Automatic Identification of Sound Recordings

Vidya Venkatachalam, Luca Cazzanti, Navdeep Dhillon, Maxwell Wells

1 Introduction

The electronic distribution of music (EDM) via the internet offers potential benefits to both sellers and buyers. Sellers benefit because many of the costs of manufacturing, transportation, storage and display are reduced or removed. Buyers benefit because they have access to a huge catalog of music, with the ability to purchase and enjoy music instantaneously. Despite these recognized advantages, the music industry has been slow to adopt EDM. The intransigence on the part of the industry is due, in part, to a fear of piracy and a fear of change. However, there is growing acceptance that EDM will be a significant distribution channel in the future. There is also general agreement that for EDM to be successful, there will need to be a system for Digital Rights Management (DRM).

We define a DRM system as one that registers ownership of content, and monitors and controls its use. Ultimately, the purpose of DRM systems is to facilitate proper compensation for the rights owners and creators of music. Regardless of the degree of control used in the systems to accomplish this, we concluded that identification, by which we mean the ability to recognize a copy of a sound recording as being the same as the original, was a necessary system component.

In this article, we describe MusicDNA, an automatic song identification component of a DRM system for music [19] that we built at Cantametrix Inc., Bellevue, WA. The key pieces of this component are currently being deployed by Gracenote Inc., Berkeley, CA. Our primary objective was to design and build a system, in a time frame of about six months, that was accurate and

scalable up to millions of songs and millions of simultaneous users. In this article, we provide details of MusicDNA, and outline the process by which we arrived at the final system configuration. Our goal is to document our experience so that it can serve as a reference to those seeking to build a complete DRM system for music.

1.1 System Goals

We generated a set of broad system goals after extensive consultations with potential customers and users. These included record companies, music publishers, the Recording Industry Association of America (RIAA), worldwide rights organizations, performance rights organizations, peer-to-peer (P2P) file sharing companies, search engines, radio stations, music monitoring companies, web broadcasters, software juke box manufacturers, music sellers, third party data providers (providing extended meta data, reviews etc), third party commerce providers (selling music-related merchandise), musicians, and music fans. Based on the system goals, we identified the following set of high-level functional objectives:

- The system should scale to millions of songs: Based on the world music catalog in 2000, we estimated that a system should be able to identify at least 10 million original song recordings.
- The system should identify both legacy and new music files: From industry sales figures, we determined that there are approximately 0.5 trillion copies of sound recordings in existence, with approximately 24 billion new copies being added every year. These figures alerted us to the fact that tracking legacy music files is at least as important as tracking new music files.
- The system should be invariant to non-malicious manipulations: The system should be able to identify sound recordings that have been subjected to a variety of common manipulations such as compression at different bit rates, volume normalization, frequency equalization, etc., even if these manipulations render the content altered from the original.

- The system should have high accuracy: Accuracy is a function of low false negatives (provision of no information) and low false positives (provision of wrong information). Potential customers informed us that false negatives are better tolerated than false positives. Based on this, we decided that the system should have a maximum false positive rate of 5%.
- The system should have high throughput: One of the potential uses for DRM is for creating a legal P2P service.[1] Using throughput figures from operational P2P services, we determined that the system should handle 200 million downloads per day, or approximately, 2500 requests for identification per second.
- The system should be operable on music devices: There is a demand for music devices, such as MP3 players and car and home stereos, that typically have limited computational power and storage capacity, and with intermittent or no connectivity to the internet. For effective operation on such devices, the identification method used in the DRM system must have high computational efficiency and a small footprint. Additionally, the system database of reference identifiers must be small enough to fit on the device.

There were several technologies that we could have used for content identification. The most popular ones are listed in Table 1, along with their advantages and limitations. From these, we chose audio fingerprinting because it was the only technology that met all of our functional objectives without requiring industry standardization or major changes in consumer behavior. In the following sections, we discuss audio fingerprinting in general and provide details about the MusicDNA audio fingerprinting system we designed.

2 Audio fingerprinting technology basics

In this section, we provide a brief review of audio fingerprinting technology. Systems using this technology are commercially available from Relatable, Audible Magic, Auditude, MusicReporter

[1] For example, the P2P music service might allow only selective sharing of its offerings. This would require identifying tracks that were allowed (white list) or tracks that were not (black list).

and Gracenote, among others. When designing a content identification method based on audio fingerprinting, there are two main issues to consider: fingerprint generation and fingerprint lookup. We will discuss both these issues in this section.

