PAPER

Robust high-capacity audio watermarking based on FFT amplitude modification

M. Fallahpour,† *student member* and D. Megias†

Summary This paper proposes a novel robust audio watermarking algorithm to embed data and extract it in a bit-exact manner based on changing the magnitudes of the FFT spectrum. The key point is selecting a frequency band for embedding based on the comparison between the original and the MP3 compressed/decompressed signal and on a suitable scaling factor. The experimental results show that the method has a very high capacity (about 5 kbps), without significant perceptual distortion (ODG about –0.25) and provides robustness against common audio signal processing such as added noise, filtering and MPEG compression (MP3). Furthermore, the proposed method has a larger capacity (number of embedded bits to number of host bits rate) than recent image data hiding methods.
Key words: *Audio watermarking, Fast Fourier Transform (FFT).*

1. Introduction

The easy transmission and manipulation of digital media has led to a strong demand for watermarking schemes. Since the human auditory system is more sensitive than the visual system, to develop a high-performance audio watermarking technique is a challenging task. Considering the embedding domain, audio watermarking techniques can be classified into time domain and frequency domain methods. Phase modulation [1] and echo hiding [2] are well known methods in the time domain.

In frequency domain watermarking [3-8], after taking one of the usual transforms such as the Discrete/Fast Fourier Transform (DFT/FFT), the Modified Discrete Cosine Transform (MDCT) or the Wavelet Transform (WT) from the signal, the hidden bits are embedded into the resulting transform coefficients. In [6, 8] the FFT domain is selected to embed watermarks for making use of the translation-invariant property of the FFT coefficients to resist small distortions in the time domain. In particular, [8] shows that the FFT domain provides excellent robustness against MP3 compression. In fact, using methods based on transforms provides better perceptual quality and robustness against common attacks at the price of increasing the computational complexity.

In the algorithm suggested in this paper, selecting the frequency band and a scaling factor are the tuning steps. We consider that a safe area for embedding information is the frequency range at which the difference between the FFT magnitudes of the original and the MP3 compressed/decompressed signals is lower than a threshold. Moreover, to strengthen the robustness against attacks, a scaling factor is used. This factor adjusts the value which is added to the FFT magnitudes in the embedding step. For embedding, the FFT magnitudes are first scaled and rounded to the nearest integer. Then, the selected frequency band is scanned and when we meet a magnitude with the value larger than one, it is incremented. If the magnitude is equal to zero it is incremented if the corresponding embedding bit is '1', otherwise the magnitude is not altered. The experimental results show that this method has a very high capacity (about 5 kbps), provides robustness against common signal processing attacks, and entails very low perceptual distortion. Using FFT magnitudes, $\sqrt{\text{real}^2 + \text{imag}^2}$, results in better robustness against attacks compared to using the real or the imaginary parts.

The rest of the paper is organized as follows. In Section 2, the proposed method is presented. In Section 3, the experimental results are shown. Finally, Section 4 summarizes the most relevant conclusions of this research.

2. Proposed method

In this scheme, we use the following method to embed a bit stream (secret bits) into a set of various numbers (FFT coefficients). In the set of numbers, the number which is most frequently encountered than others in the set is selected to embed the hidden bits. For example, in A = {0 2 3 4 7 1 0 2 0 2 1 0}, the zero value is selected. Then, all numbers larger than the selected value (in this case zero) are incremented (shifted) A' = {0 **3 4 5 8 2** 0 **3** 0 **3 2** 0}. Note that, in the shifted A', there is no number with value equal to 1. Finally, in the embedding step, the stream is scanned and the secret bits ("0110") are embedded. When we meet "0" in the stream, if corresponding secret bit is "0", it will not be changed, but if it is equal to "1" it should be incremented. The marked set of values is then A^* = {**0** 3 4 5 8 2 **1** 3 **1** 3 2 **0**}. At the detector side, the secret bits are extracted and, then, all values larger than the selected value are decremented

As mentioned above, we have chosen the FFT domain to embed the hidden data in order to exploit the translation-invariant property of the FFT transform such that small distortions in the time domain can be resisted. Compared to other schemes, such as quantization or

odd/even modulation, keeping the relationship of FFT coefficient pairs is a more realistic scheme under several distortions.

