Music Synchronization, Audio Matching, Pattern Detection, and User Interfaces for a Digital Music Library System

Dissertation

zur

Erlangung des Doktorgrades (Dr. rer. nat.)

der

Mathematisch-Naturwissenschaftlichen Fakultät

der

Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von

Verena Kriesel (geb. Thomas)

aus

Bonn

Bonn, Februar 2013

Angefertigt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Rheinischen Friedrich-Wilhelms-Universität Bonn

1. Gutachter:	Prof. Dr. Michael Clausen
2. Gutachter:	Prof. Dr. Meinard Müller

Tag der Promotion:	20.06.2013
Erscheinungsjahr:	2013

Music Synchronization, Audio Matching, Pattern Detection, and User Interfaces for a Digital Music Library System

Verena Kriesel

Abstract

Over the last two decades, growing efforts to digitize our cultural heritage could be observed. Most of these digitization initiatives pursuit either one or both of the following goals: to conserve the documents – especially those threatened by decay – and to provide remote access on a grand scale. For music documents these trends are observable as well, and by now several digital music libraries are in existence. An important characteristic of these music libraries is an inherent multimodality resulting from the large variety of available digital music representations, such as scanned score, symbolic score, audio recordings, and videos. In addition, for each piece of music there exists not only one document of each type, but many. Considering and exploiting this multimodality and multiplicity, the DFG-funded digital library initiative PROBADO MUSIC aimed at developing a novel user-friendly interface for content-based retrieval, document access, navigation, and browsing in large music collections. The implementation of such a front end requires the multimodal linking and indexing of the music documents during preprocessing. As the considered music collections can be very large, the automated or at least semi-automated calculation of these structures would be recommendable. The field of *music information retrieval* (MIR) is particularly concerned with the development of suitable procedures, and it was the goal of PROBADO MUSIC to include existing and newly developed MIR techniques to realize the envisioned digital music library system. In this context, the present thesis discusses the following three MIR tasks: music synchronization, audio matching, and pattern detection. We are going to identify particular issues in these fields and provide algorithmic solutions as well as prototypical implementations.

In *Music synchronization*, for each position in one representation of a piece of music the corresponding position in another representation is calculated. This thesis focuses on the task of aligning scanned score pages of orchestral music with audio recordings. Here, a previously unconsidered piece of information is the textual specification of transposing instruments provided in the score. Our evaluations show that the neglect of such information can result in a measurable loss of synchronization accuracy. Therefore, we propose an OCR[1]-based approach for detecting and interpreting the transposition information in orchestral scores.

For a given audio snippet, *audio matching* methods automatically calculate all musically similar excerpts within a collection of audio recordings. In this context, subsequence dynamic time warping (SSDTW) is a well-established approach as it allows for local and global tempo variations between the query and the retrieved matches. Moving to real-life digital music libraries with larger audio collections, however, the quadratic runtime of SSDTW results in untenable response times. To improve on the response time, this thesis

[1] *Optical character recognition*

introduces a novel index-based approach to SSDTW-based audio matching. We combine the idea of inverted file lists introduced by Kurth and Müller (*Efficient index-based audio matching*, 2008) with the shingling techniques often used in the audio identification scenario.

In *pattern detection*, all repeating patterns within one piece of music are determined. Usually, pattern detection operates on symbolic score documents and is often used in the context of computer-aided motivic analysis. Envisioned as a new feature of the Probado Music system, this thesis proposes a string-based approach to pattern detection and a novel interactive front end for result visualization and analysis.

Keywords: digital music representations, sheet music, optical music recognition, digital music libraries, user interfaces, automatic document organization, music synchronization, transposing instruments, audio matching, motivic analysis, pattern detection

Acknowledgments

This thesis was created during my employment at the Department of Computer Science III, University of Bonn. During my time there, I was primarily funded by the German Research Foundation (DFG). I had the pleasure of working on the projects ARMADA,[2] PROBADO,[3] and very briefly METRUM.[4]

My special thanks go to my supervisor Prof. Dr. Michael Clausen. I am grateful for the opportunity to work in his workgroup where I was able to learn not only about good science but also software design and the practical work of a computer scientist. Without his support and guidance, the creation of this thesis would have been unthinkable. I would also like to thank my second supervisor, Prof. Dr. Meinard Müller. He always had an open ear for questions and more than once provided helpful comments on my written work.

Over the years, many current and former colleagues accompanied me during the preparation of this thesis. I would like to thank David Damm, Jonathan Driedger, Sebastian Ewert, Christian Fremerey, and Harald Grohganz for the great experience of working with them and their helpful discussions. I am particularly grateful to Christian Fremerey for his support in my first months in the group and his patience while walking me through the existing code. Thanks also go to all the student assistants, in particular Polina Gubaidullina, Joscha Rieber, and Christian Wagner, for their support in testing and realizing several ideas.

Further thanks go to Felicitas von Bemberg-Flamersheim and Sebastian Merzbach for critically and carefully reading through many pages of this thesis. I am also grateful for the occasional distractions they offered from science in the form of inline skating and homemade pizza. I would also like to thank Pete Walton for the fast and thorough correction of this thesis.

I thank my family, in particular my parents Gisela Raps and Carsten Thomas, for their support and understanding during my time as a PhD student. To my soon-to-be husband David Kriesel I offer my deepest thanks for always believing in me and supporting me with his love and patience.

[2] *Automatisierte Erschließung von Musikdokumenten unter Ausnutzung verschiedener Darstellungsformen*

[3] *Prototypischer Betrieb allgemeiner Dokumente*

[4] *Mehrschichtige Analyse und Strukturierung von Musiksignalen*

Contents

1 Introduction

Some of the most important achievements of modern information technology are the development of the World Wide Web and the tremendous advances in data storage, data acquisition, and computing power. This technological progress enabled the emergence of new ways to solve problems that seemed unsolvable only a few decades ago. One such problem is the long-term preservation of our cultural heritage. Vast amounts of cultural material of all document types, such as books, newspapers, images, photos, (gramophone) records, and video tapes, are stored in libraries, museums, and private collections all over the world. Not only are these physical documents threatened by decay and thus ultimately destruction, they are also difficult to access as one has to visit the respective institution or apply for an inter-library loan. However, especially old documents are usually kept in special storage in order to slow down their decay and are therefore not available to the general public. Thanks to technological progress, the creation of digital surrogates of these documents is now a realistic enterprise, and several national and international digitization initiatives for the preservation of our cultural heritage have been launched.[1] In addition, several libraries and collections decided to create digital copies of their archives as well.

However, the generation and collection of digitized surrogates is only the first step. To avoid digital graveyards, the generated data has to be processed, analyzed, annotated, and organized. Due to the potentially large data volumes, a high degree of automation of these tasks would be desirable. Furthermore, to enable access to digital collections, intuitive interfaces that support searching, browsing, navigating, and the extraction of information from the collections should be made available. For scanned text documents, various solutions for automated document processing and document access have been proposed. Typically, these systems include optical character recognition (OCR) to extract the textual content from the images and fault-tolerant full-text indexing and retrieval. To provide access to the documents, these two representations are then combined accordingly. The scanned images are used for document presentation, while the extracted textual information (in combination with the known position in the image) is used to provide users with convenient navigation and content-based search functionalities. A well-known example of such a system is Google Books [81].[2]

In contrast to the advances in the textual domain, there is still a significant lack of corresponding systems for general digitized non-textual documents, such as audios, videos,

[1] Examples are the project *Presto Space* (`http://www.prestospace.org`, February 2013) or the internet portal *Europeana* [68].

[2] The first two paragraphs were inspired by our papers [57, 194] and borrow some phrasings from them.

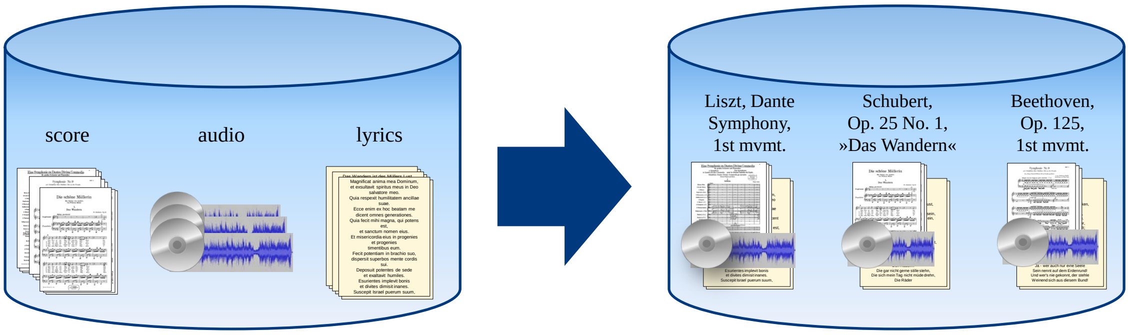

score
audio
lyrics
Liszt, Dante Symphony, 1st mvmt.
Schubert, Op. 25 No. 1, »Das Wandern«
Beethoven, Op. 125, 1st mvmt.

Figure 1.1. Change from a document- and document-type-centered data collection (left) to an arrangement focusing on pieces of music (right).

images, and 3D data. The recently terminated Probado project[3] attempted to address this imbalance by exploring and creating digital library services for non-textual documents that provide innovative user interfaces for content-based document access [20, 23, 28]. The definition and implementation of a widely automated document-processing chain (digitization, representation, indexing, annotation, content-based access, and presentation) that can be easily integrated into an existing library work flow constituted another important objective of the research initiative. Probado was further subdivided into the two sub-projects Probado 3D [21, 22, 29] and Probado Music [57, 106, 190] that focused on the support of 3D architectural models and digital music documents, respectively. Despite the restriction on these two data types, the developed architecture is not limited to these and it could be extended relatively simply to provide access to other types of multimedia documents. The research presented in this thesis was mostly inspired and driven by Probado Music and the particular challenges of creating such a digital music library system.

An important characteristic of music libraries is their inherent multimodality resulting from the large variety of available digital music representations (scanned score, symbolic score, audio and video recordings, as well as related material like musicological analyses, libretti, programs, critical reviews, photographs, and sketches for stage designs or costumes). Furthermore, for each piece of music there exists not one document of each type, but many. For example, there usually exist several score versions by different publishers and audio recordings of the same piece by different musicians. In music research, education, and experience one is often interested in accessing several documents representing the same piece of music simultaneously (to read the lyrics or the score while listening to the performance) or in direct succession (to compare different performances with each other). Therefore, digital music libraries can benefit tremendously from a *work-centered*[4] organization of their digital collection, see Figure 1.1, in combination with a multimodal document presentation. With Probado Music a prototype of such a library system was developed. In the current version, the system supports scanned scores and audio recordings, but its extensibility to further documents types, like video recordings, could be demonstrated in the context of this thesis.

While processing a real-life music collection, we gradually discovered new problems that have to be considered to allow for the envisioned functionality. In this thesis, we will focus

[3] `http://www.probado.de/en_home.html`, February 2013

[4] In the context of Probado Music a *work* denotes an individual piece of music (e.g., one movement of a piano sonata), see also Section 3.2.

on some designated challenges in music synchronization and audio matching and introduce our respective developed approaches. In addition, our work on Probado Music gave rise to a desire for several new features. Here, a particular focus was on the development of computational approaches to musicological analyses, such as structure analysis [136, 139], harmonic analysis [104], or – as in this thesis – motivic analysis [189].[5]

By means of *music synchronization*, music representations of the same work are semantically linked, see, e.g., [133, Chapter 5]. Given a score and a matching audio recording, these linking structures help to determine the measure in the score that matches the current position in the audio, and vice versa. Probado Music can thereby provide the user with *score-following* (i.e., during playback, the current position in the score is highlighted) and *score-based navigation* (i.e., selecting a measure in the score results in an update of the playback position in the audio). Given multiple audio interpretations, the synchronization data further allow the audio recording to be changed while retaining the musical position (*interpretation switching*). Most of the recently proposed synchronization approaches concentrate on piano music and chamber music. When moving on to more complex orchestral pieces, new issues emerge that currently have not been properly dealt with. For one, with increasing complexity of the score the employed optical music recognition (OMR) systems are inclined to produce more recognition errors. Another problem is the fact that no current recognition system detects transposing instruments in the score. Here, the sounding pitch produced by the instrument is several semitones higher or lower than the notes written down in the score. In this thesis, we demonstrate that neglecting the transposition information during synchronization results in a distinct quality loss. We then introduce a novel approach for reconstructing the transposition information from scanned score images. In the first step, OCR reconstructs the textual information in the score, which is subsequently translated into instrument labels and transposition information. The particular conventions in music notation lead to the fact that not all staves of a score might be equipped with textual labels. In the second step, a propagation method therefore aims at filling these potential gaps.

Similar to the full-text search in Google Books [81], content-based music retrieval techniques allow users of Probado Music to search for score samples, audio snippets, or lyrics. As a special feature, the graphical user interface of Probado Music offers the possibility to simply select a snippet from the currently visualized document as a query (*intra-collection query*). The task of searching for sections in an audio collection that present some similarity to a given audio snippet is often referred to as *audio matching*, see, e.g., [133, Chapter 6]. A powerful audio matching method is *subsequence dynamic time warping* (SSDTW) as it is capable of considering global as well as local tempo variations between the query and its reoccurrences. For music performances, where musicians often employ tempo changes as a stylistic feature, this is a most valuable property. However, the quadratic time complexity prevents the application of SSDTW in real-life systems such as Probado Music. In this thesis, we propose to exploit the specific intra-collection search scenario in Probado Music to create an index-based audio matching procedure for SSDTW that produces fast and accurate results. The underlying idea is to split the audio collection into equal-sized overlapping segments, to precompute their retrieval results, and to store these matches in

[5] The mentioned publications present work that was carried out in the extended context of Probado Music. More information and a selection of other approaches on structure analysis [61, 133, 153], harmonic analysis [18,123,179], and motivic analysis [43,90,111,128] are available in the respective cited publication.

appropriate index structures. During query processing the indexes of the segments covering the query are then merged to efficiently calculate the retrieval results.

In motivic analysis, a different type of similarity is studied in order to gain insights into the structure of a piece of music and the stylistic methods employed by the composer.[6] Motivic analysis is a delicate task, which for some subtasks requires the expert knowledge of trained musicologists. One such task is the establishment of the function and meaning of a motif. Due to the complexity and vagueness of this task, the complete automation of motivic analysis through computer algorithms has to be considered impossible. However, there are some steps that can be automated to yield systems for computer-aided motivic analysis. It is the goal of *pattern detection* methods to find repeating note sequences within a piece of music that constitute admissible motif candidates. In this thesis, we propose to add the feature of computer-aided motivic analysis techniques to digital music library systems like Probado Music. As a first contribution, we present a novel pattern detection approach. Most existing approaches only search for exact repetitions of note sequences. In contrast, we propose to also consider common motivic variations, such as inversions and retrogrades. Furthermore, we introduce a prototypical interactive front end for accessing and analyzing the detected patterns.

1.1 Contributions of this Thesis and Related Publications

We now provide a summary of the main contributions of this thesis. Parts of this work have been previously published. Therefore, a list of related publications by the author is provided for each topic. Some of the figures in this thesis were also taken from our own publications.[7]

- A first major contribution of this thesis is the integration of video and lyrics support into the Probado Music system. The following papers report on Probado Music and act as a basis for Chapter 3. Particularly noteworthy is the first publication as it provides a detailed description of our work on integrating video recordings and lyrics.

  [192] Verena Thomas, Christian Fremerey, David Damm, and Michael Clausen. SLAVE: a score-lyrics-audio-video-explorer. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 717–722, Kobe, Japan, 2009.

  [20] René Berndt, Ina Blümel, Michael Clausen, David Damm, Jürgen Diet, Dieter Fellner, Christian Fremerey, Reinhard Klein, Frank Krahl, Maximilian Scherer, Tobias Schreck, Irina Sens, Verena Thomas, and Raoul Wessel. The PROBADO project - approach and lessons learned in building a digital library system for heterogeneous non-textual documents. In *Proceedings of the European Conference on Digital Libraries (ECDL)*, pages 376–383, Glasgow, Scotland, 2010.

  [57] David Damm, Christian Fremerey, Verena Thomas, Michael Clausen, Frank Kurth, and Meinard Müller. A digital library framework for heterogeneous music collections – from document acquisition to cross-modal interaction. *International Journal on Digital Libraries*, 12(2-3):53–71, 2012.

[6] `http://de.wikipedia.org/wiki/Formenlehre_(Musik)`, February 2013

[7] Figures 1.1, 2.5, 2.9, 2.13, 2.15, 2.16, 3.10a, 3.10b, 3.11, 3.12, 4.1–4.6, 4.9, 4.12, 4.18, 5.1, and 5.2

[190] Verena Thomas, David Damm, Christian Fremerey, Michael Clausen, Frank Kurth, and Meinard Müller. PROBADO music: A multimodal online music library. In *Proceedings of the International Computer Music Conference Conference (ICMC)*, pages 289–292, Ljubljana, Slovenia, 2012.

- A novel procedure for the reconstruction of transposition information in scanned score documents constitutes another contribution of this thesis. The papers below together with the diploma thesis [204] form the basis of our description of this work.

[193] Verena Thomas, Christian Fremerey, Sebastian Ewert, and Michael Clausen. Notenschrift-Audio Synchronisation komplexer Orchesterwerke mittels Klavierauszug. In *Proceedings of the Deutsche Jahrestagung für Akustik (DAGA)*, pages 191–192, Berlin, Germany, 2010.

[195] Verena Thomas, Christian Wagner, and Michael Clausen. OCR-based post-processing of OMR for the recovery of transposing instruments in complex orchestral scores. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 411–416, Miami, FL, USA, 2011.

[194] Verena Thomas, Christian Fremerey, Meinard Müller, and Michael Clausen. Linking sheet music and audio - challenges and new approaches. In Meinard Müller, Masataka Goto, and Markus Schedl, editors, *Multimodal Music Processing*, Dagstuhl Follow-Ups, pages 1–22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 2012.

- In this thesis we also propose a novel index-based approach to audio matching in larger document collections. The chapter on this subject is based on the following paper.

[191] Verena Thomas, Sebastian Ewert, and Michael Clausen. Fast intra-collection audio matching. In *Proceedings of the ACM Workshop on Music Information Retrieval with User-Centered and Multimodal Strategies (MIRUM)*, pages 1–6, Nara, Japan, 2012.

- As a final contribution, a novel string-based pattern detection approach is introduced. Parts of the according chapter are based on the following publication.

[189] Verena Thomas and Michael Clausen. MotifViewer: Hierarchical pattern detection. In *Proceedings of the International Computer Music Conference Conference (ICMC)*, pages 555–558, Ljubljana, Slovenia, 2012.

1.2 Structure of this Thesis

This thesis consists of seven chapters. Chapter 2 introduces the different types of music representations of relevance for this thesis. Furthermore, we describe a derived feature representation and its calculation from digital music documents. Chapters 3–6 constitute the main part of this thesis and introduce our work on a digital music library system (Chapter 3) as well as new approaches to music synchronization (Chapter 4), audio matching (Chapter 5), and computer-aided motivic analysis (Chapter 6). Each chapter gives an introduction in the respective field and provides a review of related work before presenting our own contribution. We conclude the thesis in Chapter 7 with an outlook on possible future directions.

2 Music Representations, Transformations, and Music Features

Music is an acoustic art form and, therefore, comes alive through its performance. To store such a performance and enable the reproduction of a particular piece of music by other musicians, various music representations (or data formats), both physical and digital, have been developed over time. These formats differ considerably in their structure and content, which is why they usually capture different aspects of the given piece of music. In this chapter, we give a brief introduction to some existing music representations. We concentrate particularly on those (digital) representations relevant to the work presented in this thesis. Furthermore, we describe possible transformations between those representations with a special focus on the creation of symbolic score data from scanned score pages (*optical music recognition*). Both in music synchronization (Chapter 4) and in audio retrieval (Chapter 5), a derived music representation is created from the given music documents. Such a *mid-level* or *feature representation* captures only certain aspects of the piece of music, like the harmonic progression, while discarding others. Thus, the selection of an appropriate representation can significantly enhance the comparability of the music documents with respect to the given notion of similarity. At the end of this chapter, we introduce the *chroma features* used in this thesis and describe their calculation for different input formats.

2.1 Common Practice Notation

Common practice notation (CPN) is a system for the graphical representation of music. To this end, a fixed set of musical symbols and textual instructions was developed. The main goal of CPN is the creation of written instructions whereby musicians are able to rehearse and perform a piece of music. However, music notation is in many aspects restricted to providing rough instructions (e.g., tempo, dynamics, and articulation) and thereby offers performers a great deal of space for interpretation. In the remainder of this section, we describe the basic layout conventions of CPN and introduce some expressions required in this thesis.

Figure 2.1 shows an extract from a score representation of Bach's *Cello Suite No. 1*. In CPN, music events are represented by note objects capturing attributes such as pitch, musical onset time, duration, dynamics, and articulation. In addition, textual instructions, like `"Moderato"`, `"Andante con moto"` (global instructions) or `"accelerando"`, `"ritardando"`

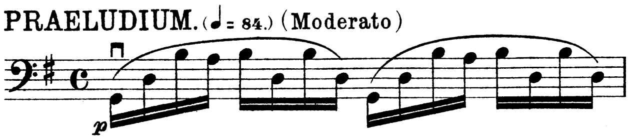


Figure 2.1. Score representation of the first two measures of Bach's *Cello Suite No. 1* (publisher: *C. F. Peters*) using CPN.

(for local tempo variations), specify the tempo. Often, the score also provides more explicit tempo instructions via metronome marks. For example, the metronome mark in Figure 2.1 indicates that the piece should be performed with a speed of 84 quarter notes per minute (84 BPM).[1] Other than the Latin writing system, CPN is a two-dimensional notation. Usually, the playing instructions for one instrument (or in orchestral music for a group of instruments) are represented in a *staff line* or *staff*, see Figure 2.2. A staff comprises five horizontal lines and normally spans the entire width of a score page. In such a staff, the horizontal position of the notes defines their timing, whereas the vertical placing (in combination with the clef) encodes their tone height (or pitch). In most classical music, several instruments perform simultaneously. To capture this parallelism, the staves of the individual instruments are placed below each other with a connecting vertical line in front of the staves. The resulting structure is referred to as *staff system* or *system*. Playing instructions for polyphonic instruments like the piano or the harp are usually provided in two separate staves connected by a *brace* (also *accolade*) at the beginning of the system. In piano music, this grouping of two staves is also called a *grand staff*.

In orchestral music a fixed seating plan – ordered by instrument families – is usually maintained during performances, see Figure 2.3. Equally, the staff order for the individual instruments typically obeys a common convention. In Table 2.1 the standard instrument order in scores for classical music is presented. While it is common practice to apply this order, there exist scores that deviate from it, see Figure 2.4. In the majority of cases, a piece of music uses only a subset of instruments and not a complete orchestra. To indicate the instrumentation of a given piece of music and which instrument is supposed to play a given staff (*instrument-staff mapping*), the first system provides textual information on the instrumentation, see Figure 2.2. In most scores, the number of staves remains constant throughout the entire piece of music. Therefore, the instrument names are often omitted after the first system. However, there also exists a non-negligible number of scores where a *compressed notation* is used instead. In compressed notation, the instrument order (usually) established in the first system remains valid throughout the score. However, the staves of pausing instruments are removed and staves of the same instruments that temporarily play in unison might be merged (or vice versa), see Figures 2.5 and 2.6.[2] Equally, additional staves for new voices of a particular instrument can appear in the course of the piece of music. Thus, the number of staves in the systems varies and the instrument-staff mapping from the first system cannot be transferred. To clarify the instrument-staff mapping in these cases, textual information on the instrumentation is provided. However, the instrument names are usually abbreviated after the first system. While some editors choose to provide instrument names in front of all systems, most only clarify the instrument-staff mapping

[1] "Tempo (i)", Grove Music Online. `http://www.oxfordmusiconline.com/subscriber/article/grove/music/27649`, February 2013.

[2] In most compressed scores, the first system provides information on all instruments occurring in the piece of music. However, a small number of scores only introduces instruments once they start playing.

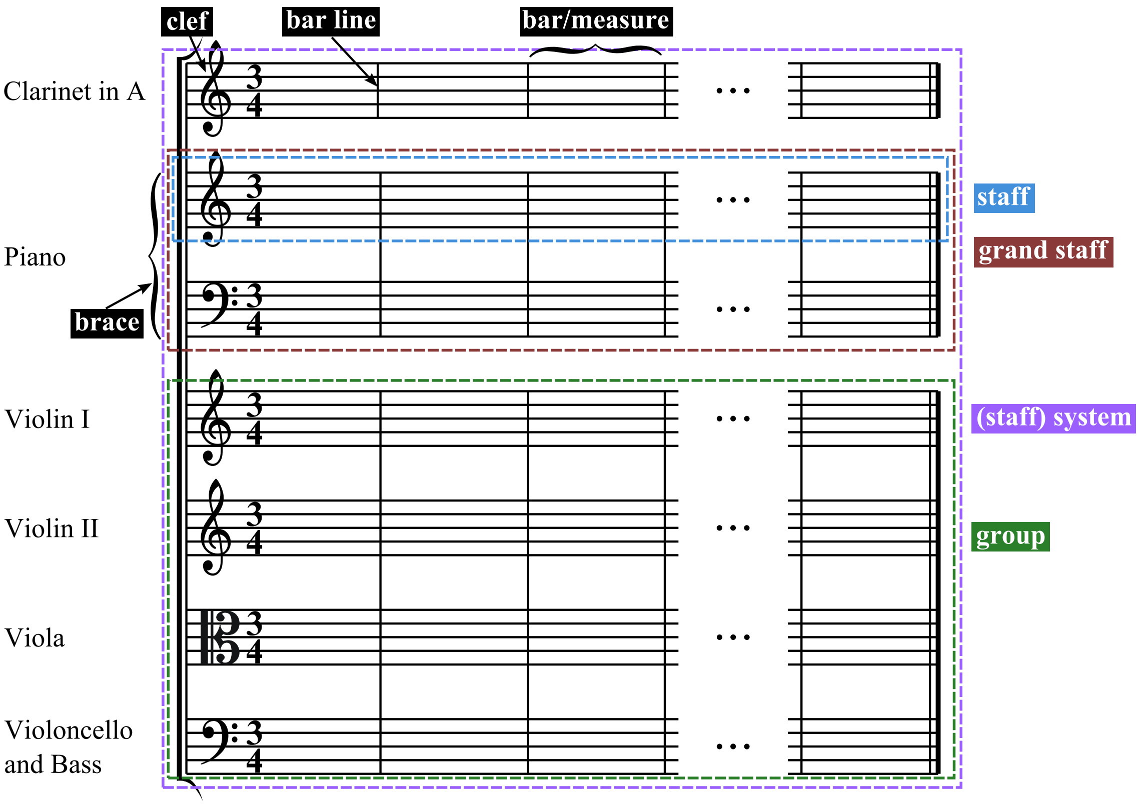


Figure 2.2. Example of CPN layout and naming conventions. The note material of an instrument is placed in a staff. To describe multiple simultaneously performing instruments, the individual staves are connected by a bracket to create a (staff) system. The score is segmented into measures (also bars), which are visually divided by measure lines. The time signature at the beginning of the score indicates the number of beats per measure. To deal with the different note ranges of instruments and the limited range provided by a staff, clefs are used. For example, for the treble clef (first staff) the middle line of a staff represents the pitch B4, whereas for the bass clef (third staff) this line coincides with the pitch D3.

in case of changes in comparison to the previous system. If the mapping is obvious for a human reader (e.g., strings are always the last instrument group), the textual information might even be omitted in spite of changes.

In addition to those text labels, braces, musical brackets, and instrument groups further help in structuring the score and determining the instrument-staff mapping. While braces mark grand staves or staves holding playing instructions for the same instrument (e.g., first staff for the first horn and the second staff for the second horn), instrument groups cluster whole instrument families. This grouping is indicated in the score by interrupting the vertical measure lines between two groups and/or by clustering the according staves with brackets, see Figure 2.2. These conventions for structuring and annotation orchestral scores are usually met in modern prints. However, for older prints we could observe a range of deviations. Examples are the usage of brackets instead of braces, the omission of measure line gaps, or their usage to only separate the strings from the rest of the orchestra. In Figure 2.7, we depict some examples of encountered instrument grouping conventions.

Besides the mere information on which staff of a system contains the playing instructions for an instrument, the instrument-staff mapping has a particular relevance in the context of

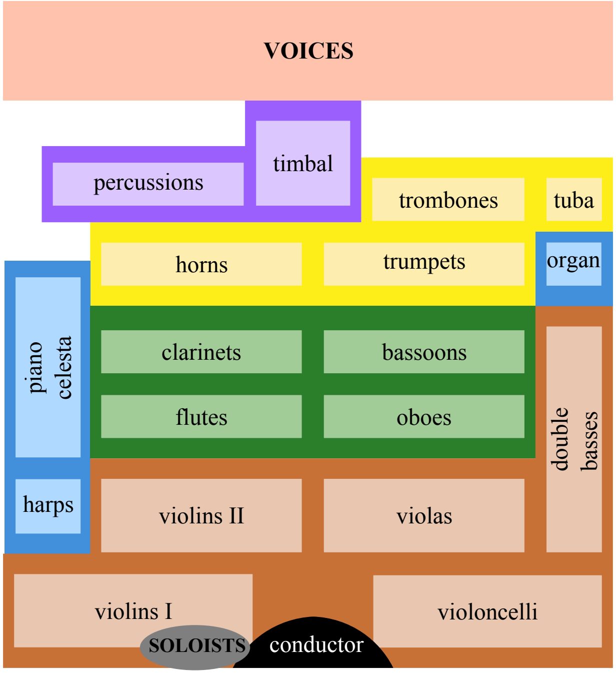


Figure 2.3. Common instrument placing in classical orchestras [130].

WOODWINDS	piccolo
	flutes
	oboes
	English horn
	clarinets
	bassoons
BRASSES	horns
	trumpets
	cornets
	trombones
	tuba
PERCUSSION	timpani
	other
PLUCKED	harp
KEYBOARDS	keyboard
SOLO	various
VOICES	choir
STRINGS	violins
	violas
	celli
	double basses

Table 2.1. Common order of instruments in CPN.

transposing instruments.[3] For these instruments, the sounding pitch differs from the notated pitch information in the score. More precisely, the produced pitch when playing a written C determines the interval of transposition for the specific instrument. A comprehensive listing of transposing instruments in Western classical music is available in Table 2.2. Note that for some instruments, the sounding pitch is still a C but in a different octave (e.g., the guitar sounds one octave higher than notated). The main motivation for the application of transpositions is a uniform fingering for the written notes of all instruments in the same instrument family – regardless of their individual size and pitch range.

Timpani are usually tuned to the tonic and dominant notes. In the 17th and early 18th century, timpani were often treated as transposing instruments as well. In transposed notation, they are notated as C and G with the tuning indicated by the text label (e.g., `"Timpani in Es-B"`). While Bach, Mozart, and in the beginning also Schubert used this notation, Handel, Haydn, and Beethoven chose to write the timpani score in concert pitch, see Figure 2.8.

Octave transpositions are either not marked in the score at all or are specified by an eight above or below the clef. In contrast, the non-octave transpositions are indicated by adding an according textual label to the instrument name, for example, `"Clarinet in A"` (denoting a three semitones lower sounding pitch). Depending on the language of the score and the editor, the names of keys, the font, and the layout of the textual labels vary, see Figure 2.9. Like instrument labels, transpositions are usually only indicated in the first system of a piece or in the case of a change (of transposition). These changes are indicated in the measure before the new transposition becomes valid through a text label,

[3] Actually, the transposition is a convention of CPN rather than a property of the instruments themselves. Nevertheless, it is common practice to refer to instruments for which the music is typically notated in transposition as transposing instruments.

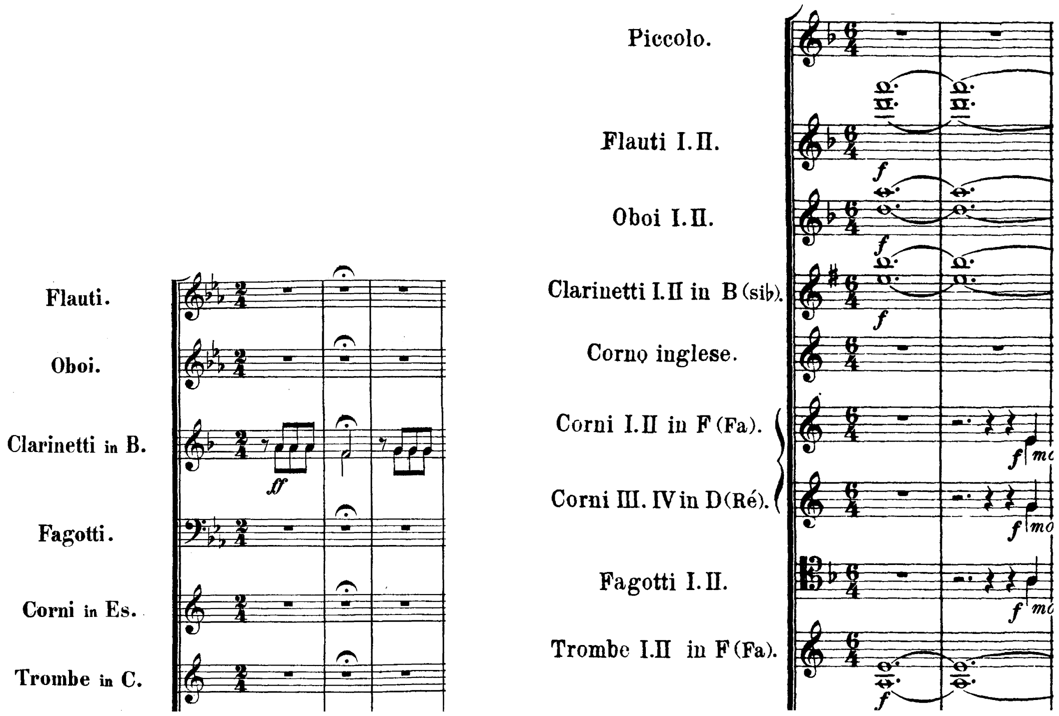


Figure 2.4. Instrument order of different scores. On the left, the beginning of the first movement from Beethoven's *Symphony No.* 5 in an edition by *Breitkopf & Härtel* is shown. The score utilizes the common instrument order as established in Table 2.1. On the right, the beginning of Wagner's opera *The Flying Dutchman* in a reprint edition by *Kalmus* is depicted. As in most orchestral pieces by Wagner we encountered, the order of the horns and the bassoons is altered.

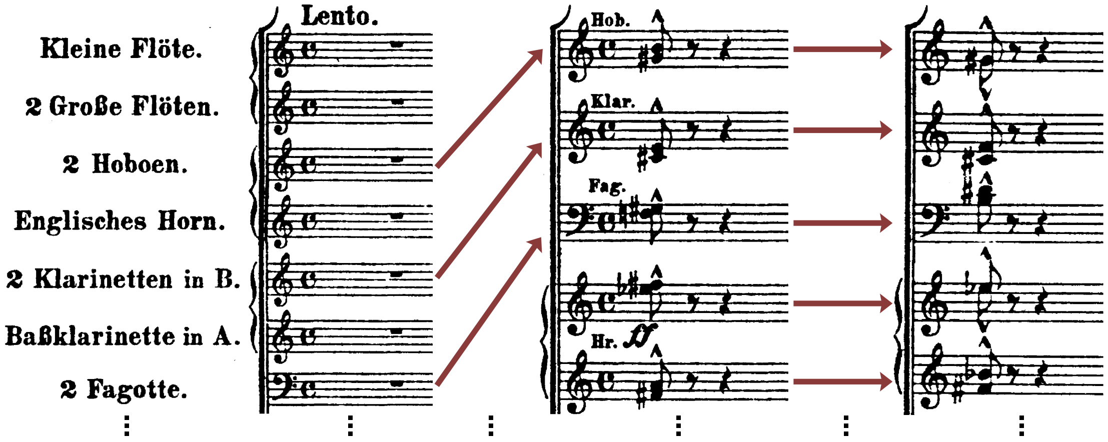


Figure 2.5. Extracts from Liszt's *A Symphony to Dante's Divine Comedy* in compressed notation (publisher: *Breitkopf & Härtel*). The overall order of the instruments in the score remains unchanged, but staves of instruments that have a long rest are temporarily removed. Thereby, the location of the score information related to a particular instrument changes. For example, in the first system the bassoons ("Fagotte" in German) are notated in the seventh staff, whereas in the two later systems their score is written down in the third staff.

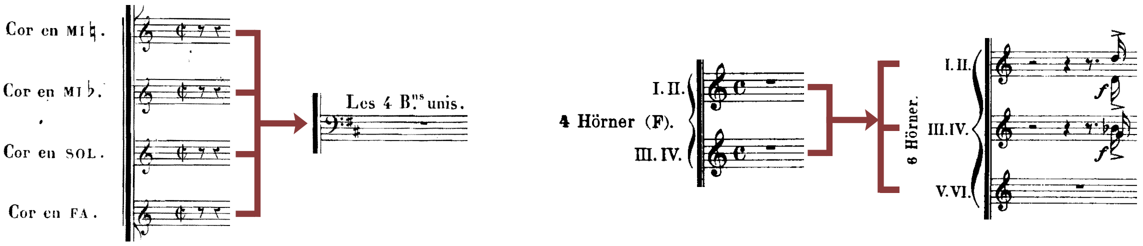


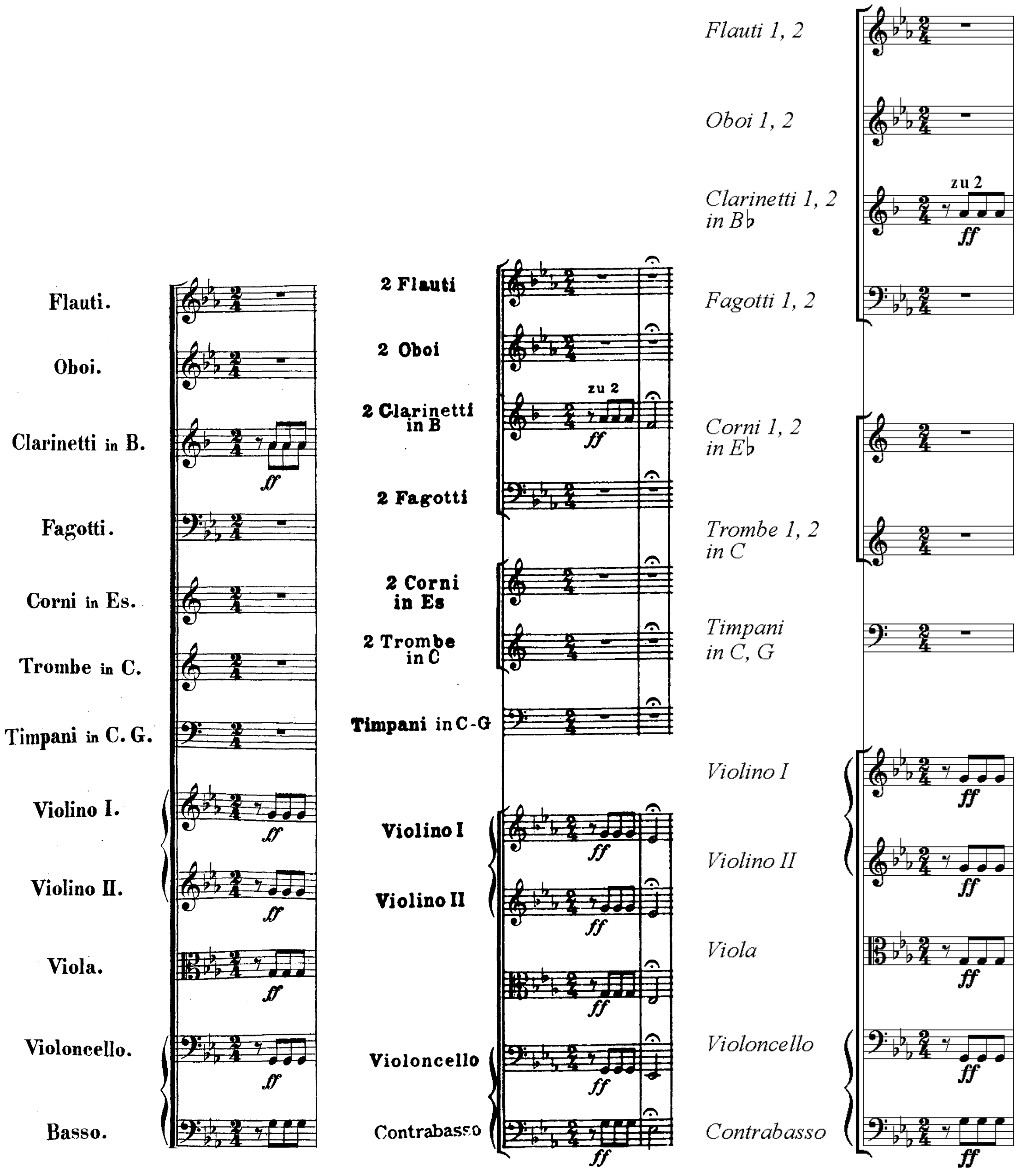
Figure 2.6. **Left:** Extracts from Berlioz's *Roméo et Juliette* (publisher: *C. Joubert & Cie.*) using compressed notation. The staves of the four horns are merged into one staff in a subsequent system of the score. **Right:** Example from *Thus Spoke Zarathustra* by R. Strauss (publisher: *Dover Publications*) where two additional horns appear in the second system of the score.

instrument	transpositions
piccolo	C, D♭
alto flute	G
oboe d'amore	A
English horn	C, F
piccolo clarinet	A♭
sopranino clarinet	D, E♭
soprano clarinet	D, E♭, G, A, B♭
clarinet	C, E♭, A, B♭
alto clarinet	E♭
bass clarinet	C, A, B♭
contra bass clarinet	E♭, B♭
horn	C, D, E, E♭, F, G, A, B♭
tenor horn	E♭
piccolo trumpet	E♭, B♭
trumpet	C, D, E, E♭, F, A, B♭
cornet	G, A, E♭, B♭
alto trumpet	F
bass trumpet	C, E♭, B♭
tuba	E♭, F, B♭
tenor tuba	E♭, B♭
bass tuba	F, B♭
flugelhorn	B♭
euphonium	B♭

Table 2.2. Transposing instruments in Western classical music and their possible transpositions. Instruments that have only octave transpositions are not listed.

e.g., `"(muta) in A"`, see Figure 2.10. Thus, to reconstruct which playing instructions in a system have to be read in transposition, both the instrument-staff mapping and the last given transposition information need to be considered.

Classical music is a highly structured art form, where whole sections are repeated to create a certain musical form (e.g., sonata form or the strophic form of songs). To save space and to highlight the structure of the piece in the score, CPN provides special symbols indicating repeating sections or jumps in the score. A comprehensive overview of repeat and jump instructions in CPN is available in Figure 2.11.



