Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/MBE

Mathematical Biosciences and Engineering, 16(4): 3183–3194
DOI: 10.3934/mbe.2019158
Received: 24 February 2019
Accepted: 01 April 2019
Published: 12 April 2019

Research article

High capacity reversible data hiding in MP3 based on Huffman table transformation

Dingwei Tan, Yuliang Lu*, Xuehu Yan, Lintao Liu and Longlong Li

National University of Defense Technology, Hefei 230037, China

* **Correspondence:** Email: publictiger@126.com.

Abstract: In practice, most audio files such as MP3 and AAC are stored and transmitted in the form of compressed files, which can serve as the cover in audio steganography. Currently, the prevailing audio steganography methods are not ideal because of the drawbacks. Some are characterized by low capacity while others are irreversible. In this paper, we propose a method to embed secret messages in MP3 encoding. Our strategy is to hide the information by Huffman table transformation. We extract secret information by analyzing side information. Experimental results show that our method can greatly improve the steganographic capacity with low distortion and high security. Meanwhile, it featured with higher decoding rate and reversible over some state-of-the-art methods.

Keywords: MP3 steganography; data hiding; Huffman table; reversible; large capacity

1. Introduction

Nowadays image-based information hiding technology has been widely studied, inspiring researchers to use the audio as the carrier to hide information. There are two types of audio-based steganography: uncompressed domain audio steganography and compressed domain audio steganography. Most of the researches focus on uncompressed domain (as wav audio) [1–5]. However, raw audios are not suitable for storage and transmission. In practice, most audio files are stored in the form of compressed files. So, more and more researchers are pay attention to explore the steganographic method for compressed audios. On the other hand, if compressed audios are served as the cover media, they will not cause more suspicious than raw audios because compressed audios are prevalently exchanged and widely used. Therefore, compressed audios have more potential to be used for steganography [11].

MP3 is becoming an increasingly popular carrier for audio steganography at present, because it's very convenient to store and share. In recent years, some steganographic methods for MP3 audios have been proposed [6–10]. MP3Stego [6] is a well-known steganographic algorithm, which has good

transparency and robustness, but it has a low capacity. Gao et al. [7] proposed a method for information hiding by constructing Huffman codeword. Yang et al. [8] also used a similar method but their method improved the capacity. Yang et al. [9] improved the method proposed in [7] by taking adaptive hiding. But it operated on codewords with high computation complexity and poor real-time performance. Zhang et al. [10] proposed a method based on MP3 linbits of Huffman codeword by analyzing the structure of MP3 linbits. In fact, most codewords don't have linbits. At the same time, changing linbits has a great influence on the perceived quality of audio and it's irreversible. Yan et al. [11] used Huffman table swapping for information hiding. However, in this method, many Huffman tables are not used. And they used different levels of Huffman table to encode, which has a great impact on audios.

In order to improve the capacity and maintain better imperceptibility and undetectability, in this paper, we propose a reversible data hiding in MP3 audios based on Huffman table transformation. We used the same level table to transform, so that the original good imperceptibility and security are further improved. At the same time, we used all the tables in the big_value region to make it possible to embed more information. Experimental results show that the hidden capacity of this method has been greatly improved, and its imperceptibility and undetectability are better than traditional methods.

The rest of this paper is organized as follows. Section 2 introduces MP3 bitstream structure. Section 3 gives Huffman table exchange strategy and how the secret message is embedded and extracted. Experimental results are in Section 4. Finally, conclusions are drawn in Section 5.

2. Structure of MP3 bitstream

MP3 encoding algorithm is divided into six steps: framing, subband filter, PAM, MDCT, quantization, and Huffman encoding [12]. Figure 1 shows the general MP3 encoding process.

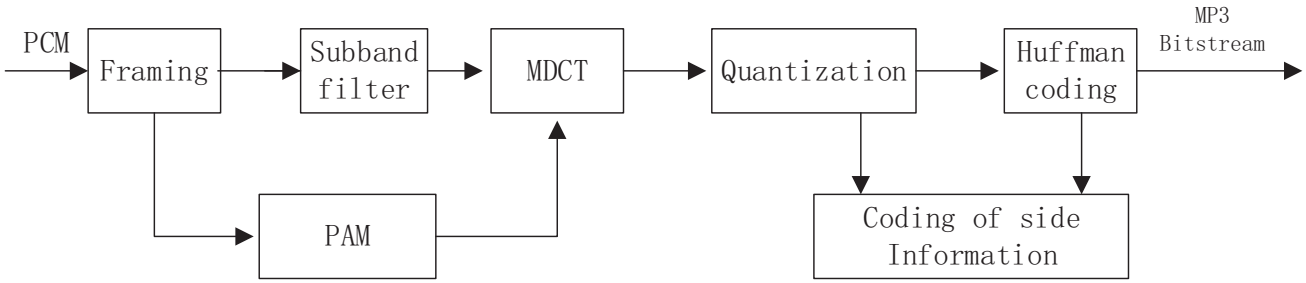


Figure 1. MP3 Encoding Process

The input PCM audio data is processed by frame, and each frame includes 1152 PCM sampling values. Frame is divided into two granules, that is, each granule contains 576 frequency coefficients. The structure of granule is shown in Figure 2.

