Journal of Information Hiding and Multimedia Signal Processing ©2010 ISSN 2073-4212
Ubiquitous International Volume **1**, Number **2**, April **2010**

Capacity Adaptive Synchronized Acoustic Steganography Scheme

Xuping Huang

School of Multidisciplinary Sciences
The Graduate University for Advanced Studies (SOKENDAI)
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, Japan 183-8512
huang-xp@nii.ac.jp

Yoshihiko Abe

Software Information Science Faculty
Iwate Prefectural University
152-52 Sugo, Takizawa, Iwate, Japan 020-0193
yoshi@iwate-pu.ac.jp

Isao Echizen
National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, Japan 183-8512
iechizen@nii.ac.jp

Received April 2009; revised January 2010

Abstract. *A new steganography scheme with capacity variability and synchronization is proposed for secure transmission of acoustic data . In this scheme, the sender records acoustic data, converts it into PCM data, embeds synchronous fix-sized divided data , interleaves and transmits the secret data to generate stego data, and the receiver extracts the secret data all in real-time. Synchronization ensures scalability of privacy protection and secures live broadcasts. The embedding capacity is adaptive to the features of the human auditory system (HAS). Depending on the imperceptibly of the scrambling bits of cover data, the embedding positions can be arbitrarily assigned according to the PCM settings and embedding algorithm. A socket model is used to transmit the stego data. Objective (signal to noise ratio) measurements and subjective evaluations demonstrate the effectiveness of the scheme.*
Keywords: Synchronized Steganography, Acoustic Data Processing, Arbitrarily Assigned Significant Bits

1. **Introduction.** Steganography means the writing of messages in such a way that only the sender and intended recipient are aware of the existence of the message. By imperceptibly scrambling bits of cover data and using a secret key, steganography embeds secret information in comprehensible materials in such a way that does not draw the suspicion of interceptors or eavesdroppers. Unlike watermarking, steganography cannot endure unintentional modifications or intentional attacks. Secret data, however, should be resistant to distortion and tampering. Since the secret data is hidden within the physical arrangement of the cover data, bit scrambling distorts the cover data. Steganography based on acoustic data takes advantage of human auditory system (HAS) characteristics whereby distortion from the embedding process cannot be distinguished by the human ear. The algorithm described in this paper results in fewer artifacts of distortion and frees up more

bit positions for embedding in comparison with previous approaches. In the case of acoustic data with a specific quantization in a certain time[1], up to the 8^{th} bit can be used for embedding without degrading the cover signal.

2. **Problems with Conventional Methods.** Empirical studies have been undertaken to embed metadata such as text and images in other media. The methods include least significant bit (LSB) embedding, discrete cosine transform (DCT) encoding, MP3Stego, spread spectrum, echo data hiding, etc. Several embedding approaches have been proposed [1, 2, 3, 4, 5, 6, 7, 8]. Most of these studies focus on image, text, and audio, though only as the cover string, and their schemes have limitations as to (1) the type of metadata media, (2) synchronization requirements, and (3) data hiding capacity.

The principal challenge is to use steganography when acoustic media is the target object. Acoustic data, especially speech data, is widely used. Privacy protection for mobile phones is now an important issue, and the current methods use cryptography. Although cryptography prevents eavesdropping by rendering the message incomprehensible, steganography is a more sophisticated method that camouflages secret data. That is, the secret data and the cover data may be speech data created by the same speaker. This makes it difficult for attackers to distinguish the secret message from the cover data.

Furthermore, acoustic media can conceal subliminal communication channels in, for example, live broadcasting, mobile communications, and teleconferences. However, audio (secret)-to-audio (cover) steganography has seldom been implemented for this purpose. Real-time signal processing is required in emergencies. Furthermore, data security could be enhanced by recording the secret data in a synchronized process. Many methods use LSB embedding, echo hiding, spectrogram, etc. Regarding LSB, although a psychoacoustic model can perceptually weight the noise and replaces audio in such a way that cannot be detected by HAS, the embedding capacity cannot meet the requirements for data transmission. Echo hiding exploits the offset or delay between cover data and stego data. The secret information is added to the initial amplitude [9, 10]. Added redundancy and limited frequency range for embedding are limitations of echo hiding. The previous studies can be divided into two main approaches: fragile and robust information hiding. The fragile approach requires a balance between capacity and data quality to take advantage of human's perception, while the robust information hiding requires algorithmic enhancements to prevent it from being attacked. Some features of the related work and the problem statement are compared as follows:

	Real-time	Audio-to-Audio	Robustness
Neil[1]	N/A	N/A	YES
Kusatsu[16]	N/A	N/A	YES
Hosei[3]	YES	N/A	YES
Petitcolas[12]	N/A	YES	YES

In Neil and Kusatsu's work, robust information hiding was implemented by using images in Hosei and Petitcolas's work, and WAV and mp3 data was used as cover data. As mentioned above, the previous methods do not meet the needs of real-time telecommunications to thwart eavesdroppers. Real-time processing of tele-communications and live broadcasts is required to ensure data integrity and security. However, if a complex algorithm is used to keep the system robust, for example, one using random PN as the

[1]For example, data sampled for 30 seconds with the following settings: (1) Cover: 32 kHz, 16 bit, 2 channels; (2) Embed: 16 kHz, 8 bit, 1 channel

key, then the time required for embedding and extraction makes real-time communication impossible. Therefore, a balance has to be struck between robustness and processing time.

2.1. **Contribution.** This paper proposes a new model and algorithm for a real-time steganography system. The system records a secret audio data stream synchronously in a WAV acoustic data stream. The embedding positions in the cover data with 16-bit sampling can be arbitrarily assigned. The main processes include PCM setting, embedding, and robustness analysis. The algorithm does a masking calculation to cover the bit stream at each sampling point. The capacity of the first signal at every sampling point of the cover data is calculated to evaluate the exact amplitude of the acoustic data. Our contributions are as follows:

The synchronized steganography of previous studies used LSB scrambling approach, which makes it possible to assemble and utilize speech information securely to thwart intentional attacks [11, 12].

LSB is susceptible to bit scrambling in that the attacker can remove or disable hidden data simply by setting all the LSBs to be 0 or 1. Additionally, if only LSB is used, the amount of data will be small. Our approach has the following attributes:

(1) Embed secret audio in cover audio

Analog acoustic signals interfere with each other when two acoustic streams are played at the same time [13, 14]. Furthermore, the chance of discrimination depends on environmental factors including the characteristics of the cover audio. The data sampling and interpolation algorithm makes it possible to embed speech data into white noise approaching the level of silence.

(2) Significant Bit Embedding

Even though multiple significant bit embedding affects the quality of stego data, the data shift and bit masking we propose reduce the degradation to indiscernible levels. The embedding bit positions can be specified by the sender at every sampling point of the cover data. Adaptive data PCM setting promises a variable embedding rate and a large data capacity.

(3) Synchronized Process

Stego data can be generated even if the secret message is recorded when the cover data is being broadcasted. The HAS has trouble detecting such hidden data, and this makes it possible to avoid intentional attacks.

2.2. **Example Application.** Figure 1 shows, an example application of the synchronized acoustic steganography scheme: secure real-time communication between two or more parties. A socket function is used to transmit the stego data. Attackers can receive only the stego data stream by eavesdropping, whereas the sender and trusted receiver can exchange private speech information freely. To start a transmission, Alice (the intended receiver) starts the socket client to apply for a connection. After listening, Bob (the supposed sender) selects a piece of acoustic data as a cover and starts the socket server. When the socket starts, Bob records the secret message via microphone, and the speech is synchronously embedded in the cover audio data stream and carried in the stego data that is transmitted to Alice. Bob and Alice share the extraction program including the embedding key in advanced. The secret-speech bit stream is extracted in real-time as the stego stream is being received by using the embedding key.

3. **Synchronized Acoustic Steganography using WAV data.** This paper proposes a synchronized steganography system for acoustic data. The synchronous aspect means that the secret data is recorded and steganographically embedded at the same time as the cover is made, and the stego data is subsequently sent or broadcast to multiple receivers.

