) require a floating value, which usually ranges from 0.0 to 1.0. Note that infinite number of volumes can be generated and stored in the volume calibration process. Therefore, dividing the volume range into certain number of block is common [18]. There is a trade-off; choosing a higher number of pairs to be stored takes longer time than a lower number, while the sound pressure difference between the alarm and the canceller is smaller. We empirically chose 200 as the total number of search and thus the volume step size is $\frac{1}{200}$. With 200 paired values, the difference between the dB of the alarm sound and the nearest dB of the canceller is about 0.13 dB, while the volume calibration process takes 4040 msec.

Once volume calibration is complete for the given environment, the canceller is ready to find the optimal cancelling sound volume to generate. As shown in the *volume lookup* procedure in Algorithm 1, when the alarm starts, the canceller looks up the previously stored paired values and searches for the nearest volume to the alarm sound. Finding the proper volume after the calibration could be done instantly, as it is a simple calculation of 200 comparisons.

D. Phase Control

As discussed in Section II-B, Android round-trip audio latency is very high and with variance, even within the same smartphone. Moreover, latency highly depends on smartphone models and Android versions [13]. Therefore, it is difficult to adopt feedback-based algorithms such as LMS [4] and FxLMS [5]. We instead adopt a novel phase control method that is not affected by the Android audio latency.

The alarm sound waves propagates through the air and all the multipath are combined as a single sine wave at the canceller's mic. For example, if two sine waves of the same frequency are combined, they become another sine wave of the same frequency. If two sine waves can be represented by one sine wave, we can say that all the sounds from multipath are represented by one sine wave. We now show the combination of two sine waves can be represented as a single sine wave. Suppose we have two sine waves of the same frequency ω, with different amplitudes A and B, and phase difference of ϕ. The two sine waves then can be written as follows:

$$A\sin(\omega t) + B\sin(\omega t + \phi). \tag{1}$$

By using the following equation (2) to expand equation (1), the result is expressed as equation (3):

$$\sin(\alpha+\beta) = \sin(\alpha)\cos(\beta) + \cos(\alpha)\sin(\beta), \tag{2}$$

$$(A + B\cos\phi)\sin(\omega t) + (B\sin\phi)\cos(\omega t). \tag{3}$$

For simplicity, we substitute the constants with the following expression:

$$\begin{cases} C = A + B\cos\phi, \\ D = B\sin\phi. \end{cases} \tag{4}$$

The above equation (3) is then:

$$C\sin(\omega t) + D\cos(\omega t) \tag{5}$$

which can be re-written as:

$$\sqrt{C^2+D^2}\left(\frac{C}{\sqrt{C^2+D^2}}\sin(\omega t) + \frac{D}{\sqrt{C^2+D^2}}\cos(\omega t)\right). \tag{6}$$

We now define θ as:

$$\theta = \tan^{-1}\frac{D}{C}. \tag{7}$$

Equation (6) then is re-expressed as:

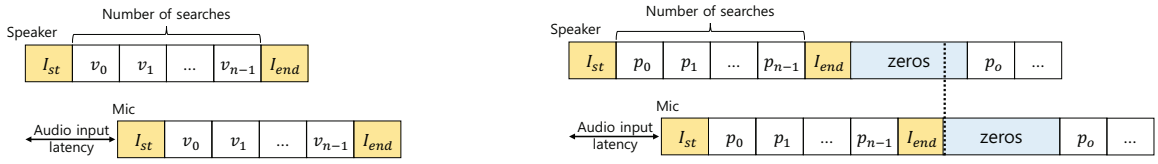
$$\sqrt{C^2+D^2}\left(\cos(\theta)\sin(\omega t) + \sin(\theta)\cos(\omega t)\right). \tag{8}$$

By using equation (2), we finally express equation (1) as a single sine wave:

$$\sqrt{C^2+D^2}\sin(\omega t + \theta). \tag{9}$$

We can see that the amplitude and the phase difference have changed, but we can express the summation of two sine waves with the same frequency as a single sine wave.

With the above equations, we need not cancel each part of the alarm sound but rather, only one sine wave, which



(a) Audio format of volume calibration.

(b) Audio format of phase control.

