A SCALABLE AUDIO FINGERPRINT METHOD WITH ROBUSTNESS TO PITCH-SHIFTING

Sébastien Fenet, Gaël Richard, Yves Grenier
Institut TELECOM, TELECOM ParisTech, CNRS-LTCI
37 rue Dareau, 75014 Paris, France
{sebastien.fenet, gael.richard, yves.grenier}@telecom-paristech.fr

ABSTRACT

Audio fingerprint techniques should be robust to a variety of distortions due to noisy transmission channels or specific sound processing. Although most of nowadays techniques are robust to the majority of them, the quasi-systematic use of a spectral representation makes them possibly sensitive to pitch-shifting. This distortion indeed induces a modification of the spectral content of the signal. In this paper, we propose a novel fingerprint technique, relying on a hashing technique coupled with a CQT-based fingerprint, with a strong robustness to pitch-shifting. Furthermore, we have associated this method with an efficient post-processing for the removal of false alarms. We also present the adaptation of a database pruning technique to our specific context. We have evaluated our approach on a real-life broadcast monitoring scenario. The analyzed data consisted of 120 hours of real radio broadcast (thus containing all the distortions that would be found in an industrial context). The reference database consisted of 30.000 songs. Our method, thanks to its increased robustness to pitch-shifting, shows an excellent detection score.

1. INTRODUCTION

Audio identification consists of retrieving the meta data associated with an unknown audio excerpt. The typical use case is the music identification service which is nowadays available on numerous mobile phones. The user captures an audio excerpt with his mobile phone microphone and the service returns metadata such as the title of the song, the artist, the album... Other applications include jingle detection, broadcast monitoring for statistical purposes or for copyright control (see [1] for more details).

Audio fingerprint is the most common way of performing audio identification when no meta data has been embedded in the unknown audio excerpt. It consists of extracting from each audio reference a compact representation (the fingerprint) which is then stored in a database. When identifying an unknown excerpt, its fingerprint is calculated. Then the best match with the unknown fingerprint is looked for in the database. The difficulty is dual. First, the captured signal has undergone a series of distortions (equalization, conversion, time-stretching, pitch-shifting, reverberation, ...). Second, the algorithm has to manage a database containing huge amounts of audio references.

Audio fingerprint has been dealt with in many previous works. Two main trends can be observed: exact-hashing and approximate-search. Exact hashing algorithms [2, 3] state that there are features in the signal which are preserved against the distortions. They extract these features and use a hash table to do the matching. Approximate search algorithms [4, 5] decode the unknown excerpt on a given alphabet and look for the closest transcription in the database. A variant is proposed in [6] where the unknown excerpt is decoded on different alphabets according to the references. The best-suited (with respect to the unknown excerpt) alphabet gives the closest reference.

In this work, we propose a novel audio fingerprint method based on hashing with a particular focus on robustness to pitch-shifting. Indeed, this distortion appears to be quite common in radio broadcasts and taking it into account allows us to show excellent results on a radio-monitoring oriented evaluation.

The paper is organized as follows. In the first section, we describe the broadcast monitoring use case. It is a typical application for fingerprinting that constitutes a demanding evaluation framework for the algorithms. It includes a wide variety of distortions that are actually performed by the radio stations. The whole methodology described in this paper can however be easily transposed to any other use case. In

THIS WORK WAS ACHIEVED AS PART OF THE QUAERO PROGRAMME, FUNDED BY OSEO, FRENCH STATE AGENCY FOR INNOVATION.

the second section, we describe in detail our method for fingerprinting. This includes the fingerprint model, the search strategy and the post-processing designed to prevent false alarms. We also describe an optional step of database pruning allowing a lower computation time while keeping a high ratio of identification. In the last section we show the results of experiments performed on real broadcast data.