2.1 Fingerprint generation

For maximum effectiveness, an audio fingerprint must be reasonably small, robust to distortion, rich in information, and computationally simple. Additionally, it must be designed for the express purpose of identifying sound recordings[2].

Techniques to create audio fingerprints fall into two broad categories. The first category includes approaches that use descriptive attributes (loudness, tempo, beat, melody [15]), and their derivatives (pitch, rhythm structure, brightness [15]). The second category includes approaches that are based on more intrinsic attributes of a recording with no explicitly identifiable descriptive qualities. We found that attributes in the first category were better suited to classification, while attributes in the second category were much more effective for identification. For the remainder of this section, we focus on techniques that use attributes belonging to the second category.

There are several methods [1-19] to compute fingerprints using intrinsic attributes of a recording. Almost all the methods involve computing features from the time-frequency spectra of a recording. While each is distinctive, all methods follow the broad framework described below:

1. Time partitioning: The recording is segmented into short time frames (often overlapping for robustness). This allows the computation of components of the fingerprint as soon as a frame of samples is collected without having to load the entire recording. Depending on the

[2] This distinction is key: for identification purposes, the representation of the recording should make it easy to distinguish it from *any* other recording; for classification purposes (clustering recordings according to genre, mood, etc.), the representation should not distinguish it from similar-sounding recordings.

application, either only a short section (10-45s) from a known or random location, or the entire recording is used. Frame sample amplitudes may be weighted using models from perceptual coding that represent typical human auditory sensitivity patterns.

2. Frequency partitioning: Each frame is partitioned into frequency bands. The frame samples are transformed into frequency/scale space using the fast fourier transform (FFT), the discrete cosine transform (DCT), the wavelet transform (WT), etc. The frequency/scale space is then partitioned. A weight function based on psycho-acoustic hearing models may be applied to the frequency spectra to attenuate frequencies to which the human ear is not sensitive. The type of transform to use and the nature of the frequency partitioning (linear, log, dyadic, etc.) are determined by the performance goals, with particular attention paid to balancing the need for high identification accuracy with computational and speed requirements.
3. Fingerprint features computation: The time and frequency partitioning results in time-frequency (T-F) blocks from which fingerprint features are computed. Examples of features are spectral/wavelet residuals [6], LPC coefficients [2], and Mel-frequency cepstral coefficients [13]. Even fairly simple features such as those based on statistical moments [10],[13], measures of variation of energy across blocks [6],[14], and principal component analysis [19] of the matrix of the T-F block energies, are remarkably effective fingerprint candidates, provided one partitions the time-frequency space optimally. Typically, several different features are extracted and concatenated to form the fingerprint.

Ultimately, the specific technique used to obtain a fingerprint depends on the associated system goals. Once such a fingerprint is obtained, a search scheme for fingerprint lookup needs to be designed. In order to meet our system goals, we focused on extracting fingerprint features, (discussed in detail in Section 3.1) with low computational load, small footprint, reduced database storage requirements and high accuracy for moderate noise applications, and designing a lookup scheme that offered high throughput. We discuss this below.

2.2 Fingerprint lookup

Even the most discriminating and robust fingerprint is rendered ineffective if we cannot design a good search scheme to retrieve the best match to a trigger fingerprint from a large database of several millions in a short time. There can be achieved using either exact matching or fuzzy matching. Performing an exact match usually means using a direct table lookup approach, which requires that we obtain fingerprints that are invariant to compression and other common recording effects[3] - a task almost impossible to achieve. It is more realistic to generate fingerprints that have the property that the intra-song (different variants of the same recording) fingerprint variation is much smaller than the inter-song (different variants of different recordings) fingerprint variation. This suggests the possibility of adopting a "measure of closeness" to compute a match. Such a match is by definition, inexact or fuzzy, and requires computing this measure for every entry in the database to determine the best match. This process is not practical in large databases. Thus, the two main issues to solve in the fuzzy match scenario are:

a) Formulating an "intelligent" strategy to reduce the search space to a manageable size.

b) Determining an objective measure of match.