(a) *Breitkopf & Härtel* **(b)** *Dover Publications* **(c)** *MuseData* [45]

Figure 2.7. Example of different instrument groupings for the *Symphony No.* 5 by L. v. Beethoven as applied by different editors. **(a)** Shows a print by *Breitkopf & Härtel* without any groupings. **(b)** The reprint edition by *Dover Publications* visibly separates the strings from the other instruments. **(c)** In the *MuseData* score all instrument families are grouped and visibly separated by disrupted measure lines.

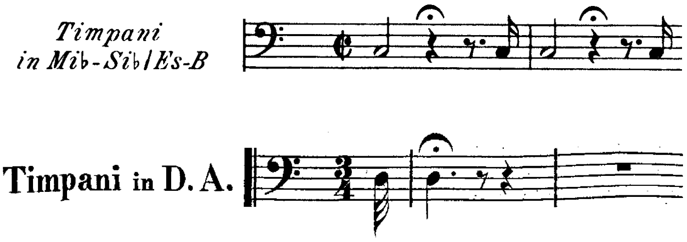


Figure 2.8. Different notation styles for timpani. In the upper extract from Mozart's opera *The Magic Flute* (publisher: *Bärenreiter*), timpani are treated as transposing instruments by notating the score in C and G while indicating the actual tuning through transposition labels in front of the staff. In the second example (extract from Beethoven's *Symphony No.* 2, publisher: *Breitkopf & Härtel*), the timpani are notated in concert pitch (albeit still indicating the tuning through textual labels).

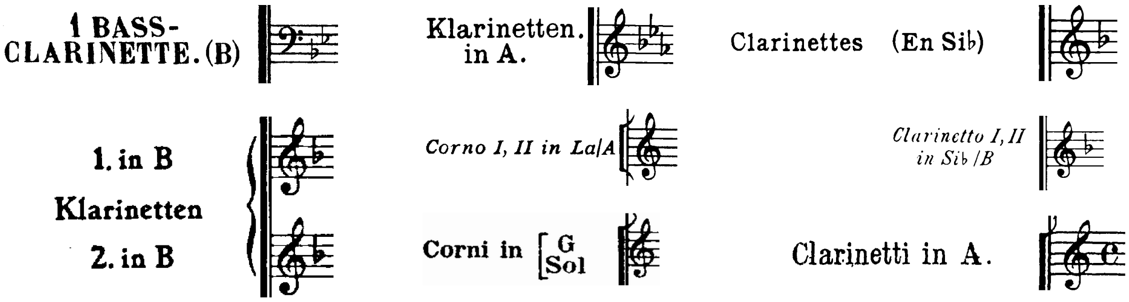


Figure 2.9. Examples of instrument and transposition labels applied by different editors and in different languages. The examples show that French, Italian, and Spanish scores use fixed do solmization (Solfège), e.g., the key A is referred to as "la".

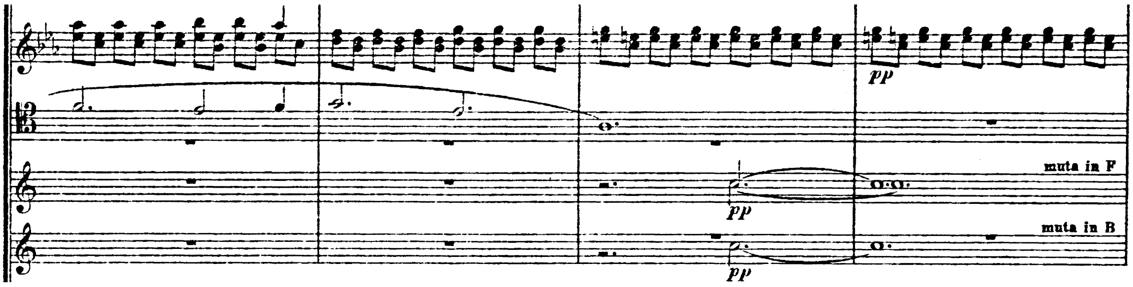
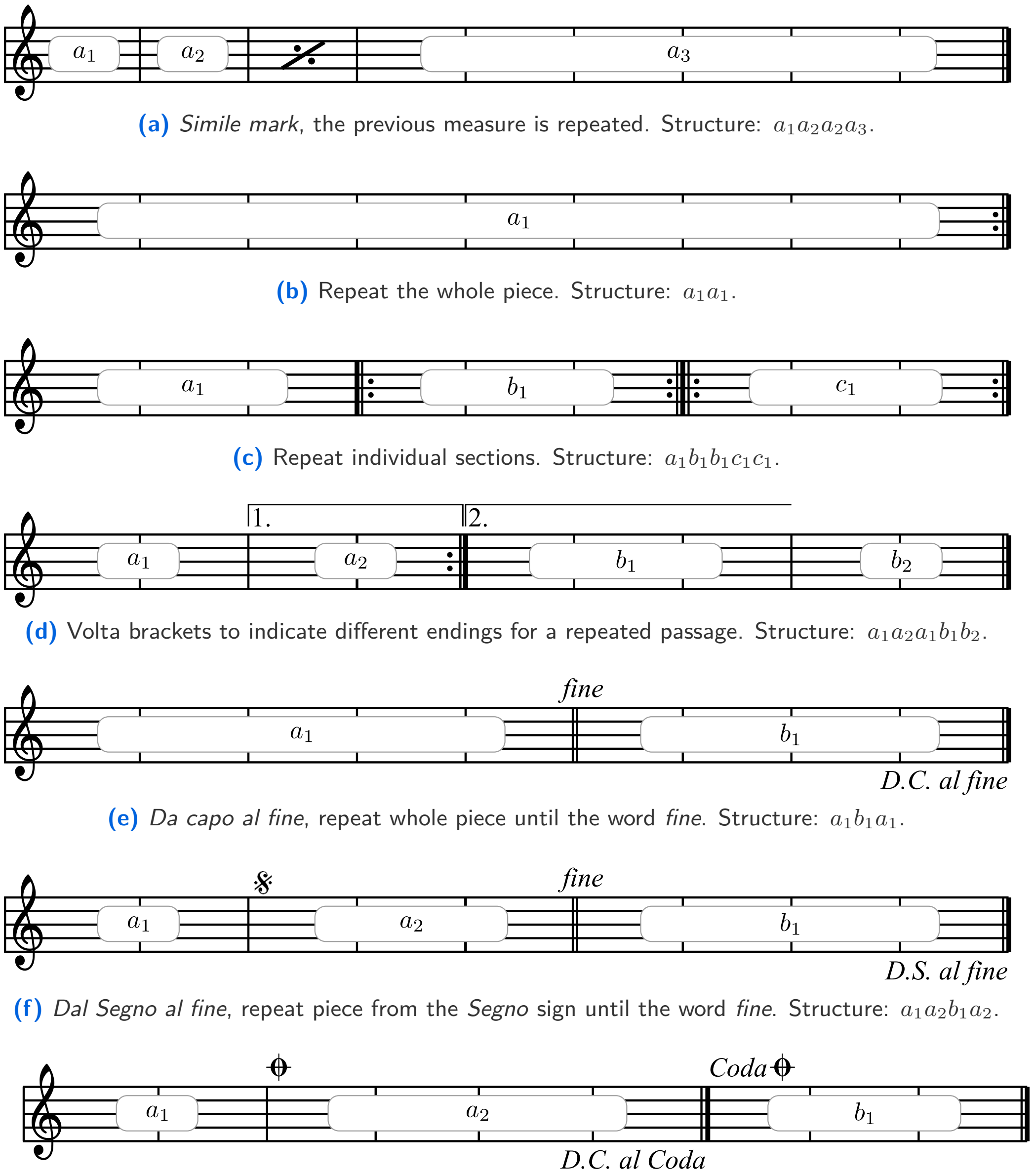


Figure 2.10. Examples of changes in transposition that take place within a piece of music (publisher: *Breitkopf & Härtel*). In the third staff, the transposition of the first and second horns is changed to F (previously in E) and in the fourth staff, the transposition of the trumpet is changed. As the instrument order did not change in this system, no instrument labels are given, thus requiring the reader to remember the instrument-staff mapping.



(a) *Simile mark*, the previous measure is repeated. Structure: $a_1a_2a_2a_3$.

(b) Repeat the whole piece. Structure: a_1a_1.

(c) Repeat individual sections. Structure: $a_1b_1b_1c_1c_1$.

(d) Volta brackets to indicate different endings for a repeated passage. Structure: $a_1a_2a_1b_1b_2$.

(e) *Da capo al fine*, repeat whole piece until the word *fine*. Structure: $a_1b_1a_1$.

(f) *Dal Segno al fine*, repeat piece from the *Segno* sign until the word *fine*. Structure: $a_1a_2b_1a_2$.

(g) *Da capo al Coda*, repeat from beginning and jump to the coda at the *Coda* sign. Structure: $a_1a_2a_1b_1$.

Figure 2.11. Jump indicators in CPN (adapted from [73]).

2.2 Digital Music Representations

Digital music representations can be divided roughly into *graphical*, *symbolic*, and *auditory score* formats. In this section, we briefly introduce each group and comment on their similarities and differences.

Graphical: Given a printed score in CPN, it can be digitized by scanning every page. As a result, a graphical representation as an image (PNG, TIFF, or JPEG) or a PDF file is created. For the human reader, the resulting digital document is as readable as the paper document. However, for the computer, these scans are merely a conglomeration of pixels without any musical meaning. In places that require a clear distinction, we explicitly state whether we are talking about printed or scanned score documents. Furthermore, we use the terms *score* and *sheet music* interchangeably.

Symbolic: The *symbolic score* class refers to digital, machine-readable data formats explicitly representing musical entities. Symbolic scores can differ largely in their structure and description level. Therefore, musical entities range from note events with explicit timing information as in MIDI files to graphical shapes with attached musical meaning, as in the `*.mro` files created by the OMR system SharpEye, see Section 2.3. Some well-established symbolic formats are MIDI, MusicXML, Humdrum, LilyPond, and NIFF. Various ways exist to create a symbolic score representation of a piece of music. First, one could manually type the code describing the score. However, this is a time-consuming and error-prone endeavor. Second, *music notation software*, such as MuseScore[4] or Capella,[5] provide convenient graphical user interfaces for the creation of digital scores. Most of these programs support the export of the resulting representations into the more popular symbolic formats. Finally, given the scanned score of a piece of music, the contained symbolic music information can be restored using *optical music recognition* software (OMR), the musical analog to *optical character recognition* (OCR), see Section 2.3. In the context of the PROBADO project, see Chapter 3, we employ OMR to enable the application of MIR techniques, while using the digitization of the printed score as visualization.

Audio: From a physical point of view, striking a tuning fork causes vibrations of the air that ultimately result in periodic changes of the air pressure at our ear drums. Basically, every sound we hear arises from such vibrations induced by some object, e.g., the aforementioned tuning fork, the vocal chords of a living being, or the vibrating string of a guitar. The induced pressure changes travel through the air as a *wave*. The time between two consecutive high pressure points of this wave is called the *period* and the amplitude of the wave corresponds to the *intensity* of the produced sound signal. The *frequency* is the reciprocal of the period, i.e., a measure of the amount of vibrations per seconds. The corresponding perceptual experience of frequency is referred to as *pitch*. For example, the frequency of the MIDI pitch 69, which is the middle A of the chromatic scale, has a frequency of 440 Hz.[6] A particular property of the chromatic scale is that octave changes result in a doubling/halving of the frequency. Therefore, A5 has a frequency of 880 Hz and the frequency of A3 is 220 Hz. The sound produced when playing a single tone on an instrument is not such a simple sound of a well-defined frequency. It rather consists of a superposition of sounds of various frequencies; the so-called *harmonics* or *overtones*.

[4] `http://musescore.org`, February 2013

[5] `http://www.capella.de/us/`, February 2013

[6] In practice, an entire frequency range is associated with a single pitch to smooth out small deviations.

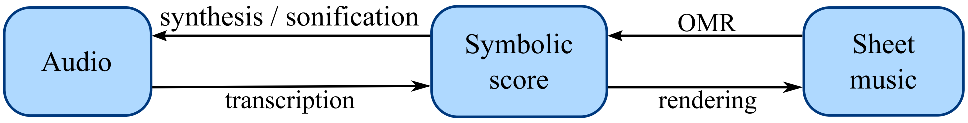


Figure 2.12. Illustration of transformations between the three types of digital music representations (adapted from [73]).

The frequencies of these harmonics differ by an integer multiple from the *fundamental frequency*, which is the frequency of the produced pitch. In addition to the harmonics, a tone usually also contains some non-periodic noise-like frequency components. The intensity distribution of the harmonics and the noise components strongly characterize the *timbre* of an instrument.

The sound waves produced by an orchestra can be captured with a microphone. Thus, the air pressure changes are converted into an electrical signal, which can subsequently be transformed into a digital, computer-readable representation. To this end, the recorded sound waves are sampled and quantized to produce a digital, discrete *audio signal*. The resulting signal can be stored as a digital *audio recording*. Common file formats are WAV and MP3. For more information on the production of sound and digital audio recordings, we refer to [119,133]. While an audio file captures all peculiarities of the recorded performance (room acoustics, deviations from the score, etc.), it does not provide sufficient information to allow a reproduction of the piece by other musicians (exceptions are simple pieces of music or exceptionally talented musicians/composers like W. A. Mozart[7]).

Having defined the three classes of digital music representations, transformations between them can now be specified, see Figure 2.12. By means of *audio synthesis* or *sonification*, symbolic score can be transformed into a synthetic audio recording. This task is basically well defined. However, the quality of the output strongly depends on the applied sonification method. The reverse transformation from an audio recording to its symbolic representation is called *audio transcription* [103] and poses a much harder problem. Especially for orchestral pieces, which feature several instruments, the reconstruction of the underlying score might not be possible at all. A given symbolic score file can be transformed into a visual sheet music representation by means of *score rendering*. Depending on the input format, this task varies in its complexity and solvability. For example, a MIDI file usually does not contain information on the key signature and clef of the piece of music and thus this information has to be reconstructed beforehand. However, an unambiguous reconstruction might not always be possible. In contrast, other symbolic formats like MusicXML or Humdrum explicitly represent all musical symbols and thus contain information on note durations, clefs, accidentals, and the current key. But due to the particular layout of CPN, score rendering remains a non-trivial task even for these formats. The reverse transformation from sheet music to symbolic score was already mentioned briefly and is referred to as OMR. In Chapter 4, we will employ this transformation to calculate sheet music-audio alignments. Therefore, a more detailed description of OMR is provided in the following section. The interested reader can find a more detailed description of the transformations in [73].

[7] According to the popular story, in April 1770, the 14-year-old Mozart produced a transcription of the *Miserere mei, Deus* by G. Allegri entirely from memory after listening to it once. Family letters support this story, see `http://www.freemedialibrary.com/index.php/Documents_describing_Mozart's_transcription_of_the_Allegri_Miserere`, February 2013.

2.3 Optical Music Recognition

Similar to OCR with the goal to reconstruct the textual information given on scanned text pages, OMR aims at restoring musical information from scanned score images.[8] However, the automatic reconstruction of music notation from scanned images has to be considered as being much harder than OCR. Music notation is two-dimensional, contains more symbols, and those symbols mostly overlap with the staves. A large number of approaches to OMR have been proposed and a range of commercial and non-commercial OMR systems are available today. Three of the more popular commercial systems are SharpEye [145], SmartScore,[9] and PhotoScore.[10] All of them operate on CPN. While the former two only work for printed sheet music, PhotoScore also offers the recognition of handwritten scores. Two prominent examples for non-commercial OMR systems are Gamera[11] and Audiveris.[12] While Audiveris is not competitive in terms of recognition rates, Gamera is actually a more general tool for image analysis. Therefore, Gamera requires training on the data to be recognized in order to yield adequate recognition results. Since the introduction of OMR in the late 1960s [155], many researchers have worked in the field and relevant work on the improvement of the recognition techniques has been reported. For further information, we refer to the comprehensive OMR bibliography by Fujinaga [78].

Three factors exist that affect the difficulty of the OMR task and the selection of the pursued approach. First, there exist different types of scores (e.g., CPN, medieval notation, or lute tablatures) that differ significantly in their symbol selection and their basic layout. Therefore, the type of music notation present on the images has to be considered. Second, the transcription format is of influence. Printed score is regular and usually well formatted, while handwritten score can be rather unsteady and scrawly. Additionally, crossing outs, corrections, and marginal notes make the interpretation of handwritten scores even more challenging. Finally, the envisioned application of the resulting symbolic representation influences the required precision. OMR results intended for playback or score rendering have to present a much higher accuracy on the note level than a reconstruction serving as score representation during sheet music-audio synchronization on the measure level, see Chapter 4, or similar MIR tasks. In the first scenario, most OMR systems support the creation of an initial approximation of a symbolic representation and provide user interfaces for manual correction.

Several studies on the performance of OMR systems and the types of errors that occur were conducted [33, 39, 40, 73]. Those studies showed that OMR systems vary with regard to their strengths and weaknesses. Nevertheless, the types or classes of recognition errors are the same for all systems. Some examples of common errors are given in Figure 2.13. Most of those errors are of local nature and concern individual music symbols or small groups thereof. Examples are articulation marks, ornaments, accidentals, dynamics, and note durations that are mistaken for some other symbol or missed altogether. But there are also types of recognition errors that influence larger areas of the score. Those might include incorrect time signatures or key signatures, missed clefs, staff systems that were

[8] The first four paragraphs of this section are to a great extent adopted from [194].

[9] http://www.musitek.com, February 2013

[10] http://www.sibelius.com/products/photoscore/ultimate.html, February 2013

[11] http://gamera.informatik.hsnr.de, February 2013

[12] http://audiveris.kenai.com, February 2013

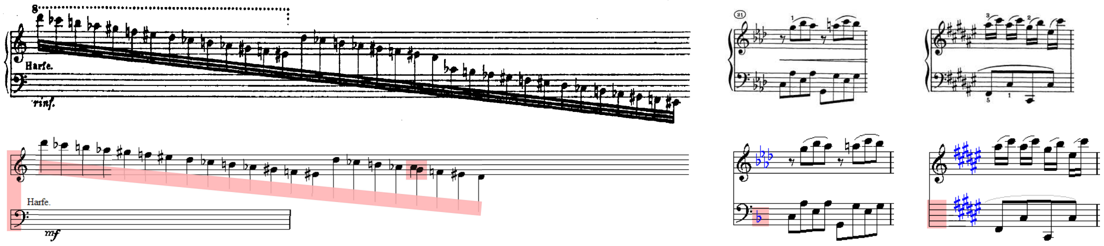


Figure 2.13. Examples of common OMR errors. **Left:** Besides incorrect note durations and an accidental that was mistaken for a note, the staff system was split into two systems and some notes were missed. **Middle:** The key signature was not correctly recognized for the lower staff. **Right:** In the lower staff, the clef was not detected.

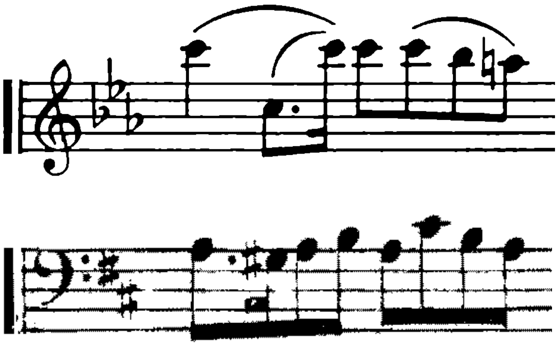


Figure 2.14. Comparison of a high-quality score scan (top) and a scan of lower image quality (bottom). The encountered scan quality has a strong impact on the error rate of the OMR output.

split up,[13] or missed repetition instructions. Two important factors influencing the overall amount of errors are the complexity of the score itself (e.g., large staff systems, closely placed notes, or many ornaments can render the recognition difficult) and the image quality of the score scans to be processed, see Figure 2.14.

Another shortcoming of most OMR systems is the interpretation of textual information in the score. While some systems are capable of determining text, such as the lyrics, correctly, text-based instructions on dynamics, title headings, and instruments are either recognized without associating their (musical) meaning or are not detected at all. For orchestral music to be recognized correctly, the most significant textual information is that on transposing instruments. If transposing instruments are part of the orchestra and their specific transposition is not considered during the reconstruction, their voices will be shifted with respect to the remaining score, see Figure 2.15. To the best of our knowledge, no OMR system considers this type of information and attempts its detection.

For the work presented in this thesis, the commercial OMR system SharpEye is used. Besides a command-line-capable recognition engine, a graphical user interface to access, edit, and correct the created recognition results is provided. SharpEye can export the created symbolic score data to MusicXML, MIDI, and NIFF. In addition, SharpEye supports a

[13] Systems might be split up due to arpeggios that travel through several staves, percussion staves in form of single horizontal lines, or textual annotations that disrupt the vertical measure lines.

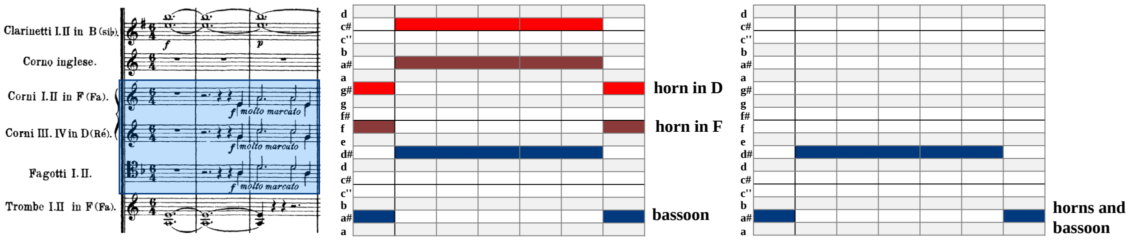


Figure 2.15. Voices of transposing instruments are shifted with respect to other voices if their transpositions are not known. **Left:** Score extract. **Middle:** Erroneous reconstruction in absence of transposition information (depicted in piano roll view). **Right:** Correct symbolic representation of the highlighted score extract.

proprietary text-based file format that uses the file extension `*.mro`. In contrast to the other supported formats, this format captures more low-level information. For example, `*.mro` files contain some information on the layout of the score, such as the placing of staves and measures in the original image.

Concluding, OMR systems are capable of providing good automatic interpretations of sheet music. Unfortunately, due to the complexity of CPN, not all information can be restored. Particularly challenging are scores of orchestral music. Here, information on instrument-staff mappings and transpositions is not reconstructed. As we will see in Chapter 4 this information is, however, crucially relevant for the calculation of high-quality sheet music-audio synchronizations of orchestral music.

2.4 Mid-Level Representations

As discussed in the previous sections, music comprises a wealth of information (instrumentation, articulation, dynamics, tempo, timbre, harmonics, etc.) and each data format captures them to a different extent and in a different manner. For most MIR tasks, certain aspects of music are of particular relevance, while others have to be disregarded altogether. For example, in some applications one may be interested in characterizing an audio recording irrespective of certain details concerning the interpretation or instrumentation. In contrast, other applications may be concerned with measuring a musician's individual articulation or emotional expressiveness. Also, automatic music processing often requires several music documents (that can be of a different type) to be compared with one another. Therefore, the first step in practically all music processing tasks is to extract a suitable *mid-level representation* (also *feature representation*) that captures key aspects relevant to the given task while disregarding those without relevance.

In the remainder of this section, we introduce the *chroma* features, a mid-level representation particularly useful in the context of music synchronization (Chapter 4) and music retrieval (Chapter 5).

2.4.1 Chroma Features

As is generally known, humans perceive two pitches as similar in "color" if they differ by one or more octaves. Using this periodicity, a pitch can be separated into the two components *tone height* and *chroma*, see [180]. Here, the chromas correspond to the 12 traditional pitch spelling attributes $\mathrm{C}, \mathrm{C}^\sharp, \mathrm{D}, \ldots, \mathrm{B}$ of the equal-tempered scale, while the tone height indicates the respective octave number of the pitch.[14] For example, the MIDI pitch $p = 37$ can be represented by chroma $\mathrm{C}^\sharp$ and octave number 2. The chroma representation of a music document is then given as a sequence of 12-dimensional chroma vectors where each vector captures the local energy distribution among those 12 pitch classes. By identifying and clustering pitches that differ by octave multiples in this manner, the chroma features show a high degree of robustness to variations in timbre and instrumentation while correlating closely to the musical aspect of harmony. Therefore, those features are particularly suited for the analysis of Western music, which is usually characterized by a prominent harmonic progression [15]. There exist both many variants of chroma features and many approaches for their computation. In this thesis, we focus on the description of a well-established chroma variant, the *Chroma Energy Normalized Statistics* (CENS) features. In particular, we describe their calculation for different digital music representations. Furthermore, we present an extension of those chroma features whereby their robustness towards timbre and instrumentation could be increased even further.

Given an audio recording, the digital music signal is first decomposed into 88 frequency bands whose center frequencies correspond to the pitches A0 to C8 (MIDI pitches 21 to 108).[15] This decomposition can be derived from the audio data in different ways, for example, by pooling Fourier coefficients obtained from one or more spectrograms [15, 67, 80], by using a constant-Q transform [31], or by applying multirate filter bank techniques [133, 141]. From this pitch representation the *short-time mean-square power* (STMSP) is calculated by squaring the samples of each sub-band using a rectangular window of fixed window size and overlap. In the next step, one obtains a chroma-pitch representation by summing up all STMSP values belonging to the same pitch class. For example, to compute the value of the chroma C, all values corresponding to the musical pitches $\mathrm{C1}, \mathrm{C2}, \ldots, \mathrm{C8}$ have to be added up. The resulting 12-dimensional chroma-pitch vector is subsequently normalized using the ℓ^1-norm.[16] Based on suitably chosen thresholds, the resulting normalized features are then quantized. By using logarithmic thresholds, one accounts for the logarithmic sensation of sound intensity [214]. Finally, we obtain CENS_d^w features through smoothing with a Hann window of length w, downsampling by a factor d, and a subsequent normalization with respect to the ℓ^2-norm. For more details on the individual steps, we refer to the literature [133, 141].

The transition from a scanned score to its chroma representation requires two additional processing steps [108]: First, the musical information is restored from the score images using OMR. Subsequently, the recognition result is used to create the note events of the score. Assuming a fixed tempo, uniform dynamics, and standard tuning,[17] a MIDI file is created. This file can then be converted into CENS in a similar way as described for audio recordings [91]. As previously discussed, the OMR extraction step is error-prone and consequently the quality of the produced chroma features might be degraded. While local errors can usually be canceled out by appropriate parameter settings (particularly the

[14] Note that different pitch spellings, such as $\mathrm{C}^\sharp$ and $\mathrm{D}^\flat$, are mapped to the same chroma.

[15] The selected pitches coincide with the 88 keys of a piano.

[16] For $p \in [1, \infty)$ the ℓ^p-norm of an n-dimensional vector $x = (x_1, x_2, \ldots, x_n)$ is defined by $\parallel x \parallel_p := \left(\sum_{i=1}^n |x_i|^p\right)^{1/p}$.

[17] For Western classical music a standard tuning of 440 Hz for the note A4 was defined. However, most orchestras slightly deviate from this tuning, see [150].

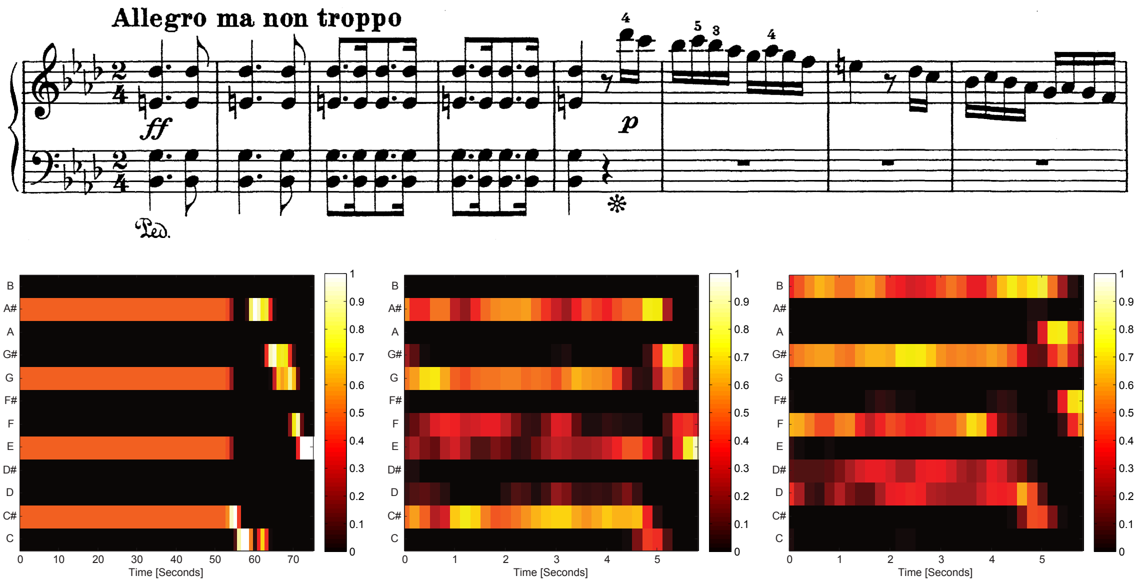


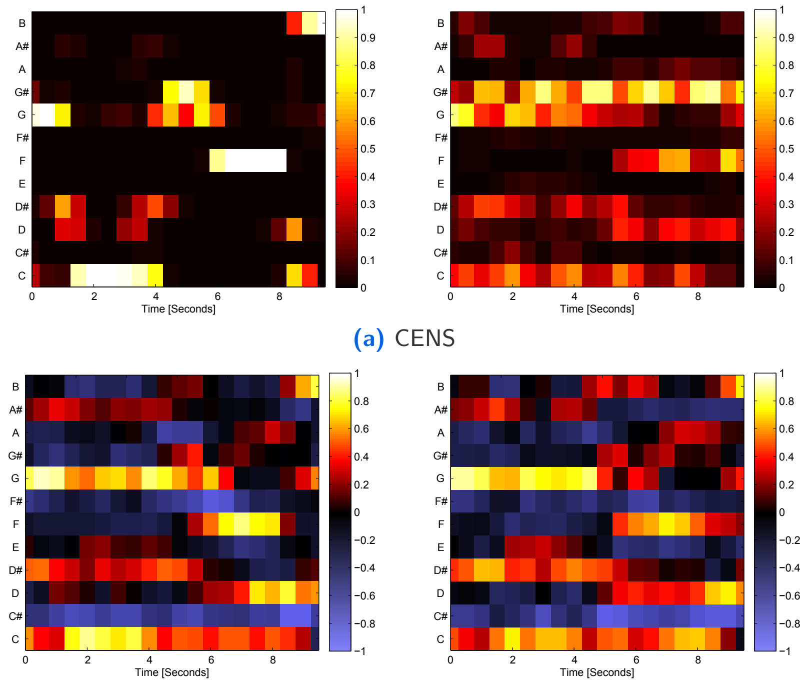
Figure 2.16. Illustration of the CENS chroma features for the first eight measures from the third movement of Beethoven's *Piano Sonata No.* 23 (publisher: *G. Henle Verlag*). The color values represent the intensity of a chroma at a given position (black: low intensity, red: medium intensity, and yellow/white: high intensity). The left diagram shows a chroma sequence that was calculated for the depicted sheet music extract. The middle and the right diagram show the chroma features for two audio interpretations of the same music extract. The chroma features manage to capture the higher tuning (by one semitone) of the second recording.

feature resolution), global misrecognitions, like missed jump instructions or transpositions, are propagated to the mid-level representation. To enable the robust comparison of sheet music with other music documents, those global errors should be identified and corrected. Some error classes particularly relevant in the context of sheet music-audio synchronization and approaches for their automatic detection and correction will be discussed in Chapter 4.

Figure 2.16 shows three CENS representations of the same musical section derived from different music documents (scanned score and two different audio recordings). A major difference of the audio-derived chromagrams compared to the chroma representation of the sheet music is their noisiness. This is mainly owing to harmonics and variations in dynamics. In contrast, harmonics are not modeled for the sheet music and a uniform volume – regardless of playing instructions and instrument-related variations – is assumed.

2.4.2 Timbre-invariant Chroma Features

It is a generally accepted observation that the lower *mel-frequency cepstral coefficients* (MFCCs) [63,118] are closely related to timbre [9,186]. Müller and Ewert [69,137] proposed to disregard information similar to those lower MFCCs for further increasing the timbre-invariance of chroma features without degrading their discriminative power. To this end, they first logarithmize the STMSPs using a nonlinear pitch scale (instead of the nonlinear mel scale used for the MFCCs). Then, a discrete cosine transform (DCT) is applied to obtain *pitch-frequency cepstral coefficients* (PFCCs). After removing the lower PFCCs (related to the aspect of timbre), the inverse DCT is applied. Finally, to yield the *chroma*



(a) CENS

(b) CRP

Figure 2.17. Comparison of CENS features **(a)** and CRP features **(b)** for the same extracts of the second Waltz of the *Jazz Suite No. 2* by D. Shostakovich. The chromagrams on the left represent the main theme played by the clarinet, and the chromagrams on the right correspond to the same theme performed by the trombone (source [69]).

DCT-reduced log pitch (CRP) features, the resulting pitch vectors are projected onto 12-dimensional chroma vectors and normalized with respect to the ℓ^2-norm.

Figure 2.17 compares the conventional chromagrams (CENS) to the corresponding CRP chromagrams. As an audio example, Shostakovich's second Waltz from the *Jazz Suite No. 2* in a recording conducted by Dimitry Yablonski is used. The theme of this piece occurs four times played in four different instrumentations (clarinet, strings, trombone, and tutti). In addition, the four occurrences exhibit significant differences in the instrumentation of the secondary voices and their dynamics (in relation to the melody). For the chroma comparisons presented in Figure 2.17, the clarinet and the trombone versions of the theme were considered. While the differences in instrumentation and voicing bring about that the CENS chromagrams deviate considerably, the corresponding CRP chromagrams coincide to a much larger degree.

3 PROBADO MUSIC

A Multimodal Digital Music Library

The Google Books project [81] is a large-scale initiative that aims at digitizing, organizing, and providing access to all books ever published.[1] Two of the most amazing features of this digital library are remote access to scanned versions of the books from any place in the world – provided an internet connection is available – and full-text search in the entire collection that even features the display of the exact match positions in the documents. In contrast to these advances in the textual domain, there is a lack of corresponding progress for general digital and digitized non-textual documents, such as audio recordings, images, videos, or 3D models. There exist several initiatives for the digitization of such documents with the goal of preventing their loss due to physical decay, and some attempts at making these collections available to the general public can be observed as well. However, what most or all of the systems reported so far lack are content-based search methods. In Google Books, a quote from a book suffices for retrieval. In contrast, digital library systems for non-textual documents are usually constrained to searches based on textual meta data.

The recently finished PROBADO project[2] was funded by the German Research Foundation (DFG). The project aimed at developing prototypes of digital library systems that provide innovative interfaces for content-based access and presentation of selected types of non-textual documents. The five involved institutions[3] concentrated their efforts on two document types, namely music documents (PROBADO MUSIC) and 3D architectural models (PROBADO 3D). For the music part, the Bavarian State Library was in charge of collecting, digitizing, and preparing a collection of music documents. At the Multimedia Signal Processing Group in the Department of Computer Science from the University of Bonn, the digital music library system was developed. Besides the front end for document access, a whole work flow for the preparation of music documents and their deployment was defined and implemented. Furthermore, various innovative tools for the automated processing and content-based access to music documents have been developed by the group. The research questions we discuss and address in this thesis arose in the course of the group's work on

[1] As of March 2012, more than 20 million of the approximately 130 million unique existing books worldwide have been scanned by Google.

[2] "**Pro**totypischer **B**etrieb **a**llgemeiner **Do**kumente" (engl.: Prototypical Operation of Common Documents), `http://www.probado.de/en_home.html`

[3] University of Bonn – Computer Science III, Multimedia Signal Processing; University of Bonn – Computer Science II, Computer Graphics; Technical University Darmstadt, Interactive Graphics Systems Group; German National Library of Science and Technology (TIB Hannover); and Bavarian State Library, Munich [16].

Probado Music. This chapter, therefore, provides some details on digital music libraries (Section 3.1) and Probado Music (Section 3.2) in particular.

3.1 Background to Existing Digital Music Libraries and Collections

Hankinson et al. [88] evaluated several digital music library systems [6,30,66,81,98,117,188, 197–201,208] with respect to their user interfaces and identified three main drawbacks that may be observed in most of them: First, they usually do not maintain document integrity and, for instance, present scores as a series of separate images. Second, simultaneous presentation of related music documents is often not possible. As a third drawback, the meta data of the currently selected music document cannot be accessed at a glance, omitting further valuable information.

Probably the worst shortcoming is the lack of simultaneous access to multiple documents as users are thereby restricted in their possibilities of experiencing a musical work. A piece of music has various representations, describing it on different semantic levels and addressing different modalities. Therefore, a digital music library system should offer access to as many different representations as possible. In the context of the ongoing digitization efforts for the preservation of our cultural heritage, several institutions/projects aim at establishing such multimodal digital library systems for general document types. Naturally, those collections also contain large amounts of music-related documents. For example, the Europeana project [68] offers open online access to a large collection of text, audio, video, and image documents of different European cultural institutions. A similar initiative for the digitization of European libraries is the project Quaero [156,157] that focuses on the same data formats. Further examples of multimodal general digital libraries are the Internet Archive [99] and the World Digital Library [201]. The Greenstone project [148,209] is another interesting project in the context of multimodal digital libraries. In contrast to the projects mentioned so far, Greenstone aims at offering tools for the creation and management of digital libraries. Some of its main features are the support of multimodal document collections, possibilities for content-based retrieval, a plug-in mechanism to add functionalities, and a basic, extensible user interface. Furthermore, a tool for the creation of a digital library from a given digital document collection was proposed in [11].[4]

In the music domain, many institutions by now provide online access to their collections. However, several of these institutions only hold music data of one format. Examples are the sheet music collections Chopin Early Editions [198] – based on the Greenstone system – and the Schubert-Autographe [208]. There also exist various digital audio collections of contemporary and public domain audio recordings, such as British Library Sounds [7], Piano Society [181], or the online music library of the Isabella Stewart Gardner Museum [95]. For symbolic score files similar offers exist, e.g., Kern Scores [102] (formats: Humdrum, MIDI and rendered score PDF), or Mutopia Project [146] (formats: Lilypond, MIDI, and rendered score PDF).

[4] The first two paragraphs of this section were in large parts taken from [57]. Portions of the rest of this section, e.g., the paragraph on the IEEE 1599 standard, take this publication as a basis as well.

The International Music Score Library Project (IMSLP or Petrucci Music Library) [87] started as a sheet music digitization and collection initiative providing access to public domain sheet music material. Since its launch in 2006, over 221,000 scanned scores for more than 60,000 works have been added to the collection (numbers are taken from [87], February 2013). Some time ago, the system was extended to also contain public domain audio recordings (approx. 19,000). Due to a work-based organization of the collections, users receive a direct overview of and fast access to all documents related to a certain work. Furthermore, a score viewer was recently added. Thus, the simultaneous playback of a recording while reading a score is made possible. However, the system lacks synchronization data and, therefore, cannot offer score-following and score-based navigation. To extend the meta data search by content-based retrieval, the Peachnote Music Ngram Viewer [203] was recently integrated. Thus, users can search for melodies, chord sequences, and rhythmic patterns. The Internationale Stiftung Mozarteum provides an online version of the Neue Mozart-Ausgabe (New Mozart Edition, publisher: *Bärenreiter-Verlag*) [197]. In addition to the scanned score – digitized by the Bavarian State Library – the online collection provides critical comments and audio recordings for most of the pieces. The web interface also allows for a visualization of the sheet music while one of the available interpretations is being listened to. However, automatic score-following is not available either.

The last two systems fulfill most of the requirements for digital music libraries established by Hankinson et al. [88]. Two further examples of such systems are Variations2 [66] and EASAIER [59, 109]. Variations2 is a digital music system intended for educational institutions. The main objective is the realization of shared access to music collections throughout a classroom. The system was established by Indiana University and has been in use since 1996. Besides user interfaces for the simultaneous visualization of meta data information, audio tracks, and sheet music, the system offers tools for manual music analysis (musical structure, musical beat). Furthermore, the manual creation of sheet music-audio alignments is supported. In [162] work towards automatic synchronization was reported. To offer enhanced search functionalities, a query-by-humming system was proposed [25, 26]. In the context of several follow-on projects (Variations/FRBR,[5] Variations3[6]), the system was converted to use a FRBR-compliant data model and distributed to other institutions (e.g., New England Conservatory, Ohio State University, and the University of Maryland). Variations on Video[7] is an ongoing research effort with the goal of adapting the Variations technology to manage and access both audio and video documents. EASAIER enables access and simultaneous visualization of various music representations like audio, score, and images. In addition to content-based search mechanisms, several different audio analysis and processing tools are available, e.g., time stretching and source separation. In the project goals,[8] *'multi-media synchronization'* is mentioned, but so far no details on this topic have been reported.

The social website `musescore.com` allows users to access and download digital scores created with the free notation software MuseScore (available at `http://musescore.org`). In contrast to the previously mentioned projects and websites, `musescore.com` already utilizes synchronization techniques to enhance the document presentation in a similar way as proposed in PROBADO MUSIC (in particular, score-following and score-based navigation).

[5] `http://www.dlib.indiana.edu/projects/vfrbr`, February 2013

[6] `http://www.dlib.indiana.edu/projects/variations3`, February 2013

[7] `https://wiki.dlib.indiana.edu/display/VarVideo`, February 2013

[8] `http://www.elec.qmul.ac.uk/easaier/index-1.html`, February 2013

However, musescore.com exploits the symbolic score information from the MuseScore documents and a MIDI sequencer to provide sonifications, which is why no alignments have to be calculated. An additional feature are the Videoscores where MuseScore documents are presented together with YouTube videos. In this scenario, alignments have to be created to support score-following and score-based navigation. The video tutorial "How To Synchronize Your Score With A Youtube Video – MuseScore"[9] demonstrates that these have to be created manually by the users. To the best of our knowledge, no future plans regarding automatic music synchronization have been mentioned so far.

Recently, a new standard – IEEE 1599 – to encode music with XML was published [120]. The new format offers the possibility to combine all information related to a musical work (different audio interpretations, scores, lyrics, images, annotations) in a single XML file. The standard also provides for the possibility of including synchronization information [54] (currently manually created) as well as MIR models [154] to the XML file. Using this standard, user interfaces for the holistic presentation of musical works were proposed, see, e.g., [10]. Work towards a web player that supports multimedia document access and provides three different interface sections (Enjoy, Interact, and Create) was reported in [12, 13].