From Figure 2 a granule is orderly divided into three kinds of regions: big_value region, count1 region and rzero region. Since the values in the rzero region are all zeros, these coefficients are not necessary to be encoded. Thirty-four Huffman tables are used to encode the codeword of the coefficients. Table 32 and table 33 are used for count1 and the values of this region belongs to $\{\pm 1, 0\}$. The big_value region is divided into region0, region1 and region2. Each region is encoded independently by Huffman tables from table 0 to table 31. From Table 1 we know the max value of each table. Table

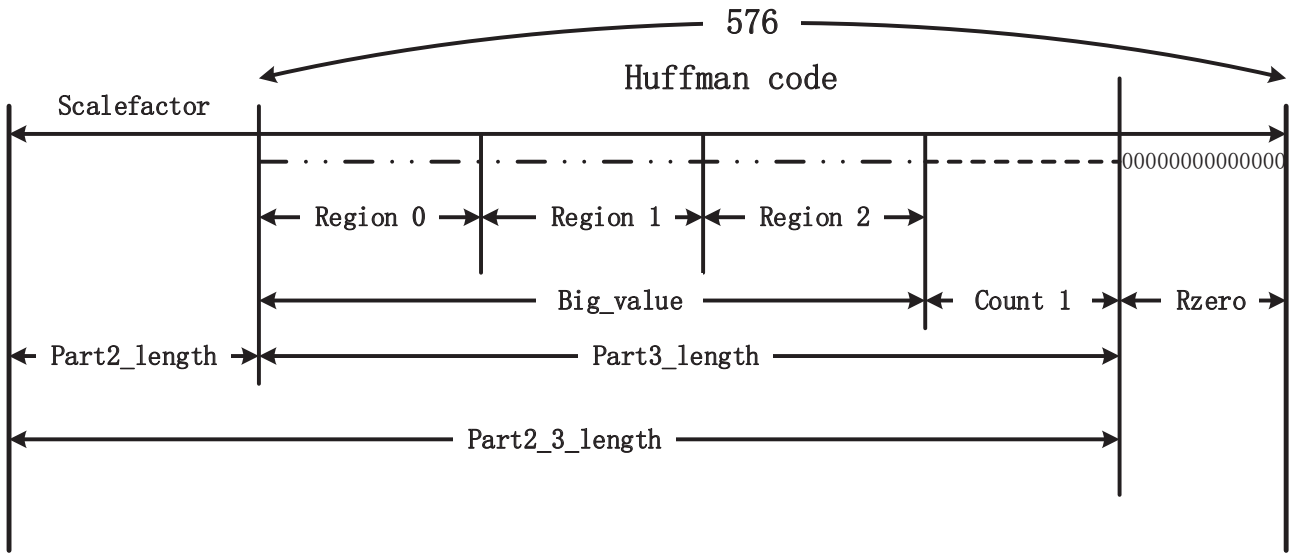


Figure 2. The Structure of Granule.

4 and table 14 are not used in MP3 standard. If two tables have the same maximum, we consider them at the same level. For example, table 5 and table 6. In addition, the codeword in all tables can represent a maximum of 15. If the coefficient is greater than 15, a bit stream called linbits is used to represent the extra value. The maximum value of the binary encoding is $15 + 2^{linbits}$. Some tables have the same codeword with different linbits. For example, tables 16 through 23 and 24 through 31 operates in the same codeword with different linbits.

3. Steganography and extraction algorithm

Quantization in MP3 standard is achieved through two nested iteration loops: a rate control loop (inner loop) and a distortion control loop (outer loop). The inner loop conducts the quantization of the MDCT coefficients, determines the required quantization step size, and selects the Huffman table. Quantization is a lossy process. If embedding before quantization, hidden message may be lost and cannot be extracted completely. Instead, Huffman coding is a lossless process, so we hide information by changing the table selection. The coefficients are quantized with an increasing step size to meet the available bits from using one Huffman table. The outer loop then controls the noise generated from quantization. The two-stage loops are repeated until the noise below the masking threshold.

3.1. Huffman table transform

As we can see from Table 1, some tables have the same maximum value, so they can encode the same range of values. As an example, we consider the quantization values from a subregion: (1, 0, 0, 1, 0, 2). According to the MP3 standard encoding rules, a table's maximum value not less than 2 can encode it. However, in order to minimize the encoded bit stream, the MP3 standard rules encode with table 2 and 3 which are closest to subregion's maximum value. At the same time, the standard rules select a table which required a smaller number of bits. In this example, we can see from Table 2 that the standard rules will select table 3 for encoding. And we do data embedding by changing the

Table 1. Properties of Huffman tables in MP3 standard.

