Lyrics-based Audio Retrieval and Multimodal Navigation in Music Collections

Meinard Müller, Frank Kurth, David Damm, Christian Fremerey, and Michael Clausen**

Department of Computer Science III, University of Bonn,
Römerstraße 164, 53117 Bonn, Germany
{meinard,frank,damm,fremerey,clausen}@iai.uni-bonn.de
WWW home page: http://www-mmdb.iai.uni-bonn.de

Abstract. Modern digital music libraries contain textual, visual, and audio data describing music on various semantic levels. Exploiting the availability of different semantically interrelated representations for a piece of music, this paper presents a query-by-lyrics retrieval system that facilitates multimodal navigation in CD audio collections. In particular, we introduce an automated method to time align given lyrics to an audio recording of the underlying song using a combination of synchronization algorithms. Furthermore, we describe a lyrics search engine and show how the lyrics-audio alignments can be used to directly navigate from the list of query results to the corresponding matching positions within the audio recordings. Finally, we present a user interface for lyrics-based queries and playback of the query results that extends the functionality of our SyncPlayer framework for content-based music and audio navigation.

1 Introduction

Recent activities in integrating music and audio documents into the holdings of existing digital libraries have emphasized the importance of appropriate tools for automatically organizing and accessing large music collections. As opposed to existing collections of homogeneous document types like text databases, musical information is represented in various different data formats such as text, score, or audio, which fundamentally differ in their structure and content. Hence there is a particular challenge to develop suitable techniques for searching and navigating through existing heterogeneous collections of digital music document.

As an example, consider a user who only recalls a few words of a song's lyrics like, for example, parts of the hook line or of the chorus. Using these words as a query, a music search engine based on classical text-based retrieval may be used for searching a database of *text documents* containing the lyrics of a collection of songs. In this case, the retrieval results displayed to a user would consist of text passages corresponding to occurrences of the query terms within

** This work was supported in part by Deutsche Forschungsgemeinschaft (DFG) under grant 554975 (1) Oldenburg BIB48 OLof 01-02.

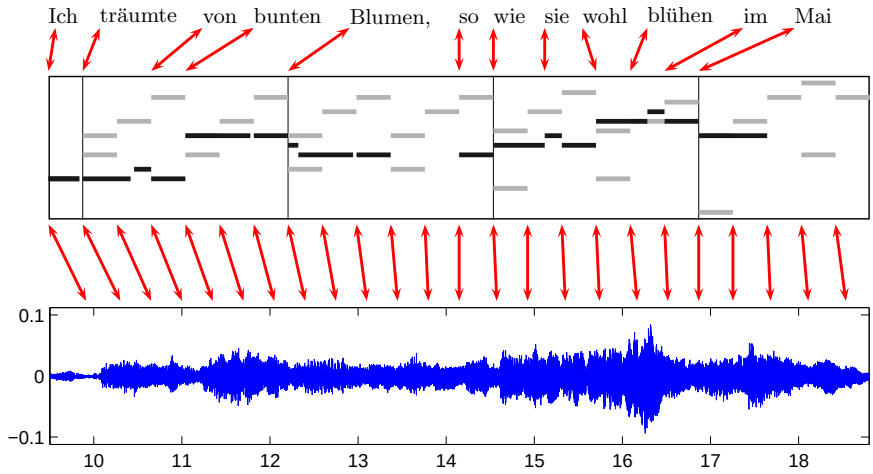

Fig. 1. Lyrics, MIDI version, and the waveform of an audio recording by Allen of measures 5 to 8 of Schubert's piano song D 911, No. 11 from the Lied cycle "Winterreise". The MIDI version is shown in piano-roll, where the black bars encode the vocal and the gray bars the piano track. The generated time alignments are indicated by the two-headed arrows.

the lyrics. However, in the music domain, the retrieval results are most naturally presented by acoustically playing back parts of an actual *audio recording* that contain the query terms, while a *musical score* may be the most appropriate form for visually displaying the query results. Such applications for *multimodal* music retrieval and navigation rely on the availability of suitable annotations and time alignments for connecting or *linking* the different types of available information related to a particular piece of music. In the latter example, an alignment of the lyrics to time positions in a corresponding audio recording would constitute such linking information.