For purposes of speed and ease of implementation, we need a measure of match that is simple, yet effective. Common examples that fit this bill are correlation [10], the Itakura distance [20-21], the Manhattan (L_1) distance [6], and the Euclidean (L_2) distance [1,13]. Appropriately choosing the measure of match can greatly enhance the discriminating capability of the identification system.

As noted earlier, it is impractical to compute match measures for every fingerprint in a database of millions. We need to partition the entire search space into non-overlapping regions, isolating the target song (correct match) in a small set from which we can determine the best match using

[3] For example, this means that the fingerprint computed from a song on a CD must be exactly identical to the fingerprint of the same song computed from its MP3-encoded version.

the chosen match measure. A popular approach to partition a given space is *clustering*[4]. In our experiments with clustering using several candidate features, we found two serious problems:

a) Cluster assignment was very sensitive to the initial training set of vectors, and there were no clear guidelines as to the best method to pick the training set.
b) Addition of new data required retraining of all clusters causing cluster reassignment and a general shake-up of the existing database structure.

The pitfalls of using clustering led us to explore other options. Our explorations led to the design of an alternative approach called the *Search by Range Reduction* (SRR) technique [19]. The SRR is a highly effective, albeit simple, technique to search through massive databases in a reasonable time. It works on the principle of a successive pruning of the search space. We start with a search space containing all fingerprints in our database (stage 0). At stage J, the search space is reduced to all fingerprints in the database whose first J components are each within some distance of the first J components of the query fingerprint (FP_1, FP_2, ...FP_J). The process is continued until the search space is pruned to a size small enough to accommodate throughput requirements. The simple case of a 2-component fingerprint vector is shown in Figure 1. The chosen measure of match is then computed only for the fingerprints in the final pruned space. The SRR method uses a vector of ranges (T_1, T_2, ...T_N) for the N fingerprint components for pruning, and a threshold for the match computation to determine the final result. Thus, the method has the desirable property of disqualifying fingerprints in the search space that may be 'close' to the query using the measure of match, but differ greatly in shape from the query, as illustrated in Figure 2. We found the SRR approach to be robust, fast, effective and highly amenable to database scaling, thus satisfying our primary system goals. In the next section, we discuss the MusicDNA system and provide details of our implementation of the SRR approach.

[4] In this process, the entire space is partitioned into clusters, each of which contains a manageable number of entries that are "close" to each other using some chosen criterion. The query to be matched is deemed to belong to the cluster to which it is "closest" using the same criterion, and the best match is determined from all the entries in this cluster.

3 MusicDNA System Design

The MusicDNA system consists of the MusicDNA Client and the MusicDNA Server. The MusicDNA Client extracts the fingerprint, and the MusicDNA Server serves as a lookup engine that contains the database of fingerprints and the search algorithm parameters.

3.1 MusicDNA Client - Fingerprint Extraction

The primary purpose of the MusicDNA Client is to extract the fingerprint from an input recording. The fingerprint extraction algorithm it uses consists of two main stages: a *signal conditioning* stage and a *signal analysis* stage. The conditioning stage reads files in different formats corresponding to different codecs, bit rates and sampling frequencies, and transforms them into a stream of pulse code modulated (PCM) data representing a monaural analog waveform sampled at 11025 Hz. We chose this sampling frequency because it represented a good compromise between fingerprint quality and data size. It limits the bandwidth of the input signal to 5512 Hz, thus eliminating the high frequencies that are often more susceptible to noise, while still retaining enough signal information to allow accurate identification.

The analysis stage executes the core signal processing algorithms and extracts the attributes that make up the fingerprint. For speed and efficiency, we chose to analyze only a short section of the recording (15s) from a known location, for the fingerprint computation. First, the 15s section is divided into 12 frames of 3s each with a frame overlap of 1.9s. Next, histogram equalization is applied to the sample amplitudes in each frame to provide robustness to common signal manipulations such as volume normalization, followed by the DCT. The result is then partitioned into 15 non-overlapping frequency bands with the band edges corresponding to the first 15 bands used in the MP3 encoding scheme. We found the combination of the DCT and the MP3-bands frequency partitioning highly effective and efficient in terms of discrimination ability and

fingerprint size. From each block obtained from the time and frequency partitioning, the total block energy is computed, and a 15 x 12 matrix of energies is obtained corresponding to the 15 frequency bands and the 12 time frames. The process is illustrated in Figure 3. From this matrix, two vectors, each of length 15, are obtained. One vector is the square root of the mean energy across time for each frequency band. The second vector is the standard deviation across time of the RMS power in each frequency band [19]. The two vectors are normalized individually and concatenated to form a 30-component fingerprint. Our choice of fingerprint features was driven by our system goals – we chose these features because they were easy to compute, were effective at discriminating between millions of recordings using only 30 components, and were reasonably robust to moderate distortion.