Table index	Max value	Linbits	Table index	Max value	Linbits
0	0	without	16	16	1
1	1	without	17	18	2
2	2	without	18	22	3
3	2	without	19	30	4
4	not used	without	20	78	6
5	3	without	21	270	8
6	3	without	22	1038	10
7	5	without	23	8206	13
8	5	without	24	30	4
9	5	without	25	46	5
10	7	without	26	79	6
11	7	without	27	142	7
12	7	without	28	270	8
13	15	without	29	526	9
14	not used	without	30	2062	11
15	15	without	31	8206	13

Table 2. Huffman table example selected in standard coding.

Quantized values	Codeword in Table 2	Codeword in Table 3
(1,0)	011	001
(0,1)	010	10
(0,2)	000001	00001
Total bits	12 bits	11 bits

selection of table.

In the embedding procedure, embedding rules are shown in Table 3. From the table if the max quantization value is 7, the table required the least number of bits in tables 10, 11, and 12 is selected in the standard MP3 encoding process. Our method select table based on the bit we need to embed. If the coding table selected by the MP3 standard rules is 10, and the embedded secret bit is ”1”, it can be embedded without changing the information. If the embedded secret bit is ”0”, the selected table is changed to 11 to perform encoding process. If the coding table selected by the MP3 standard rules is 12, and the embedded secret bit is ”1”, the selected table is also changed to 10 and encoded with it. If the embedded secret bit is ”0”, it can be embedded without changing the information. As a result, the optimal searched Huffman table is modified to carry the secret messages. In the extracting procedure, we can directly judge the hidden bit is ”0” or ”1” by the selected table.

The reason we construct the transformation table like this is as follows: (1) Table 0 has no Huffman codeword, and table 4 and table 14 are not used, so these tables are not changed. (2) In order to minimize the change, we select the same level table to swap. (3) In order to achieve blind extraction, each table only corresponds to one at the time of extraction. By such construction, if the index of the

Table 3. Rules for Huffman tables transform.

Table index	Table index(transformed) bit=1	bit=0	Table index	Table index(transformed) bit=1	bit=0
0	\	\	16	16	17
1	1	3	17	18	17
2	2	3	18	18	19
3	2	3	19	20	19
4	\	\	20	20	21
5	5	6	21	22	21
6	5	6	22	22	23
7	7	8	23	31	23
8	7	8	24	25	24
9	9	8	25	25	26
10	10	11	26	27	26
11	10	11	27	27	28
12	10	12	28	29	28
13	13	15	29	29	30
14	\	\	30	31	30
15	13	15	31	31	23

table is in $H_0 = \{3, 6, 8, 11, 12, 15, 17, 19, 21, 23, 24, 26, 28, 30\}$, it means hidden information "0"; if it is in the index of $H_1 = \{1, 2, 5, 7, 9, 10, 13, 16, 18, 20, 22, 25, 27, 29, 31\}$, it means hidden information "1".

3.2. Embedding and extracting procedure

The embedding procedure is integrated with MP3 encoding process. As shown in Figure 1, secret information is embedded after quantization when Huffman encoding. The details of the information embedding process are as follows:

Input A raw cover audio, a stream of secret bits

Output A MP3 stego-audio

Step 1: Firstly, the secret information is converted into 0, 1-bit stream, which is preprocessed to scramble and encrypt the secret information to generate the encrypted secret information. Then, the length of the secret information is connected with the encrypted secret information to form the embedded secret information.

Step 2: Check the selection of the current table for the current granule. If it is in tables of {0,4,14}, move on to the next region of big_value or the next granule. If not, go to step 3.

Step 3: Change the selected table according to the rules and secret information in Table 3.

Step 4: Repeat step 2-3 until the end of MP3 file or all the secret information is embedded, an MP3 stego-audio is obtained.

The extraction process of the proposed method is also integrated with the MP3 decoding process. The details of the information extraction procedure are as follows:

Table 4. Comparison of maximum capacity for various embedding algorithms(bps).

Cover audio	96 kbps			128 kbps			192 kbps			256 kbps		
	[6]	[11]	proposed	[6]	[11]	proposed	[6]	[11]	proposed	[6]	[11]	proposed
blues	76.56	89.87	220.72	76.56	92.74	220.73	76.56	98.29	220.73	76.56	93.79	220.73
classical	76.56	88.63	229.65	76.56	95.21	229.66	76.56	94.37	229.66	76.56	93.57	229.66
country	76.56	98.69	229.35	76.56	97.77	229.36	76.56	99.82	229.36	76.56	88.09	229.36
folk	76.56	92.60	229.43	76.56	97.00	229.44	76.56	100.49	229.44	76.56	95.63	229.44
jazz	76.56	91.45	229.67	76.56	95.07	229.67	76.56	97.65	229.67	76.56	97.91	229.67
pop	76.56	91.99	225.27	76.56	91.83	225.28	76.56	101.14	225.28	76.56	98.32	225.28

Input A MP3 stego-audio

Output A stream of secret bits

Step 1: Decode the encrypted MP3 partly and check the selected table of current region.

Step 2: If the index of selected table belongs to H_0, extract bit information "0"; if the index of selected table belongs to H_1, extract bit information "1". And then go to the next region or next granule.

