Reliable Short-Distance Data-Transmission Mechanism Using Inaudible High-Frequency Sound

Insu Jang[1], Myoungbeom Chung[2], and Hyunseung Choo[1]
[1]*College of Information and Communication Engineering, Sungkyunkwan University, Gyeonggi-do 440-746, Republic of Korea*
[2]*Division of Computer Engineering, Sungkyul University, Gyeonggi-do 430-742, Republic of Korea*

Abstract. This paper proposes a data communications method using the on-off keying technique and inaudible high-frequency sound signals. The proposed method can be used to send multiple bits simultaneously by mixing specific frequencies that are mapped to each bit. To identify data-transmission errors due to interference from nearby frequency signals, cyclic redundancy check (CRC) technique is used. Applications had been built to test performance in terms of data-transmission speed and accuracy, and the results were compared with those of the existing system. At the setting that showed the highest data transmission, our system sent 40 bits of data in 92 ms, 30 times faster than the existing system, and detected every error. This method can be applied to various environments by changing variables and can be used to send a large amount of data with high accuracy compared to the existing system. Therefore, wireless communications with inaudible high-frequency signals that are used in only limited areas because of low speed will now be able to be utilized in more diverse areas.

Keywords: Inaudible high frequency, Data transmission, Smart device, Cyclic redundancy check, Signal processing

1. Introduction

In the past five years, there have been significant advances in mobile devices and wireless communications. The emergence of smartphones and tablet PCs has enabled us to complete different tasks while in transit, and wireless communications technologies such as near-field communication (NFC) and Bluetooth embedded in mobile devices have enabled the development of electronic-payment and file-transfer systems that take advantage of short-distance communications. However, devices cannot communicate using these technologies unless each device has the required additional modules and they complete an additional “pairing” process to connect [15]. Bluetooth, specifically, may behave incorrectly depending on the host operating system (OS); for example, Apple smartphones running iOS cannot communicate via Bluetooth with smartphones running Android. To solve these problems, many researchers, such as Bihler, Chung, and Kim, have conducted studies about close-range communications using inaudible frequencies higher than 18 kHz.

Pascal Bihler studied the use of inaudible frequencies to help museum visitors understand works and designed the Smart Guide system, which transmits 8 bits of data in 208 ms [3]. Although high inaudible frequencies were used, a noise occurred by rapidly changing frequencies, thereby decreasing correctness. After this research, Kim proposed using these frequencies and users’ smartphones to authenticate users rather than requiring a separate security token [9], but it took long to transmit the signal; hence, this would not be effective as a data communications method. Chung proposed a wireless-control technology, modifying the signal-processing method proposed by Bihler [6]. It showed a 96% success rate at a 7-m distance, but it was optimized for controlling devices and is inappropriate for data communications. For these reasons, research regarding the use of inaudible frequencies as carriers of wireless data communications has been limited to only a few fields, such as transferring trigger signals or sending authentication keys.

Therefore, this paper proposes a new mechanism with a higher transmission speed to use wireless communications with inaudible high frequencies in more fields. A sending device maps each bit in the data to a specific frequency, creates a signal by mixing mapped frequencies according to the data values, and transmits the signal. A receiving device checks inaudible high frequencies and can extract data by decoding the bit values mapped with the signal frequencies. A CRC is used to detect transmission errors. This method can adjust the value of variables. That is, it has the advantage that it can control a balance between transmission speed and precision. To prove its feasibility, experiments were performed using diverse variables. Forty bits were transmitted in 92 ms at the setting for high speed, and the success rate was about 99.8% at the setting for high accuracy. We compared this performance with that of the Smart Guide system proposed by Bihler. The transmission speed of our method was 30 times higher than that of the Smart Guide system, and the accuracy of our method was 100%, proving that it has great ability for transmitting data. Therefore, more fields can use this wireless communications technology using inaudible high frequencies.

This paper is organized as follows; in section 2, we explain the existing research regarding wireless communications technologies with inaudible high frequencies. In section 3, we define variables to allow this method to be used in various conditions and explain an architecture, encoding process, and decoding process. In section 4, we implement this method into applications and measure performance using diverse variables. We also implement the Smart Guide system into application and measure performance of it. Lastly, in section 5, we report and compare the performance results and suggest a future research direction.

2. Previous work

In this section, we explain how existing studies have used inaudible high frequencies in data communications. Bihler proposed the Smart Guide system, which transmits information to smartphone users in a museum using inaudible high frequencies. This system runs on Android operating systems, receives high frequencies that are emitted from speakers installed in the museum through microphones built in users' smartphones, and gets appropriate information from a Web server. This method uses the frequency shift keying (FSK) technique with 20 kHz and 22 kHz frequencies as the signaling bits. It generates one frequency in 26 ms to send one bit, which means it sends 8 bits in 208 ms, and uses a Hamming code to reduce the error rate. An 8-bit 3.2 MHz Freescale microcontroller and a simple piezo speaker are used as the hardware to send data. The use of this system resulted in several incorrect data being sent and noise being heard when high frequencies were converted in a short time.