Making lyrics-based audio retrieval feasible for larger scale music collections, this paper presents techniques for automatic lyrics-audio synchronization, for text-based lyrics search, as well as for multimodal music access and data presentation. As our first contribution, in Sect. 2 we describe a method to automatically generate audio *annotations* by temporally aligning the lyrics of a piece of music to audio recordings of the same piece. To solve this task, we exploit the availability of music documents in different data formats that describe a given piece of music at various semantic levels. In particular, we assume the availability of a MIDI representation (a kind of mid-level representation in between audio and score as will be described in Sect. 2), which serves as a "connector" in the lyrics-audio synchronization process: first we align the lyrics to the MIDI representation and then align the MIDI to an actual audio recording. This idea is illustrated by Fig. 1, which shows the lyrics, a MIDI version, and the waveform of an audio recording for measures 5 to 8 of Schubert's piano song D 911,

No. 11 from the Lied cycle "Winterreise". This piece, in the following simply referred to as Schubert example, will serve as running example throughout this paper. Fig. 1 also shows two time alignments (a lyrics-MIDI and a MIDI-audio alignment), which are indicated by the bidirectional arrows. The availability of such alignments allows for accessing the audio recording exactly at the positions where a particular lyrics' term is sung by the vocalist.

In Sect. 3, we present effective as well as efficient methods for *searching* music collections based on textual queries by using a combination of indexing techniques from text retrieval and prior knowledge on the particularities of music lyrics. For evaluating this query-by-lyrics scenario, we integrated the proposed retrieval algorithms in the existing SyncPlayer framework, as will be discussed in Sect. 4. The SyncPlayer is basically an enhanced audio player providing a plug-in interface for multidmodal presentation, browsing, and retrieval of music data, see [1]. For *presenting* the query results to the user, we implemented a SyncPlayer plug-in for synchronously displaying the lyrics along with the audio playback. Based on the lyrics-audio alignments, the user may directly navigate from the list of lyrics-based query results to the corresponding matching positions within the audio recordings. Finally, in Sect. 5, we give an example to illustrate the interplay of various multimodal navigation and visualization tools and give prospects on future work. References to related work are given in the respective sections.

2 Alignment of Lyrics and Music Audio

In this section, we present a procedure for automatically annotating audio recordings of a given song by its corresponding lyrics. To this end, we will exploit the existence of various music representations in different data formats conveying different types of information on a piece of music. Before describing the actual lyrics-audio alignment procedure, we briefly discuss the involved data types and give references to related work on music alignment.

We start with the symbolic *score format*, which contains explicit information on the notes such as musical onset time, pitch, duration, and further hints concerning dynamics and agogics. In contrast, the purely physical *audio format* encodes the waveform of an audio signal as used for CD recordings. In generally, it is very difficult or even infeasible to extract musical note parameters from a given waveform, in particular for complex polyphonic music. The *MIDI format* may be thought of as a hybrid of the last two data formats which explicitly represents content-based information such as note onsets and pitches but can also encode agogic and dynamic subtleties of some specific interpretation. Finally, the *lyrics* represent the textual information of a song or opera. For an example, we refer to our Schubert example shown in Fig. 1.

A key idea for automatically organizing and annotating large music collections is to exploit the availability of different music representations at various semantic levels. To this end, one needs *alignment* or *synchronization* algorithms that automatically link and interrelate the differently formatted information sets

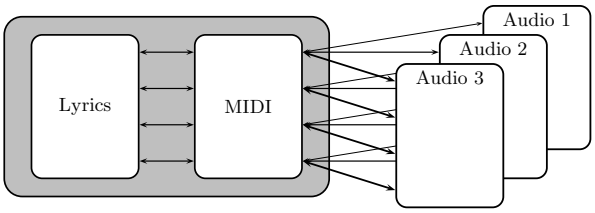

Fig. 2. Lyrics-audio alignment via MIDI-audio synchronization.

related to a single piece of music [2]. Here, *synchronization* is taken to mean a procedure which, for a given position in one representation of a piece of music, determines the corresponding position within another representation, see Fig. 1. In the last few years, extensive research has been conducted on automatic music synchronization and great advances have been achieved in aligning score, MIDI, and audio representations of a given piece, see [2–8] and the references therein. In contrast, the automatic alignment of lyrics to a corresponding audio recording of the underlying song, also referred to as *lyrics-audio synchronization*, is a very hard problem. In particular, the automatic recognition of vocals within a song seems infeasible without any additional assumptions. To alleviate the problem, Wang et al. [9] present an approach to automatic lyrics-audio alignment, which strongly relies on musical a priori knowledge of the song's and the lyrics' structure. Furthermore, the authors aim at a relatively coarse per-line alignment roughly estimating the start and end times for each lyrics line within the audio.