Fig. 5. Audio format of (a) volume calibration and (b) phase control. In each process, the delay of getting audio samples from the mic is expressed as audio input latency. There are zero-padded audio samples in the phase control process.

has unknown amplitude and phase shift at the mic. As the alarm sound will be combined as a single sine wave at the canceller, there exists an optimal anti-phase sine wave that cancels the alarm sound. For example, since we set the default frequency of alarm sound as 500 Hz, one period is 2 msec ($f = 1/T$). Under the 48 kHz sampling rate, which is a common sampling rate of recent smartphones, there are a total of 96 samples in one period; thus, 96 possible phase shifts. Suppose the alarm sound is $sin1000\pi x$, then the canceller's sound is $sin1000\pi(x - p_i)$ after equalizing the amplitude to the alarm, where p_i is possible phase shifts where $i \in [0, 95]$. The optimal phase shift p_o to successfully cancel the alarm sound hence always lies in [0, 95].

Algorithm 2 Phase Control

```
1:  procedure PHASE CONTROL
2:      N ← total number of samples in a period of the alarm sound
3:      alarm ← audio samples of the alarm sound
4:      Set the volume of alarm by Volume Lookup procedure
5:      min ← ∞
6:      anti_phase ← ∞
7:      for i ∈ {0, ..., N − 1} do
8:          phase_i ← shifted samples of alarm by i
9:          Play cancelling samples with phase_i
10:     Wait until the end of playbacks
11:     for i ∈ {0, ..., N − 1} do
12:         dB_i ← dB of the i^th audio sample
13:         if dB_i < min then
14:             min ← dB_i
15:             anti_phase ← phase_i
        return anti_phase
```

In order to find the optimal phase shift p_o, the canceller tries all possible phase shifts (in the case of 500 Hz, 96 possible shifts) by shifting the audio sample data to different phase shift p_i at the canceller's mic and recording the dB changes. The canceller then plays the cancelling sound with the optimal phase shift p_o. We summarized the phase control process in Algorithm 2. First, the canceller sets the cancelling sound volume with the *Volume Lookup* procedure of Algorithm 1, in line 4. In lines 8 and 9, the phase-shifting and sample generation are shown. The samples represent each phase shift p_i. After finding the best anti-phase, it returns the audio sample with the anti-phase.

E. Audio Format

Mitigating Audio Latency: Figure 5 shows the audio format used in volume calibration and phase control. In both processes, there is a time delay, called *audio input latency*, between the playback of the canceller's audio and the availability of the audio received from the canceller's mic. After the canceller obtains the dB values from the mic, identifying the index i of dB_i is challenging due to the audio input latency. To solve this latency problem, we attached additional inaudible sound signals (i.e., I_{st} and I_{end}) at the end of both formats. We used 22 kHz as the inaudible signal frequency, given that over 20 kHz sound is almost inaudible to human [19]. With this mechanism, we obtain the index of the signal by getting the time interval between the two peaks at the 22 kHz in the recorded audio and dividing it into the total number of indices, as expressed in the following:

$$dB_i = \frac{I_{end} - I_{start}}{N} * i$$

where I_{start} and I_{end} represent the start and the end time of the inaudible signals, respectively, and N is the total number of indices ($N = 200$ in volume calibration and $N = 96$ in phase control).

Symbol Duration: Symbol duration determines the entire processing time of the cancellation; a shorter duration brings shorter cancellation time, and longer duration brings longer cancellation time. To monitor 500 Hz decibel, the canceller needs continual Fourier Transform to acquire the frequency information of sound. We used Short-Time Fourier Transform (STFT) library [20], which requires 512 samples for one STFT. This number of required samples consumes around 5 msec to collect enough audio data for one STFT under 48 kHz sampling rate. We set the duration of each symbol (i.e., I_{st}, I_{end}, v_i and p_i) to 20 msec to collect enough dB data. With this duration, the canceller obtains 3∼4 data of 500 Hz decibel for each symbol. We used the average data as a representative value for each symbol.

Zero Padding: In phase control, the canceller should continuously generate the optimal cancelling sound after finding the anti-phase. Due to the Android audio input latency however, the canceller cannot consider the last several dB values when the canceller tries to find the optimal cancelling sound right after playing I_{end}. We thus add additional zero-padded audio, lasting for about 200 msec (Figure 5(b)) to avoid the effect of the audio input latency. The dotted line represents the

time when a canceller can obtain the best anti-phase from all measured dB values at the mic. Note that the zero-padded audio should be longer than the audio input latency, so that all the phase shift p_i can be measured through the mic. The zero-padded audio stream does not include the actual "sound," and thus would not bother anyone. After waiting the zero-padded audio stream, the canceller starts continuously generating the optimal cancelling sound.

