Honors in CS Fall 2010/Spring 2011
Digital Audio Watermarking
Thomas Ramfjord

Introduction to Digital Audio Watermarking

Digital audio watermarking is a process through which a secret signal is embedded in a digital audio file. In this paper we will go over the requirements of audio watermarks, their applications, and how the details of their implementation. Audio watermarking is an extremely diverse field, with many different algorithms aimed at different applications. This paper will merely provide an introduction to the subject, focusing on one simple application, and avoiding many of the more complex techniques. The goal will instead be to develop a solid base in every aspect required for the understanding of more complicated watermarking schemes. In addition to the general overview, we will examine in depth two seminal papers in the field of audio watermarking. The first was written by Ingemar Cox in 1997, and describes a general purpose technique for watermarking media files called spread spectrum watermarking. Originally applied to images, spread spectrum watermarking has gained wide usage in all types of digital media, including audio and video. The second was written by Ernst Terhardt in 1981, and describes a psychoacoustic model for determining how salient a pitch is to the human ear, given a frame of audio data. Psychoacoustic models are extremely important in audio watermarking due to the particularities of the human auditory system (HAS). Lossy audio compression has gained widespread usage, and typically frequency components that are not audible will be stripped from a signal as it is compressed. Finally, an implementation of the described techniques will be provided. Before explaining these details, we will cover what specifically an audio watermark is, and what it is designed to do. The following requirements and applications were described by Nedeljko Cvejic and Tapio Seppanen.

Requirements and aspects of Audio Watermarks

- **Perceptual Transparency:** Perhaps the most basic requirement of any digital watermark, not specifically audio, is that it should be perceptually transparent. In audio watermarking, there should be no audible difference between the watermarked and unwatermarked signal.

- **Watermark Bit Rate:** The bit rate of a watermark, usually expressed in bits per second, indicates how much information can be encoded in the watermarked signal. The acceptable bit rate varies tremendously from application to application, and sometimes bit rates as low as .5 bps are acceptable. Higher bit rates are the territory of steganography more than watermarking, and some algorithms have succeeded in embedding up to 150 kbps without violating perceptual transparency. The amount of bits per second that can be encoded is heavily dependent on the audio encoding scheme. A typical LPCM (linear pulse code modulation) encoding on a CD contains precisely 1411.2 kbps (2 channels, 44,100 hz samples, 16 bits per sample). A lossily compressed mp3 of the same perceptual quality may only use 64 kbps. Clearly it is possible to store more audio data on raw LPCM data than lossily compressed data.

- **Robustness:** Robustness is the ability to detect a watermark after common signal processing procedures. Robustness is typically inversely correlated with bit rate - there is no way

you will be able to encode 150 kbps if you want your watermark to be resistant to common lossy audio compression. In "fragile" watermarking algorithms robustness is considered undesirable, but for most it is necessary.

- **Blind or Informed Detection:** Informed detection algorithms require the use of the original audio in order to detect the watermark. There exist Blind detection algorithms, which allow the watermark to be detected with nothing but the watermarked algorithm and a secret key. The remainder of this paper will focus on informed detection, as it is far less complicated.
- **Security:** In digital watermarking it is typically desirably that nobody besides the owner of the original audio should be able to detect the watermark, let alone extract it.
- **Complexity:** Complexity is a metric of the length of time it takes to encode/decode the watermark. For an application such as audio fingerprinting, complexity is of little concern. Some applications must encode the watermark in real time, and for those complexity can be a serious limitation.

Example Applications of Audio Watermarks

- **Ownership Protection:** The goal of audio watermarking in ownership protection is to be able to demonstrate ownership of a digital audio signal. The algorithm should be robust and secure, but often a low bit rate is acceptable. This paper focuses on ownership protection because it's in many ways the simplest application of audio watermarking.
- **Authentication / Tampering Detection:** Here the watermark is used to ensure that the audio file has not been tampered with. Robustness is undesirable - one should only be able to detect the watermark if the file is in its original state. Blind detection is a strict requirement if the algorithm is to be run by a user.
- **Fingerprinting:** A file is watermarked with a "fingerprint" for a specific user before being sold to them. This allows the owner of the file to detect who a file was sold to if it has been redistributed. The algorithm must be highly robust, but a low bit rate is once again acceptable. One further requirement is unique to fingerprinting: anti-collision. Any two fingerprints must be different enough that there are few false positives when looking for a watermark.
- **Broadcast Monitoring:** In broadcast monitoring, a signal is watermarked in real time with information about the source. This requires high capacity, fast computation, and blind detection, but robustness is typically not emphasized.