Chung proposed a new algorithm to control smart devices remotely by utilizing the signal transmission method of data communications systems proposed by Bihler. This system sends three frequencies at the same time by combining one control signal and two base signals, 19 kHz and 22 kHz. A receiver accepts a command when the control signal is recognized over a predefined time period called the *threshold*. This method can control devices normally even if a portion of the signal is lost or noise is introduced. Even though this method showed a 100% success rate within a 2-m distance in both indoor and outdoor settings without considering any noise, it focused on controlling devices only and is therefore not appropriate for communicating data.

Kim proposed a method for authenticating smartphone users using high inaudible frequencies, with one frequency being generated at each stereo channel. When receiving a bit that is represented by a combination of two frequencies four times, this method recognizes it as one challenge, but it takes 8 s to recognize a 2-byte challenge. Although this method uses smartphones as hardware tokens and thereby minimizes inconvenience, it is too slow to be used for data communications.

As previously mentioned, existing studies have included the invention of wireless data communications methods that use inaudible high frequencies to send small amounts of data; however, they have data-transmission limitations and cannot be used in various areas. Therefore, if we develop a new technology with a high data-transmission speed, wireless communications using inaudible high frequencies will be able to be used in more diverse fields.

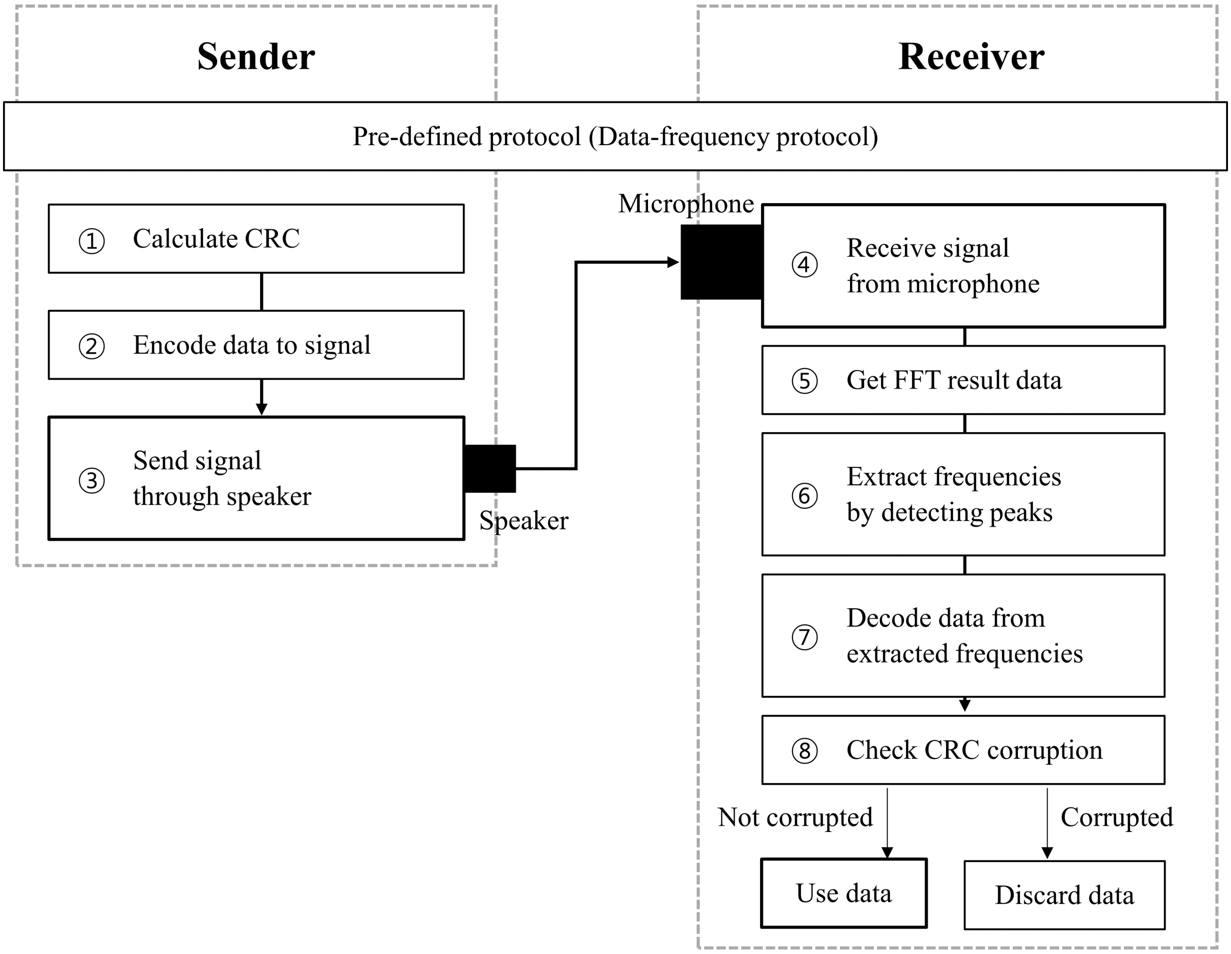

Figure 1. A general architecture of the proposed method

3. Data transmission mechanism

In this section, we propose a new wireless data communications method based on a signal mixed with several high frequencies using speakers and microphones of smartphones. A sender device sends a signal consisting of a mixture of inaudible frequencies, each of which represents one bit of data, and a receiver device restores a bit of data by decoding the signal that it receives through its microphone. A CRC technique is used to detect errors during recognition procedures. This method can transfer data based on sound signals with high accuracy and high transmission speed.

3.1. Overview of the proposed method's architecture

Speakers and microphones equipped in recent smart devices can generate or recognize sound frequencies up to 24 kHz. Therefore, the proposed method sends data using a portion of the 18–24 kHz area that humans cannot hear [20, 21]. Figure 1 represents a general architecture of the proposed method. As shown in Figure 1, the sender converts data to an inaudible high-frequency signal and sends it, and a receiver receives the signal and converts it to data. The sender and the receiver define a data-frequency protocol (DFP) that is necessary to communicate with each other in advance. The sender generates CRC data and combines it into the data to send (①), encodes a whole bit sequence to the mixed frequency signal using the DFP (②), and emits this signal through the speaker (③).

