Noname manuscript No.
(will be inserted by the editor)

A score identification parallel system based on audio-to-score alignment

A.J. Muñoz-Montoro · R. Cortina · S. García-Galán · E.F. Combarro · J. Ranilla

Received: date / Accepted: date

Abstract This paper presents a parallel system for searching a digital score of classical music in a personal library. The application scenario of the system is for a musician who wants to search for a specific score in its own device by playing an excerpt of a few seconds of the composition. We propose a solution, based on audio-to-score alignment, which allows to identify the correct score in a database of musical pieces in real time. This is a challenging task because we focus on a real time system targeted for handheld devices characterized by both mobility and low power consumption. Experimental results show that it is possible to achieve real time execution in the tested scenarios using parallel computing techniques with ARM processors.

Keywords Score identification · Parallel computing · Real-time · Audio-to-score alignment · DTW · Audio processing

A.J. Muñoz-Montoro
Department of Telecommunication Engineering, Universidad de Jaén, Spain
E-mail: jmontoro@ujaen.es

R. Cortina
Department of Computer Science, Universidad de Oviedo, Spain
E-mail: raquel@uniovi.es

S. García-Galán
Department of Telecommunication Engineering, Universidad de Jaén, Spain
E-mail: sgalan@ujaen.es

E.F. Combarro
Department of Computer Science, Universidad de Oviedo, Spain
E-mail: efernandezca@uniovi.es

J. Ranilla
Department of Computer Science, Universidad de Oviedo, Spain
E-mail: ranilla@uniovi.es

1 Introduction

Paper sheet music is often a source of annoyance for musicians for several reasons. Music scores can be voluminous to store and uncomfortable to handle while performing. They can become deteriorated from repeated use and brittle with age. When a music score is placed on a music stand, it can be easily dislodged causing some very anxious moments for the performer. Furthermore, nothing is more frustrating than thumbing through paper sheet music trying to find the desired music. In this way, advancements in technology during the last decade have resulted in rapid growth of software for displaying digital scores. Examples include Beatik[1], which is a page-turning software that adjusts to the musician playing in real time; Antescofo[2], which is a musical accompaniment software for classical musicians; Tonara[3], which is a music tracking platform focusing on the amateur musicians; and Newzik[4], which is a digital score display system. All of these applications allow users to store digital scores in their personal libraries. However, as users use this kind of tools, their personal libraries can grow to the point that searching a specific digital score can become a tedious task.

This paper addresses the problem of *searching a digital score* of classical music in a personal library. Imagine a musician who wants to search for a specific score in its own device and decides to play an excerpt of a few seconds; the system should present the user with the title, the composer, and display the digital sheet music. Therefore, given a short audio query corresponding to a music interpretation, the goal is to identify in real-time the score on which the performance is based. To do this, the application keeps a database of scores in a symbolic format (e.g. MIDI) and decides which one best matches the input audio. This problem can be considered as a particular case of cover identification. In classical music, this is not an easy task, because different interpretations of the same piece often have considerable differences in terms of tempo, loudness and other important aspects. Moreover, the matching process must be as fast as possible, because the database may contain a large number of entries.

One popular approach used in music identification is *audio fingerprinting*. A typical example of this strategy is the well-known service Shazam[5] [24], which is mainly aimed at popular music. In audio fingerprinting, a set of fingerprint tokens is generated for each piece, each with a time stamp. These tokens are stored in a table, with a portion of the token acting as a hash key to provide a quick access to the token. When a query is presented, the system computes fingerprint tokens from the audio and extracts matching tokens from the database via the hash key. The piece that shares the largest continuous sequence of tokens with the query is returned as a result. This approach is very

[1] `https://beatik.com`
[2] `https://www.antescofo.com`
[3] `https://tonara.com`
[4] `https://newzik.com`
[5] `https://www.shazam.com`

fast and enables online applications with millions of pieces, but it requires that the input audio is almost an exact copy of the reference version stored in the database. In [3], a more flexible variation of the algorithm is proposed for score identification in classical music, where the tokens are constructed using triplets of note pitches. Unfortunately, the algorithm requires a music transcriber to obtain the audio tokens, which introduces some errors and limits the system to piano music.

Another method to address this task is based on *audio alignment* [11]. In this case, each score is represented as a sequence of audio-like features, obtained from a synthesized version of the score. Given a query, a full pairwise distance matrix of audio and score features is computed, and the alignment path through this matrix is obtained with Dynamic Time Warping (DTW). The process is repeated against each candidate score, choosing the score with the lowest alignment cost. The main drawback of this method is its highest computational intensity, despite being generally more flexible than fingerprinting approaches. Additionally, if the systems used by the musicians are devices characterized by both mobility and low power, such as smartphones or tablets, identifying in real-time the score on which the performance is based will be a greater complexity problem. For this reason, this strategy is preferred for offline applications, such as automatic indexing of audio or MIDI databases. Some efforts have been made to reduce the computation time. In [13], a novel indexing strategy is used to greatly reduce the cost. In [17], feature vectors are mapped to a space where the distortion is much more efficient, and a reduced number of vectors can be used.