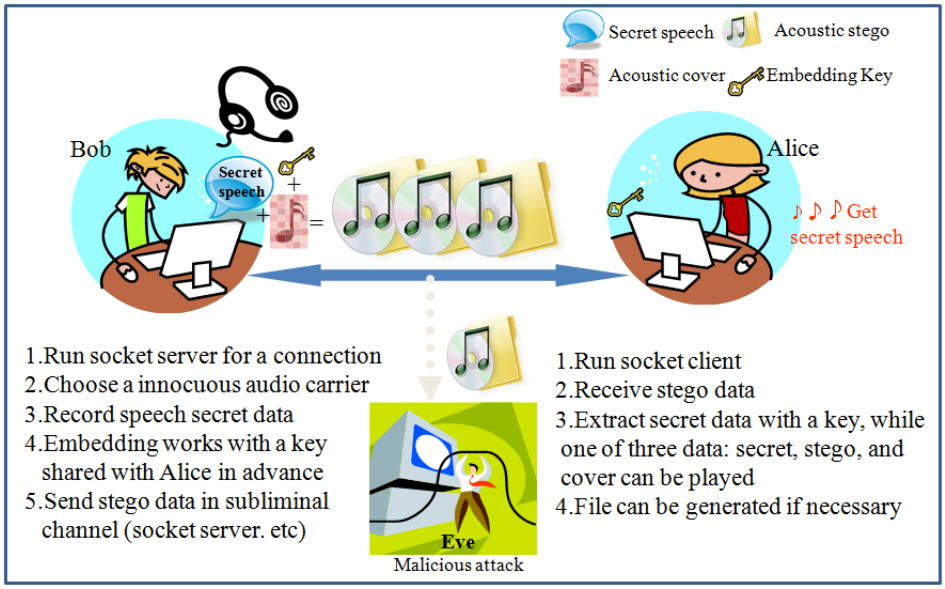


FIGURE 1. Illustration of Synchronized Acoustic Steganography Scheme

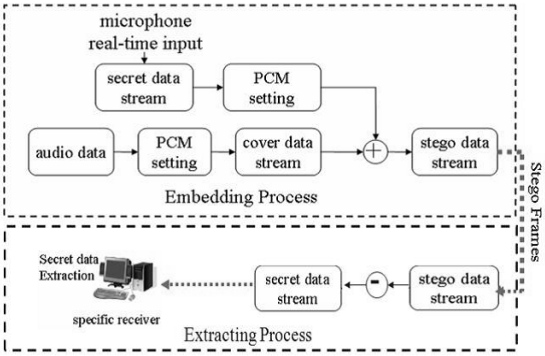


FIGURE 2. Synchronized steganography scheme

However, only a trusted receiver can extract the secret data by using a secret key shared with the sender. The acoustic data is transmitted through socket communication.

3.1. **Statement of Purpose.** Although many methods have been proposed, none of them can be used as a real-time scheme during emergencies when electronic monitors, and the Internet can not be used. However, acoustic data can be carried through the air via radio in emergencies. Even though multimedia data can be put in multi-media channels of broadcasting, it is necessary to make sure that various channels are available in emergencies. Steganography provides a way to embed many kinds of data in the same carrier and not have them interfere with each other; it does not weaken the quality of the carrier data. The algorithm presented in this work permits a perceptual degradation only if the change to the data causes no noticeable difference in perceptual tolerance. Steganography can be used to protect valuable information from possible sabotage or unauthorized viewing, and it can use multiple media in emergencies. Figure 2 shows the steganography scheme.

The purpose of this research is to secure private digital multi-media information sent through communications networks. The fidelity and integrity of the secret data must be

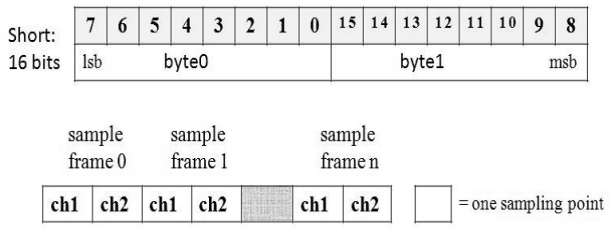


FIGURE 3. Endian order of stereo data

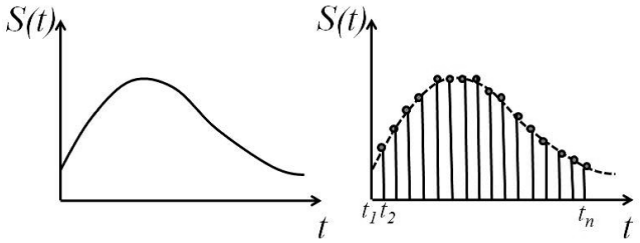


FIGURE 4. Analog Signal

assessed against the human auditory system. Live broadcasting and mobile communications are possible applications. There are three keywords phrases. (1) *Steganography*: To hide and transmit acoustic data through apparently innocuous acoustic carriers to conceal the existence of secret audio data; (2) *Real-time*: The secret audio data is recorded when the embedding scheme starts to run and is dynamically synchronized with cover sequences;

(3) *Arbitrarily assigned multiple significant bits embedding*: A sufficient number of embeddable coefficients are present in each shuffled frame, though shuffling in significant bits may increase the sensitivity to intentional attacks aimed at rendering the embedded signals comprehensible. The embedding capacity and positions can be arbitrarily assigned up to the 8^{th} significant bit from the least significant bit (LSB) at each sampling point of 16-bit cover data.

3.2. **Acoustic Data Function.**

3.2.1. *Data Function.* Different from image and text data, the acoustical data format has a special function and header file. Sampling frequency, sampling size, and number of channels affect audio quality, which in turn affects embedding quality [15, 16, 17]. Data is stored in 8-bit bytes, arranged in Intel 80x86 format. This format suits the peculiarities of the Intel CPU, such as little-endian byte order, as shown in Figure 3. Figures 4 and 5 illustrate how to convert an analog signal into a digital signal. As Figure 3 shows, multiple bytes are stored with the low-order (i.e., least significant bytes first.) As Figure 4 shows, sampling is done at equal intervals t_1, t_2, t_n. *etc.* Sampling points with different amplitudes are represented with the same bit depth.

3.2.2. *WAV Format Structure.* A WAVE file is a collection of different chunks. The Format 'fmt' chunk contains important parameters of a waveform, such as sampling rate, and the data chunk contains the actual waveform data. These chunks are required, while other optional chunks included in the file structure define cue points, list instrument parameters, store application-specific information. Figure 6 is a graphical overview of a minimal WAVE file consisting of a single WAVE with the two required chunks.

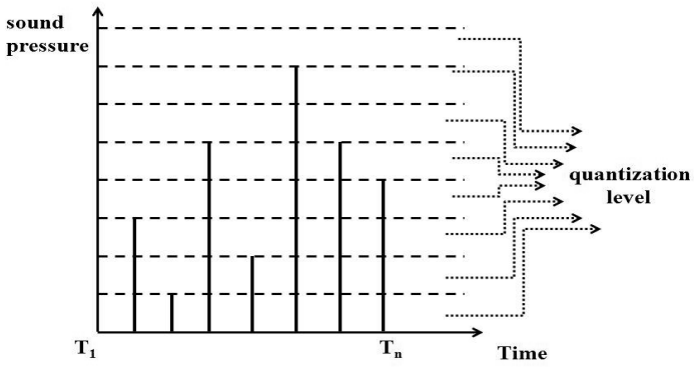


FIGURE 5. Digital signal

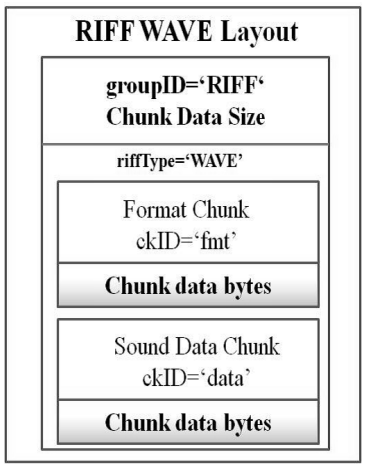


FIGURE 6. Basic chunk file structure

3.2.3. *WAV Header Information.* After the secret data frames have been extracted from the stego stream, the WAV header information has to be appended to the frames to ensure the integrity of the secret data. Unlike image data, acoustic sequences require the correct time stamp. When the receiver gets continuous frames, the head file has to be appended to each frame to enable the embedded audio to be played. Table 1 lists the WAV data information that has to be appended to the extracted frames.