The receiver receives through the microphone the mixed frequency signal that the speaker sent (④) and decodes the data through the following processes: First, it figures out which frequencies are contained in the signal by using the fast Fourier transform (FFT) technique (⑤); it extracts valid frequency values through a peak detection procedure (⑥); it decodes from the frequency data to the bit data based on the predefined DFP (⑦); and it checks whether there are errors in the decoded data using the CRC (⑧) and either discards the data if it is corrupted or uses the data if it is correct.

3.2. Procedures for encoding data from bits to inaudible frequency sound signals

The sender encodes the data into an inaudible high-frequency signal using the DFP. Each bit in a data is mapped with specific frequencies that are different from each other and has a guard bandwidth at each end of its mapped frequency to avoid an inference among bits. Therefore, as in Figure 2, the frequency that represents a particular bit is the center value of the guard bandwidth.

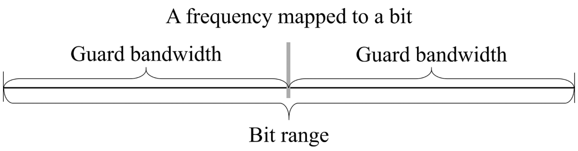

Figure 2. How bit range is determined and how a bit is mapped to a frequency

The bit range is the length of the bit's guard bandwidth. For example, if we set the guard bandwidth to 50 Hz, the bit range would be 100 Hz, and if this bit range is the section from 18 kHz to 18.1 kHz, the frequency that is mapped with this bit is the center value, 18.05 kHz.

Each bits composing one set of data has the same bit range, and the bits are arranged sequentially in the whole frequency area. That is, if the bit range is 100 Hz and the length of data is 10 bits, the frequency area would have a required length of 1 kHz and could be placed in any area in the 18–24 kHz range, such as 18–19 kHz. The area that is chosen is called the frequency range. Figure 3 depicts a frequency range of 18–19 kHz, which is divided into ten sections by thin black vertical lines. The gray lines depict the center values of each section and indicate the positions of the frequencies that are mapped to corresponding bits.

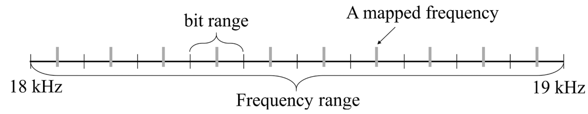

Figure 3. 18–19 kHz frequency range with bit range 100 Hz that is used to send ten bits of data

Both the bit range and the frequency range should be defined to transmit data successfully. Hence, we define the *DFP* equation to cover both of them and represent it as Eq. (1).

$$DFP = (B_r, f_r) \quad (1)$$

B_r is the bit range, and f_r is the frequency range. If the bit range is 100 Hz and the frequency range is 18–19 kHz, for instance, *DFP* is represented as *DFP* = (100 Hz, 18–19 kHz), and the length of this data should be 10 bits.

Next, data encoding can be done by mixing only frequencies that are mapped with bits of value 1. For example, if the 10 bits of data are 1100010010, the encoded signal is a combination of frequencies that are mapped to the first, second, sixth, and ninth bits. In the case of *DFP* = (100 Hz, 18–19 kHz), the first bit represents 18050 Hz; the second bit, 18150 Hz; the sixth bit, 18550 Hz; and the ninth bit, 18850 Hz. The encoded signal is a mixture of these frequencies, as depicted in Figure 4.