In this paper, we propose a real-time score identification parallel system based on audio-to-score alignment. For this purpose, we decompose the problem into two main stages. Firstly, a feature extraction process from the audio signal is carried out to characterize some specific information about the musical content. Then, the alignment is performed over all the entries of the digital score database. The kernel of this proposal is based on our robust system proposed in [1,2], employing a fast spectral factorization algorithm and DTW.

Unlike the previous studies [1,2], where the audio-to-score alignment is used for tracking the reproduction of a musical piece over its corresponding digital score, here we address the problem of searching for the score that is more similar to a given musical interpretation among all those stored in a database. The challenge is clearly greater, not only in terms of computational complexity by having to align a greater number of MIDI scores in real-time, but also because a new mechanism is needed to measure similarity. From a signal processing point of view, the novelty lies in the development of a new model which uses the β-divergence to measure the similarity between an interpretation and a MIDI score. Consequently, this work proposes a completely different approach that incorporates new functional modules, and modifies others, to address the new formal and theoretical aspects. Accordingly, a new prototype is implemented using a mixed parallelism scheme to obtain high accelerations.

The paper shows a study of the behaviour of the application running on a NVIDIA Jetson AGX Xavier development kit. The AGX Xavier has a low-energy ARM processor with four power envelopes in seven different modes, allowing to simulate a wide range of mobile devices, from the least powerful to those with the greatest performance. This set of test combinations allows us to validate our system framework as a useful tool for solving the online score identification problem. According to the best of our knowledge, no holistic, flexible, free and cross-platform system that addresses this problem on systems-on-chips (SoC) with ARM architecture has been presented yet. As a proof of concept, some experiments are carried out on two different databases, showing reliable results.

The paper is organized as follows. In Section 2, we review related works on audio-to-score alignment. The proposed score identification system is described in Section 3. Section 4 details the parallel approach developed in this work. Experimental results are shown in Section 5, and conclusions are finally outlined in Section 6.

2 Background of the alignment system

Audio-to-score alignment is the task of synchronizing a musical recording with its corresponding score. This automatic alignment spares manual alignment which is very tedious since musicians usually interpret each piece in a personal way by introducing variations in the tempo, dynamics and/or articulation. This topic has been intensely researched since the mid 1980s. Traditionally, these systems consist in two steps: *feature extraction* and *alignment*.

Firstly, both the score and the audio signal are represented by two feature sequences that must be aligned: $U = (\mathbf{u}_1, \ldots, \mathbf{u}_n, \ldots, \mathbf{u}_N)$ and $V = (\mathbf{v}_1, \ldots, \mathbf{v}_t, \ldots, \mathbf{v}_T)$, where N represents the number of time instants in the score and T is the number of time instants in the audio. In the literature, several methods have been proposed to design these features in order to capture robustly the musical content and to be as much discriminative as possible between different musical situations. The most common approaches are chroma vectors [11], semigrams [8], decaying locally adaptive normalized chroma onset (DLNCO) [9], beat-tracking [16], peak structure distance (PSD) [23] or measures derived from analysis with Non-negative Matrix Factorization (NMF) [7]. The goal of the alignment stage is to find corresponding time points in the two features sequences (i.e. the corresponding position in the score for each point in time). For this purpose, a comparison measure for each pair from U and V is computed by the majority of the methods. Then, the best general alignment is found by employing stochastic (e.g. hidden Markov models) [19] or, more commonly, DTW [15].

Several approaches have addressed the online alignment task. Dixon [8] proposed to use a local cost function to compare pairs of audio and score segments

$$d(n,t) = f(\mathbf{u}_n, \mathbf{v}_t), \tag{1}$$

where $f(\cdot)$ is any type of feature comparison function, and $d(n,t)$ can be viewed as the cost of aligning $\mathbf{u}_n$ with $\mathbf{v}_t$. Then, an accumulated cost matrix is constructed by Eq. 1, containing the value of the minimum cost path up to (n,t). Given a specific time t, the result of Dixon's method is the point (n,t) with the minimum cost. In this implementation, only the neighboring cells around the current position are considered, thus saving processing time and memory consumption. The main drawback of the framework described in [8] is that the algorithm tends to get lost in the score, and is not able to recover due to the heavy constraints imposed on the path. Other modifications of the classic DTW algorithm have been proposed to reduce memory cost or increase speed. In [12], a short-time version of DTW is proposed, which performs the matching by dividing the sequences into shorter portions, providing the same result than standard DTW under some hypothesis. Unfortunately, the algorithm is not intended to work online, but only as a tool to reduce memory consumption.

Carabias et al. [5] proposed an online audio-to-score alignment where the feature extraction stage pre-processes the MIDI score to represent it in a suitable format. Firstly, the score information is analyzed to detect how many unique combinations of notes occur during the piece. Each unique combination of notes is called a *score unit*. These combinations take into account the instruments involved, such that two combinations having identical note numbers but different instruments are considered as two different score units. Observe that this approach is suitable for both monophonic and polyphonic interpretations, since a score unit can represent a single note or a chord compounded by many notes of a single instrument or multiple instruments. Moreover, the number of units will be often much smaller than the number of frames, since the music is repetitive and many combinations of notes usually span across multiple frames.

