Content-Adaptive Musical Audio Watermarking based on the Music Theory of Chords

Arjun Yogeswaran, Pierre Payeur, and Jiying Zhao
School of Information Technology and Engineering
University of Ottawa
Ottawa, ON, Canada
[ayoge099, ppayeur, jzhao]@uottawa.ca

Abstract—**This paper proposes a novel content-adaptive music watermarking technique which uses the principles of music theory to enhance the capacity and robustness of watermark embedding. Using the musical concepts of key and chords, certain notes, which are musically coherent with the work, can be added or removed without impacting the listeners experience. The notes serve as the carrier of the watermarked bit. Since these notes are still within the human hearing range, with high enough strength, they will still remain intact after various audio compressions or distortions. The scheme does not require the original work to extract the watermark and serves as a framework to involve musical theory in optimizing watermark embedding algorithms.**

Keywords-audio watermarking; music watermarking; music theory; chords; content-adaptive.

I. INTRODUCTION

Digital watermarking has become a widespread technology, immersing itself into many different types of digital media including audio and video. Digital watermarking is the concept of altering a work to embed a message. Invisible or inaudible watermarking is the concept of imperceptibly altering a work to embed a hidden message.

Watermarking has become very useful in the digital medium for many reasons, but most dominantly for protecting intellectual property rights, especially for the digital audio medium and the distribution of music. Many digital audio watermarking techniques make use of the analysis of the human auditory system (HAS) to hide the watermark [1,2,3]. Many techniques also use adaptive watermarking to modify the embedding method to more effectively embed the watermark based on the characteristics of the audio [4].

This paper proposes a new content-adaptive algorithm, designed specifically for music, which exploits musical theory to more effectively embed robust and imperceptible watermarks. The proposed method uses frequency analysis to determine the chords that are present in the music, and then proceeds to add or remove notes that belong in that chord to carry the watermark. By altering notes that belong in the chord, the correctness of the audio is not altered, and the listener should not experience any discomfort. Though there may be a slightly noticeable difference between the original and the watermarked work, the quality of the watermarked work will remain high and the listener should not notice that it has been altered. It is robust since compression and other distortions would not likely remove the embedded note due to its perceptibility from a human auditory system (HAS) standpoint. For detection, this technique does not require the original work, and uses side information to extract the watermark.

II. AUDIO WATERMARKING

There are many different techniques for digital audio watermarking which are generally classified into time-domain methods and frequency-domain methods.

Time-domain methods generally embed and extract the watermark in the time domain. Two significant techniques that fall under time-domain methods are least significant bit embedding and echo hiding. Least significant bit embedding alters the least significant bits of each audio sample to embed the watermark [5,6]. On the extraction side, the least significant bits, containing the watermark, are read. Though it may be imperceptible, it is not very robust since any simple distortion can alter the least significant bits, and the watermark may be lost. Echo hiding is a method which is robust and remains quite imperceptible. The principle of the technique is to add a slight echo to frames of the audio [7]. This echo is of low amplitude and fits the audio so it is not likely to be noticed, however it can be easily identified by attackers [8].

The concept of frequency-domain methods is that the embedding and extraction of the watermark is done in the frequency domain. Phase coding embeds bits in the frequency domain by altering the relative phase of certain frequency components [9]. Slight variations in phase have proven to be imperceptible to the listener. However, compression algorithms may alter the phase and destroy the embedded data. There are also frequency-domain techniques that use very high frequencies and very low frequencies to carry binary data. This is useful for imperceptibility since the human ear is less sensitive to those frequencies, however compression algorithms will definitely degrade them since audio compression is usually based on HAS.

III. MUSIC THEORY

Musical theory is a mathematical language, defining the rules in creating music, and aiding in the analysis of music.

978-1-4244-4218-8/09/$25.00 ©2009 IEEE

Music theory can also be used to predict what components of music can be altered, while still being imperceptible to the listener. This can be exploited to provide more robust embedding of watermarks, as well as increasing the capacity.

A. Notes

A musical note is one of the most basic elements of music. A note can be generated by an instrument and consists of certain characteristics. The most important characteristics for our purposes are pitch, duration, and timbre.