Finally, the total cancellation processing time (in msec) is:

$$D \times (2 + N) + 200$$

where D is the symbol duration, 2 is the number of inaudible signals, N is the total search number, and 200 is zero padding.

IV. Implementation

We implemented the alarmer on Nexus 5X and the canceller on Galaxy S7, respectively. Both the alarmer and the canceller run Android 6.0.1. We used Android AudioTrack class to generate the alarm and the cancelling sound from the audio data. We utilized the the audio spectrum analyzer [20] to monitor the dB of the target frequency of the alarm (500 Hz) and the inaudible signal (22 kHz). The audio sampling rate is 48 kHz, which is commonly used in recent smartphones. We use NTP to synchronize the time of the alarm between the canceller and the alarmer during the offline phase. Before the start of the alarm, each device synchronizes the time with the same NTP server, and sends the alarm time to the canceller via Bluetooth. Just before the time of the alarm, the canceller starts monitoring the mic and when the alarm sound is detected, starts the cancellation process.

In order to eliminate the effect of different acoustic environments, we set the default environment (lab environment) as shown in Figure 7(a). Most experiments are done in the lab environment, except for the evaluation on Section V-B where we use various environments. In the lab environment, the distance between the alarmer and the canceller is one meter. We measured the dBA (the relative loudness of sounds in the air as perceived by the human ear) through ARCO AR-824 decibel meter. We set the volume of the alarm sound to the loudest, which is measured to be around 82.0 dB at the alarmer and 62.5 dB at the canceller. Both smartphones were placed facing up.

V. Evaluation

With our prototype implementation, we evaluate the feasibility of achieving active noise cancellation using only smartphones without wearables. Specifically, we examine (i) the tradeoff of search space and delay, (ii) the size of two-dimensional alarm sound cancelled zone in three different environments, (iii) the effect of different frequencies of the alarm, and (iv) the performance gain with an alternative design using an external mic.

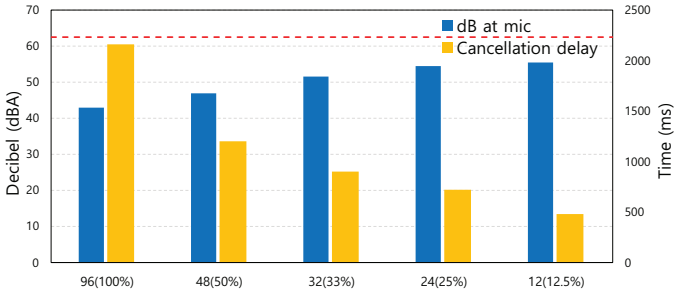


Fig. 6. Tradeoff between the search space and cancellation delay.

A. Impact of Search Space

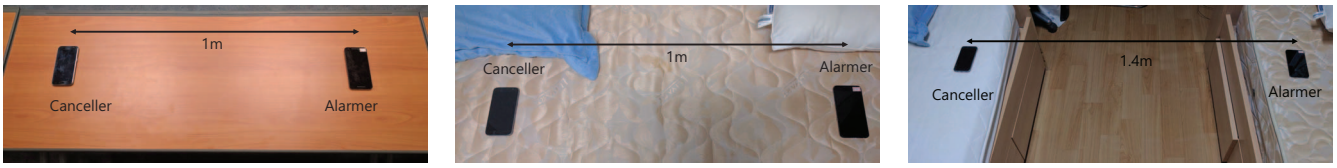
As discussed in Section III-D, 500 Hz sound has 96 samples per period under the 48 kHz sampling rate. Therefore, there are 96 phase shift candidates. Note that the reduction of dB and the required delay in the cancellation process depend on the search space. To investigate the relation between the search space and the performance of alarm sound cancellation, we experiment with different search space in the lab environment.

Figure 6 shows the result of each search space that is averaged over 20 measurements. The red-dotted line represents the dB of the original alarm (62.5 dB) before the cancellation. As the number of search decreases, the required cancellation delay also decreases, while the dB of the cancelled sound increases (i.e., cancelled less). When the search space is 96 (100% of the total search space), the cancelled sound is about 42.9 dB, which is 19.5 dB decreased from the original alarm sound. In terms of cancellation latency, it requires 2160 msec.