Once the score is arranged into score units, a single spectral pattern for each of them is learnt in the feature extraction module. For this purpose, a software synthesizer is used to convert the MIDI score into an audio file. The result is a synthetic low-quality signal which is converted to the time-frequency domain and decomposed using supervised NMF. As a result, a single spectral pattern is obtained for each score unit.

Each score feature $\mathbf{u}_n$ in the sequence U is the spectral pattern corresponding to the active unit in instant n. Concerning the audio signal, Carabias et al. proposed to extract a vector of features $\mathbf{v}_t$ per each frame acquired at time t. This feature vector $\mathbf{v}_t$ is computed from the input frame as its conversion to the frequency domain.

In the alignment module, the matching cost between $\mathbf{v}_t$ and every element in the score sequence U is computed for each input frame. The corresponding position in the score is determined from the accumulated costs given by an implementation of online DTW. In this approach, the cost measure between t and every instant in the score is given by the following distortion:

$$d(n,t) = D_\beta\left(g_{n,t}\mathbf{u}_n|\mathbf{v}_t\right), \tag{2}$$

where $D_\beta(\cdot)$ is the β-divergence function [6,10], $\beta \in [0,2]$, and $g_{n,t}$ is obtained by the following fast spectral decomposition equation [5]:

$$g_{n,t} = \frac{|\mathbf{v}_t\mathbf{u}_n^{(\beta-1)}|_1}{|\mathbf{u}_n^{\beta}|_1}. \tag{3}$$

In the field of music signal processing, the β-divergence has been widely used obtaining reliable results, since the parameter β essentially controls the assumed statistics of the observation noise and can be either fixed or learned from training data or by cross-validation.

In [2], we proposed a Real-time Musical Accompaniment System (ReMAS[6]) based on the online alignment system described in [5]. ReMAS was design to track the reproduction of a musical piece with the aim to match the score position to its symbolic representation on a digital sheet. ReMAS is a parallel and efficient system that was implemented and optimized for low-power processors, such as ARM processors, which are the heart of smartphones, laptops, tablets and other embedded systems.

3 Proposed score identification system

In this paper, we propose a parallel system to identify the digital score corresponding to a piece of classical music by analysing an audio excerpt of only a few seconds. In particular, we propose a framework, based on the audio-to-score alignment presented in [5], which we have called ShizMidi[7]. For this purpose, we have addressed major design changes with respect to ReMAS by incorporating new computational modules and proposing a new parallel approach to adapt it to this new problem. The software solution has been developed satisfying two basic requirements: real time and mobility. Therefore, this design should take the low computational power of the handheld devices, especially the cheapest ones, into consideration, and should deeply use the possibilities offered by parallel architectures.

The proposed algorithm outputs a ranked list of digital scores stored in a previously trained database. This list is sorted by the scores which are most similar to the audio excerpt captured by the microphone. To measure this similarity, we use the accumulated cost computed by DTW over the divergence matrix (see Eq. 2). In this way, the score corresponding to the audio performance will accumulate the lowest cost. Fig. 1 displays the full system. As it can be observed, the problem is decomposed in two main stages: *feature extraction* and *alignment*.

[6] `https://gitlab.com/SSPressing/ReMAS`
[7] `https://gitlab.com/SSPressing/shizmidi`

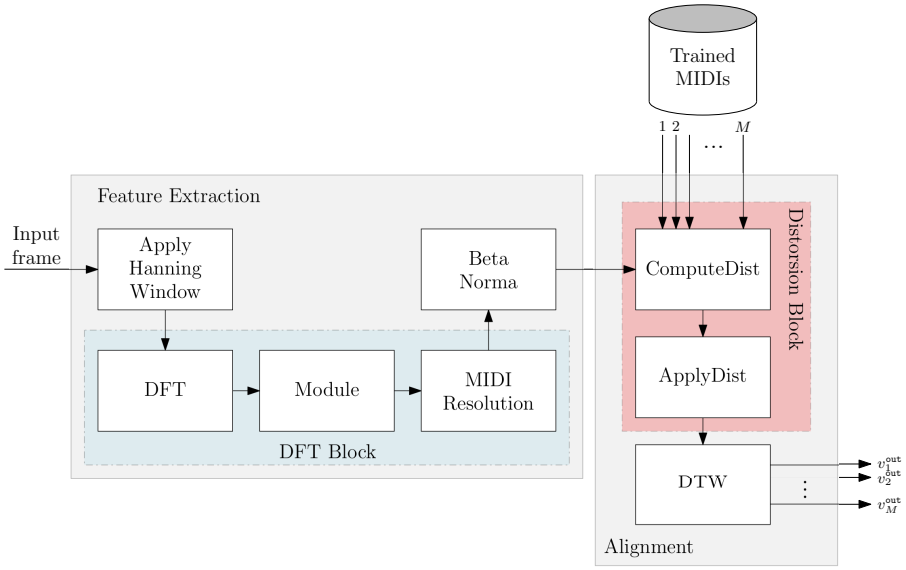


Fig. 1: Block diagram of the proposed system.