The *pitch* of a note is the frequency of its vibration, and determines how high or low the note is. It also identifies what note name it is based on the frequency table shown below.

Sub-band No		01	02	03	04	05	06	07	08
Octave scale		~ B1	C2 ~ B2	C3 ~ B3	C4 ~ B4	C5 ~ B5	C6 ~ B6	C7 ~ B7	C8 ~ B8	All the higher octaves in the (8192 ~ 22050) Hz frequency range
Freq. range (Hz)		0 ~ 64	64~128	128~256	256~512	512~1024	1024~2048	2048~4096	4096~8192	
12 Pitch class notes	C		65.406	130.813	261.626	523.251	1046.502	2093.004	4186.008	
	C#		69.296	138.591	277.183	554.365	1108.730	2217.460	4434.920	
	D		73.416	146.832	293.665	587.330	1174.659	2349.318	4698.636	
	D#		77.782	155.563	311.127	622.254	1244.508	2489.016	4978.032	
	E		82.407	164.814	329.628	659.255	1318.510	2637.02	5274.04	
	F		87.307	174.614	349.228	698.456	1396.913	2793.826	5587.652	
	F#		92.499	184.997	369.994	739.989	1479.978	2959.956	5919.912	
	G		97.999	195.998	391.995	783.991	1567.982	3135.964	6271.928	
	G#		103.826	207.652	415.305	830.609	1661.219	3322.438	6644.876	
	A		110.000	220.000	440.00	880.000	1760.000	3520.000	7040.000	
	A#		116.541	233.082	466.164	932.328	1864.655	3729.310	7458.62	
	B		123.471	246.942	493.883	987.767	1975.533	3951.066	7902.132	

Figure 1. Frequencies of notes and frequency ranges of octaves [10]

One can use frequency analysis to identify what notes are being played and thus which chords are being played, leading to which frequencies can be altered imperceptibly.

The *duration* of a note is the length of time that note is held. This is important for frame segmentation used in the synchronization of the watermark embedder and detector.

The *timbre* of a note encompasses the characteristics of a note that are not defined by the pitch or duration. Different instruments have different timbre because of the way they generate the note. Timbre can be analyzed by examining extra frequencies generated by the instrument that do not belong to the pitch. Also, the type of wave that is generated is not always a perfect sine wave. This concept is important when trying to seamlessly add or remove notes into a work.

B. Octave

An octave consists of all the notes between one note letter and another note letter, as shown in Figure 2. In western music, an octave consists of twelve steps between the two notes bordering the octave.

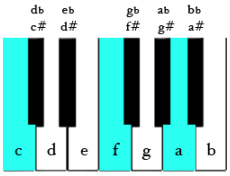

Figure 2. Octave on a piano starting at C

Mathematically, it is all the pitches that occur between one frequency and the doubling of that frequency.

C. Scale

Combinations of notes can be arranged into different scales, and these scales define which key the musical piece belongs to. Some notes fit the key and some do not, which means some notes sound correct in the piece while others sound incorrect. This will be exploited when deciding which notes to alter, while keeping the music sounding correct.

This work concentrates on western music theory, where a scale is 7 different notes that are acceptable at any octave. The scale begins with the first note in the scale, which is known as the tonic, and will be the *root note* of the chord. The 3rd note is known as the *median note*, and is important for generating a chord also. The *dominant note*, which is the 5th note in the scale, is also important in generating a chord.

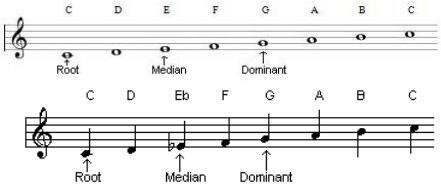

Figure 3. Top: C major scale. bottom: C minor scale

D. Chords

A chord consists of three or more notes being played in harmony. For the purposes of this work, tertiary chords will be used, since they are common in most music. Tertiary chords require that the harmonic interval between the notes in a chord is a third, so they consist of a root note, a median note, and a dominant note. Chords are important, since the dominant note is used to embed the watermark bit.

IV. PROPOSED METHOD

The proposed method is a novel concept which uses detailed musical theory knowledge to alter the work and embed a watermark in a unique fashion.