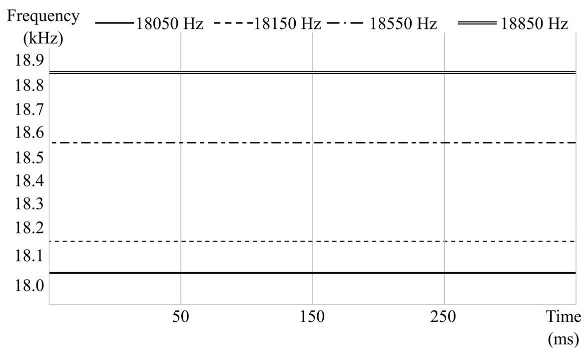

Figure 4. An example of sending ten bits of data 1100010010 using *DFP* = (100 Hz, 18–19 kHz)

This method can send many data bits at once by sending multiple frequencies simultaneously, and because it has no changing frequencies during transmission, noise due to converting frequencies rapidly, as seen with the Smart Guide system proposed by Bihler, will not occur. Also, even if the receiver fails to recognize the data, the sender is still sending the same mixture of frequencies so the receiver can try again.

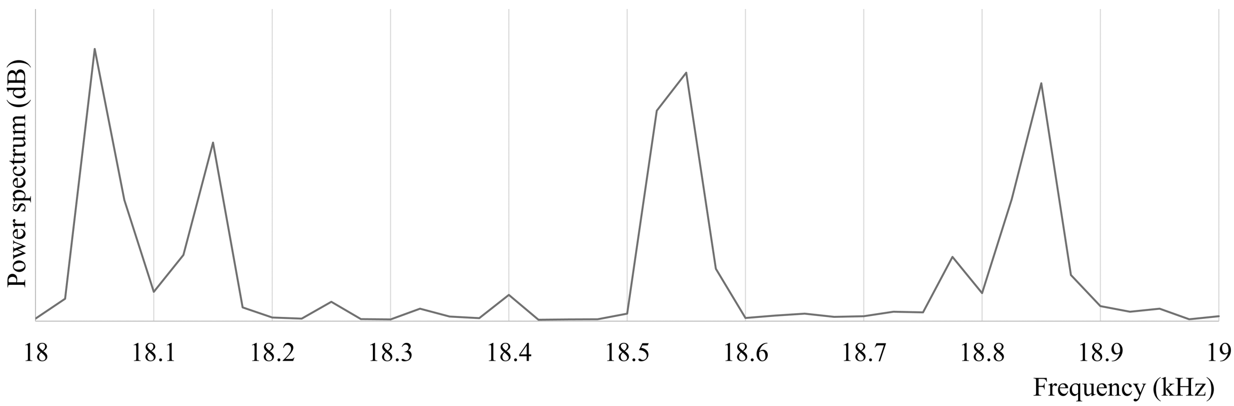

Figure 5. FFT results of 10 bits of data 1100010010, *DFP* = (100 Hz, 18–19 kHz)

3.3. Procedures for decoding data from inaudible frequency sound signals to bit data

The receiver decodes signals received from its microphone into data. The time required to receive the data once depends on how many sound samples the receiver receives. Smart devices can adjust the number of sound samples they receive in 1 s during recording; this number is c alled the *sampling rate* (*SR*) and is related to the maximum frequency that the receiver can recognize. If the receiver records 48000 samples per second, *SR* is 48000, and it can recognize sound with frequencies up to 48000/2 = 24000 Hz, according to the Nyquist-Shannon theorem [12]. Also, the proposed method requires a minimum number of sound samples necessary to extract data from signals; this required number of sound samples is called the *sample size* (*SS*). The received sound samples are discrete sound wave figures; hence, we have to extract frequencies in these samples using the FFT to decode data [16]. The accuracy and processing time of the FFT vary according to the value of SS. When the value of SS is increased, the result of the FFT become more accurate [8, 18]. Next, after finishing the FFT, the receiver extracts the bit data from the FFT results using a modified on-off keying technique [5]. If a frequency mapped with each bit is contained in the signal, the value of the corresponding bit is 1; otherwise, the value of the bit is 0. From the results of the FFT, the receiver finds valid peaks and the frequencies of those peaks, and sets the values of the corresponding bits to 1. For example, the results of the FFT would be as shown in Figure 5 when the receiver receives 10 bits of data 1100010010 using *DFP* = (100 Hz, 18–19 kHz). In Figure 5, we can determine that valid peaks are in the ranges of 18.0–18.1 kHz, 18.1–18.2 kHz, 18.5–18.6 kHz, and 18.8–18,9 kHz; hence, we set the values of the corresponding (first, second, sixth, and ninth) bits to 1. Therefore, the decoded bit value is 1100010010.