First, when a new frame arrives at time t, the process starts at the feature extraction stage. As in ReMAS [2], this stage extracts the features which characterize the musical content of the input audio frame. To do this, a low-level spectral representation of the audio data (time–frequency representation) is computed by a Hanning-windowed fast Fourier transform (DFT). Afterwards, the magnitude spectrum is computed from the complex output of the DFT and converted from linear frequency to MIDI resolution summing up the frequency bins belonging to the same MIDI interval. Note that the number of MIDI pitches corresponds with the range of notes that a piano can play in MIDI scale.

After the DFT block, we have implemented a new module, called Beta Norma (see Fig. 1). This module normalizes the magnitude spectrum of the input frame to the β–norm as

$$x_t'(f) = \frac{x_t(f)}{\sqrt[\beta]{\sum_f x_t(f)^\beta}} \tag{4}$$

where $x_t(f)$ is the magnitude spectrum computed in the DFT block. The used β-divergence value is 1.3 in line with other works in the state-of-the-art [4, 20, 14].

This normalization step is essential for the proper performance of the system, since, as explained in the previous section, the cost function used in Eq. 2 to measure the similarity between the audio frames and the score units defined in a MIDI score is the β-divergence. Therefore, when both the magnitude

spectrum of the audio frames and the spectral patterns of the score units are normalized to the β-norm, the cost function can only reach values defined in the range $[0, 1/\beta]$. Thus, if a frame has a strong similarity with a certain score unit, the β-divergence returns a value close to zero, and vice versa. Observe that, without this normalization process, the system would not be able to identify the target score in the database, since the accumulated cost for each score would depend on the energy of its spectral patterns and the audio signal; and, therefore, it could not be compared.

Once the information related to the musical content of the input frame has been extracted, the alignment process is performed over all the entries of the database. In this sense, the main function of this stage is to compute the alignment path with the minimum accumulated cost. For this purpose, the distortion block computes the β–divergence between the magnitude spectrum of the input frame and the spectral patterns of all the concurrent notes (score unit) which compound the score using Eq. 2. Note that a low value of β–divergence is obtained when the notes of a specific score unit match to the notes of the input audio. In this way, the score corresponding to the audio performance will obtain frame-by-frame lower divergence values than the other scores.

Then, DTW is used to compute the minimum value of the accumulated cost at frame t. From the local distances $d(n,t)$ computed in Eq. 2, when a new audio frame arrives, the accumulated cost matrix D is computed using the following recursion:

$$D(n,t) = \min_{c_n, c_t} \left\{ \begin{array}{c} D(n-1, t-c_t) + d(n,t)\sigma_{1,c_t} \\ \vdots \\ D(n-c_n, t-1) + d(n,t)\sigma_{c_n,1} \end{array} \right\} \tag{5}$$

where c_n and c_t are the step sizes at each dimension, and whose values are the integers in the range $c_n \in [1, C_n]$ and $c_t \in [1, C_t]$. Scalars C_n and C_t are then the maximum allowed step sizes in the score and in the performance, respectively. The weights σ control the bias toward diagonal steps. Unlike ReMAS, here we have set it to $\sigma_{x,y} = y$, so the path is biased towards horizontal steps (i.e. towards interpretation times). In this way, DTW is forced to progress through the interpretation time regardless of the score duration. This is an important point, since without this consideration, the system would penalize those scores with a long duration, which could accumulate a higher cost, compared to shorter duration scores. Observe that $D(n,t)$ is the accumulated cost matrix of the minimum-cost path from $(1,1)$ to (n,t), and that $D(1,1)$ is initialized to $d(1,1)$, because the alignment result is constrained to include the point $(1,1)$.

The accumulated cost matrix D is filled as new audio frames arrive to the system. At each time t, the corresponding minimum cost value is estimated directly from the information accumulated up to t, which can be considered as a sub-optimal solution. The algorithm simply returns the minimum value associated to the best path, that is, $v_{\texttt{out}} = \arg\min_n D(n,t)$.

Note that, for a specific frame, both the feature extraction stage and the alignment stage have to be run before the next frame arrives in order to reach the real-time requirement.

Finally, after analyzing all the frames related to the audio excerpt captured by the microphone, the system outputs a ranked score list based on the minimum cost value accumulated over the whole audio segment.

4 ShizMidi parallel design

As mentioned, ShizMidi is based on our system ReMAS, and it has been developed satisfying real time and mobility. In [2], a study of the theoretical computational complexity of ReMAS is carried out. Eq. 6 shows the complexity of the sequential version:

$$O\Big(T + F\log_2(F) + N_M U + S\Big) \tag{6}$$

where T is the frame length, F is the fast Fourier transform length, U is the number of score units, N_M is the number of notes in MIDI scale and S is the number of states. As in [21], we have used $F = 16,384$ bins, since this value is chosen to have enough frequency resolution for low frequency sounds, and T is fixed as 5,700 samples to have enough temporal resolution. We have selected $N_M = 114$ that corresponds to a range of notes of 9.5-octaves in one sample per semitone of MIDI resolution. On the other hand, U and S depend on the composition and are obtained from the reading of the MIDI score.