2. BROADCAST MONITORING

2.1 Use case description

The task consists of detecting the broadcasting of any audio reference of a given database in an audio stream. Practically the database will be a set of songs and the stream will be the one of a radio station. We have to note that the broadcast stream not only contains references but also non-referenced items (such as advertisements, speech, unreferenced songs). Also the broadcast references have undergone a series of processes applied by the radio station, such as: compression, equalization, enhancement, stereo widening, pitch-shifting, ... (see [4] for more details about the radio stations processing). If we denote by $m_1, m_2, ..., m_N$ the references, by $\tilde{m}_1, \tilde{m}_2, ..., \tilde{m}_N$ their broadcast (and distorted) versions and by n the rest of the broadcast (considered as noise for the algorithm), the task can be illustrated as in Figure 1.

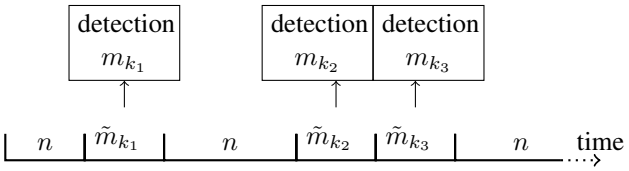

Figure 1: Broadcast monitoring

2.2 Focus on pitch shifting

The large majority of the methods from the state of the art rely on a spectral representation of the signal. Therefore these methods are possibly sensitive to modifications of the frequency content [7].

A very common distortion in the radio broadcasts is pitch-shifting. When this distortion occurs, all the frequencies in the spectrum are multiplied by a factor K. Pitch-shifting could be generated on its own by some signal processing on the frequency content. But in the context of the radio broadcasts, it is strongly linked with time-stretching. Indeed, the radio stations frequently shorten the music they play. To this end, most radio sound engineers will simply accelerate the reading of the music (by changing the sampling rate). This will change the duration of the music, but will also cause pitch-shifting as a side effect. This processing allows the stations to precisely fit their time constraints and to give the impression that the music is more lively in their broadcasts.

3. SYSTEM OVERVIEW

3.1 Architecture

As shown in Figure 2, the system is made of four units. First, the audio stream is cut in *analysis frames* of length l_a with an overlap o_a. The fingerprint of each analysis frame (called frame-based fingerprint) is calculated according to the methodology described in section 3.2. The *matching* unit then finds in the database the best match to the frame-based fingerprint. Finally, the best match is post-processed in order to discriminate out-of-base queries (when the audio stream corresponds to none of the references).

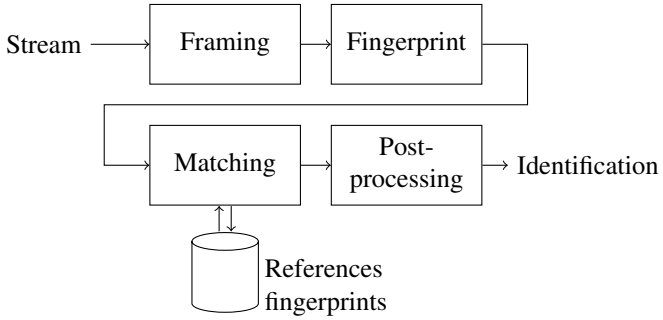

Figure 2: Architecture of the system

3.2 Fingerprint

Our fingerprint relies on a spectrogram calculated with "constant Q transforms (CQT)" [8] [9]. The constant Q transform is well adapted to musical signals in the sense that its frequency bins are geometrically spaced. As the notes of the western scale are geometrically spaced as well, this transform yields a constant number of bins per note. Moreover pitch-shifting becomes a translation in the CQT domain. That is, a frequency which is located in bin b will have its pitch-shifted version located in bin $b + K'$. In our implementation, we use a CQT with 3 bins per note performed on frames of signal with a 10ms increment.

In order to compact the spectrogram, we use a 2 dimensional peak-picking inspired by [2]. We tile the spectrogram with rectangles of width ΔT seconds and height ΔB bins of frequency (typical values for ΔT and ΔB are $\Delta T = 0.4$s, $\Delta B = 12$bins). In each rectangle, we set the maximum point to 1 and all the other points to 0. The result is a binary spectrogram containing sparse points at 1. They correspond to the points with the highest energy in the original spectrogram.