Step 3: Repeat above-mentioned steps until all the secret information is extracted.

If necessary, we can recover the quantized values by reanalyzing the MP3 bit stream after extracting the secret bits, and then recover the original audio using the MP3 encoding standard.

4. Experimental results and analysis

For methods like ours, such as in [6] and [11], the advantages of our method are manifested through experimental results. For methods in [9] and [10], we mainly conduct qualitative analysis due to the distinction between application scenarios and features. In order to verify the effectiveness of this method, we have done a lot of experiments. The experimental environment is as follows.

Hardware Environment: Intel Core i7-47900 CPU 3.60 GHz, 16.00 GB of RAM.

MP3 carriers: The experiment utilizes four MP3 sample library, each kind of sample library audio contains 80 MP3, including pop songs, rock songs, classical songs, folk songs, blues songs and jazz songs. Each samples sampling frequency is 44.1 kHz. Bit rate includes 96 kbps, 128 kbps, 192 kbps, and 256 kbps. The length of sample is 3 min.

Secret information: For convenience, random bits of 0 and 1 are used as secret information and are not encrypted or compressed before embedding.

4.1. Capacity

We measure the embedded capacity by the ability of unit time audio signal to hide the number of the secret information's bits. Bit/second (BPS) is defined as the ratio of the maximum capacity of the secret information to the length of the audio. Table 4 shows the experimental results with maximum hiding capacity. As can be seen from the table, the capacity of our method has greatly improved compared with those of other methods.

For MP3Stego, because it hides information by the part2_3_length, a granule can hide only one bit. If a mono cover audio is sampled at 44.1KHz, MP3Stego can achieve the maximum embedding rate at about $76bit/s$ $(44100 * 2/1152)$. For method [11], the capacity is about 1.2 to 1.3 times as much as that

Table 5. Comparison of SNR values.

Cover audio	Payload(bits)	96 kbps			128 kbps			192 kbps			256 kbps		
		[6]	[11]	proposed	[6]	[11]	proposed	[6]	[11]	proposed	[6]	[11]	proposed
blues	100	58.7454	62.0284	65.7372	61.1040	63.2581	63.8399	63.1180	65.3679	64.9515	64.9139	66.2401	66.9586
	200	59.4831	61.0547	62.6102	61.2061	62.1036	62.9865	62.9489	65.0339	64.7557	64.9049	65.5504	66.1522
	400	58.8429	60.6348	61.8303	60.9506	61.7054	61.9960	63.2284	64.1304	64.9694	64.8051	64.8835	65.3633
	600	58.8322	60.2345	61.3832	60.8527	61.6006	61.8381	62.8808	64.0773	64.3699	64.8216	64.6805	65.0411
classical	100	63.1325	66.5221	66.4820	64.0489	66.9734	63.8496	65.6666	68.0458	67.5608	66.3660	68.2919	65.9545
	200	63.1689	65.3453	66.3777	64.2051	66.3562	63.8489	65.6844	67.3449	67.6260	66.2930	67.8187	65.9489
	400	63.1521	65.0614	65.8525	64.0783	65.9827	63.8465	65.4745	66.5838	67.4403	66.2954	67.3010	65.9472
	600	63.1929	64.5768	65.3600	64.1640	65.7487	63.8335	65.6465	66.0927	67.0041	66.2938	66.4360	65.9341
country	100	62.7145	64.7231	64.9205	64.0900	66.1658	64.6848	65.6381	66.3828	67.7287	66.7950	67.9264	66.6125
	200	62.4478	64.1723	64.7286	64.1423	65.7296	64.6543	65.6128	66.3452	67.4288	66.8495	67.7183	66.6033
	400	62.4417	64.2462	64.5544	64.0470	65.3867	64.6039	65.6601	66.1581	66.3107	66.8382	67.2137	66.5790
	600	62.4839	63.8544	64.3372	64.0980	64.9326	64.5279	65.6432	66.0192	66.2142	66.8394	66.8297	66.5361
folk	100	59.5737	64.6741	67.4804	61.8503	65.4304	65.2386	64.4957	67.4898	67.6670	65.6349	68.2773	67.0935
	200	59.5723	64.1829	64.8691	61.7325	64.0466	65.2072	64.6881	66.2610	67.5224	65.5320	67.1758	67.0893
	400	60.5274	63.5734	64.9838	61.7904	63.6777	63.9351	64.5484	65.7909	66.5783	65.6657	66.5733	66.7551
	600	59.4366	60.0637	63.9421	62.1790	63.6173	63.9009	64.6095	65.1815	65.9381	65.4049	66.3406	66.6028
jazz	100	62.9431	65.7202	66.1969	63.4770	66.4231	64.6046	65.7856	67.1380	67.6897	66.5277	67.9240	66.5536
	200	62.9318	65.1733	65.5491	63.4632	65.9379	64.5669	65.4968	66.5707	66.5868	66.6791	67.5606	66.5486
	400	62.7745	64.1920	65.3207	63.4761	65.1841	64.5334	65.6472	66.0487	66.4677	66.4041	66.9545	66.4311
	600	62.7591	63.8605	65.0292	63.4728	65.0812	64.4878	65.6342	65.8460	66.2650	66.5280	66.6762	66.4183
pop	100	64.4774	66.2964	66.3017	65.5209	67.0340	65.7124	67.2542	68.2291	68.4043	68.0871	69.1222	67.5092
	200	62.6189	65.8030	66.1769	65.5897	67.1128	65.7076	63.8377	68.0676	68.3080	67.9254	68.8239	67.5019
	400	62.6128	65.3172	65.9884	65.4582	66.4150	65.6658	66.4507	63.8390	68.0819	68.0110	68.5200	67.4789
	600	62.6125	65.1119	65.6701	65.4974	66.0899	65.6303	63.8284	63.8384	63.8394	68.0080	68.3105	67.4742

