Improved MP3 Hiding Algorithm Based on Huffman Coding

Jun Ao, Rui Li and Chunbo Ma*
Guangxi Key Laboratory of Precision Navigation Technology and Application, Guilin University of Electronic Technology, Guilin, P.R. China
*Corresponding author

***Abstract*—To further increase the hiding capacity of MP3 files, an improved algorithm is proposed in this paper. Compare to previous work, this algorithm searches out additional 10 Huffman codeword pairs which meet the transparence requirements. With the newly found pairs, the hiding capacity is increased and the transparence is still preserved.**

Keywords-MP3; concealed message; Huffman codeword; codeword pair

I. INTRODUCTION

Encryption is an ordinary method to prevent unauthenticated person from obtaining sensitive messages. It exists for a long time and has been studied for several centuries. Until now, it is still a major approach to protect messages during network transmission. However, encryption has its problems. When an attacker finds some random binary data stream across the internet, he would guess that it is sensitive information that will not be recognized by anyone else. Then the attacker can intercept or even destroy the encrypted message. Actually, it is the randomness of the binary data stream that exposes the existence of the sensitive message. Is there a transmission technology which can transmit sensitive message at the same time conceal its existence? Yes, information hiding technology[1]. First of all, appropriate carrier type should be decided to carry out information hiding.

MP3 has very high compress ratio and good fidelity, and has become a very popular audio format in nowadays. Every day, large amount of MP3 documents are exchanged or distributed over internet. Obviously, it is an ideal carrier to carry hidden information, and raise few concerns.

In this paper, based on the work of Gao[2], we search out more Huffman code words which compliance with the transparence requirements. With the new found code pairs, the embedding capacity is improved, and the tone still keeps transparence. Experiments show that the performance of the algorithm is improved and the security still satisfy the transmission requirements.

II. PRELIMINARY WORK

As we have mentioned above, MP3 is an ideal carrier to execute information hiding. Many scholars have studied this topic. In 1997 and 2004, Yardimci[3] and XiaoXiao Dony[4] respectively studied the stegano method of phase modulation according to the property that human ear is insensitive in phase change. The advantage of this method is that it can resist MP3 compression and decompression.

Song, Xing and Li[5] designed an information hiding tool, named MP3Stego. They changed the size of the quantization error, and embedded the sensitive message into the length of the quantization.

Gao[2] proposed a Huffman based MP3 hiding algorithm. This algorithm hides the concealed message by changing the specific Huffman code in big-value area. Experiments show that the tone between the MP3 that contains the secret message and the original one is no difference. The success of the algorithm lies in the choice of the codeword pairs. Altogether ten Huffman codeword pairs are presented in their paper. In 2007, Gao add another three Huffman code pairs in their scheme, however, the embedding capacity did not significant increase.

The Huffman code that meets the requirements is in the big-value area. Then we firstly show the location of this area in Figure I.

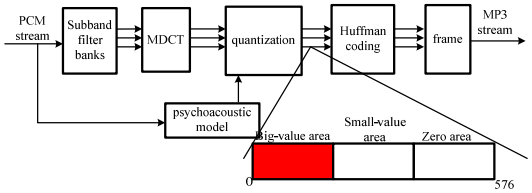

FIGURE I. THE LOCATION OF BIG-VALUE AREA

According the description in [2], the 32 Huffman code words used in their work are listed as follows.

TABLE I. 32 HUFFMAN TABLE CHARACTERS USED IN BIG-VALUE AREA

Huffman table index	Max code value	Linbits	Huffman table index	Max code value	Linbits
0	0	0	16	16	1
1	1	0	17	19	2
2	2	0	18	23	3
3	2	0	19	31	4
4	NULL	0	20	79	6
5	3	0	21	271	8
6	3	0	22	1039	10
7	5	0	23	8207	13
8	5	0	24	31	4
9	5	0	25	41	5
10	7	0	26	79	6
11	7	0	27	143	7
12	7	0	28	271	8
13	15	0	29	527	9
14	NULL	0	30	2016	11
15	15	0	31	8207	13

Obviously, the more Huffman codeword pairs are found, the greater the capacity grows. In other words, find new codeword pairs are a direct way to improve the embedding capacity.