The methodology used ensures that there is one point set to 1 per rectangle of size $\Delta T \times \Delta B$ (2-dimensional homogeneity). Thus, this representation is robust to compressors (which change the dynamic of the audio with respect to time) and equalizers (which change the dynamic of the audio with respect to frequency). Furthermore, the fact that we do only keep points with maximum energy makes the representation robust to most additive noises.

3.3 Indexing the references

As we are dealing with an exact-hashing approach, the matching step relies on the indexing of the references. As Wang suggests, we use pairs of peaks (points set to 1 in the fingerprint step) to index the fingerprints of the references. We will first describe how to encode a pair of peaks. Then we will describe the hash function.

t_1 and t_2 being the times of occurrence of the two peaks involved in a pair, b_1 and b_2 being their frequency bins, the encoding we suggest for a pair of peaks is the following:

$$[\,\widehat{b_1}\,;\, b_2 - b_1\,;\, t_2 - t_1\,]$$

with $\widehat{b_1} = \lfloor \frac{b_1}{6} \rfloor$, a sub-resolved version of b_1. The first component ($\widehat{b_1}$) is a rough frequency location of the pair of peaks. The second component ($b_2 - b_1$) is the spectral extent of the pair in the CQT domain. The third component ($t_2 - t_1$) is its time extent. This encoding has several advantages. As it only takes into account relative time information, it is robust to cropping. Also, it is robust to pitch-shifting. Indeed the use of the constant Q transform implies the pitch-shifting invariance for the second component: a reference having peaks at frequency bins b_1 and b_2 will have them at frequencies $b_1 + K'$ and $b_2 + K'$ in its pitch-shifted version. And we actually have:

$$(b_2 + K') - (b_1 + K') = b_2 - b_1 \qquad (1)$$

The first component ($\widehat{b_1}$) is chosen on a sufficiently coarse representation (bin resolution divided by 6) to make it invariant with the common pitch-shifting ratios ($\leq 5\%$). It is worth mentioning that pitch-shifting will still move some pairs close to the border of one sub-resolved bin to the next. However, similarly to Wang's methodology, an exact matching of all pairs is not required. Indeed, the histogram step described thereafter only requires that the majority of the pairs are preserved.

As for the hash function, we build an index over all the pairs of peaks of all the references. More precisely, we build a function h_1 which, for any pair of peaks p returns all the references containing this pair with the time of occurrence of p in the references.

$$h_1 : p \longmapsto \{(m_i, t_{p,m_i}) / \text{ p occurs in } m_i \text{ at } t_{p,m_i}\} \qquad (2)$$

Let us note that in order to prevent an explosion of the number of pairs, we only consider pairs of peaks whose spectral extent is smaller than a threshold Δb_{max} and whose temporal extent is smaller than a threshold Δt_{max} (typical setup for this limitation is $\Delta t_{max} = 1.2$s and $\Delta b_{max} = 24$bins).

3.4 Matching

When identifying the fingerprint of an analysis frame, we extract all its pairs of peaks with their times of occurrence $\{(p, t_{p,af})\}$. Thanks to the hash function h_1 we can efficiently compute the differences $\{t_{p,m_i} - t_{p,af}\}$ for all pairs of the frame-based fingerprint and for each reference m_i . We store these differences in histograms (one histogram per reference).

If the analysis frame is actually an excerpt of the reference m_0 starting at time s, the m_0 histogram will show a maximum at value s. Moreover this maximum should be higher than any other histogram maximum. Indeed if the analysis frame corresponds to m_0 its fingerprint will have more pairs in common with m_0's fingerprint than with any other reference fingerprint. Furthermore, the pairs should all occur in the frame-based fingerprint s seconds earlier than in the reference's. Thus the histogram should show a majority accumulation for this reference at this value.

So, in order to perform the identification we look for the reference whose histogram has the highest maximum. This reference is considered to match the analysis frame. The argument of the maximum of the histogram gives the start time of the analysis frame in the reference.