Other content-adaptive methods for embedding watermarks in music involve the analysis of certain music characteristics; however, the involvement of musical theory is kept to a minimum. In an innovative approach recently proposed by Xu *et al.* [10], synchronization of the audio watermarking scheme was achieved by measuring the shortest note length in the piece, and using the duration of each note as a frame of audio. The watermark was detected, by using the start of notes as synchronization, and the phase shift of a certain frequency indicated the embedding of bits in that respective frame. Though basic music theory is used to segment the audio into frames, no more musical theory has been exploited to achieve increased performance.

In the method proposed here, fixed-frame synchronization is used, where the audio is segmented into fixed-length frames with each frame holding a bit of information. Performance can be improved by using the note-length frame synchronization proposed in [10], however for this research, fixed-frame synchronization was sufficient.

A novel direction is proposed for embedding and detecting watermarks by using more in-depth musical theory. Using the musical concepts of chords, we can determine that there are certain notes that can be added or removed without perceptibly impacting the quality of the music, or altering the "correctness" from a musical theory standpoint.

This method analyzes the frame in the frequency domain, and finds the most likely chord being played. It adds the

dominant note of the chord to embed a 1, and removes the dominant note to embed a 0. The method also does not discriminate whether works contain a watermark or do not and assumes that all works passed to the detector contain a watermark. This will be discussed in the *Limitations* section.

To either embed or detect a watermark in the music, the work undergoes a 3-step process, with the first two steps being the same for both. The first step is to divide the work into fixed-length frames. The second step involves analyzing the frequencies in the frame, to find the most likely root chord. The final step, to embed, involves the embedding of the bit, by inserting a note into the chord, while the final step, to detect, analyzes if the appropriate note carrying the bit exists.

A. Fixed-Length Frame Segmentation

The audio is divided into fixed-length frames, to allow the embedding of multiple bits throughout the entire work. Each frame is currently defined to be approximately 417 milliseconds.

The number of samples per frame was arrived at experimentally, based on an analysis of several works. A short sample proved to give some inconsistent results when converting the frame into the frequency domain while a longer sample caused too many frames to contain a change in chords.

The algorithm works best when each frame contains only one chord so the inserted note can be musically coherent with the chord in the frame. For this reason, fixed-length segmentation will not provide the best results. This will be explored further in the *Limitations* section.

B. Chord Analysis

The second step in the process is to analyze the most likely root chord present in the frame. The frame is transformed from the spatial domain into the frequency domain using a Fourier transform. The results are sorted by amplitude, and the frequencies with the highest 50 amplitudes are assessed.

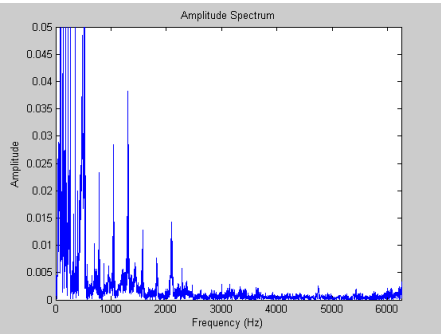

Figure 4. Frequency-domain plot of audio frame

Since each note corresponds to a frequency, the strongest 50 frequencies are converted into their respective musical notes, according to Figure 1. To determine the chord, only the root notes and the dominant notes are considered. Each note is tested as a possible root note. For each potential root note, the amplitudes of all the corresponding root note frequencies and dominant note frequencies appearing in the top 50 frequencies are summed. This is done until all the notes in the top 50 frequencies have been exhausted. Whichever amplitude sum is greatest corresponds to the most likely root note for the current chord.

C. Note Insertion

The final step when embedding the watermark is the embedding of the bit itself, by ensuring the chord contains a specific note, or ensuring that it does not. The algorithm uses the dominant note of the identified chord to be the carrier of the bit. Since this note is important, and is acoustically perceptible, it is unlikely to be lost in compression or other types of analog distortion, but may be noticed by the listener. This will be discussed further in following sections.