of MP3stego, which is consistent with the experimental results in [11]. The reason why our capacity can increase so much compared with [11] is that all tables in big_value region are used.

In our method, three tables are selected for each granule, and only table 0 is excluded. Therefore, our capacity should be nearly three times as much as MP3stego. Experimental data showed that our capacity is 2.88 to 3 times as much as MP3stego, which means the capacity of our method greatly improved. In addition, our method doesn't conflict with MP3Stego. We can combine the two methods to further improve the capacity, but it also reduces undetectability.

4.2. Imperceptibility

According to [13], signal to noise rate (SNR) and objective difference grade (ODG) are generally used to evaluate imperceptibility. The SNR value is computed between the original audio (raw and unmarked) and the stego-audio (decompressed and marked). The experimental results are shown in Table 5. As can be seen from the table, the SNR of method [11] is much higher than that of [6], and the SNR of our method is higher than that of method [11] in most cases. Compared with method [11], our method uses the same level table to exchange, but there is no guarantee that our changes will be certainly smaller. The more hidden information we have, the more likely we have a higher SNR. ODG indicates how similar the test audio to the reference audio. Its range is [-4,0], where -4 denotes very annoying and 0 denotes imperceptible. If its value is greater than 0, it means the test audio has better perceived quality than the original audio. The experimental results are shown in Table 6. From the table, our method and [11] are superior to [6]. In addition, sometimes the experimental results of our method will be slightly worse than [11]. Because even if the same level table is selected for encoding, the size of the formed bit stream is not completely better than the different level table due to the difference in the quantized values, but the overall result is better.

Table 6. Comparison of ODG values.

Cover audio	Payload(bits)	96 kbps			128 kbps			192 kbps			256 kbps		
		[6]	[11]	proposed	[6]	[11]	proposed	[6]	[11]	proposed	[6]	[11]	proposed
blues	100	0.0028	-0.0207	-0.1380	0.0590	-0.0599	-0.1722	0.0592	0.0498	0.0074	0.0059	0.0056	0.0094
	200	0.0062	0.0484	-0.0346	0.0786	0.0265	-0.1912	0.0285	0.0498	0.0498	0.0060	0.0057	0.0094
	400	0.0034	0.0552	0.0183	0.0622	0.0698	-0.0534	0.0579	0.0503	0.0498	0.0060	0.0059	0.0095
	600	0.0040	0.0736	0.0308	0.0652	0.0766	-0.0386	-0.0241	0.0498	0.0498	0.0060	0.0059	0.0095
classical	100	0.0054	0.0147	0.0106	-0.0956	0.0049	-0.2418	0.0041	0.0023	0.0023	0.0021	0.0020	0.0062
	200	0.0195	0.0463	0.0115	-0.0966	0.0052	-0.2417	0.0028	0.0023	0.0023	0.0021	0.0020	0.0062
	400	0.0136	-0.0136	0.0199	-0.1134	0.0053	-0.2416	0.0043	0.0024	0.0023	0.0021	0.0020	0.0062
	600	0.0216	-0.0786	-0.0623	-0.0909	0.0071	-0.2415	0.0029	0.0027	0.0023	0.0021	0.0021	0.0062
country	100	0.0554	-0.0551	-0.1083	-0.0494	0.0282	-0.2090	0.0272	0.0270	0.0268	0.0223	0.0224	0.0243
	200	0.0516	-0.0364	-0.0674	-0.0708	0.0309	-0.2099	0.0270	0.0270	0.0268	0.0224	0.0224	0.0243
	400	0.0517	-0.0217	-0.0592	-0.0758	0.0313	-0.2090	0.0274	0.0271	0.0269	0.0224	0.0224	0.0243
	600	0.0506	0.0005	-0.0677	-0.0732	0.0371	-0.2090	0.0271	0.0272	0.0271	0.0223	0.0224	0.0243
folk	100	0.0469	0.0137	0.0289	-0.0338	0.0730	-0.2323	0.0269	0.0265	0.0264	0.0261	0.0259	0.0297
	200	0.0452	0.0285	-0.0012	0.0551	0.0854	-0.2341	0.0270	0.0267	0.0264	0.0260	0.0259	0.0297
	400	0.0394	0.0525	0.0203	-0.0037	0.0857	-0.1127	0.0269	0.0267	0.0268	0.0261	0.0259	0.0298
	600	0.0215	0.0585	0.0434	0.0926	0.0857	-0.1126	0.0270	0.0268	0.0268	0.0261	0.0260	0.0298
jazz	100	0.0198	-0.0795	-0.1668	-0.2047	0.0124	-0.2617	0.0299	0.0293	0.0289	0.0270	0.0265	0.0303
	200	0.0209	-0.0485	-0.1129	-0.2302	0.0128	-0.2617	0.0304	0.0294	0.0297	0.0265	0.0265	0.0303
	400	0.0145	-0.0023	-0.0859	-0.1638	0.0166	-0.2617	0.0306	0.0302	0.0298	0.0270	0.0265	0.0303
	600	0.0123	0.0311	-0.0504	-0.2242	-0.0086	-0.2621	0.0309	0.0303	0.0302	0.0270	0.0265	0.0302
pop	100	-0.0073	-0.0261	-0.0310	-0.1176	-0.1030	-0.2343	0.0049	0.0028	0.0027	0.0029	0.0028	0.0067
	200	0.0120	-0.0056	-0.0395	-0.1475	-0.0867	-0.2364	0.0090	0.0029	0.0027	0.0029	0.0028	0.0067
	400	-0.0048	0.0086	-0.0155	-0.1074	-0.0567	-0.2748	-0.3400	0.0107	0.0028	0.0029	0.0028	0.0067
	600	0.0007	0.0176	-0.0120	-0.1208	-0.0491	-0.2739	-0.3388	0.0110	0.0107	0.0029	0.0028	0.0067