Despite a great deal of activity in the field of digital music libraries, to our knowledge, no project has reported on performing synchronization of scanned score and audio data on a larger scale. However, with multiple documents available, it is an obvious goal to pursue. We will now report on the PROBADO MUSIC project where automatic measure-wise sheet music-audio synchronization is performed for large document collections. The resulting alignments are subsequently used to provide a novel, truly multimodal interface for document presentation. Furthermore, in PROBADO MUSIC a complete processing chain from document acquisition to document access was designed, implemented, and tested in a real-life library setting.

3.2 The PROBADO MUSIC System

In PROBADO MUSIC, a prototype of a digital music library system that incorporates content-based retrieval and provides innovative user interfaces for document access was developed. More precisely the project goals can be summarized as follows.

- Establishment of a (semi-)automated document management process.
- Inclusion of state-of-the-art MIR techniques.
  - **Content-based music retrieval:** provide efficient indexing methods and user interfaces for the formulation of queries.
  - **Music synchronization:** perform pair-wise synchronization between all documents that represent the same piece of music (during preprocessing).

[9] http://www.youtube.com/watch?v=Eya3eQfjvzs&feature=youtu.be, February 2013

- Development of an innovative user interface for remote access to music data. The system should include means for content-based retrieval via the query-by-example paradigm[10] and multimodal document visualizations using the precalculated linking structures.
- Installation of a prototype of the developed system.

The Bavarian State Library [16] in Munich, Germany, and the Multimedia Signal Processing Group at the University of Bonn, Germany,[11] were the two project partners involved in Probado Music. While the Bavarian State Library digitized and collected large parts of their music documents and created the according meta data, at the University of Bonn algorithms and tools for the (semi-)automated organization of large music collection and their presentation were developed. The devised systems were constantly tested at the Bavarian State Library and improved by means of a close feedback loop between the project partners. The idea for Probado Music and the basic synchronization and matching techniques are based on work by Michael Clausen, Frank Kurth, and Meinard Müller. In the course of the project, several developers were involved in the implementation of the system architecture, in particular David Damm, Christian Fremerey, and the author of this thesis. Several components were developed in close cooperation between the developers, and thus a detailed description of the individual contributions might be difficult. Overall, David Damm made tremendous contributions to the general architecture of Probado Music and had a leading role in the implementation of the server component – which we do not further discuss in this thesis – and the Probado Music web interface, see [55, 57]. Christian Fremerey developed first versions of the Probado Music front end for document access and the management tool Macao, see [73]. The author of this thesis joined the project at an advanced stage and took over the work on the Probado Music front end and Macao. Significant contributions are the conversion of the front end into a Java applet that can be integrated into the Probado Music web interface, the integration of lyrics and video into the front end and the extension of Macao with new editing masks for the correction of OMR errors and the repeat structure of the music documents.

The primary music representations currently considered in Probado Music are sheet music and audio recordings of Western classical music. Over the course of the project, the Bavarian State Library digitized a total of approx. 72,000 score pages and roughly 6,600 audio recordings (≈ 491 hours). During the preparation of the collection, the sheet music data are processed by the OMR system SharpEye to reconstruct the contained music information. In the process, most of the text in the sheet music – in particular the lyrics – is reconstructed as well. Thus, by means of appropriate postprocessing methods the lyrics can be extracted and provided as additional music representation [55, 174].

To offer conventional text-based search, a rich set of meta data annotations was prepared by the Bavarian State Library. To this end, Probado makes use of the popular Functional Requirements for Bibliographic Records (FRBR) model [94]. FRBR is an entity-relationship model where the data are organized in the four main entities: *work*, *expression*, *manifestation*, and *item*. In addition, information on contributors, such as

[10] Here, queries are created by example extracts from a music document such as an audio snippet, a score extract, or a text string from the lyrics.

[11] `http://www-mmdb.iai.uni-bonn.de/index.php?language=en`, February 2013

composer, performer, or publisher, can be modeled. As the above-mentioned entities will be encountered again in this thesis, we summarize the most important information below.[12]

Work: In the music context, a work denotes a piece of music as an abstract entity, i.e., independent of any performance or score representation. The work is, so to say, the mental idea we conjure when thinking of Schubert's song *Halt!* from the cycle *Die Schöne Müllerin.* To meet the special requirements of music collections, the FRBR model had to be extended in some places. The most important modification concerns the work entity. As our example already demonstrates, pieces of music often present a hierarchical structure. There is the whole song cycle, which is a work by Schubert, but equally each song constitutes an individual subwork of the cycle. To describe this hierarchy, the relation `"part_of"` between works was added. Through this relationship, each work can contain several (child) works and can itself be contained in one distinct parent work.

Expression: An expression denotes a particular performance or score edition of a work as an abstract entity. Examples would be the performance of the song *Halt!* by Ian Bostridge and Mitsuko Uchida or the score of the song as published by *C. F. Peters.*

Manifestation: A manifestation is a particular physical realization of an expression. For the music representations considered in PROBADO MUSIC, therefore, a manifestation is an audio CD (or a whole CD collection) or a printed score book. According to this definition, a manifestation can and usually does comprise several expressions (of different works).

Item: An item represents a particular copy of a manifestation. For example, a CD copy of the *Schöne Müllerin* performed by I. Bostridge and M. Uchida and released in 2005 by *EMI Classics* stored at the Bavarian State Library denotes a different item than a CD copy of the same performance located at the British Library.

For more detailed information on the PROBADO MUSIC database model we refer to [64].

To realize the targeted goals, the PROBADO MUSIC system was implemented as a classic three-layer architecture comprising a repository layer, a server layer, and a presentation layer. In the repository layer, search indexes, meta data annotations, linking structures, and derived data for document dissemination are created in an offline preprocessing step. To this end, the document management system MACAO was designed and implemented. In Section 3.2.1, details on the designed preprocessing work flow and the MACAO system are provided. In the server layer, the delivery of the musical content to the user is handled. Furthermore, access to the precalculated index structures (to process content-based queries) and the linking structures is processed here. For details on this layer, we refer to [55]. Finally, the presentation layer consists of user interfaces for document presentation, content-based search, navigation, and browsing within the documents and the collection as well as synchronized playback of audio, sheet music, and lyrics. The PROBADO MUSIC front end is introduced in more detail in Section 3.2.2. Further information on the PROBADO MUSIC project and the individual system components is available in [55–58, 73, 106, 190].

[12] Parts of the provided information on the PROBADO MUSIC database model are based on the descriptions in [73, Section 4.1].

3.2.1 The Music Administration System MACAO

The administration system Macao ("**M**usic **A**dministration for **C**ontent **A**nalysis and **O**rganization") implements a process chain for digitizing, processing, organizing, annotating, indexing, and linking digital music collections. The system was developed in a close feedback loop between users and developers to create a clearly defined and optimized work flow. The concept for Macao as well as the first version were created by Christian Fremerey, see [73]. In the context of this thesis, several new features have been added to the system. The two main contributions are the editing masks for OMR errors and jump instructions, see Figures 3.8 and 3.9. In this section, we nevertheless provide a full description of the system to give the reader an idea of the essential work flow for creating, preparing, and managing a music collection.

As input data, the system accepts scanned scores (as PDF or TIFF images) and audio recordings (as WAV and/or MP3 files). Furthermore, Gracenote data[13] can be added in order to help with the segmentation and mapping of the audio content. The given input data are then organized and prepared by complying with the following steps.[14]

Meta data annotation: In cooperation with the Bavarian State Library, an entity-relationship model based on the FRBR model [94] was developed. Using this model the meta data information of the music collection is created. To help with this manual step, Macao provides convenient input masks for adding and editing work and manifestation entities, see Figures 3.1 and 3.3.

Dissemination preparation: To enable streaming and presentation of music documents, derived file types need to be created (e.g., textures for the score and audio visualizations or MP3 files). In addition, several file types that are only required for the subsequent preprocessing steps are derived from the input data. Examples are symbolic score data and JPEG versions of the score scans in lower resolution (for visualization purposes in Macao). Upon adding a CD or a score book to the collection, these derived file formats are created completely automatically.

Content extraction: Given scanned sheet music pages, their musical content has to be reconstructed using OMR techniques. The resulting symbolic score formats contain all music-related information available on the scanned images. The lyrics of pieces containing voice parts are usually recognized by the OMR system as well. In Probado Music, this information is used as the lyrics data presented to the user. Thus, the additional effort of finding and digitizing libretti can be avoided. For the upcoming music synchronization and indexing, score documents and audio files need to become comparable. Therefore, they are converted into a common mid-level feature representation. For the given data types and the intended MIR tasks, chroma features are a well-suited representation, see Chapter 2. Their calculation can again be performed fully automatically and no user interaction is required.

Segmentation and work identification: The content of a new music document has to be split into individual segments, each associated with a single work. Afterwards, the according meta data entries of the pieces of music have to be mapped to the segments. Automatic segmentation techniques [73, 74], filters, and input masks support the user in

[13] `http://www.gracenote.com/`, February 2013

[14] The description of the work flow is taken from our publication [190].

accomplishing this task. In particular, MACAO provides the user with access to the score images and an integrated audio player to enable fast and convenient manual validation and editing of segmentations, see Figures 3.4 and 3.5.

Synchronization: Music synchronization techniques, see Chapter 4, are employed to enable score-following and score-based navigation. Once the input data were correctly associated to the pieces of music, the linking data are calculated without requiring further user interaction. Using the sheet music-audio synchronization results in combination with the lyrics extracted from the score scans, lyrics-audio synchronizations are created quickly as well.

Content-based indexing: The indexes for content-based search are calculated fully automatically. In Chapter 5, content-based audio retrieval is discussed in more detail. For further information on score retrieval and lyrics retrieval we refer to [55, 57].

Revision: The employed synchronization method can produce erroneous linking structures, which will result in a poor music presentation by the PROBADO MUSIC front end. The main error source is introduced by the OMR process. Although the recognition rates of current OMR systems are already remarkable, they will probably never be perfect. Fremerey [73] identified a set of critical error classes that have a strong influence on the synchronization result (e.g., unrecognized or misrecognized transposing instruments, accidentals, and clefs), see Section 2.3. In the context of this thesis, we created a graphical interface for validating and editing the according score information, see Figure 3.8. Furthermore, performance-related deviations in the repeat structure can occur and might require manual rework. Figure 3.9 shows the interface for the correction of jump instructions, which we have developed.

While the above list properly represents the required steps in preparing a music collection, they may not reflect well which individual operations have to be performed by a user. For example, all derived files are created completely automatically. Given a scanned score and matching audio recordings, we now walk through all of the required steps from a user perspective.

- All work entries for the pieces given in the score or the audio have to be created. This also includes the generation of parent works. In Figure 3.1 the input mask for work entries is depicted.
- The score manifestation is created next. This task involves several steps.
  - First, the files containing the music document have to be specified via the interface in Figure 3.2. After choosing to create the new manifestation, all required derived documents, such as the OMR output, textures, and low-resolution JPEGs, are created fully automatically.
  - Subsequently, the manifestation editing mask shown in Figure 3.3 appears. Here the user has to add the meta data for the new manifestation.
  - Finally, the manifestation has to be segmented into individual expressions. The user can either create the expression entries manually or use a basic segmentation algorithm to obtain a proposed division. Then, the user has to either set or check the boundaries of each expression through the score visualization shown in Figure 3.4.

After mapping an expression to the correct work entry, its meta data, e.g., creation date and contributors, have to be edited.

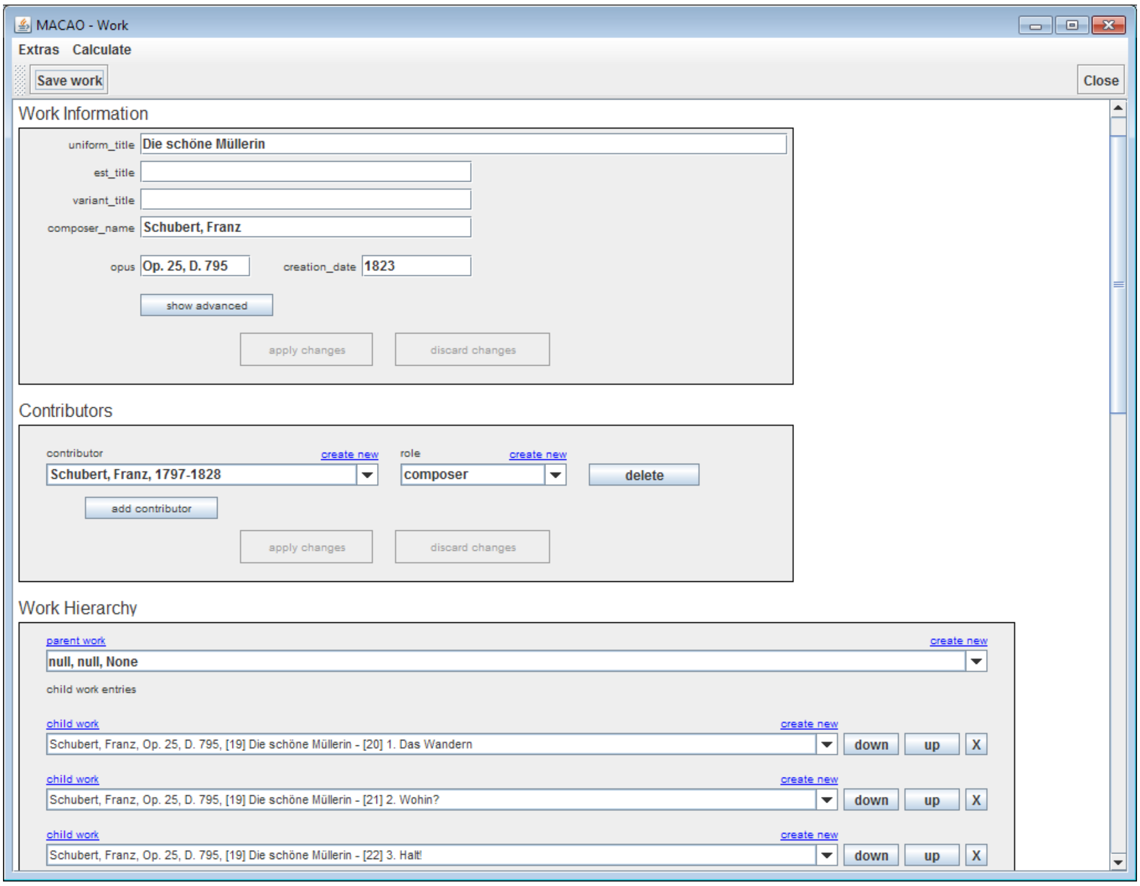
- To create the audio manifestation, the user has to proceed in the same manner as for the score manifestation. Instead of a visualization of the score, the expression editing environment provides an integrated audio player, see Figure 3.5.

At this point, the new documents are already properly accessible through the Probado Music front end. However, they cannot yet be searched by means of content-based search mechanisms and their views are not aligned to provide score-following and score-based navigation. Therefore, the following steps are necessary.

- The search indexes for audio matching, lyrics search, and symbolic search have to be updated, see Figure 3.6. In addition the browsing tree, see Figure 3.10d, has to be recreated.
- The synchronization files have to be created next, see Figure 3.7. For convenience, three possible choices exist: delete all existing synchronizations and create a fresh alignment index, create only alignments for missing pairs of expressions that are associated to the same work, or select a list of works whose synchronizations should be (re-)created.

Now, the new manifestations are properly accessible in Probado Music. If the quality of the created alignments is not acceptable, the user can go back and do some revisions on the input data.

- Critical OMR errors can be corrected via the interface shown in Figure 3.8.
- The jump instructions in the score might have been recognized incorrectly or the repeat structure of the audio can deviate from the one in the score. In this case, the respective jump instructions can be checked and validated, see Figure 3.9.
- After correcting the OMR data and/or the jump instructions, the affected alignments and search indexes have to be recreated.



Work Information

uniform_title: Die schöne Müllerin

est_title:

variant_title:

composer_name: Schubert, Franz

opus: Op. 25, D. 795

creation_date: 1823

Contributors

contributor: Schubert, Franz, 1797-1828

role: composer

Work Hierarchy

parent work: null, null, None

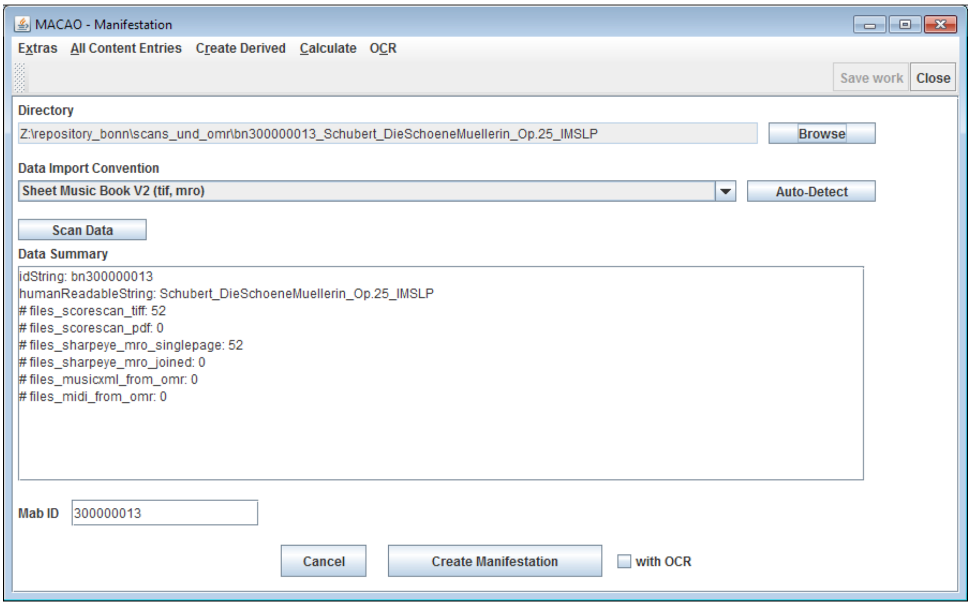
child work entries

child work: Schubert, Franz, Op. 25, D. 795, [19] Die schöne Müllerin - [20] 1. Das Wandern

child work: Schubert, Franz, Op. 25, D. 795, [19] Die schöne Müllerin - [21] 2. Wohin?

child work: Schubert, Franz, Op. 25, D. 795, [19] Die schöne Müllerin - [22] 3. Halt!

Figure 3.1. Macao input mask for creating and editing work entities. Here, the meta data for the song cycle *Die schöne Müllerin* by F. Schubert are depicted. The mask for work entries consists of three sections. The *Work Information* section provides text fields for adding or editing general information, such as title, opus number, creation date, etc. In the next box, *Contributors* can be added to the work. Each contributor also has a role, such as composer, librettist, or lyricist, assigned. The currently edited work represents the whole song cycle and is thus the parent work of all individual songs that are part of the cycle. These hierarchical relations between various work entities are managed via the last box of the depicted mask.



Directory
Z:\repository_bonn\scans_und_omr\bn300000013_Schubert_DieSchoeneMuellerin_Op.25_IMSLP

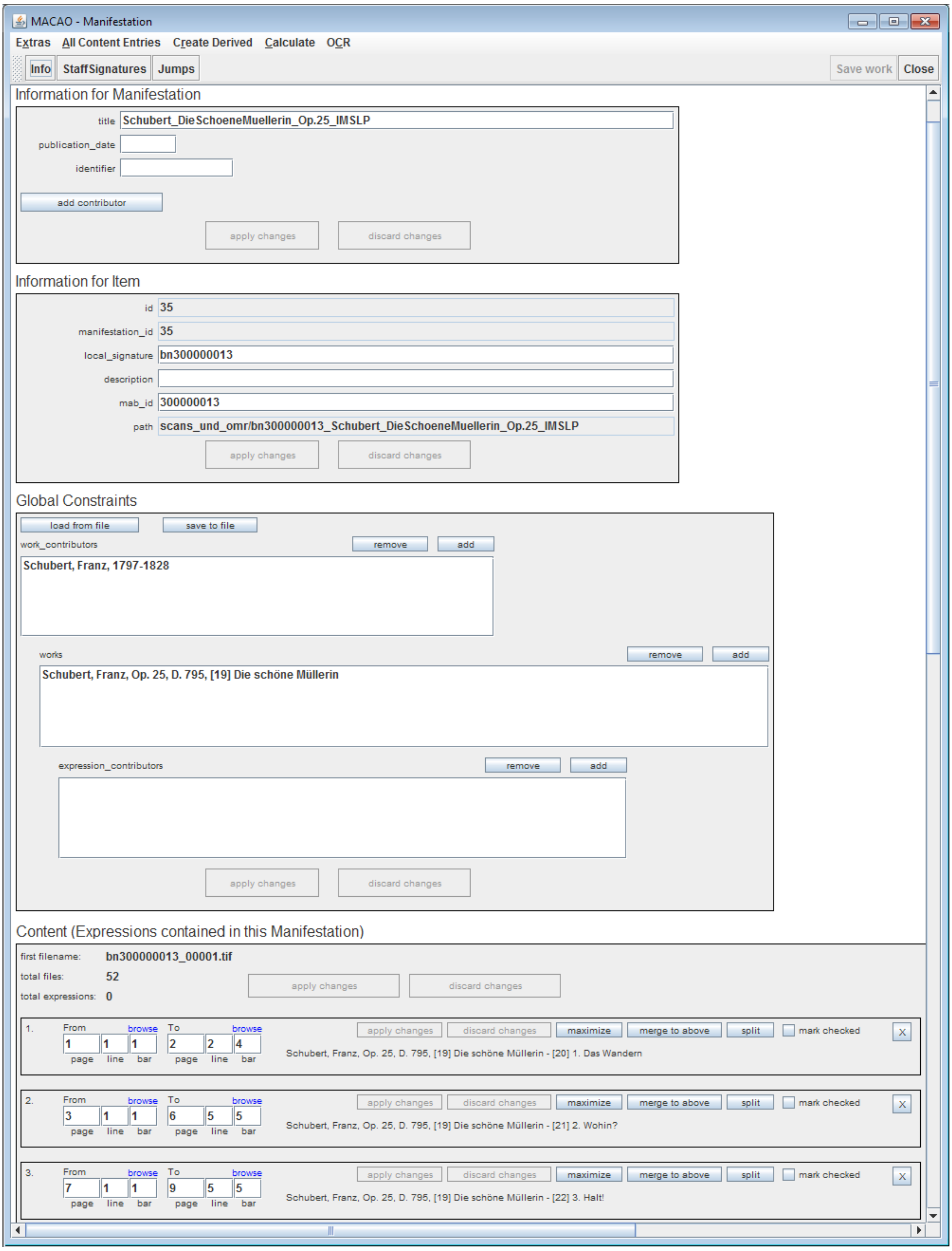
Data Import Convention
Sheet Music Book V2 (tif, mro)

Data Summary

```
idString: bn300000013
humanReadableString: Schubert_DieSchoeneMuellerin_Op.25_IMSLP
# files_scorescan_tiff: 52
# files_scorescan_pdf: 0
# files_sharpeye_mro_singlepage: 52
# files_sharpeye_mro_joined: 0
# files_musicxml_from_omr: 0
# files_midi_from_omr: 0
```

Mab ID 300000013

Figure 3.2. Macao interface for adding a new manifestation to the collection. Here, the TIF images and OMR files of a score version of *Die schöne Müllerin* have been selected. Upon creating the new manifestation, the steps *dissemination preparation* and *content extraction* are performed fully automatically. Afterwards, the editing mask depicted in Figure 3.3 appears.



Information for Manifestation

title: Schubert_DieSchoeneMuellerin_Op.25_IMSLP
publication_date:
identifier:

Information for Item

id: 35
manifestation_id: 35
local_signature: bn300000013
description:
mab_id: 300000013
path: scans_und_omr/bn300000013_Schubert_DieSchoeneMuellerin_Op.25_IMSLP

Global Constraints

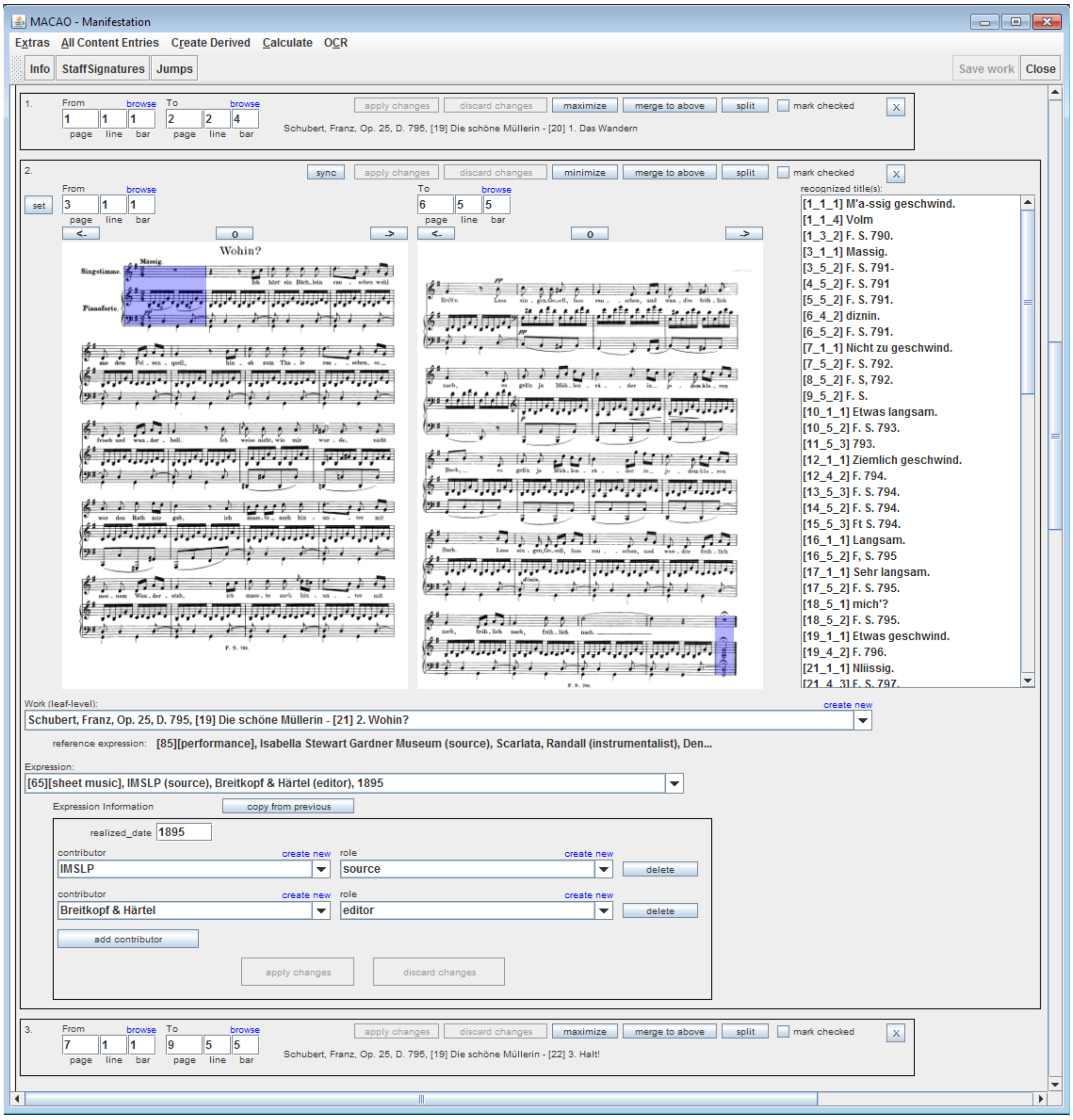
work_contributors: Schubert, Franz, 1797-1828
works: Schubert, Franz, Op. 25, D. 795, [19] Die schöne Müllerin
expression_contributors:

Content (Expressions contained in this Manifestation)

first filename: bn300000013_00001.tif
total files: 52
total expressions: 0

	From page	From line	From bar	To page	To line	To bar	
1.	1	1	1	2	2	4	Schubert, Franz, Op. 25, D. 795, [19] Die schöne Müllerin - [20] 1. Das Wandern
2.	3	1	1	6	5	5	Schubert, Franz, Op. 25, D. 795, [19] Die schöne Müllerin - [21] 2. Wohin?
3.	7	1	1	9	5	5	Schubert, Franz, Op. 25, D. 795, [19] Die schöne Müllerin - [22] 3. Halt!

Figure 3.3. Mask for manifestation editing in MACAO. The figure shows the already edited sheet music manifestation of *Die schöne Müllerin*. The two upper grey areas contain general meta data on the manifestation. In the box labeled *Global Constraints*, filters can be added to help with the segmentation and annotation process. Underneath there follows a list of all expressions contained in the manifestation with individual foldout editing masks, see Figures 3.4 and 3.5.



1. From: page 1, line 1, bar 1 — To: page 2, line 2, bar 4
Schubert, Franz, Op. 25, D. 795, [19] Die schöne Müllerin - [20] 1. Das Wandern

2. From: page 3, line 1, bar 1 — To: page 6, line 5, bar 5

recognized title(s):
[1_1_1] M'a-ssig geschwind.
[1_1_4] Volm
[1_3_2] F. S. 790.
[3_1_1] Massig.
[3_5_2] F. S. 791-
[4_5_2] F. S. 791
[5_5_2] F. S. 791.
[6_4_2] diznin.
[6_5_2] F. S. 791.
[7_1_1] Nicht zu geschwind.
[7_5_2] F. S. 792.
[8_5_2] F. S, 792.
[9_5_2] F. S.
[10_1_1] Etwas langsam.
[10_5_2] F. S. 793.
[11_5_3] 793.
[12_1_1] Ziemlich geschwind.
[12_4_2] F. 794.
[13_5_3] F. S. 794.
[14_5_2] F. S. 794.
[15_5_3] Ft S. 794.
[16_1_1] Langsam.
[16_5_2] F, S. 795
[17_1_1] Sehr langsam.
[17_5_2] F. S. 795.
[18_5_1] mich'?
[18_5_2] F. S. 795.
[19_1_1] Etwas geschwind.
[19_4_2] F. 796.
[21_1_1] Nliissig.
[21_4_3] F. S. 797.

Work (leaf-level): Schubert, Franz, Op. 25, D. 795, [19] Die schöne Müllerin - [21] 2. Wohin?

reference expression: [85][performance], Isabella Stewart Gardner Museum (source), Scarlata, Randall (instrumentalist), Den...

Expression: [65][sheet music], IMSLP (source), Breitkopf & Härtel (editor), 1895

Expression Information

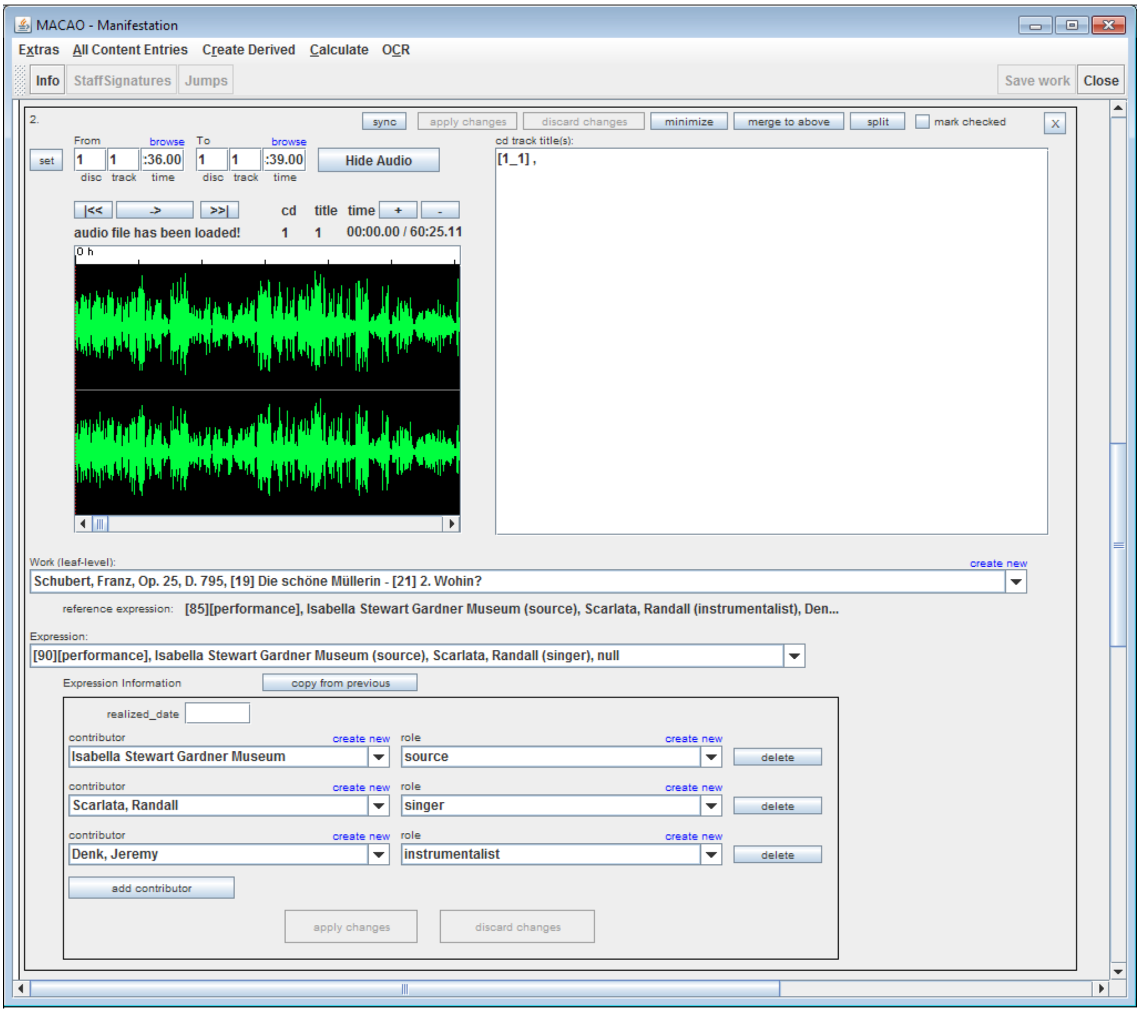
realized_date: 1895

contributor: IMSLP — role: source

contributor: Breitkopf & Härtel — role: editor

3. From: page 7, line 1, bar 1 — To: page 9, line 5, bar 5
Schubert, Franz, Op. 25, D. 795, [19] Die schöne Müllerin - [22] 3. Halt!

Figure 3.4. Integrated score view for expression editing showing the expression of the second song (*Wohin?*) from *Die schöne Müllerin*. The visualized score scans are enriched by OMR information on the position of the measures, and thus allow for selection-based determination of the start and end point of the expression. The currently set boundaries are marked blue. On the right, all detected text elements from the score pages are listed. These might provide some help in manually segmenting the score. Below the score, a drop-down list provides a quick means of selecting the correct work entry for the expression. Furthermore, the expression meta data can be edited.



2.

From: disc 1, track 1, time :36.00 — To: disc 1, track 1, time :39.00

cd track title(s): [1_1] ,

cd 1 title 1 time 00:00.00 / 60:25.11

audio file has been loaded!

0 h

Work (leaf-level): Schubert, Franz, Op. 25, D. 795, [19] Die schöne Müllerin - [21] 2. Wohin?

reference expression: [85][performance], Isabella Stewart Gardner Museum (source), Scarlata, Randall (instrumentalist), Den...

Expression: [90][performance], Isabella Stewart Gardner Museum (source), Scarlata, Randall (singer), null

Expression Information

realized_date

contributor: Isabella Stewart Gardner Museum — role: source

contributor: Scarlata, Randall — role: singer

contributor: Denk, Jeremy — role: instrumentalist

Figure 3.5. The expression editing mask for audio manifestations provides an audio player for fast and convenient segmentation. This feature is especially relevant if – as in the depicted example – one audio recording contains multiple works. On the right, Gracenote data can be made available to help with the segmentation and work identification. Again, components for work selection and meta data annotation are available below the player.

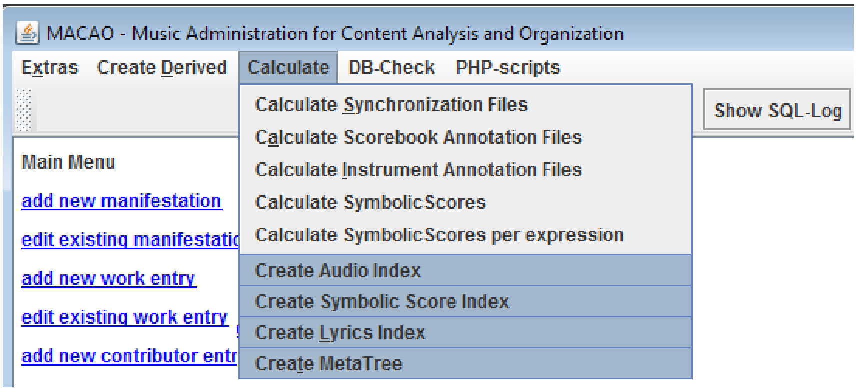

MACAO - Music Administration for Content Analysis and Organization
Extras
Create Derived
Calculate
DB-Check
PHP-scripts
Show SQL-Log
Main Menu
add new manifestation
edit existing manifestatio
add new work entry
edit existing work entry
add new contributor entr
Calculate Synchronization Files
Calculate Scorebook Annotation Files
Calculate Instrument Annotation Files
Calculate SymbolicScores
Calculate SymbolicScores per expression
Create Audio Index
Create Symbolic Score Index
Create Lyrics Index
Create MetaTree

Figure 3.6. Menu entries for creating the retrieval indexes of the content-based search methods provided by PROBADO MUSIC.

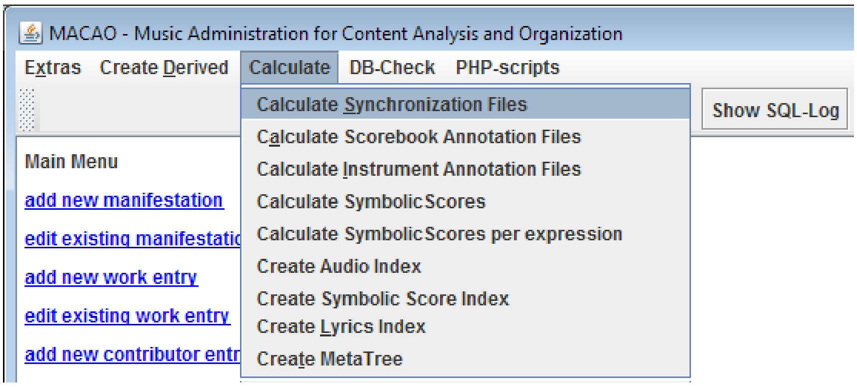

MACAO - Music Administration for Content Analysis and Organization
Extras
Create Derived
Calculate
DB-Check
PHP-scripts
Show SQL-Log
Main Menu
add new manifestation
edit existing manifestatio
add new work entry
edit existing work entry
add new contributor entr
Calculate Synchronization Files
Calculate Scorebook Annotation Files
Calculate Instrument Annotation Files
Calculate SymbolicScores
Calculate SymbolicScores per expression
Create Audio Index
Create Symbolic Score Index
Create Lyrics Index
Create MetaTree

Figure 3.7. Menu entry to perform the synchronization. Before the calculation starts, the user can choose whether to recreate all alignments, create only missing alignments, or select a list of works whose alignments should be calculated.

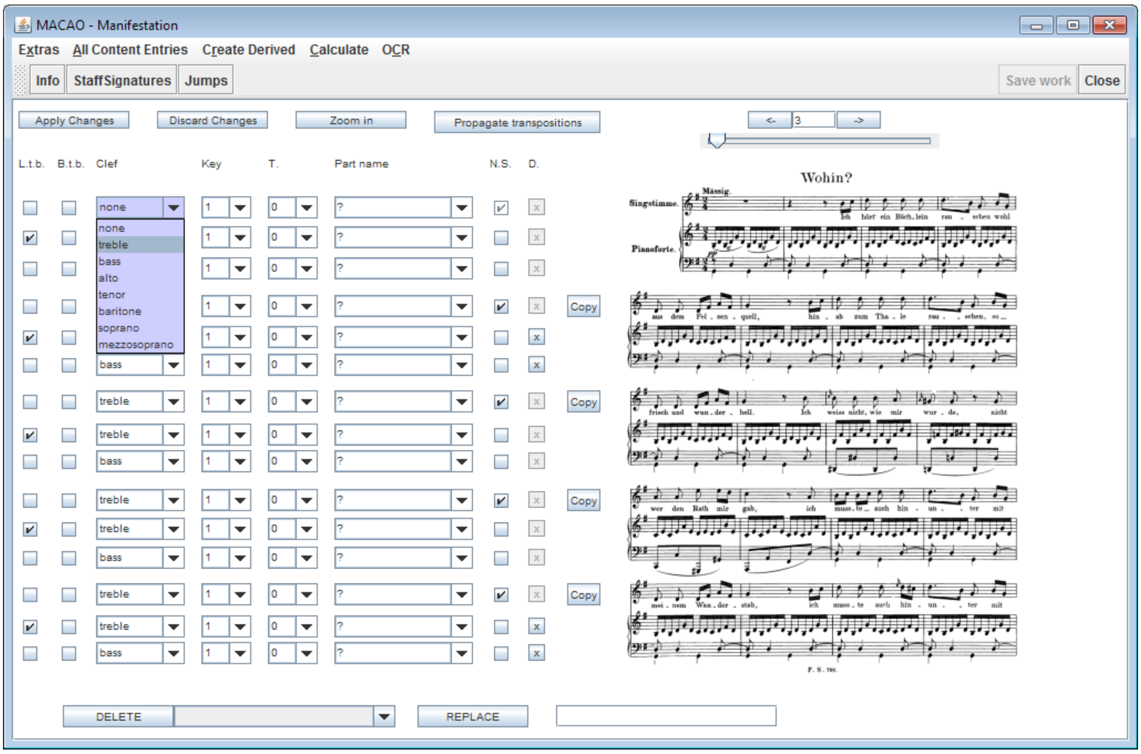
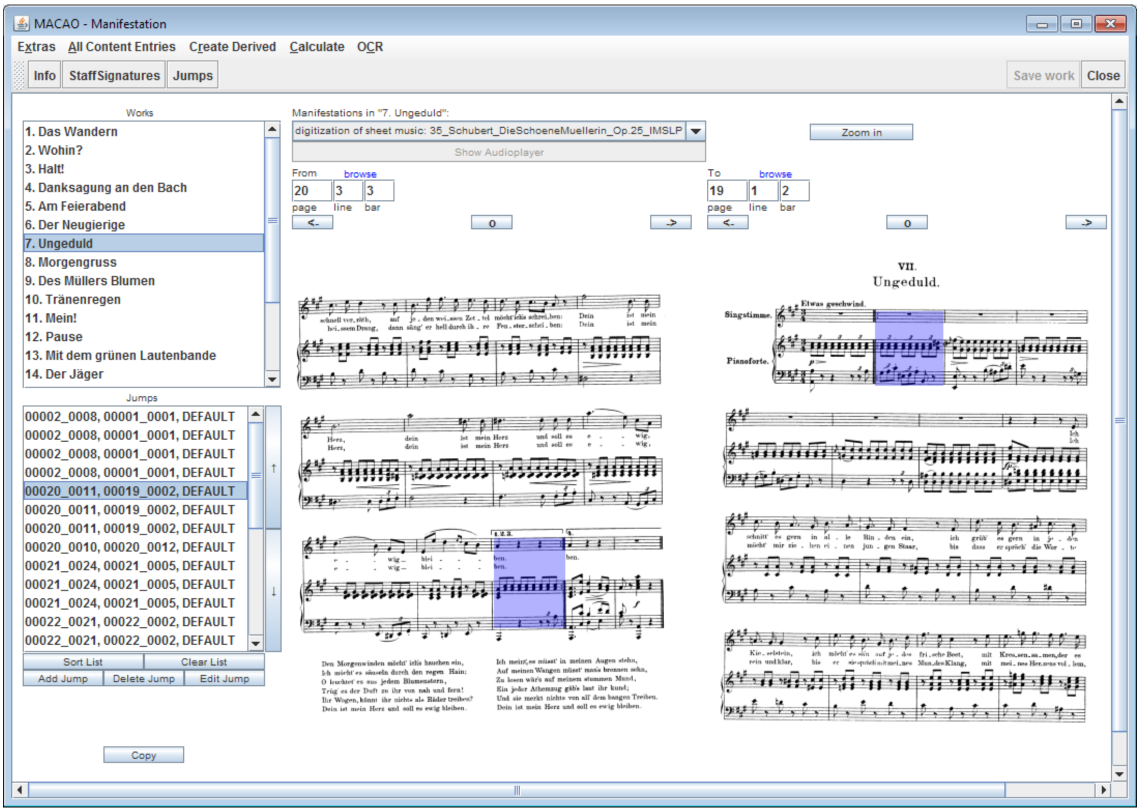


Figure 3.8. User interface for validation and correction of the symbolic score data created by the OMR system. A single score page is visualized on the right. The user can browse through the score by means of the slider or the arrows above the image. The according recognition information is depiced on the left. For each staff, the following information can be edited (from left to right): is the staff the upper staff of a grand staff, is the staff in a brace with the staff below, clef, key signature, transposition (in semitones), instrument name, and is the staff the first staff of a new system?