To embed a bit, the dominant note of the chord is altered in the 1st octave. Since the chord can be spread out over many octaves, the dominant note of the chord is altered specifically in the 1st octave since it is perceptible, yet low enough that it is less perceptible than the higher octaves. Also, the low octaves may be more robust since they can survive downsampling. To improve imperceptibility, a lower frequency can be used as the bit carrier, at the expense of robustness. It is also important that the frequencies that define the root chord are not significantly altered, so that the root chord can still be determined. If the root chord is altered, the detection of the bit will be incorrect since the detector will use the wrong frequency to determine the root chord, and thus use the wrong dominant note to determine the embedded bit. This will be discussed further below.

The process begins by assessing if the dominant note of the chord in the 1st octave is present. The assessment is done by getting the Fourier transform of the frame, and looking for the dominant note frequency within the strongest N frequencies. N can differ depending on whether a 1 is being embedded or a 0 is being embedded. This is for robustness purposes, since some frequencies may change amplitude during compression or other distortions. By assigning a small value, N_1, to N when embedding a 1, the algorithm will ensure that the dominant note is one of the strongest frequencies. By assigning a larger value, N_0, to N when embedding a 0, the algorithm will ensure that the dominant note is not present in near the strongest frequencies. Using these two different values of N will give better results in extracting the embedded watermark by allowing a certain range of error in the position of the dominant note on the list of strongest frequencies.

Four scenarios must be considered: embedding a binary 0 when the dominant note does not exist; embedding a binary 0 when the dominant note does exist; embedding a binary 1 when the dominant note does not exist; and embedding a binary 1 when the dominant note does exist. For each case, the procedure is as follows:

- To embed a binary 0, if the dominant note does not exist, the process is complete, since the algorithm must ensure that the dominant note does not exist if it is to embed a 0.
- To embed a binary 0, while the dominant note exists, the algorithm is to remove that frequency from the N strongest frequencies. In order to achieve that, the concept of destructive wave interference is used to gradually decrease the amplitude of the dominant note frequency until it is no longer

one of the N strongest frequencies. The dominant note frequency and its phase are recorded, and a phase-shifted sine wave of the same frequency is added to the frame in the spatial domain. The phase-shifted sine wave is added in small amplitude increments to try to satisfy the dual requirements that the dominant note frequency is no longer one of the N strongest frequencies, and that the root chord frequencies have not been significantly altered. If these cannot both be satisfied, the bit cannot be embedded correctly.

- To embed a binary 1, while the dominant note exists, the process is complete, since the algorithm is to ensure that the dominant note exists if it is to embed a binary 1.
- To embed a binary 1, while the dominant note does not exist, the algorithm is to add the dominant note frequency such that it figures in the N strongest frequencies of the frame. A sine wave of the dominant note frequency is generated and added to the frame. Again, since the root chord frequencies should not be altered, the algorithm must add the sine wave in small amplitude increments, until it is one of the N strongest frequencies. Currently, the phase of the root note frequency is used to create the sine wave. To increase the likelihood that the root chord does not get inadvertently altered due to the attempted insertion of the dominant note, a sine wave of low amplitude with the frequency and phase of the root note is added in small increments. This should ensure that the root chord will be detected properly, even after compression and other distortions.