However, in Figure 5, the small peak in the area 18.7–18.8 kHz may be counted, according to some algorithms for extracting peaks, or the peak in 18.1–18.2 kHz might not be counted as valid because it has small power. In both of these cases, the received data are incorrect; hence, an error-detecting algorithm is required to check these errors. A Hamming code, which is used in the Smart Guide system, can only detect up to two bit errors, which is inferior to our method that can detect multiple bit errors [11]. For this reason, we use a cyclic redundancy check (CRC) technique to find errors [13]. The **m** bits of data that are used for the CRC are not sent separately but are attached to the tail of the real data, and they are sent together. If **m** = 3, for instance, the last 3 bits of data are for error detection, and the remaining 7 bits are real data. The 10 bits of data in Figure 5, 1100010010, are actually a combination of the last 3 bits 010 for error correction and the first 7 bits 1100010 for real data. Although the CRC bits are not used by users, this method includes them when calculating the length that it sends. That is, though the length of data that users actually use is 7 bits, this method represents that it sends 10 bits including the CRC data. The receiver discards data for which it recognizes errors through the CRC and accepts the data if there are no errors.

4. Experiments and evaluation

In this section, we explain the environment used in our experiments and the methods by which performance was measured. We also measure the performance of the existing system in the same environment and compare it with the results of ours.

4.1. Environment of experiments for measuring performance of the proposed method

We implemented the sender as a Web page playing multiple single-frequency sounds simultaneously using JavaScript and transmitted the signal through speakers in the laptop. We made the receiver an Android application recognizing sound using an SR of 48000. We used the FFT library implemented by Baoshe Zhang at the University of Lethbridge [2]. This algorithm requires a power-of-two SS based on the Cooley-Tukey FFT algorithm [4, 7]. We also used the library that implements "Simple Algorithms for Peak Detection in Time-series" proposed by Girish Keshav Palshikar to detect peaks [14, 22]. The receiver application has one button, as seen in Figure 6, and displays the recognition result, extracted data, and processing time when we push the button. The recognition result can be one of three: 1) right data is accepted, 2) wrong data is accepted, or 3) data is discarded due to a CRC error. Among these, "wrong data is accepted" means that the receiver accepts the data because it does not appear to have any errors after performing CRC error detection, but actually it has bit errors; hence, the received data is different from what the sender actually sends.

In a real communication situation, the receiver will not be able to determine whether the data is wrong if there are no CRC errors; however, it is determined in advance which data the sender was going to send. Hence, it distinguishes between cases 2) and 3) by checking whether the received data is the same as the data sent by the sender. Figure 6 shows the results of a test that the received data was coincident with what the sender sent, and the processing time was 410 ms.

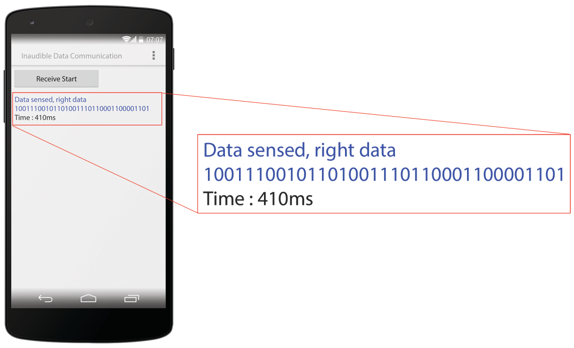

Figure 6. Screenshot of the receiver Android application

Next, we measured the performance of the proposed method by two factors similar to throughput and error rate [24]. These two factors are transmission rate (*TR*) and error-detection rate (*EDR*), and they are calculated as shown in Eqs. (2) and (3).

$$TR = L_d * R_{CD} * \frac{1}{T}\ (bits/s) \tag{2}$$

$$EDR = \frac{R_{DD}}{R_{WD} + R_{DD}}\ (\%) \tag{3}$$

TR is similar to throughput and indicates how many correct bits can be sent in a unit of time. L_d is the length of the data in bits, R_{CD} is the rate of correct data, R_{WD} is the rate of wrong data, and R_{DD} is the rate of discarded data. *T* means the average processing time, and the unit time for these experiments was 1 s. The *EDR* represents how well the error-detection algorithm detects errors and is calculated by the proportion of discarded data (R_{DD}) to errant data (R_{WD}+R_{DD}). We used five randomly generated sets of 40 bits of data, as shown in Table 1. Each 40-bit data set consists of 32 bits of actual data and an 8-bit CRC that is attached to its tail. We calculated the CRC for each 32 bits of data by using the generator polynomial that is used in WCDMA wireless communications: 0x9B ($x^8 + x^7 + x^4 + x^3 + x + 1$) [1, 10, 17].