3.3. **Principle of Proposed Synchronized Steganography.** There are four steps in this method:
(1) The secret data is recorded and embedded in real-time.
(2) PCM information of the cover data and the embedded data are set to specified values. The playback lengths of the cover data and embedded data are equal.
(3) The coded data (wave format) is divided into fix-sized frames suitable for streaming and reconstructing signals from two different signal sequences.
(4) To playback the embedded data after it has been extracted from the stego data, it is necessary to append wave head information corresponding to its PCM information.

The system converts the analog input signal into digital data directly via microphone and embeds bit signals of secret data in the cover audio data. The real-time processing uses a narrow frequency band covering the span of frequency of the audio file. The real-time process is divided into steps during which the secret audio data is synchronously embedded in the cover audio data. Real-time pertains to the whole process of producing

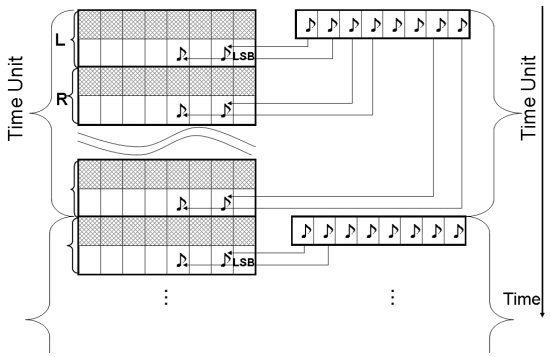


FIGURE 7. Multi-bit embedding process

the secret data, embedding it in the cover, thereby creating the stego data, and extracting the secret data from the stego data.

Point 1 It is possible that a third party will get the transmitted information and the stego-key to render the stego-data vulnerable. To combat against this, the embedded data could be encrypted to make a hybrid encryption system. After a intneded receiver extracts the embedded data with the stego key, the embedding data can be generated as a wave format file.

Point 2 The cover data is stored on a hard disk. Many different experiments using different types of music as the cover have shown that this method works effectively. For instance, we recorded natural background noise in an unoccupied lab at midnight and used it as quiet cover to embed a loud piece of audio data.

Point 3 The cover data and secret data are as follows:

(1) The cover is 32-kHz,16-bit, 2-channel audio data.

(2) The secret audio is 16-kHz, 8-bit, 1-channel data.

(3) The cover data and secret data are synchronized; thus, a multiplicative factor determines the number of bits to be embedded in the sampling point.

Point 4 The Stego data is sent through the internet or any broadcasting system. The security and integrity of the secret data can be guaranteed even if the stego data is intercepted.

Point 5 Audio data are coded into a digital signal and then decoded. For this, it is necessary to append wav header information onto the extracted frames.

Both the cover and the embedded data are divided into fix-sized frames, and bit scrambling is done on each frame of the cover.

3.4. **Algorithms.**

3.4.1. *Embedding Process.* In Fig. 7, the sequence s(n) stands for stego data, which is modeled as a sample drawn from a sampling terminal. c(n) means there are n sampling points of cover data, and each point equals a 16-bit data stream. Two bits of secret data are embedded in the cover sampling point. The process of recording the secret data stream is synchronized with the embedding process. The data stream on the left side (presented in 16 bits) is the cover data c(n), and embedded data e(n) is on the right side (presented in 8 bits). It is possible to embed two bits of secret data in two arbitrary bit positions $[1^{st}, 8^{th}]$ from the least significant bit (LSB). Once the positions have been assigned, all the secret data stream will be embedded in the assigned bit positions.

There are three steps involved in embedding two bits of secret data.

Input: cover stream and secret stream;

Output: stego stream;
StepE1: Clear Data Bit in Cover:
The first step is to set the data at the embedding location to be "0". A cmask is used to make the designated location available for secret data. The following function accomplishes this task.

$$\begin{aligned} cmask1 = (2^{loc1-1}) &\oplus (0\mathbf{x}FF) \\ cmask2 = (2^{loc2-1}) &\oplus (0\mathbf{x}FF) \\ cmask = cmask1 &\wedge cmask2 \end{aligned}$$

StepE2: Copy Secret Data:
The second step is to put two bits of secret data into the bit location assigned by the user. Two consecutive bits of secret data are read from and written into the designated bit location; the other bits in the 8-bit data unit are cleared to be "0". The following function accomplishes this task.

$$\begin{aligned} &\mathbf{for}(i = 0, 0 \leq i \leq 4, i++)\mathbf{do} \\ &e1 = embed >> (7 - 2i) \\ &e1 = e1\&(0\mathbf{x}01) \\ &e1 = e1 << (loc1 - 1) \\ &e2 = embed >> (6 - 2i) \\ &e2 = e2\&(0\mathbf{x}01) \\ &e2 = e2 << (loc2 - 1) \end{aligned}$$

i is the ordered sequence of 2-bit secret data units counting from the left of the insignificant 8-bit unit of cover data at the sampling point.
StepE3: Execute Embedding:
The third step generates the stego data as follows.

$$c = cover\&cmaskstego = c|e1|e2$$

The following example concretely illustrates this process. The designated positions are *loc1=4, loc2=2, and i = 1* .

Step 1 First, a *cmask* is generated. The 2-bits unit [2] of the cover has 8 insignificant bits in which *loc1* and *loc2* are available for secret data.

8^{th}	7^{th}	6^{th}	5^{th}	4^{th}	3^{rd}	2^{nd}	lsb
1	1	1	1	0	1	0	1

Step 2 Second, two bits of secret data are read and put into the designated bit positions by shifting and taking bitwise AND. Take the following data as an example: When *i=1*, it means that the 6^{th} *e1* and 5^{th} *e2* bit data will be read and put into the 2^{nd} and 4^{th} bit positions of the cover.

The consecutive bits of secret data are:

1	0	1	0	0	1	0	1

Accordingly, we embed the 6^{th} *(e1=1)* in the 4^{th} LSB of the cover and the 5^{th} *(e2=0)* into the 2^{nd} LSB. The same process is used to put *e2* in bit position *loc2*.

Step 3 The next step is to embed *e1* and *e2* in the cover data prepared by *cmask*.

For example, a unit of cover data including 8 consecutive insignificant bits is:

1	0	1	1	0	0	1	1

[2] The first 2-bits unit of the loop. i=1 means $(7^{st}, 8^{th})$ bit positions from LSB

Data *e1* takes the place of the 4^{th} bit position with the original data "0" in it, and similarly, *e2* takes the place of the 2^{nd} bit position with the original data "1" in it. Thus, the stego data are:

1	0	1	1	1	0	0	1

3.5. **Extraction and key.** Information hidden in the cover data cannot be extracted if only the detection is successful and key information is known [18, 19, 20]. To deal with the possibility that the attacker knows of the existence of the secret speech and somehow learns the key, say, by detecting data packages on the Internet, we can use socket transmission to improve the security of the secret data. Even though the attackers may get all of the data frames, the secret speech is not available unless specific wav header information is appended before the secret data frames. The header information includes the PCM set of the secret data and cover data, and the setting is performed synchronously on the cover data and speech bit stream while they are being read. The PCM settings of the cover data and stego data are the same. When a trusted receiver uses the key and listens to the socket server, the header information can simply be fetched by copying the parameters from the server. When data transmission starts, the extraction program automatically executes the header appending program on the socket client terminal.