When searching was done at every other samples of the total search space (i.e., search space 48), the cancelled sound is 46.9 dB and cancellation delay is 1200 msec. As expected, there is a trade-off; when the search space is small, the required cancellation delay reduces while the dB of sound level increases. Note that both the decibel and the duration of sound affect sleep disturbance [21]. Previous studies on sleep disturbance induced by noise set the default noise duration to at least 2 sec [22], and if less than 1 sec, there is almost no effect [21]. We hence set our search space to 96 (which in consequence sets the cancellation delay to 2160 msec) in the following experiments, so that we can maximize the cancellation effect.

B. Cancelled Zone

Since the canceller refers its built-in mic for the cancellation process, the cancelled dB is optimal at the canceller's built-in mic. However, the cancellation effect would be different to the user near the canceller. The canceller thus should create a *cancelled zone* (i.e., the area of reduced alarm sound after cancellation) to effectively provide cancellation to the user. In addition, the zone should be robust in practical settings, i.e., common sleep scenario in a room that has multipath, obstacles, etc. We hence measure two-dimensional cancelled zone to answer the following two questions:



(a) Lab environment. (b) Practical environment I: On a bed. (c) Practical environment II: On two beds.

Fig. 7. Experimental environments.

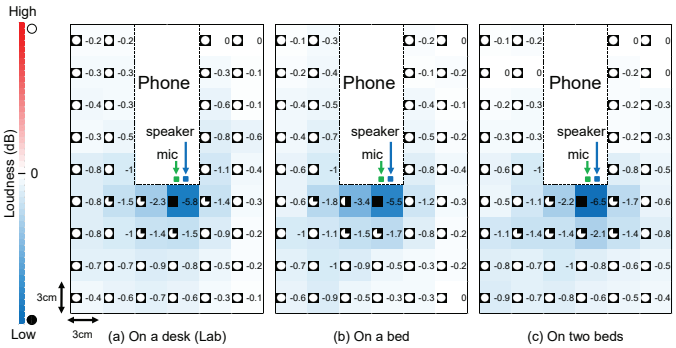


Fig. 8. Cancelled zone on three different environments.

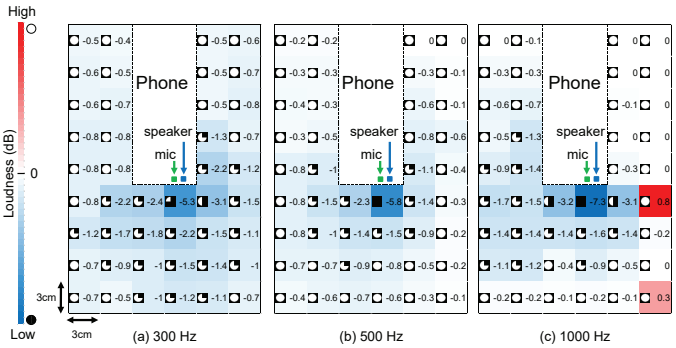


Fig. 9. Cancelled zone with different frequencies (300, 500 and 1000 Hz).

- What is the size of the cancelled zone and the amount of noise reduction of those areas?
- How does the room environment affect the cancelled zone?

We measure the cancelled zone on three different environments, as shown in Figure 7. Figure 7(a) is the default lab environment on a desk with less obstacles. The other two environments are where smartphones are placed in bed; Figure 7(b) represents the scenario where the alarmer and the canceller are on the same bed (i.e., a couple sharing a bed), and Figure 7(c) represents the alarmer and the canceller are on the different beds (i.e., roommates in a dorm room). The distances between the alarmer and the cancer in three environments are 1, 1 and 1.4 meter away, respectively. In all the environments, we set the volume of the alarm to maximum, which are measured as 62.5, 44.0 and 41.0 dB at the canceller for each environment. The difference between the magnitude of the alarm at the canceller is mainly caused by the absorption of sound.

The result of 2D cancelled zone for each environment is shown in Figure 8. We measured the dB difference relative to the dB of the original alarm sound, and recorded the average of the five measurements for each 3×3 cm^2 grid. For all environments, the sound is reduced by around 19 dB at the canceller's mic. If the color of the grid is closer to blue, the more effective is the cancellation. On the other hand, if the color is closer to red, cancellation was less effective.