ReMAS is mainly conformed by loops with independent iterations which can be easily divided among CPU cores, excepting DTW, which includes a reduction operation, and the DFT, where an external optimal code is used. ReMAS parallel complexity is shown in Eq. 7,

$$O\left(\frac{T}{p} + F\log_2\left(\frac{F}{p}\right) + \frac{N_M U}{p} + \frac{p\log_2(p) + S}{p}\right) \tag{7}$$

where p is the number of processors or cores used.

However, the design of fine-grain parallelism of ReMAS is not suitable for ShizMidi. ReMAS was designed to track in real time the reproduction of musical pieces as long as possible. In that case, U and S are greater than T, F and N_M, as the number of score units and states grow as the duration of the composition increases.

Nevertheless, ShizMidi only needs to track a short excerpt of each MIDI score, as we will show in Section 5. Therefore, in this case the number of score units and states are smaller than T and F. In addition, the number of scores to be analyzed in the database is a new variable that affects the alignment stage (see Fig. 1). Under these conditions, keeping the design of fine-grain parallelism in the alignment stage would not allow to obtain high accelerations due to the high overhead resulting from the creation of a large number of low computational intensity processes.

Considering all these aspects, ShizMidi adapts a mixed parallelism scheme. On the one hand, it keeps the grain-fine for the feature extraction, which is applied only once per frame. On the other hand, coarse-grained parallelism is applied at the alignment stage, each score being a concurrent task. Thus, the parallel theoretical computational complexity of ShizMidi is shown in Eq. 8,

$$O\left(\begin{array}{l} \frac{T}{p} + F\log_2\left(\frac{F}{p}\right) + \frac{N_M}{p} \quad \textit{feature stage} \\ \\ \frac{D_S}{p_1}\left(\frac{N_M U}{p_2} + \frac{p_2\log_2(p_2)+S}{p_2}\right) \quad \textit{alignment stage} \end{array}\right) \tag{8}$$

where p is defined as $p = p_1 + p_2$, being $1 \leq p_1, p_2 \leq p$, and D_S is the number of analyzed MIDI scores from the database.

Regarding the alignment stage, ShizMidi can control the granularity by enabling/disabling nested parallelism. In the case of medium/large databases, p_2 is set to one and, therefore, the theoretical alignment stage complexity is $O\left(\frac{D_S}{p}\left(N_M U + S\right)\right)$. On the other hand, when D_S is small and U and S are large, p_1 is fixed to one and the complexity is $O\left(D_S\left(\frac{N_M U}{p} + \frac{p\log_2(p)+S}{p}\right)\right)$.

5 Evaluation and experimental results

In this section, we are going to analyze the results obtained in two different types of experiments. First, we have tested the reliability of our proposed system over the MusicNet database [22]. This dataset is a collection of 330 freely-licensed classical music recordings gathered from various musical archives. The database includes also their corresponding not-aligned MIDI scores, which are used to build our score database. This database features 11 different instruments arranged in small chamber ensembles, ranging from solos to octets, under various studio and microphone conditions.

The second experiment has been carried out on a synthetic database to analyze the performance of the application in terms of efficiency and speedup. In this regard, the number of digital scores stored in the database varies widely, from twenty-five to more than one thousand.

Regarding the used testbed, we have focused our interest on the NVIDIA Jetson AGX Xavier development kit, which is an embedded system-on-chip (SoC) with an eight-core ARM v8.2 CPU and a NVIDIA 512-core Volta GPU. It can operate at 2.26 GHz and runs a version of the Linux operating system especially tailored to this device. Xavier supports different kinds of running modes and it can be configured with the *NVPModel* command tool. This fact allows to simulate a wide range of mobile devices such as smartphones, laptops, tablets, and other embedded systems. Table 1 lists the details of all the configuration modes. We have also included for each mode the first equivalent ARM architecture with its market release year. As observed, four different power envelopes in seven different modes are defined. The power envelopes are n/a, 30W, 15W and 10W. Moreover, the possible number of running cores are

NVPModel Configuration				
Mode	No. CPUs	Power Budget (W)	Max. frequency (MHz)	1st Equiv. Architecture
0	8	n/a	2266	ARMv8 (2016)
1	2	10	1200	ARMv7 (2011)
2	4	15	1200	ARMv7 (2012)
3	8	30	1200	ARMv8 (2015)
4	6	30	1450	ARMv8 (2016)
5	4	30	1780	ARMv8 (2013)
6	2	30	2100	ARMv8 (2016)

Table 1: NVPModel mode definition.

2, 4, 6 and 8 with various CPU frequencies. Mode 1 is a special one, rebooting is required for changing to/from this mode to adjust the system services. For this reason, it has not been included in the experimentation.

In our first experiment, we have tested the reliability of our system and determined the audio segment needed to perform the score identification. In this way, for each audio file of the database, six segments of 3, 6, 9, 12, 15 and 180 seconds starting from the beginning of the file were selected and entered as a query into the system. Table 2 summarizes the obtained results in terms of accuracy. Here, we define accuracy as the percentage of searches in which the correct MIDI score is in the first position of the ranked list.