The extraction is done as follows:

Input: stego stream;

Output: embedded stream;

StepD1: Preparation for extraction:

$$mask_1 = 0X01 \ll (loc1 - 1);$$
$$mask_2 = 0X01 \ll (loc2 - 1);$$

StepD2: Extraction signal from loc1 and loc2; i is the sequence number of 2-bit unit.

$$e1 = \sum_{i=0}^{8} (stego(2i) \& mask_1) \gg (loc1 - 1)$$
$$e2 = \sum_{i=0}^{8} (stego(2i) \& mask_2) \gg (loc2 - 1)$$

StepD3: Create embed signal; p is the index of extracted steam.

$$embed[j]| = (e1 \ll 6 - 2p)$$
$$embed[j]| = (e2 \ll 5 - 2p)$$

StepD4: Append wav header upon extracted stream; The extracted stream will have errors and the content will not be recognized unless the wav header is appended. The information is edited according to the PCM settings. The header information process is included in the extraction process (see Table 1).

Extraction Example:

Data in buffer shown as follows plots stego stream on the x-axis and time-scale on the y-axis. Each sampling point is represented by 16 bits. The least significant bit of each sampling point are 1,0,1,1 and secret data is only included in the 8-bit unit counting from LSB on the right side.

1	1	1	0	1	1	1	0	1	0	1	0	0	0	1	1
0	1	0	1	1	1	0	0	0	1	0	0	1	1	1	0
0	1	1	1	0	1	0	1	1	0	0	1	1	0	0	1
1	1	0	1	1	0	1	1	1	0	0	0	0	1	1	1

TABLE 1. Wav data header information

Size(bytes)	Value	Description
4	'R' 'I' 'F' 'F'(0x52494646)	chunk ID
4	(file size)-8	chunk size
4	'W' 'A' 'V' 'E' (0x57415645)	RIFF type
4	'f' 'm' 't' ' ' (0x666D7420)	format space
4	16 (10 00 00 00) linear PCM	byte of fmt chunk
2	1 (01 00) linear PCM	assumed format ID
2	mono 1 (01 00) stereo 2(02 00)	number of channels
4	32000Hz (32 AC 00 00)	sampling rate
4	128000(00 F4 01 00) (32000*2*2=128000)	data transfer speed 32 KHz 16-bit stereo
2	1*1=1 (01 00) 8-bit mono 2*2=4 (04 00) 16-bit stereo	block alignment
2	8 (08 00 00 00) 16 (10 00 00 00)	significant bits per sample
2	null (if linear PCM)	size of extra format bytes
n	0 -65535 (null if linear PCM)	extra format bytes
4	'd' 'a' 't' 'a'	data chunk ID
4	n	chunk size, type: dword depending on sample length and compression
n	audio data length	wave format chunk values

Then the extracted bit stream is:

0	1	1	1	1	0	0	1

4. **Evaluation.** The cover data are files with specific PCM settings and the embedding process starts when secret acoustic data are recorded by microphone.

4.1. **Investigation of Data Fundamental Specification.** The steganographic capacity, embedding capacity, embedding efficiency, and data payload affect the embedding quality. To specify the properties of the acoustic signals in our experiment, we used the first "1" signal from the significant bit (16^{th} bit) at each sampling point. We calculated the number of sampling points that had the same bit position as the first "1" signal to determine the physical volume of the signal in an objective way. Furthermore, there are tradeoffs between imperceptibility, differentiation between cover and stego, and data classification. The data classification was thus based on volume, bit-rate, and variety of audio files. Audio magnitude was measured by counting sampling points where the first "1" signal appeared at the same bit position.If most first "1"s appeared at significant bit positions, the auditory impression was of noisiness.

Figure 8 plots bit position on the x-axis and sampling points on the y-axis[3]. The objective measurement's results matched those of subjective auditory impression. Most of the first "1"s in Jazz and classical music data are from the 6^{th} to 15^{th}, and the 12^{th} and 13^{th} bits. The speech files thus had about 150000 sampling points that had the first bit in the 14^{th} bit. In contrast, background noise was relatively low when most first "1"s were in the least significant bit position. Figure 8 plots bit position on the x-axis and sampling

[3]Each piece of sampling data was 30 seconds long (32 kHz, 16 bits, two channels)

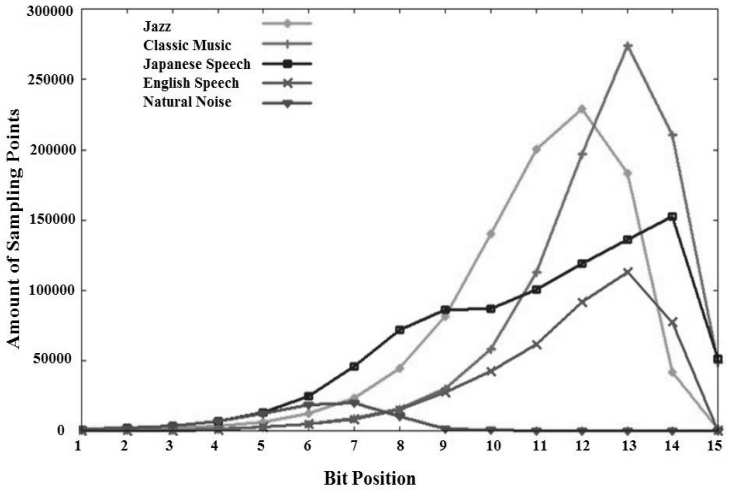


FIGURE 8. First "1": Structural information of significant bits

points on the y-axis[4]. The objective measurement's results matched those of subjective auditory impression. Most of the first "1"s in Jazz and classical music data are from the 6^{th} to 15^{th}, and the 12^{th} and 13^{th} bits. The speech files thus had about 150000 sampling points that had the first bit in the 14^{th} bit. In contrast, background noise was a quiet one when most first "1"s were in the least significant bit position.

4.2. **Overview of Experiments.** The experiments and their purposes are described below:

Experiment 1: Evaluation of extreme case

Digital speech signals typically have white noise as well as quiet parts. To test whether it is possible to embed an acoustic signal sequence in another quiet signal, we recorded natural background noise occurring at the midnight in our lab as cover data. The secret Japanese speech data was recorded when the embedding process started. To analyze the differences between the stego and cover data, the cover quiet natural noise was initially normalized to "0".

Experiment 2: Evaluation of bit positions

This experiment tried to determine whether our method is robust when secret data is embedded into significant bit positions in the cover. In it, we

(1) considered the interference spectrum of the significant-bit embeddings,

(2) designated the bit locations: 1^{st} and 2^{nd}, 1^{st} and 4^{th}, etc., and

(3) repeated the embedding process with two designated bit positions from the 1^{st} bit (LSB) to the n^{th} bit ($n\varepsilon[1,8]$) of the cover. Once the embed positions were decided, the secret data was embedded in those positions of successive frames of the cover.

Experiment 3: Acoustic source

This experiment tested whether music data of different genre should have different embedding efficiencies. We

(1) considered the efficiency of the data structures' coincidence,

(2) used music data bit streams (the cover and embedded data were of the same category) of jazz, pop, rock, classic, natural noise, and speech, and

(3) set the embedding position to be the 7^{th} and 8^{th} significant positions.

[4]Each piece of sampling data was 30 seconds long (32 kHz, 16 bits, two channels)

Table 2. Mean Opinion Score

MOS	Embed Quality	Impairment
5	Excellent	Imperceptible
4	Good	Difficult to Distinguish
3	Fair	Slightly Annoying
2	Poor	Different Audio
1	Bad	Annoying

5. **Results.**

5.1. **Procedures.** The embedding should be of high quality, which means unperceivable even if a large amount of data is embedded in the cover. The extraction reliability of a steganography scheme generally depends on the features of the original data, on the embedding distortion, and on the robustness (the amount of calculations needed by the attacker [21]). There are two ways of evaluating the embedding quality: subjective and objective. The mean opinion score (MOS) provides a numerical indication of the perceived quality of received media after compression or transmission, and it is calculated by averaging the results of a set of standard subjective tests in which a number of listeners rate the audio quality of test sentences read aloud by male and female speakers over the communications medium being tested.

5.1.1. *SNR Analysis.* As a measure of embedding quality, the SNR of each stego data can be calculated from the quantization error of each sampling point of cover and stego data as follows:

$$SNR = 10 \log_{10} \frac{\sum_n c^2}{\sum_n d^2(c,s)} [dB] \tag{1}$$

Here, c stands for the original signal, s stands for the stego signal, and d stands for the relative difference in amplitude between the corresponding sampling points of the cover and stego data, with n meaning the number of sampling points during a certain time interval.