2.1 Tuning

This method is based on using near zero values of FFT magnitudes in a selected frequency band. The method needs integer values for the FFT magnitude in the embedding step. Thus, the magnitudes in the selected band are multiplied by a scaling factor *s* and then rounded to the nearest integer. This scaling and rounding process generates a great deal of zero values in the integer magnitudes. The frequency band and the scaling value (*s*) are the two parameters of this method which adjust capacity, perceptual distortion and robustness.

To select the frequency band, the considered points are as described below:

1. The selected band should have as many zeros as possible after scaling (multiplying by *s*) and rounding. The number of zeros identifies the capacity.
2. In the selected band, the difference between the magnitudes of the FFT coefficients of the original and the MP3 compressed/decompressed signals should be small.

To select the scaling factor *s*, the following points should be considered:

1. By increasing the scaling factor *s*, the error of the scaling and rounding step decreases and results in better perceptual distortion and lower capacity.
2. After the embedding steps, the magnitudes with value zero at which "1" is embedded are changed to 1/*s*. To obtain the secret bits after attacks, 1/*s* should be larger than difference between the original magnitudes and the attacked magnitudes.

Fig. 1 shows the flowchart for the selection of the tuning parameters. We select the frequency band from 15 kHz as low frequency and the cut-off frequency of MP3 as the high frequency. For example, the cut-off frequency of MP3-128 is around 17 kHz. If there are not enough magnitudes with zero value (after scaling and rounding), the selected frequency band should be expanded by decreasing the low frequency band. Since 8 kHz is start of high frequencies it is selected as the limitation. In the flowchart, the required capacity is denoted by *cap*, N_z is the number of zeros in selected frequency band and L_f is low frequency of the selected band. As the flowchart shows, decreasing *s* increases 1/*s* which is used for detecting secret information. In the initialization, the parameters *s* is equal to 10.0. Most FFT magnitudes in the selected frequency band are between 0 and 10, hence with *s* in the interval [0.1, 10.0] with 0.1 steps we do not miss significant magnitudes and increase the number of zeros in the frequency band simultaneously.

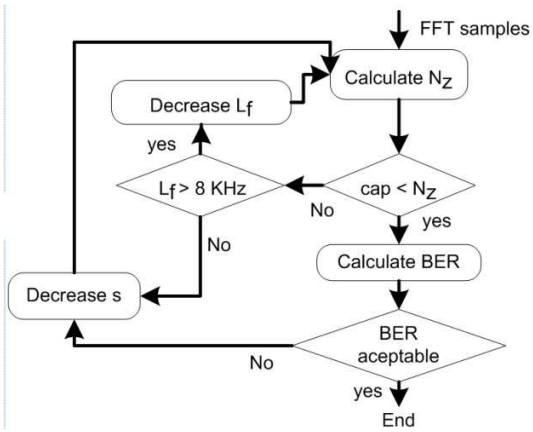

Fig. 1. Flowchart of tuning steps

The watermarking scheme presented here is positional. This means that the detector must be synchronized in order to recover the embedded bits correctly. In a real application, the cover signal would be divided into several blocks of a few seconds and it is essential that the detector can determine the position (the beginning sample) of each of these blocks. One of the most practical solutions to solve this problem is to use synchronization marks such that the detector can determine the beginning of each block. Several synchronization strategies have been described in the literature (for example [10, 11]) and any of them can be used together with the method described here in order to produce a practical self-synchronizing solution. A self-synchronized version of the proposed scheme (using the synchronization approach described in [10]) has been implemented and the results are shown in Section 3.