3.2 MusicDNA Server - Lookup Engine

The MusicDNA Server is our implementation of an audio fingerprint lookup system. Its core functionality includes the ability to accept a fingerprint identification request, quickly identify the song, return additional requested metadata or business rules, and log the transaction. In this section, we discuss the search process parameter tuning and the general architecture of the Server.

3.2.1 Tuning/Parameter Configuration

The search algorithm used in the MusicDNA Server to perform identification is the SRR technique described earlier, together with the Itakura distance (ID) measure, adapted to our needs[5]. From several competing measures, we chose the Itakura distance measure, despite its asymmetric nature, because it provided the best discrimination between different recordings for our analysis data sets which were carefully selected sets of fingerprint data that were representative of the range of inputs the Server was expected to process. Additionally, using the

[5] $\mathbf{ID}(\mathbf{FP^m}, \mathbf{FP^n}) = \log\left(\frac{1}{\mathbf{N}}\sum_{\mathbf{i}=1}^{\mathbf{N}}\mathbf{e_i}\right) - \frac{1}{\mathbf{N}}\sum_{\mathbf{i}=1}^{\mathbf{N}}\log(\mathbf{e_i}) = \log\left(\frac{\text{Arithmetic Mean}(\mathbf{e_i})}{\text{Geometric Mean}(\mathbf{e_i})}\right); \quad \mathbf{e_i} = \frac{\mathbf{FP_i^m}}{\mathbf{FP_i^n}}$

Itakura distance measure effectively neutralized the effect of having fingerprint component values that varied from each other by several orders of magnitude.

The factors that critically affect the efficiency and speed of the SRR search scheme are the order in which the fingerprint components are applied to prune the search space, the SRR range vector, and the Itakura distance cut-off threshold. We adjusted these parameters to optimize for false positive (finding a wrong match) and false negative (failure to find a match) errors, and for search speed. We ordered the fingerprint components so that the most discriminating ones were placed in the front of the vector. The discriminating capability of each fingerprint component was determined based on the reduction in search space size using only that component for a database of about 10 thousand songs. This ordering helped prune the search space more efficiently. The SRR range vector (T_1, T_2,...T_N) was determined based on the variation (as measured by the standard deviation) of each fingerprint component in our analysis data sets. To set the Itakura distance cut-off threshold, we did an extensive analysis of the distributions of the correct match (measure of intra-song distance) and the best non-match (measure of inter-song distance) on our data sets. This enabled us to gain insight into how large we could set the threshold, which would lower false negative errors, while still maintaining acceptable false positive error rates.

3.2.2 Server Architecture

The main function of the MusicDNA Server is to receive fingerprint requests and perform the fingerprint lookup in a short time. Since the need for high throughput is critical, we configured the Server to use an LRU (Least Recently Used) request cache for all fingerprint lookups. Industry representatives informed us that typically, 95% of search requests were for variants of a small set of songs. The Server stores fingerprint information for all individual variants of these songs and the corresponding song information in the LRU cache. Thus, the Server rarely needs

to perform an SRR search in its database for identification; in most cases, it successfully identifies the song via a simple table lookup in the cache, greatly increasing the throughput.

The Server processes all fingerprint requests via a MusicDNA Server HTTP Servlet interface, and returns its responses in XML. We chose HTTP as the transport interface because it is the most widely used protocol; it does not require any special firewall modifications for use, HTTP traffic is very difficult to block, and many third party tools are available, including proxy caches that may be leveraged in the future to aid in world wide deployment and 3 tier (web server/application server/database) scalable architecture components. We selected XML as the response format because it allows for flexible data type definition, and many third party tools are available. XML is an industry wide standard; parsers for many languages are readily available, making it easier for the system to work with different languages and platforms in the future.