We performed experiments by placing the smartphone running the receiver application at the point in which sound from the sender was recognized as 75 dB with 50 dB noise, which is a similar level to that of a general home [19]. We used a MacBook Pro to run the sender Web page and a Google Nexus 5 with Android version 4.4 to run the receiver application.

Table 2. Five randomly generated sets of 40 bits of data that are used in experiments

	32 bits of data	8-bit CRC	Hexadecimal
1st	10011100 10110100 01100111 01100011	00001101	0x9CB467630D
2nd	11000100 00100010 00110100 10101010	11111010	0xC42234AAFA
3rd	10100110 11101101 11010010 10010001	10100010	0xA6EDD291A2
4th	00010001 00110001 10010110 01011111	11011001	0x1131965FD9
5th	11001000 11101111 11001101 11101011	01101011	0xC8EFCDEB6B

Table 1. Data-transmission performance depending on each value of *SS* with *DFP* = (50 Hz, 18–20 kHz)

DFP = (50 Hz, 18–20 kHz)	R_{CD} (%)	R_{WD} (%)	R_{DD} (%)	*T* (ms)	*TR* (bits/s)	*EDR* (%)
SS = 2048	0.04	0.16	99.80	106.03	0.15	99.84
SS = 4096	76.84	0.00	23.16	94.25	326.22	100.00
SS = 8192	99.40	0.00	0.60	224.13	177.42	100.00
SS = 16384	99.68	0.00	0.32	412.32	96.70	100.00
SS = 32768	99.88	0.00	0.12	773.01	51.68	100.00

Table 3. Data-transmission performance depending on each value of *DFP* with value of *SS*=32768

SS = 32768	R_{CD} (%)	R_{WD} (%)	R_{DD} (%)	*T* (ms)	*TR* (bits/s)	*EDR* (%)
DFP = (25 Hz, 18–19 kHz)	99.68	0.00	0.32	772.31	51.63	100.00
DFP = (50 Hz, 18–20 kHz)	99.88	0.00	0.12	773.01	51.68	100.00
DFP = (75 Hz, 18–21 kHz)	99.88	0.00	0.12	773.26	51.67	100.00
DFP = (100 Hz, 18–22 kHz)	99.28	0.00	0.72	773.17	51.36	100.00

4.2. Performance test of the proposed method

The performance of the proposed method can be measured differently depending on the value of the *SS* and *DFP*. Therefore, we measured the ability to transmit data in various settings.

- *DFP* = (25 Hz, 18–19 kHz), (50 Hz, 18–20 kHz), (75 Hz, 18–21 kHz), (100 Hz, 18–22 kHz)
- *SS* = 2048, 4096, 8192, 16384, 32768

We checked the transition of the performances depending on varying the value of *DFP* with the value of *SS* = 32768. We measured 2500 times at each value of *DFP*, and the results are shown in Table 2.

The rate of receiving correct data was over 99% for all cases, and the value of *TR* was the highest when *DFP* = (50 Hz, 18–20 kHz).

Next, we observed the changes in performance depending on the value of *SS* with *DFP* = (50 Hz, 18–20 kHz), which performed the best. In the same environment, we tested 2500 times at each value of *SS*, and the results are shown in Table 3.

When *SS* = 2048, data transmission almost failed because *SS* was too small, and the receiver could not decode the data from the results of the FFT. When *SS* = 4096, although R_{CD} was 76.8%, which is much lower than 99% in $SS \geq 8192$, the value of *TR* was the highest due to a reduced recognition time *T*. In environments that need more precise transmission

Table 5. Data-transmission performance of the Smart Guide system

R_{CD}	R_{WD}	R_{DD}	T	TR	EDR
28.32 %	29.32 %	42.36 %	208.07 ms	10.89 bits/s	59.10 %

Table 4. Random 8-bit data sets that are used to measure the performance of the Smart Guide system

	8 bits with Hamming code	4 bits without Hamming code	Hexadecimal
1st	10110100	1010	0xB4
2nd	10011001	0100	0x99
3rd	00101101	1110	0x2D
4th	10101010	1101	0xAA
5th	11010010	0001	0xD2