Perhaps the most researched and useful of the above applications are ownership protection and fingerprinting. The MPAA estimated in 2002 (Valenti) that the movie industry loses over US$3 billion annually solely through physical piracy. It is now possible to make a perfect illegal digital copy of a media file, and distribute it widely over the internet. As a result, the problem of effective copyright protection has been gaining interest worldwide among both academic and business communities. My focus will be on describing a simple but effective audio watermarking technique for copyright protection. Thus, the watermarking scheme will emphasize robustness and security.

Sound File Basics

Now that we have covered what an audio watermark is, and what one may be used for, we can start to understand how to build one. Sound, at it's most basic form, is nothing but variations in air pressure over time. We could therefore think of an audio file as a continuous function S :TIME $\rightarrow$ PRESSURE. In order to encode sound data in a computer, the air pressure level is sampled at regular intervals. Audio CD's use the Red Book standard, which specifies that a sample sound pressure level be encoded with 16 bits (unsigned), 44,100 times every second. An audio file in a CD contains little information besides the number of samples, followed by the set of samples, because all CDs adhere to the same standard.

When humans here sounds, we don't hear variation in the sound pressure level. Instead we hear frequencies (technically pitches), or atonal noise, over time. The sampling rate is intimately related with which pitches can be heard, and it was no accident that the sampling rate of a CD was chosen to be what it is. According to Nyquist's theorem, the maximum frequency that may be encoded into a signal of sampling rate r is $\frac{r}{2}$. All frequencies higher than $\frac{r}{2}$ are subject to aliasing, and will sound like lower frequencies. Therefore, the highest frequency that may be encoded into a CD audio file is 22.05 kHz. Typically the maximum audible frequency for humans is around 20 kHz. To avoid higher frequencies being aliased, as a song is recorded onto a CD it passes through an analog low pass filter which removes all the frequencies above 22.05 kHz

Audio Signal Processing

The following section describes some introductory signal processing ideas that are necessary to understand audio watermarking, but don't fall under the domain of audio watermarking at all. The frequencies of a time domain signal are extracted through a Fourier or Fourier like transform. The idea behind a FLT (Fourier like transform) is that any signal or function can be represented as a sum of sine waves of different frequencies and phases. While the original Fourier transform was applied to analog or continuous signals, I will describe a variant for discrete signals called the DFT (discrete Fourier transform).

It might seem strange that an aperiodic signal can be represented as a finite sum of sine waves. The truth is that it can't - it would in fact require an infinite number of sine/cosine waves to represent any finite length signal. Therefore when calculating the DFT of a discrete time domain signal, it is assumed that the signal repeats indefinitely after the range of the signal.

Any length N signal of time length s can be represented by the sum of $N/2 + 1$ sine waves. Due to the Nyquist limit, none of these frequencies may be larger than $N/(2s)$. There is a sine wave with frequency 0, $1/s$, $2/s$, and so on up until $N/(2s)$. Due to the way the DFT is calculated, each wave encodes phase by encoding the amplitude of both a sine and cosine component, in the form of a complex number. The real component represents the amplitude of the cosine wave at a given frequency, and the imaginary component represents the amplitude of the sine component.

A couple things are worth noting in order to understand the how the DFT encodes a signal. Firstly, imagine we have a sine wave $a\sin(ft)$ and a cosine wave $b\cos(ft)$, where f, a and b are constants, and t is a variable. Because both waves have the same frequency, they will sum to form a single sine/cosine wave of the same frequency and a new phase.

$$a\sin(ft) + b\cos(ft) = \text{sgn}(a)\sqrt{a^2+b^2}\cos\left[ft + \tan^{-1}\left(\frac{b\sin f\pi/2}{a + b\cos f\pi 2}\right)\right]$$

where $\text{sgn}(a)$ is just the sign (+-) of a. Further any equation of the form $a\sin(0x)$ will be the zero function. Any equation of the form $a\sin(2\pi t)$ will also be 0 at each of the N points it is sampled at, because it's period is N/s and it will be 0 at the beginning of each period. Because these two components are useless, the DFT really transforms N time domain components with exactly N frequency domain components. The 0 frequency cosine component, on the other hand, will just be a horizontal line. It's called the DC offset, and it's value will represent the average of all the time domain signals.