4 MusicDNA System Performance

This section describes MusicDNA performance in terms of its efficacy (robustness and sensitivity) and throughput (extraction and lookup speed, and fingerprint footprint).

4.1 Efficacy

To achieve high efficacy, the fingerprint must minimize intra-song distance (distance to correct match in the reference database) while maximizing inter-song distance (distance to best non-match in the reference database). Figure 4 provides a visual aid for this evaluation. The curves shown are the cumulative distribution functions (CDFs) of the intra-song and inter-song distances of fingerprints taken from some of our analysis data sets (150 songs each). The reference database contained fingerprints of original songs ripped from CDs. We used such CDF plots to determine the best fingerprint candidates, the optimum distance measure and cut-off threshold. Observe that

robustness and sensitivity are related to the individual CDF slopes and the separation between the two CDFs. A good fingerprint should have a steep slope for the correct match CDF, and a gentle slope for the best non-match CDF, thus indicating less crowding of the fingerprint search space. It should also produce a large separation between the two CDFs, indicating greater sensitivity. The crowding effect associated with scaling can often lead to high false positive errors. To allow for good scaling performance, we must have large separation on small databases, so that even under scaling, there would be sufficient separation to keep error rates at acceptable levels. From extensive testing on small (150 songs) and large databases (million+ songs), we found that the CDF plots were reliable predictors of efficacy performance under scaling. If the two distributions overlapped or had little separation on small sets, identification errors were more likely to occur when the system was scaled up[6]. With this in mind, we performed several small-scale efficacy tests on different data sets and tuned our parameters based on the corresponding CDFs. The results after final parameter tuning are shown in Table 2. For comparison, the large-scale efficacy results using the same parameters on selected codec variants are shown in Table 3. Observe that while there is, as expected, a significant increase in false positive error rates as the database scales up, we met our error requirements even under large-scale testing conditions. The false negative error rates, which are not much affected by crowding, match well with the results in Table 2.

4.2 Throughput

To measure system throughput, we obtained benchmarks on the MusicDNA Server using an internally developed request generator. We obtained a mean fingerprint extraction time under 2s

[6] As database size increases, it becomes increasingly difficult to separate fingerprints effectively as they begin to *crowd* together. Features that work very well on small databases often fail to identify correctly when the size of the database increases by several orders of magnitude. In order to get realistic efficacy information, it is necessary to test performance of candidate fingerprints against massive databases, a process that can often be difficult and time-consuming. Thus, devising a method to predict large-scale performance from smaller data sets is very beneficial since testing on smaller data sets is far less time consuming. However, while we used the CDF plots to aid parameter tuning, all search parameters were finalized after verifying performance on representative database sizes (million+).

with a fingerprint footprint under 100 bytes, both well under our target limits. We achieved a throughput of 16 lookups/s with a mean lookup time of less then 300 ms running Microsoft SQLServer on a single CPU 864 MHz Dell Power Edge 4400 running Win2K. The middle tier was running IBM Websphere on a Sun E420 Solaris 2.74CPU 2GRAM machine, and was very lightly loaded. Only 2 CPUs were utilized. In order to meet the 2500 lookups/s requirement, we would need a maximum of 160 CPUs to handle the SRR load. This was well within our cost constraints for full system deployment. However, we expected the actual number of CPUs required to be significantly lower, depending on how heavily the LRU cache was used.

5 Conclusions

In this article, we discussed the need for a DRM system for music, and established that content identification was a necessary component for its successful operation. We reviewed common methodologies for content identification, with particular focus on audio fingerprinting technologies. We then discussed our experience in building a fast and accurate identification system using audio fingerprinting for environments with moderate noise[7]. Specifically, we elaborated on the process of defining system goals, developing the system design, architecting the system, and evaluating the system performance. The fast look-up, small footprint and database storage requirements, while maintaining high efficacy under scaling, are some key factors that distinguish our fingerprinting approach from existing fingerprinting technologies. We hope that this article served to provide insight into understanding why DRM systems for music are necessary, and to appreciate the challenges of designing effective DRM systems with the capability of efficiently handling millions of music recordings.