5.2. **MOS Evaluation.** Besides the SNR evaluation, we took the subjective MOS (mean opinion score) to be an indicator of the perceived quality of the stego data. About 15 members of our lab rated the quality of the stego data. Cover data and stego data were played at random. Listeners chose which one was stego and which one was cover data. Then, they rated the embedding quality by using the ranking of Table 2. According to auditory impression, most of the listeners said they could not distinguish between the cover data and stego data at all (MOS score: 5), while two of them gave MOS scores of 4, meaning that they found it difficult but possible to distinguish the two audio data. These results indicated to us that the embedding was successful and the quality impairment to the cover data was low.

5.3. **Test with Weighted Noise Cover Data.** As Figure 9, Figure 10 and Figure 11 show, the stego data sounded like quiet natural noise even when it had high-amplitude speech embedded in it.

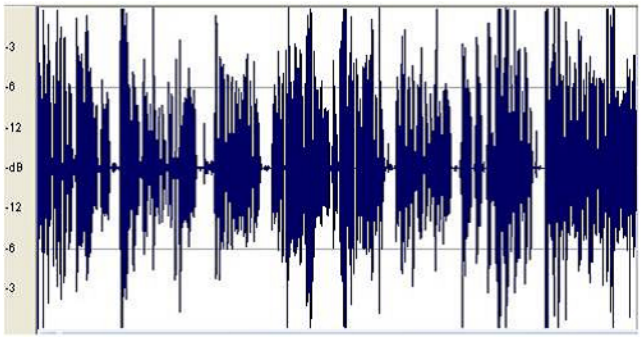


FIGURE 9. Embedded data: Japanese speech

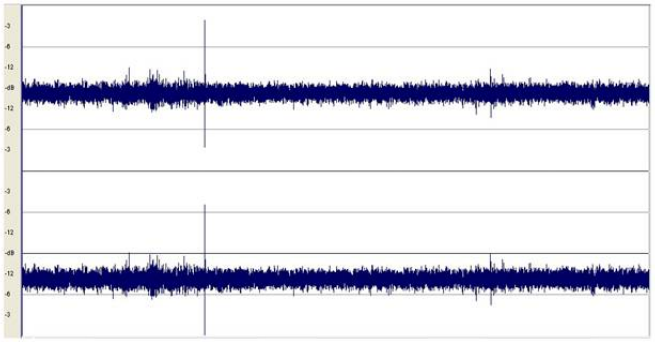


FIGURE 10. Cover data: natural noise

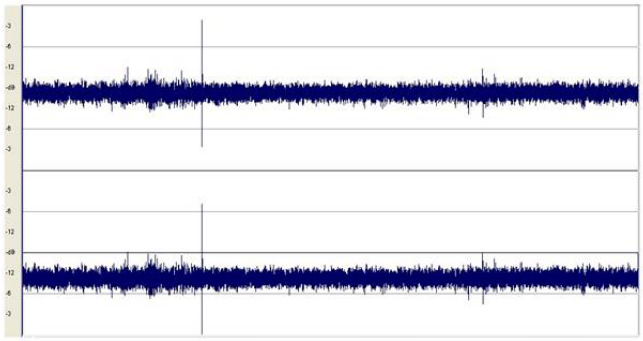


FIGURE 11. Stego data: natural noise with speech

5.3.1. *Evaluation by Significant Bit.* We did simulations to see how much of a difference there is between stego data with embeddings at positions with different levels of significance. We found that
(1) the embedding quality was the best when the secret data was embedded in the least significant bit positions: 1^{st} and 2^{nd};
(2) it was still difficult to distinguish the differences between cover data and stego data even when the secret data was embedded in the 7^{th} and 8^{th} significant bit positions.
(3) The SNR ranks were [52.15, 88.28], and there were few differences when one of the embedding positions was the 8^{th} bit (group (1,8), (5,8), (7,8), as shown in Fig. 11(a).

5.4. **Evaluation by Audio Category.** As shown in Fig.11(b),
(1) There were no obvious differences in embedding efficiency using different acoustic categories;
(2) The embedding system worked even when speech was embedded into white noise;

Table 3. System Configuration

OS	Windows Vista Business
CPU	Intel Core 2 Quad CPU Q9650 @ 3.00 GHz 3.00 GHz
Memory(RAM)	2GB
Programming Language	JAVA

Table 4. Real-time Ability: Experiment A (with real-time recording)

Source signal	Hiding time (μ s)	Extraction time(μ s)
Music	0.733	0.5997
Natrue Noise	1.3329	0.8664
Speech English(Female)	1.3329	0.5998

Table 5. Real-time Ability: Experiment B

Value	Hiding time (μ s)	Extraction time(μ s)
Max	0.9997	0.9330
Min	0.0666	0.0666
Average	0.4704	0.26655

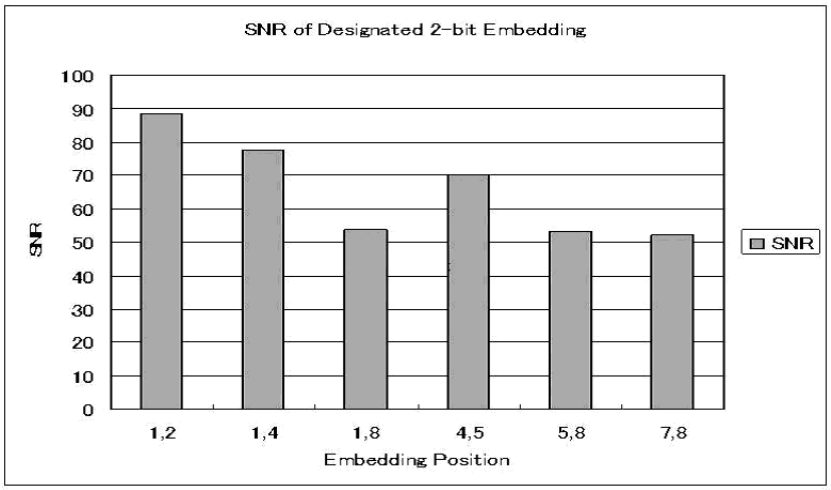
(3) The SNR ranks [51.68, 55.18]. The embedding quality was the best when the cover and secret data were both *speech* data. In this case, the cover and embed data consisted of a large speech data streams. Furthermore, the more the cover and embed data coincided, the better the embedding quality was.

5.5. **Real-time Performance.** We determined that it would be possible to have real-time secure communications between two PC terminals by recording, embedding, transmitting, and extracting the secret data in real-time. We performed an experiment to check whether the stego data could be generated and transmitted to another host. The cover data was sampled for about 30 seconds. The sender used a microphone to record the secret speech while the cover data was being played as a stream. Stego data was generated by the algorithm and sent to the receiver terminal.

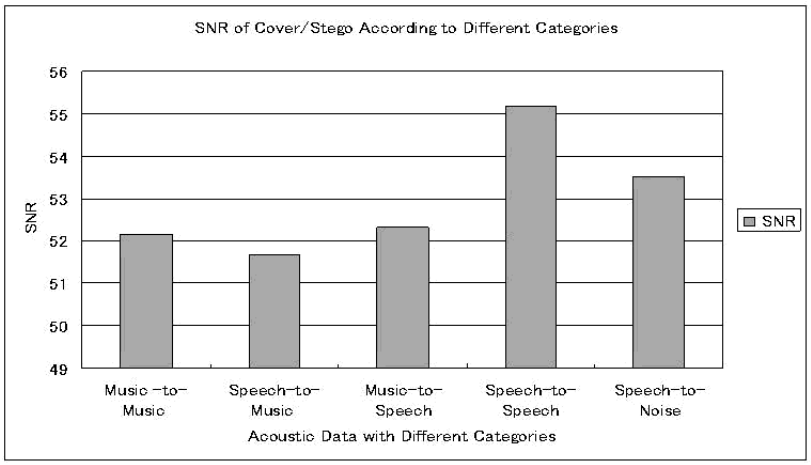
To assess the running times of hiding and extraction, we embeded speech in other speech with LSB insertion. The running times with different assigned hiding bit positions and different audio categories are supposed to be similar. In Experiment A, we recorded secret data in real-time when hiding process started (stream-into-file) ; in experiment B, we embedded the wav speech file generated in Experiment A (secret data) in other speech (English, Female) file (file-into-file). The specifications of the PC we used in this experiment are listed in Table 3. Many experiments were run and Table 4 and Table 5 list the maximum, minimum and average values.