3.5 Post-processing

For any analysis frame, the matching unit returns its best match among the references. This means that the case of an out-of-base query is not managed.

A simple approach would consist of setting a *threshold* on the common number of pairs between the frame-based fingerprint and its best match. If the frame-based fingerprint has more than *threshold* pairs in common with the best match, we deduce that the identification is correct. Otherwise we deduce that this is an out-of-base query. Unfortunately, on real data with classical distortions such a threshold is virtually impossible to setup. It happens that, due to the distortions applied to the stream, a best match has a low number of pairs in common with the frame-based fingerprint even though it is a correct identification. Besides, such a *threshold* would depend on the transmission channel and would have to be tuned for each different use case.

This is why we propose a post-processing unit based on a majority vote. The unit considers P successive analysis frames $\{a_j\}_{j=1..P}$ and their matching results (m_j, s_j). If among these P identifications, more than T_{vote} of them are

coherent the best match is considered to be a correct identification. Otherwise, it is an out-of-base query. Two matching results $(m_i, \Delta t_i)$ and $(m_j, \Delta t_j)$ of the i^{th} and the j^{th} analysis frames are coherent if:

$$\begin{cases} m_i = m_j \\ s_i - i.l_a.(1 - o_a) = s_j - j.l_a.(1 - o_a) \end{cases} \quad (3)$$

T_{vote} can take any integer value between 0 and P. A small value for T_{vote} will increase the risk of false alarms whereas a high value for T_{vote} will increase the risk of missed detections. In practice, a reasonable value for T_{vote} is:

$$T_{vote} = \left\lceil \frac{P}{2} \right\rceil \quad (4)$$

3.6 Database pruning

We propose an optional step meant to decrease the complexity of the overall processing. First, we define a simplified hashing function which, for each pair of spectral peaks, returns only the references possessing that pair.

$$h_2 : p \longmapsto \{m_i / \, p \text{ occurs in } m_i\} \quad (5)$$

N being the total number of references, we define the significance of a spectral pair p by:

$$s(p) = \frac{N - \text{card}(h_2(p))}{N} \quad (6)$$

Basically a pair which appears in many references will not bring a lot of information during the identification process (and thus has a low significance). Furthermore, it will intervene in many reference histograms and will thus involve many calculations. On the other hand, a pair which points to a small number of references allows to converge more quickly towards the best match.

Pruning the database consists of, for a given threshold T_{prune}, erasing from the database all the pairs verifying $s(p) < T_{prune}$. When doing so, we suppose that for any reference there will be a sufficient number of pairs kept in order to ensure a correct identification. This, of course, depends on the statistical distribution of the pairs and on the selected threshold T_{prune}. We have experimentally verified that the use of a reasonable threshold leads to a significant complexity gain while keeping similar performances (see section 4.3.4).

4. EVALUATION

4.1 Framework

The evaluation framework used in this work is similar to the one developed in the European project OSEO-Quaero [1]. It is defined as follows. The audio stream is the broadcast of a radio station. As the corpus comes from real radio broadcasts, it potentially contains all the radio sound processing we described (see section 2). The references are 1 minute-long excerpts of songs. The broadcast stream has been manually annotated and can thus serve for direct evaluation. For each broadcast reference, the annotation states the identifier of the reference, its broadcast time and duration.

The task of the algorithm is to scan the broadcast and output a detection message whenever a song among the references occurs in the stream. The algorithm gives the identifier of the detected song as well as its occurrence time. If the detection time is comprised between the annotated start time and the annotated end time of one occurrence of the same song, we make this occurrence a *detected occurrence*. Let us note that multiple detection messages of the same occurrence will be counted only once. If the algorithm detects a song during an empty slot, or during a slot containing another song, we count one false alarm. We do not limit the counting of false alarms.

4.2 Comparative experiment

4.2.1 Objectives

We have compared three different algorithms according to the framework described above. The first one ("Wang") is our own implementation of Wang's method [2]. The second one ("I B&S") is the algorithm called IRCAM Bark & Sone in [10]. The last one ("SAF", for Scalable Audio Fingerprint method) is the method exposed in this article.