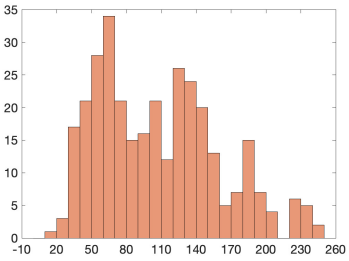
Time (s)	3	6	9	12	15	180
Accuracy (%)	57,58	82,12	91,21	92,73	94,85	98,48

Table 2: Accuracy results as a function of the duration of the analyzed audio segment.

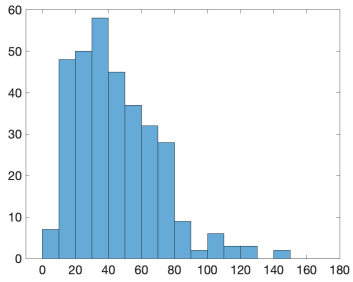
As shown, the accuracy of the score identification improves significantly as the length of the audio segment increases. For a query length of 15 seconds the correct score is detected in about 95% of the occasions, while a significant loss of accuracy occurs when queries of 3 seconds (57%) are employed instead. Note that the 100% of accuracy is never reached in this database because we have detected five MIDI scores badly annotated, so the maximum is approximately 98%.

According to the experiment, for segments longer than 15 seconds, the accuracy is always very similar, improving only a 3% for the case of 180 seconds. This is an important finding for our application, because it reveals that the matching measure does not improve with longer queries.

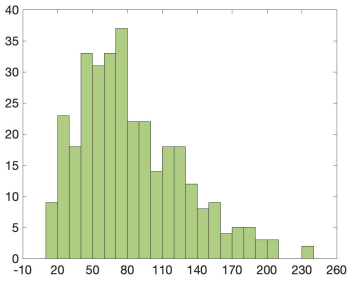
For the second experiment, we generated a synthetic database to test and explore the limits of the system. The complexity of the algorithm per frame depends on the number of scores to be analyzed. Therefore, in our tests we have varied the size of the database from 25 to 1500 scores. Moreover, the



(a) Tempo



(b) Number of score units

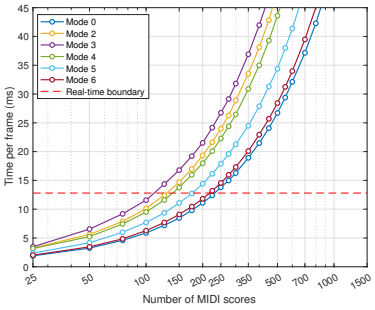


(c) Number of states

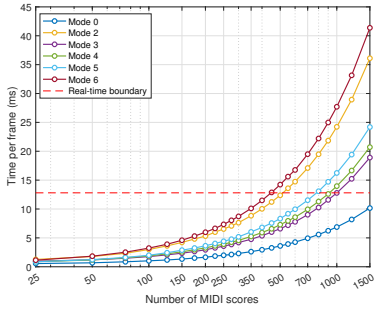
Fig. 2: Statistics about the MIDI parameters encrypted in the 330 MIDI files of the MusicNet database. The histogram in the top row shows the distribution of tempos and the histograms in the bottom row show the distributions of the score units and states across all MIDI files.

duration of audio file used to identify the score was set to 15 seconds, since this is the minimum duration to ensure an acceptable performance/the correct score identification , as previously seen.

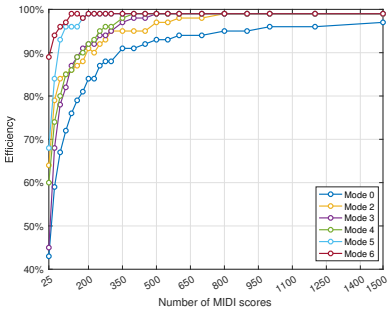
On the other hand, the complexity of the alignment stage per MIDI score mainly depends on the number of score units and states (see [1]). However, only the score units and states corresponding with the first 15 seconds are required to be analyzed. For a specific composition, the tempo annotated in the MIDI file determines the pace of the performance, and therefore, it has a strong relation with the notes played (i.e. score units) in a fraction of time. In [18], a statistical analysis of the information available in MIDI files was carried out over a huge database. This analysis demonstrated that the range of tempos most frequently used varies between 60 and 130 bpm. In order to determine the number of score units and states required for our synthetic database, we have carried out a similar statistic over the proposed MusicNet database. Fig. 2 displays the histogram of the distribution of tempos. As it can be observed, the range of tempos mainly varies between 60 and 130 bpm as in [18]. Regarding score units and states, the average value obtained is 40 and 80 respectively. Therefore, for our synthetic database we have used these average



(a) Execution times measured in milliseconds per frame for the sequential code.



(b) Execution times measured in milliseconds per frame for the parallel code.

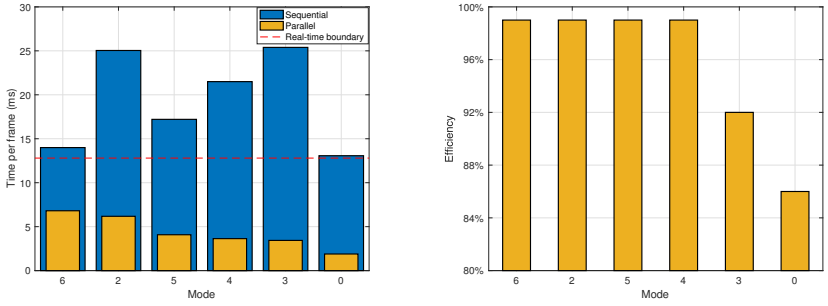


(c) Evolution of the efficiency.