For telephony to be successful, callers can speak simultaneously if the latency is 150 millisecondsor less according to ITU-T Y.1541 Standard [22]. In this paper, the maximum hiding time is 0.9997 milliseconds and extraction time is 0.9330 milliseconds, which means hiding and extraction are in real-time. Furthermore, by comparing the running-time performances, it became clear that there was little difference between hiding a stream into a file (while recording in real-time) and hiding a file into another file.

6. **Conclusion.** We developed and evaluated a real-time multi-bit audio-to-audio steganography method. Embedding quality varies according to the significant bit of the cover data



(a) SNR of 2-bit embedding



(b) SNR for different audio categories

FIGURE 12. SNR robustness

in which the secret data is embedded. In this method, secret speech data is recorded using a microphone and embedded in acoustic cover data synchronously. The intended receiver extracts the secret speech data from the stego data using a shared secret key. We measured the embedding performance of different acoustic categories.

Our method had optimal embedding performance even when 2 bits of secret data were simultaneously embedded in the 7^{th} and 8^{th} bit positions at a sampling point of the cover data stream. Subjective tests proved that it is difficult to distinguish the cover data from the stego data.

The SNR analysis revealed a few differences when the acoustic data came from different categories. Socket covert channels can enhance the robustness of this scheme.

6.1. **Future Work.** We used a two-bit arbitrarily assignment embedding model in our experiment. In the future, we shall analyze the coincidence between the cover and the embedded data before evaluating the embedding performance. Furthermore, we can extend our method to 4-bit embedding. It still remains for us to work out whether it is safe to embed more continuous secret data bits and how many bits at most can be embedded in a sampling point of the cover at one time.

(1) The embedding positions were arbitrarily assigned. However, once the embedding

positions are determined, all of the secret data will be embedded in these specified positions. This is a weak point that can be exploited by statistical analysis or repetition attacks. This paper did not discuss detection or countermeasures against such attacks. Steganalysis will be considered in a future study.
(2) Our scheme uses only wav data. Wav data has high acoustic quality but its amount grows large if the sampling rate is high. Speech recognition doesn't require high acoustic quality, so embedding together with a synchronized compression process should be considered.

REFERENCES

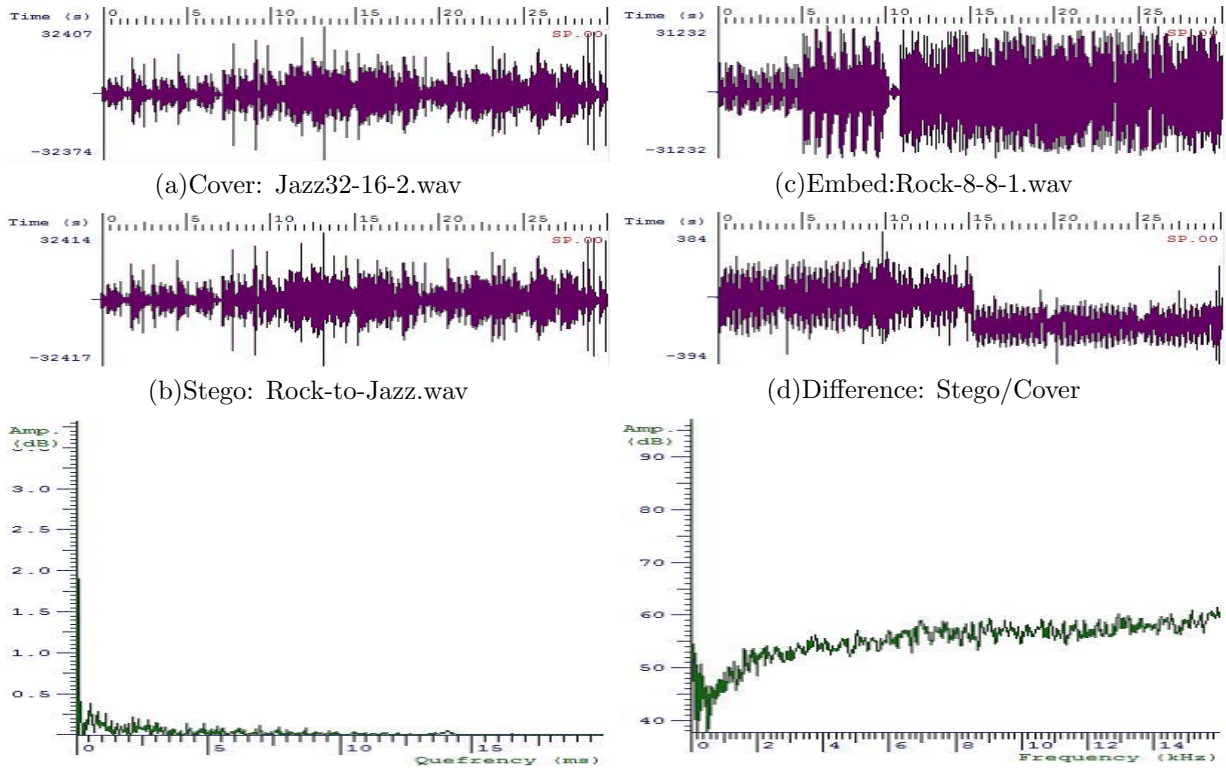
[1] N. F. Johnson, Z. Duric, and S. Jajodia, *Information Hiding Steganography and Watermarking-Attacks and Countermeasures*, Kluwer Academic Publishers, 2001.
[2] N. Segawa, Y. Murayama, and M. Miyazaki, Information hiding with a handwritten message in vector-drawing code, *Proc. of the 35th Annual Hawaii International Conference on System Sciences*, pp. 4027-4033, 2002.
[3] M. Hosei, Acoustic data hiding method using sub-band phase shifting, *Technical Report of IEICE*, EA, vol. 106, pp. 7-11, 2006.
[4] J. S. Pan, H. C. Huang, and L. C. Jain, *Intelligent watermarking techniques*, World Scientific Publishing Company, Singapore, Feb. 2004
[5] J.S Pan, H.C Huang ,L.C. Jain, and W.C Fang, *Intelligent Multimedia Data Hiding: New Directions*, Springer, Berlin-Heidelberg, Germany, Apr. 2007
[6] G. Cao, Y. Zhao and R. R. Ni, Edge-based blur metric for tamper detection, *Journal of Information Hiding and Multimedia Signal Processing*, vol. 1, no. 1, pp. 20-27, Jan. 2010
[7] H. Wang, J. Liang and C. C. Jay Kuo, Overview of robust video streaming with network coding, *Journal of Information Hiding and Multimedia Signal Processing*, vol. 1, no. 1, pp. 36-50, Jan. 2010
[8] H. C. Huang, Y. HChen, and A. Abraham, Optimized watermarking using swarm-based bacterial foraging, *Journal of Information Hiding and Multimedia Signal Processing*, vol. 1, no. 1, pp. 51-58, Jan. 2010
[9] X. P. Huang, R. Kawashima, N. Segawa, and Y. Abe, The Real-time steganography based on audio-to-audio data bit stream, *Technical Report of IEICE*, ISEC, vol. 106, pp. 15-22, September 2006.
[10] X. P. Huang, R. Kawashima, N. Segawa, and Y. Abe, Design and implementation of synchronized audio-to-audio steganography scheme, *Proc. of the 4th Int'l Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pp. 331-334, 2008.
[11] M. Wu and B. Liu, *Multimedia Data Hiding*, Springer-Verlag, New York, 2003.
[12] S. Katzenbeisser and F. A. Petitcolas (Eds.), *Information Hiding Techniques for Steganography and Digital Watermarking*, Artech House London, 2000.
[13] S. Chakrabartty, Y. Deng, and G. Cauwenberghs, Robust speech feature extraction by growth transformation in reproducing kernel Hilbert space, *IEEE Trans. Audio, Speech, and Language Processing*, pp. 473-480, 2007.
[14] L. Boney, A. H. Tewfik, and K. N. Hamdy, Digital watermarks for audio signals, *Proc. of International Conference on Multimedia Computing and Systems*, pp. 473-480, 1996.
[15] A. Ito and S. Makino, Designing side information of multiple description coding, *Journal of Information Hiding and Multimedia Signal Processing*, vol. 1, no. 1, pp. 10-19, Jan. 2010.
[16] I. Kusatsu, M. Niimi, H. Noda, and E. Kawaguchi, A large capacity steganography using acoustic dummy data, *Technical Report of IEICE*, pp. 27-32, 1998.
[17] S. K. Pal, P. K. Saxena, and S. K. Muttoo, The future of audio steganography, *Proc. of Pacific Rim Workshop on Digital Steganography*, pp. 136-145, 2002.
[18] I. Cos, M. L. Miller, and J. A. Bloom, *Digital Watermarking*, Morgan Kaufmann Pub, 2002.
[19] S. Wang, B. Yang and X. Niu, A secure steganography method based on genetic algorithm, *Journal of Information Hiding and Multimedia Signal Processing*, vol.1, no. 1, pp. 28-35, Jan. 2010.
[20] Z. H. Wang, T. D. Kieu, C. C. Chang, and M. C. Li, A novel information concealing method based on exploiting modification direction, *Journal of Information Hiding and Multimedia Signal Processing*, vol.1, no. 1, pp. 1-9, Jan. 2010.
[21] D. R. Stinson, *Cryptography-Theory and Practice*, CRC Press, 1995.
[22] G. Oliver, *Understanding and Managing Delay in the Global Voice Network*, PMC-Sierra webinar, 2003