As far as our implementations are concerned (Wang and SAF), they both rely on the same architecture, as described in section 2. All the parameters which are not directly linked to the fingerprint (framing parameters and post-processing parameters) are the same for both algorithms. In other words, the two systems have the same architecture with the same parameters. Only the fingerprint model does differ.

4.2.2 Data

In this experiment, the stream is made of 7 days of the French radio RTL. The one minute long references are extracted from 7309 songs. The broadcast stream contains 459 occurrences of these references.

Let us note that it happens that a given version of a music title is in the references, whereas another version of the same title is broadcast. This typically happens when an artist is invited on a radio show and performs some of his titles live. In this case, even if the studio versions of the artist's titles are in the references, the algorithm is not required to match the studio version with the live performance. Indeed, the recognition of different interpretations of the same song is considered to be out of the scope of this work.

[1] http://quaero.org

4.2.3 Parameters

We have used 5s long analysis frames with a 50% overlap. The post-processing parameters have been set to $P = 12$ and $T_{vote} = 6$. This means that the detection is performed on 30s of signal, and requires that at least half of the matching during these 30s has given a coherent identification. Such parameters insure a very low rate of false alarms, which is required in many use-cases for audio-fingerprint.

4.2.4 Results

Algorithm	Detected occ. / Total nb	False Alarms
Wang [2]	381 / 459 (=83.0%)	0
I B&S [10]	445 / 459 (=96.9%)	2
SAF (proposed)	447 / 459 (=97.4%)	0

Table 1: Results of the comparative experiment

We can see in Table 1 that the detection ratio is much higher with our fingerprint than the original model of Wang. As far as we can tell, this really comes from the fact that a non-negligible number of broadcast songs are pitch-shifted. These results therefore show that, in addition to being robust to the same distortions as Wang's model, our fingerprint has an increased robustness to pitch-shifting. Besides, we can see that the post-processing plays its role very efficiently. It has prevented all the false alarms (in both algorithms Wang and SAF) and still has allowed a very high detection rate.

4.2.5 Runtime

We will give here some figures about the processing times of the algorithms. These figures are given on the basis of our Matlab® 64-bits implementations, running on an Intel® Core 2 Duo @ 3,16 GHz with 6MB of Cache and 8GB of RAM. We are aware that these figures give no absolute truth, since the processing times highly depend on the machines, the programming language and the optimization of the code. They nevertheless give an order of magnitude of the runtimes with such a configuration. Besides, they allow a comparison of the different algorithms since all running times are given on the same basis.

The algorithm "Wang" has a processing time of 0.08s per second of signal. The algorithm "SAF" has a processing time of 0.43 seconds per second of signal. The difference mainly comes from the extra time required for the calculation of the constant Q transform. If we apply the pruning technique described in section 3.6 with $T_{prune} = 0.5$, we obtain a speed-up factor of 35%. This reduces the processing time of the second algorithm to 0.28 seconds per second of signal with the exact same identification score.

4.3 Scaling experiment

4.3.1 Objectives

We have led a second experiment in order to validate the potential scalability of the system we propose. The framework is the same as in the previous experiment, but we now run the algorithm with a much larger references database.

4.3.2 Data

In this experiment the stream is made of 5 days of radio broadcast coming from 2 different French radio stations (RTL, Virgin Radio). The references set is much larger as it contains 30.000 songs.

4.3.3 Results

Algorithm	Detected occ. / Total nb.	False Alarms
SAF (proposed)	496 / 506 (=98.0%)	0

Table 2: Results of the scaling experiment (30.000 songs)

The results clearly show that the algorithm is scalable. It has achieved a detection performance which is comparable to its performance in the first experiment. Though, the references database is more than 4 times larger in this experiment. It is particularly noticeable that in spite of the enlargement of the database, the system has still not output any false alarm. The multiplication of the songs in the database had yet highly increased the risk of having close fingerprints for different songs.

As far as the detection performance is concerned, the results of this experiment show that the algorithm we propose has the ability to handle industrial sized databases.