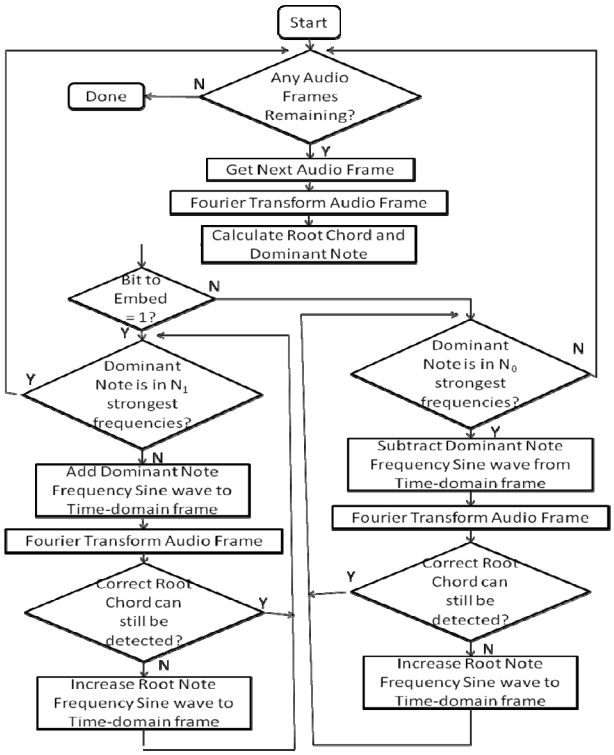

Figure 5. Watermark embedding flowchart

D. Note Detection

The final step when extracting the watermark is the note detection. The algorithm checks if the dominant note is one of the strongest N_d frequencies. N_d should be chosen such that it falls between N_1 *and* N_0. This is for robustness purposes, such that there is a larger range to detect the presence of the embedded dominant note, or a smaller range to detect its absence. The chord analysis will determine the root chord, and thus the root chord and dominant note in the 1st octave can be found, as it was in the embedder. Each of the strongest N_d frequencies is compared to the dominant note in the 1st octave. If a match is found, within a given frequency threshold, the dominant note exists and the current frame contains an embedded 1. If not, the frame does not contain the note and therefore contains an embedded 0.

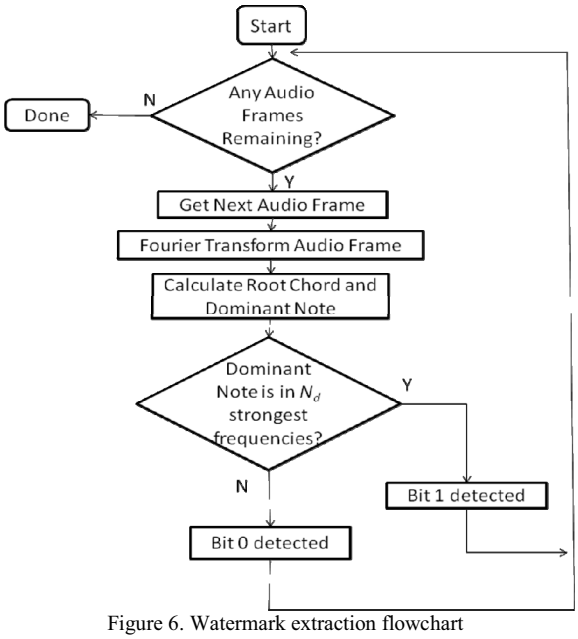

Figure 6. Watermark extraction flowchart

V. TESTING

Several tests were conducted on this algorithm to prove that it works. The algorithm was tested for effectiveness, robustness and imperceptibility.

A. Effectiveness

To test effectiveness, the embedding and extraction algorithms were run with random bit patterns as watermarks, on different works. Clips of 10 to 16 seconds were chosen from pieces of music from each of 4 broad categories of music: Rock, Pop, Rap, and Classical. The Rock clip was taken from Green Day's American Idiot, and contains loud instrumental components as well as vocal components. The Pop clip was taken from The Backstreet Boys' Incomplete, and has quieter instrumental components and vocal components. The Rap clip was taken from Fabolous' Breathe, and contains loud instrumentals and vocal components. Finally, Mozart's 4th Symphony was used for Classical, and contains only instrumental components.

25 random bit patterns were embedded and extracted from each of the works, which are WAV files at 16-bit, 48-KHz. The averages for each of the works are displayed in Figure 7.

Work	Avg. % of bits correctly detected
Rock	100
Pop	100
Classical	100
Rap	100

Figure 7. Effectiveness test results

The results show that the effectiveness is 100% in embedding and extracting random bit patterns in the above mentioned clips of music.

B. Robustness

To prove the algorithm's robustness to compression algorithms, after the embedding of the watermark, MP3 compression was used and then the watermarks were extracted and compared with the original pattern. The compression and decompression was done manually, using BladeEnc as a compressor and WinAmp's Disk Writer to return to the WAV format. Because of the time-intensive nature of manually compressing and decompressing the audio, only a few of the tests were done. The MP3 compression was set to 96-kbps, 128-kbps, and 192-kbps, all at 48-KHz with 16-bits. One test for each style of music was performed.