Works

1. Das Wandern
2. Wohin?
3. Halt!
4. Danksagung an den Bach
5. Am Feierabend
6. Der Neugierige
7. Ungeduld
8. Morgengruss
9. Des Müllers Blumen
10. Tränenregen
11. Mein!
12. Pause
13. Mit dem grünen Lautenbande
14. Der Jäger

Jumps

00002_0008, 00001_0001, DEFAULT
00002_0008, 00001_0001, DEFAULT
00002_0008, 00001_0001, DEFAULT
00002_0008, 00001_0001, DEFAULT
00020_0011, 00019_0002, DEFAULT
00020_0011, 00019_0002, DEFAULT
00020_0011, 00019_0002, DEFAULT
00020_0010, 00020_0012, DEFAULT
00021_0024, 00021_0005, DEFAULT
00021_0024, 00021_0005, DEFAULT
00021_0024, 00021_0005, DEFAULT
00022_0021, 00022_0002, DEFAULT
00022_0021, 00022_0002, DEFAULT

Manifestations in "7. Ungeduld":
digitization of sheet music: 35_Schubert_DieSchoeneMuellerin_Op.25_IMSLP

From: page 20, line 3, bar 3

To: page 19, line 1, bar 2

Figure 3.9. Editing mask for the repeat structure in the score and audio data. Given a correctly annotated score interpretation, the employed sheet music-audio synchronization algorithm [73, 75] attempts an automatic computation of the jump structure in the audio recordings. This data can be subsequently revised. The upper box on the left provides quick navigation between the different works in the score. To select which expression should be edited, a drop-down list of all expressions for the currently selected work is available above the score. In the bottom left, all jumps within the selected expression are listed. By selecting an entry, the according positions in the score are highlighted. To validate the jump instructions of audio expressions, the mask also provides an integrated audio player (not depicted).

Implementation Details

The meta data of a music collection managed with Probado Music are stored in a MySQL database.[15] Furthermore, the database contains information on the start and end points of the individual works within the manifestations. The actual documents in the collection are stored in the file system and mapped by the database. The Java application Macao encapsulates all required read and write accesses to this database (and the therefore required MySQL statements) by means of intuitive input masks. To this end, the JDBC (Java Database Connectivity) driver for MySQL is employed.[16]

To provide the functionalities described in the previous section, Macao uses a variety of additional Java libraries and programs. The most important dependencies are listed below. We introduce the external libraries and programs as met during the document preparation work flow.

ImageMagick: Macao allows for sheet music documents to be added in form of PDF files (single- and multi-page) or TIFF images (single page). PDF input is immediately converted into single page TIFF images by means of the command-line file convert mechanism of ImageMagick.[17]

Liszt OMR engine: For the automatic calculation of OMR data from the given (or previously derived) bitmap images, the SharpEye [145] command-line application Liszt OMR engine is accessed.

JOGL: The sheet music documents are visualized in the Probado Music front end by texture mapping with the Java Binding for the OpenGL API (JOGL).[18] To provide this visualization, the textures have to be created during preprocessing in Macao. Thus, the management system requires JOGL as well.

FFmpeg: The free command-line tool FFmpeg[19] is used for audio conversion.

MP3 SPI: Java Service Provider Interface (SPI) that supports the MP3 audio format and adds streaming and playback functionalities to Macao.[20]

3.2.2 The PROBADO MUSIC Front End

When first accessing Probado Music, several masks for the formulation of queries are offered to the user, see Figure 3.10.[21] Besides meta data-based search, the system includes content-based search mechanisms. For each supported modality (lyrics, score, and audio), the system implements according MIR techniques to search through all documents of that modality. Therefore, the user can also use lyrics to search for a piece of music. Furthermore, a score editor allows for the formulation of symbolic queries. Audio matching techniques are available as well. However, rather than free query formulation, the user can use extracts from the document collection for search. We will explain this type of

[15] `http://www.mysql.com`, February 2013

[16] `http://dev.mysql.com/downloads/connector/j/`, February 2013

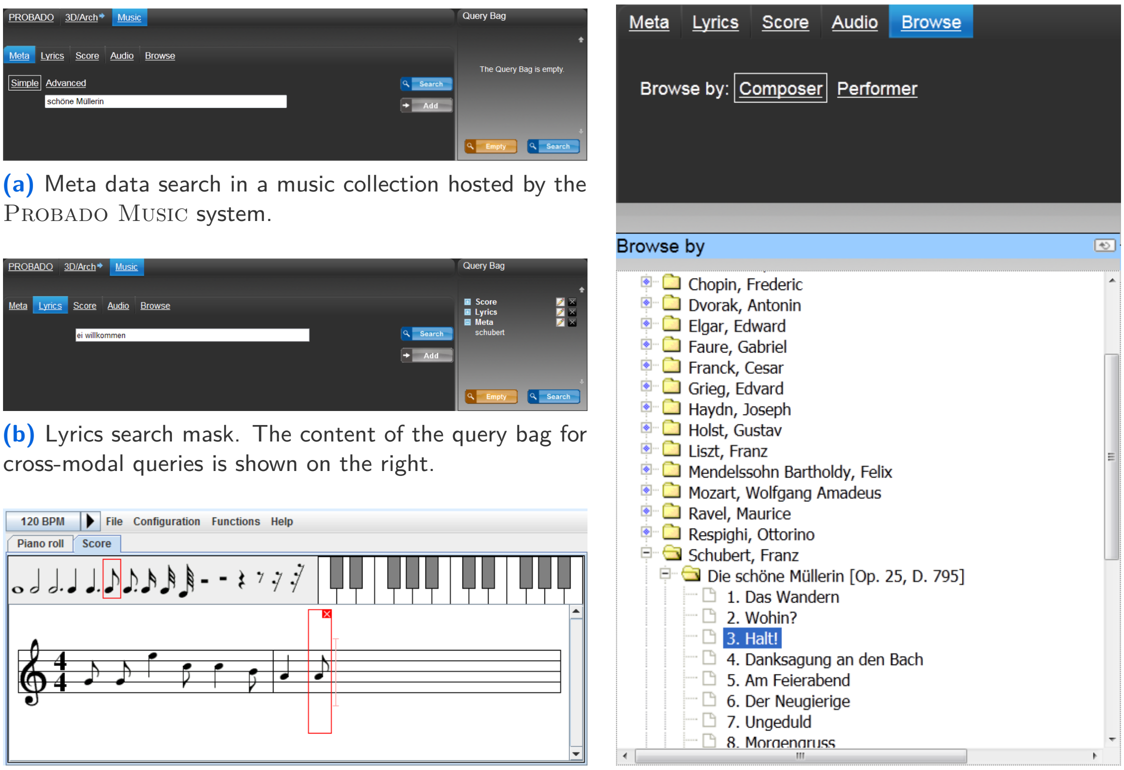
[17] `http://www.imagemagick.org`, February 2013

[18] `http://jogamp.org/jogl/www/`, February 2013

[19] `http://ffmpeg.org/`, February 2013

[20] `http://www.javazoom.net/mp3spi/mp3spi.html`, February 2013

[21] The majority of this section is adopted from our publication [190].



schöne Müllerin

ei willkommen

(a) Meta data search in a music collection hosted by the PROBADO MUSIC system.

(b) Lyrics search mask. The content of the query bag for cross-modal queries is shown on the right.

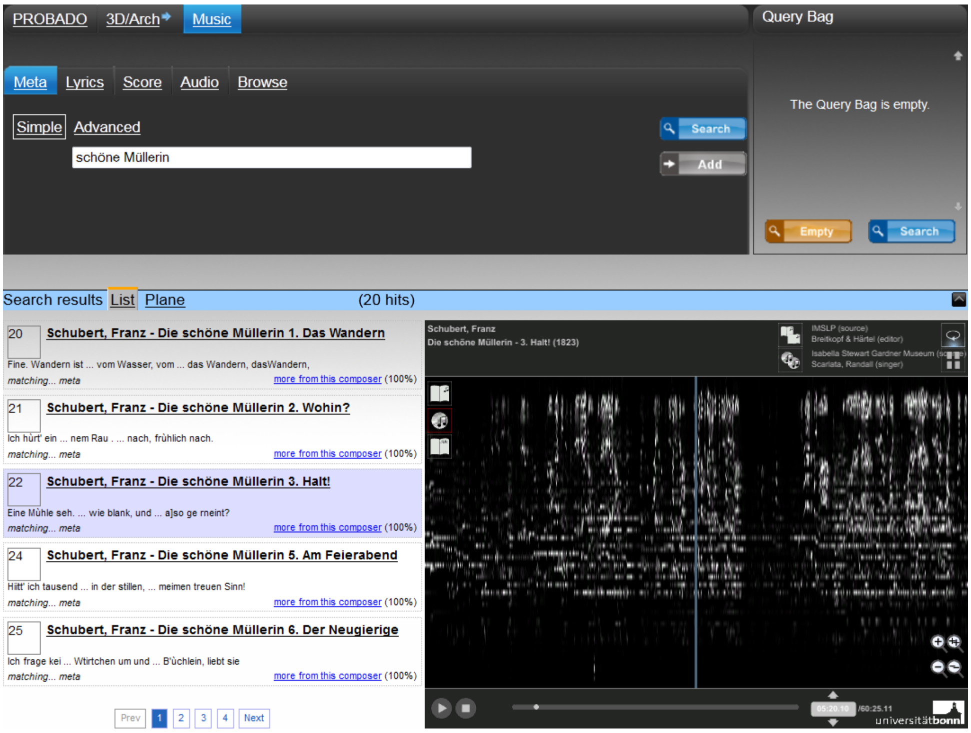
(c) Editor for symbolic score queries. The user can choose between a classic score view and a more technical piano roll visualization (not depicted).

(d) Tree-based access to the music collection.

Figure 3.10. Various search masks of the PROBADO MUSIC web interface, see Figure 3.11.

query formulation later on. In addition, PROBADO MUSIC introduced the new concept of cross-modal queries [55, 57]. The user can mix information of different modalities and combine it into a *query bag* to form one query. For example, the user can search for a piece of music by Schubert (meta data) containing the song text `"ei willkommen"`. Probably, the user might even remember a melody fragment which can be used as additional score information. As a final option, a tree-based presentation of all pieces of music contained in the music collection is offered, see Figure 3.10d. This tree can be sorted either by composer or artist (musician, conductor).

After starting a search (e.g., searching for the string `"schöne Müllerin"` in the meta data), the hit list is presented to the user, see Figure 3.11. In PROBADO MUSIC, a work-centered document access, as illustrated in Figure 1.1 on page 2, is pursued. Therefore, rather than listing all documents matching the current query, work entries are returned as hits. Upon selecting a result, all documents containing the according piece of music are made available for presentation. The current PROBADO MUSIC prototype supports three document types – sheet music, audio, and lyrics – and offers visualizations for each of them, see Figures 3.11, 3.12, and 3.13. After a piece has been selected for visualization, a document of the according document type is opened in every view. However, the user can easily exchange the document selected for presentation through lists containing all sheet music versions and all recordings of the current piece of music, respectively, see Figure 3.15a.



schöne Müllerin

Search results List Plane (20 hits)

20 **Schubert, Franz - Die schöne Müllerin 1. Das Wandern**
Fine. Wandern ist ... vom Wasser, vom ... das Wandern, dasWandern,
matching... meta more from this composer (100%)

21 **Schubert, Franz - Die schöne Müllerin 2. Wohin?**
Ich hùrt' ein ... nem Rau . ... nach, frùhlich nach.
matching... meta more from this composer (100%)

22 **Schubert, Franz - Die schöne Müllerin 3. Halt!**
Eine Mùhle seh. ... wie blank, und ... a]so ge rneint?
matching... meta more from this composer (100%)

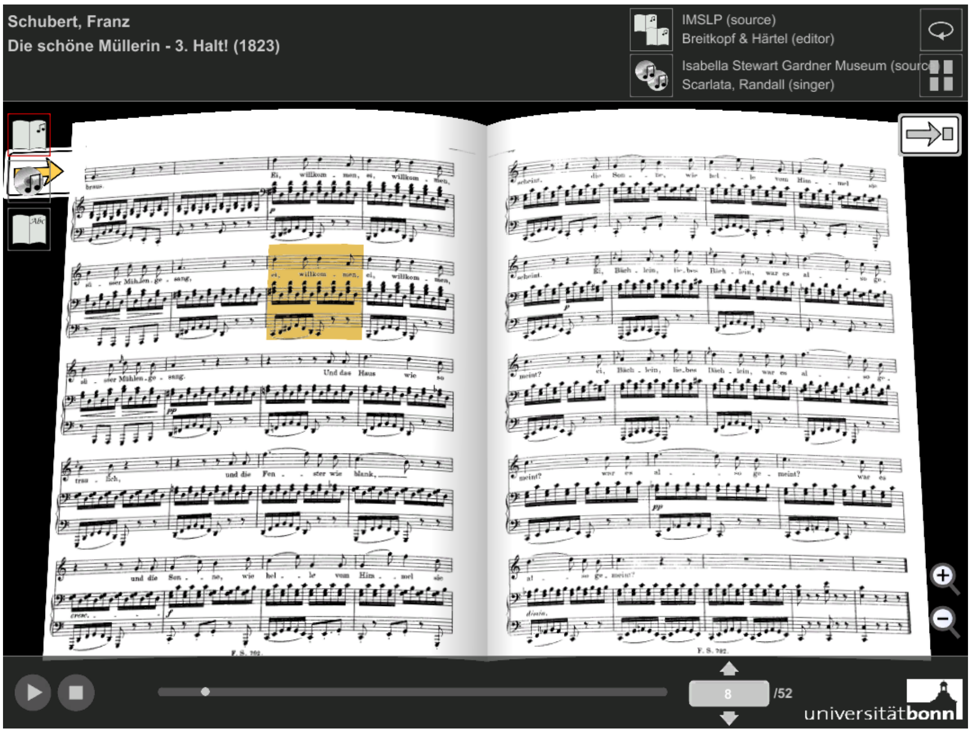
24 **Schubert, Franz - Die schöne Müllerin 5. Am Feierabend**
Hiitt' ich tausend ... in der stillen, ... meimen treuen Sinn!
matching... meta more from this composer (100%)

25 **Schubert, Franz - Die schöne Müllerin 6. Der Neugierige**
Ich frage kei ... Wtirtchen um und ... B'ùchlein, liebt sie
matching... meta more from this composer (100%)

Schubert, Franz
Die schöne Müllerin - 3. Halt! (1823)

IMSLP (source)
Breitkopf & Härtel (editor)
Isabella Stewart Gardner Museum (source)
Scarlata, Randall (singer)

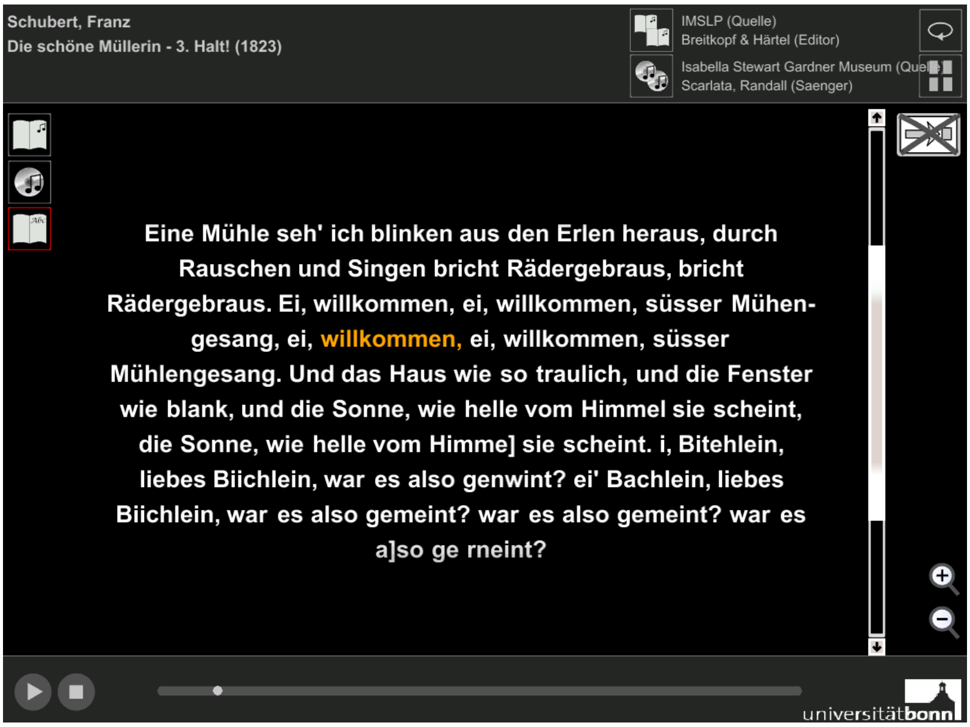
Figure 3.11. Web interface for query formulation (top) and result list (bottom left). The music documents are presented on the bottom right. Here, the audio player view offering common audio player capabilities together with a spectrogram visualization of the recording is shown.



Schubert, Franz
Die schöne Müllerin - 3. Halt! (1823)

IMSLP (source)
Breitkopf & Härtel (editor)
Isabella Stewart Gardner Museum (source)
Scarlata, Randall (singer)

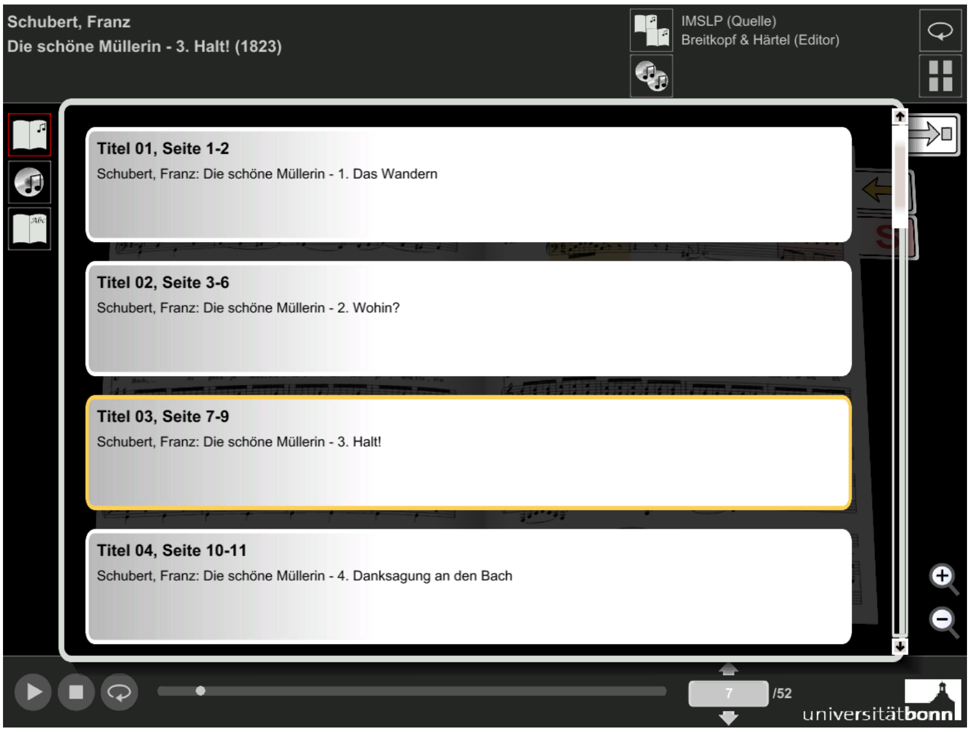
Figure 3.12. Presentation of a scanned score book in the PROBADO MUSIC front end. The current measure is highlighted and updated during audio playback. Above the document, the related meta data are depicted. Below, media player controls are provided.


Schubert, Franz
Die schöne Müllerin - 3. Halt! (1823)

IMSLP (Quelle)
Breitkopf & Härtel (Editor)
Isabella Stewart Gardner Museum (Quelle)
Scarlata, Randall (Saenger)

Eine Mühle seh' ich blinken aus den Erlen heraus, durch
Rauschen und Singen bricht Rädergebraus, bricht
Rädergebraus. Ei, willkommen, ei, willkommen, süsser Mühen-
gesang, ei, willkommen, ei, willkommen, süsser
Mühlengesang. Und das Haus wie so traulich, und die Fenster
wie blank, und die Sonne, wie helle vom Himmel sie scheint,
die Sonne, wie helle vom Himme] sie scheint. i, Bitehlein,
liebes Biichlein, war es also genwint? ei' Bachlein, liebes
Biichlein, war es also gemeint? war es also gemeint? war es
a]so ge rneint?

Figure 3.13. In the lyrics visualization, the current musical position is highlighted (on the word level). The lyrics are automatically extracted from the scanned score pages, see [55, 174], and may thus contain some misrecognitions.


Schubert, Franz
Die schöne Müllerin - 3. Halt! (1823)

IMSLP (Quelle)
Breitkopf & Härtel (Editor)

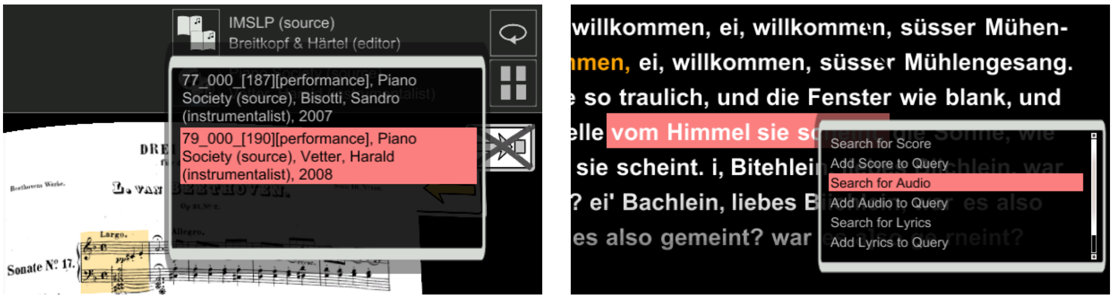
Titel 01, Seite 1-2
Schubert, Franz: Die schöne Müllerin - 1. Das Wandern

Titel 02, Seite 3-6
Schubert, Franz: Die schöne Müllerin - 2. Wohin?

Titel 03, Seite 7-9
Schubert, Franz: Die schöne Müllerin - 3. Halt!

Titel 04, Seite 10-11
Schubert, Franz: Die schöne Müllerin - 4. Danksagung an den Bach

Figure 3.14. List of contents for the currently selected sheet music manifestation. By selecting a work, the position in the score is updated accordingly and all related audio recordings are determined and made available.


IMSLP (source)
Breitkopf & Härtel (editor)
77_000_[187][performance], Piano Society (source), Bisotti, Sandro (instrumentalist), 2007
79_000_[190][performance], Piano Society (source), Vetter, Harald (instrumentalist), 2008
Sonate N° 17.
Largo.
willkommen, ei, willkommen, süsser Mühen-
men, ei, willkommen, süsser Mühlengesang.
so traulich, und die Fenster wie blank, und
lle vom Himmel sie sc
sie scheint. i, Bitehlei
? ei' Bachlein, liebes B
es also gemeint? war
Search for Score
Add Score to Query
Search for Audio
Add Audio to Query
Search for Lyrics
Add Lyrics to Query

(a) Direct access to all available interpretations of a work. The user can change the audio recording (depicted) and sheet music edition during playback and thereby directly compare the documents.

(b) Formulation of content-based queries from the PROBADO MUSIC front end. The user can select a region in any visualization – score, audio, or lyrics (depicted) – and use it as score, audio, or lyrics query or add it to the query bag.

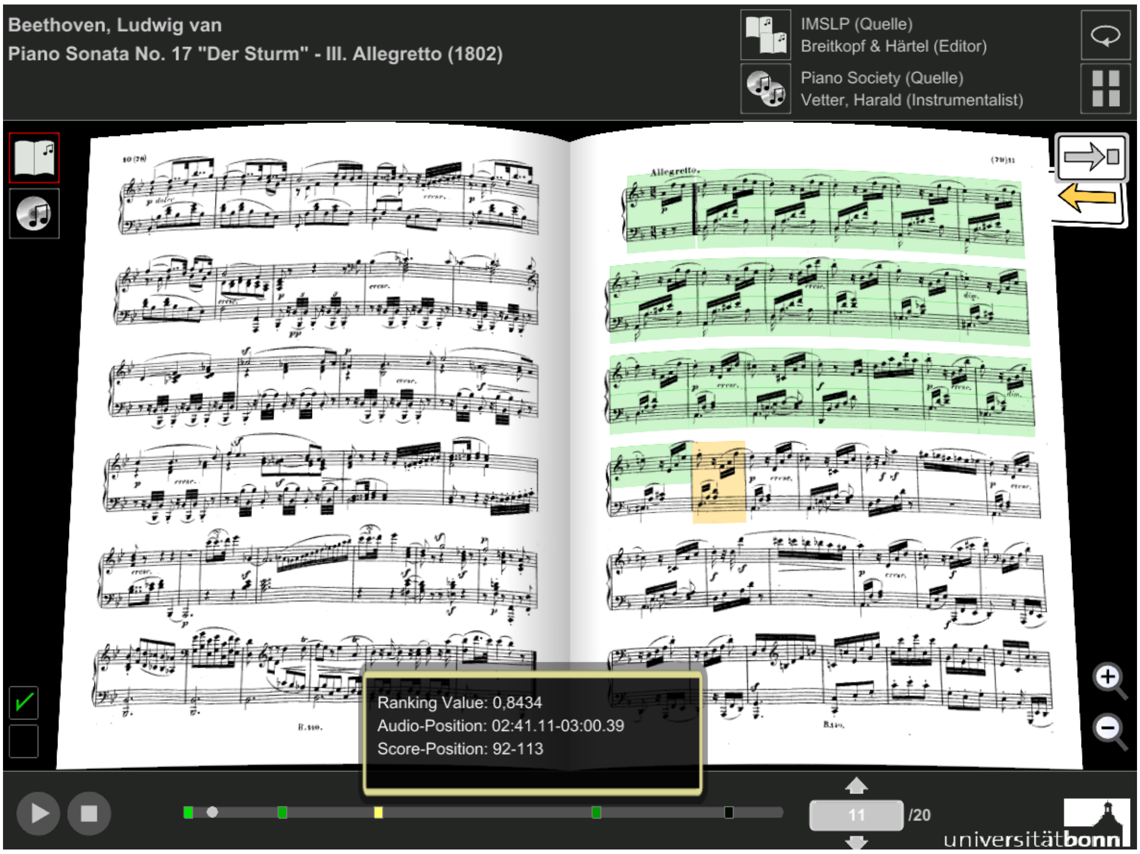
Figure 3.15. Features of the PROBADO MUSIC front end: interpretation switching and query formulation.

A further innovation of PROBADO MUSIC are multimodal navigation functionalities, realized through the inclusion of sheet music-audio synchronization techniques. As one benefit, these techniques enable *score-following*. While playing the audio, the currently audible measure is highlighted in the score. Another convenience introduced by sheet music-audio synchronization is *score-based navigation*. The user can freely browse through the currently loaded score book. Upon selecting a measure in the score, the audio recording automatically jumps to the according time position and playback will continue from there. In addition, the employed synchronization allows for keeping the musical position while exchanging the score or audio document selected for visualization. Thus, the user can quickly compare different recordings of a piece of music without repeatedly searching for the specific position he/she is interested in. Similarly, lyrics following and lyrics-based navigation are available.

In addition to the previously described search masks, the user can create content-based queries from within the visualized documents, see Figure 3.15b. In each view, the user can mark an arbitrary region. Due to the previously described synchronization, the user can then decide whether to use the according score-, audio-, or lyrics-extract as query. Upon accessing the result of a content-based query, the exact match positions are visualized in the documents, see Figure 3.16. The user can thereby quickly navigate through all matches and compare them.

Implementation Details

The PROBADO MUSIC front end is divided into an HTML- and JavaScript-based web interface for retrieval and presentation of the result list and a Java applet for document presentation that is integrated into this website (compare Figure 3.11). Similar to MACAO, the Java applet utilized JOGL and the MP3 SPI for document presentation (texture rendering and audio playback). The two components of the front end communicate with each other through JavaScript (e.g., to select a query result for presentation or to trigger a content-based query from a music example selected in the applet, Figure 3.15b). The communication between the front end and the PROBADO server – also implemented in



Beethoven, Ludwig van
Piano Sonata No. 17 "Der Sturm" - III. Allegretto (1802)

IMSLP (Quelle)
Breitkopf & Härtel (Editor)
Piano Society (Quelle)
Vetter, Harald (Instrumentalist)

Ranking Value: 0,8434
Audio-Position: 02:41.11-03:00.39
Score-Position: 92-113

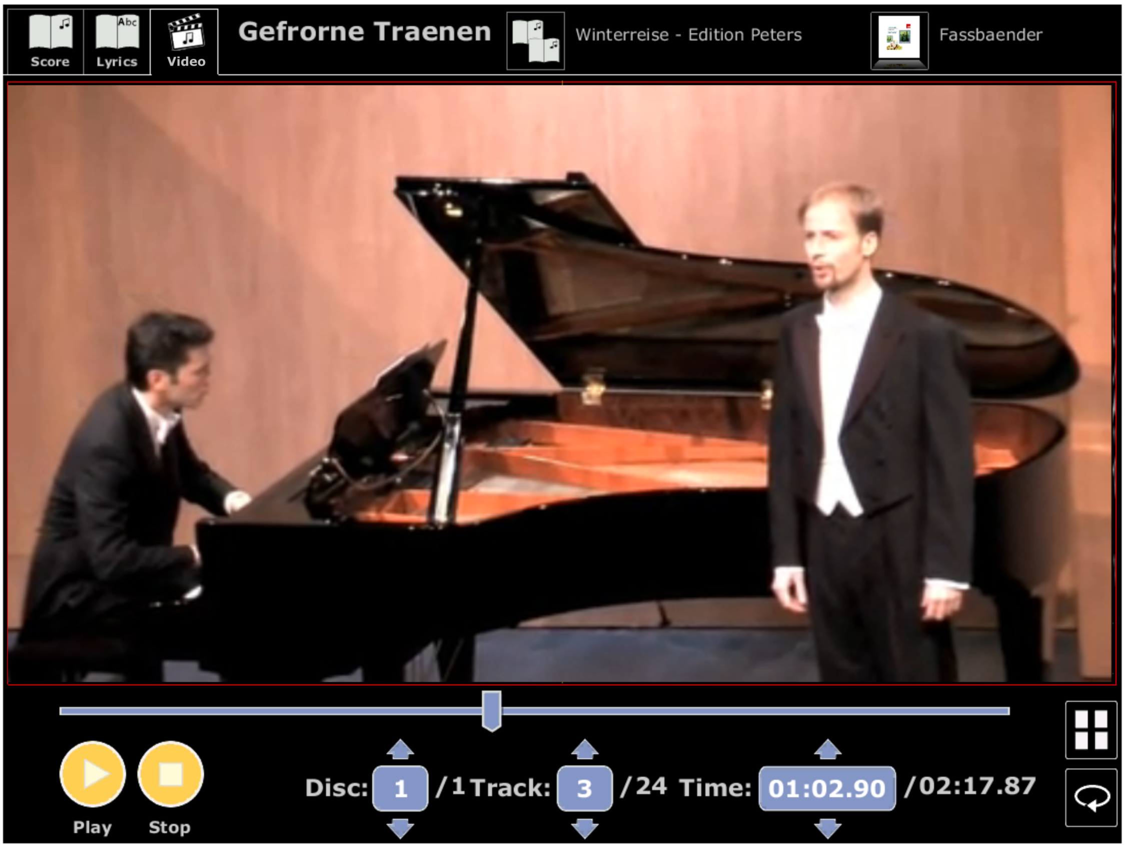
Figure 3.16. Hit visualization for an audio query consisting of the first 15 measures from the third movement of Beethoven's *Piano Sonata No. 17*. The matching regions are highlighted both in the music documents and on the timeline below. The color intensities on the time line indicate the ranking values of the matches.

Java – is realized through the Simple Object Access Protocol (SOAP)[22] and the Java Remote Method Invocation (RMI) technology.[23] For further details, we refer to [55].

3.2.3 Supporting Video Recordings

The most frequently encountered digitally available music representations are scanned sheet music and audio recordings. Therefore, those two file types were considered in PROBADO MUSIC and served as proof of concept. Of course, several similarly popular and important music representations exist (symbolic score, libretti, video recordings) that would also qualify for consideration in a digital music library. Extending digital music libraries to provide multimodal access to as many different media sources and types as possible can support the process of experiencing and analyzing the music. To demonstrate the general possibility of processing and presenting further document types in PROBADO MUSIC we conducted a feasibility study and added video support [192].

[22] `http://www.w3.org/TR/soap/`, February 2013
[23] `http://www.oracle.com/technetwork/java/javase/tech/index-jsp-136424.html`, February 2013



Gefrorne Traenen

Winterreise - Edition Peters

Fassbaender

Disc: 1 /1 Track: 3 /24 Time: 01:02.90 /02:17.87

Figure 3.17. Experimental support of video recordings in a previous version of the PROBADO MUSIC front end (source [57]).

Nowadays, most performances are recorded as video to be distributed to a broad audience via DVD, television, or online services.[24] Furthermore, specific video productions of pieces of music are available, too. After digitizing these documents they can be added to the PROBADO MUSIC library. By extracting the audio tracks of the video recordings, all content-based preprocessing steps for audio files, such as chroma calculation, indexing, or the calculation of music synchronizations, can be reused for the videos. To access the video recordings in the PROBADO MUSIC front end, a video player was integrated. Figure 3.17 shows a previous version of the PROBADO MUSIC front end that offers video visualization and playback. The concept of content-based linking of all music representations associated with the same piece of music was extended to include video recordings. Thereby, score-following and score-based navigation is available for videos as well. In addition, smooth transitions between different audio and video recordings are possible. To provide the user with only one auditory music representation at a time, the video and the audio player are exchanged as necessary.

In order to implement the described video support, the media player from the JavaFX framework[25] was employed. The player supports all required playback functions (open a video file, play, pause, jump in video, and get current position in video). At the time when

[24] For example, the *Digital Concert Hall* offers access to concerts of the *Berliner Philharmoniker*, `http://www.digitalconcerthall.com`, February 2013.

[25] `http://www.javafx.com/`, February 2013

video support was integrated, the front end was a standalone Java application with direct file system access for audio and video playback. Later, the system was converted into an applet and file streaming was added. While this was not done for video recordings, the used media player would be capable of video streaming and could thus be easily extended.

Although, simultaneously looking at both the video and the score scans might be impracticable, having a time aligned view constitutes several advantages. In longer video recordings, it might be cumbersome to search for a specific point in time, whereas using the score for navigation is easier and faster for most users. Furthermore, we envision the application of our system in the education of prospective conductors. The students can analyze and compare recordings of different conductors and thus improve their style. Having a time-aligned score can help in understanding difficult sections and the decisions of the recorded conductors. This type of visual learning can also be advantageous in other areas, for example, in dance. Dancers, choreographers, or dance theorists can use Probado Music to compare different performances of the same piece. Equally, recording a performance from various angles can help in restages. During rehearsals, dancers can switch between the different recordings to analyze the movements more thoroughly. Furthermore, aspects such as orchestra arrangements, stage designs, costumes, and make-up can be compared with Probado Music if video recordings are available.

4 Music Synchronization

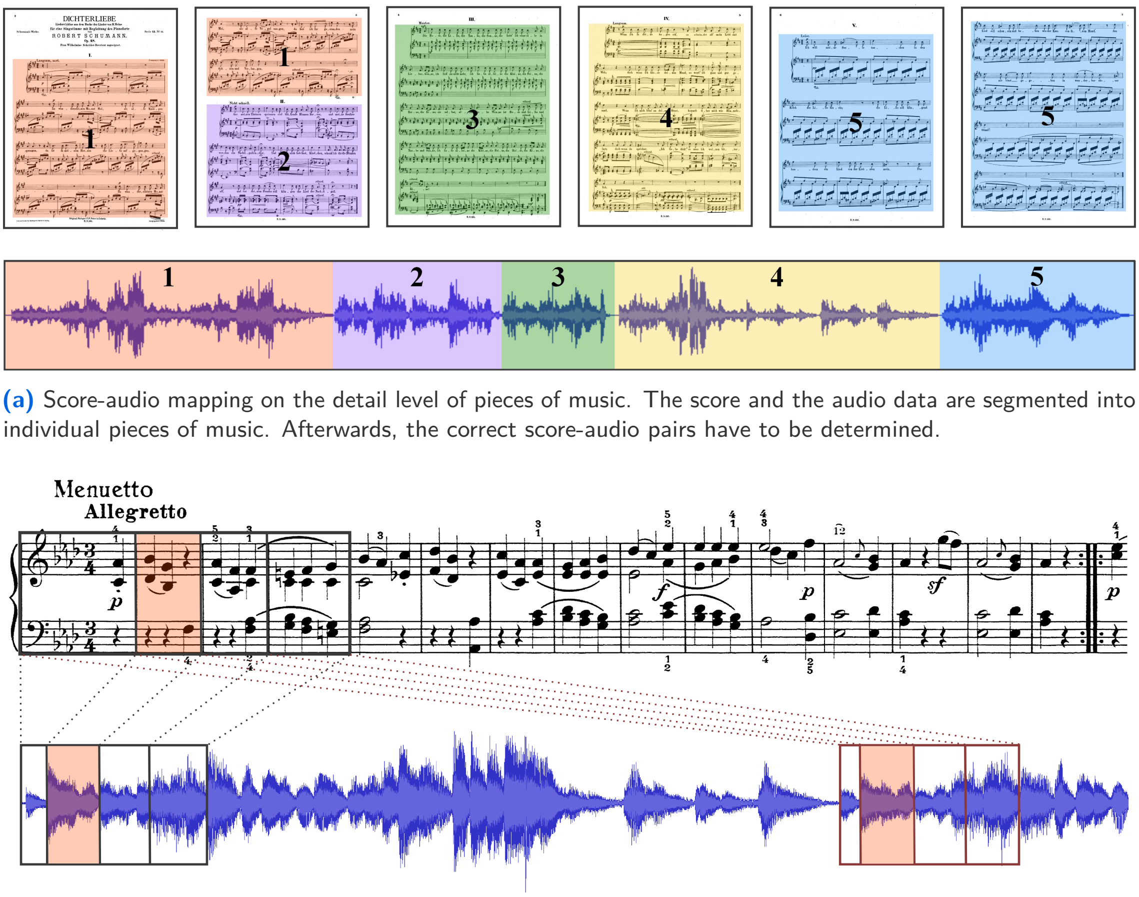
Aligning Sheet Music and Audio

In Chapter 2, we discussed the most common music representations. Afterwards, the previous chapter described the PROBADO MUSIC system. One of the most innovative features of the PROBADO front end is its capability of indicating which measure in a score is currently being played back in an audio recording. In this chapter, we address the problem of automatically generating such musically relevant linking structures between scanned score images and audio recordings by means of a process referred to as *sheet music-audio synchronization*. We introduce the synchronization task and the specific issues to be dealt with in more detail in Section 4.1. In Section 4.2, we subsequently describe a basic alignment technique called *dynamic time warping* (DTW). Before DTW can be applied to music documents in a larger music collection, a mapping between audio recordings and sheet music documents that represent the same musical content has to be created. This task is called *sheet music-audio mapping* and an approach developed in the context of the PROBADO MUSIC project is briefly described in Section 4.2.2. In Section 4.2.3, we discuss some ideas on how to deal with repetition-related structural differences between two music documents. The complexity of the music and thus also of the score is significantly higher for orchestral pieces of music, and new issues such as transposing instruments have to be considered during synchronization. In Section 4.3, we present a novel approach for reconstructing the transposition information for orchestral scores. In addition, we discuss evaluation results that establish a distinct improvement of the synchronizations by applying the proposed method. Concluding, we discuss some possible applications of sheet music-audio alignments in Section 4.4.

Parts of this chapter have previously been published. Sections 4.1, 4.2, and 4.4 are to a great extent based on our publication [194]. Equally, Sections 4.3.2 and 4.3.3 are in large parts based on our paper [195].

4.1 Task Specification

The goal of music synchronization is the generation of semantically meaningful bidirectional mappings between two music documents representing the same piece of music. Those documents can be of the same data type (e.g., audio-audio synchronization) or of different data types (e.g., score-audio synchronization or lyrics-audio synchronization). In the case of score-audio synchronization, the created linking structures map regions in a musical



(a) Score-audio mapping on the detail level of pieces of music. The score and the audio data are segmented into individual pieces of music. Afterwards, the correct score-audio pairs have to be determined.