[7] For noisy applications such as streaming media, cell phones, etc. with modified/corrupted/incomplete files and no prior knowledge of sample location, [2-5, 8-9, 14] offer very robust audio fingerprinting solutions.

6 Acknowledgments

The authors gratefully acknowledge the contribution of Kwan Cheung in the design of the MusicDNA fingerprint and the SRR search technique. The authors also thank the reviewers for their comments and suggestions that helped to improve the readability of this article.

7 References

1. J. Laroche, “Process for identifying audio content,” WIPO Patent WO0188900A2, Creative Technology Ltd., 2001.
2. A. L. Wang and J. O. Smith III, “System and methods for recognizing sound and music signals in high noise and distortion,” US Patent US20020083060A1, Shazam Entertainment Ltd., 2002.
3. A. L. Wang and J. O. Smith III, “Method for search in an audio database,” WIPO Patent WO0211123A3, Shazam Entertainment Ltd., 2002.
4. W. Y. Conwell, B. A. Bradley and G. B. Rhoads, “Content identifiers triggering corresponding responses through collaborative processing,” US Patent US20020028000A1, 2002.
5. G. B. Rhoads and K. L. Levy, “Content identifiers triggering corresponding responses,” WIPO Patent WO02093823A1, Digimarc Corporation, 2002.
6. S. Ward and I. Richards, “System and method for acoustic fingerprinting,” US Patent US20020133499A1, 2002.
7. J. Herre, E. Allamanche, O. Hellmuth, T. Kastner and M. Cremer, “Method and device for producing a fingerprint and method and device for identifying an audio signal,” WIPO Patent WO03007185A1, Fraunhofer IIS-A, Germany, 2003.

8. W. D. Moon, R. J. Weiner, R. A. Hansen and R. N. Linde, “Broadcast signal identification system,” US Patent US3919479, The First National Bank of Boston, 1975.
9. J. G. Lert Jr., P. W. Lert and J. F. Cornelius, “Broadcast program identification method and system,” US Patent US4230990, 1980.
10. S. C. Kenyon, “Signal recognition system and method,” US Patent US5210820, Broadcast Data Systems Limited Partnership, 1993.
11. S. C. Kenyon, L. J. Simkins, L. R. Brown and R. Sebastian, “Broadcast signal recognition system and method,” US Patent US4450531, Ensco, Inc., 1984.
12. J. T. Foote, “Content-Based Retrieval of Music and Audio,” Proc. of SPIE, Multimedia Storage and Archiving Systems II, Vol. 3229, pp 138-147, 1997.
13. T. L. Blum, D. F. Keislar, J. A. Wheaton, E. H. Wold, “Method and article of manufacture for content-based analysis, storage, retrieval, and segmentation of audio information,” US Patent US5918223, Muscle Fish, 1999.
14. J. Haitsma, T. Kalker, and J. Oostveen, “Robust Audio Hashing for Content Identification,” Proceedings of the Content-Based Multimedia Indexing, 2001.
15. E. Wold, T. Blum, D. Keislar, and J. Wheaton, "Content-Based Classification, Search, and Rerieval of Audio," IEEE Multimedia Magazine, Fall 1996.
16. R. Gonzalez and K. Melih, "Content Based Retrieval of Audio," Proc. ATNAC ’96, 1996.
17. R. G. Lamb, A. M. Economos, E. F. Mazer, “Method and apparatus for recognizing broadcast information using multi-frequency magnitude detection,” US Patent US5437050, 1995.
18. C. Papaodysseus, G. Roussopoulos, D. Fragoulis, T Panagopoulos and C. Alexiou, “A new approach to the automatic recognition of musical recordings,” Journal of Audio Engineering Society, Vol. 49 (1/2), 2001.
19. M. J. Wells, V. Venkatachalam, L. Cazzanti, K. F. Cheung, N. Dhillon, S. Sukittanon, “Automatic identification of sound recordings,” WIPO Patent WO03009277A2, Gracenote Inc., 2002.

20. A. H. Gray, Jr., J. D. Markel, “Distance Measures for Speech Processing,” IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-24, No. 5, October 1976.
21. F. Itakura, “Minimum Prediction Residual Principle Applied to Speech Recognition,” IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-23, No. 1, February 1975.