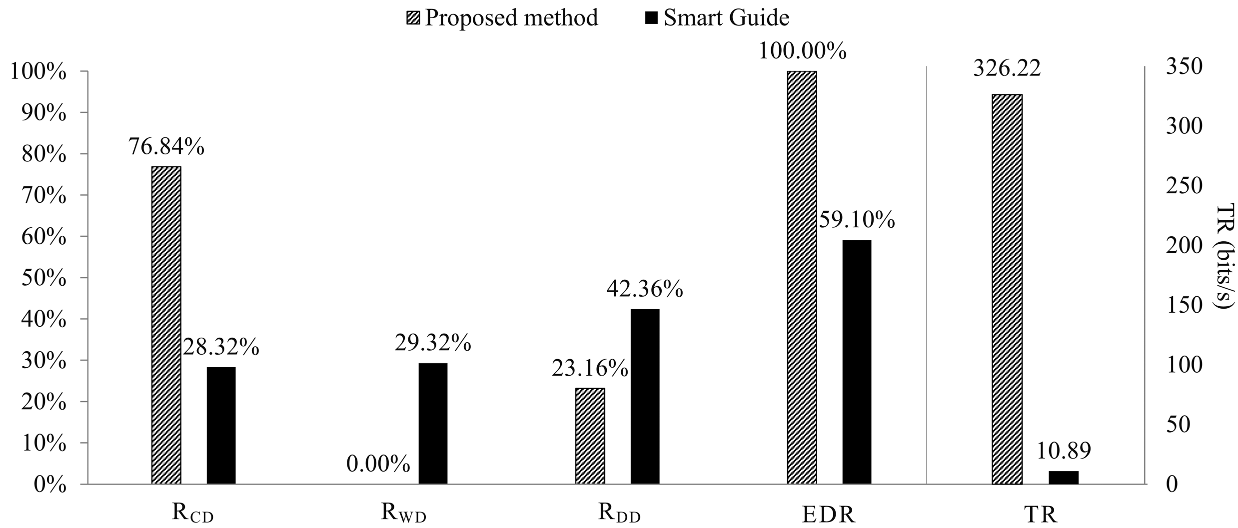

Figure 7. Performance comparison between the proposed method and the Smart Guide system

rather than a high *TR*, increasing the value of *SS* will produce the desired results. We use the result with *SS* = 4096, which showed the highest value of *TR*, to compare our performance with that of the Smart Guide system proposed by Bihler.

4.3. Performance test of the Smart Guide system and performance comparison

We also implemented the Smart Guide system proposed by Bihler in the same environment. Because the Smart Guide system has different techniques for sending data, we made a new sender application and a new receiver application. Although Bihler had measured the performance at a 2-m distance, we measured the success rate by placing the smartphone running the receiver application at the point in which sound sent by the sender is recognized as 75 dB instead of at a 2-m distance, because the level of sound can be measured differently based on the volume of the speaker of the sender. Eight bits of data, including [8, 4] Hamming code were used in the experiment, and we created five randomly generated data sets as in the previous experiments. These are shown in Table 4.

The 4-bit Hamming code is represented by the first, second, fourth, and eighth bits. When the receiver receives 8 bits of data, it corrects a wrong bit using the Hamming code and checks the eighth bit, called a parity bit. It discards the data when the parity bit is wrong. We measured the rate of correct data and the rate of wrong data by comparing the parity bit checked result with what the sender sent. As in the previous experiments, we tested 2500 times. The results are shown in Table 5.

During our testing, the rate of correct data R_{CD} was 28.3%. The R_{WD} was 29.3%, which is so high because of the limitations of Hamming code error detection and correction. The comparison between the best performance results of the proposed method and that of the Smart Guide system is shown in the Figure 7 graph.

The proposed method showed an R_{CD} that was three times higher than that of the Smart Guide system. The Smart Guide system had a 29.3% R_{WD}, proving its data to be less reliable than that of the proposed method.

Considering that currently used data-transmission technologies do not allow for the packet error rate (PER) to be more than 1%, the R_{CD} value of the Smart Guide system was too high [23]. R_{DD}, meaning the discarded data by error-detection algorithms, was higher with the Smart Guide system than with the proposed method. However, this does not mean that the error-detection algorithm that we used has bad performance, but rather, that errors do not occur as often when using the proposed method because it has better data-transmission performance, considering its R_{CD} and *EDR*. The proposed method showed an *EDR* of 100%, detecting every error, which is better than the Smart Guide system's failure to detect 40% of errors. Nevertheless, we determined that the proposed method can send 30 times more data than the Smart Guide system in the same unit of time.

5. Conclusion

In this paper, a new method that can send more data using inaudible high frequencies for short-distance data communications than the existing systems is proposed. As you can see in section 4, a strength of the newly proposed method is that it can be applied in various environments by adjusting variables to either increase the data-transmission rate or enhance recognition accuracy. The proposed method showed a higher *TR* by sending 40 bits of data in 92 ms while detecting every error. With this method, wireless communications using inaudible high frequencies can be used in more diverse areas.

In future research, we will investigate the method for increasing reliability and efficiency by reusing data that was discarded due to CRC errors. We will also evolve the method to be able to send data with a length of more than 40 bits so that wireless communications using inaudible high frequencies can be used in much more diverse fields by dividing data into several 40-bit data packets before sending them.

Acknowledges

This research was supported in part by Ministry of Education, under Basic Science Research Program (NRF-2013R1A1A2061478), respectively.