2.2 Embedding algorithm

The embedding steps are as follows:

1. Calculate the FFT of the audio signal. We can use the whole file (for short clips, *e.g.* with less than one minute) or blocks of a given length (*e.g.* 10 seconds) for longer files.
2. Use the *s* selected in the tuning step as a parameter to convert the FFT magnitudes in the selected frequency band to integer values (multiplying them by *s* and then rounding).
3. Scan all the integer FFT magnitudes in the selected band. If a magnitude is larger than zero, then increase it by one. After this step we have no magnitude with the value one.
4. Scan again all integer FFT magnitudes in selected band. When a zero magnitude is found, if the corresponding embedded bit is "1" add one to the magnitude. Otherwise, the magnitude is not changed. After this step all magnitudes with value zero or one represent an embedded bit.

5. Embed *s* and the frequency band limits in their reserved positions as described in the end of this section. This step must take into account security concerns, as detailed below.
6. The marked (FFT) signal is achieved by dividing all the magnitudes by *s*.
7. In the previous embedding steps, the FFT phases are not altered. The marked audio signal in the time domain is obtained by applying the inverse FFT with the new magnitudes and the original FFT phases.

2.3 Extraction algorithm

The watermark extraction is performed by using the FFT transform and the tuning parameters. Since the host audio signal is not required in the detection process, the detector is blind. The detection process can be summarized into the following steps:

1. Calculate the FFT of the marked audio signal.
2. Extract the *s* and the frequency band from special positions (this step requires the use of a secret key).
3. To achieve the scaled FFT magnitudes in selected frequency band, multiply them by *s*.
4. Scan all the scaled FFT magnitudes in the selected band. If a magnitude with value in the interval [0, 1/2) is found, then the corresponding embedded bit is equal to "0" and the restored magnitude equals to zero. If the magnitude value is in the interval [1/2, 3/2), then the corresponding embedded bit is equal to "1" and the restored magnitude equals to zero.
5. Scan all the scaled FFT magnitudes in the selected band. For each magnitude value in the interval [k+1/2, k+3/2), the restored magnitude equals to k (for $k > 1$).
6. The restored magnitudes are achieved by dividing them by *s*.
7. Finally, use the IFFT to achieve the restored audio signal.

For example, assume that the magnitudes at the selected frequency band are (0.9 0.4 0.2 0.1 1.4 0.15), $s = 2$ and the secret bit stream is "010". Table 1 summarizes all steps of embedding and extracting.

Table 1. embedding and extracting steps

steps		FFT magnitudes					
embedding	calculate FFT	0.9	0.4	0.2	0.1	1.4	0.15
	scaling	1.8	0.8	0.4	0.2	2.8	0.3
	rounding	2	1	0	0	3	0
	shifting	3	2	0	0	4	0
	embedding	3	2	**0**	**1**	4	**0**
	Dividing by *s*	1.5	1	0	0.5	2	0
extracting	Scaling	3	2	0	1	4	0
	Detecting	3	2	0	0	4	0
	Shifting back	2	1	0	0	3	0
	Dividing by *s*	1	0.5	0	0	1.5	0

It is worth pointing out that the tuning parameters (*s* and the frequency band) should be used in receiver to detect the secret information. In the embedding steps, a few special spaces are kept for saving tuning parameters. The FFT magnitudes in special frequencies such as 12, 13, 14, 15 and 16 kHz, which are reserved for scaling factor, are changed by the value of *s*. Consequently, in the receiver *s* will be available. Similarly, we use a 16-bit space available for embedding secret information which begins after the first FFT coefficient with a zero magnitude from a selected frequency (*e.g.* 15 kHz) to embed the values of the low and high frequencies of the selected frequency band. For example, if we embed at the frequency band from 12.3 to 16.7 kHz, we multiply them by ten and change them to binary values

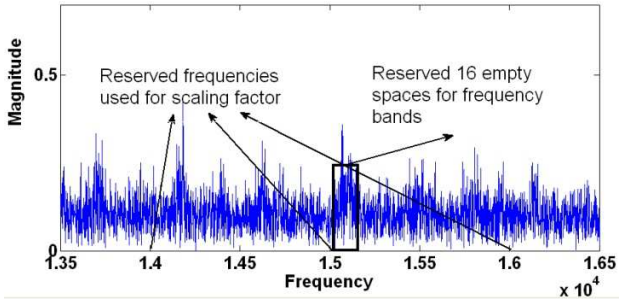

Fig. 2. Reserved positions for *s* and frequency bands

$(01111011)_2 = 123$ and $(10100111)_2 = 167$. After that, these binary streams are embedded in the free space found next to the selected frequency (15 kHz). The first bit of 123, is embedded in first available space after the selected position, and so on. Fig. 2. shows an example of the reserved positions for *s* and the frequency band.