Fig. 3: Experimental results as a function of the operating mode of the NVIDIA AGX Xavier and the number of MIDI scores stored in the synthetic database.

values plus the typical deviation (i.e. 69 and 128, respectively) to consider the worst-case scenario.

Experimental results obtained for the synthetic database are summarized by Fig. 3. These results are presented as a function of the operation mode of the NVIDIA AGX Xavier (see Table 1) and the number of MIDI scores stored in the database. Firstly, Fig. 3a shows the results of the sequential code in terms of time per frame in milliseconds. As it can be seen, the execution time increases as the number of scores in the database grows and the CPU frequency decreases. Note that the number of scores that can be analyzed in real-time oscillates between 100 and 225 for mode 3 (the one with lowest CPU maximal frequency) and mode 0 (the one with highest CPU maximal frequency), respectively. On the other hand, Fig. 3b outlines the execution time results of the parallel code. As expected, the fastest modes are those which use a greater number of cores, being the CPU frequency and the power budget only relevant when using the same number of cores. In this case, our system can deal with collections of different sizes, from small (400 MIDI scores) to large (more than 1500 MIDI scores). For devices with more than two available



(a) Execution times measured in milliseconds per frame.

(b) Efficiency.

Fig. 4: Experimental results of the MusicNet database as a function of the operating mode of the NVIDIA AGX Xavier.

cores and a high power consumption (i.e. mode 0, 3, 4 and 5), a large amount of MIDI scores can be analyzed (from 700 to more than 1500). Moreover, using our parallel approach allows to reduce the power consumption, respecting the use of battery, and address a collection of up to 500 scores in real-time (i.e. mode 2 is faster than mode 6 consuming half power). Regarding the efficiency of the system (see Fig. 3c), it is close to one in all modes when the number of MIDI scores grows. Therefore, we can assert that, when the number of processors and the size of the problem grow, our system scales correctly.

The obtained results for the MusicNet database can be seen in Fig. 4. In this case, real-time execution is achieved by the parallel approach for all the operation modes of the NVIDIA AGX Xavier. On the contrary, the sequential version does not reach real time in any case, even in mode 0, where more resources are used. Concerning the efficiency, it is very high, except for modes 3 and 6. MusicNet database is not large enough for these modes to reach the permanent regime for efficiency (see Fig. 3). Therefore, for this database it is enough to use the lowest performance modes, e.g., mode 2 (15W) and 6 (2 cores).

6 Conclusion

In this paper, we have proposed a score identification parallel system based on audio-to-score alignment. To the best of our knowledge, this is the first implementation in real time which addresses this problem. Our system has focused on achieving real time execution using handheld devices characterized by both mobility and low power consumption. We have decomposed the task into two main stages: a feature extraction stage, where the input audio signal is characterized; and an alignment stage, where all the entries of the digital score database are analyzed in parallel. The proposed system has been evaluated

using a synthetic and a real database. Experimental results show that reliable results for the score identification task can be achieved in real time.

Acknowledgements This work has been supported by the Regional Ministry of the Principado de Asturias under grants FC-GRUPIN-IDI/2018/000226 and the University of Jaén under the program "Acción 1. Apoyo a las estructuras de investigación de la Universidad de Jaén para incrementar su competitividad atendiendo a sus singularidades".