4.3.4 Runtime

The basis for the following calculation time is the same as in section 4.3.4. With the 30.000 songs database, the algorithm (without pruning) runs at a speed of 1.44 seconds per second of signal. If we compare this running time with the one of the smaller scale experiment, we notice that the multiplication of the database size by 4 has lead to a multiplication of the processing time by 3,3. The increase of the running time is thus sub-linear with the number of references. We can also note that, even though the code has not been fully optimized, the algorithm almost runs in real-time.

5. CONCLUSION

In this article, we have proposed a new fingerprint model. We have included this fingerprint in a global architecture.

The overall system is able to process audio streams in accordance with a radio monitoring use-case. The fingerprint we propose is inspired by Wang's work [2] from which we have reproduced the indexing scheme based on pairs of spectral peaks. But our use of the constant Q transform and our proposition of a different encoding for pairs of peaks allows us to show a much increased robustness to pitch-shifting. This, in turn, greatly improves our identification results on real radio broadcasts, as it has been shown in the comparative experiment presented. As far as scalability is concerned, we presented a second experiment which is based on a 30.000 songs database. This proved that our system easily scales up, while keeping a high detection ratio and a reasonable calculation time. In the future, we will focus on the problem brought up in section 4.2.2. The annotations we used indeed contain an average 7% of live versions of titles stored in the references database in their studio versions. Matching the ones with the others is a problem that lies somewhere between audio fingerprint and cover song detection. It will be interesting to study an extend of the fingerprint system which would be able to do this matching. Such an extended system will probably need to integrate more semantically based information.

6. REFERENCES

[1] P. Cano, E. Batlle, E. Gomez, L. de C.T. Gomes, and M. Bonnet, "Audio Fingerprinting: Concepts and Applications," in *1st International Conference on Fuzzy Systems and Knowledge Discovery*, (Singapore), November 2002.

[2] A. Wang, "An Industrial-strength Audio Search Algorithm," in *ISMIR 2003, 4th Symposium Conference on Music Information Retrieval*, (Baltimore, Maryland, USA), pp. 7 – 13, October 2003.

[3] J. Haitsma, T. Kalker, and J. Oostveen, "Robust audio hashing for content identification," in *CBMI, Content-Based Multimedia Indexing*, (Brescia, Italy), September 2001.

[4] P. Cano, E. Battle, H. Mayer, and H. Neuschmied, "Robust Sound Modeling for Song Detection in Broadcast Audio," in *AES, 112th Audio Engineering Society Convention*, (Munich, Germany), p. 5531, May 2002.

[5] E. Weinstein and P. Moreno, "Music identification with weighted finite-state transducers," in *ICASSP '07, IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 2, (Honolulu, HI), pp. 689–692, April 2007.

[6] E. Allamanche, J. Herre, O. Hellmuth, B. Frba, T. Kastner, and M. Cremer, "Content-based Identification of Audio Material Using MPEG-7 Low Level Description," in *ISMIR 2001, 2nd International Symposium on Music Information Retrieval*, (Bloomington, Indiana, USA), October 2001.

[7] E. Dupraz and G. Richard, "Robust frequency-based audio fingerprinting," in *ICASSP 2010, IEEE International Conference on Acoustics, Speech and Signal Processing*, (Dallas,USA), pp. 2091–2094, March 2010.

[8] J. C. Brown, "Calculation of a constant Q spectral transform," *Journal of the Acoustical Society of America*, vol. 89, no. 1, pp. 425–434, 1991.

[9] J. C. Brown and M. S. Puckette, "An efficient algorithm for the calculation of a constant Q transform," *Journal of the Acoustical Society of America*, vol. 92, no. 5, pp. 2698–2701, 1992.

[10] M. Ramona and G. Peeters, "Audio Identification based on Spectral Modeling of Bark-bands Energy and Synchronization through Onset Detection," in *ICASSP 2011, IEEE International Conference on Acoustics, Speech and Signal Processing*, (Prague, Czech Republic), May 2011.