Although the algorithm we actually use for calculating DFT's, the fast Fourier transform (FFT), is more complicated than I would like to go into, the standard method is instructive. To find the amplitude of a sine/cosine component at frequency f, we can *cross-correlate* a frequency f sine/cosine wave with the time domain signal. The cross-correlation of two signals is simply their dot-product. We will use this definition of correlation (though later it will be normalized) later to determine whether a watermark is present in a signal. Therefore, the frequency component of signal s at index k can be found by the following equation (recall that cosine amplitude is encoded in the real value, and the sine value is encoded in the imaginary value).

$$f_k = \sum_{j=0}^{N} s_j \cos\left(\frac{2\pi kj}{N}\right) + i\left[s_j \sin\left(\frac{2\pi kj}{N}\right)\right] = \sum_{j=0}^{N} s_j e^{-\frac{2\pi i}{N}kj}$$

The reversal of this process - calculating a time domain value given the set of frequency domain values, is obtained by simply summing the sine and cosine functions at each time.

$$s_j = \sum_{k=0}^{N/2+1} \text{real}(f_k)\cos\left(\frac{2\pi kj}{N}\right) + \text{imag}(f_k)\sin\left(\frac{2\pi kj}{N}\right)$$

The formulation of this using Euler's formula is not evident in the real to complex Fourier transform, but naturally extends from the complex to complex Fourier transform. For more information on Fourier transforms and the FFT, there are several resources available, such as (SIGNAL PROCESSING BOOK).

Lastly, because the DFT assumes that the signal it is given represents a single period of a repeating function, distortions often arise in the frequency domain. To reduce the effect of periodic distortion, various window functions are often applied to the time domain before a Fourier analysis is performed. Each element of the time domain is multiplied by a window coefficient. Common window functions include the Hamming window, the Hann window, or simply the square window (in which all the coefficients are 1).

The Cox et al. Spread Spectrum Technique

Note: from this point on, I will include example code in ANSI c of an implementation of the cox et al algorithm specifically for audio watermarking. The entirety of the code used for this project is available separately.

In 1997, Ingemar Cox, Joe Kilian, F. Thomson Leighton, and Talal Shamoon described a watermarking technique that has since gained widespread usage. Before this several digital watermarking methods had been proposed. Turner proposed a method for inserting an identification string into the least significant bits of an audio stream. Caronni suggested adding small geometric patterns to digitize images at brightness levels that are imperceptible. One could imagine doing something similar to an audio stream with bit sequences at imperceptible noise levels. Tenaka *et al.* described schemes in which watermarks are made to resemble quantization noise and embedded. This last method relies on the fact that quantization noise is typically imperceptible to viewers.

The intellectual leap made by Cox *et al.* was that, for a watermark to be robust, it must be embedded in the perceptually *significant* regions of a signal. Although Cox was concerned with image watermarking, the idea holds true for audio. One of the most common attacks (intentional or unintentional) to digital watermarks is lossy compression. If a watermark is embedded in the least significant bits, a lossy compression of the signal will likely not encode the watermark at all. Similarly, high frequency noise, or anything the compression algorithm deems imperceptible, is likely to be removed.

Cox viewed the frequency domain of a signal as a communication channel, through which the watermark is transmitted. They drew from spread spectrum communications in the formulation of their algorithm. In spread spectrum communication, a radio signal is spread thinly over a large bandwidth. The receiver then "despreads" the signal in order to decode it. One advantage of this in radio communication is that if there is a large jamming signal in a small frequency range, when the signal is despread the jamming signal will actually be spread out, and leave the original signal relatively unaltered. To jam a spread spectrum radio signal, one would have to jam all the frequencies.

The equivalent in watermarking is to spread the watermark over many frequency bins. In order to remove the watermark, one would have to drastically alter many important frequencies. The watermark will remain undetectable because in each frequency the watermark is embedded with low energy. It is possible to increase the energy present in some sections using knowledge of masking phenomena, which for audio signals will be described in the psychoacoustics section.