Overall, the alarm cancellation was at its best near the canceller's mic (around 6 dB reduction at the nearest grid). This result is similar to traditional ANC systems: 6 dB reduction when 1.5 cm away from the mic [23]. Compared with the cancellation effect at the mic, dBs around the canceller are less effective. One important reason is that the mic is *inside* the canceller, and thus the cancellation is optimized for the canceller's mic. We perform further measurements on this issue in Section V-D. The cancelled zone is affected by many factors, including the attenuation of sound with distance, multipath, floor material, smartphone model, obstacles, etc. Interestingly, the cancelled zone looks similar across different environments.

C. Different Frequencies

As discussed in Section II, ANC works well on frequencies below 500 Hz because of the longer wavelength, which directly affects the cancellation zone. To examine the relation between the frequency and the cancelled zone, we measured the cancelled zone on 300 Hz and 1000 Hz sine waves, as shown in Figure 9. Again, the average of five dB measurements is shown in each grid. The zone in Figure 9(b) is the same as in Figure 8(a). The cancelled zone under the 300 Hz alarm has a wider area than 500 Hz. On the other hand, the zone of the 1000 Hz alarm shows a smaller area than 500 Hz.

Moreover, the zone of the 1000 Hz alarm has steeper changes in dB than lower frequencies. There are even increased grids (0.8 and 0.3 dB), caused by unexpected constructive interference stemming from the short wavelength. To summarize, lower frequency alarm shows a wider zone while higher frequency alarm shows a smaller zone. Still, the 500 Hz alarm sound is effective for its cancellation zone as well as the ability to wake up the intended target [17].

D. External Mic

Our scenario assumes that the canceller is near the user's ear, which could maximize the cancellation effect near the user. However, it requires user proximity to the canceller

Fig. 10. External mic design.

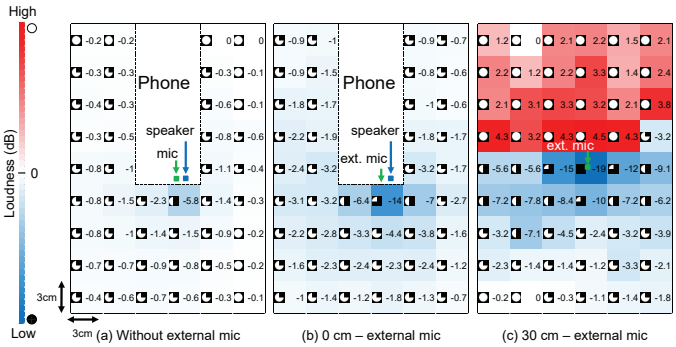


Fig. 11. Cancelled zone on the external mic design with different distance between the mic and the canceller ((b) 0 and (c) 30 cm).

due to the small cancellation zone. We thus evaluate another design using an *external mic*, to overcome the short distance requirement between the canceller and the user. We added an external mic to the canceller as shown in Figure 10. The canceller cancels the alarm sound with respect to the external mic, instead of the built-in mic, and thus the cancellation effect is optimized for the external mic.

The resulting zone is shown in Figure 11. The average of five dB measurements is shown in each grid. Figure 11(a) is the same as in Figure 8(a). First, to compare the difference between using the built-in mic and the external mic, we placed the external mic at the same position as the built-in mic. Figure 11 (b) shows the cancelled zone for that setting. Interestingly, the cancelled zone is wider than the case with a built-in mic. This is mainly because of different acoustic environments between inside and outside of the smartphone; when using the built-in mic, the cancellation effect is optimized for inside the phone. This result indicates that the canceller can mitigate the problem of small zone by using an external mic. Note that there is a trade-off; basic design requires no additional hardware, while the alternative design requires an external mic that creates a broader cancelled zone.

The main purpose of the external mic design is to position the cancelled zone by moving the mic. A user can put the external mic near the user's ear, to maximize the cancellation effect. However, if the mic is certain distance away from the speaker, the secondary path effect increases [24]. To evaluate the effect of the distance between the mic and the speaker of the canceller, we measured the cancelled zone with the mic 30 cm away from the canceller. Figure 11 (c) shows the result where the cancelled zone is broader and the cancellation effect is larger than other settings. Interestingly, the alarm sound at the upper half is amplified. This is due to the louder volume of the canceller. As the distance between the speaker and the mic is longer than other experiments, the canceller must generate louder cancelling sound to reach the external mic. The multipath of the louder cancelling sound can then spoil the zone; the impact of multipath is larger than the shorter distance case. As a result, one should be cautious of the unexpected amplified alarm sound when using an external mic, even though it can create broader cancellation zone and further reduce the alarm sound.