In the following, we describe a simple but effective procedure for automatic lyrics-audio synchronization, which works for large classes of music and generates precise alignments on the word or even syllable level. In our approach, we assume the existence of a MIDI file, which represents the symbolic score information and contains the lyrics along with MIDI time stamps. Then, the lyrics can be located within a given audio recording by aligning the MIDI note parameters to the audio data. In other words, we solve the original problem by computing a lyrics-MIDI alignment and then by applying a MIDI-audio synchronization, which can be done very efficiently as described below. To legitimate this procedure, we note that attaching lyrics to MIDI data has to be done only *once* independent of a particular interpretation or instrumentation. At the end of this section, we describe how this can be done using a semi-automatic procedure. Such an enriched MIDI can then be used for lyrics-audio alignment for *all* available audio recordings of the respective piece, see Fig. 2. This situation applies to a wide range of pieces from Western classical music, where one often has a few dozens of different CD recordings of an opera or a song.

In the first step of our algorithm for MIDI-audio synchronization, we transform the MIDI as well as the audio data into a common mid-level representation, which allows for comparing and relating music data in various realizations and

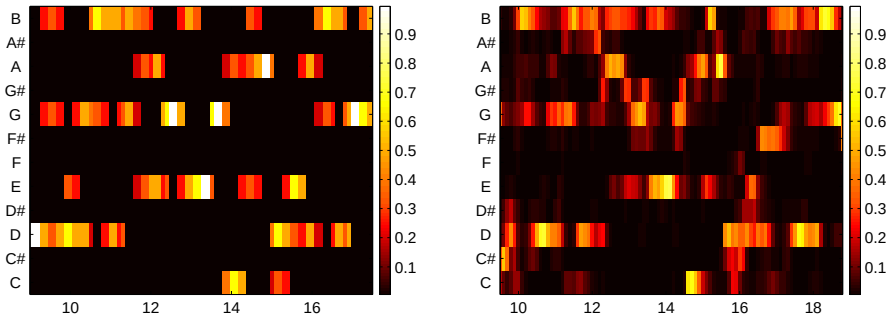

Fig. 3. Normalized chroma representations for the Schubert example (Fig. 1) derived from a MIDI representation (left) and an audio recording by Allen (right).

formats. In particular, we use chroma-based features, where the *chroma* correspond to the twelve traditional pitch classes of the equal-tempered scale. These features account for the well-known phenomenon that human perception of pitch is periodic in the sense that two pitches are perceived as similar in "color" if they differ by an octave [10]. Assuming the equal-tempered scale, the chroma correspond to the set $\{\mathrm{C}, \mathrm{C}^\sharp, \mathrm{D}, \ldots, \mathrm{B}\}$ that consists of the twelve pitch spelling attributes as used in Western music notation. Note that in the equal-tempered scale, different pitch spellings such $\mathrm{C}^\sharp$ and $\mathrm{D}^\flat$ refer to the same chroma. Now, using the explicit MIDI pitch and timing information, one can directly transform a MIDI data stream into a sequence of normalized 12-dimensional chroma vectors, where each vector covers a specific time interval. Such a chroma representation is also referred to as *chromagram*. To compute a MIDI chromagram, as suggested in [4], each pitch is associated to a corresponding chroma unit vector. Then, for a fixed time interval, one adds up the chroma unit vectors of all active MIDI pitches and normalizes the resulting sum vector. In our implementation, we work with a resolution of 10 features per second, where each feature vector corresponds to a 200 ms time interval. Similarly, the audio data stream is transformed into a chromagram representation. This can be done by suitably binning spectral coefficients [10] or by employing a suitable pitch filter bank [11]. Fig. 3 shows a MIDI chromagram as well as an audio chromagram for our Schubert example. Note that a normalized 12-dimensional chroma vector expresses the relative distribution of the signal's local energy content within the 12 chroma bands. A chromagram shows a high degree of robustness to variations in dynamics, timbre, as well as articulation and strongly correlates to the harmonic progression of the underlying pieces.