(b) Score-audio synchronization on the measure level. Time segments in the audio stream are mapped to individual measures in the score representation. The depicted audio track contains a repetition. Therefore, the according score measures have to be mapped to both audio segments.

Figure 4.1. Examples for score-audio synchronization on different detail levels.

score, e.g., pages or measures, to semantically corresponding sections in an audio stream, see Figure 4.1.

Although the task of score-audio synchronization appears to be straightforward, several aspects exist along which the task and its realization can vary, see Figure 4.2. The particular choice of settings with respect to these aspects is always influenced by the intended application of the synchronization results.

The first choice concerns the sought detail level or granularity of the synchronization. A very coarse synchronization level would be a mapping between score and audio sections representing the same piece of music, see Figure 4.1a. This type of alignment is also referred to as *score-audio mapping*. The Neue Mozart-Ausgabe [197], for example, employs score-audio mapping to provide online access to scanned sheet music and corresponding audio recordings. Finer detail levels include page-wise [8, 66], system-wise, measure-wise [108], or note-wise [17, 161] linking structures between two music documents. The choice of granularity can in turn affect the level of automation. The manual annotation of the linking structure might be achievable for page-wise synchronizations. For finer granularities semi-automated or automated synchronization algorithms would be preferable. While automatic approaches do not need (and also not allow) any user interaction, in semi-

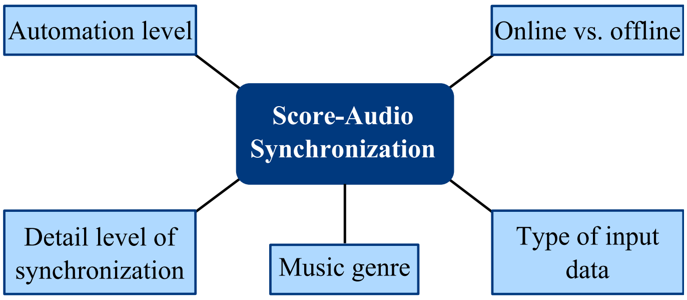


Figure 4.2. Aspects of score-audio synchronization.

automatic approaches some user interaction is required. However, the extent of the manual interaction can vary between manually correcting a proposed alignment on the selected detail level and correcting high-level aspects like the repeat structure before recalculating the alignment. The selected automation level also depends on the amount of data to be processed. For a single piece of music given only one score and one audio interpretation, a full-fledged synchronization algorithm might not be required. But for the digitized music collection of a library, manual alignment becomes impossible. Finally, reliability and accuracy requirements also play a role in the automation decision.

Another huge differentiation concerns the runtime scenario. In *online synchronization*, the audio stream is only given up to the current playback position, and the synchronization should produce an estimation of the current score position in real-time. There exist two important applications of online score-audio synchronization techniques, namely *score following* and *automated accompaniment* [52, 60, 131, 132, 161, 163, 202]. The real-time requirements of this task turn local deviations between the score and the audio into a difficult problem. Furthermore, recovery from local synchronization errors is problematic. In contrast, in *offline synchronization* the complete audio recording and the complete score data are accessible throughout the entire synchronization process [108, 149]. Also, the computation is not required to run in real-time. Due to the less strict calculation time requirements and the availability of the entire audio and score data during calculation, offline synchronization algorithms usually achieve higher accuracies and are more robust with regard to local deviations in the input data. The calculated linking structures can afterwards be accessed to allow, e.g., for score-based navigation in audio files.

The genre/style of the music to be synchronized also influences the task of score-audio synchronization. While Western classical music and most popular music feature strong melodic/harmonic components, other music styles like African music may mainly feature rhythmic drumming sounds. Using harmonic information for the synchronization of rhythmic music will prove ineffective and, therefore, different approaches have to be employed.

The type of input data – more precisely the score representation – constitutes the last aspect of score-audio synchronization. The score data can either be available as scanned score images or as symbolic score, e.g., MIDI or MusicXML. Obviously, the choice of score input affects the type of challenges to be mastered during synchronization. While symbolic score representations are usually of reasonable quality and the extraction of the individual music events is straightforward, some sort of rendering is required to present the score data. In contrast, scanned sheet music already provides a visualization. However, the music information needs to be reconstructed from the image data before the linking structures can be calculated. OMR systems approach this task and achieve high reconstruction rates for printed Western music. Nevertheless, the inclusion of OMR into the synchronization process may result in defective symbolic score data. Usually, the errors are mainly of local nature. Thus, by choosing a slightly coarser detail level (e.g., measure level) sound synchronization results can be achieved. For a differentiation between these two types of input data, the term *sheet music-audio synchronization* is often utilized if scanned images are given as score input.

Various researchers are active in the field of score-audio synchronization and work on all settings of the listed aspects has been reported. In this chapter, we focus on the task of automated offline sheet music-audio synchronization for Western classical music producing linking structures on the measure level. Furthermore, the processing of large music collections should be possible.

The basic idea in most score-audio synchronization scenarios is to transform both input data types into a common mid-level representation. These data streams can then be synchronized by applying standard alignment techniques, see Section 4.2. Regardless of the selected approach, one has to cope with the following problems to get reasonable synchronization results.

Differences in structure: A score can contain a variety of symbols representing jump instructions (e.g., repeat marks, segno signs, or keywords such as `"da capo"`, `"Coda"`, or `"Fine"`, see Figure 2.11 on page 15). While OMR systems are capable of detecting repeat marks, they often fail to reliably detect most other jump instructions in the score. Therefore, the correct playback sequence of the measures cannot be reconstructed. However, even if all jump instructions are correctly recognized, the audio recording may reveal additional repeats or omissions of entire passages notated in the score. Again, the given sequence of measures does not coincide with the one actually played in the audio recording. Such structural differences lead to major challenges in score-audio synchronization.

Differences between music representations: Score pages and audio recordings represent a piece of music on different levels of abstraction and capture different facets of the music. One example is the tempo. Music notation may provide some written information on the intended tempo of a piece of music and tempo changes therein (e.g., instructions such as `"Allegro"` or `"Ritardando"`). However, those instructions provide only a rough specification of the tempo and leave a great deal of space for interpretation. Therefore, different performers might deviate significantly in their specific tempo choices. Most musicians even add tempo changes that are not specified by the score in order to emphasize certain musical passages. For an example piece with substantial tempo changes, we refer to Figure 4.3.

The differences in the loudness of instruments and the loudness variations during the progression of a piece of music are further important characteristics of a given performance. Just like tempo, loudness is notated only in a very vague way and OMR systems often fail to detect the few available instructions. Similarly, music notation only provides timbre information through instrument labels. Therefore, timbre-related sound properties, such as instrument-dependent overtone energy distributions, are not explicitly captured by the score.

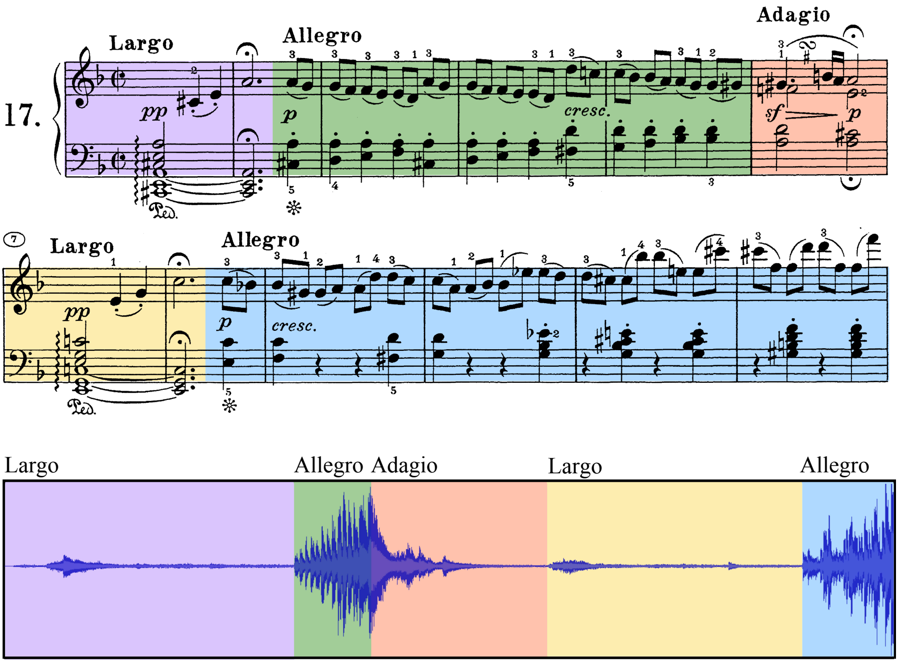


Figure 4.3. Extract of Beethoven's *Piano Sonata No.* 17 (publisher: *G. Henle Verlag*, pianist: V. Ashkenazy). In the first nine measures alone, four substantial tempo changes are performed. Thus, the duration of the measures in the audio recording varies significantly. However, the score only provides vague instructions that result at best in an approximation of the intended tempo changes.

In conclusion, in view of practicability, score-audio synchronization techniques need to be robust towards variations in tempo, loudness, and timbre to deal with the mentioned document type-related differences.

Errors in the input data: As already discussed in Section 2.3, OMR is not capable of reconstructing the score information perfectly. The errors introduced by OMR can be divided into local and global ones. Local errors include, e.g., misidentifications of accidentals, missed notes, or incorrect note durations. In contrast, examples for global errors are errors in the detection of the musical key or the ignorance of transposing instruments. While errors in the score are introduced during the reconstruction from the scanned images, the audio recordings themselves can be erroneous. The performer(s) may locally play some wrong notes or a global detuning may occur. For example, most international orchestras deviate slightly from the standard tuning of A4 = 440Hz, see [150]. Furthermore, in Baroque music a deviation by a whole semitone is common practice.

Score-audio mapping: Especially in library scenarios, the goal is not the synchronization of one piece of music. Usually, the input consists of whole sheet music books and whole CD collections. Therefore, the scanned score and the audio data need to be segmented into individual pieces of music. As the order in the sheet music books and on the CDs might differ, a mapping on this level of granularity needs to be created before the synchronizations on a finer level of detail can be calculated.

Although we focused on the task of synchronizing scanned scores and audio recordings when we prepared the presented listing, most of the mentioned problems also exist for other score-audio synchronization variants.

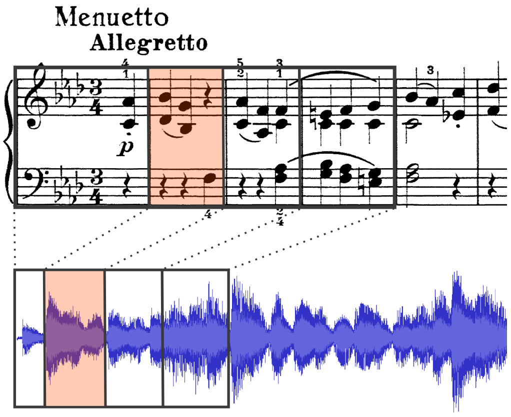


Figure 4.4. Schematic example for score-audio synchronization on the measure level. Time segments in the audio stream are mapped to individual measures in the score.

4.2 Alignment Techniques

The goal of sheet music-audio synchronization is to link regions in two-dimensional score images to semantically corresponding temporal sections in audio recordings, see Figure 4.4. To compare the sheet music of a piece of music with an audio recording thereof, both representations are converted into a common mid-level representation. In the synchronization context, chroma-based music features turned out to be a powerful and robust mid-level representation. These features have the property of eliminating differences in timbre and loudness to a certain extent while preserving the harmonic progression in the music. Therefore, their application is most suitable for music with a clear harmonic progression, like most Western classical music. In addition, by appropriately choosing the size of the sections represented by individual chroma vectors, local errors in the input data can be canceled out for the most part. For details on how the chroma representation of audio recordings and scores can be calculated, we refer to Section 2.4.1.

As we first transform the input documents into chroma features, the following steps for the calculation of sheet music-audio alignments is equally applicable to other music synchronization scenarios, such as audio-audio, symbolic score-audio, or score-score synchronization. Given the feature sequences, most alignment procedures define a local cost measure. Afterwards, the actual synchronization result is calculated by using a suitable alignment strategy. A commonly used computational approach to this task is a dynamic programming approach called *dynamic time warping* (DTW). This approach is employed for the calculation of music alignments in Probado Music and forms the base for most of the work presented in the remainder of this chapter. A description of DTW for music synchronization is provided in the following section. In combination with chroma features, some variations in timbre, loudness, and tempo as well as small deviations in the data streams (due to errors) can be handled. Thus, the presented DTW approach already copes with some of the problems mentioned in Section 4.1. However, it is assumed that only one sheet music representation and one audio interpretation of the same piece of music are given. An approach for sheet music-audio mapping is presented in Section 4.2.2. A further assumption is that the structure of the score and the audio recording coincide. Some ideas on how to handle structural differences will be presented in Section 4.2.3.

4.2.1 Dynamic Time Warping

Dynamic time warping was originally used for the comparison of different speech patterns [158] and has since then successfully been applied to other research areas, such as data mining, information retrieval, and MIR. In general, the goal of DTW is to find an optimal non-linear alignment between two given sequences that observes certain restrictions. In the following, we will introduce the main idea of DTW. Our descriptions in this section follow [133, Section 4.1].

Let $V := (v_1, v_2, \ldots, v_N) \in \mathcal{F}^N$ and $W := (w_1, w_2, \ldots, w_M) \in \mathcal{F}^M$ be the chroma feature sequences that represent the two documents to be aligned. Here, $\mathcal{F}$ denotes the underlying *feature space*. For chroma features, $\mathcal{F}$ consists of all elements in $[0, 1]^{12}$. To compare two chroma vectors $v_n, w_m \in \mathcal{F}$, a *local cost measure* $c : \mathcal{F} \times \mathcal{F} \to [0, 1]$ on $\mathcal{F}$ is defined by $c(v_n, w_m) := 1 - \langle v_n, w_m \rangle$ (which is the cosine measure for normalized vectors). Pairwise comparison of the feature vectors of the two sequences with this cost measure yields an $(N \times M)$-*cost matrix* C defined by $C(n, m) := c(v_n, w_m)$. Then, the goal in music synchronization is the identification of a path through this cost matrix C that connects the beginnings and endings of the two feature sequences. This path should further be optimal with respect to the local costs along the path. More formally, we define an (N, M)-*warping path* (also *alignment path*) as a sequence $p = (p_1, p_2, \ldots, p_L)$ with $p_\ell = (n_\ell, m_\ell) \in [1\colon N] \times [1\colon M]$ for $\ell \in [1\colon L]$ that satisfies the following conditions.[1]

1. The path connects the beginnings and the endings of the two feature sequences: $p_1 = (1, 1)$ and $p_L = (N, M)$.
2. The path moves monotonic through V and W: $n_1 \leq n_2 \leq \ldots \leq n_L$ and $m_1 \leq m_2 \leq \ldots \leq m_L$.
3. The path only proceeds according to a set of admissible step sizes Σ: $p_{\ell+1} - p_\ell \in \Sigma$ for $\ell \in [1\colon L - 1]$. Typical choices for Σ are $\Sigma = \Sigma_1 := \{(1, 1), (1, 0), (0, 1)\}$ or $\Sigma = \Sigma_2 := \{(1, 1), (2, 1), (1, 2)\}$.

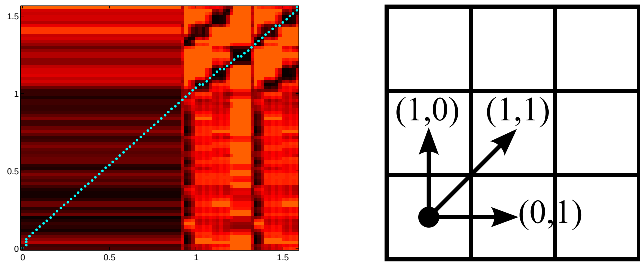
We would like to remark that the third condition indirectly results in condition 2 being true. Figure 4.5 shows an example of such an alignment path and the set Σ_1 of admissible step sizes. Given an eligible alignment path p for two sequences V and W over $\mathcal{F}$, its *total cost* $c_p(V, W)$ with respect to the cost measure c is defined as the sum of the local cost covered by the path

$$c_p(V, W) := \sum_{\ell=1}^{L} C(p_\ell).$$

The synchronization between two music representations is then encoded by an *optimal alignment path* p^*, i.e., the alignment path having the minimal cost among all possible paths. More precisely, the *DTW distance* $DTW(V, W)$ between V and W is defined as the total cost of the optimal alignment path p^*

$$DTW(V, W) := c_{p^*}(V, W) = \min\{c_p(V, W) \mid p \text{ is a valid path between } V \text{ and } W\}. \tag{4.1}$$

[1] We distinguish between $[0\colon n] := \{x \in \mathbb{Z} \mid 0 \leq x \leq n\}$ and $[0, n] := \{x \in \mathbb{R} \mid 0 \leq x \leq n\}$.



(a) (b)

Figure 4.5. **(a)** Local cost matrix for the score chromagram and one of the audio chromagrams depicted in Figure 2.16 on page 22. The optimal alignment path is highlighted in light blue. **(b)** Visualization of the allowed steps of $\Sigma = \Sigma_1 := \{(1,1),(1,0),(0,1)\}$ in the DTW procedure.

Instead of computing and testing all possible paths through C, a dynamic programming algorithm with computational complexity $\mathcal{O}(NM)$ can be used. To this end, we define the *accumulated cost matrix* $D \in \mathbb{R}^{N\times M}$ by

$$D(n,m) := DTW(V(1\colon n), W(1\colon m)).$$

Here, $V(1\colon n) := (v_1, v_2 \ldots, v_n)$ with $n \in [1\colon N]$ and $W(1\colon m) := (w_1, w_2 \ldots, w_m)$ with $m \in [1\colon M]$ are defined as the prefix sequences of V and W, respectively. Intuitively, at position (n,m) the matrix D stores the minimum cost of any admissible alignment path, i.e., the cost of the optimal alignment path, starting at position $(1,1)$ and ending at position (n,m). For the set of admissible steps Σ_1 the accumulated cost matrix can be computed by means of the following recursion

$$D(n,m) = \begin{cases} C(1,1) & \text{if } n = m = 1 \\ \sum_{k=1}^{n} C(k,1) & \text{if } n \in [2\colon N] \text{ and } m = 1 \\ \sum_{k=1}^{m} C(1,k) & \text{if } n = 1 \text{ and } m \in [2\colon M] \\ C(n,m) + \min \begin{cases} D(n-1,m) \\ D(n,m-1) \\ D(n-1,m-1) \end{cases} & \text{otherwise.} \end{cases} \tag{4.2}$$

For other step conditions, such as Σ_2, the accumulated cost matrix can be computed in a similar way.

Starting with $p_L = (N,M)$, an optimal warping path $p^* = (p_1, \ldots, p_L)$ can then be computed in reverse order. To this end, we check at each position (n,m) which of the possible predecessor positions, e.g., $(n-1,m), (n,m-1)$, and $(n-1,m-1)$ for Σ_1, has the lowest accumulated costs.

The presented approach does not consider tuning differences between the music documents. In the case of varying tunings, however, the feature sequences may show significant differences that can result in poor synchronization quality, see Figure 2.16 on page 22. To suitably adjust the chroma features in the case of small tuning deviations, a tuning estimation step can be included in the feature calculation process [65]. If the tuning differences exceed a semitone, e.g., because the piece was transposed to match the vocalists range, one may apply brute-force techniques such as trying out all possible cyclic shifts of the chroma features [83, 135].

4.2.2 Sheet Music-Audio Mapping

Arranging the music data of a digital library in a work-centered way or, more precisely, piece of music-wise has proven beneficial. Thus, in the context of a digitization project to build up a large digital music library, one important task is to group all documents that belong to the same piece of music. Note that in this scenario, the music documents to be organized are not given as individual songs or movements, but rather as complete scanned score books or audio CD collections that usually contain several pieces of music. In addition, we typically have to deal with numerous versions of audio recordings of one and the same piece of music[2] and with a number of different score versions (different publishers, piano reductions, orchestra parts, transcriptions, etc.) of that piece. Thus, the final goal at this level of detail is to segment both the score books and audio recordings in such a way that each segment corresponds to one piece of music. Furthermore, each segment should be provided with the appropriate meta data. This segmentation and annotation process, called *sheet music-audio mapping*, is a crucial prerequisite for sheet music-audio synchronization on a higher level of detail. One possibility to solve this task is to manually perform this segmentation and annotation. However, for large collections this would be an endless undertaking. Thus, semi-automatic or even fully automatic mapping techniques should be developed.

For audio recordings and short audio extracts, music identification services, like *Shazam*,[3] can provide meta data. Furthermore, ID3 tags, CD covers, or annotation databases, such as Gracenote[4] and Music Info Disc,[5] can contain information on the recorded piece of music. However, their automated interpretation can quickly become a challenging task. To name just two prominent issues, the opus numbers given by the different sources might not use the same catalog or the titles might be given in different spellings or different languages. Furthermore, the mentioned services do not provide information for public domain recordings. Another issue can be introduced by audio tracks that contain several pieces of music. Here, the exact start and end positions of the individual pieces of music have to be determined.[6] However, this information is usually not provided on CD covers or in meta data databases. Still, the mentioned information sources can be used to support the manual segmentation and annotation process. The automatic extraction and analysis of textual information on scanned score images has to be considered at least as challenging.

Given one annotated audio recording of all the pieces contained in a score book, Fremerey et al. [73, 76] proposed an automatic identification and annotation approach for sheet music that is based on content-based matching. First, chroma features are calculated for all music documents. Then, to identify the audio recording that contains the same material as a given score segment, audio matching as described in Chapter 5 is performed. By retrieving consecutive segments of the score a so-called *mapping matrix* is constructed. In such a matrix, the rows correspond to the audio documents, the columns to the consecutive score

[2] For example, the British Library Sounds includes recordings of about 750 performances of Beethoven String Quartets, as played by 90 ensembles, see `http://sounds.bl.uk/Classical-music/Beethoven`, February 2013.

[3] `http://www.shazam.com`, February 2013

[4] `www.gracenote.com`, February 2013

[5] `http://mid-music-info-disc.software.informer.com`, February 2013

[6] Usually, longer periods of silence can hint at the beginning of a new piece. However, the direction "`attacca`" resulting in two successive movements played without a pause, can prevent this clue from existing.

queries, and the individual entries hold the costs of the top k matches of each query. This mapping matrix combined with the information on the audio segmentation finally gives both the segmentation and the annotation of the scanned sheet music.

In the same manner, additional audio recordings of already known pieces can be segmented and annotated. Therefore, through the presented approach the manual processing of only one manifestation of each piece of music is required.

4.2.3 Dealing with Structural Differences

When comparing and synchronizing scores and performances, it may happen that their global musical structures disagree due to repeats and jumps performed differently than suggested in the score. These structural differences have to be resolved to achieve meaningful synchronizations. In the scenario of online score-audio synchronization, this issue has already been addressed [8, 112, 152, 185]. Pardo and Birmingham [152] and Arzt et al. [8] both used structural information available in the score data to determine music segments where no jumps can occur. The first publication employs an extended *Hidden Markov Model* to allow for jumps between the known segment boundaries. In the second approach, an extension of the DTW approach to music synchronization is used to tackle structural differences. At each ending of a section, three hypotheses are pursued in parallel: First, the performance continues on to the next section. Second, the current section is repeated. Third, the subsequent section is skipped. After enough time has passed in the performance, the most likely hypothesis is kept and followed. Besides approaches exploiting structural information available from the score, Müller et al. [134, 136] approached a more general case where two data sources (e.g., two audio recordings) are given but no information on allowed repeats or jumps is available. In this case, only partial alignments of possibly large portions of the two documents are computed.

Fremerey et al. [73, 75] presented a method for offline sheet music-audio synchronization in the presence of structural differences, called *JumpDTW*. Here, jump information is derived from the sheet music reconstruction to create a block segmentation of the piece of music, see Figure 4.6. As already mentioned, OMR systems may not recognize all types of jump instructions and especially textual instructions are often missed. Therefore, bold double bar lines, which can be detected with a high reliability, are used as block boundaries. At the end of each block, the performance can then either continue to the next block or jump to the beginning of any other block in the piece, including the current one (in contrast to [8] where only forward jumps skipping at most one block are considered). To allow for jumps at block endings, the set of DTW steps is modified. For all block endings, transitions to all block starts in the score are added to the usual steps. By calculating an optimal alignment path using a thus modified accumulated cost matrix, possible jumps in the performance can be detected and considered during the synchronization process.

4.3 Orchestral Scores and Transposing Instruments

Due to the large number of instruments in orchestral music, the score notation inevitably becomes more complex. Typically, this results in a reduced OMR accuracy. Furthermore, orchestral scores contain information commonly neglected by OMR systems. One very

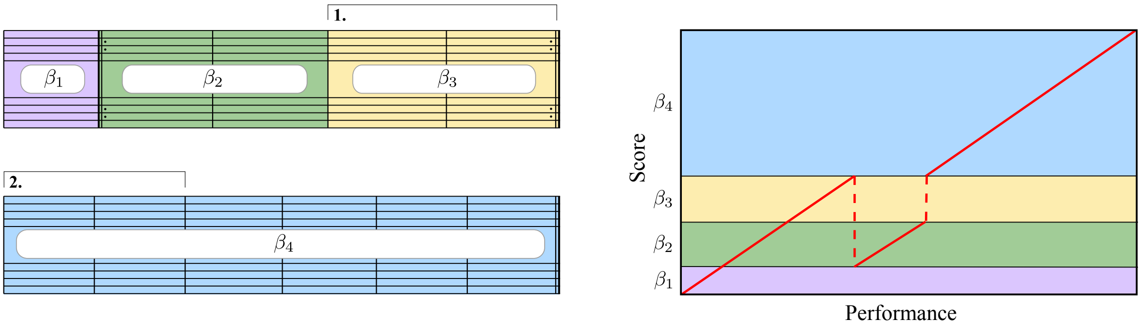


Figure 4.6. Score block sequence $\beta_1\beta_2\beta_3\beta_4$ created from the notated score jumps and alignment path for an audio with block structure $\beta_1\beta_2\beta_3\beta_2\beta_4$ (adapted from [73]).

important example is the transposition information. The specific transposition of an instrument is usually marked in the score by textual information such as "`Clarinet in E`", see Figure 2.9 on page 14. If this information is disregarded during the OMR reconstruction, the pitch information for transposing instruments will be incorrect. In Section 4.3.1, we will present an evaluation of the impact of different types of OMR errors on the synchronization quality. The results show that the lack of transposition information can have a strong impact on the accuracy of the calculated alignments and has to be considered the worst OMR error – apart from missing jump instructions that lead to structural mismatches. In our experiments, we also included sheet music of piano reductions. Interestingly, those proved to be equally well or even better suited for sheet music-audio synchronization than the original orchestral scores lacking transposition information.

In Section 4.3.2, we introduce an approach for the reconstruction of transposition information from sheet music. The basic idea of the proposed method is to combine the results of OMR and OCR to regain the information available through text annotations in the score. In addition, a method is proposed for reconstructing the instrument and transposition information for staves where text annotations were omitted or not recognized. The presented approach was initially studied in the context of a diploma thesis project [204] and was subsequently further extended and properly formalized [194, 195]. Furthermore, we built on the initial evaluations and performed a set of experiments to evaluate the impact of the automatic transposition reconstruction, see Section 4.3.3. The results again establish the need for knowing the transposition information and, on top of that, prove the positive impact of the proposed reconstruction procedure on the alignment quality.

4.3.1 Evaluation of OMR Errors

In this section, we present the results of an evaluation studying the impact of OMR errors and missing transposition information on the quality of sheet music-audio synchronization. As a ground truth, we employ the beat annotation of the RWC Music Library [84]. We generated reference synchronizations on the measure level by extracting the measure starting points from these beat annotation files. Our test data comprise the audio recordings of the 11 orchestral pieces in the RWC library that contain at least one transposing instrument, see Table 4.1. For each recording, the respective sheet music was processed with the OMR system SharpEye (data sources: IMSLP [87] and Bavarian State Library [16]). Then, we calculated for each piece of music four synchronizations

Label	Composer	Work	Publisher
BE1	Beethoven	*Symphony No.* 5, 1st mvmt.	Breitkopf & Härtel
BR1	Brahms	*Horn Trio in Eb major*, 2nd mvmt.	Peters
BR2	Brahms	*Clarinet Quintet in B minor*, 3rd mvmt.	Breitkopf & Härtel
HA1	Haydn	*Symphony No.* 94, 1st mvmt.	Kalmus
MO1	Mozart	*The Marriage of Figaro*, Overture	Bärenreiter
MO2	Mozart	*Symphony No.* 40, 1st mvmt.	Bärenreiter
MO3	Mozart	*Clarinet Quintet in A major*, 1st mvmt.	Breitkopf & Härtel
MO4	Mozart	*Violin Concerto No.* 5, 1st mvmt.	Bärenreiter
ST1	Strauss	*The Blue Danube*	Dover Publications
TC1	Tchaikovsky	*Symphony No.* 6, 4th mvmt.	Dover Publications
WA1	Wagner	*Tristan and Isolde*, Prelude	Dover Publications

Table 4.1. Test data for evaluating the impact of OMR errors and transposing instruments on the synchronization quality. For each piece of music, an audio snippet of roughly two minutes' length and the according sheet music extract were used.

between an audio extract of roughly two minutes' length and the according score clipping. First, the unaltered OMR data were aligned to the audio extracts. In the other cases we manipulated the OMR result before performing the synchronization. In the second case, we manually corrected the OMR (OMR^c).[7] In the third case, we annotated the missing transposition labels in the score (OMR^t) and in the last setting, the OMR was corrected and supplemented with transposition labels (OMR^{c,t}). For each setting, the difference between the measure starting points calculated by the alignment and those given by the RWC beat annotations was determined. Table 4.2 shows the mean and standard deviations for all the mentioned settings. As the numbers indicate, both correcting the OMR result and adding transposition information have a positive impact on the synchronization quality. However, the improvements achievable by adding transposition information are distinctly higher. In summary, the presented evaluation supports the necessity of identifying and annotating transposing instruments in orchestral scores to improve the synchronization accuracy.

Piano Reductions

In a *piano reduction* or *piano transcription*, the music material initially composed for an orchestra is adapted and reduced to its most basic components to produce a version playable by one – sometimes two – piano(s), see Figure 4.7. Thus, other than the original score, the piano reduction does not contain any transposing instruments. In this additional experiment, we investigated whether synchronizing a piano reduction with a full orchestral recording of a piece of music is possible and how good the results are compared to a synchronization with the full orchestral score.

We only found piano reductions (PR) for three of the orchestral pieces from the RWC Music library (BE1, MO2, and MO4). The results are presented in Table 4.3. To our surprise, the piano reduction without any manual correction on average achieves distinctly better results than OMR, OMR^c, or OMR^t and only slightly weaker results than the corrected and annotated orchestral score OMR^{c,t}. Even upon inspecting the individual results, the

[7] OMR errors such as incorrect clefs, key signatures, and time signatures and misrecognized/missing accidentals and notes were manually corrected in the SharpEye user interface.

Label	**OMR**		**OMR^c**		**OMR^t**		**OMR^{c,t}**	
	mean	std	mean	std	mean	std	mean	std
BE1	462	700	342	528	265	391	151	169
BR1	390	671	376	651	110	125	92	80
BR2	266	803	265	803	124	84	125	82
HA1	456	1016	265	287	283	441	268	281
MO1	247	349	244	343	128	178	130	178
MO2	93	88	89	83	93	86	89	81
MO3	243	383	257	392	65	53	64	52
MO4	79	81	103	181	69	66	68	66
ST1	451	658	451	658	310	492	328	477
TC1	434	502	418	470	385	378	368	304
WA1	1005	980	1018	967	889	884	936	856
∅	**375**	**566**	**348**	**488**	**247**	**289**	**238**	**239**

Table 4.2. Overview of the deviation of the different synchronization results from the ground truth (in ms). OMR: the recognition result as produced by the SharpEye system. OMR^c: OMR output adjusted by the recognition errors of SharpEye. OMR^t: OMR result annotated with transposition information. OMR^{c,t}: corrected OMR with transposition information.

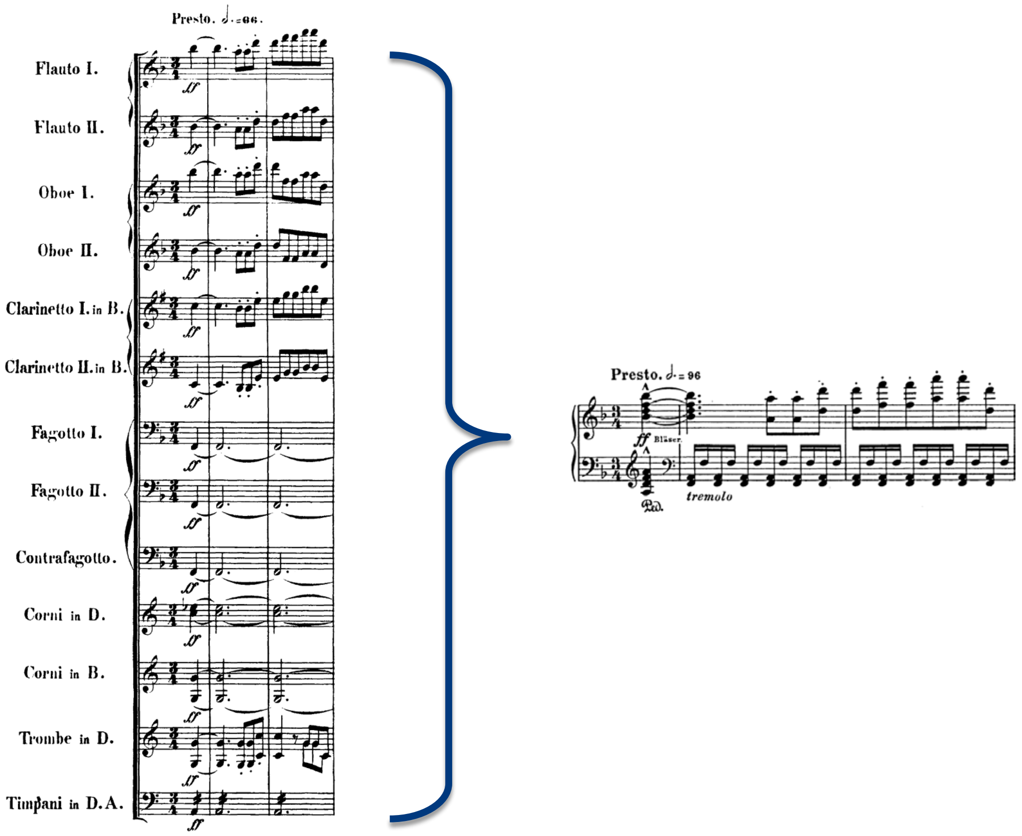


Figure 4.7. Example of a piano transcription of the Presto from the 4th movement of *Symphony No.* 9 by L. v. Beethoven (publisher: *Breitkopf & Härtel*). The material, which was distributed to multiple instruments in the original orchestral score, is compressed and transferred into a piano score.

Label	OMR		OMR^c		OMR^t		$\mathrm{OMR}^{c,t}$		PR	
	mean	std	mean	std	mean	std	mean	std	mean	std
BE1	462	700	342	528	265	391	151	169	178	201
MO2	93	88	89	83	93	86	89	81	93	95
MO4	79	81	103	181	69	66	68	66	82	80
∅	**211**	**290**	**178**	**264**	**142**	**181**	**103**	**105**	**118**	**125**

Table 4.3. Comparison of the synchronization results for the various orchestral score versions and the piano reductions (PR). Again, the numbers indicate the mean and standard deviation in milliseconds.

Label	Composer	Work
BE2	Beethoven	*Symphony No.* 9, 4th mov., Presto
BE3	Beethoven	*Symphony No.* 9, 4th mov., Vivace
BE4	Beethoven	*Symphony No.* 9, 4th mov., *Ode to Joy* instrumental
BE5	Beethoven	*Symphony No.* 9, 4th mov., *Ode to Joy* vocals

Table 4.4. List of additional test data for the evaluation of piano reductions. All score material was published by *Breitkopf & Härtel*.

piano reduction proved to be always at least as good as the orchestral score without any corrections.

In addition, an orchestral score and a piano reduction of several extracts from the fourth movement of *Symphony No.* 9 by L. v. Beethoven were compared, see Table 4.4. For each extract two recordings, one conducted by Wilhelm Furtwängler and the other by Rafael Kubelik, were available. As we lack a proper ground truth, the alignment results for the corrected and annotated orchestral score ($\mathrm{OMR}^{c,t}$) were used as the basis of comparison for the unaltered OMR results of the orchestral score and the piano reduction. The evaluation results are depicted in Table 4.5. Again, the numbers indicate a much better or similar performance of the piano transcription.

As the experiments show, it is possible to produce synchronizations of a full orchestral recording with a piano reduction. If the orchestral score was not corrected (of OMR errors or transposition information), the quality of the alignment even turned out to be distinctly higher most of the time. An idea for obtaining high-quality sheet music-audio alignments of orchestral pieces would be to synchronize the piano reduction with both the orchestral score and the audio recording. Subsequently, the alignment path for the piano reduction and the audio could be transferred to the orchestral score. While this is a reasonable approach, piano transcriptions do not exist for all orchestral pieces. Furthermore, a library might not accept the additional effort for digitizing and preparing the piano reductions. In the next section, we therefore present an automatic approach for annotating orchestral scores with transposition information.

4.3.2 Reconstruction of Transposition and Instrument Information

In Western classical music, the score notation usually obeys some common typesetting conventions. Examples are the textual transposition information, but also the introduction of all instruments playing in a piece of music by labeling the staves of the first system.

Label	**OMR**		**PR**	
	mean	std	mean	std
BE1	398	726	86	139
BE2, W. Furtwängler	12,047	5,824	312	557
BE3, W. Furtwängler	440	969	90	158
BE4, W. Furtwängler	97	214	156	359
BE5, W. Furtwängler	1,097	1,948	183	529
BE2, R. Kubelik	11,911	6,018	194	321
BE3, R. Kubelik	262	391	102	255
BE4, R. Kubelik	91	179	109	225
BE5, R. Kubelik	143	312	184	472
$\varnothing$	**2,943**	**1,842**	**157**	**335**

Table 4.5. Synchronization results for the test data listed in Table 4.4. For each piece of music, a recording conducted by W. Furtwängler and one conducted by R. Kubelik was used. The first column shows the mean and standard deviation (in ms) between the alignment path for the orchestral score with the path calculated for the corrected and annotated OMR data of the same score (OMR^{c,t}). In the second column, the alignments for the piano reduction and for OMR^{c,t} are compared.

Furthermore, a fixed instrument order and the usage of braces and accolades help in reading the score [171]. However, despite of all these rules, the task of determining which instrument is supposed to play in a given staff (instrument-staff mapping) and whether or not it is a transposing instrument can be challenging. For most scores the number of staves remains constant throughout the entire piece of music. Therefore, the instrument names and transposition information are often omitted after the first system, and the information given in the first system needs to be passed on to the remaining systems. The task of determining the instrument of a staff and its transposition becomes even more complicated for compressed score notations where staves of pausing instruments are removed, see Figure 2.5 on page 11. Here, the instrument order is still valid, but some of the instruments introduced in the first system may be missing. To clarify the instrument-staff mapping in these cases, textual information is given. However, in these cases the instrument names are usually abbreviated and, therefore, more difficult to recognize. Furthermore, transposition information is often only provided in the first system of a piece or in the case that the transposition changes. The textual information might be omitted altogether if the instrument-staff mapping is obvious for a human reader (e.g., strings are always the last instrument group in a system).[8]

In this section, we present our method for reconstructing the instrument and transposition labels in staves of orchestral scores. Basically, the algorithm can be subdivided into three parts. In the first part of the process, the relevant text areas on the score scans are identified and processed by an OCR software. Subsequently, the recognition results are transformed into instrument labels and matched to the corresponding staves. In Section 4.3.2.1, we give a detailed description of the OCR-based label reconstruction. After this step, all staves, where textual information was given in the score and recognized by the OCR software, possess an instrument label. But in orchestral scores, instrument text labels are often omitted after the first system. In the second step of the algorithm, missing labels are, therefore, reconstructed by propagating existing labels, see Section 4.3.2.2. Afterwards, each staff has an instrument label associated with it. In the final step of the algorithm the

[8] This paragraph originates from our publication [194].

transposition labels that were found in the first system are propagated through the score, see Section 4.3.2.3.

We impose some assumptions on the scores processed with our method.

- The first system contains all instrument names that occur in the piece. Some older editions do not stick to this convention and use compressed notation from the start, see Figure 4.8.
- The instrument order established in the first system is not changed in subsequent systems.
- A maximum of two staves share a common instrument text label.
- When first introduced, full instrument names are used.
- For compressed scores, text labels are given if the instrumentation changed compared to the preceding system.
- Changes for transposing instruments are indicated at the beginning of a staff, directly after the instrument label.

For most orchestral scores these assumptions are met.

We will now provide a detailed account of the three steps of the instrument and transposition labeling algorithm. For an even more extensive description, we refer to [204].

4.3.2.1 OCR-based Labeling

In this part of the reconstruction, we analyze textual information given on the score sheets to create instrument and transposition labels. The rough work flow for this part is depicted in Figure 4.9.