The security of this method requires that the frequency band and the scaling factor *s* are not known by an attacker. Note that if an attacker does not know the scaling factor *s*, it will not be possible for him or her to analyze the values of the FFT magnitudes to determine the position of the embedded bits. For example, the rounded FFT magnitudes after scaling by $s = 0.2$ or $s = 0.4$ are completely different. If the attacker does not know the frequency band either, it becomes even more difficult for him or her to try to determine the interval of the FFT spectrum which carries the secret information.

In order to keep both the frequency band and the scaling parameter secret, there are two possibilities. The first one would be to consider both *s* and the frequency band as part of the secret key. In that case, the values of these parameters should **not** be embedded in the marked audio sequence and they should be transmitted as side information over a secured channel. At the receiver side, this information would be given to the extractor in order to recover the hidden data.

A second possibility introduces security even if the frequency band and the scaling parameter are embedded as suggested above. The following security measures are required:

1. The values of *s* and the frequency band should not be embedded as clear text. The bits which form the values must be scrambled using a Pseudo Random Binary Sequence (PRBS) generated through a secret key (seed) and the embedded values would be the result of an XOR sum of the bits of the original parameters (*s* and the frequency band limits) and the bits of the PRBS. The secret key would be also needed at the detector side in order to unscramble the values of *s* and the frequency band.
2. The FFT positions for embedding the values of *s* and the frequency bands should **not** be fixed. Instead of using fixed frequencies (like 12, 13, 14, 15 or 16 kHz) the position must be also generated with a Pseudo Random Number Generator (PRNG) in some interval (*e.g.* [12, 16] kHz) using a secret key (seed) which is required at both the sender and the receiver. This procedure makes it impossible for an attacker to destroy the values of *s* and the frequency band, since he or she cannot know the position of these data in the FFT spectrum. In order to destroy them, it would be required to disturb a wide interval of the spectrum and, thus, the quality of the attacked signal would also be damaged and would become unusable.

To increase security even further, a PRNG can also be used to change the secret bit stream to a scrambled stream. For example, the embedded bitstream can be constructed as the XOR sum of the real watermark and a PRBS. The seed of the PRNG would be required as a secret key both at the embedder and the detector. The usage of PRNG to increase the security of watermarking schemes is discussed in the literature (for example in [8]).

3. Experimental results

To evaluate the performance of the proposed method, male speech in English in *spme50_1*, violoncello in *vioo10_2*, trumpet in *trpt21_2*, soprano in *sopr44_1*, quartet in *quar48_1* have been selected from the Sound Quality Assessment Material (SQAM) [12]. Also, to consider the applicability of the scheme in a real scenario, the song "Thousand Yard Stare" (3:57) included in the album *Rust* by No, Really [13] has been selected. All audio clips are sampled at 44.1 kHz with 16 bits per sample and two channels. The experiments have been performed for each channel of the audio signals separately.

The Objective Difference Grade (ODG) is used to evaluate the transparency of the proposed algorithm. The ODG is one of the output values of the ITU-R BS.1387 PEAQ [14] standard, where ODG = 0 means no degradation and ODG = –4 means a very annoying distortion. Additionally, the OPERA software [15] based on the ITU-R BS.1387 has been used to compute this objective measure of quality.