Although Cox watermarked images in his 1997 paper, as I have said before the content applies equally well to audio watermarks. For the remainder of this section I will unfaithfully write about his spread spectrum technique referring to an audio signal rather than an image.

The watermark should consist of a sequence of real numbers $X = x_1, ..., x_n$, where each x_i is chosen independently from a distribution with a mean of 0 and a variance of 1 (denoted $N(0, 1)$). To implement this, one might generate a normal distribution X, and for each bit b_i of the spread watermark, multiply x_i by -1 if $b_i = 1$. Although other distributions such as $\{-1, 1\}$ are possible, or indeed more intuitive, $N(0, 1)$ will be useful in quantifying the likelihood that a watermark was embedded in a signal. The detection probability exploits the fact that the watermark is chosen from a normal distribution. Furthermore, many distributions leave the watermark vulnerable to attacks which embed multiple other watermarks into the same signal.

Practically speaking, the Box-Muller transform can be used to generate a normal distribution from a uniform distribution. This is useful because a typical random number generator emulates a uniform, rather than normal, distribution. Below, norm_double() is a sample implementation of the Box-Muller transform for generating a number from $N(0, 1)$, using rand_double to generate a double from a uniform distribution.

```
double norm_double(){
   static double Z2;
   static int Z2_def = 0;

   if(Z2_def){
      Z2_def = 0;
      return Z2;
   }

   // generate two new independent normally distributed numbers, Z1 and Z2, from
   // uniformly distributed numbers U1 and U2.  U1 and U2 must be from the
   // uniform distribution (0,1] for Z1 and Z2 to be from the standard
   // distribution with mean 0 varience 1.
   double U1 = rand_double();
   double U2 = rand_double();
   if(U1 == 0) U1 = 1;
   if(U2 == 0) U2 = 1;

   Z2_def = 1;
   Z2 = sqrt(-2 * log(U1))*sin(2 * M_PI * U2);
   // below would be Z1
   return sqrt(-2 * log(U1))*cos(2 * M_PI * U2);
}
```

From this we can easily generate a random sequence of real values via generate_noise below.

```
void generate_noise(double *buffer, int buffer_len)
{
  for(int i = 0; i < buffer_len; i++)
```

```
    buffer[i] = norm_double();
}
```

Finally, we can generate a new random noise sequence that depends specifically on the watermark message, via embed_to_noise, where wmark->message is the message of length wmark->len we wish to embed. This simply flips the sign of each double, making each message W unique to both the random number generator seed and the watermark message itself. Our final W results from embed_to_noise being called on a noise sequence generated by generate_noise.

```
void embed_to_noise(double *noise_seq, int noise_len)
{
   static int index = 0;
   static int bit_mask = 1;
   char *message = wmark->message;

   for(int i = 0; i < noise_len; i++){
      if(message[index] & bit_mask)
         noise_seq[i] *= -1;

      bit_mask <<= 1;
      if(! (bit_mask & 0xff)){
         bit_mask = 1;
         index++;
         if(index >= wmark->len)
            index = 0;
      }
   }
}
```

Inserting and Extracting the Watermark

From the original audio file A, a sequence of values $V = v_1, ..., v_n$ is extracted. Cox proposed that the V should consist of the n highest magnitude coefficients in the frequency domain. An example for obtaining the indices of V is presented below in the method get_n_biggest, which accepts a complex buffer of length len, and an integer n to specify the size of V. This could be used on a complex buffer on an individual frame of audio. Specifically in audio watermarking, car must be taken to avoid choosing the DC offset and the nyquist frequency when obtaining V.

```
int get_min_from_indices(int *indices, complex *buffer, int n)
{
  if(indices[0] == -1) return 0;

  int min = 0;
  int min_index = 0;
  double min_pow = pow_elem(buffer[indices[0]]);
  for(int i = 1; i < n; i++){
    if(indices[i] == -1)
```

```
      return i;
    if(min_pow > pow_elem(buffer[indices[i]])){
      min_pow = pow_elem(buffer[indices[i]]);
      min = indices[i];
      min_index = i;
    }
  }
  return min_index;
}

int *get_n_biggest(complex *buffer, int len, int n)
{
  int *indices = (int *) malloc(n * sizeof(int));
  for(int i = 0; i < n; i++) indices[i] = -1;

  double min_pow = -1;
  int min_index=0;
  for(int i = 0; i < len; i++){
    if(pow_elem(buffer[i]) > min_pow){
      indices[min_index] = i;
      min_index = get_min_from_indices(indices, buffer, n);
      min_pow = (indices[min_index] == -1) ? -1 :
                pow_elem(buffer[indices[min_index]]);
    }
  }
  return indices;
}
```