First, regions of a scanned score image that possibly contain text/words naming an instrument or a transposition are detected. By convention, the instrumentation and transposition information is either placed in front of the staff system or above the staves directly at the start of the system, see Figure 4.10. It is therefore sufficient to search these areas – instead of the whole image – for eligible text labels. To determine the area to be checked, the staff location information from the OMR data are employed. Then, connected components (CCs) of black pixels in these areas are determined [169, 204]. Afterwards, CCs that definitely do not contain letters are discarded. Using a sweep line algorithm [19], horizontally neighboring CCs are then merged to form words. Subsequently, the determined image areas are used as input for the ABBYY FineReader 10 OCR software.[9]

At this point, we have a list of CCs, their OCR results, and their positions on the score scans. To achieve a proper instrument labeling two additional steps are required: First, the recognized text is compared against an instrument library, see Table 4.6. The library contains names and abbreviations for typical orchestral instruments in German, English, French, and Italian. Using the Levenshtein distance [116], the library entries with the longest word count, which are the most similar to the recognitions, are identified and

[9] http://finereader.abbyy.com, February 2013

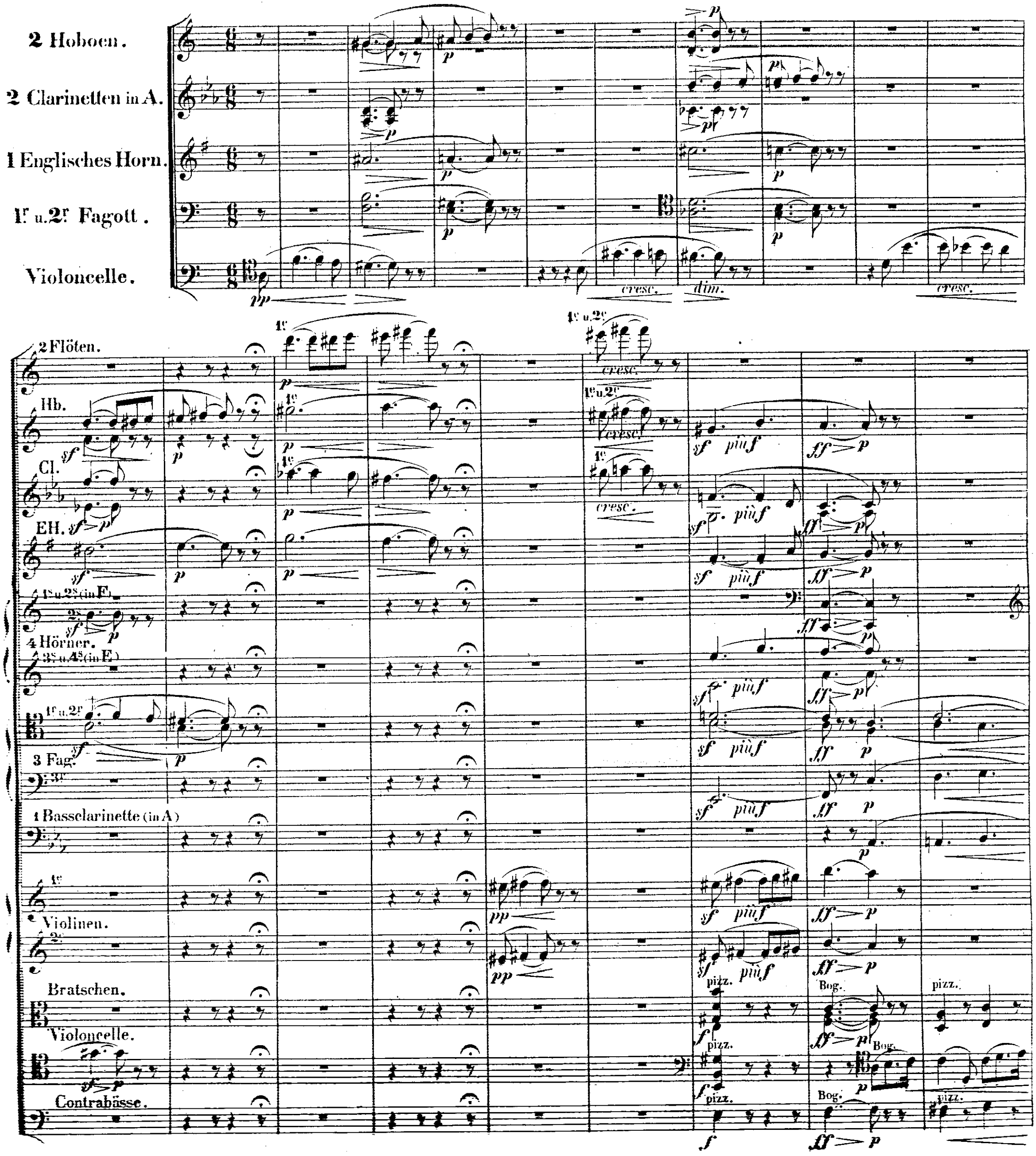
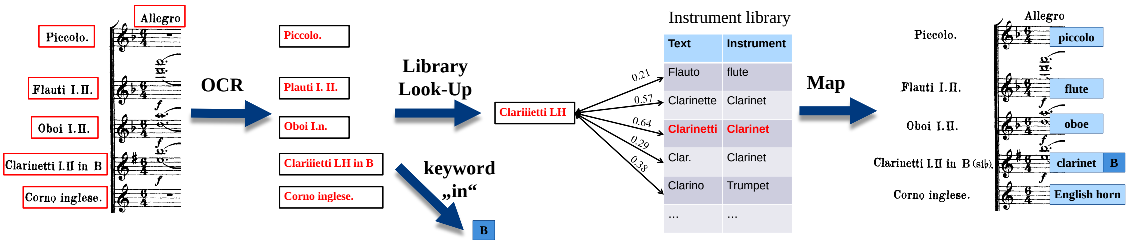


Figure 4.8. Beginning of Wagner's opera *Tristan and Isolde* in a *Breitkopf & Härtel* edition from 1860. Other than most scores – in particular modern editions – new instruments are introduced after the first system.

used as instrument labels in the according text areas.[10] Secondly, using the staff position information available in SharpEye, the identified instrument labels are mapped to the according staves of the score, see Figure 4.11.

In the majority of cases, transposition information is available from text labels like "clarinet in A". To detect transpositions, we therefore search for occurrences of text labels

[10] In the event of two equally good matches, currently the first library entry is applied. An extension of the approach to, for example, check the previous systems for occurrences of the candidates would be possible.



Text	Instrument
Flauto	flute
Clarinette	Clarinet
Clarinetti	Clarinet
Clar.	Clarinet
Clarino	Trumpet
…	…

Figure 4.9. Overview of the reconstruction of instrument and transposition labels from the textual information in the score.

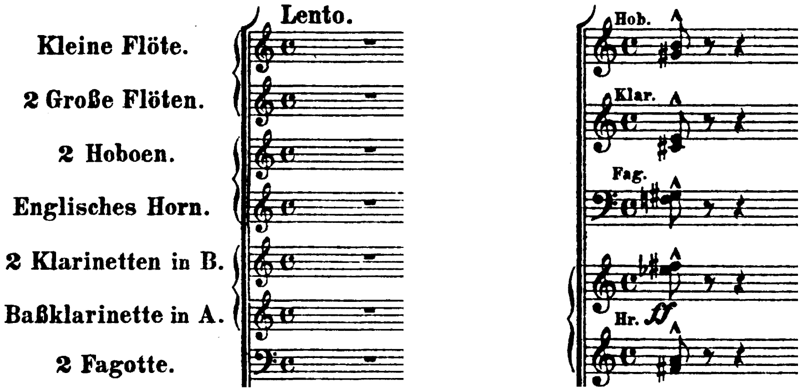


Figure 4.10. Common placement of instrument and transposition text labels in CPN using the example of Liszt's *A Symphony to Dante's Divine Comedy* (publisher: *Breitkopf & Härtel*). Given the OMR of a score image, the search for text labels can be constrained to the area in front of the staves and directly above the beginning of the staves.

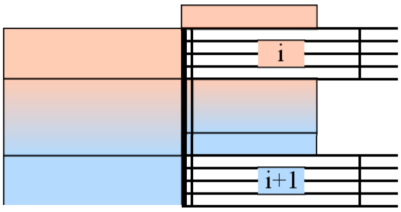


Figure 4.11. Mapping text labels to staves. While text labels placed within the red/blue area are mapped to staff i or staff $i+1$, respectively, those in the regions with a color gradient are mapped to both staves.

containing the keyword "in" followed by a valid transposition. The detected transposition labels are also mapped to the according staves.

Label	Language(s)	Group	Instrument	Type
Große Flöte.	ger	woodwind	flute	full
Große Flöten.	ger	woodwind	flute	full
Gr. Fl.	ger	woodwind	flute	abbreviation
Fl.	it ger eng fr	woodwind	flute	abbreviation
Flauti.	it	woodwind	flute	full
Flöte.	ger	woodwind	flute	full
Flöten.	ger	woodwind	flute	full
Flauto.	it	woodwind	flute	full
Flutes.	fr	woodwind	flute	full
Kl. Fl.	ger	woodwind	piccolo	abbreviation
kl. Flöte	ger	woodwind	piccolo	abbreviation
kleine Flöte.	ger	woodwind	piccolo	full
Flauto piccolo.	it	woodwind	piccolo	full
Piccolo Flöte.	ger	woodwind	piccolo	full
Piccolo-Flöte.	ger	woodwind	piccolo	full
Picc.	it	woodwind	piccolo	abbreviation

Table 4.6. Extract of the employed instrument library including full names and common abbreviations.

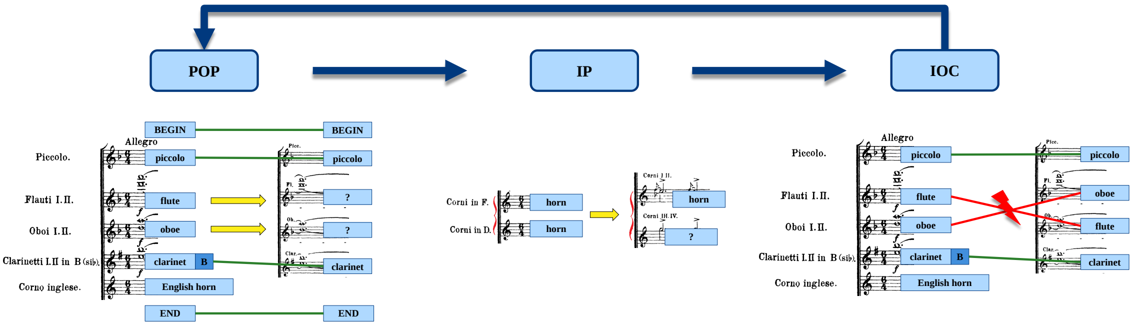


Figure 4.12. Overview of the iterative approach to the reconstruction of missing instrument and transposition labels.

4.3.2.2 Instrument Label Reconstruction

As we discussed earlier, text labels containing instrumentation and/or transposition information are often omitted if they are not essential for understanding. Therefore, after finishing the OCR-based staff labeling described in the previous section, not all staves might have a label associated. In this section, we use the labeling from the previous section as the initialization of an iterative process to reconstruct the labeling for all staves, see Figure 4.12. As both the OCRreconstruction and all information deduced through musical knowledge are uncertain, all instrument-staff mappings are equipped with plausibility values. Besides filling missing mappings, the following iterative update process also strengthens/weakens existing plausibilities.

Given a score, we define the sequence of all systems $M = (M_1, \ldots, M_m)$ and the set of all instrument labels $\mathcal{I}$ in M that were reconstructed in Section 4.3.2.1, see Figure 4.13. With $S = [1\!:\!N]$ we number all the staves in M and let $S_a \subset S$ denote the staff numbers corresponding to M_a. Furthermore, we create a matrix $\pi \in [0,1]^{|S| \times |\mathcal{I}|}$, where $|S| = N$ and

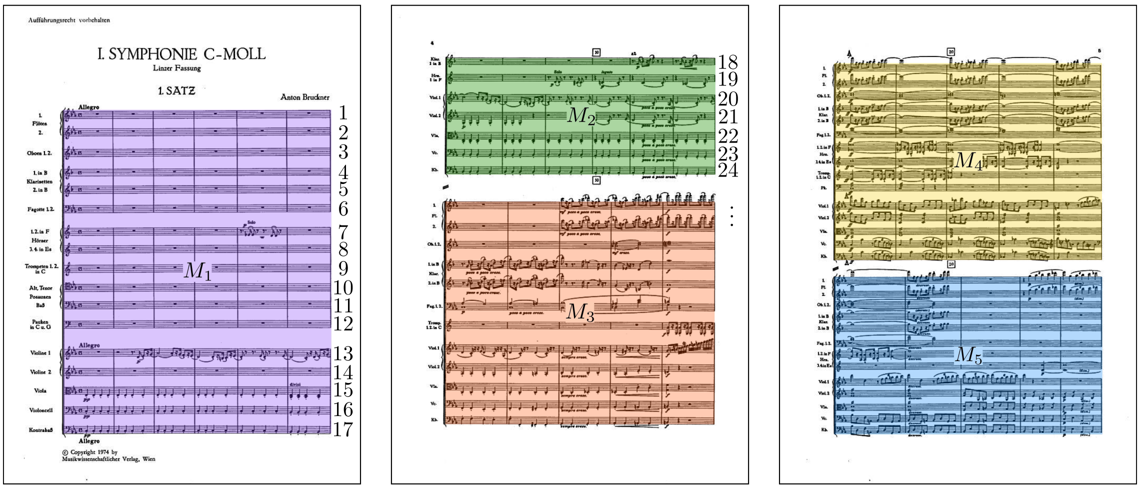


Figure 4.13. Example of system labels $M = (M_1, M_2, M_3, M_4, M_5)$ for the depicted score. Furthermore, all staves are numbered consecutively to yield $S = [1:N]$ (here $N = 64$). Thus, for example, the staff numbers of M_2 are given as $S_2 = (18, 19, 20, 21, 22, 23, 24)$.

$$\pi_1 = \begin{bmatrix} 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.965 & 0.003 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.002 & 0.888 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.003 & 0.0 & 0.95 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.997 & 0.0 & 0.002 & 0.002 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.997 & 0.0 \end{bmatrix}$$

Figure 4.14. Plausibility matrix π_1 for the first system (M_1) of score example MO3, see Table 4.1 on page 60, after the OCR-based label reconstruction.

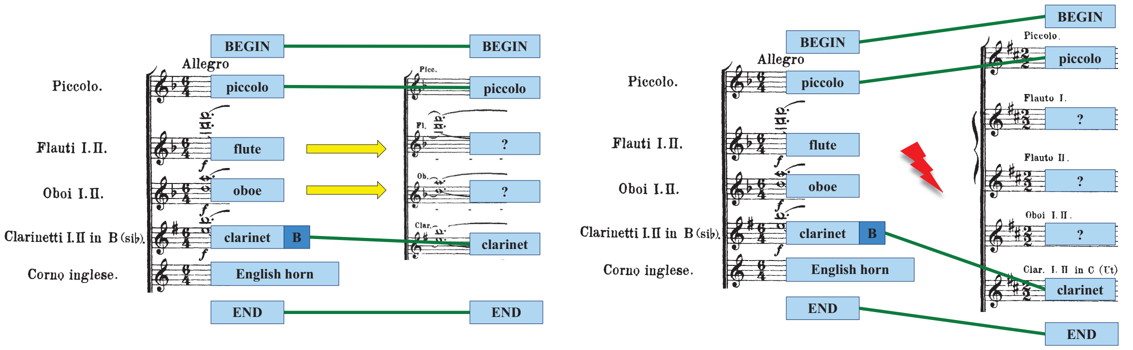
$|\mathcal{I}|$ are the cardinality of S and $\mathcal{I}$, respectively. Each entry $\pi(i, I)$ of this matrix will be interpreted as the "plausibility" of staff i having the instrument label I. The submatrix $\pi_a \in [0,1]^{|S_a| \times |\mathcal{I}|}$ corresponds to M_a. We initialize π with the instrument labels determined in the previous section, see Figure 4.14. As plausibility values, the Levenshtein distances between the instrument labels and the original instrument text on the score sheets are applied. Note that due to this initialization, several instruments might be mapped to one staff (e.g., for the text label `"viola and violoncello"`). Afterwards, the plausibility matrix $\pi^0 := \pi$ is iteratively updated using an update method that can be subdivided into three steps

$$\pi^{k+1} = IOC \circ IP \circ POP(\pi^k).$$

We will now explain these three steps of the update process in chronological order.

Propagation of Plausibilities (POP)

In this step, we propagate the already detected instrument plausibilities from system M_a to system M_b, for several $a < b$ specified below. Here, we employ the convention that the initially established instrument order in the score is not altered and apply the



(a) Valid propagation. In both systems, there exist two staves between the piccolo and the clarinet. Therefore, the instrument plausibilities of these two staves in the first system are propagated to the second system.

(b) Example of a setup where propagation is not possible. In the first system two staves are located between the piccolo and the clarinet. In contrast, there are three in the second system. Thus, an unambiguous mapping of the intermediate staves is not possible.

Figure 4.15. Sketch of the plausibility propagation step. For each system, two additional labels at the beginning and end of the system are added (BEGIN and END). These assure that a propagation can be performed, e.g., in full scores where only the first system contains text labels and all subsequent systems have the same number of staves.

following propagation rule. If two instruments occur in both systems and the number of intermediate staves between these instruments coincides, the instrument information of the intermediate staves of system M_a is propagated to the according staves in M_b, see Figure 4.15. More precisely, we first calculate the set $C_{a,b} \equiv C_{a,b}(\pi_a, \pi_b)$ consisting of all triples $(i, j, I) \in S_a \times S_b \times \mathcal{I}$ whose joint plausibility $\pi_a(i, I) \cdot \pi_b(j, I)$ is positive. We then reduce $C_{a,b}$ by removing all crossings. A crossing between two triples (i, j, I) and (k, ℓ, K) with $i < k$ occurs if $j > \ell$. In case of a crossing, the triple with smaller joint plausibility is removed. The resulting set will be denoted by $C'_{a,b}$. By projecting the elements of $C'_{a,b}$ onto the first two components, $(i, j, I) \mapsto (i, j)$, we end up with the set $C^{\times}_{a,b} \equiv C^{\times}_{a,b}(\pi_a, \pi_b)$. To deal with uninitialized systems and full scores, we add the pairs $(0, 0)$ and $(|S_a|+1, |S_b|+1)$ to $C^{\times}_{a,b}$. After sorting $C^{\times}_{a,b}$ lexicographically, we perform the following update process $\uparrow(\pi_b|\pi_a)$ for π_b given π_a.

1. For the smallest element $(i, j) \in C^{\times}_{a,b}$, search the minimal $t \geq 1$ such that $(i+t, j+t) \in C^{\times}_{a,b}$.
2. If no such t exists, goto 5.
3. Compute P_{ij} consisting of all $(i+s, j+s) \in S_a \times S_b \setminus C^{\times}_{a,b}$ such that $s \in [1 : t-1]$ and staff $i+s$ and staff $j+s$ share the same clef label.
4. For all $(\ell, I) \in S_b \times \mathcal{I}$ update π_b as follows:
   $\pi_b(\ell, I) = \max\left(\{\pi_b(\ell, I)\} \cup \{\pi_a(k, I) \mid (k, \ell) \in P_{ij}\}\right)$.
5. Update $C^{\times}_{a,b}$ by removing (i, j).
6. If $|C^{\times}_{a,b}| > 1$, goto 1.

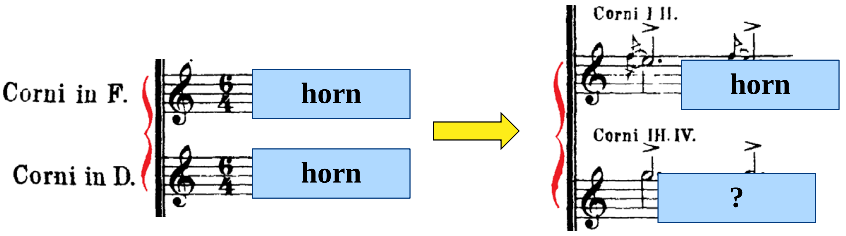


Figure 4.16. Example of how instrument properties can be applied. On the left, we detected that the two staves connected by a brace are both played by horns. On the right, only the upper staff was identified as the score for a horn. Furthermore, the brace connecting the two staves was detected. Using the knowledge from the left system, we can assume that the lower staff will be played by a horn as well and the plausibility is increased according to Equation 4.3.

Using this local update instruction, we define $POP(\pi^k)$ in two steps: First, we calculate $\tilde{\pi}_b^k :=\uparrow(\pi_b^k|\pi_1^k)$ for all $b \in [2:m]$ and then $POP(\pi_b^k) :=\uparrow\left(\tilde{\pi}_b^k|POP(\pi_{b-1}^k)\right)$. Before we proceed to the next step, we redefine $\pi^k := POP(\pi^k)$.

Applying Instrument Properties (IP)

We extract knowledge from the plausibility matrix to reconstruct missing instrument labels and to fortify already existing plausibility entries, see Figure 4.16. We define some staff-related properties $E_1, \ldots, E_p$ as subsets of S where $i \in E_j$ means that staff i has property E_j (e.g., staff i has treble clef or staff i is the first/last staff in the system). Similarly, we define properties $F_1, \ldots, F_q \subset \bigcup_{a=1}^m S_a \times S_a$ between two staves of the same system (e.g., staff i is in the same brace as staff j). We now use these staff-related properties and π to deduce instrument-related properties.

For each instrument I, we calculate the probability distribution P_I on $\mathbb{E} := \{E_1, \ldots, E_p\}$ given π:

$$P_I(E|\pi) := \frac{\sum_{i \in E} w_i \cdot \pi(i, I)}{\sum_{E' \in \mathbb{E}} \sum_{i \in E'} w_i \cdot \pi(i, I)},$$

where $w_i = \frac{3}{4}$ for staves i in S_1 and $w_i = \frac{1}{4}$ otherwise. For $(I, F) \in \mathcal{I} \times \mathbb{F}$ with $\mathbb{F} := \{F_1, \ldots, F_q\}$, we compute the probability distribution $P_{I,F}$ on $\mathcal{I}$ given π:[11]

$$P_{I,F}(J|\pi) := \frac{\sum_{(i,j) \in F} w_i \sqrt{\pi(i, I) \cdot \pi(j, J)}}{\sum_{J' \in \mathcal{I}} \sum_{(i,j) \in F} w_i \sqrt{\pi(i, I) \cdot \pi(j, J')}}.$$

Using these global instrument properties, we now define the plausibility increase

$$\pi_\Delta(I, i) := \sum_{E \in \mathbb{E}\,:\, i \in E} w_E P_I(E|\pi) + \sum_{j \in S, J \in \mathcal{I}} \; \sum_{F \in \mathbb{F}\,:\,(i,j) \in F} w_F \sqrt{\pi(j, J) P_{I,F}(J|\pi)}, \tag{4.3}$$

where w_E, w_F are suitable property weights. Using π_Δ, we define $IP(\pi^k) := \mathrm{N}(\pi^k + \pi_\Delta^k)$, where for a non-zero matrix X, $\mathrm{N}(X) := X / \max_{ij} |x_{ij}|$. We redefine $\pi^k := IP(\pi^k)$.

[11] We chose two different probability distributions to account for the differences between the two sets of properties $\mathbb{E}$ and $\mathbb{F}$.

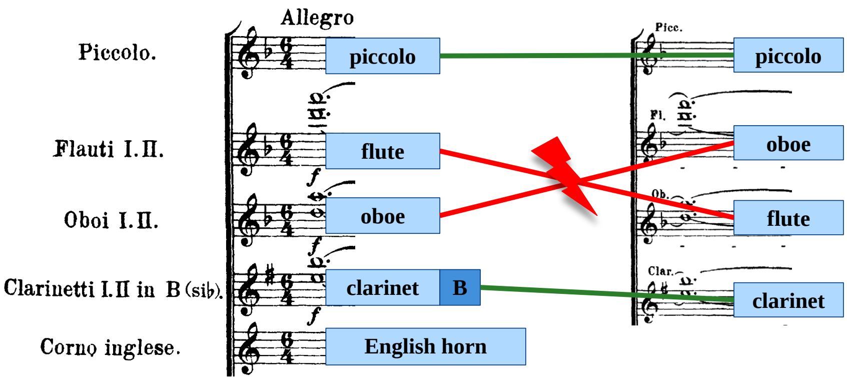


Figure 4.17. Example of a crossing between two systems. As the flute and the oboe changed their order, the corresponding plausibilities are reduced according to Equation 4.4.

Exploiting the Instrument Order Constraint (IOC)

A common convention for score notation is that the instrument order established in the first system is not altered in subsequent systems. Therefore, we use the instrument labels of S_1 to penalize systems where the instrument order established by S_1 is violated, see Figure 4.17.

Given M_1 and a system M_a, $a > 1$, we extract the sequences $\mathcal{I}_1 = (I_1, \ldots, I_{|S_1|})$ and $\mathcal{I}_a = (J_1, \ldots, J_{|S_a|})$ of most plausible instrument labels. Afterwards, we calculate the set L_{1a} of all pairs $(i, j) \in S_1 \times S_a$ with $I_i = J_j$ for which a pair $(k, \ell) \in S_1 \times S_a$ exists with $I_k = J_\ell$ such that (i, j, I_i) and (k, ℓ, I_k) constitute a crossing. The plausibility decrease

$$\pi_{\nabla,a}(j, J_j) := \lambda \sum_{i\colon (i,j)\in L_{1a}} \pi_a(i, I_i) \tag{4.4}$$

with suitable parameter $\lambda > 0$ is calculated for all $a \in [2\colon m]$. Finally, the plausibility update using the instrument order constraint is given by $IOC(\pi^k) := \mathrm{N}(\pi^k - \pi^k_\nabla)$, where $\pi^k_\nabla = \left(\pi^k_{\nabla,1}, \ldots, \pi^k_{\nabla,m}\right)$.[12]

4.3.2.3 Transposition Propagation

During the OCR-based reconstruction of the instrument labels, the available transposition information is also transformed into transposition labels and subsequently mapped to the according staves. After the reconstruction process described in the previous subsection has terminated, the transposition labels from the first system are propagated through the entire score. For each staff in S_1 holding a transposition label, the occurrences of its instrument label in the rest of the score are determined. The concerned staves will then be assigned with the transposition label from S_1.

In the context of our evaluation in Section 4.3.3 we used this method to propagate manually corrected transposition labels in the first system to the whole score.

[12] By definition $\pi^k_{\nabla,1} = 0_{|S_1|,|\mathcal{I}|}$, which is the $(|S_1| \times |\mathcal{I}|)$-null matrix.

	Instrument labels		**%**	**Transposition labels**		**%**
	total	errors		total	errors	
Compressed	401	53	87	75	17	77
Full	63	1	98	12	3	75
Total	**464**	**54**	**88**	**87**	**20**	**77**

Table 4.7. Percentage of correctly reconstructed text labels for the test collection introduced in Table 4.1.

We are aware of the fact that some orchestral scores contain transposition information next to arbitrary staves. However, extracting those short text labels that often use a very small font, see Figure 2.10 on page 14, is a new challenge and is left for future investigation.

4.3.3 Evaluation

In this evaluation, we employ the same test data as we did in Section 4.3.1, see Table 4.1 on page 60. The score editions of four pieces in our collection use a compressed notation (HA1, TC1, MO1, and WA1). Before presenting the synchronization results, we first wish to briefly comment on the accuracy of the instrument-labeling results of the proposed method. For our test data there was a total of 464 instrument text labels given in the score. In addition, 87 transposition text labels were found. The proposed label reconstruction method could correctly determine 88% of the instrument and 77% of the transposition labels, see Table 4.7. The error sources are diverse (e.g., OCR misrecognitions, unconsidered instrument abbreviations) and some will be discussed after the presentation of the synchronization results.

As in the evaluation presented in Section 4.3.1, we use the beat annotation from the RWC Music Library as ground truth and compare the mean and standard deviation from this ground truth for four different settings, see Table 4.8. The first two are the unaltered OMR output (OMR) and the OMR result with manually annotated transpositions (OMR^t) we already presented earlier. They represent the worst case (no transposition labels are available) and the best case (all transpositions are correctly annotated) achievable by our reconstruction approach. The third column of Table 4.8 shows the synchronization accuracy after applying the label reconstruction method described in Section 4.3.2 (OMR+LR).[13] In the last case, we manually corrected the transposition labels in the first system before the transposition propagation is performed (OMR+LR*).

For six pieces – one of which has a compressed score – our method produced equally good alignments as OMR^t (WA1, BR1, BR2, MO3, MO4, and ST1, see column OMR+LR). For the remaining pieces, other than HA1 and MO2 the method improved the synchronization results compared to not applying any post-processing. By annotating the transposition labels in the first system manually before propagating them through the score (OMR+LR*), the results became equal to OMR^t for all full scores and the compressed score HA1. Although manual interaction was still required, only annotating the first system constitutes a significant improvement compared to annotating all systems of an orchestral piece manually. For the compressed scores TC1 and MO1, a correct reconstruction of the transposition labels was not possible. In addition, using the propagation of the transposition labels from

[13] We performed 18 iterations of the propagation step and chose suitable experimentally determined parameter settings.

Type	**Label**	**OMR**		**OMR^t**		**OMR+LR**		**OMR+LR***	
		mean	std	mean	std	mean	std	mean	std
Compressed	HA1	456	1,016	283	441	456	1,016	283	441
	TC1	434	502	385	378	424	504	425	503
	MO1	247	349	128	178	134	183	181	247
	WA1	1,005	980	889	884	889	884	889	884
	∅	**536**	**712**	**421**	**470**	**476**	**647**	**445**	**519**
Full	BE1	462	700	265	391	284	493	265	391
	BR1	390	672	110	125	110	125	110	125
	BR2	266	803	124	84	124	84	124	84
	MO2	93	88	93	86	93	88	93	86
	MO3	243	383	65	53	65	53	65	53
	MO4	79	81	69	66	69	66	69	66
	ST1	451	658	310	492	310	492	310	492
	∅	**283**	**484**	**148**	**185**	**151**	**200**	**148**	**185**

Table 4.8. Overview of the deviation of the different synchronization results from the ground truth (in ms). OMR+LR: OMR data with automatically annotated instrument and transposition labels. OMR+LR*: the manually corrected transposition labels in the first system were propagated through the score using the reconstructed instrumentation information.

the first system results in a degradation of the synchronization compared to OMR+LR (due to instrument-labeling errors).

We will now discuss the labeling results for some scores in more detail. For two pieces the transposition text labels given in the score were not recognized. The score notation in HA1 uses an unusual setting of the transposition text labels, see Figure 4.18a. This particular text labeling results in the recognition of three separate text labels ("in", "G", and "Sol") instead of one text label (e.g., "in G"). Therefore, our method could not reconstruct the transposition labeling. In MO2 the alignment of the transposition text labels would allow for a successful recognition, but the OCR engine produced results such as "i n Sol" or "inSiw", see Figure 4.18b. In both of these examples the keyword "in" with a subsequent space followed by a transposition identifier was not available. Although for all other pieces the transposition labels in the first system were correct, some instrument-labeling errors occurred which sometimes influenced the transposition labeling of subsequent systems in a negative manner. Some of these errors result from incorrect OCR recognitions (e.g., recognition of "FI." instead of "Fl." (flute) resulted in a mapping to "Fg." (Fagott, German for bassoon)). Furthermore, some text labels are incorrectly interpreted as instrument text labels and thereby produce erroneous instrument labels. An interesting mix-up occurred for MO1. Here, Italian text labels are used and both the clarinet and the trumpet are part of the instrumentation. In Italian the trumpet is called "clarino" which is abbreviated by "Cl." However, in English this abbreviation is used for the clarinet.

4.3.4 Outlook

In this section, we present some recent work towards increasing the recognition accuracy of the proposed label reconstruction method.

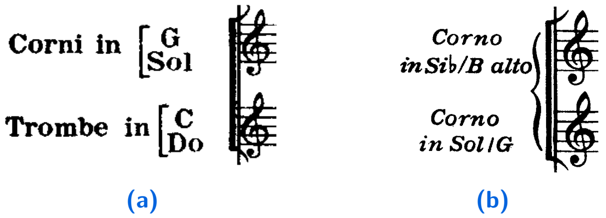


Figure 4.18. Examples of missed transposition text labels.

Language-based Filtering of OCR results

Upon evaluating the proposed method, we noticed that the label "`Cl.`" is a valid abbreviation for "`clarino`" – Italian for trumpet – as well as "`clarinet`" (English),"`clarinette`" (French), or "`Clarinette`" (German). A straightforward idea to prevent those types of language-related mix-ups is to determine the predominant language of the instrument labels in the first staff system of the score and to filter the instrument library accordingly. In the first system of the score, the full instrument names are given. Therefore, the labeling errors here are marginal and the language can usually be determined unambiguously. We employed the policy that more than half of the labels of the first system have to be identified as being of the same language. After adding the language-based filtering to our algorithm, the mentioned mislabeling of the trumpet did not occur any more.

Detection of Braces and Instrument Groupings

In the reconstruction step of our approach, the properties "staff i is in the same brace as staff j" and "staff i is in the same group as staff j" can be used to determine the correct instrumentation of a system. Unfortunately, SharpEye does not reliably support the reconstruction of this information.[14] To evaluate the effect of the two aforementioned properties, Wagner [204] manually annotated the missing data for one piece of music. He concluded that knowing and using the braces and groupings does not have a positive effect on the outcome of the reconstruction. However, a more extensive evaluation might be advisable. Furthermore, new rules could be designed to exploit this information. For example, given the groupings in the first system, the plausibility of instrument pairs in later systems can be reduced if they are in the same group by mistake. As a first step, we want to propose two approaches for the reconstruction of groups and braces, respectively.

To determine the instrument groups in a staff system, one has to check for gaps in the measure lines. We already know the approximate location of the measure boundaries from the SharpEye OMR reconstruction. Using this information, we follow a measure line and search for gaps of sufficient length (at least half the height of a staff line). To deal with the possibility of the measure line being crooked, we re-estimate the current center of the line after each step along the y-axis. After all gaps in the system were detected, the respective staves above and below each gap are determined. Commonly, all measure lines – except for the first and the last – are disrupted between two instrument groups. For the purpose of robustness, see Figure 4.19, we perform the described gap detection for all inner measure

[14] The brace detection of SharpEye retrieved less than 10% of the braces in our test collection, and instrument groups were not detected at all.

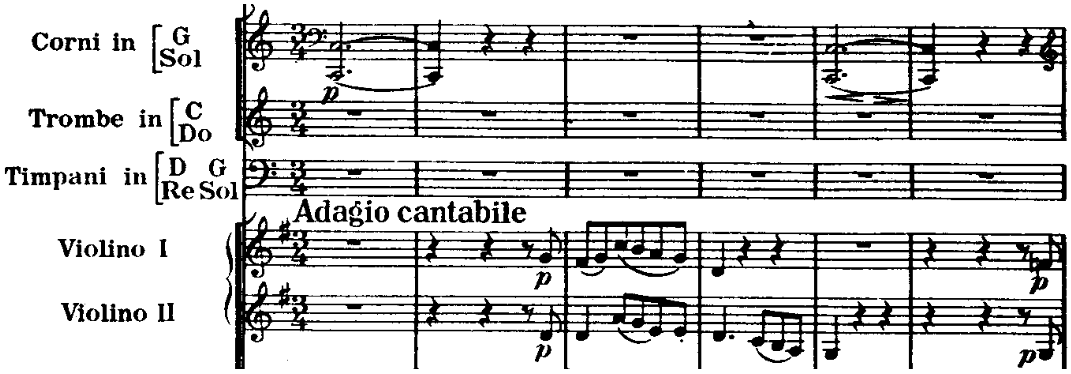


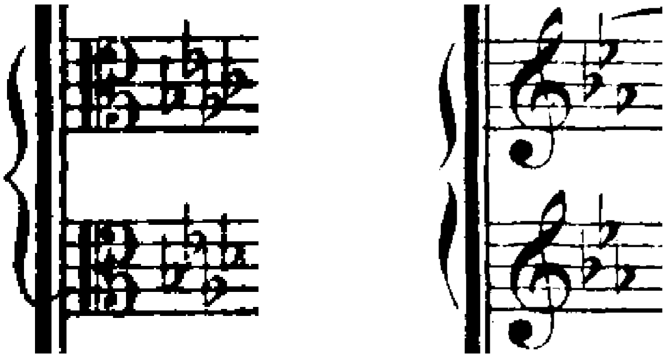
Figure 4.19. Example of a problematic gap detection. If only the end of the first measure in the system was checked for sufficiently large gaps, the gap between the timpani and the violin would not be detected. By considering all measure lines, the beginning of a new group could correctly be detected (score extract from *Symphony No.* 94 by J. Haydn, publisher: *Kalmus*).

lines in the staff system. A gap denotes the beginning of a new group if this gap was detected for at least half of the inner measure lines in the system.

In CPN, it is a hard rule that the braces are placed directly in front of the staff system. We further know that a brace is a rather narrow symbol. By searching for CCs (i.e., connected components) of black pixels in the immediate vicinity of the beginning of the staff system, we will find all of the braces. Especially in older print editions, the ink often spilled and, therefore, the braces sometimes touch the staff system, see Figure 4.20a. To separate braces from the system, we perform a modified version of the line-following employed for the detection of instrument groups. After determining the position of the first vertical line in a system, we flip all corresponding pixels to white. Thus, we remove the line and reliably separate braces from the staff system. After retrieving all CCs in front of the system, we apply various filters to eliminate false positives. Two examples are that the CCs have to exceed a minimal number of pixels and the width/height ratio has to indicate a vertically long and horizontally narrow symbol. Often, the print or the scan are of lower quality and, therefore, some pixels might erroneously be white instead of black. This can lead to a brace being split into two or more CCs, see Figure 4.20b.[15] To reunite a brace, we merge pairs of CCs that are sufficiently close to each other. Finally, the detected brace candidates are mapped to the staves.

An OMR system capable of recognizing braces and groups is capella-scan.[16] In an evaluation on our previously introduced test collection 91% (182 out of 201) of the braces were correctly recognized. Disrupted measure lines and thus instrument groups were detected more or less correctly. However, five systems were incorrectly split into multiple systems. In comparison, our approach detected all instrument groupings and reconstructed 196 braces (96%).

[15] This issue also has to be considered by the applied filters. Otherwise, parts of a brace might be dismissed.
[16] `http://www.capella.de/de/index.cfm/produkte/capella-scan/info-capella-scan`, February 2013



(a) Publisher: *Dover Publications*

(b) Publisher: *Breitkopf & Härtel*

Figure 4.20. Examples of critical braces. **(a)** the brace touches the line of the staff system. **(b)** the brace is disrupted in the middle. These types of printing-related issues occur frequently and have to be considered by the brace detection method.

4.4 Applications

First, we give a detailed overview of existing user interfaces that employ synchronization techniques (using scanned sheet music or symbolic score data). Then, we focus on current issues in MIR and show how to exploit sheet music-audio synchronization to solve them.

4.4.1 Graphical User Interfaces

Wedelmusic [17] is one of the first systems presenting score images and audio data simultaneously. During playback a marker moves through the sheet music to identify the currently audible musical position. In addition, page turning is performed automatically by gradually replacing the current sheet/system with the next one. However, the employed automatic synchronization approach is rather simple. Using the start and end points in the sheet music and the audio as anchor points, linear interpolation was applied. As local tempo deviations may result in alignment errors, a user interface for the manual rework of the proposed synchronization was available. In Section 3.1, we gave an overview of existing digital music library systems. Some of those systems, i.e., Variations2, the IEEE 1599 standard, and `musescore.com`, utilize score-audio alignments to support new and enhanced document-access mechanisms. However, all of them require the manual calculation of the synchronization data, and only initial work towards large-scale automatic computation has been reported. With the Probado Music system introduced in Chapter 3, a digital music library system for the management of large document collections was developed. The front end of Probado Music employs sheet music-audio alignments to provide a multimodal music presentation with score-following and score-based navigation, see Figure 3.12 on page 43. The required alignment paths are calculated nearly automatically using the techniques presented in this chapter. The linking structures further allow for score-based query formulation. More precisely, Probado Music offers techniques for score-retrieval, audio-retrieval, and lyrics-retrieval using the query-by-example paradigm. As all documents related to the same piece of music are linked to each other, the user can utilize the score or any other visualization to formulate a query of any one of these types.

Xia et al. [210] presented a rehearsal management tool for musicians that exploits semi-automated score-audio synchronization. Here, recordings of various rehearsals are clustered and aligned to a score representation of the piece of music. Additional challenges are introduced by the fact that the recordings can differ in length and may cover different parts of the piece. Another application designed to support musicians is automated accompaniment. To this end, online score-audio synchronization determines the current position in the score as well as the current tempo to replay a time-stretched audio recording. Two well-known accompaniment systems are *Music Plus One* by Raphael [161, 163] and ANTESCOFO by Cont [51, 52].

4.4.2 MIR Research

There are various MIR tasks that exploit score information as additional knowledge. For example, in score-informed source separation one assumes that along with the audio recording a synchronized MIDI file is given. With this file, the occurring note events as well as their position and duration in the audio are specified. We refer to Ewert and Müller [71] for an extensive overview. At the moment, all approaches use symbolic score data (e.g., MIDI), but score scans may be applicable as well. However, in this case recognition errors need to be considered by the source separation method. A similar task is the estimation of note intensities in an audio recording where the notes are specified by a symbolic representation [70]. Again, to avoid the manual creation of a MIDI file, the exploitation of score scans together with sheet music-audio synchronization techniques seems reasonable.

Another important research topic is lyrics-audio synchronization [77, 207]. Instead of using the commonly employed speech analysis techniques, sheet music can be added as additional information. Thereby, the lyrics can be derived from the OMR results. Afterwards, the lyrics-audio alignment can be calculated by means of the sheet music-audio synchronization [55, 142, 174].

There are several other tasks where score-audio synchronization could reduce the complexity of the problem. Some examples are structure analysis [140], chord recognition [47, 104], and melody extraction [96, 196].

5 Audio Matching

Digital audio formats in combination with today's storage capabilities enabled the development of digital audio collections on a grand scale. Thereby, content-based retrieval techniques are becoming increasingly important. Content-based retrieval for audio files can be categorized into four types. Given an audio snippet, the task of *audio identification* is to find the exact audio recording it originates from. In *version identification* or *cover song retrieval*, for a given piece of music or a sufficiently large audio snippet thereof, all corresponding recordings that can be regarded as a version or reinterpretation of the same piece, e.g., played by different musicians or using different instrumentations, are to be retrieved. In *audio matching*, rather than whole recordings, audio extracts that are similar to a query snippet are to be reported as retrieval results. The last retrieval type is *category-based music retrieval*, where the recordings in the music collection are categorized or clustered with respect to some predefined cultural or musicological category, e.g., genre or mood. Given a recording, category-based retrieval can then propose similar songs to the user. For details on the different approaches and an extensive bibliography, we refer to [86].

In this chapter, we focus on the audio matching scenario. Here, queries are formed (at least implicitly, see Section 5.3) by audio snippets of arbitrary length, and the retrieval results are arbitrary audio extracts from the collection that bear some similarity to the query. In particular, one recording might even contain several hits. Therefore, to find all occurrences of a given query, a comparison of the query with all feasible contiguous subsegments of the collection is necessary. Two prominent approaches for solving this task are *diagonal matching* (or *linear scan*) and *subsequence dynamic time warping* (SSDTW), see Section 5.1 for details. In diagonal matching, a sequential warping-free comparison of the query with subsegments of the collection (having the same length) is performed. By using appropriate feature representations, local variations (e.g., in timbre, harmony, or instrumentation) can be canceled out. However, the approach is constraint to comparing sequences of equal length and, therefore, does not consider tempo variations. To allow for global tempo variations of ±40%, Kurth and Müller [107] proposed the retrieval of several time-stretched versions of the query. In contrast, SSDTW enables the comparison of feature sequences with different length by performing non-linear warping. Thereby, SSDTW allows for both global and local tempo variations. In addition, a higher robustness towards local variations, e.g., insertions and deletions, is achieved.