Table 2 illustrates the tuning parameters, perceptual distortion and payload for six mono signals for BER equal zero under the MP3-128 attack. The tuning parameters have been chosen manually just to test the system for different tuning settings (*i.e.* we have not followed the flowchart depicted in Fig. 1). Table 3 shows the effect of various attacks, provided by the Stirmark Benchmark for Audio (SMBA) v1.0 [16], on ODG and BER for the five selected SQAM signals. *E.g.* the row "Amplify" shows that the changes in volume of the watermarked signal has BER equal to zero when the alteration of volume is within the interval [0.8, 1.45]. As described in Section 2, the frequency band and the scaling factor *s* are the two parameters of the method. These parameters were selected for each signal, then the embedding method has been applied, the Stirmark Benchmark for Audio (SMBA) software has been used to attack the marked files and, finally, the detection method has been performed for the attacked files. The ODG in Table 3 is calculated between the marked and the attacked-marked files. The parameters of the attacks are defined based on SMBA web site [16]. For example, in AddBrumm, 1-7000 shows the strength and 0-14000 shows the frequency. This row illustrates that any value in the range 1-7000 for the strength and 1-14000 for the frequency could be used without any change in BER. In fact, this table provides the worst and best results for the five test signals based on BER and, in the case with the same BER, based on the limitation of the parameters. The only attack in Table 3 which removes the hidden data is *FFT_Stat1*, which is able to remove the secret data for one of the SQAM files (BER = 27%). Note, however, that the ODG of this attack is extremely low (–4). This means that the attack does not only removes the hidden data, but also destroys the perceptual quality of the host signal.

Table 2: Parameters and results of 5 mono signals (BER=0 under MP3-128)

Audio File	***s***	***Frequency band (kHz)***	***ODG of marked signal***	***Payload (bps)***
spme50_1	0.3	15.5-16.9	– 0.18	1478
vioo10_2	0.5	8-16.9	– 0.34	8719
trpt21_2	3.5	8.5-16.9	– 0.39	7934
sopr44_1	0.35	10-16.9	– 0.30	7006
quar48_1	0.7	13.5-16.9	– 0.34	3177
Thousand Yard Stare	[0.1, 0.6]	15-16.9	– 0.78	1849

The SQAM files are short clips (30 seconds or less), and it is not necessary to use synchronization marks with them, since the whole file can be used in the embedding and extracting processes with short enough CPU time.

Table 3: Robustness test results for five SQAM selected files

Attack name	***State***	***ODG of attacked file***	***BER %***	***parameters***
No attack	–	0	0	–
MP3	–	–0.1	0	≥ 128
AddBrumm	best	–1.75	0	1-7000,1-14000
	worst	–3.58	0.5	1-7000,1-750
AddDynNoise	best	– 0.71	0	1-4
	worst	– 0.32	0.5	1
ADDFFTNoise	best	–1.47	0	4096,1-8800
	worst	–1.48	0	2,1-150
Addnoise	best	–1.46	0	1-115
	worst	–1.94	0	1-21
AddSinus	best	–1.25	0	No Limitation
	worst	–3.33	0	
Amplify	–	– 0.1	0	80-145
FFT_Invert	best	–1.61	0	No limitation
	worst	–3.66	0	
FFT_RealReverse	best	–3.56	0	No limitation
	worst	–3.96	0	
FFT_Stat1	best	–3.98	0	1024
	worst	-3.96	27	
Invert	best	–1.63	0	–
	worst	–3.63	0	
RC_LowPass	best	– 0.1	0	Selected frequency band in passing area
	worst	– 0.25	0	
RC_HighPass	best	–3.29	0	Selected frequency band in passing area
	worst	–3.59	0	

Table 4: Robustness test results for “Thousand Yard Stare”