Work	% of bits correct at 96-kbps	% of bits correct at 128-kbps	% of bits correct at 192-kbps
Rock	78	71	78
Pop	83	92	96
Classical	85	100	100
Rap	88	88	88

Figure 8. MP3 compression robustness test results

Figure 8 shows that the effectiveness slightly decreases after MP3 compression. The effectiveness decreases in the Rap and Rock pieces due to the complexity of the tempo and complexity of the clips selected. The results may be improved by selecting better N_1, N_0, and N_d values. Also, the error can be effectively reduced using a better segmentation method. This will be discussed in the next section.

To prove the algorithm's robustness to downsampling, after embedding the watermark, the result was downsampled to 11-KHz, 22-KHz, and 32-KHz.

Work	% of bits correct at 11-KHz	% of bits correct at 22-KHz	% of bits correct at 32-KHz
Rock	97	87	97
Pop	96	100	100
Classical	96	100	100
Rap	96	100	100

Figure 9. Downsampling robustness test results

The algorithm is fairly robust to resampling, as shown in Figure 9. The extraction effectiveness decreases slightly when the 48-KHz watermark embedded work is downsampled to 11-KHz. The reason for the decrease in effectiveness of the Rock clip over the various downsampled tests is because the music is complex, and the algorithm has difficulty accurately determining the root chord. Similar to the MP3 compression robustness, this can be improved by selecting better N_1, N_0, and N_d values and using a better method of segmentation.

C. Imperceptibility

This is the least exact component of the testing. In this case, two sets of tests were conducted to evaluate how perceptible the watermark was in the work.

The first test asked 5 test subjects to see if they could identify whether the music sounded like there was something wrong with it, or whether they experienced any audible artifacts, when only hearing the watermarked works. They were asked to rate each sample on a scale from 1 to 4. 1 means artifacts are audible, and unpleasant throughout the clip. 2 means artifacts are audible, and unpleasant in only some parts of the clip. 3 means artifacts are audible, but not unpleasant. 4 means artifacts are not audible. The scores are reported in Figure 10.

Subject #	Rock	Pop	Classical	Rap
1	4	4	4	4
2	4	4	4	4
3	4	4	4	4
4	4	4	4	4
5	4	4	4	4

Figure 10. Imperceptibility test when hearing only the watermarked work

As Figure 10 shows, the test subjects could not identify perceptible alterations in the watermarked works.

The second test asked the same test subjects to gauge how different the original work was from the watermarked work, and if it was noticeable after hearing both the original and the watermarked work. The scale was from 1 to 4 again, where 1 was very noticeable, and 4 was not noticeable at all.

Subject #	Rock	Pop	Classical	Rap
1	4	4	4	4
2	4	4	4	4
3	4	4	4	4
4	4	4	4	4
5	4	4	4	4

Figure 11. Imperceptibility test when the original work is compared to the watermarked work

As Figure 11 shows, even when comparing the original works to the watermarked works, the alteration is imperceptible.

VI. Limitations and Further Improvements

Though this application presents a novel method for embedding watermarks in musical audio, there remains many limitations. This section discusses the most important ones and proposes further improvements to address these issues.

A. Embedding a 1

When embedding a binary 1 as a watermark, the dominant note of the chord is added. In order to add a note that does not exist and have it fit the chord, the correct frequency of the note to embed must be identified, a sine wave of that frequency generated, and then added to the frame. When the frequency is added at low amplitude, it might not be heard at all. This can be a problem because compression and other distortion may get rid of this note, which makes the algorithm less robust.

To compensate for this problem, notes are added at higher amplitudes. If they are louder it will not necessarily seem out-of-place. But also, if the note sounds unnatural musically, the listener will notice. Sometimes, even with the correct phase and frequency, adding a sine wave might be a little too perfect. The timbre generated by musical instruments will produce a waveform that is not exactly a sine wave, and may have other frequencies that are also generated. To overcome this problem, pre-recorded waveforms of notes generated by various instruments can be frequency-shifted to the correct note and added to the audio frame. This should solve the issue of incorrect timbre causing a listener to experience discomfort.

C. Fixed-Length Frame Segmentation

Using fixed-length frame segmentation for synchronization involves significant limitations. Robustness against cropping is the most obvious problem, since cropping would destroy the detector's ability to synchronize and detect the correct bits.