Unfortunately, the good retrieval quality comes at the cost of SSDTW being rather slow. For DTW, the imposition of global constraints on the admissible warping paths is a common approach to improving the runtime [133, Section 4.2.3]. However, this approach is not applicable to SSDTW as employed in the audio matching context. Besides, the runtime of classical diagonal matching becomes problematic for large datasets as well. For both approaches, the precalculation of a retrieval index has proven to be a reasonable approach for handling large collections. Kurth and Müller [107] proposed an indexing approach for diagonal matching that significantly speeds up the retrieval process while causing only minor quality losses, see Section 5.1.2. The general idea is to use a set of reference feature vectors (codebook vectors) and to store their occurrence positions in the database using inverted lists. During retrieval the matches can be determined using shifted versions of the relevant inverted lists in combination with fast list intersections. Another approach widely used in audio identification and version identification is indexing based on locality-sensitive hashing (LSH), see, e.g., [44, 85, 211]. After converting the audio data into some feature representation, the feature sequence of the audio collection is segmented into equal-sized subsequences (*shingles*), which are subsequently stored using LSH. During retrieval, the feature sequence of a query is split into shingles as well. For each query shingle, all shingles with the same hash value are retrieved from the index before applying some merging approach to determine valid matches. For example, Casey et al. [44] counted the number of matching shingles, while Yang [211] employed the Hough transform.

The issue with DTW (in general) is that it cannot be indexed without loss of quality. Instead, several approaches suggest the introduction of a *lower bounding function* (LBF) for DTW that enables a fast filtering of match candidates and can be indexed using multidimensional indexing methods such as R-trees, e.g., [72, 101, 151, 160, 212]. After index look-up, these candidates are subsequently verified by calculating the DTW distance. Agrawal et al. [3] combined the LBF-based indexing approach with the previously mentioned shingling to deal with subsequence matches (i.e., the query is part of one of the documents). However, all of these approaches are constraint to DTW comparisons of the query (or query shingles) with equal-sized subsequences from the database and thus weaken the benefits of SSDTW-based matching.

In conclusion, diagonal matching achieves short response times but lacks the flexibility of SSDTW to search for arbitrarily time-warped (globally and locally) versions of a query. Similarly, the discussed approaches to indexing of SSDTW can only retrieve matches of the same length as the query and require online verification of candidates via DTW calculation. For large candidate lists, this step can potentially result in long response times. In Section 5.2, we propose a new procedure whereby the advantages of index-based retrieval and SSDTW are combined. To this end, we utilize the fact that in practical applications queries are often given as audio extracts from the music collection itself (*intra-collection query*). As we will see, this leads to a simple yet very efficient and effective retrieval approach that combines the efficiency of indexing techniques with the retrieval quality of classical SSDTW-based matching. Evidently, indexing the retrieval results for all possible queries, which would be the obvious first idea, is not feasible for larger collections. Instead, we follow the shingling approach and split the dataset into overlapping segments of equal length, calculate the corresponding audio matches, and store them as search indexes. During query processing the indexes of the segments covering the query are merged to calculate the retrieval result. In Section 5.2.2, we present a set of experiments for evaluating the proposed method. Depending on the size of the music collection we observed speed-up factors between 42 and 311 in comparison to classical SSDTW-based audio matching.

Our descriptions in this chapter, in particular Sections 5.1.1, 5.2.1, 5.2.2, 5.3, and this introduction, in large parts follow our publication [191].

5.1 Audio Matching

In this section, we provide a detailed description of two common approaches to audio matching, i.e., diagonal matching and SSDTW, which served as basis and inspiration for the intra-collection audio matching method proposed in Section 5.2.

The first step in both approaches is to transform the audio collection and the query into a suitable feature representation. Let Q be the query audio clip and let $(D_0, D_1, \ldots, D_N)$ be the collection of audio recordings. To simplify matters, we create a large dataset document D by concatenating all documents $D_0, D_1, \ldots, D_N$, where we keep track of the document boundaries in a supplemental data structure. Subsequently, Q and D are transformed into feature sequences $\mathbf{Q} = (\mathbf{Q}_0, \mathbf{Q}_1, \ldots, \mathbf{Q}_K) \in \mathcal{F}^{K+1}$ (with $|\mathbf{Q}| = K+1$) and $\mathbf{D} = (\mathbf{D}_0, \mathbf{D}_1, \ldots, \mathbf{D}_L) \in \mathcal{F}^{L+1}$, respectively. A valid feature choice for Western classical music collections would be one of the chroma features introduced in Section 2.4.1. Using the previously introduced cost measure $c(x, y) := 1 - \langle x, y \rangle$ subsequences $(\mathbf{D}_k, \mathbf{D}_{k+1}, \ldots, \mathbf{D}_{k+M})$, with $k \in [0 : L - M]$, of $\mathbf{D}$ that are sufficiently similar to the query sequence $\mathbf{Q}$ are determined. While the dynamic programming approach in Section 5.1.1 is capable of detecting matching subsequences with $M \neq K$, the diagonal matching approach in Section 5.1.2 retrieves subsequences with the same total length as the query. Our descriptions of the two audio matching approaches closely follow [133, Sections 4.4 and 6.4].

5.1.1 Subsequence Dynamic Time Warping

In this section, we describe a variant of the DTW approach introduced in Section 4.2.1 that is called *subsequence DTW* (SSDTW). Instead of calculating a global alignment, the objective in SSDTW is to search for subsequences within a long sequence that optimally fit a much shorter sequence.

Given the cost measure $c(x, y) := 1 - \langle x, y \rangle$, we define a distance function $\Delta_{\mathbf{Q}}^{\mathbf{D}} : [0 : L] \to [0, \infty]$ between $\mathbf{Q}$ and $\mathbf{D}$ that locally compares $\mathbf{Q}$ to subsequences of $\mathbf{D}$

$$\Delta_{\mathbf{Q}}^{\mathbf{D}}(\ell) = |\mathbf{Q}|^{-1} \min_{a \in [0 : \ell]} \left(\mathrm{DTW}(\mathbf{Q}, \mathbf{D}(a : \ell)\right). \tag{5.1}$$

Here, $\mathbf{D}(a : \ell)$ denotes the subsequence of $\mathbf{D}$ starting at index a and ending at index ℓ and $\mathrm{DTW}(\mathbf{Q}, \mathbf{D}(a : \ell))$ denotes the DTW distance between $\mathbf{Q}$ and $\mathbf{D}(a : \ell)$ as defined in Equation 4.1 on page 55.

Each entry $\Delta_{\mathbf{Q}}^{\mathbf{D}}(\ell)$ of the distance function measures the distance between $\mathbf{Q}$ and the subsequence $\mathbf{D}(a_\ell : \ell)$ of $\mathbf{D}$, where $a_\ell = a_\ell(\mathbf{Q}, \mathbf{D})$ denotes the minimizing index in Equation 5.1, see Figure 5.1 for an example. As we apply DTW, it is usually true that $|\mathbf{Q}| \neq |\mathbf{D}(a_\ell : \ell)|$.

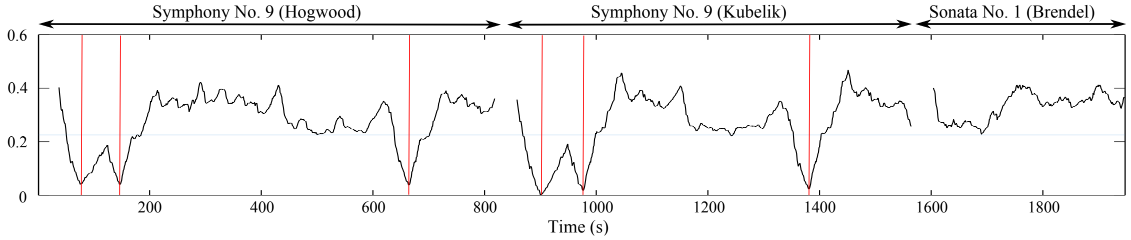


Figure 5.1. Distance function with respect to a query consisting of the beginning of *Symphony No.* 9, *Molto Vivace* by L. v. Beethoven in an interpretation conducted by R. Kubelik. For two different interpretations of the *Molto Vivace*, the distinct peaks at the three match locations in each document are visible (vertical red lines). The horizontal blue line indicates the ranking threshold $\theta = 0.225$. As expected, for a different piece (*Piano Sonata No.* 1 by L. v. Beethoven), no matches are reported.

To remove the penalization of omissions at the beginning and the end of the alignment path, we modify the definition of the accumulated cost matrix D as follows

$$D(n,m) = \begin{cases} C(0,0) & \text{if } n = m = 0 \\ \sum_{k=0}^{n} C(k,0) & \text{if } n \in [1:K] \text{ and } m = 0 \\ C(0,m) & \text{if } n = 0 \text{ and } m \in [1:L] \\ C(n,m) + \min \begin{cases} D(n-1,m) \\ D(n,m-1) \\ D(n-1,m-1) \end{cases} & \text{otherwise.} \end{cases}$$

As in Equation 4.2 on page 56, Σ_1 was employed as set of admissible steps. The last row $D(K,:)$ of the accumulated cost matrix then coincides with the output of the distance function $\Delta_{\mathbf{Q}}^{\mathbf{D}}$.

The best match between Q and D is now encoded by the index $\ell_0 \in [0:L]$ minimizing $\Delta_{\mathbf{Q}}^{\mathbf{D}}$. The distance value $\Delta_{\mathbf{Q}}^{\mathbf{D}}(\ell_0)$ is also referred to as the *ranking value* of the match corresponding to the feature sequence $\mathbf{D}(a_{\ell_0} : \ell_0)$. As the goal is to find all audio extracts that are similar to the query, the calculation then continues by searching for the second best match. But first a neighborhood of ℓ_0 is excluded from further considerations to avoid overlaps between matches. In our implementation, we exclude the region $[a_{\ell_0} : \ell_0 + 0.85 \cdot (\ell_0 - a_{\ell_0})]$ by setting the respective $\Delta_{\mathbf{Q}}^{\mathbf{D}}$-values to ∞. Then, to find subsequent matches, the above procedure of identifying the minimum of $\Delta_{\mathbf{Q}}^{\mathbf{D}}$ is performed repeatedly until the minimal distance exceeds a specified distance threshold θ or until a certain number of matches is obtained. This way, we iteratively compute all matches of $\mathbf{Q}$ in $\mathbf{D}$ (using the threshold-condition)

$$H(\mathbf{Q}) := \left\{ (a_\ell, \ell, r) \mid \ell \in [0:L], r = \Delta_{\mathbf{Q}}^{\mathbf{D}}(\ell) \leq \theta \right\}.$$

For more details on SSDTW, we refer to [133, Section 4.4].

5.1.2 Diagonal Matching

Diagonal matching can essentially be seen as a special case of SSDTW where the set of admissible step sizes is set to $\Sigma = \{(1,1)\}$. This equals a point-wise comparison of the query sequence $\mathbf{Q} = (\mathbf{Q}_0, \mathbf{Q}_1, \ldots, \mathbf{Q}_K)$ with all subsequences $(\mathbf{D}_x, \mathbf{D}_{x+1}, \ldots, \mathbf{D}_{x+K})$, for $x \in [0:L-K]$. The ranking value is simply calculated as the sum of the local cost $\Delta_{\mathbf{Q}}^{\mathbf{D}}(x) = \sum_{i=0}^{K} c(\mathbf{Q}_i, \mathbf{D}_{x+i})$.

Codebook Vectors and Index-based Retrieval

Calculating the matches as proposed results in a computational complexity of $\mathcal{O}(L)$. To improve the runtime, Kurth and Müller [107] developed an index-based approach that uses inverted file indexes. In what follows, we describe the fundamental idea of the approach.

A crucial component of the proposed index-based retrieval is the definition of a suitable *codebook* consisting of a finite set $\mathcal{C}$ of characteristic feature vectors $\mathcal{C}_0, \ldots, \mathcal{C}_R \in \mathcal{F}$. To determine a suitable codebook, unsupervised learning techniques can be applied. Instead, domain knowledge can be exploited to manually identify a good set of feature vectors. In addition to the codebook, a quantization function $\mathcal{Q} : \mathcal{F} \to [0\!:\!R]$ is defined whereby every feature vector $v \in \mathcal{F}$ is assigned to a class label $\mathcal{Q}[v] \in [0\!:\!R]$ defined by

$$\mathcal{Q}[v] := \underset{r \in [0:R]}{\operatorname{argmin}} \left(\arccos\left(\langle v, \mathcal{C}_r \rangle\right)\right).$$

We will now define the retrieval index for a document collection D and explain how it is calculated. First, the feature sequence $\mathbf{D} = (\mathbf{D}_0, \mathbf{D}_1, \ldots, \mathbf{D}_L)$ is transformed into a quantized sequence $\mathcal{Q}[\mathbf{D}] := (r_0, \ldots, r_L)$ where $r_i := \mathcal{Q}[\mathbf{D}_i]$ is the quantization of feature vector $\mathbf{D}_i$ with respect to the codebook $\mathcal{C}$. For each class label $r \in [0\!:\!R]$ we calculate an *inverted list*

$$\mathcal{L}(r) := \{m \in [0\!:\!L] \mid r_m = r\}$$

of all index positions m in $\mathcal{Q}[\mathbf{D}]$ whose quantized vector equals $\mathcal{C}_r$. The *inverted file index* of the database D consists of all inverted lists $(\mathcal{L}(r))_{r \in [0:R]}$. We can precompute this index in a preprocessing step.

To process a query Q, we compute its feature sequence $\mathbf{Q} = (\mathbf{Q}_0, \ldots, \mathbf{Q}_K)$ and subsequently the quantized sequence $\mathcal{Q}[\mathbf{Q}] = (s_0, \ldots, s_K)$. Given an inverted list, we define its p-shifted version by

$$\mathcal{L}(r) - p := \{m - p \mid m \in \mathcal{L}(r)\}.$$

Now, the set of all match positions is given by

$$H(\mathcal{Q}[\mathbf{Q}]) := \{k \in [0\!:\!L - K] \mid \forall n \in [0\!:\!K] : s_n = r_{k+n}\}$$

and can be calculated efficiently by intersecting suitably shifted inverted lists

$$H(\mathcal{Q}[\mathbf{Q}]) = \bigcap_{n \in [0:K]} (\mathcal{L}(s_n) - n).$$

5.2 Fast Intra-Collection Audio Matching

By employing SSDTW, a high robustness towards global and local tempo variations as well as small local variations is achieved. However, for large datasets, the required sequential scanning results in long response times. For applications where the query originates from within the dataset, we therefore propose the calculation of an audio matching index that allows for fast SSDTW-based audio matching in larger datasets. The general idea is to split the dataset into small overlapping segments, perform the previously described SSDTW-based audio matching procedure, and to store the result lists as retrieval index. During retrieval, those lists are used to efficiently determine the matches of a given query.

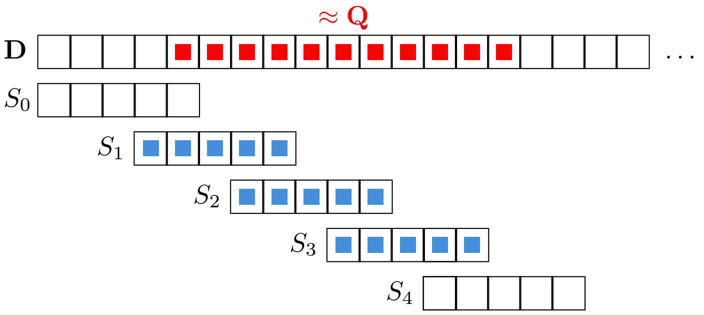


Figure 5.2. Segmentation of $\mathbf{D}$ ($\lambda = 5$, $\tau = 3$). For the approximate feature representation $\mathbf{D}(4\colon 14)$ of query Q (red) the best-fitting segment subsequence is $S_{\mathbf{D}}(1\colon 3)$. In a sense, this best-fitting coverage corresponds to minimizing the symmetric difference between the red and the blue squares projected onto the $\mathbf{D}$-strip.

5.2.1 Index Creation and Retrieval Strategy

Given Q and D, we calculate their respective feature sequences $\mathbf{Q} = (\mathbf{Q}_0, \mathbf{Q}_1, \ldots, \mathbf{Q}_K) \in \mathcal{F}^{K+1}$ and $\mathbf{D} = (\mathbf{D}_0, \mathbf{D}_1, \ldots, \mathbf{D}_L) \in \mathcal{F}^{L+1}$. In our implementation, we chose the CRP features introduced by Müller and Ewert, see Section 2.4.2, using their implementation provided by the Chroma Toolbox [138]. We employ non-overlapping features with a window size of one second. For CRP features the feature space $\mathcal{F}$ consists of all elements in $[-1, 1]^{12}$ that have euclidean length 1. The cost measure c and the distance function $\Delta_{\mathbf{Q}}^{\mathbf{D}}$ are defined as in Section 5.1.1.

Index Calculation

Given a segment length $\lambda \in \mathbb{N}, \lambda > 1$ and a step size $\tau \in [1\colon \lambda]$, the segmentation $S_{\mathbf{D}}$ of feature sequence $\mathbf{D}$ is defined as

$$S_{\mathbf{D}} = (S_0, S_1, \ldots, S_M)$$

where $M = \left\lceil \frac{|\mathbf{D}|-\lambda}{\tau} \right\rceil$ and $S_m = \mathbf{D}(m\tau\colon m\tau + \lambda - 1)$ for $m \in [0\colon M-1]$ and $S_M = \mathbf{D}(M\tau\colon |\mathbf{D}| - 1)$, see Figure 5.2. If not stated otherwise, we use $\lambda = 20, \tau = 5$.

For each segment S_m, $m \in [0\colon M]$, we perform the SSDTW-based audio matching procedure presented in Section 5.1.1 and store the respective retrieval results as inverted lists. We use the modified list definition

$$\mathcal{L}(m) = \{m\} \times H(S_m) \tag{5.2}$$

containing all tuples (m, a_ℓ, ℓ, r) with $\ell \in [0\colon L]$ and $r = \Delta_{S_m}^{\mathbf{D}}(\ell) \leq \theta$ (we set $\theta = 0.225$).

The computational complexity of this preprocessing step is in $\mathcal{O}(\lambda \cdot M \cdot |\mathbf{D}|) = \mathcal{O}(|\mathbf{D}|^2)$. Therefore, its calculation becomes rather time consuming for larger collections. However, the index creation can be accelerated significantly by employing distributed processing. Furthermore, a given index does not need to be recalculated if a new audio track D_{N+1} is added. Instead, two steps are required: First, the inverted lists for all segments in D_{N+1} have to be calculated (using $(D_0, D_1, \ldots, D_{N+1})$ as music collection). Second, all segments in $(D_0, D_1, \ldots, D_N)$ need to be queried in D_{N+1} to update their respective inverted lists. In this way, the computational complexity for adding a new document D_{N+1} (usually: $|\mathbf{D}_{N+1}| \ll |\mathbf{D}|$) is in $\mathcal{O}(|\mathbf{D}_{N+1}| \cdot |\mathbf{D}|)$.

Retrieval Method

As the given query Q is a subsegment of D, we can approximate its feature representation by a subsequence $\mathbf{D}(i:j)$ of $\mathbf{D}$, with $0 \leq i < j < |\mathbf{D}|$. Then, we determine the subsequence $S_{\mathbf{D}}(i^*:j^*)$ of segments in $S_{\mathbf{D}}$ through which the query feature sequence $\mathbf{D}(i:j)$ is properly represented. We use a best-fitting coverage of the query by setting $i^* = \left\lfloor \frac{i}{\tau} \right\rceil$ and $j^* = \left\lfloor \frac{j+1-\lambda}{\tau} \right\rceil$, see Figure 5.2.[1]

Subsequently, the inverted lists $\mathcal{L}(m), m \in [i^*:j^*]$, of all segments in $S_{\mathbf{D}}(i^*:j^*)$ are loaded. Using these, we will now calculate the index-based (approximate) set of matches $\tilde{H}(\mathbf{Q})$.[2] To this end, we employ a similar approach as the index-based diagonal matching method described in Section 5.1.2. A fundamental observation is that for a subsegment $\mathbf{D}(u:v)$ to be a valid retrieval result, a sufficient number of inverted lists $\mathcal{L}(m), m \in [i^*:j^*]$, need to contain a match whose start and end points lie within $[u:v]$. To determine such subsegments $\mathbf{D}(u:v)$ efficiently, we exploit the fact that the regions of elements in the individual inverted lists belonging to the same match can be made overlapping by shifting them appropriately. For a list $\mathcal{L}(m)$, $m \in [0:M]$, as defined in Equation 5.2, we define its *p-shifted* version by

$$\mathcal{L}(m) - p := \{(m, a_\ell - p, \ell - p, r) \mid (m, a_\ell, \ell, r) \in \mathcal{L}(m)\}$$

and create the list of all shifted segment retrieval results

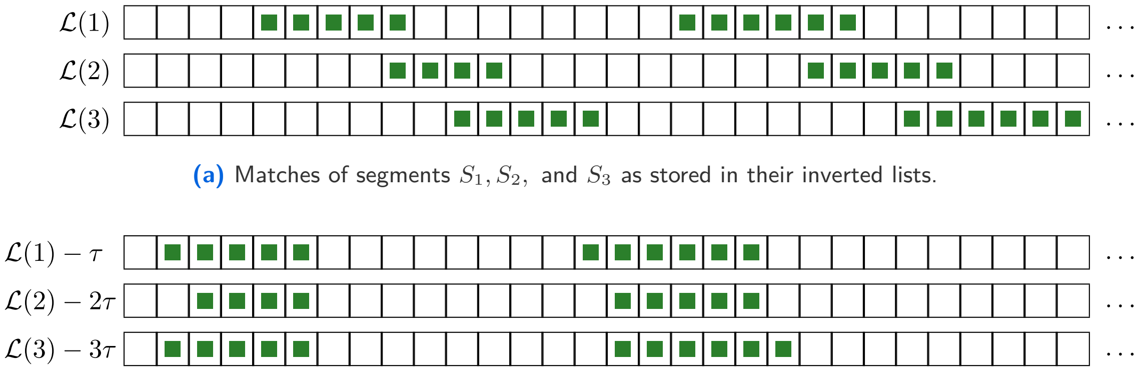
$$B(\mathbf{Q}) = \bigcup_{m \in [i^*:j^*]} \mathcal{L}(m) - m\tau. \tag{5.3}$$

The shifting procedure is illustrated in Figure 5.3. As we employ SSDTW, the regions are of varying length and, therefore, do not necessarily become identical after list shifting. Using $B(\mathbf{Q})$ we will now calculate $\tilde{H}(\mathbf{Q})$ in two steps: First, we determine all regions $\mathbf{D}(u:v)$ that contain sufficiently overlapping elements of $B(\mathbf{Q})$, see Algorithm 1. Second, the ranking value of these regions is calculated and their eligibility as retrieval results is tested, see Algorithm 2.

For a match $(m, a_\ell, \ell, r) \in \mathcal{L}(m)$, let $\overline{a}_\ell := a_\ell - m\tau$ and $\overline{\ell} := \ell - m\tau$ represent the $m\tau$-shifted versions of the start and end positions of the match, a_ℓ and ℓ, respectively. By sorting $B(\mathbf{Q})$ by the shifted start indexes of the matches, we now derive the sequence $\mathcal{B}(\mathbf{Q}) = ((s_k, \boldsymbol{a}_k, \boldsymbol{\ell}_k, r_k))_{k\in[0:|B(\mathbf{Q})|-1]}$ with $\boldsymbol{a}_k := \overline{a}_{\ell_k}, \boldsymbol{\ell}_k := \overline{\ell}_k$ and $\boldsymbol{a}_0 \leq \boldsymbol{a}_1 \leq \ldots \leq \boldsymbol{a}_{|B(\mathbf{Q})|-1}$. Further, s_k denotes the segment index the according match originates from and $r_k = \Delta^{\mathbf{D}}_{S_{s_k}}(\ell_k)$ is its ranking value. Then, we define $\mathcal{S}_{u,v} := \{s_u, s_{u+1}, \ldots, s_v\}, u \leq v \in [0:|\mathcal{B}(\mathbf{Q})| - 1]$, as the duplicate-free set of segment indexes between u and v. In Algorithm 1, we now step through $\mathcal{B}(\mathbf{Q})$ and combine successive entries to form the set $\mathcal{I}(\mathbf{Q})$ of merged retrieval candidate regions. Here, only those segments that overlap sufficiently with the other segments of a candidate region are added to this region, see line 4 in Algorithm 1. There are two conditions a group of segment matches has to satisfy to be a valid retrieval candidate: First, the offset between the shifted start position of the first and the last segment in the group must be smaller than λ. Second, the shifted start positions of subsequent segment matches must not differ by more than $\lambda/4$.

[1] $\lfloor x \rceil$ rounds x to the nearest integer.

[2] For $i^* = j^*$, $\tilde{H}(\mathbf{Q}) = \mathcal{L}(i^*)$.



(a) Matches of segments $S_1, S_2,$ and S_3 as stored in their inverted lists.

(b) Shifted versions of the segment matches. According to Equation 5.3, the matches of segment S_m have to be shifted by $-m\tau$.

Figure 5.3. Illustration of the shifting of inverted lists. We continue the example from Figure 5.2 and load the inverted lists of the segments $S_1, S_2,$ and S_3 and shift the contained matches accordingly. As the example demonstrates, the shifted match positions do not need to be exactly the same. Therefore, the lists have to be merged (instead of intersected as in the index-based diagonal matching approach described in Section 5.1.2).

Algorithm 1. merge regions

```
$\mathcal{I}(\mathbf{Q}) \leftarrow \emptyset$
$k \leftarrow 0$
while $k < |\mathcal{B}(\mathbf{Q})|$ do
    search maximum $t \in [0 : |\mathcal{B}(\mathbf{Q})| - 1 - k]$ with
    (1) $|\boldsymbol{a}_{k+t} - \boldsymbol{a}_k| < \lambda$
    (2) $\forall p < t : |\boldsymbol{a}_{k+p+1} - \boldsymbol{a}_{k+p}| \leq \lambda/4$
    if $t > 0$ then
        $\mathcal{I}(\mathbf{Q}) \leftarrow \mathcal{I}(\mathbf{Q}) \cup \{(k, k+t)\}$
    end if
    $k \leftarrow k + |\mathcal{S}_{k,k+t}|$
end while
```

The ranking value of a region in $\mathcal{I}(\mathbf{Q})$ is defined as the average mean of the ranking values from the partial matches in the segments $S(i^* : j^*)$ (only one ranking value per segment is used). In addition, we apply a penalty factor $(j^* - i^*) \cdot |\mathcal{S}_{u,v}|^{-1}$ whereby the ranking value of matches with only a few contributing segments is degraded, see line 3 in Algorithm 2.

In line 4 of Algorithm 2, the eligibility of a match as a retrieval result for Q is tested. To this end, we apply two conditions: First, matches formed by at least half the segments present in the query that have a ranking value $R_{u,v} \leq \theta$ are valid matches. However, by means of the second condition, we also allow partial matches with $|\mathcal{S}_{u,v}| < \frac{1}{2}(j^* - i^*)$ that have a very good ranking.

Algorithm 2. verify candidates

```
$\tilde{H}(\mathbf{Q}) \leftarrow \emptyset$
for all $(u,v) \in \mathcal{I}(\mathbf{Q})$ do
  $R_{u,v} := \left(\sum_{j\in\mathcal{S}_{u,v}} r_j\right) \cdot (j^* - i^*) \cdot |\mathcal{S}_{u,v}|^{-2}$
  if $\left[R_{u,v} \le \theta \textbf{ and } |\mathcal{S}_{u,v}| \ge \frac{1}{2}(j^* - i^*)\right]$
   or $\left[R_{u,v} \cdot (j^* - i^*)^{-1} \cdot |\mathcal{S}_{u,v}| < 0.1 \cdot \theta\right]$ then
    $\tilde{H}(\mathbf{Q}) \leftarrow \tilde{H}(\mathbf{Q}) \cup \left\{\left(\min_{t\in[u:v]}(a_{\ell_t}), \max_{t\in[u:v]}(\ell_t), R_{u,v}\right)\right\}$
  end if
end for
```

5.2.2 Evaluation

In this section, we report on a series of experiments to indicate how the proposed index-based audio matching approach for intra-collection retrieval performs in comparison to the classical SSDTW-based matching. First, we show that through indexing the response times decrease considerably. Afterwards, we examine the quality of the reported matches.

Experimental Setup

We prepared two collections featuring audio recordings of Western classical music. The first dataset C_1 comprises 444 tracks that contain four different interpretations of all piano sonatas by L. v. Beethoven. In total, C_1 consists of 44.7 hours of audio. While the first collection is constrained to piano music, the second set C_2 ($C_2 \supset C_1$) additionally contains orchestral music and several songs for voice and piano. In particular, C_2 contains six recordings of the *Symphony No.* 9 by L. v. Beethoven, two of which are piano versions based on the piano transcription by F. Liszt. Overall, the second collection is significantly larger as it comprises 2,012 audio tracks yielding a total of 141 hours of music.

All algorithms were implemented in the Matlab (7.11.0) environment and all experiments were conducted on a standard PC. Furthermore, we used a feature resolution of 1 Hz and employed the Σ_2 set of admissible steps. If not stated otherwise, the parameter settings for our indexing method are: $\theta = 0.225, \lambda = 20$, and $\tau = 5$.

Query Length and Response Time

In this experiment, we compare the performance of the classical SSDTW-based audio matching procedure described in Section 5.1.1 (label: SSDTW) to the performance of our index-based matching approach (label: $\text{SSDTW}_{\text{index}}$) by measuring the average response times. Besides comparing the two approaches, we focused on three aspects: the impact of the query length on the runtime by using audio snippets with durations of $25 - 125$ seconds as queries, the impact of the size of the data set on the response time by performing searches for all queries in C_1 as well as C_2, and finally we compare the runtime of the index-based method for indexes consisting of all retrieval results with a ranking value $\le \theta$ ($\text{SSDTW}_{\text{index}}^{\le\theta}$) to indexes containing at most the best 1,000 matches of each segment

query	C_1			C_2		
length (s)	SSDTW	$\text{SSDTW}^{\leq\theta}_{\text{index}}$	$\text{SSDTW}^{1{,}000}_{\text{index}}$	SSDTW	$\text{SSDTW}^{\leq\theta}_{\text{index}}$	$\text{SSDTW}^{1{,}000}_{\text{index}}$
25	3.15	0.13	0.08	19.51	0.28	0.06
45	4.45	0.15	0.08	24.43	0.30	0.08
65	5.79	0.20	0.11	30.35	0.45	0.11
85	7.23	0.23	0.12	34.84	0.51	0.13
105	8.85	0.27	0.15	39.81	0.61	0.16
125	10.63	0.29	0.16	45.02	0.66	0.18

Table 5.1. Comparison of the response times (in seconds) for SSDTW, $\text{SSDTW}^{\leq\theta}_{\text{index}}$ and $\text{SSDTW}^{1{,}000}_{\text{index}}$.

($\text{SSDTW}^{1{,}000}_{\text{index}}$). For each setup, we performed 24 runs. The measured average runtimes are depicted in Table 5.1.

For $\text{SSDTW}^{\leq\theta}_{\text{index}}$ the speed increase over SSDTW ranges from 24 (25s query in C_1) to 81 (45s query in C_2), whereas for $\text{SSDTW}^{1{,}000}_{\text{index}}$ the speed-up factors increase even further and go from 39 to 325. In addition, comparing the runtimes for C_1 and C_2, the scalability of the index-based procedure with respect to the size of the audio collection becomes apparent. While for SSDTW the response times for queries in C_2 on average increase by a factor of five (compared to C_1), with $\text{SSDTW}^{1{,}000}_{\text{index}}$ they remain nearly constant.

Furthermore, the evaluations show that with increasing query length both approaches decrease in their performance. However, for the 125s queries $\text{SSDTW}^{1{,}000}_{\text{index}}$ still achieves response times below 0.2s. Finally, with respect to the runtime a clear advantage of applying a top-1,000 strategy – especially for larger datasets – over the threshold-based indexing becomes apparent (up to 4.5-times-faster responses).

In a further set of experiments we evaluate the competitiveness (in terms of runtime) of our approach with DTW indexing methods that apply a lower bounding function, e.g., [101,212]. For this purpose, we calculated the average runtime of the required candidate verification step, which yields a lower bound on the total runtime. Using queries of $25 - 125$s length the verification of 20 candidates required $0.15 - 0.24$s. For 100 candidates we observed response times of 0.66 to 1.22s. These results suggest that our method is a competitive alternative for intra-collection audio matching scenarios. All reported evaluations were performed for audio collection C_2.

All in all, the presented evaluations show a significant efficiency boost by applying the proposed index-based audio matching method. However, to access its practical usability one should also examine the quality of the created matches.

Matching Quality

We present a variety of experiments on the performance of $\text{SSDTW}_{\text{index}}$ in terms of the matching quality. As the proposed method is intended as a fast index-based approximation of SSDTW-based audio matching, we compare the generated matches to those calculated by SSDTW.

In the first set of experiments, we evaluate the impact of the segment length λ used during index creation on the quality of the retrieval results. On the one hand, the selected segment

C_1	$\lambda = 10$			$\lambda = 20$		
	EM	T 20	T 30	EM	T 20	T 30
Q_1	1.00	0.45	0.53	1.00	0.80	0.70
Q_2	1.00	0.60	0.53	1.00	0.75	0.67
Q_3	1.00	0.55	0.40	1.00	0.65	0.60
Q_4	1.00	0.40	0.33	1.00	0.65	0.60
Q_5	1.00	0.65	0.50	1.00	0.70	0.53
Q_6	1.00	0.55	0.50	1.00	0.80	0.77
Q_7	0.88	0.60	0.43	1.00	0.85	0.80
Q_8	0.75	0.30	0.23	1.00	0.80	0.67
$\varnothing$	**0.95**	**0.51**	**0.43**	**1.00**	**0.75**	**0.67**

Table 5.2. Recall of $\mathrm{SSDTW}_{\text{index}}^{\leq\theta}$ for different segment length λ in relation to the matches of SSDTW-based audio matching. The results of $\mathrm{SSDTW}_{\text{index}}^{1{,}000}$ coincide with the depicted values.

length naturally directly influences the minimal query length processable by $\mathrm{SSDTW}_{\text{index}}$. Therefore, too large values will render the approach useless for real-life applications. On the other hand, too short queries usually result in numerous insignificant matches. We chose to compare the performance of $\mathrm{SSDTW}_{\text{index}}$ for $\lambda = 10$ and $\lambda = 20$. In our experiment, we use C_1 as the data collection and take the first 21 measures of the *Piano Sonata No.* 1 by L. v. Beethoven as the query. This extract is played twice during each of the four performances in C_1 (due to a repetition). We use both the first and the second repetition in each performance as query, thereby receiving a total of eight queries ($Q_1 - Q_8$) with durations between 21 and 25 seconds. Obviously, the collection contains eight exact matches for each query. The classical SSDTW approach ranks them as the top eight matches (no matter which query $Q_1 - Q_8$ is used). The column labeled “EM” in Table 5.2 presents the recall values for those exact matches using $\mathrm{SSDTW}_{\text{index}}$ (i.e., ratio of exact matches occurring in the top eight matches). While for $\lambda = 10$ some queries do not result in a perfect recall, no qualitative difference to the classical SSDTW approach is observable for $\lambda = 20$.

Furthermore, we evaluate the recall for the 20/30 best-ranked queries (i.e., ratio of 20/30 best matches calculated by SSDTW also belonging to the 20/30 best retrieval results when using $\mathrm{SSDTW}_{\text{index}}$). The results are shown in the columns “T 20” and “T 30” of Table 5.2. Here, the impact of the segment length on the result quality becomes even more distinct. While with $\lambda = 10$ only one half of the top 20 matches was retrieved, for $\lambda = 20$ an accordance of 0.75 could be achieved. This supports our assumption that too short queries seem to result in a great deal of insignificant matches and thus consequently reduce the overall retrieval accuracy.

In the experiment presented in Table 5.1, we already showed that the truncated index $\mathrm{SSDTW}_{\text{index}}^{1{,}000}$ attains up to 4.5 times better response times compared to $\mathrm{SSDTW}_{\text{index}}^{\leq\theta}$. In the next experiment, we now compare the performance of the two indexes in terms of matching quality. Furthermore, we will extend our experiments to the larger collection C_2 and introduce a second set of queries to evaluate the effect of larger collections and the chosen query on the retrieval results.

Again, we use $Q_1 - Q_8$ as queries and perform audio matching on C_1 (see Table 5.2, results for $\lambda = 20$) as well as C_2, see Table 5.3. For C_2 we additionally use audio snippets from the beginning of *Symphony No.* 9, *Molto vivace* by L. v. Beethoven as queries ($Q_9 - Q_{14}$, 70 − 76s). As the beginning of the *Molto vivace* is repeated twice during the piece, each

C_2	$\mathrm{SSDTW}^{\leq\theta}_{\mathrm{index}}$			$\mathrm{SSDTW}^{1{,}000}_{\mathrm{index}}$		
	EM	T 20	T 30	EM	T 20	T 30
Q_1	1.00	0.75	0.63	1.00	0.75	0.63
Q_2	1.00	0.70	0.60	1.00	0.70	0.60
Q_3	1.00	0.85	0.70	1.00	0.85	0.70
Q_4	1.00	0.75	0.63	1.00	0.75	0.63
Q_5	1.00	0.80	0.83	1.00	0.80	0.83
Q_6	1.00	0.75	0.77	1.00	0.75	0.77
Q_7	1.00	0.65	0.87	1.00	0.65	0.87
Q_8	1.00	0.70	0.63	1.00	0.70	0.63
∅	**1.00**	**0.74**	**0.71**	**1.00**	**0.74**	**0.71**
Q_9	1.00	0.90	0.63	1.00	0.90	0.67
Q_{10}	1.00	0.90	0.63	1.00	0.90	0.67
Q_{11}	1.00	0.90	0.67	1.00	0.90	0.63
Q_{12}	1.00	0.90	0.63	1.00	0.90	0.67
Q_{13}	1.00	0.90	0.67	1.00	0.95	0.67
Q_{14}	1.00	0.95	0.70	1.00	0.90	0.73
∅	**1.00**	**0.91**	**0.66**	**1.00**	**0.91**	**0.67**

Table 5.3. Recall values for $\mathrm{SSDTW}^{\leq\theta}_{\mathrm{index}}$ and $\mathrm{SSDTW}^{1{,}000}_{\mathrm{index}}$.

interpretation contributes three exact matches, thereby generating a total of 18 exact matches for $Q_9 - Q_{14}$ (six of which are extracts of the piano recordings).

For $Q_1 - Q_8$ (both in C_1 and in C_2) no differences between the matching results of $\mathrm{SSDTW}^{1{,}000}_{\mathrm{index}}$ and $\mathrm{SSDTW}^{\leq\theta}_{\mathrm{index}}$ could be detected. In contrast, subtle difference could be observed for $Q_9 - Q_{14}$. However, the overall performance remains unchanged, see Table 5.3.

The presented results, in combination with the runtime evaluations discussed in the previous section, suggest that $\mathrm{SSDTW}^{1{,}000}_{\mathrm{index}}$ with $\lambda = 20$ constitutes a good trade-off between speed, processable query length, and performance.

5.3 Applications

Usually, libraries and museums that provide access to their digital audio collections (possibly consisting of thousands of audio tracks) do not allow visitors to connect their USB devices to upload queries. Therefore, the users are constrained to searching within the given collection using extracts of the available audio as queries (i.e., intra-collection search). The proposed method was designed specifically with such library systems in mind and aims at providing fast and accurate retrieval results for these intra-collection query scenarios. This way, our procedure allows users, for example, to quickly find and access repetitions of a music extract in all available recordings of the underlying piece or to search a database for pieces of music that borrow ideas from other pieces.

The digital music library system PROBADO MUSIC, see Chapter 3, is also constrained to intra-collection audio retrieval.[3] Intra-collection retrieval is particularly interesting as it allows convenient user interfaces for query formulation to be created. For example, rich

[3] In contrast, the free creation of lyrics and score queries through adequate search masks is provided as these do not require query-by-humming or plugging in a USB device, see Figure 3.10b and Figure 3.10c on page 42.

audio visualizations like spectrograms, see Figure 3.11 on page 43, can be offered to assist the user in selecting a query within the audio collection. Furthermore, for a given music recording, the corresponding scanned score sheets can be made available. In these cases, each position in the score can be linked to a corresponding position in an audio recording by using the sheet music-audio synchronization techniques described in Chapter 4. Thus, queries can be formulated using an intuitive score-based interface where the computed linking information is used to automatically translate the queries into the audio domain. Equally, the linking information can be employed to indicate the matches in the score, see Figure 3.16 on page 46.

6 Motivic Analysis

In the previous chapter, we introduced the task of content-based audio retrieval. We particularly focused on audio matching and aimed at finding sections in a large audio database that are in some respect similar to a given audio snippet. For other types of music representations such music matching tasks can be defined as well. Given a collection of lyrics, similar text passages can be retrieved, and for symbolic scores musically similar score extracts can be searched. The Probado Music system introduced in Chapter 3 implements such *lyrics retrieval* and *score retrieval* algorithms and provides user interfaces for query formulation and result visualization. For details on the employed methods, we refer to Chapter 3 and the references cited therein.

Besides a differentiation of the type of music representation used for retrieval, one can also distinguish retrieval tasks based on their search space. While in *inter-opus* retrieval a whole collection of music documents is queried, in *intra-opus* retrieval only the document the query originates from is searched. In the case of focusing on one document, another scenario is the automatic detection of interesting repeating fragments and their match positions. Repetition is an important stylistic element of music from all areas, genres, and cultures, and it is used on all detail levels. Large sections are repeated (unchanged or in slight modification) to give a piece of music its structure, also called the musical form. This occurs both in classical music, e.g., the typical fugue forms, and in popular music by means of dividing a song into choruses and verses. On finer detail levels, short note sequences are reused to capture the listener's attention. Furthermore, those *motifs* are used to develop the next-larger form elements such as *phrases* and *figures* through repetition, variation, and imitation of their musical material.

So far, the Probado Music system does not provide any functionality for intra-opus retrieval or for the automatic detection of repeating fragments within a piece of music. In this chapter, we want to present our work towards supporting those features for symbolic music documents. The motif is commonly regarded as the smallest structural unit in music that still maintains independence. Thus, it can be seen as the basic element to be considered when analyzing/determining the musical form of a piece of music. The detection of motifs as well as their repetitions within a piece of music constitutes the goal of the musicological discipline of *motivic analysis*. In Section 6.1, we give a brief introduction to *musicology* before presenting some details on motivic analysis and typical types of motif variations in Western classical music. In Section 6.2, we present related work on computer-aided motivic analysis, and in Section 6.3, we introduce our own computational approach. Using similarity matrices exact repetitions and repetitions in retrograde and/or inversion are detected. In contrast to most approaches, both pitch intervals and durations are considered individually and in combination to account for different types of variation. Following the idea of subsequence relations between musical patterns presented by Adiloğlu and Obermayer [2], a pattern hierarchy is calculated as well. In addition, we introduce an interactive graphical user interface for online motivic analysis, see Section 6.4.1.