Appendix: Frequency Figures of Experiment Results

Since embedding quality depends on acoustic source, we performed laboratory experiments to validate the proposed algorithm using different acoustic files as shown in Fig.13,14,15,16,17. Groups of sample data are used for the validation. They are:

(1) Group 1: Embed Music (Rock) into Music (Jazz)
(2) Group 2: Embed Music (Pop) into Speech (JPNSpeech-Male)
(3) Group 3: Embed Speech (JPNSpeech-Male) into Music (Classic)
(4) Group 4: Embed Speech (JPNSpeech-Male) into Noise (Backgroup Nature Noise at midnight in the Lab)
(5) Group 5: Embed Speech (JPNSpeech-Male) into Speech (ENGSpeech-Female)

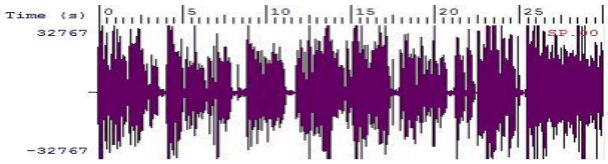
To check the embedding capacity, embedding positions are assigned to be 7^{th} and 8^{th} bits. Frequency of the difference between stego signal and cover signal is illustrated, including the Spectrum[5] and Cepstrum[6] information. Graphs with Sound Pressure Value[7] on the y-axis and Time(s) on the x are to represented the target data in (a),(b),(c),(d) in each result graphs. To detect the tiny variation, we zoom in the value of Y-axis in difference graph(d).



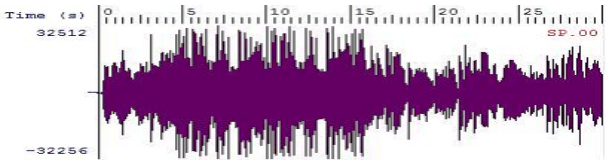
(a)Cover: Jazz32-16-2.wav
(c)Embed:Rock-8-8-1.wav
(b)Stego: Rock-to-Jazz.wav
(d)Difference: Stego/Cover
(d1)Cepstrum of difference
(d2)Spectrum of difference

FIGURE 13. Group 1: Embed Music into Music

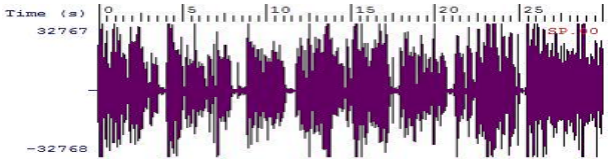
[5]This spectrum signal is generated from a pre-emphasised and windowed section of the signal.
[6]The cepstrum is the spectrum of the log magnitude spectrum of the signal.
[7]Each sample points is stored as a linear, 2's-complement value. Complement here is the significance bit order, for example, 7^{th} (complement value is 7)and 8^{th}, etc.



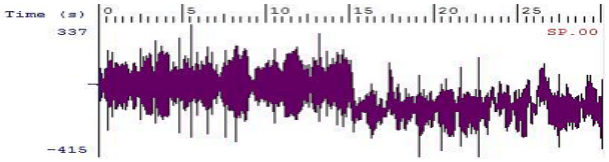
(a)Cover: JPNSpeech32-16-2.wav



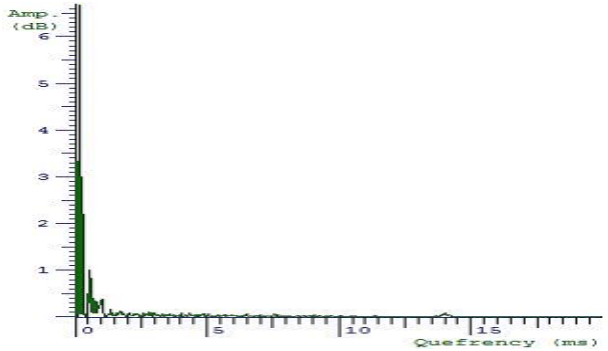
(c)Embed:Pop-8-8-1.wav



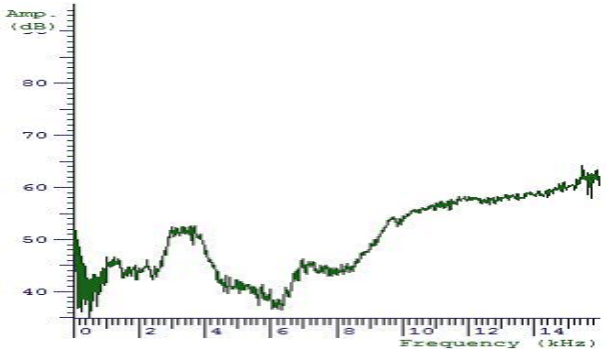
(b)Stego: Pop-to-JPNSpeech.wav



(d)Difference: Stego/Cover

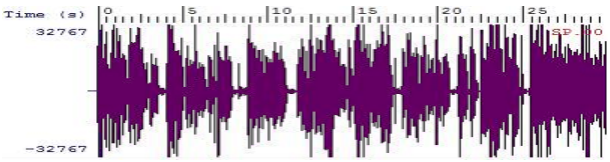


(d1)Cepstrum of difference

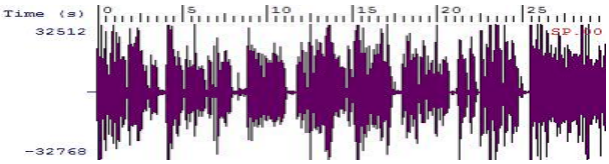


(d2)Spectrum of difference

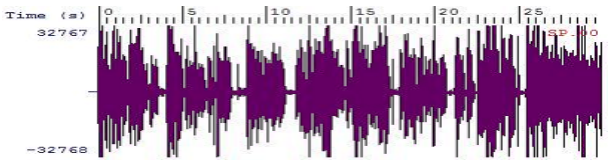
FIGURE 14. Group 2: Embed Music into Speech