4.3. Undetectability

We use variance, skewness and kurtosis to evaluate the undetectability [14–16]. In the experiments, we randomly select folk and classical music. Figure 3 and Figure 4 show the variance, skewness and kurtosis of the audio with different secret information embedded. From the figure, our statistical characteristics are very close to the clean version audio. At the same time, in the comparison with undetectability of method [6], we can see that our statistical characteristics are superior. In the case of a large number of experiments, the statistical characteristic is also slightly improved compared with [11]. To further prove the undetectability of our method, we used some steganographic analysis software for testing. The experimental results show that they cannot detect the secret information.

Compared with [9] and [10], due to the characteristics of our method, its capacity is smaller than that of [9] and [10]. But it also meets a large number of practical application requirements. Since we don't need to modify the quantized values, our method is better than [9] and [10] in terms of imperceptibility and undetectability. At the same time, our method is reversible. [9] and [10] need to read the Huffman linbits bit or codeword when extracting secret bits. Our method only needs to read the side information, making the extraction faster.

We also found that the size of the file will be different during the experiments, which means the bitstream we generated is not the same as the bitstream encoded by standard MP3 rules. However, first, compared with [11], our method uses the same level of table to transform, so the changes will be smaller. Second, we can see in the experiments that the change in bitstream size has little effect on the overall file size. When compressing with different compression software, the size of the generated MP3 is also different, and this difference is much larger than that in our scheme. Therefore, we believe that this change is secure when the attacker cannot obtain the original file.