VI. Limitations & Future Work

We discuss the limitations of our current design and possible future directions.

Human Perception: Aside from the reduced decibel of the original alarm, human perception on the cancelled alarm sound might be different; it depends not only on the reduced decibel, but also on subjective sensitivity per human (called psychoacoustics). Conducting a user study could help understand the human perception of the cancelled alarm sound. Furthermore, one speaker is not sufficient to perfectly cancel the alarm sound as humans have two ears. We believe cancellation for two ears could be possible by leveraging two speakers in a smartphone or other devices with a speaker.

Different Alarm Sound: Our current implementation uses a simple sine wave as the alarm sound. However, one might prefer to use other preferred alarm sounds. As we investigated in Section V-D, the frequency of the alarm sound affects the cancelled zone; an alarm sound of high frequency degrades the cancelled zone. Theoretically and practically, high frequency sound above 500 Hz is difficult to cancel by ANC principles. It might require sound-isolating materials to leverage the passive noise control (PNC) effect.

Environmental Change Tolerance: Our current design focuses on generating the optimal cancelling sound given an acoustic environment. However, an environment could change during the cancellation process (e.g., when a user turns off the alarm, when a user tosses and turns during sleep, etc.). Environmental changes can be detected by monitoring the decibel at the canceller, and using a simple threshold-based approach could work; re-cancelling when the canceller detects sudden dB increases.

VII. Related Work

We summarize prior works in two aspects: active noise control (ANC) and audio synchronization.

Active Noise Control: Many ANC techniques have been proposed and deployed. The most widely used algorithms are Least Mean Square (LMS) [4] and Filtered-x LMS (FxLMS) [5] (and their variants). These are feedback-based algorithms that mimic the desired filter to cancel the noise by calculating the least mean square error. These algorithms use gradient descent method to cancel the noise. FxLMS has one more filter to mitigate the secondary path effect, compared to the LMS algorithm. However, these traditional solutions do not work well with commodity smartphones. Since these algorithms are based on a feedback loop,

the round-trip audio latency is required for every iteration. Moreover, these algorithms require initial parameters, the step size and the initial filter. Setting the parameters affects the required number of iterations to converge, and the optimal values depend on the environment [15].

Audio Synchronization: Instead of ANC, one might suggest synchronizing the alarmer and the canceller, and when the alarm starts, the canceller simultaneously generates the exact anti-phase of the alarm. This requires audio synchronization between the alarmer and the canceller. Previous works examined the audio synchronization with smartphones: using the preamble symbol [25] or leveraging the impulse response of the audio signal [26]. Both studies showed around 100 μsec error in audio synchronization. Note that ANC requires precise time synchronization. A slight time difference has a huge effect; 100 μsec difference represents five sample errors in phase control, which is very apart from the optimal cancelling sound.

VIII. Conclusion

We presented Virtual Earplugs that use only commodity smartphones, without wearable devices or specialized hardware, to reduce unwanted alarm sounds. We highlighted practical challenges in applying traditional active noise cancellation techniques in our setting, and presented solution to overcome some of the issues. Our prototype evaluation demonstrated the feasibility; Virtual Earplugs reduce the alarm sound by up to 19 dB in only 2.1 seconds of processing delay. While some challenges still remain to fully realize our vision and widely deploy our system in practice, we believe our efforts signify the possibility of utilizing technologies with only commodity devices to enhance user experience.

Acknowledgements

We thank Jung-Woo Choi and Bowon Lee for sharing their expertise in audio signal processing. We also thank Jinyeob Kim, Sunwoo Kim and Chunjong Park for participating in early parts of this research. This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No.2016R1A2B4014068) and Institute for Information & Communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (No.B0717-16-0034, Versatile Network System Architecture for Multi-dimensional Diversity).

References

[1] L. Paul, "Process of silencing sound oscillations," Jun. 9 1936, US Patent 2,043,416.