In the second step, the MIDI and audio data streams can be directly compared on the chroma representation level. Denoting the feature sequence of the MIDI file by $V := (v_1, v_2, \ldots, v_N)$ and of the audio file by $W := (w_1, w_2, \ldots, w_M)$, one builds an $N \times M$ cross-similarity matrix by calculating a similarity value for each pair of features (v_n, w_m), $1 \leq n \leq N$, $1 \leq m \leq M$. An alignment path of

maximum similarity is determined from this matrix via dynamic programming. Note that the time and memory complexity of this problem is proportional in the product $N \times M$, which becomes problematic for long pieces. To overcome this issue, the calculation is iteratively performed on multiple scales of temporal resolution going from coarse to fine. The alignment results of the coarser scale are used to constrain the calculation on the finer scales. For details we refer to [6]. The resulting optimal path encodes the MIDI-audio alignment as indicated by the bidirectional arrows in Fig. 1.

Extensive tests on a large corpus of Western music as described in [6] have shown that our synchronization algorithm yields accurate MIDI-audio alignments at a 100 ms resolution level (only in few cases there are some deviations of up to a second), which is sufficient for our lyrics-based audio retrieval and navigation application. As was mentioned above, we need enriched MIDI files that contain the lyrics along with MIDI time stamps. Since such MIDI files are rarely available, we semi-automatically annotated the lyrics for several popular songs as well as the 24 songs of Franz Schubert's Lied cycle Winterreise (op. 89, D. 911). To this end, we collected freely available lyrics (often already containing suitable syllable divisions) as well as corresponding MIDI files from the WWW. We then manually processed the lyrics by attaching the number of musical notes corresponding to the respective word or syllable. As it turns out, this process is not too laborious since in most cases each given word or syllable corresponds to exactly one note. Actually, this information was sufficient to automatically derive the desired MIDI time stamps for the lyrics simply by sequentially reading off the MIDI note onsets from the vocal track. In Sect. 4, we describe how the synchronization results are integrated into our SyncPlayer system.

3 Lyrics-based Music Retrieval

We now describe our index-based method for lyrics-based retrieval. In the following, we assume that the lyrics for our collection of N audio recordings are stored in N text files $\mathcal{L} := (L_1, \ldots, L_N)$ where a file L_i consists of a sequence $(t_{i1}, \ldots, t_{in_i})$ of terms. Our indexing technique uses inverted files which are well known from classical text retrieval [12]. In lyrics-based music retrieval, users are likely to query catchy phrases as they frequently occur in the chorus or hook line of a song. Therefore, our basic indexing strategy presented next is designed to efficiently retrieve exact sequences of query terms. Later on, this basic strategy is extended to allow fault tolerant retrieval.

In a preprocessing step, for each term t an inverted file $H_{\mathcal{L}}(t)$ is constructed from our text files. $H_{\mathcal{L}}(t)$ contains all pairs (i, p) such that t occurs as p-th lyrics term within text file L_i, i.e., $t_{ip} = t$. Using inverted files, query processing may then be performed simply by using intersections of inverted files: assume a query is given as a sequence of words $q := (t_0, \ldots, t_k)$. Then, the set of *matches*

$$H_{\mathcal{L}}(q) := \bigcap_{j=0}^{k} H_{\mathcal{L}}(t_j) - j \tag{1}$$

can be easily shown to contain all pairs (i, p) such that the exact sequence of terms q occurs at position p within the i-th document. To make this basic matching procedure robust towards errors such as misspelled or wrong words, we introduce several methods for incorporating fault tolerance. To account for typing errors, we preprocess each query term t_j and determine the set T_j of all terms in our dictionary of inverted files (i.e., the set of all terms with an existing inverted file) having a small edit distance to t_j. Then, instead of only considering the exact spelling t_j by using $H_{\mathcal{L}}(t_j)$ in (1), we consider the union $\cup_{t \in T_j} H_{\mathcal{L}}(t)$ of occurrences of all terms which are close to t_j with respect to their edit distance. To account for term-level errors such as inserted or omitted words, we first preprocess all word positions occurring in (1) by a suitable quantization. This amounts to replacing each of the inverted files $H_{\mathcal{L}}(t)$ by a new set $\lfloor H_{\mathcal{L}}(t)/Q \rfloor \cdot Q$, where each entry (i, p) of $H_{\mathcal{L}}(t)$ is replaced by a quantized version $(i, \lfloor p/Q \rfloor \cdot Q)$ for a suitably chosen integer Q ($Q = 5$ was used in our tests). Furthermore, we replace $H_{\mathcal{L}}(t_j) - j$ of (1) by $H_{\mathcal{L}}(t_j) - \lfloor j/Q \rfloor \cdot Q$ prior to calculating the intersection. The latter yields a list $(m_1, \ldots, m_\ell)$ of matches which is subsequently ranked.