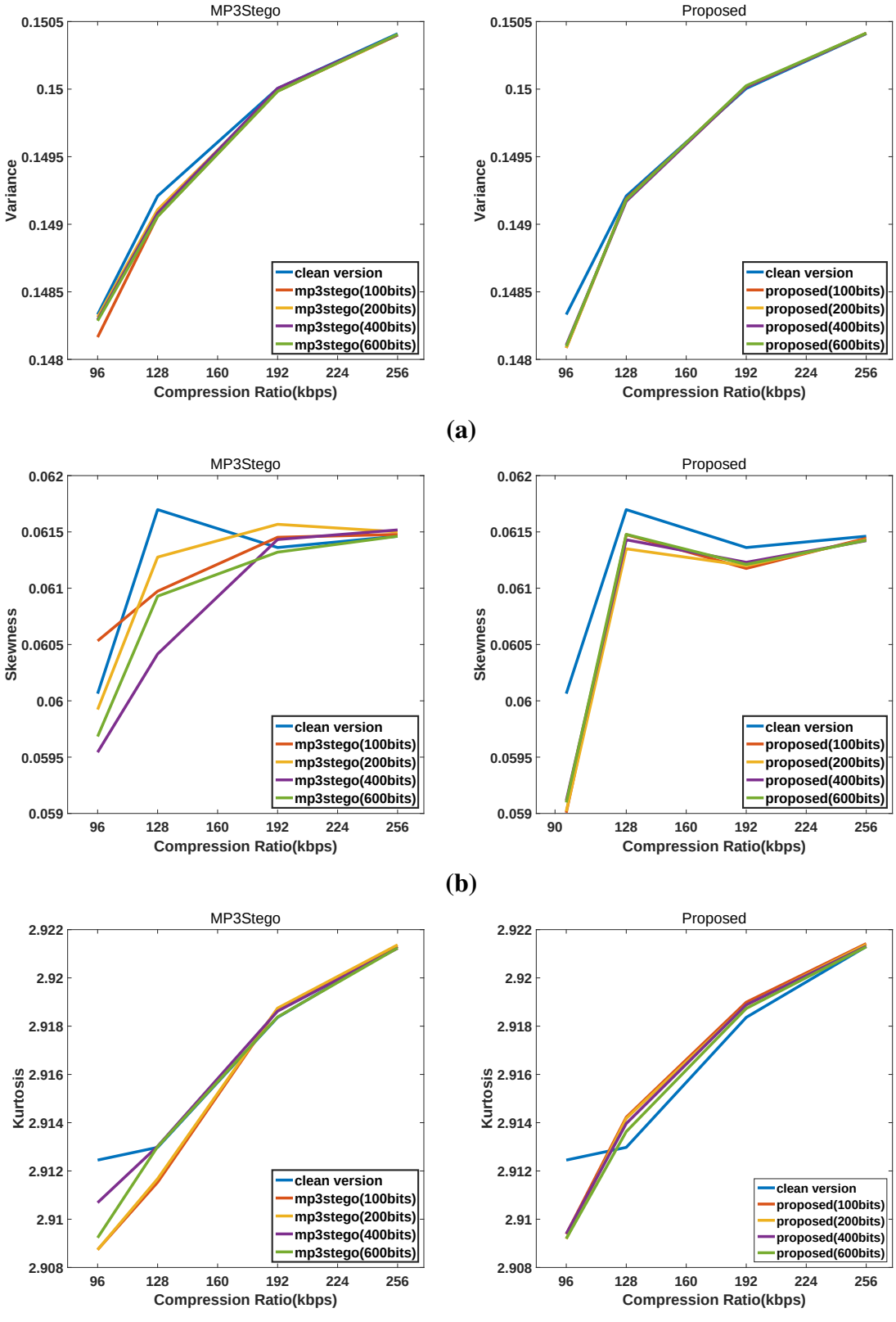
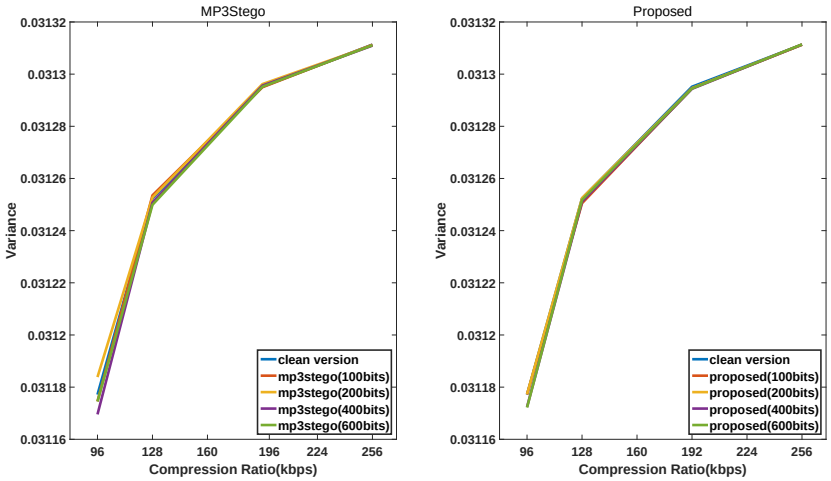
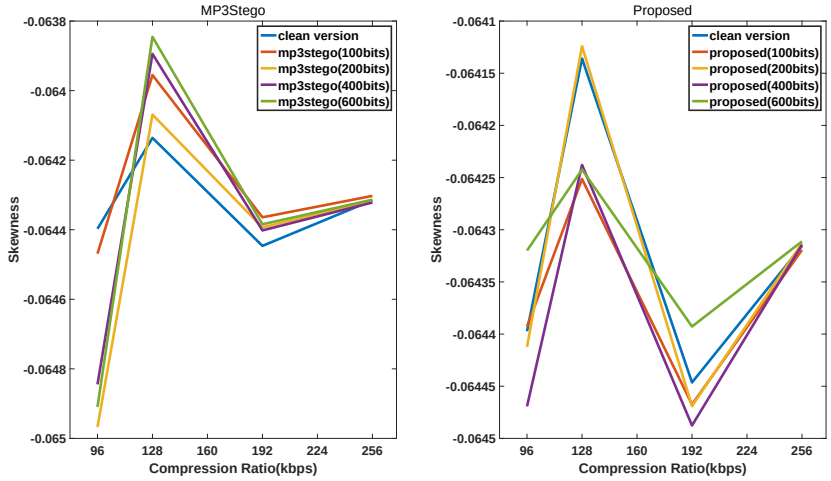