8 Figures and Tables

Table 1: Technological alternatives for identifying digital music

Method	Description	Advantages	Disadvantages
Tagging	Embeds or attaches a textual description	i) Easy to attach	i) Easy to remove
Hashing	Creates a hash key based on the digital qualities of the music file; e.g., Secure Hash Algorithm (SHA)	i) Fast and easy to compute ii) Can use exact matching algorithms to perform searches iii) Can show if the file has been altered	i) Different formats of a song will produce different hash keys, and so the size of the database to incorporate all variants of all songs would be very large
Watermarks	Places an inaudible and indelible signal in the music	i) Can include business rules for sharing ii) Rules-of-use can be imposed by non-networked players and devices iii) Resistant to noise and other non-malicious manipulations	i) Inapplicable to legacy content ii) Neither inaudible nor indelible iii) Requires standardization iv) Susceptible to hacking v) Consumer resistance to purchasing hardware that does less than before
Encryption	Uses tags or watermarks for identification, and in addition uses techniques to make the music unusable without possession of a special code or key	i) All of the advantages of tagging and watermarking ii) Locks up music iii) Complex rules of use can be associated with individual songs	i) Does not work for legacy content ii) Requires industry standardization iii) Consumer resistance to purchasing hardware that does less than before
Audio fingerprinting	Uses the inherent qualities of the music to uniquely identify it by comparing it against a database of known music	i) Works for legacy content ii) Has no impact on sound quality (no additions) iii) Does not require industry standardization iv) Compatible with other methods of protecting music v) Completely transparent to the consumer	i) Can be computationally intensive ii) Database can be large for some implementations

Table 2: Small-scale test results

Format/Process	False Positive	False Negative
MP3 128 kbps	0%	0%
MP3 32 kbps	0%	3.36%
WMA 64kbps	0%	0%
Time shift (0.5 s)	0%	0.84%
Volume normalization	0%	2.54%
Pitch invariant speed-up by 6%	0%	0.84%

Table 3: Large-scale test results

DB Size	217,000		527,000		1,042,000	
Format	False Positive	False Negative	False Positive	False Negative	False Positive	False Negative
MP3 128 kbps (Blade, Lame encoders)	1.53%	0.42%	2.11%	0.4%	2.68%	0.32%
MP3 32 kbps (Blade, Lame encoders)	1.96%	2.58%	2.69%	2.50%	3.40%	2.33%

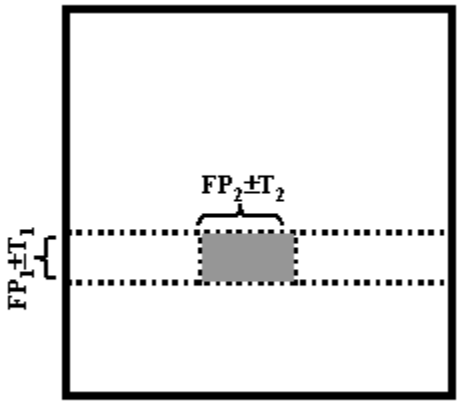

Figure 1: Square represents the space of all fingerprints for N = 2. First fingerprint component FP_1 with its range T_1 defines the initial pruning of the whole fingerprint space (horizontal dotted lines). Second fingerprint component FP_2 with its range T_2 defines the second pruning, (vertical dotted lines). Shaded gray rectangular area defines the space of fingerprints for which distance measures are computed.

Figure 2: Thick line represents 7-component fingerprint (each dot represents a component of the fingerprint vector). All fingerprints in the pruned space lie within the space defined by the dotted lines. Thus, the actual shape of the query fingerprint determines the fingerprints included in the pruned search space.