References

1. Alonso, P., Cortina, R., Rodríguez-Serrano, F.J., Vera-Candeas, P., Alonso-González, M., Ranilla, J.: Parallel online time warping for real-time audio-to-score alignment in multi-core systems. The Journal of Supercomputing **73**(1), 126–138 (2017). DOI 10.1007/s11227-016-1647-5
2. Alonso, P., Vera-Candeas, P., Cortina, R., Ranilla, J.: An efficient musical accompaniment parallel system for mobile devices. Journal of Supercomputing **73**(1), 343–353 (2017). DOI 10.1007/s11227-016-1865-x
3. Arzt, A.: Flexible and Robust Music Tracking. Ph.D. thesis, Johannes Kepler University Linz (2016)
4. Carabias-Orti, J.J., Cobos, M., Vera-Candeas, P., Rodríguez-Serrano, F.J.: Nonnegative signal factorization with learnt instrument models for sound source separation in close-microphone recordings. EURASIP Journal on Advances in Signal Processing **2013**(1), 184 (2013). DOI 10.1186/1687-6180-2013-184. URL `http://asp.eurasipjournals.com/content/2013/1/184%5Cnhttp://link.springer.com/article/10.1186/1687-6180-2013-184https://asp-eurasipjournals.springeropen.com/articles/10.1186/1687-6180-2013-184`
5. Carabias-Orti, J.J., Rodriguez-Serrano, F., Vera-Candeas, P., Ruiz-Reyes, N., Canadas-Quesada, F.J.: An audio to score alignment framework using spectral factorization and dynamic time warping. ISMIR: proceedings of the International Conference of Music Information Retrieval pp. 742–748 (2015)
6. Cichocki, A., Amari, S.i.: Families of alpha- beta- and gamma- divergences: Flexible and robust measures of Similarities. Entropy (2010). DOI 10.3390/e12061532
7. Cont, A.: Realtime Audio to Score Alignment for Polyphonic Music Instruments, using Sparse Non-Negative Constraints and Hierarchical HMMS. In: 2006 IEEE International Conference on Acoustics Speed and Signal Processing Proceedings, vol. 5, pp. V–245–V–248. IEEE (2006). DOI 10.1109/ICASSP.2006.1661258
8. Dixon, S.: Live tracking of musical performances using on-line time warping. In: DAFx, pp. 1727–1728 (2005)
9. Ewert, S., Muller, M., Grosche, P.: High resolution audio synchronization using chroma onset features. In: 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 1869–1872. IEEE (2009). DOI 10.1109/ICASSP.2009.4959972
10. Févotte, C., Idier, J.: Algorithms for Nonnegative Matrix Factorization with the β-Divergence. Neural Computation **23**(9), 2421–2456 (2011). DOI 10.1162/NECO{_}a{_}00168. URL `http://www.mitpressjournals.org/doi/10.1162/NECO_a_00168`
11. Hu, N., Dannenberg, R.B., Tzanetakis, G.: Polyphonic audio matching and alignment for music retrieval. In: IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, vol. 2003-Janua, pp. 185–188. IEEE (2003). DOI 10.1109/ASPAA.2003.1285862
12. Kaprykowsky, H., Rodet, X.: Globally Optimal Short-Time Dynamic Time Warping, Application to Score to Audio Alignment. In: 2006 IEEE International Conference on Acoustics Speed and Signal Processing Proceedings, vol. 5, pp. V–249–V–252. IEEE (2006). DOI 10.1109/ICASSP.2006.1661259
13. Kurth, F., Muller, M.: Efficient Index-Based Audio Matching. IEEE Transactions on Audio, Speech, and Language Processing **16**(2), 382–395 (2008). DOI 10.1109/TASL.2007.911552

14. Munoz-Montoro, A.J., Carabias-Orti, J.J., Vera-Candeas, P., Canadas-Quesada, F.J., Ruiz-Reyes, N.: Online/offline score informed music signal decomposition: application to minus one. Eurasip Journal on Audio, Speech, and Music Processing **2019**(1), 23 (2019). DOI 10.1186/s13636-019-0168-6. URL `https://asmp-eurasipjournals.springeropen.com/articles/10.1186/s13636-019-0168-6`
15. Orio, N., Schwarz, D.: Alignment of monophonic and polyphonic music to a score. In: Proceedings of the International Computer Music Conference, pp. 155–158 (2001)
16. Otsuka, T., Takahashi, T., Okuno, H.G., Komatani, K., Ogata, T., Murata, K., Nakadai, K.: Incremental polyphonic audio to score alignment using beat tracking for singer robots. In: 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2289–2296. IEEE (2009). DOI 10.1109/IROS.2009.5354637
17. Raffel, C., Ellis, D.P.W.: Large-Scale Content-Based Matching of Midi and Audio Files. In: Proceedings of the International Society for Music Information Retrieval Conference (2015)
18. Raffel, C., Ellis, D.P.W.: Extracting Ground Truth Information from MIDI Files: A Midifesto. Proc. 17th International Society for Music Information Retrieval Conference pp. 796–802 (2016)
19. Raphael, C.: Music Plus One and Machine Learning. In: Proceedings of the 27th International Conference on Machine Learning, pp. 21–28 (2010)
20. Rodríguez-Serrano, F.J., Carabias-Orti, J.J., Vera-Candeas, P., Canadas-Quesada, F.J., Ruiz-Reyes, N.: Monophonic constrained non-negative sparse coding using instrument models for audio separation and transcription of monophonic source-based polyphonic mixtures. Multimedia Tools and Applications **72**(1), 925–949 (2014). DOI 10.1007/s11042-013-1398-8
21. Rodriguez-Serrano, F.J., Duan, Z., Vera-Candeas, P., Pardo, B., Carabias-Orti, J.J.: Online Score-Informed Source Separation with Adaptive Instrument Models. Journal of New Music Research **44**(2), 83–96 (2015). DOI 10.1080/09298215.2014.989174
22. Thickstun, J., Harchaoui, Z., Kakade, S.: Learning Features of Music from Scratch. In: ICLR, pp. 1–14 (2017)
23. Turetsky, R., Ellis, D.: Ground-Truth Transcriptions of Real Music from Force-Aligned MIDI Syntheses. Proceedings of the 4th International Symposium on Music Information Retrieval pp. 135–141 (2003). DOI 10.7916/D8S472CZ
24. Wang, A.L.C.: An Industrial Strength Audio Search Algorithm. Proceedings of the 4th International Society for Music Information Retrieval Conference (ISMIR 203), Baltimore, Maryland (USA), 26-30 October 2003 pp. 7–13 (2003). DOI 10.1109/IITAW.2009.110