References

[1] Baicheva T, Dodunekov S, Kazakov P (1998) On the cyclic redundancy-check codes with 8-bit redundancy. Computer Communications, 21(11), 1030-1033. doi: 10.1016/S0140-3664(98)00165-0

[2] Baoshe Z (2005) Java FFTPack. http://jfftpack.sourceforge.net. Accessed 6 April 2015

[3] Bihler P, Imhoff P, Cremers AB (2011) SmartGuide - A Smartphone museum guide with ultrasound control. Procedia Computer Science 5:586-592. doi: 10.1016/j.procs.2011.07.076

[4] Blahut RE (2010) The Cooley-Tukey fast Fourier transform. In: Proc. Fast Algorithms for Signal. Cambridge, UK, pp. 68-71

[5] Chitode JS (2008) Digital amplitude modulation or amplitude shift keying. Analog and Digital Communication India: Technical Publication

[6] Chung MB, Choo HS (2014) Near wireless-control technology between smart devices using inaudible high-frequencies. Multimedia Tools and Applications. pp. 1-17. doi: 10.1007/s11042-014-1901-x

[7] Cooley JW, Tukey JW (1965) An algorithm for the machine calculation of complex Fourier series. Mathematics of computation, 19(90), 297-301.

[8] Dorrer C, Belabas N, Likforman JP, Joffre M (2000) Spectral resolution and sampling issues in Fourier-transform spectral interferometry. Journal of the Optical Society of America B, 17(10), 1795-1802. doi: 10.1364/JOSAB.17.001795

[9] Kim JB, Song JE, Lee MK (2012) Authentication of a smart phone user using audio frequency analysis. The Journal of the Korea Institute of Information Security and Cryptology 22: 327-336

[10] Koopman P, Chakravarty T (2004) Cyclic redundancy code (CRC) polynomial selection for embedded networks. In Dependable Systems and Networks, 2004 International Conference on (pp. 145-154). IEEE. doi: 10.1109/DSN.2004.1311885

[11] Kumar UK, Umashankar BS (2007) Improved hamming code for error detection and correction. In: Proc. 2nd International Symposium on Wireless Pervasive Computing (ISWPC'07). San Juan, Puerto Rico. doi: 10.1109/ISWPC.2007.342654

[12] Marks II, Robert J (1991) Introduction to shannon sampling and interpolation theory. Springer-Verlag New York
[13] Moreira JC, Farrell PG (2006) Cyclic codes. Essentials of Error-Control Coding. West Sussex, England: John Wiley & Sons. pp. 81-96
[14] Palshikar G (2009) Simple algorithms for peak detection in time-series. In: Proc. 1st Int. Conf. Advanced Data Analysis, Business Analytics and Intelligence
[15] Pekka J, Chia-Hao Y, Klaus D, Cassio R, Carl W, Klaus H, Olav T, Visa K (2009) Device-to-device communication underlaying cellular communications systems. Int. Journal of Communications, Network and System Sciences, 2009. doi: 10.4236/ijcns.2009.23019
[16] Rao KR, Kim DN, Hwang JJ (2011) Fast Fourier transform-algorithms and applications: algorithms and applications. Springer Science & Business Media. pp. 1-2
[17] Richardson A (2005) Error detection coding and CRCs. In: Proc. WCDMA Design Handbook. Cambridge, England, pp. 211-213
[18] Schatzman, JC (1996) Accuracy of the discrete Fourier transform and the fast Fourier transform. SIAM Journal on Scientific Computing, 17(5), 1150-1166. doi: 10.1137/S1064827593247023
[19] Singh N, Davar SC (2004) Noise pollution-sources, effects and control. The Journal of Human Ecology, 16(3), 181-187.
[20] Stelmachowicz PG, Beauchaine KA, Kalberer A, Jesteadt W (1989) Normative thresholds in the 8-to 20-kHz range as a function of age. The Journal of the Acoustical Society of America, 86(4), 1384-1391. doi: 10.1121/1.398698
[21] Takeda S, Morioka I, Miyashita K, Okumura A, Yoshida Y, Matsumoto K (1992) Age variation in the upper limit of hearing. European journal of applied physiology and occupational physiology, 65(5), 403-408. doi: 10.1007/BF00243505
[22] Tinevez JY (2011) A standard peak detector in time series. https://code.google.com/p/fiji-bi/source/browse/src-plugins/FlowMate_/fiji/plugin/flowmate/analysis/PeakDetector.java?name=flow-mate&r=0ec4620b8e4aaebd183c2b57c89390595f574564. Accessed 6 April 2015
[23] Tiso J (2011) Designing cisco network service architecture (ARCH). In: Proc. 3rd ed., Indianapolis, Cisco Press
[24] Yoon DG, Shin SY, Park JH, Park HS, Kwon WH (2007) Performance analysis of IEEE 802.11 b under multiple IEEE 802.15. 4 interferences. In: Proc. Wired/Wireless Internet Communications. Springer Berlin Heidelberg, pp. 213-222. doi: 10.1007/978-3-540-72697-5_18