Attack name	***ODG of attacked file***	***BER %***	***SYNC error***	***parameters***
No attack	0	0	0	–
MP3	–0.1	0	0/23	≥ 128
AddBrumm	–3.6	1	0/23	1-7500,1-5000
AddDynNoise	– 2.2	0	0/23	1-2
ADDFFTNoise	–1.1	0	0/23	4096,1-8800
Addnoise	–0.5	0	0/23	1-60
AddSinus	–3.3	0	0/23	No Limitation
Amplify	–0.1	0	0/23	80-145
FFT_Invert	–3.7	0	0/23	No limitation
FFT_RealReverse	–3.7	3	1/23	No limitation
FFT_Stat1	–3.9	36	3/23	1024
Invert	–3.7	0	0/23	–
RC_LowPass	–0.2	0	0/23	Selected frequency band in passing area
RC_HighPass	–3.7	0	0/23	Selected frequency band in passing area

In order to reduce computation time and memory usage, the near 4-minute long “Thousand Yard Stare” song was divided into 23 clips of 10 seconds each. Then, the synchronization method described in [10] and the embedding algorithm described in this paper was applied for each clip separately. For this song, 16 synchronization bits, “1 0 1 1 0 0 1 1 1 1 0 0 0 0 1 0” with a quantization factor equal to 0.125, were embedded in the first 80 samples of each clip and then the information watermark was embedded in the remaining samples of the 10-second segment. Finally all these 10-second clips were joined together to generate the marked signal. We have used different scaling factors in the range [0.1, 0.6] for different clips. The payload and transparency results given in Table 2 for this file consider the effect of both the synchronization codes and the information watermarks. Table 4 shows the effect of various attacks on ODG and BER for the marked “Thousand Yard Stare” signal. The whole file is attacked, then it is scanned in time domain to find the synchronization codes and, finally, the secret information of each clip is extracted. The “SYNC error” column shows the detection error of synchronization code after attacks, which shows that the synchronization algorithm [10] is robust against attacks. Figure 3 visualizes the test results. This plot shows how the capacity and perceptual distortion are changed with different tuning parameters. The BER for all test results under the MP3-128 attack on this plot is equal to zero.

Only a few attacks, such as low pass filter ─which only leaves low frequencies unaltered─ with a cut-off frequency less than 6 kHz damage the hidden data. However, the ODG of this attack is extremely low (about –3.5, *i.e.* very annoying). This means that the attack does not only remove the hidden data, but also destroys the perceptual quality of the host signal. On the other hand, if the cut-off frequency is larger than 8 kHz the BER is about zero and the ODG of attack is in the acceptable range.

A very relevant issue in audio watermarking is computation time. As FFT is a fast transform, this method is very useful for real-time applications. Table 5 illustrates the embedding and extracting times and compares them with the computation time of FFT and the Daubechies wavelet transform. The results for the song “Thousand Yard Stare” are the average of all the 10-second clips. It is worth mentioning that these computation times have been obtained with an Intel (R) core (TM) 2 Duo 2.2 GHz CPU and 2 GB of RAM memory. It can be noticed that the extracting time is one order of magnitude smaller than the file playing time. Thus, it is perfectly possible to recover the embedded data in a real-time scenario.

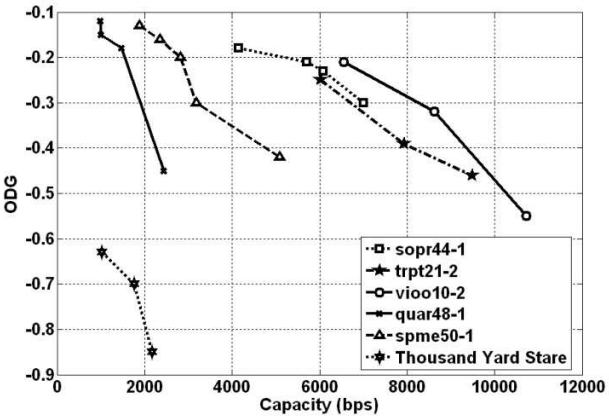

Fig 3. Comparison between payload (bps) and Transparency (ODG) for BER=0 under MP3 attack (bitrate 128 kbps)