[2] R. J. Bernhard, "The state of the art of active-passive noise control," in *Noise Con 1994: Proceedings of the 1994 National Conference on Noise Control Engineering*, vol. 1, 1994, pp. 421–428.

[3] S. J. Elliott and P. A. Nelson, "Active noise control," *IEEE signal processing magazine*, vol. 10, no. 4, pp. 12–35, 1993.

[4] S. Haykin and B. Widrow, *Least-mean-square adaptive filters*. John Wiley & Sons, 2003, vol. 31.

[5] M. T. Akhtar and W. Mitsuhashi, "Improving performance of fxlms algorithm for active noise control of impulsive noise," *Journal of Sound and Vibration*, vol. 327, no. 3, pp. 647–656, 2009.

[6] W. S. Gan and S. M. Kuo, "An integrated audio and active noise control headset," *IEEE Transactions on Consumer Electronics*, vol. 48, no. 2, pp. 242–247, 2002.

[7] S. M. Kuo and R. Gireddy, "Real-time experiment of snore active noise control," in *Proceeding of CCA*. IEEE, 2007, pp. 1342–1346.

[8] "Quieton, find your silence," http://www.quieton.com/.

[9] "Quite life: Silent partner smartmask, comfortably quiet snoring sound." https://quietlife.tech/sale1.html.

[10] "Vitalsleep: Finally a good night's sleep for you and your partner." http://www.vitalsleep.com/.

[11] D. Massie and J. Laroche, "Low latency active noise cancellation system," Sep. 30 2014, US Patent 8,848,935.

[12] "Guides for android audio latency," https://developer.android.com/ndk/guides/audio/audio-latency.html.

[13] "Audio latency measurements," https://goo.gl/LU3KEd.

[14] "Zoiper Audio Latency Benchmark." https://play.google.com/store/apps/details?id=com.zoiper.audiolatency.app.

[15] S. M. Kuo and D. R. Morgan, "Active noise control: a tutorial review," *Proceedings of the IEEE*, vol. 87, no. 6, pp. 943–973, 1999.

[16] D. L. Mills, "Internet time synchronization: the network time protocol," *IEEE Transactions on communications*, vol. 39, no. 10, pp. 1482–1493, 1991.

[17] Maria, "Low frequency requirements: When, where, and why?" http://www.systemsensorblog.com/2013/11/low-frequency-requirements-when-where-and-why/.

[18] M. Azizyan, I. Constandache, and R. Roy Choudhury, "Surroundsense: mobile phone localization via ambience fingerprinting," in *Proceedings of MobiCom*. ACM, 2009, pp. 261–272.

[19] L. Deshotels, "Inaudible sound as a covert channel in mobile devices." in *WOOT*. USENIX, 2014.

[20] E. Xiao, "Audio spectrum analyzer for android." https://github.com/bewantbe/audio-analyzer-for-android.

[21] K. S. Pearsons, D. S. Barber, B. G. Tabachnick, and S. Fidell, "Predicting noise-induced sleep disturbance," *The Journal of the Acoustical Society of America*, vol. 97, no. 1, pp. 331–338, 1995.

[22] S. Fidell, K. Pearsons, B. Tabachnick, R. Howe, L. Silvati, and D. S. Barber, "Field study of noise-induced sleep disturbance," *The Journal of the Acoustical Society of America*, vol. 98, no. 2, pp. 1025–1033, 1995.

[23] R. Serizel, M. Moonen, J. Wouters, and S. H. Jensen, "A zone-of-quiet based approach to integrated active noise control and noise reduction for speech enhancement in hearing aids," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 6, pp. 1685–1697, 2012.

[24] I. T. Ardekani and W. Abdulla, "Fxlms-based active noise control: A quick review," in *Asia Pacific Signal and Information Processing Association Annual (APSIPA) Summit and Conference, Xi'an, China*, 2011.

[25] Q. Wang, K. Ren, M. Zhou, T. Lei, D. Koutsonikolas, and L. Su, "Messages behind the sound: real-time hidden acoustic signal capture with smartphones," in *Proceedings of MobiCom*. ACM, 2016, pp. 29–41.

[26] H. Kim, S. Lee, J.-W. Choi, H. Bae, J. Lee, J. Song, and I. Shin, "Mobile maestro: enabling immersive multi-speaker audio applications on commodity mobile devices," in *Proceedings of UbiComp*. ACM, 2014, pp. 277–288.