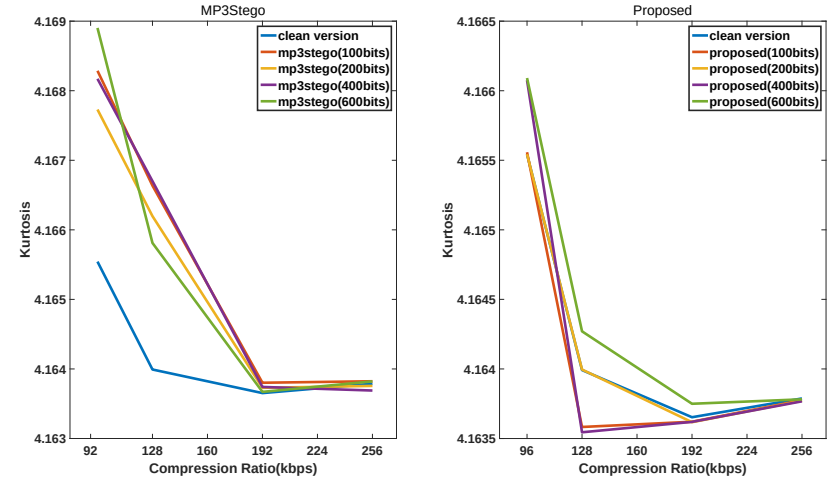
Figure 3. Comparison of folk audios (a) variance (b)skewness (c)kurtosis.



(a)



(b)



(c)

Figure 4. Comparison of classical audios (a) variance (b)skewness (c)kurtosis.

5. Conclusion and future work

In this paper, we propose an MP3 Huffman table transform steganography method. The method takes advantage of more Huffman tables and uses same level of Huffman table for transformation. The experimental results show that the capacity of our method has greatly increased compared with [6] and [11]. Meanwhile, when hiding the same number of bits, the imperceptibility and undetectability are also improved. Compared with other methods, our method also has the advantages of reversibility. We also found that due to the particularity of the table 15, which is at the segmentation point of whether linbits is used, the probability of table 15 being selected in the standard MP3 encoding process is much higher than that of table 13. However, our method will make the probabilities of the two tables close. Therefore, the next step is to control the changes to table 15 and use MP3Stego or other methods together to further increase the capacity. In addition, in the future work, we will study intelligent audio steganography and audio steganography based on air-gap [17–19].

Acknowledgments

This research was funded by [National Natural Science Foundation of China] grant number [61602491] and the Key Program of the National University of Defense Technology grant number [ZK-17-02-07].

Conflict of interest

The authors declare no conflict of interest.

References

1 I. Cox, M. Miller, J. Bloom, et al., *Digital watermarking and steganography*, 2008.

2 V. Viswanathan, Information hiding in wave files through frequency domain, *Appl. Math. Comput.*, **201** (2008), 121–127.

3 O. T. C. Chen and W. C. Wu, Highly robust, secure, and perceptual-quality echo hiding scheme, *IEEE Transact. Audio Speech Language Process.*, **16** (2008), 629–638.

4 M. L. Wang, H. X. Lin and M. T. Lee, Robust audio watermarking based on mdct coefficients, in *International Conference on Genetic and Evolutionary Computing*, 2013.

5 M. Bellaaj and K. Ouni, A robust audio watermarking technique operates in mdct domain based on perceptual measures, *Int. J. Adv. Comput. Sci. Appl.*, **7** (2016), 169–178.

6 F. Petitcolas, Mp3stego. computer laboratory, cambridge.

7 H. Y. Gao, The mp3 steganography algorithm based on huffman coding, *Acta Sci. Nat. Uni. Sunyatseni*, **46** (2007), 32–35.

8 D. Q. Yan, R. D. Wang and L. G. Zhang, A high capacity mp3 steganography based on huffman coding, *J. Sichuan Uni.*.

9 K. Yang, X. Yi, X. Zhao, et al., *Adaptive MP3 Steganography Using Equal Length Entropy Codes Substitution*, 2017.

10 Z. Ru, J. Liu and Z. Feng, A steganography algorithm based on mp3 linbits bit of huffman codeword, in *International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, 2017.

11 D. Yan and R. Wang, Huffman table swapping-based steganograpy for mp3 audio, *Multim. Tools Appl.*, **52** (2011), 291–305.

12 ISO/IEC, Information technologycoding of moving pictures and associated audio for digital storage media at up to about 1.5 mbit/s, 11172–3.

13 ITU, Methods for objective measurements of perceived audio quality, BS.1387.

14 C. Cachin, *An Information-Theoretic Model for Steganography*, 1998.

15 M. S. Atoum, *A Comparative Study of Combination with Different LSB Techniques in MP3 Steganography*, 2015.

16 X. Yan, X. Liu and C. N. Yang, An enhanced threshold visual secret sharing based on random grids, *J. Real-Time Image Process.*, **14** (2018), 61–73.

17 J. S. Pan, P. W. Tsai and H. C. Huang, Advances in intelligent information hiding and multimedia signal processing, *Smart Innovat. System. Technol.*, **81**.

18 M. Hanspach and M. Goetz, On covert acoustical mesh networks in air, *arXiv preprint arXiv:1406.1213*.

19 M. Guri, Y. Solwicz, A. Daidakulov, et al., Mosquito: Covert ultrasonic transmissions between two air-gapped computers using speaker-to-speaker communication.

AIMS Press