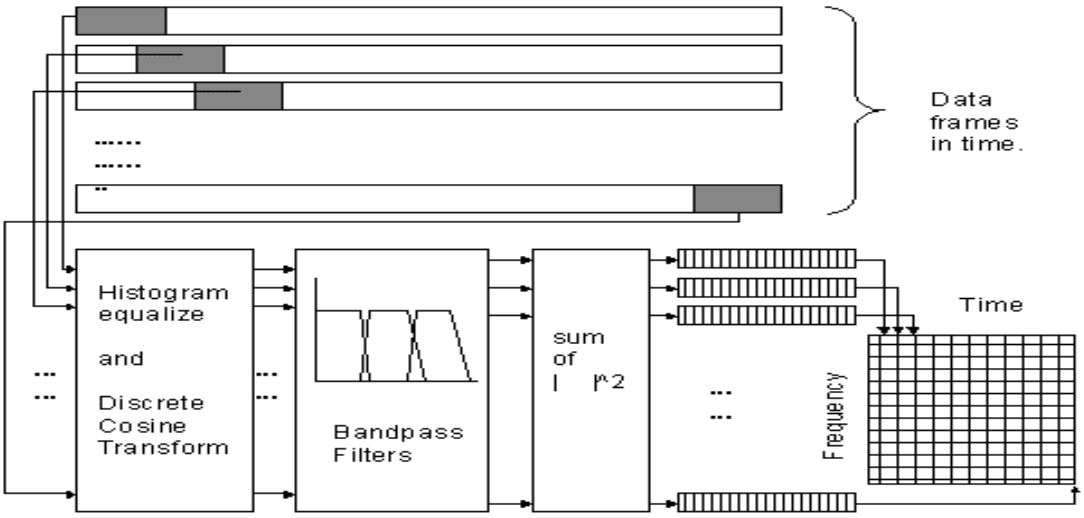

Figure 3: The analysis stage of the fingerprint extraction algorithm. The first 15 seconds of the recording are processed in 12 overlapping frames of 3 seconds each. Each frame is histogram equalized, and transformed via the DCT. The result is divided into 15 frequency bands, and the energy values are computed for each band, to form the Time-Frequency matrix. The matrix elements are processed as described in the paper to obtain the features that make up the fingerprint.

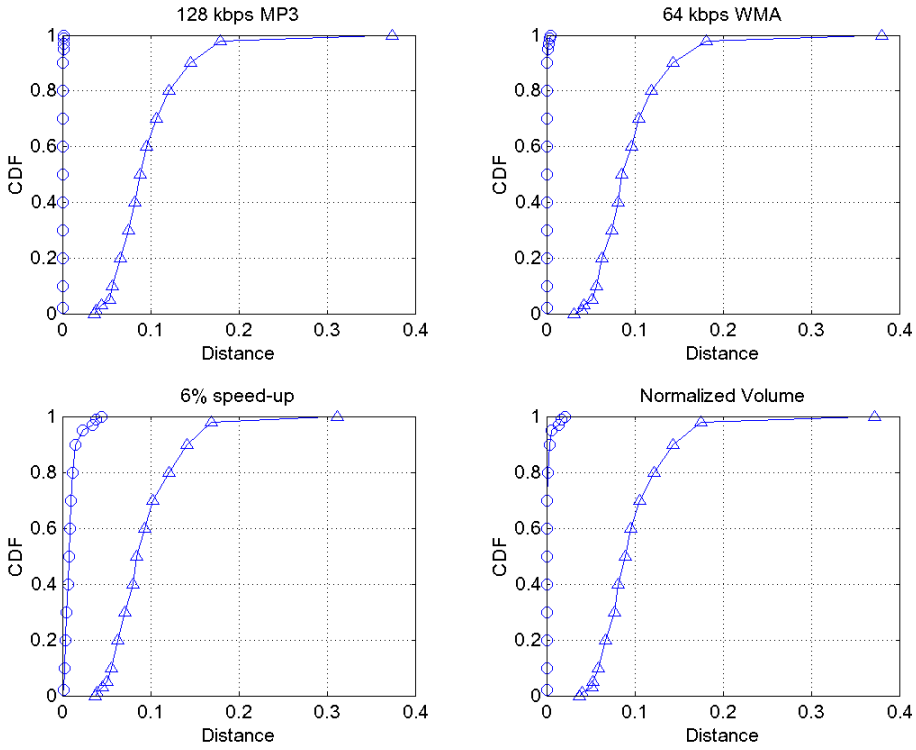

Figure 4: Cumulative distribution function (CDF) curves of correct match and best non-match song distances for various encoders and effects on sets of 150 songs each. The circles represent the correct match curve, and the triangles the best non-match curve. The clear separation between the two curves demonstrates the discriminating capability of the fingerprint, and indicates that the underlying fingerprint will hold up well under scaling.