III. MP3 FORMAT-BASED AUDIO STEGANOGRAPHY ALGORITHM

A. *Searching New Huffman Codeword Pairs*

In the algorithm mentioned by Gao, the hiding process is implemented by changing the special Huffman code words. Different MP3 has different number of codeword pairs which meet the requirements. That means, at the beginning, we should compute the conceal capacity of the MP3 file, and the decisive factor is the number of the special code words. We search and find additional 10 codeword pairs that meet the transparence requirements, and the conceal capacity is improved greatly.

The concrete search method and process is illustrated in Figure II. Assume that "idx", "hlen" and "hcod" denote the location of the coding table, codeword length, and codeword, respectively. Without loss of generality, we will describe how to search codeword which length is 8 in coding table t15.

By checking table and computing, the code words which length is 8 in coding table t15 are listed as follows.

TABLE II. CODEWORD WITH LENGTH 8 IN CODING TABLE

idx	x	y	hlen	hcode
22	1	6	8	61
23	1	7	8	51
24	1	8	8	42
38	2	6	8	59
39	2	7	8	48
40	2	8	8	40
53	3	5	8	63
54	3	6	8	55
68	4	4	8	67
69	4	5	8	57
80	5	0	8	77
83	5	3	8	66
84	5	4	8	58
85	5	5	8	52
98	6	2	8	60
99	6	3	8	56
100	6	4	8	50
113	7	1	8	53
114	7	2	8	49
130	8	2	8	41
146	9	2	8	67

With formulation, $idx = x * ylen + y$ one can get the x and y which corresponds to a codeword. Here $ylen$ denotes the max length of y in coding table t15. Based on the Table II, we describe the concrete steps and process to search a Huffman codeword which meet the transparence requirements.

(1) Determine whether "0" exists in the x and y corresponding to the Huffman codeword. If it is true, then reject the codeword; otherwise go to the next step.

(2) Two ways are parallel executed. In one way, we assume the is constant, and the Euclidean distance between and is 1; in another way, we assume the is constant, and the Euclidean distance between and is 1. Since these two ways implement the same process, we just describe one way in detail.

(3) Verify whether the newly found codeword can be used in hiding message. If it is true, compute the hiding capacity $Vlm1$ of the codeword pair, or else reject this codeword.

(4) To judge in the same way whether the newly found codeword pair contains a codeword which belongs to other pairs. If it is true, compute the hiding capacity and compare with $Vlm1$ to select the big one to save as $Vlm1$, and then execute follow step. If it is false, directly implement next step.

(5) To judge whether any codeword appears in two ways. If it is true, compare $Vlm1$ with Vlm2, and save the big codeword pair. If it is false, directly finish the step 5.

FIGURE II. SEARCHING PROCESS

As above described, we can search correct codeword pairs with different length in each coding table. With this approach, additional 10 new Huffman codeword pairs are obtained and shown in the following table.

TABLE III. NEW HUFFMAN CODEWORD PAIRS

x	y	hlen	hcode
1	4	8	11010(0x1a)
1	5	8	11111(0x1f)
5	3	8	1000010(0x42)
6	3	8	0111000(0x38)
3	5	8	111111(0x3f)
3	6	8	110111(0x37)
4	4	8	1000011(0x43)
4	5	8	0111001(0x39)
5	4	8	111010(0x3a)
6	4	8	110010(0x32)
2	7	8	110000(0x30)
2	6	8	111011(0x3b)
4	2	9	111100(0x3c)
4	3	9	111001(0x39)
3	8	9	1001100(0x4c)
4	8	9	1001000(0x48)
1	6	9	110101(0x35)
1	5	9	111110(0x3e)
14	2	12	10100(0x14)
14	3	12	10111(0x17)

B. *Embedding Algorithm*

To embed a message, the first step is to search the Huffman code words that compliance with embedding requirements. For example, assume that a code pair is 0X1a and 0X1f, after

embedding message, the codeword is “HuffCode”, the length of the codeword is “CodeLen”, and the embedded message is “W”. The embedding algorithm can be described as follows.

$$HuffCode' = \begin{cases} 0x1a, if: CodeLen = 8, HuffCode = 0x1a, W = 0; \\ 0x1f, if: CodeLen = 8, HuffCode = 0x1a, W = 1; \\ 0x1a, if: CodeLen = 8, HuffCode = 0x1f, W = 0; \\ 0x1f, if: CodeLen = 8, HuffCode = 0x1f, W = 1. \end{cases}$$

After embedding the message, the original codeword “HuffCode” is replaced by “HuffCode’”, and then generates the MP3 file which contains embedded message at last.