Each value of V is then modified to produce V' in one of the following manners, where α is a constant. V' is then inserted back into A.

$$v_i' = v_i + \alpha x_i \tag{1}$$
$$v_i' = v_i(1 + \alpha x_i) \tag{2}$$
$$v_i' = v_i(e^{\alpha x_i}) \tag{3}$$

Equation (1) is always invertible, while the others are invertible if $v_i \neq 0$, which is almost universally the case. Equation (1) is unsuitable to V's which vary widely; if v_i is very small the signal will be distorted too strongly, and if it is too large v_i will be distorted too weakly. Cox *etal.* noted that (2) and (3) gave similar results when ax_i was small, and used (2) for most of their experiments. Cox et al. also suggested modifying the α values for each i, in order to better account for the amount of information that may be embedded into a signal without losing audio transparency. Although I have mostly used (2) myself, I suspect that (3) would be more resistant to audio compression, as in lossy compression the signal is often stored logarithmically.

To extract the watermark, V^* is extracted from a A^*, a potentially modified version of A. This operation may use information about A, thus this is an informed detection watermarking schema.

The simplest way to extract V^* would in fact just be to extract the indices of V from A, and use the same indices on the transform domain of A^*. X^* is extracted from V^* using the inverse of the appropriate method above, and compared with X using the normalized cross-correlation shown below, where $\langle a, b\rangle$ denotes the inner (dot) product of a and b

$$\text{sim}(X, X^*) = \frac{\langle X^*, X\rangle}{\sqrt{\langle X^*, X^*\rangle}}$$

There will always be the possibility that through random chance X and X^* will be very similar, and therefore it would be useful to compute the probability of a false positive. We will assuming that the manipulations on A^* are not dependant on X, meaning that X^* and X are independent. X^* is not assumed to be normally distribute, but rather each x_i^* is assumed to be a constant. Using the well-known formula for the distribution of a linear combination of independent variables where one is normally distributed,

$$N\left(0, \sum_{i=1}^{n} x_i^{*2}\right) = N\left(0, \langle X^*, X^*\rangle\right)$$

Because we divide similarity by $\sqrt{\langle X^*, X^*\rangle}$, the standard deviation of $\text{sim}(X, X^*)$ is actually 1. To decide whether a watermark was present, one determines whether $\text{sim}(X, X^*) > T$, for some threshold T. The probability of a false positive is equal to the probability of a random variable exceeding it's mean by T standard deviations. The authors used $T = 6$.

Although the length of the watermark isn't implicit in the definition of similarity, $E\left[\langle X, X\rangle\right] = \sqrt{n}$ if X has mean 0 and variance 1. Therefore, similarity is proportional the square root of the length of the watermark, as well as clearly being linearly proportional to α. The similarity can also be boosted via post processing X^*. It was found that $E[X^*]$ was not always 0, and the transformation $x*_i \rightarrow x_i^* - E[X^*]$ improved similarity measures. Also it was found that there were often outliers which had changed dramatically. Although the effect of these is slightly reduced by the spread spectrum technique itself, it can be further reduced by simply setting very abnormal values to 0. The goal of such a transformation would be to lower $\langle X^*, X^*\rangle$.

Limits in Audio Applications

Although the Cox *et al.* spread spectrum technique is resistant to a number of attacks, including jpeg compression of magnitude 10, there is one area that is typically improved upon in audio watermarking schemas. MPEG 3 Layer 3 compression typically reduces a file by a factor of 13, and much of the data that is in the original watermark is simply dropped. It is possible to do this while maintaining audio fidelity due to the sophistication of the human auditory system. The loudest sound a human can perceive is 10^{14} times louder than the quietest. Additionally, the highest audible frequency is over 5 thousand times higher than the lowest. Finally, complex masking effects are applied to an audio signal before it is perceived by the brain. In order to adapt spread spectrum watermarking to audio signals while maintaining robustness, we will need to add a psychoacoustic model.