For each match m_i we obtain a ranking value r_i by combining classical ranking criteria (r_i^1, r_i^2 and r_i^3 in what follows) with criteria accounting for the peculiarities of the lyrics-based scenario (r_i^4 in what follows). As for the classical criteria, each match m_i is assigned a ranking value r_i^1 that essentially measures the deviation of the query terms occurring in m_i from their correct ordering as specified by q. To account for term-level mismatches, a ranking value r_i^2 counts the total number of query terms occurring in m_i. Note that r_i^2 may be obtained efficiently by using a dynamic programming technique [13] while simultaneously calculating the set of matches (1). A further ranking value r_i^3 accounts for the total edit distance of the query terms to the terms matched in m_i. Exploiting the lyrics-audio alignment corresponding to m_i, we obtain a ranking value r_i^4 by suitably weighting the temporal distance (within the audio recording) of the first and the last query term occurring in m_i. Finally, an overall ranking value for each match is obtained as $r_i := \sum_{j=1}^{4} w_j r_i^j$, where $w_1, \ldots, w_4$ denote some suitably chosen real-valued weighting factors. In future work, it will be interesting to include even more music-specific knowledge into the ranking procedure. As an example, one might exploit available information about the structure of the audio recording to give lyrics terms more weight if they occur in structurally salient passages such as in the chorus sections.

The proposed methods for indexing and retrieval can be realized by properly adapting well-known text retrieval techniques. To verify the efficiency of the proposed methods, we created an index for a test corpus of approximately 110.000 lyrics documents of mainly popular music crawled from the web. As described in [13], retrieval using the proposed inverted-file based approach can be performed very efficiently in both of the cases of exact and fault tolerant retrieval including the proposed ranking.

To conclude this section, we note that in the above we have tacitly assumed that the aligment information for linking the lyrics to the audio recordings is stored in some suitable secondary data structure that can be accessed efficiently.

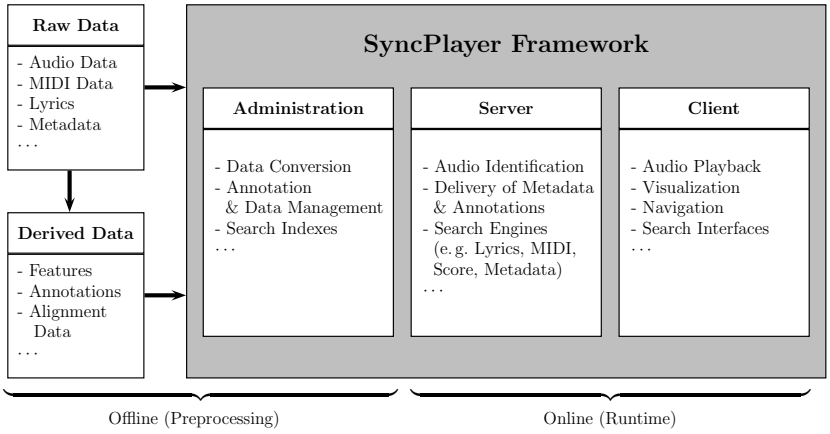

Fig. 4. Overview of the SyncPlayer framework.

In our implementation, we use an additional file format to store this information which turns out to perform sufficiently well.

4 A Prototypical System for Lyrics-Based Audio Retrieval

In this section, we present a prototypical implementation of a system for lyrics-based audio retrieval. Our system has been realized based on the existing SyncPlayer framework, which basically consists of an advanced audio player offering a plug-in interface for integrating content-based MIR applications. In Sect. 4.1, we first briefly summarize the SyncPlayer framework and its components. Subsequently, Sect. 4.2 demonstrates how the methods for annotation and retrieval presented in Sect. 2 and Sect. 3 are integrated into this framework.