Table 5: Computation time

Audio File	*Time (sec)*	*FFT time*	*Db2 time*	*Embedding time (sec)*	*Extracting time (sec)*
spme50_1	18	0.46	0.50	1.78	1.67
vioo10_2	30	0.57	0.47	4.30	4.01
trpt21_2	17.8	0.58	0.48	1.89	1.76
sopr44_1	23.5	0.60	0.53	3.65	3.30
quar48_1	23	0.59	0.48	2.59	2.29
Thousand Yard Stare	23×10	0.17	0.15	0.69	0.59

The method proposed in this paper has been compared with several recent audio watermarking strategies. It must be taken into account that none of the works in the reviewed literature produce capacity of the order of 5 kbps, such as the proposed scheme. All the audio data hiding schemes which produce very high capacity are fragile against signal processing attacks. Because of this, it is not possible to establish a comparison of the proposed scheme with other audio watermarking schemes which are similar to it as capacity is concerned. Hence, we have chosen a few recent and relevant audio watermarking schemes in the literature. In Table 6, we compare the performance of the proposed watermarking algorithm and several recent audio watermarking strategies robust against the MP3 attack. The results are given for SQAM files. [4-6, 9] use SQAM [12] files for evaluating their suggested schemes. All the schemes in this table are robust against MP3 compression with a 128 kbps bitrate. Under this attack, the BER is equal to zero for all the compared schemes.

[7] Evaluates distortion by mean opinion score (MOS), which is a subjective measurement, and achieves transparency between imperceptible and perceptible but not annoying, MOS = 4.7. [4, 5, 9] have a low capacity but are robust against common attacks.

Capacity, robustness and transparency are the three main properties of an audio watermarking scheme. Considering a trade-off between these properties is necessary. *E.g.* [4] proposed a very robust, low capacity and high distortion scheme. However [7] and the proposed scheme introduce high capacity and low distortion technique but they are not as robust as the low-capacity method described in [4].

This comparison shows the superiority in both capacity and imperceptibility of the suggested method with respect to other schemes in the literature. This is particularly relevant, since the proposed scheme is able of embedding much more information and, at the same time, introduces less distortion in the marked file.

Table 6: Comparison of different watermarking algorithms

Algorithm	Capacity (bps)	Imperceptibility in SNR (dB)	Imperceptibility (ODG)
[4]	2	42.8 to 44.4	–1.66 to –1.88
[9]	2.3	Not reported	Not reported
[5]	4.3	29.5	Not reported
[6]	Coded image with 9×35 bits	24.3 to 34.4	–0.36 to –0.72
[7]	689	Not reported	Not reported
proposed	1478 to 8719	35.2 to 44.6	–0.18 to –0.78

In the last few years, very good results in image data hiding have been published. Ni *et al*. [17] proposed a high capacity data hiding with very low distortion. For general test images such as Lena and Baboon they embedded about 5 kbit in the whole image with PSNR above 40 dB, *i.e.* the embedding rate for a 512×512×8 image is 0.0024 bits of information per each image bit. The proposed method in this paper embeds about 5 kbit in a second. It means 5 kbit in 44100×16 bits that equals to 0.0071 per audio bit. If we consider the compression rate of MP3-128 (about 12:1), since this method completely robust against MP3-128, the embedding rate for each bit of audio sample equals **0.085**, that is 35 times more than the information bit rate achieved with the image method of Ni *et al*. Some other image data-hiding schemes have been presented [18] increasing the payload up to 0.02 bits per image bit. Even in this case, the suggested audio scheme presented here achieves more than four times that capacity.

4. Conclusion

In this paper, we describe a high-capacity watermarking algorithm for digital audio which is robust against common audio signal processing. A scaling factor (s) and the selected frequency band to embed the hidden information are the two parameters of this method which regulate the capacity, the perceptual distortion and the robustness of the scheme. Furthermore, the suggested scheme is blind, since it does not need the original signal for extracting the hidden bits. The experimental results show that this scheme has a high capacity (about 5 kbps) without significant perceptual distortion and provides robustness against common signal processing attacks such

as noise, filtering or MPEG compression (MP3). Besides, the proposed method achieves a higher embedded bit to host bit rate than recent image data hiding methods. In addition, the CPU time required by the proposed scheme is short enough to use the scheme in real-time applications.