Psychoacoustic Model

A psychoacoustic model takes an input audio signal and tries to determine how it will be heard as pitches by the human ear. Often this is a small fraction of the frequency components in a given window. As a basic example, imagine a conversation is being recorded as a fire alarm goes off. A person might not be able to hear the conversation anymore, but it is still being recorded. Psychoacoustic models are heavily used in lossy audio compression, because it is only necessary to encode the components that people hear in order to reproduce a perceptually faithful signal. In 1981, Ernst Terhardt, Gerhard Stoll, and Manfred Seewann proposed an effective psychoacoustic model for tonal signals that is widely used in lossy compression algorithms such as MPEG to this day.

Before going into depth with Terhardt's model, it will be useful to cover some basic precepts of the HAS (human auditory system). As sound enters the ear, different frequencies are perceived by different locations along the cochlea. In effect, the cochlea acts as a series of low-pass filters, where at each filter the higher frequencies are absorbed and converted into neural signals. This could also be modeled as a set of band-pass filters, which are commonly called critical bands. As early as 1961 Eberhard Zwicker had measured the size of these filters, and developed a frequency scale to represent them which he called the Bark scale. We can convert a frequency f in Hz to Barks by

$$Bark = 13 \arctan(0.00076f) + 3.5 \arctan((f/7500)^2)$$

The Bark scale is roughly linear at first, and then logarithmic. Additionally, when a frequency is heard by the ear it is referred to as a pitch, as the what we hear is often slightly different from the frequencies present in the sound. For example, a frequency of high amplitude will often sound as if it has a higher frequency as well.

Terhartd's model is based on the concept that the pitches we hear include "virtual pitches," in addition to the "spectral pitches" present in an audio signal. A virtual pitch would be the note one hears when an instrument plays a tone, and the spectral pitches would be the frequency components that we hear (including harmonics). In order to extract the virtual pitches, a spectrum analysis is applied in the form of a DFT with a Hann window. From the DFT, we obtain the set of wave amplitudes L, which are converted to decibels (dB). The first step in obtaining pitch salience is determining the tonal components of the spectral sample. There are several ways of determining pitch, but in his paper Terhardt specifies that a frequency component L_i is tonal if the following conditions are met.

$$L_{i-1} < L_i \geq L_{i+1}$$
$$L_i - L_{i+j} \geq 7 \text{ dB for } j = -3, -2, +2, +3$$

If this is the case, it is assumed that the considered group of 7 frequency samples represents a tonal component. The threshold of 7 dB was determined experimentally by Terhardt. If these criteria do not hold for any frequencies in a group of 7, those frequencies are considered to be noise components rather than tonal components - like a drum or static noise. This makes sense, because

if a tone is audible then one frequency component should outshine the rest. An approximation of the precise component frequency f_c can then be made with the following formula.

$$f_c = f_i + 0.46(\text{Hz/dB})(L_{i+1} - L_{i-1})$$

The masking effects between pitches can then be evaluated. The degree to which a tonal component is salient is describe by it's sound-pressure level excess LX. The sound-pressure level is just the amplitude of the frequency component expressed in dB. The SPL excess of the u'th tonal component, LX_u can be expressed as

$$LX_u = L_u - 10\log_{10}\left[\left(\sum_{v\neq u}^{\text{tonal}} 10^{L_{Ev}(f_u)/(20dB)}\right)^2 + I_{Nu} + 10^{L_{TH}(f_u)/(10dB)}\right] \text{dB}$$