C. *Extracting Algorithm*

To extract the concealed message from the MP3 file, the first step is to search the two code words 0X1a and 0X1f. The extracting algorithm is as follows.

$$W = \begin{cases} 0, if: CodeLen = 8, HuffCode = 0x1a; \\ 1, if: CodeLen = 8, HuffCode = 0x1f. \end{cases}$$

IV. ANALYSIS

A. *Hiding Capacity*

The hiding capacity of the improved algorithm is decided by the number of the specific Huffman codeword in big-value area. That is different MP3 files has different hiding capacity. The following table lists the average hiding capacity comparison results between the original algorithm and the improved algorithm.

TABLE IV. HIDING CAPACITY

MP3 file /kb	Capacity of original algorithm /byte	Capacity of improved algorithm /byte	Increased capacity /%
2226	6381	8959	40.4
3204	7413	10730	44.7
3841	6932	10160	46.6
4064	10774	15173	40.8
4801	10109	14369	42.1

As listed in the above table, the hiding capacity of the improved algorithm is 40%~50% percent more than that of the original algorithm.

B. *Transparence*

The work [2] proposed using sectional average SNR[6] to judge the level of transparence. If we consider the concealed message as Noise, and the original MP3 is Signal, then the human ear is inaudible to the Noise if the SNR is higher than 66dB. According to Gao’s experiments, the average SNR of the MP3 files which contains the concealed message is 67.48>66. It means that human ear can’t distinguish the MP3 containing concealed message from the original MP3.

We can use the same method to calculate the hiding capacity of the improved algorithm. Let N denote the data section length of MP3, x_i denote the sampling data, and $\hat{x}_i$ denote the sampling data after embedding concealed message. Then we have

$$\mathrm{SNR} = 10\log_{10}\frac{\sigma^2}{D}$$

where:

$$\sigma^2 = \frac{1}{\mathrm{N}}\sum_{i=0}^{N-1}(x_i - x)^2$$

$$D = \frac{1}{\mathrm{N}}\sum_{i=0}^{N-1}(x_i - \hat{x}_i)^2$$

$$x = \frac{1}{\mathrm{N}}\sum_{i=0}^{N-1} x_i$$

According above formulations, the average SNR of 10^5 frames containing concealed message is about 66.52dB, which a little smaller than 67.48dB and a little bigger than 66dB. That means, the improved algorithm’s transparence is a little bit worse than the method presented by Gao, but still inaudible to human ear.

V. CONCLUSION

An improved information hiding method is proposed for MP3 files in this paper. It stems from the work of Gao. By using additional new Huffman codeword pairs which meet the transparence requirements, the hiding capacity of the improved algorithm has been greatly improved up to 40~50%. The transparence is preserved.

VI. ACKNOWLEDGEMENT

This paper was supported by Guangxi Key Laboratory of Precision Navigation Technology and Application, Guilin University of Electronic Technology (No. DH201503)

REFERENCES

[1] F. A. P. Petitcolas, R. J. Anderson, M. G. Kuhn, “Information hiding-a survey”, Proceedings of the IEEE, 1999, 87(7):1062 - 1078.

[2] H. Y. Gao, “The MP3 steganography algorithm based on Huffman coding”, ACTA SCIENTIARUM NATURALIUM UNIVERSITATIS SUNYATSENI, 2007, 46(4):32-35.

[3] Y. Yardimci, A. E. Çetin, R. Ansari, “Data hiding in speech using phase coding”, Esca Eurospeech, 1997, 3:1679-1682.

[4] X. Dong, M. F. Bocko, Z. Ignjatovic, “Data hiding via phase manipulation of audio Signals”, International Conference on Acoustics. US, 2007:V - 377-80.

[5] H. song, Q. L. Xing, W. Q. Li and Y. Q. Dai, “MP3Stego Hiding and Detecting Research”, ACTA SCIENTIARUM NATURALIUM UNIVERSITATIS SUNYATSENI, 2004, 43(S2):221-224.

[6] C. H. Zhao, F. C. Li, “digital audio watermark technique: retrospect and prospect”, Journal Harbin Engineering University, 2002, 23(6):57-61.