For the purpose of watermarking, fixed-length frame segmentation poses a different problem, and interferes with imperceptibility. Sometimes the segmented frame contains two chords, as shown in Figure 12, so the frequency analysis will not correctly identify a single chord. Embedding the dominant note may produce unpleasant results, and will be perceptible since the note is incorrect. Also, the embedded note may fit one chord, but in the transition and in the second chord it will be out-of-place.

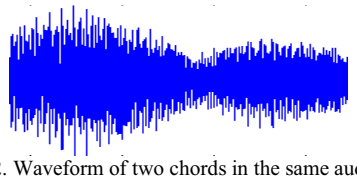

Figure 12. Waveform of two chords in the same audio frame

Robustness to cropping, compression, and downsampling can all be improved by using the note-length frame segmentation method described in [10].

D. Watermark Detection

The proposed detection method does not distinguish between watermarked works and works that contain no watermark. It assumes that all works passed to the decoder contain a watermark. A few techniques may be used, such as using a gradual phase-shift to identify the frequencies which carry the watermark. Gradual phase-shifts are usually imperceptible to the human ear, and are rarely found in original musical works [10]. This may work when adding a note to the audio frame, but not for removal. For the phase-shifted approach to work, the detector must assume that at least one bit with a binary 1 has been embedded. If the work contains at least one binary 1, identified with an added dominant note with a phase-shift from the start of the frame, a watermark exists. Otherwise, it contains no watermark.

VII. Conclusion

This paper proposes some original ideas for an algorithm which can serve as the basis for a new direction in watermarking musical audio. Music contains a mathematical language that can well be exploited to embed imperceptible watermarks. Currently, the algorithm uses only the dominant note to serve as the carrier of the watermarked bit, but the capability of the algorithm can be improved by using several different notes which belong in the chord to embed many bits. To improve robustness, several different notes can embed the same bit in a spread-spectrum method.

Audio watermarking has had many different techniques, involving analysis and modification in both the frequency domain and spatial domain. Some content-adaptive algorithms have proposed the use of musical theory and musical structure analysis to effectively embed watermarks. This paper proposes another algorithm which uses more in-depth musical theory to aid in the embedding of watermarks.

References

[1] A.H. Tewfik, M. Swanson, B. Zhu, and L. Boney, "Robust audio watermarking using perceptual masking," Signal Processing, vol. 66, no. 3, pp. 337–355, 1998.

[2] S.K. Lee and Y.S. Ho, "Digital audio watermarking in the cepstrum domain," *IEEE Transactions on Consumer Electronics*, vol. 46, no. 3, pp. 744-750, 2001.

[3] J. Seok, J. Hong, and J. Kim, "A Novel Audio Watermarking Algorithm for Copyright Protection of Digital Audio," *ETRI J.*, vol. 24, no. 3, pp. 181-189, June 2002.

[4] S. Xiang, J. Huang, and R. Yang. "Time-scale Invariant Audio Watermarking Based on the Statistical Features in Time Domain," *Proc. of the 8th Information Hiding Workshop*, 2006

[5] I. Cox and M. Miller "Electronic watermarking: the first 50 years", *Proc. 4th IEEE Workshop on Multimedia Signal Processing*, Cannes, France, pp. 225-230, October 2001.

[6] F. Hartung and M. Kutter "Multimedia Watermarking Techniques", *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1709-1107, July 1999.

[7] D. Gruhl, A. Lu, and W. Bender, "Echo hiding," in *Information Hiding*, Springer Lecture Notes in Computer Science, vol 1174, pp. 295-315, 1996.

[8] F.A.P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Attacks on copyright marking systems," in *Information Hiding*, Springer Lecture Notes in Computer Science, vol 1525, pp. 218-238, 1998.

[9] W. Bender, D. Gruhl, N. Morimoto, and A. Lu, "Techniques for data hiding," *IBM Systems Journal*, vol 35, no. 3-4, pp. 313-335, 1996.

[10] C. Xu, N.C. Maddage, X. Shao, and Q. Tian,"Content-Adaptive Digital Music Watermarking based on Music Structure Analysis," *ACM Trans. Multimedia Computing, Comm. and Appl.*, vol 3, 1, Article 1, 2007