References

[1] N. Lie, L. C. Chang, "Multiple Watermarks for Stereo Audio Signals Using Phase-Modulation Techniques", *IEEE Trans. Signal Processing*, Vol. 53, No. 2, pp. 806–815, Feb. 2005.

[2] H. J. Kim, Y. H. Choi, " A novel echo hiding scheme with backward and forward kernels", *IEEE Trans. Circuit and Systems*, Vol. 13, pp. 885-889, Aug. 2003.

[3] S. Esmaili, S. Krishnan, K. Raahemifar, "A novel spread spectrum audio watermarking scheme based on time - frequency characteristics". *IEEE Conf. Electrical and Computer Engineering*, Vol. 3, pp. 1963-1966, May 2003.

[4] S. Xiang, H.J. Kim, J. Huang, "Audio watermarking robust against time-scale modification and MP3 compression," *Signal Processing*, Vol.88 n.10, pp.2372-2387, October, 2008.

[5] M. Mansour and A. Tewfik, "Data embedding in audio using time-scale modification," *IEEE Trans. Speech Audio Process.*, Vol. 13, no. 3, pp. 432–440, 2005.

[6] Y.Q. Lin, W.H. Abdulla, "Multiple Scrambling and Adaptive Synchronization for Audio Watermarking", *IWDW*, LNCS 3304, Springer-Verlag, pp. 456-469, 2007.

[7] J. J. Garcia-Hernandez, M. Nakano-Miyatake and H. Perez-Meana, "Data hiding in audio signal using Rational Dither Modulation", *IEICE Electron. Express*, Vol. 5, No. 7, pp.217-222, 2008.

[8] D. Megías, J. Herrera-Joancomartí, and J. Minguillón. "Total disclosure of the embedding and detection algorithms for a secure digital watermarking scheme for audio". *Proceedings of the Seventh International Conference on Information and Communication Security*, pp. 427-440, Beijing, China, December 2005.

[9] W. Li, X. Xue, "Content based localized robust audio watermarking robust against time scale modification" *IEEE Trans. Multimedia*, Vol. 8, No. 1, pp. 60-69, Feb. 2006.

[10] X.-Y. Wang and H. Zhao. "A novel synchronization invariant audio watermarking scheme based on DWT and DCT". *IEEE Trans. on Signal Processing*, Vol. 54(12), pp. 4835–4840, December 2006.

[11] Y. Lin and W. Abdulla. "A secure and robust audio watermarking scheme using multiple scrambling and adaptive synchronization". In Proceedings of the *6th International Conference on Information, Communications & Signal Processing*, pp. 1–5, 2007.

[12] SQAM Sound Quality Assessment Material, http://andrew.csie.ncyu.edu.tw/html/mpeg4/sound.media.mit.edu/mpeg4/audio/sqam/index.html

[13]No, Really, "Rust". http://www.jamendo.com/en/album/7365.

[14] T. Thiede, W. C. Treurniet, R. Bitto, C. Schmidmer, T. Sporer, J. G. Beerens, C. Colomes, M. Keyhl, G. Stoll, K. Brandenburg, and B. Feiten, "PEAQ - The ITU Standard for Objective Measurement of Perceived Audio Quality," *Journal of the AES*, vol. 48(1/2), pp. 3–29, 2000.

[15] OPTICOM OPERA software site. http://www.opticom.de/products/opera.html

[16] Stirmark Benchmark for Audio. http://wwwiti.cs.uni-magdeburg.de/~alang/smba.php.

[17] Ni Zhicheng, Y.Q. Shi, N.Ansari, and W. Su, "Reversible data hiding," *IEEE Trans. on Circuits and Systems for Video technology*, 16(3):354–362, March 2006.

[18] D. M. Thodi and J. J. Rodriguez, Expansion embedding techniques for reversible watermarking, *IEEE Trans. on Image Processing*, Vol. 16, no. 3, pp.721−730, 2007.