4.1 The SyncPlayer Framework

The SyncPlayer is a client-server based software framework that integrates various MIR-techniques such as music synchronization, content-based retrieval, and multimodal presentation of content-based audiovisual data [1]. The framework basically consists of three software components as depicted in Fig. 4: a server component, a client component, and a toolset for data administration.

The user operates the *client component*, which in its basic mode acts like a standard software audio player for `*.mp3` and `*.wav` files. Additional interfaces, e.g., for performing content-based queries as well as various visualization tools, are provided through plug-ins (see Fig. 5). A remote computer system runs the *server component*, which supplies the client with annotation data such as synchronization information and controls several query engines for different types

of content-based audio retrieval. Several server-side *administration tools* are used for maintaining the databases and indexes underlying the SyncPlayer system.

The SyncPlayer framework offers two basic modes for accessing audio documents and corresponding content-based information. First, a user operating the client system may choose locally available audio recordings for playback. The client then extracts features from the audio recordings which are sent to the remote SyncPlayer server. The server subsequently attempts to *identify* the audio recording based on the submitted features. Upon success, the server searches its database for available annotations (such as lyrics or notes) which are then sent back to the client. The client system offers the user several visualization types for the available annotations. Two examples are a karaoke-like display for lyrics information and a piano-roll style display for note (MIDI-) information. Fig. 5 shows the SyncPlayer client (top left) along with the MultiVis plug-in for displaying lyrics synchronously to audio playback (bottom left).

The second method for accessing audio documents using the SyncPlayer is by means of appropriate query engines. In this scenario, the user operates a query plug-in offered by the client. Queries are submitted to the SyncPlayer server which, depending on the query type, schedules the queries to an appropriate query engine. A ranked list of retrieval results is returned to the client and displayed to the user. The user may then select particular query results for playback which are subsequently streamed from the server along with available annotation data. Note that the latter type of streaming is generally only allowed for authorized users. For more detailed information on the SyncPlayer framework, we refer to [1, 14]. A demo version of the SyncPlayer is available for download at the SyncPlayer Homepage [15].

4.2 The Lyrics Search Plug-in

The methods for aligning lyrics to audio recordings presented in Sect. 2 as well as the procedures for creating the lyrics-based search index (see Sect. 3) have been integrated into the SyncPlayer administration tools. A query engine for lyrics-based search according to Sect. 3 has been implemented and connected to the SyncPlayer server.

On the client side, a query interface for textual queries has been developed. The interface is shown in the right part of Fig. 5. Query results returned by the server are displayed in the same window according to the formerly described ranking criterion. Fig. 5 shows a query and corresponding query results for a lyrics fragment taken from the song *Yellow Submarine* by the Beatles. The first two matches are displayed in the lower part of the interface. Note that due to our matching strategy, a match not only consists of a particular document ID but also of the precise position (in seconds) of the query terms within the corresponding audio recording. Upon selecting one of the matches in the result list, the audio recording is transfered to the SyncPlayer client (provided the audio recording is available for download) and playback starts directly at the matching position.

At the moment, lyrics search in the online version of the SyncPlayer framework works on our test corpus of about 100 audio recordings including the 24

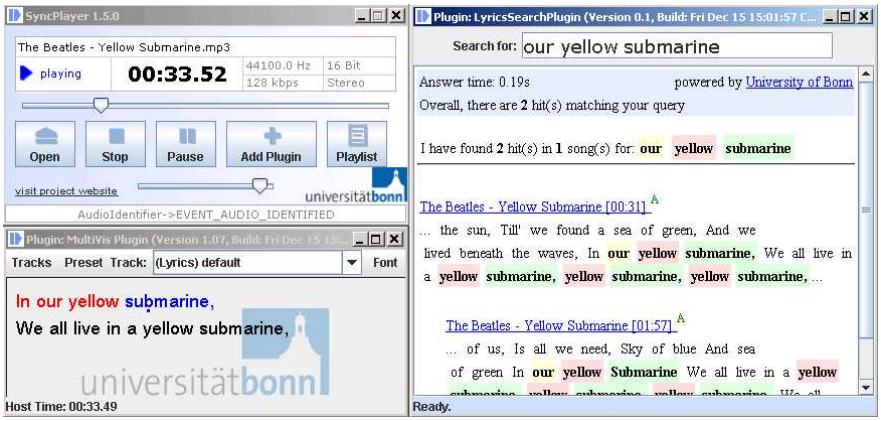

Fig. 5. SyncPlayer client (top left), MultiVis plug-in for displaying lyrics synchronously to audio playback (bottom left) and Lyrics Search plug-in for textual queries (right).

piano songs as discussed above. A larger scale evaluation of the Lyrics Seeker including relevance-based measures for retrieval performance will be conducted within the Probado library project which is summarized in the next section.