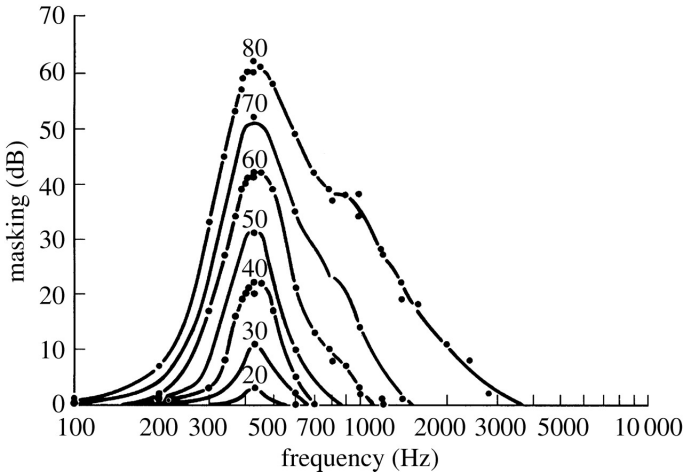
L_u is the component's SPL, and $L_{Ev}(f_u)$ is the amount the tonal component v masks u. I_{nu} is the masking effect of the noise (the power of frequencies that can't be assigned to a tonal component) around f_u. $L_{TH}(f_u)$ is the hearing threshold at frequency f_u. The minimum threshold of hearing varies with frequency, and is typically lower towards the middle of the audible range expressed logarithmically. Terhardt uses model to calculate L_{Ev} which is linear on the Bark scale. To calculate the effect of a frequency higher than f_u, a constant slope is used. For lower frequencies, the slope varies according to the strength of the signal. Specifically, we define

$$L_{Ev}(f_u) = L_v - s(z_v - z_u)$$

where z_v and z_u are f_v and f_u represented in Barks respectively. The slope is defined as

$$s = \begin{cases} 27\text{dB / Bark} & \text{if } f_u \leq f_v \\ [-24 - (0.23\text{ kHz}/f_v) + (0.2L_v/\text{dB})]\text{dB / Bark} & \text{if } f_u > f_v \end{cases}$$

As we can see, in this equation the masking effect of a component v on all the other components is approximated as a sort of triangle (linear on the bark scale), although in reality it is shaped with a more gentle curve, shown below (Moore)



To calculate I_{Nu} Terhardt adds all of the non-tonal components within .5 barks of u. Here, a

component is non-tonal if it is not within 2 frequencies of a tonal component. That is to say, given tonal component v, $f_{v-2}, \ldots, f_{v+2}$ are all considered tonal, but f_{v-3} or f_{v+3} are not. Recall that to determine whether a component is tonal, $f_{v-3}, \cdots, f_{v+3}$ are all examined. Finally the threshold of quiet L_{TH} for frequency f in Hz can be calculated with

$$L_{TH}(f_u) = \{3.64(f_u/1000\text{Hz})^{-0.8} - 6.5e^{-0.6(f_u/1000\text{Hz}-3.3)^2} + 10^{-3}(f_u/1000\text{Hz})^4\} \text{ dB}$$

Now that we have calculated the SPL excess, our job is not entirely done. According to Terhardt, the "salience" of a pitch depends not only on our ability to distinguish it, but also on it's frequency. In the following formula, WS_u approximates the salience of a pitch according to the HAS.

$$WS_u = \left[1 - e^{\left(\frac{-LX_u}{15 \text{ dB}}\right)}\right] \left[1 + 0.07\left(\frac{f_u}{0.7 \text{ kHz}} - \frac{0.7 \text{ kHz}}{f_u}\right)^2\right]$$

Note that the effect of SPL excess decreases exponentially with magnitude, and particularly starts to wane around 20 dB. This means that if there are multiple frequencies which are clearly audible, how "loud" one is perceived to be is largely dependent on its frequency.

Terhard further refines this model with the notion of virtual pitches, or pitches that are not directly in the signal but which we hear due to the nature of the HAS. The virtual pitches are the fundamental frequencies of the spectral components. They are weighted and calculated in the same manner, but given some factor of precedence in the total weights. Terhardt suggests the further study is needed to determine the precise factor, but recommends further weighting the spectral pitches with a factor of .5 when examining all of the weights combined. Because we are concerned with embedding a watermark in a real signal, it is not necessary to examine its virtual pitches.

Attacks on Audio Watermarks

A spread spectrum audio watermark with a decent psychoacoustic model is resistant to most common signal processing attacks, such as compression, and even "cropping." If only a section of the signal is available to test, the rest is filled in with the original (non-watermarked) sample, and a standard cross-correlation is used. There is one attack, however, which this technique is vulnerable to.

Desynchronization Attacks

If the received file A^* is altered in such a way that the frames of A^* overlap little or not at all with the frames of A, it can be difficult to correlate the extracted watermark with the original - they are no longer synchronized. Below are a few of the many techniques have been proposed to combat this.

- **Exhaustive Binary Search:** The most intuitive solution would be to search for the watermarked signal taking into account all possible attacks. This is impractical not only because it's computationally infeasible, but because as the search space grows, so does the probability of a false positive.