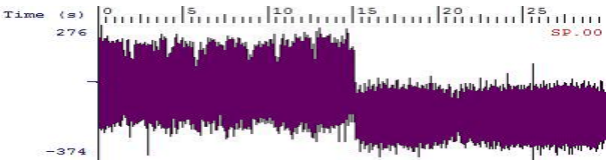
(a)Cover: Classical32-16-2.wav



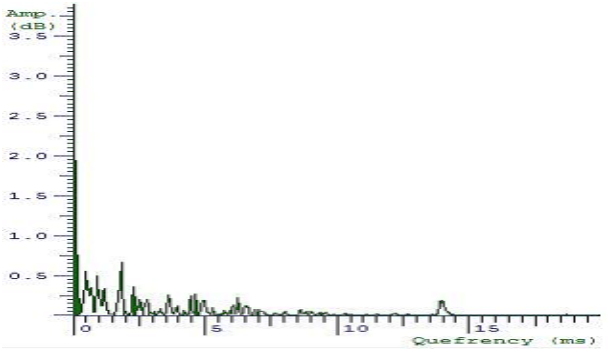
(c)Embed:JPNSpeech-8-8-1.wav



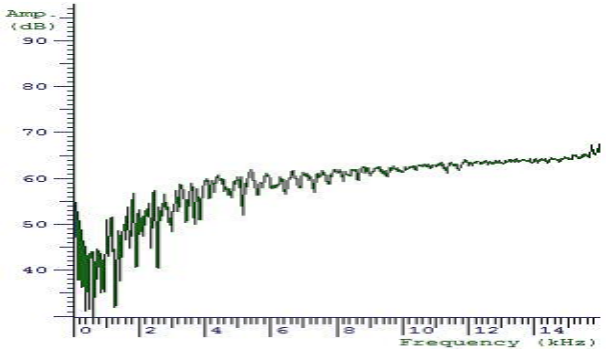
(b)Stego: JPNSpeech-to-Classic Music.wav



(d)Difference: Stego/Cover



(d1)Cepstrum of difference



(d2)Spectrum of difference

FIGURE 15. Group 3: Embed Speech into Music

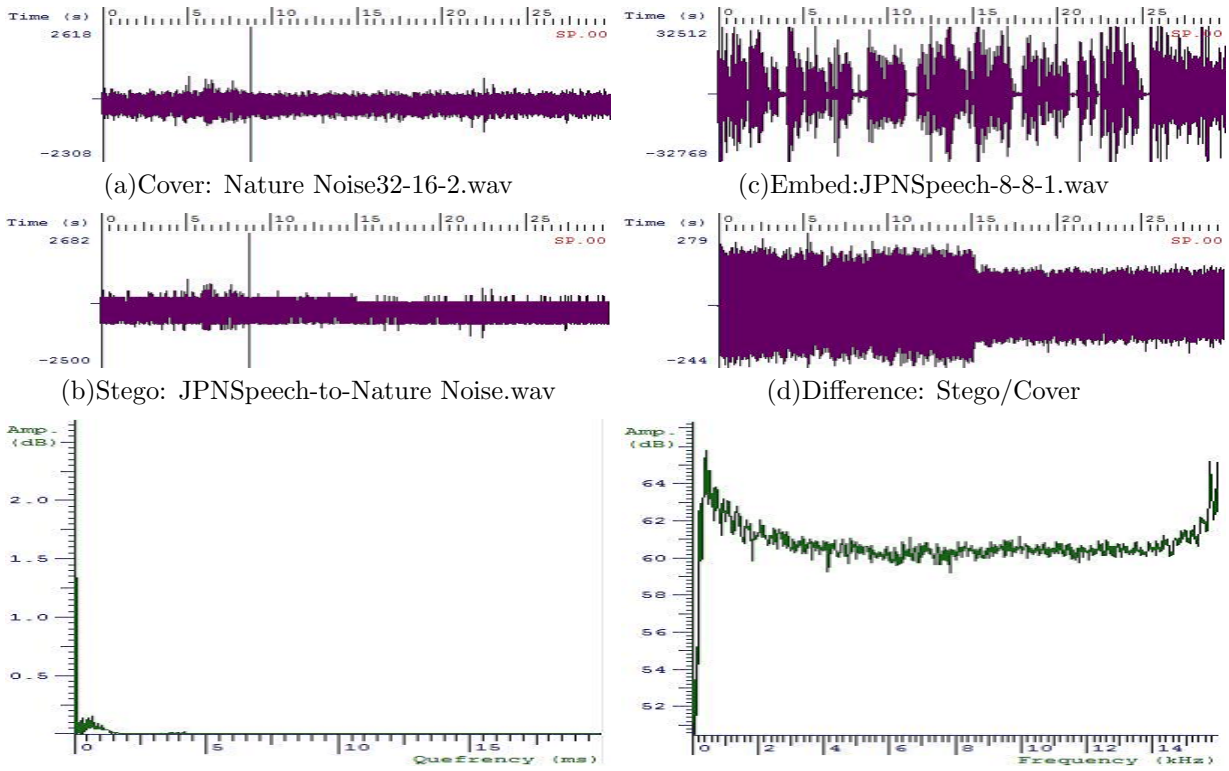


FIGURE 16. Group 4: Embed Speech into Weighted Noise

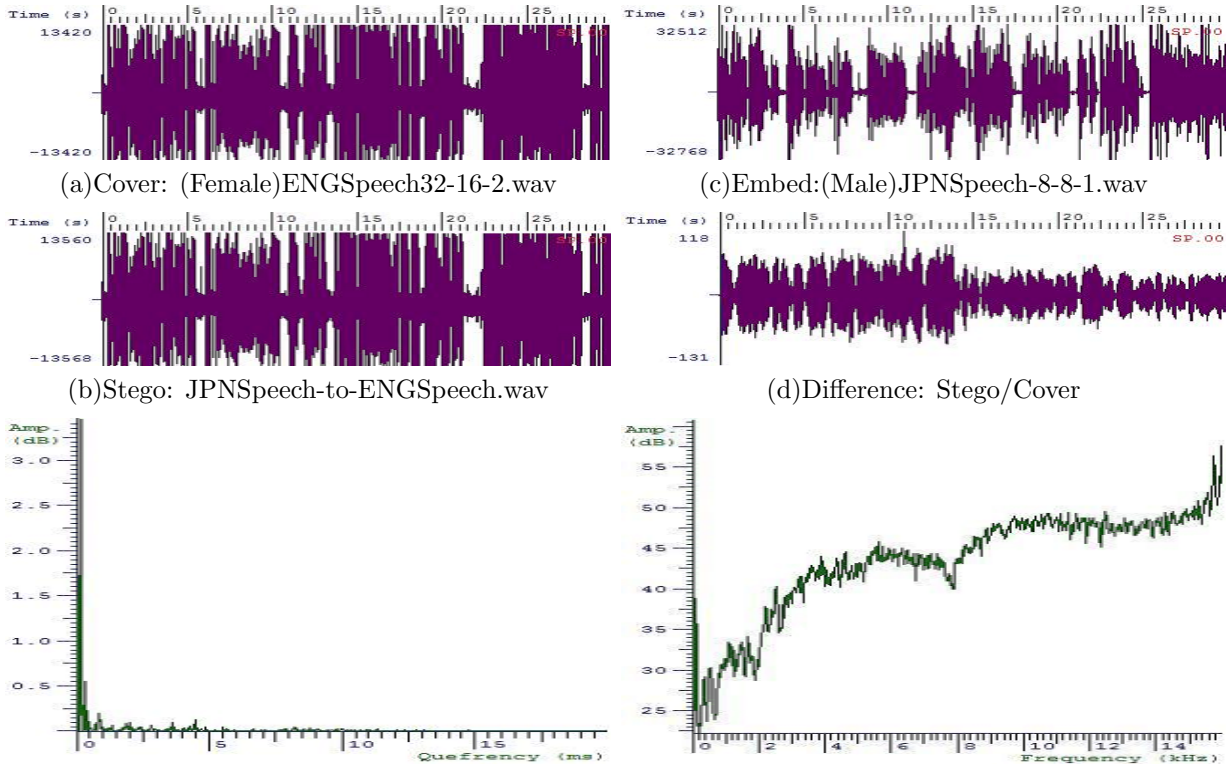


FIGURE 17. Group 5: Embed Speech into Speech