5 Conclusions and Future Work

In this paper, we have introduced a combination of methods facilitating lyrics-based audio retrieval in digital music collections. Based on automatically generated lyrics-audio alignments, the user is enabled to directly access audio material from the result list obtained by our query engine for text-based lyrics retrieval. As illustrated by Fig. 6, these functionalities can be combined with further browsing and visualization tools to allow for multimodal inter- and intra-document navigation in inhomogeneous and complex music libraries. Here, the SyncPlayer plug-in concept allows any number of plug-ins to be opened at the same time. For example, the lyrics search plug-in affords lyrics-based audio retrieval, while the lyrics visualization plug-in displays the text as in typical karaoke applications. At the same time the audio structure plug-in facilitates intra-document browsing on the basis of the repetitive structure of the respective audio recording, where the audio structure has been extracted in a fully automated process, see, e. g., [11]. Similar to our lyrics-based audio retrieval scenario, the availability of MIDI-audio alignments can be used to facilitate score-based audio access as well as synchronous score visualization as indicated by the piano-roll representation in Fig. 6. Further plug-ins may be used for displaying the waveform, a spectogram, a chormagram, or other derived audio representations.

There are many meaningful ways to add functionalities for content-based, multimodal music retrieval and navigation. For example, a functionality for synchronously displaying high quality scanned scores is currently being developed.

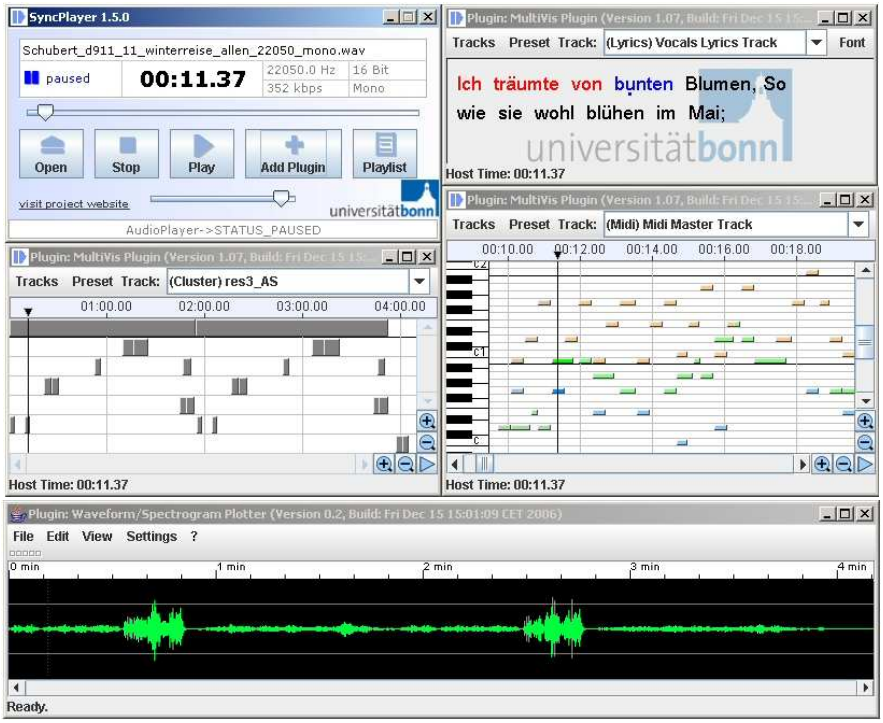

Fig. 6. SyncPlayer client with several of the available plug-ins for browsing and visualization illustrated for the Schubert example from Fig. 1. The figure shows the main audio player (upper left) as well as plug-ins for visualizing lyrics (upper right), audio structure (middle left), piano-roll (middle right), and the waveform signal (bottom).

This also requires an automated procedure to time align the pixels of scanned sheet music to corresponding time positions within an audio recording.