- **Redundant Watermark Embedding:** If instead of embedding a signal once, it is repeated throughout the audio file, then the decoder will have a better chance of synchronizing. The normalized correlation may be performed only at the center of each signal through windowing. The frequency components will be largely similar, and the correlation will be correct as long as the linear shift is less than $floor(N/2)$ samples (He and Scordilis).
- **Synchronization Marks:** Synchronization masks know to both the embedder and decoder are encoded at various points in the signal. They typically don't carry any watermark information. The synchronization marks often PN sequences, and can be searched for by the decoder. Several authors (Garcia, 1999, Jung et al., 2003) have successfully used PN sequences as synchronization marks to achieve robustness against synchronization attacks. These marks have their own limitations, however, as it may be possible for an attacker to guess the PN sequence and perform template estimation attacks (Gnomes, 2001).
- **Feature Points or Content Synchronization** Salient points in the host signal that are invariant, or nearly so, are identified and extracted by both the encoder and the decoder. These points may be perceptually significant and not altered, to ensure that they still exist after signal processing techniques have been applied. Several authors have used feature points for robust watermarking schemes, but it can be difficult for this method to achieve accurate sample to sample synchronization. The advantage is that the watermark bit rate is unaffected.

Results

Although my implementation of the Cox et al. and Terhardt algorithms are incomplete (and still a little buggy), already it shows resistance to lossy compression and common signal processing techniques. Embedding a W of size 50 in each frame of size 1024, the similarity is in the end 56.467413. Lossily encoding and decoding the watermarked file retains a similarity of 13.945359, which is still exceedingly good. Unlike the Cox et al.'s implementation, my version did not demonstrate robustness against multiple watermarks being embedded. The current version of the source can be found at *https://github.com/ramfjord/audiomark*

Future Research

There are many aspects of spread spectrum audio watermarking that have not even been covered in this paper. The use of transforms other than the DFT is becoming prominent in watermarking schemas. Particularly the discrete wavelet transform is seeing use in psychoacoustic models, because it, like our ears, places more emphasis on lower frequencies than higher ones. Synchronization methods are an active area of research, as they are one of the strongest attacks available to spread spectrum watermarks. Time or frequency scaling of the order of 5-10% introduces little distortion to audio quality, but can be extremely debilitating to spread spectrum watermarks (He and Scordilis). Blind watermarking techniques have been completely omitted. Finally, watermark fingerprinting is an extremely interesting application, which specifically requires low collision rates among different watermarks.

References

Caronni, G. *Assuring ownership rights for digital images.* Proc Reliable IT systems, VIS'95.

Cox, I. J., Kilian, J., Thomson Leighton, F. Talal Shamoon, *Secure Spread Spectrum Watermarking for Multimedia.* IEEE Transactions on Image Processing vol. 6, No. 12, December 1997.

Cvejic, N., Seppanen, T. *Introduction to Digital Audio Watermarking.* Digital Audio Watermarking Techniques and Technologies, 2008.

Garcia, R. A. *Digital watermarking of audio signals using a psychoacoustic model and spread spectrum theory.* Proceedings of the Audio Engineering Society Meeting, New York, September 1999.

He, X., Scordilis, M. *Spread Spectrum for Digital Audio Watermarking.* Digital Audio Watermarking Techniques and Technologies, 2008.

Moore BCJ, *Basic auditory processes involved in the analysis of speech sounds* Philosophical Transactions of the Royal Society, London B Biological Sciences, 2008, 363:947-63

Smith, S. W. *"The Scientist and Engineer's Guide to Digital Signal Processing.* copyright 1997-1998

Tanaka K., Nakamura Y., Matsui K., *Embedding secret information into a dithered multi-level image.* Proc. 1009 IEEE Military Communications Conference, 1990

Terhardt, E., Stoll, G., Seewann, M. *Algorithm for extraction of pitch and pitch salience from complex tonal signals.* Institute of Electroacoustics, Techical University, D-800 Nybucg 2, September 1981.

Turner, L.F. *Digital data security system.* Patent IPN WO 89/08915, 1989

Valenti, J., *Piracy threatens to destroy movie industry and U.S. economy.* Testimony before the US Senate foreign relations committee. February, 2002