In future work, we will investigate novel ways for automatic lyrics to audio alignment based on scanned score material. Furthermore, we plan to integrate several other query engines into the SyncPlayer framework facilitating, e.g., audio matching [16] and score-based retrieval [13]. The methods and software components presented in this paper will be used within the German digital library initiative *Probado* [17] that aims at integrating (non-textual) multimedia documents into the workflow of existing digital libraries. In the Probado project, we are currently setting up a repository of digitized music documents consisting of audio recordings and scanned sheet music. In this context, the proposed methods will be used for the tasks of automatic annotation of music documents (lyrics-audio alignment, Sect. 2), content-based audio retrieval (lyrics search, Sect. 3), and content-based navigation in audio recordings (SyncPlayer and plug-ins, Sect. 4).

In conclusion, we hope that our advanced audio player opens new and unprecedented ways of music listening and experience, provides novel browsing and retrieval strategies, and constitutes a valuable tool for music education and music research. For the future, large-scale evaluations and systematic user studies have to be conducted to identify user needs and to convert the SyncPlayer into a system which is suitable for permanent application in existing digital libraries.

References

1. Kurth, F., Müller, M., Damm, D., Fremerey, C., Ribbrock, A., Clausen, M.: SyncPlayer — An Advanced System for Multimodal Music Access. In: ISMIR, London, GB. (2005)
2. Arifi, V., Clausen, M., Kurth, F., Müller, M.: Synchronization of music data in score-, MIDI- and PCM-format. Computing in Musicology **13** (2004)
3. Dannenberg, R., Raphael, C.: Music score alignment and computer accompaniment. Special Issue, Commun. ACM **49**(8) (2006) 39–43
4. Hu, N., Dannenberg, R., Tzanetakis, G.: Polyphonic audio matching and alignment for music retrieval. In: Proc. IEEE WASPAA, New Paltz, NY. (October 2003)
5. Müller, M., Kurth, F., Röder, T.: Towards an efficient algorithm for automatic score-to-audio synchronization. In: Proc. ISMIR, Barcelona, Spain. (2004)
6. Müller, M., Mattes, H., Kurth, F.: An efficient multiscale approach to audio synchronization. In: Proc. ISMIR, Victoria, Canada. (2006) 192–197
7. Soulez, F., Rodet, X., Schwarz, D.: Improving polyphonic and poly-instrumental music to score alignment. In: Proc. ISMIR, Baltimore, USA. (2003)
8. Turetsky, R.J., Ellis, D.P.: Force-Aligning MIDI Syntheses for Polyphonic Music Transcription Generation. In: Proc. ISMIR, Baltimore, USA. (2003)
9. Wang, Y., Kan, M.Y., Nwe, T.L., Shenoy, A., Yin, J.: Lyrically: automatic synchronization of acoustic musical signals and textual lyrics. In: MULTIMEDIA '04: Proceedings of the 12th annual ACM international conference on Multimedia, New York, NY, USA, ACM Press (2004) 212–219
10. Bartsch, M.A., Wakefield, G.H.: Audio thumbnailing of popular music using chroma-based representations. IEEE Trans. on Multimedia **7**(1) (2005) 96–104
11. Müller, M., Kurth, F.: Towards structural analysis of audio recordings in the presence of musical variations. EURASIP Journal on Advances in Signal Processing **2007** (2007) Article ID 89686, 18 pages doi:10.1155/2007/89686.
12. Witten, I.H., Moffat, A., Bell, T.C.: Managing Gigabytes. 2nd edn. Van Nostrand Reinhold (1999)
13. Clausen, M., Kurth, F.: A Unified Approach to Content-Based and Fault Tolerant Music Recognition. IEEE Transactions on Multimedia **6**(5) (2004)
14. Fremerey, C.: SyncPlayer – a Framework for Content-Based Music Navigation. Diploma Thesis, Dept. of Computer Science, University of Bonn (2006)
15. Multimedia Signal Processing Group Prof. Dr. Michael Clausen: SyncPlayer Homepage. Website (January 2007) http://www-mmdb.iai.uni-bonn.de/projects/syncplayer/index.php.
16. Müller, M., Kurth, F., Clausen, M.: Audio matching via chroma-based statistical features. In: Proc. ISMIR, London, GB. (2005)
17. Steenweg, T., Steffens, U.: Probado – non-textual digital libraries put into practice. ERCIM News **Special Theme: European Digital Library** (July 2006) 47–48