6.1 Musicological Theory

Musicology (*Musikwissenschaft* in German) can be defined as the academic study of music. It emerged as its own discipline during the second half of the 19th century. In 1885, Guido Adler proposed the subdivision of what we will refer to as the *classic musicology* into two sub-disciplines, see [121, *Musicology, §I: The nature of musicology*]. On the one hand, there is *historical musicology* where the music history of Western culture is studied. More precisely, this sub-discipline focuses on the development of notation types, musical forms and genres, historical laws, and music instruments over time. On the other hand, Adler introduced a field labeled *systematic musicology*, which again contains several sub-fields that loosely speaking address questions about music in general. Grove Music Online [121] defined systematic musicology as a field that studies the *'tabulation of the chief laws applicable to the various branches of music'* and includes music theory, aesthetics, music education, and (comparative) musicology as its sub-fields.[1] At the end of the 20th century, the study of music as a social force became a new trend and resulted in the emergence of the *new musicology* (also *critical* or *cultural musicology*). Studies are no longer restricted to Western classical music, and the fields of research include, among others, culture, context, gender, and identity. Despite this change of focus, classic musicology and its subdisciplines remain an important part of musicological studies.

The composer, pianist, and music critic Robert Schumann stated in his *Musikalische Haus- und Lebensregeln*, *'The Spirit will not become clear to you, before you understand the forms of composition'* [177, page 32], [5]. With exactly this intention the systematic musicological discipline *Formenlehre* (*study of form* in English) developed during the 19th century. Following Schumann's opinion, the basic objective of the study of form is to facilitate the access to current and past music through the description of musical form principles [5]. In the beginning, the studies were closely oriented towards the instrumental pieces by Ludwig van Beethoven. This shaped the field notably as even today studies of musical form mostly concentrate on instrumental pieces of music from the time period between 1600 and 1900 [105]. Blessinger [27] described the study of form as a discipline that aims at detecting common external characteristics of classical instrumental pieces to create a fixed number of types that have to be sorted systematically.[2] This definition illustrates two things: First, the previously mentioned concentration on instrumental music and, second, the fact that the study of form is about defining musical forms and classifying pieces with regard to those forms. Musical form itself can be defined as the constructive, organizing element in music that governs the presentation, development, and interrelationship of ideas [121]. Further, form does not only comprehend the basic structure of a work, but

[1] `http://www.oxfordmusiconline.com/subscriber/article/grove/music/46710`, February 2013

[2] *'...stellt sie sich die Aufgabe, die den klassischen Instrumentalsätzen gemeinsamen äußeren Merkmale zu erkennen, zu sammeln und daraus eine Anzahl fester Typen zu konstruieren, die systematisch zu ordnen sind.'* [27, page 11]

also the techniques and procedures used to develop ideas within the structure. Techniques commonly referred to are: repetition, variation, contrast, development, diversity, sequence, and unrelatedness [105], [205, *Musical form*]. Other important components of musical form are the structural units (also elements, models) of music which, by applying the stated techniques, give rise to the musical forms.[3] In the literature, one can find a variety of definitions of the structural units in music. For example, Berry stated that the *'progress of music in time achieves form and intelligibility through the occurrences of small groupings of sound'* [24, page 1] and *'[it] is from small structural units – especially the motives and phrases – that large forms evolve'* [24, page 2]. Similarly, Stöhr [183] stated that the understanding of forms requires the knowledge of the elements building those forms.[4] The most important units of musical form are (roughly ordered by size): motif, soggetto, figure, phrase, theme, period, group, part, and movement [130].

6.1.1 Motivic Analysis

For this section, we used a large collection of literature. For the sake of readability, we refrain from stating the source of each individual statement – except for original quotes. Instead, we now provide a list of all used literature [5,24,27,89,100,105,114,121,147,164, 172,173,175,178,182,183].

According to Scruton, motivic analysis *'shows how the audible structure of a piece is derived from basic elements or motifs'* [178, page 398]. Thus, it can be considered fundamental for the study of form. Effectively, the overall goal of motivic analysis is the detection of all motifs as well as all their reoccurrences/variations in a piece of music. To this end, Gingerich [79] stated that the process of creating a motivic analysis of a piece of music can be subdivided into three stages: 1) identify all motifs in the given piece of music, 2) detect all their reoccurrences and describe how they are varied, and 3) determine the function of the motivic development with respect to the whole piece of music. These stages do not need to be processed in the above order, and often a proper analysis even requires going back and forth between them. While this general description appears rather clear and comprehensible, this often does not translate into practice, *'In practice, analysis operates on the basis of fuzzy and ill-defined terminology – so much that, when all is said and done, motifs, themes, or phrases will often be identified intuitively'* [147, page 159]. Upon consulting the literature for definitions of the musical term *motif*, the mentioned ill-definedness becomes quickly apparent. Most definitions only differ in their nuances and essentially capture very similar notions. However, sometimes one also encounters contradicting statements. Two such examples of contradictions will be presented in the course of this section. Several textbooks on musicology and the study of form even hint at these inconsistencies [27,100,147]. Even if one ignores the vagueness of the definition there is vast space for interpretation when it comes to determining the motifs of a piece of music. For example, Berry stated that *'[the] lines of distinction, especially between motive and phrase, are often a question of subjective impression; precise and absolute definitions that will apply for all listeners in all cases are not possible'* [24, page 3]. When it comes to

[3] *'[Music] is essentially abstract, and its structural components achieve integration chiefly by their corroboration through repetition'* [24, page 1].

[4] Original: *'Das Verständnis der Formen setzt die Kenntnis der dieselben zusammensetzenden Elemente voraus.'* [183, page 76]

developing computer-based or computer-aided systems for motivic analysis, this lack of clarity and objectiveness constitutes a major concern.

We will now try to give a description of the term *motif* by collecting and combining the definitions encountered in the literature. A notion that is common to nearly all definitions is that of the motif being the smallest/shortest musical unit or cell that in some way is self-existent, intelligible, or characteristic. At this point, we already encounter the first contradiction [173]. On the one hand, most textbooks and dictionaries define the motif as being the smallest independent musical unit. On the other hand, a motif can be further subdivided into submotifs.[5] Thus, strictly speaking, the motif containing the submotifs cannot be the smallest musical unit. Often coupled with the notion of the motif being a cell is its significance as motivating idea for the rest of the piece. Some examples are, *'smallest characteristic unit whose significance is established in development'* [24, page 4], *'energy source and developable seed'*,[6] *'seed of musical development'*.[7] A closely related aspect that also occurs quite often in definitions is the repetition or restatement of the motif in the piece of music.[8,9] Basically, those statements say that a sequence of notes has to occur at least twice within a single piece of music to be considered a motif. The restatement does not need to be an exact repetition of the sequence but can be a variation. We will discuss in detail the types of possible variations of a motif at the end of this section. The motif as a *musical seed* together with its variations eventually gives rise to larger musical structures (e.g., phrase, period) and thus constitutes the foundation of any musical form.

But what part of a given note sequence turns it into a motif? Which components have to reoccur to call a later note sequence a variation of the motif? In principle, a motif can be of melodic, rhythmic, or harmonic nature or any combination of these. In other words, the restatement of a rhythmic motif has to have a very similar rhythm, while the harmony and melody can be completely different. In our initial definition of the motif it was also referred to as a characteristic unit. So, what makes a note sequence characteristic? Altmann [5] said that to be characteristic and to be perceived as such by the listener, the note sequence has to be rather fast with no or only very short pauses in between. In addition, the already mentioned restatement has to be made in order to remind the listener of the note sequence and to strengthen his or her recollection of it.

In addition to the features mentioned thus far, some textbooks also comment on the typical length of motifs (roughly $2 - 12$ notes [182]) and their position and distinction within the notes through rests or other musical incisions, *'Often, they [the motifs] are punctuated by means of metric division, by rests, by articulation, or by a momentary cessation of movement on a longer note'* [24, page 4].

Historically, two types of motif can be distinguished. In the polyphonic music until the middle of the 18th century, the so-called *Fortspinnungsmotiv* (*spinning-forth motif* in English) was predominant. Here, the motif was mainly of a melodic-rhythmic character. As the name indicates, the motivic material is constantly repeated whilst freely continued

[5] *'Das Motiv kann aus ... Teilmotiven bestehen.'* [213, page 141]

[6] Original: *'Energiequelle und entwicklungsfähiger Keim'* [114, page 24]

[7] Original: *'Keimzelle einer musikalischen Entwicklung'* [27, page 60]

[8] *'Thus, the potential and significance of a thematic fragment may be unapparent until it is subjected to manipulation in the course of a work.'* [24, page 3]

[9] *'Die Eigenschaft Motiv zu sein kommt einer Tonfolge nur rückwirkend zu: als Folge des Wiederauftauchens ... in derselben oder in variierter Form.'* [173]

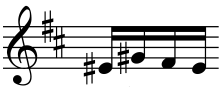
using the motivic impulse to create new variations which are often difficult to recognize. In the homophonic oriented music of the 18th and 19th century, the *Entwicklungsmotiv* (*developmental motif* in English) was commonly used. Unlike in the spinning-forth motif, all variations are clearly recognizable and the original motivic material is still apparent in all variations. In the classical music of the 20th century, both motif types could be found. For example, Schönberg's famous *twelve-tone technique* included rules for the restatement of the set[10] in prime, inversion, retrograde, and retrograde-inversion (see below for the definitions) [176]. When speaking of motifs and their variations, most often the developmental motif is actually meant. The same holds for the motif variations we will discuss below – they are predominantly variations typical for developmental motifs.

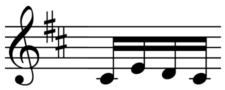
One or multiple motifs give rise to the *theme*, a characteristic musical unit. A piece of music, i.e., a movement, usually only contains a few themes,[11] while there can exist a significantly larger number of motifs. In the definition of the theme as a note sequence composed of motifs we could find a further inconsistency. Stöhr [183] described the theme as a *'stretched and furthermore slowly played melody'*.[12] However, as previously stated, a motif – from which a theme develops – is often understood to consist of a fast note sequence.

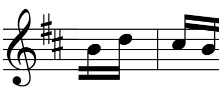
Before concluding our musicological excursion, we will present a list of the most commonly used and cited motivic variations. We want to clarify that the presented list is and cannot be absolute. Over the course of their professional lives, composers usually develop their own recognizable style. Of course, this can also include their own types of motivic variations.[13]

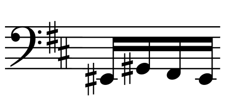
Repetition

The most basic restatement is that of repeating the motif in the same voice (*sequence*) or in another voice (*imitation*), possibly in a different pitch (transposition). One further distinguishes between *real repetitions*, where the size of the intervals between consecutive notes remains unchanged and *tonal repetitions* that allow for small interval changes in favor of keeping the key. Below, we illustrate the different types of repetitions using the example of a four-note motif from the *Invention in B minor* by J. S. Bach (original source [24]). The score extracts are based on the free scores available from the *Werner Icking Music Archive* (WIMA) [93].


Original motif


Real repetition


Tonal repetition


Real imitation

[10] As basis of the twelve-tone technique, the twelve tones of the chromatic scale are arrange in a specific ordering. This ordering is called the *set* (*Grundgestalt* in German) on which a piece is based.

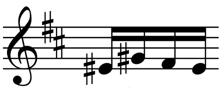
[11] Barlow and Morgenstern [14,187] created a dictionary of 9,825 themes for the works of over 150 composers. Most of the pieces contain 1 – 3 themes but for some the number is considerably higher. *An Alpine Symphony, Op.* 64 by R. Strauss and *A Children's Overture* by R. Quilter both contain 12 themes – the highest reported number of themes in the dictionary.

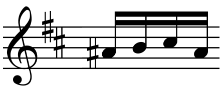
[12] Original: *'eine länger gedehnte, noch dazu langsam ablaufende Melodie,..., sondern wir geben einer solchen Tonfolge den Namen "Thema".'* [183, page 77]

[13] *'Some of the transformations defined here are familiar, such as transposing or inverting an entire motif. Others are less familiar, but common in the music of [Charles] Ives...'* [79, page 76]

Retrograde

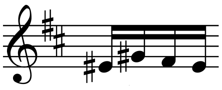
In a *retrograde*, the motif is played backwards, i.e., there occurs a mirroring along the vertical axis. There exist different variations: either only the melody line is reversed while the rhythm remains unchanged, the rhythm is also reversed resulting in a *complete retrograde*, or the melody is reversed and the rhythm is freely altered. A retrograde can also occur as a real or a tonal repetition.

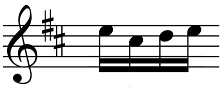

Original motif


Real retrograde

Inversion

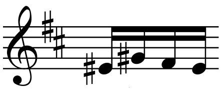
The *inversion* can be imagined as a mirroring along the horizontal axis resulting in the original motif to be played upside-down. Here, the rhythm can remain unchanged or might be altered freely as well.

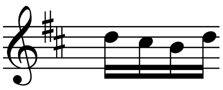

Original motif


Tonal inversion

Retrograde Inversion

The *retrograde inversion* – as already suggested by the name – combines the two previous types of variations resulting in the motif to appear backwards and turned upside-down.

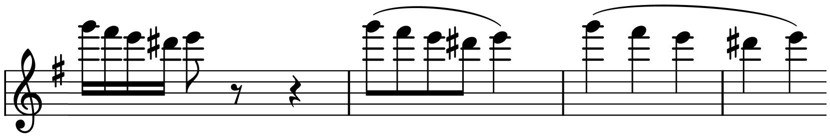

Original motif


Real retrograde inversion

Rhythmic and Melodic Variations

In *rhythmic variations*, the melody remains the same, while the rhythm is altered. Equally, the reverse applies for *melodic variations.* We will now list the different rhythmic and melodic variations we found in the literature.

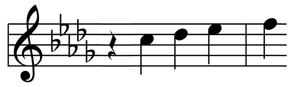
Rhythmic augmentation/diminution of whole motif: The durations of all notes in the motif are doubled/halved.



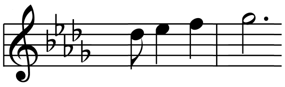
Rhythmic augmentation in *Piano Sonata No.* 27 by L. v. Beethoven (publisher: *Mutopia* [146])

Rhythmic augmentation/diminution of an individual note: The duration of a note in the motif is freely altered.

Free rhythmic variation: The rhythm is freely modified. An example of a *free rhythmic variation* can be found in the *Fugue* 22 from *The Well-Tempered Clavier, Book* 1 by J. S. Bach [82].

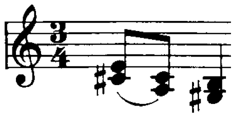


Inversion of the main motif (publisher: *Kern-Scores* [102])

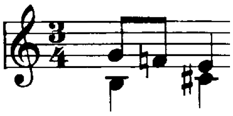


Free rhythmic variation of the motif inversion

Augmentation/diminution of all intervals: All intervals are increased/decreased.[14] A *melodic diminution* can be found in the *Sarabande No.* 1 by J. Brahms.[15]



Original motif (publisher: *Breitkopf & Härtel*)



Interval diminution

Augmentation/diminution of parts of the intervals: The pitch of individual notes is shifted up/down.

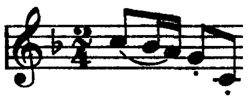
Free melodic variation: The melody (size and direction of intervals) is relatively freely modified, while the rhythm of the original motif is maintained.

Splitting and Shortening

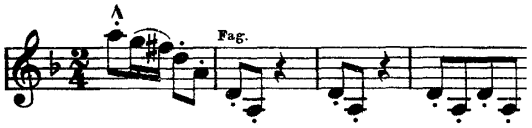
A motif can contain small parts that are themselves subject to restatements and, therefore, submotifs of the original motif. If the motif is subdivided into several submotifs, a *splitting* occurred. In contrast, a motif can also contain only one (shorter) submotif. In this case, musicologists speak of a *shortening*. In the *Symphony No.* 6 by L. v. Beethoven a motif shortening occurs [183].

[14] The diminution can be performed to such an extent that all notes have the same pitch and thus only the rhythm remains (*'Bei der melodischen Verkleinerung kann das Melodische soweit schwinden, daß nur noch das rhythmische Element übrig bleibt'* [114, page 24]).

[15] `http://solomonsmusic.net/Brahmsara.htm`, February 2013



Original motif (publisher: *Breitkopf & Härtel*)

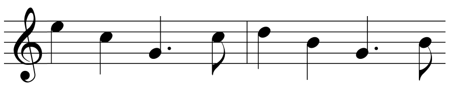


Sequence of the original motif and four repetitions of the last two notes of the motif as shortened submotif

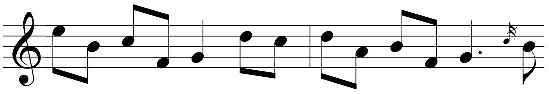
Ornamentation

The basic outline of a motif is extended by trills, turns, mordents, or "free" material. In *ornamentations* the fundamental pitches of the motif have to appear in their original metric position [53].

The first example is an artificial illustration of the concept of ornamentation from [53].

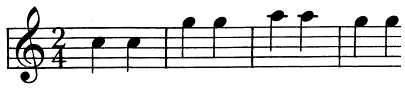


Original motif (source [53])

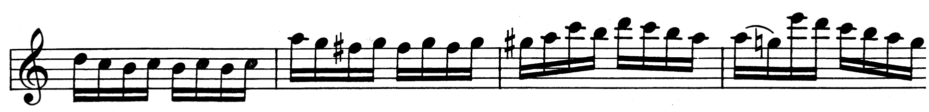


Ornamentation of the original motif

In freer realizations of the concept, the original pitches of the motif are allowed to alter their metric position. The example below shows the original (main) motif and a free ornamentation from the *Twelve Variations on "Ah vous dirai-je, Maman"* by W. A. Mozart.



Original Motif (publisher: *NMA* [197])



Free ornamentation of the original motif

In addition to the mentioned variations, any combination of those might be considered a valid motivic variation. Sometimes the modification of a motif can be carried to such extremes that the original motif is nearly unrecognizable.

6.2 Computer-aided Motivic Analysis

Judging from the previous section, motivic analysis can be considered a highly time-consuming process. Therefore, its automation through appropriate algorithms and tools is a vital field of MIR research. Similar to the three steps in manual motivic analysis, the ongoing research efforts can also be subdivided into different categories. *Pattern detection* algorithms determine all repeating patterns within a given piece of music.[16] In contrast, the goal of *pattern matching* is to determine all occurrences of a given pattern in a piece of music. Here, the types of (motif) variations detected by the matching algorithms can vary strongly. The two tasks employ the term *pattern* rather than *motif* as repetition is a necessity but not a sufficiency for a note sequence to be a motif. In addition, more musically conditions – see also the definition of *motif* in the previous section – such as the pattern being musically interesting, characteristic, or self-existent have to be considered. That is why most pattern detection approaches also contain some *pattern ranking* function. Again, the approaches vary significantly. For example, some only consider pattern length and number of repetitions [41,49], whereas others also include measures such as the melodic and rhythmic diversity of a pattern [97][17] or other musically reasonable features [127]. While rankings can help in eliminating patterns that are obviously irrelevant, they can only provide support in determining the motifs of a piece of music. A musicologist performing a motivic analysis uses a lot of additional knowledge that cannot or only with difficulty be put into a ranking equation. Furthermore, the last step in motivic analysis – determining the function and meaning of a motif in the context of the piece of music – requires musicological experts even more. For example, musicologist might perform cross-comparisons with other pieces from the same composer, the same area, or of the same genre to describe a motif properly. Due to these restrictions, current algorithms can only support musicologists in their work. This is why one usually speaks about *computer-aided* motivic analysis rather than *computer-based*. Furthermore, the restricted automation possibilities strongly call for interactive graphical user interfaces for result presentation and manipulation by human users, see Section 6.4.

Computer-aided motivic analysis (or similar music analysis of, e.g., themes) is usually performed on symbolic score data. However, some work on audio-based pattern detection and pattern matching has been reported [34,62,143,170]. Some of the mentioned publications analyze non-Western music genres which often lack score representations. As our work focuses on symbolic score data, we will not give a detailed account of existing approaches for audio-based motivic analysis and instead concentrate on reporting on pattern detection and pattern matching approaches for symbolic score data.

Approaches to pattern detection can roughly be divided into string-based [41, 43, 46, 90, 97, 110, 111, 113, 126, 127, 166–168, 184] and geometric [50, 128] approaches. String-based methods transform the given piece of music into a string[18] and therein search for repeating substrings. While some approaches only consider pitches and/or a pitch contour [90,110,111,127], other approaches additionally consider note durations in their string representation [41,43,126,166]. Furthermore, a variety of different techniques has been proposed. For example, in [110,111,126,166] similarity matrices are employed to find repeating patterns, whereas Hsu et al. [90], Takasu et al. [184], and Jekovec et al. [97] used trees for the same task. Other than the string-based methods, the geometric approaches are applicable to arbitrary polyphonic pieces; they are capable of detecting non-consecutive patterns extending to several voices and consider pitch and onset time of the individual

[16] It is sometimes also referred to as *pattern discovery*, *pattern extraction*, *pattern identification*, or *pattern mining*.

[17] A melodic pattern with many pitch changes is usually perceived as more diverse than a pattern that only features one pitch. Equally, a lot of rhythmic changes make a pattern rhythmically more interesting/diverse.

[18] Details on how to convert a polyphonic piece of music and score material of polyphonic instruments into strings will be presented in Section 6.3.2.

notes. The mentioned geometric approaches only consider patterns that are repeated unmodified, except for time translation and pitch transposition. In contrast, some string-based approaches also account for some other pattern variation types. For instance, Adiloğlu et al. [1] proposed an approach where repetitions as well as pattern occurrences in inversion are detected. Furthermore, small melodic variations of the original pattern are considered, while rhythm is disregarded altogether. Additionally, they extended their framework and computed substring relationships between musical patterns [2]. In [126,166] similarity matrices are employed to detect exact melodic repetitions of patterns as well as patterns appearing in retrograde and/or inversion. To allow for small interval variations, which are often required in tonal compositions, a generic interval division was chosen, see Section 6.3.2. In contrast to the two previous approaches, Lehmann [113] employed a two-stage approach to find most of the variations introduced in the previous section. First, he detected repeating patterns using the melody contour of the piece, and afterwards he performed pattern matching on different string representations to find the occurrences of the detected patterns in retrograde, inversion, retrograde inversion, augmentation, and diminution. The geometric pattern detection approaches can be extended and thus used for pattern matching as well [128, 129]. In the context of motivic analysis, several other approaches to pattern matching have been proposed [4, 37, 42, 48, 115, 122]. Some of them focus on finding repetitions while others also search for variations of the given pattern.

6.3 String-Based Pattern Detection using Similarity Matrices

Usually, motifs identified by musicologists are of monophonic character. Therefore, in combination with an appropriate resolution of polyphony, string-based approaches to pattern detection are of high relevance for motivic analysis. In the following, we present a string-based pattern detection method intended for computer-aided motivic analysis. Our approach is capable of identifying patterns of all lengths (starting from a minimal length) that repeat at least once within the piece of music. Due to the usage of several modified versions of a similarity matrix, occurrences of a pattern in inversion, retrograde, and retrograde inversion are detected in addition to its repetitions.

6.3.1 Algebraic Formalization

The *Mathematical music theory* as introduced by Mazzola [124,125] is a general theory for a mathematical approach to music and music theory. The theory includes a formal language for the description of musical and musicological objects and their relations. Furthermore, various models of musical phenomena have been developed. Some examples are harmony, rhythm, the theory of music performance, and motivic analysis. The mathematical model for motivic analysis is based on topological spaces of motifs [35, 37]. Here, structural relationships between melodic shapes are determined by a suitably chosen subgroup of the affine group. This so-called *paradigmatic motivic analysis* also defines an inheritance property whereby similarities between musical objects and their sub-objects and thus motif hierarchies are modeled. Before presenting the details of the proposed pattern detection algorithm, we give an algebraic formalization of a pattern/motif and its variations. Our terminology largely follows the concepts of the paradigmatic motivic analysis.

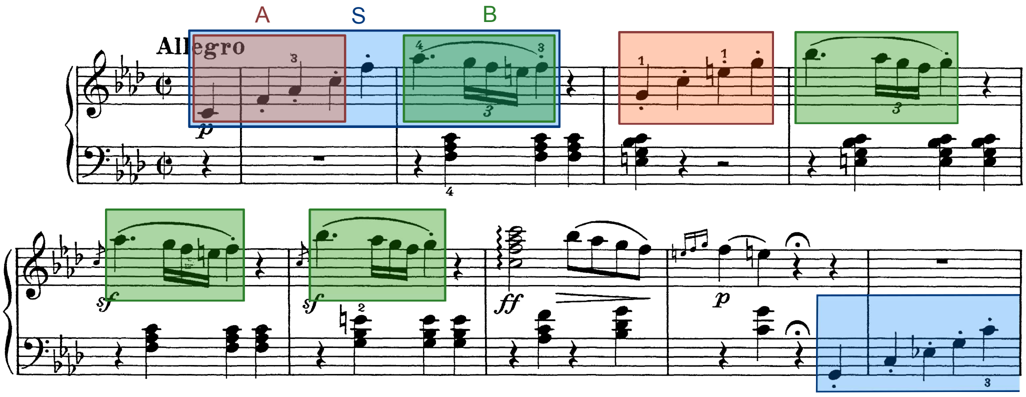


Figure 6.1. Hierarchical motifs in the first movement of *Piano Sonata No. 1* by L. v. Beethoven. Motifs *A* and *B* are submotifs of *S*. As they are also repeated outside the context of *S*, they are independent.

The most important parameters describing a musical note are onset time t, pitch p, and duration d. Using those three parameters, a piece of music M can roughly be represented by a finite set of notes $[t, p, d]$. We are going to study repeated patterns within M. In general, a *pattern* in M is just a non-empty subset M' of M. To specify repetitions, we let the counter point group CP act on the (idealized) universe $N = \mathbb{R} \times \mathbb{R} \times \mathbb{R}_{>0}$ of all notes. This group is generated by all time and pitch shifts together with time and pitch inversions. In this group, every element g is specified by $g = (\epsilon_t, \epsilon_p, \tau, \pi)$ with $\epsilon_t, \epsilon_p \in \{\pm 1\}$ and $\tau, \pi \in \mathbb{R}$. Such a group element g shifts the note $[t, p, d]$ to $g \cdot [t, p, d] := [\epsilon_t \cdot t + \tau, \epsilon_p \cdot p + \pi, d]$. Thus, $\epsilon_t = -1$ induces a time inversion (retrograde), whereas $\epsilon_p = -1$ corresponds to a pitch inversion. By setting both ϵ_p and ϵ_t to -1 retrograde inversions of musical patterns are described. The action of CP on N induces an action of CP on subsets X of N via $g \cdot X := \{g \cdot x \mid x \in X\}$. If P is a subgroup of CP and M' is a pattern in M, then

$$P \cdot M' := \{g \cdot M' \mid g \in P \wedge g \cdot M' \subseteq M\}$$

is the set of all *P-repetitions* of M' in M. The elements in $P \cdot M'$ are called *P-equivalent*. The P-equivalence classes form a partially ordered set: If M_1 and M_2 are two subsets of M, then, by definition, $P \cdot M_1 \leq_P P \cdot M_2$ iff $g \cdot M_1 \subseteq M_2$, for some $g \in P$. In this case, we say that M_1 is a *P-subpattern* of M_2 in M, $M_1 \leq_P M_2$, see Figure 6.1. M_1 is called an *independent* P-subpattern of M_2 iff the number of patterns in M that are P-equivalent to M_1 exceeds the corresponding number w.r.t. M_2. Thus, independency guarantees that the subpattern occurs at least once outside the context of the superpattern.[19]

6.3.2 Pattern Detection Method

String-based approaches to motivic analysis work on *monophonic* note sets M.[20] In such sets, no two notes are active[21] at the same time. Thus, a monophonic note set M can be viewed as a sequence of notes ordered by onset times: $([t_1, p_1, d_1], \ldots, [t_n, p_n, d_n])$ with $t_1 < \ldots < t_n$ and $t_i + d_i \leq t_{i+1}$ for $i < n$. To support the analysis of polyphonic music, the note sequences of the individual instruments are concatenated (i.e., the instruments are

[19] This paragraph originates from our publication [189].
[20] The descriptions in this section largely follow our publication [189].
[21] By definition, a note $[t, p, d]$ is active in the time interval $[t, t + d)$.

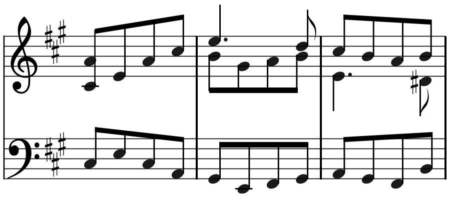


Figure 6.2. Extract from the first movement of Beethoven's *Piano Sonata No. 2*. In the right hand (upper staff) two independent voices are notated.

played consecutively rather than in parallel). In CPN polyphonic instruments, such as the piano, organ, or harp, are usually assigned multiple staves. Thereby, the same rule of concatenating the individual staves/voices can be applied. However, in some pieces the score becomes even more complicated and there are more melodic lines than staves assigned to the instrument, see Figure 6.2. In these cases, most symbolic score formats keep track of those melody lines and the appropriate separation of the note material is possible. For chords or in case the symbolic file does not offer the described functionalities (e.g., MIDI files), melody extraction algorithms can be applied to achieve the required monophony [96, 196]. For this purpose, we implemented the straightforward but rather effective skylining approach to melody extraction. This method assumes that the melody is always located in the highest pitches. Therefore, only the note elements with the highest pitch are kept if overlaps have to be resolved. Of course, this method might fail for orchestral pieces where the main theme is often repeated in the lower voices, like the violas or the double basses, while the higher tuned voices play accompaniment. But as most symbolic score formats at least provide a separation of the individual instrument voices, this will not be an issue.

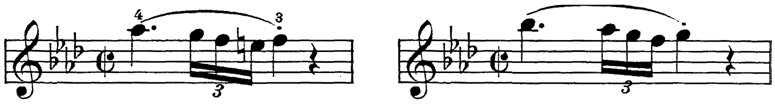
String-based approaches as proposed in this chapter are concerned with consecutive patterns in M, i.e., we consider subsequences

$$M_{ij} := ([t_i, p_i, d_i], [t_{i+1}, p_{i+1}, d_{i+1}], \ldots, [t_j, p_j, d_j]),\ 1 \leq i < j \leq n.$$

In the first step of the algorithm, M is transformed into a string $s_M = s = (s_i)_i$. We use P-invariant transformations, where P denotes the subgroup of CP consisting of all $g = (1, 1, \tau, \pi)$. P-invariant mappings satisfy $s_M = s_{g \cdot M}$, for all $g \in P$. The benefit of P-invariance is that P-equivalent substrings of M are mapped to identical substrings in s_M. In the presented approach, we use two such invariant mappings: the duration string $D_M = D = (d_i)_{i \in [1:n]}$ and the generic pitch interval string $G_M = G = \left(I_g^{-1}(p_{i+1} - p_i)\right)_{i \in [1:n-1]}$.[22] For *generic intervals*, the number of diatonic scale steps between two consecutive notes rather than the number of semitones $(p_{i+1} - p_i)$ is counted. Thus, generic intervals represent sets of semitones

$$\begin{aligned} \ldots, I_g(-1) &:= \{-1, -2\}, \\ I_g(0) &:= \{0\}, \\ I_g(1) &:= \{1, 2\}, \\ I_g(2) &:= \{3, 4\}, \\ I_g(3) &:= \{5, (6)\}, \end{aligned}$$

[22] We would like to explicitly point out that in this chapter D no longer represents the document collection but the string of note durations.



score		
pitch	80 79 77 76 77	82 80 79 77 79
pitch interval	−1 −2 −1 +1	−2 −1 −2 +2
generic pitch interval	−1 −1 −1 +1	−1 −1 −1 +1

Figure 6.3. Example of different string representations for the first two occurrences of motif *B* from Figure 6.1. In pitch and pitch-interval representation, the two motifs are represented by different strings. By contrast, the two are the same for the generic pitch interval representation.

$$\begin{array}{rcl} I_g(4) & := & \{(6), 7\}, \\ I_g(5) & := & \{8, 9\}, \\ I_g(6) & := & \{10, 11\}, \\ I_g(7) & := & \{12\}, \ldots \end{array}$$

A chord of six semitones can be interpreted as augmented fourth or diminished fifth and can therefore appear either in $I_g(3)$ or $I_g(4)$. In our implementation, we instead decided to merge the two and consider five, six, and seven semitones to be tonal similar. Using this notation, $I_g^{-1}(x)$ denotes the index j satisfying $x \in I_g(j)$. By using G as string representation, transposed and translated tonal repetitions of a melody will be represented by the same string as the original melody, see Figure 6.3.

By transforming M into a string s, via $M \mapsto D_M$ or $M \mapsto G_M$, the task of pattern detection is transformed into the task of finding all repeating independent substrings of length $\ell \geq \ell_{\min}$ in the string s, where $\ell_{\min}$ defines the minimal length to be considered. As the results are required for motivic analysis, all occurrence positions of a detected pattern should be reported. Subpattern relationships constitute important information in the context of motivic analysis. Therefore, all subpattern relationships between pairs of independent patterns should be detected as well.

To achieve the stated goals, the ℓ-gram similarity matrix $S_\ell[s,t] \in \{0,1\}^{|s|-\ell+1 \times |t|-\ell+1}$ between two string s and t is defined

$$S_\ell[s,t](i,j) := \begin{cases} 1, \text{ if } (s_i, \ldots, s_{i+\ell-1}) = (t_j, \ldots, t_{j+\ell-1}) \\ 0, \text{ otherwise.} \end{cases}$$

Calculating $S_\ell[s] := S_\ell[s,s]$ yields the ℓ-gram self-similarity matrix of a string s and thus allows for repeating patterns to be detected in s. By definition $S_\ell[s](i,j) = S_\ell[s](j,i)$ so that it is sufficient to compute only half the matrix. In addition to patterns of length ℓ, represented by non-zero entries in S_ℓ, longer patterns can also be deduced from the matrix. An *ℓ-diagonal* of length k is a non-extendable diagonal of k non-zero entries in S_ℓ. Such a diagonal represents a substring of length $k + \ell - 1$ that occurs both in string s and in string t, see Figure 6.4. We can then define the set of all ℓ-diagonals of length k by

$$\begin{aligned} \Delta_\ell^k[s,t] := \{(i,j) \mid \forall (m,n) \in \{(i+\kappa, j+\kappa) \mid \kappa \in [0:k)\} \colon & S_\ell[s,t](m,n) = 1 \\ & \wedge S_\ell[s,t](i-1, j-1) = 0 \\ & \wedge S_\ell[s,t](i+k, j+k) = 0\}. \end{aligned}$$

Furthermore, subpattern relationships are easily deduced from S_ℓ as well. To this end, we

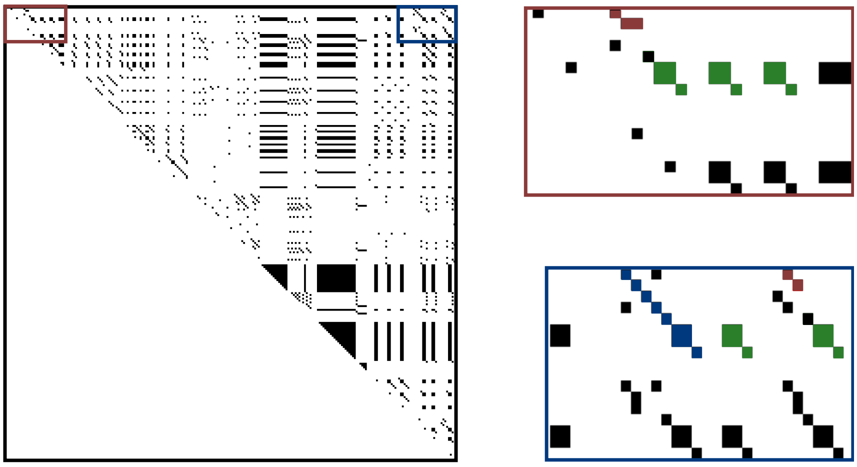


Figure 6.4. Upper half of the 3-gram self-similarity matrix for the generic pitch interval representation of the first movement of the *Piano Sonata No. 1* by L. v. Beethoven (left) and enlargements of the marked regions (right). Matrix entries $S_3(i,j) = 1$ are depicted in black, while zero-entries are white. The colored diagonals in the enlargements coincide with the patterns depicted in Figure 6.1 and their repetitions.

proceed as follows. Let $(i,j) \in \Delta_\ell^k$ be an ℓ-diagonal of length k. The pattern represented by this diagonal is a subpattern if its position interval $[i : i+k+\ell-2]$ in string s constitutes a proper subset of the position interval (in s) of another diagonal in S_ℓ, see Figure 6.4. More precisely, let $(i,j) \in \Delta_\ell^k$ and $(a,b) \in \Delta_\ell^c$ with $k < c$ be the start positions of two diagonals in S_ℓ. Then, the pattern $(s_i, \ldots, s_{i+k+\ell-2})$ is a subpattern of the pattern $(s_a, \ldots, s_{a+c+\ell-2})$ iff $a \leq i \wedge a+c \geq i+k$.

By calculating the self-similarity matrix $S_{\ell_{\min}}[D]$, rhythmic patterns of length $\geq \ell_{\min}$ and all their occurrence positions are determined. Equally, $S_{\ell_{\min}-1}[G]$ detects pitch sequences and their translated and transposed repetitions.[23] To find pattern occurrences in inversion, the string G is compared with its sign inversion $-G = (-G_i)_{i\in[1:n-1]}$ by means of the similarity matrix $S_{\ell_{\min}-1}[G,-G]$. For retrograde inversions $S_{\ell_{\min}-1}[G,G']$ with $G' = (G_{n-1}, G_{n-2}, \ldots, G_2, G_1)$ is used. Finally, retrogrades are detected by combining the two. As the last type, rhythmic pattern retrogrades are considered by computing $S_{\ell_{\min}}[D,D']$.

It is obvious that a pattern might be detected in more than one of these similarity matrices. Equally, an occurrence might match with respect to several variation types. All information on pattern occurrence positions as well as their variation types are stored. Based on this information, the detected patterns are ranked using a ranking strategy that considers pattern length, number of pattern repetitions as well as their respective variation types. Given a pattern p in s with length $|p|$ and its occurrences $o_1, \ldots, o_k$, we calculate the ranking value $r(p)$ of p by

$$r(p) = \frac{|p| \cdot \sum_{i=1}^{k} w(o_i)}{|s|}, \tag{6.1}$$

where $w(o) = w_m \cdot b_m(o) + w_r \cdot b_r(o)$ describes the importance of occurrence o. Here, $b_m(o)$ and $b_r(o)$ represent the binary information for whether o is a melodic, respectively rhythmic

[23] Note that a substring $(s_i)_{i\in[j:k-1]}$ of G corresponds to the note sequence $([t_i, p_i, d_i])_{i\in[j:k]}$.

variation. Moreover, the weights w_m and w_r influence the individual impact of the two types of variations.

6.3.3 Evaluation

Jekovec et al. [97] proposed a method for string-based pattern detection with suffix trees. For the evaluation of their approach, they performed theme detection on the 48 fugues of Bach's *Well-Tempered Clavier*. They searched for patterns of $4-30$ notes' length and considered a theme as detected if a pattern starting with the very first note of the piece of music was among the ten top-ranked patterns. With this setup, their proposed suffix-tree approach correctly detected the themes of 19 fugues. In this section, we present the results of a similar evaluation we conducted for the previously introduced pattern detection approach.[24]

The *Dictionary of Musical Themes* [14] by Barlow and Morgenstern provides a list of themes for most classical pieces of music. The stated 48 themes of the fugues from *The Well-Tempered Clavier* contain between 12 and 31 notes and are exclusively located at the beginning of the respective piece of music. In the first run of our evaluation, we use the same values for the minimal and maximal pattern length as Jekovec et al., i.e., four and 30. However, we also consider a pattern as a valid result if it was not located at the very beginning of the piece of music. Thereby, all sufficiently long subsequences of the sought theme are admissible matches as well. With this setup a valid subpattern of the theme was among the ten highest-ranked patterns for all 48 fugues.[25] For 22 fugues the pattern was even located right at the beginning.

A theme is constructed from one or multiple motifs and thus is usually longer than a regular motif. Scanning the dictionary by Barlow and Morgenstern, we found that a theme usually consists of at least ten notes. However, only six of the successfully detected patterns were of length ≥ 10. The remaining patterns had an average length of 5.6 notes. In a second experiment, we therefore changed the pattern-length range to $10-30$ notes, which seems a more suitable choice for theme detection. Here, 44 of the themes could be identified by one of the ten top-ranked patterns (34 of the patterns were located at the very beginning of the corresponding fugue). In Figure 6.5, an example of a successfully detected theme is shown, and in Figure 6.6 we discuss one of the fugues where the theme detection was unsuccessful.

The results demonstrate that musicological tasks such as theme detection can be supported by automatic approaches as the one presented in Section 6.3 but still require human interaction for verification, modification, and interpretation of the detected patterns. To this end, human experts need intuitive and visually appealing graphical user interfaces. In the following section, we discuss existing work in this field and introduce a new front end for motivic analysis (and theme detection).

[24] Even though we have so far focused on the task of motivic analysis, the proposed pattern detection approach can also be used for the detection of larger musical forms, such as themes or periods.

[25] The themes of *The Well-Tempered Clavier* all feature a very characteristic melody. In contrast, the rhythm of the fugues is often limited to repetitive unvarying note durations. Therefore, we chose $w_m = 1.0, w_r = 0.0$ as the weights for our ranking function, see Equation 6.1.

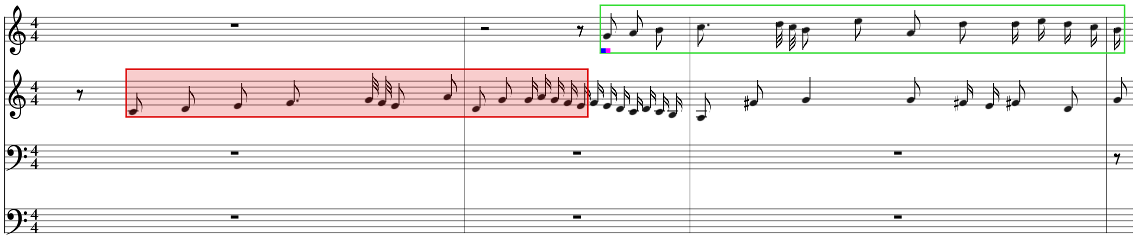
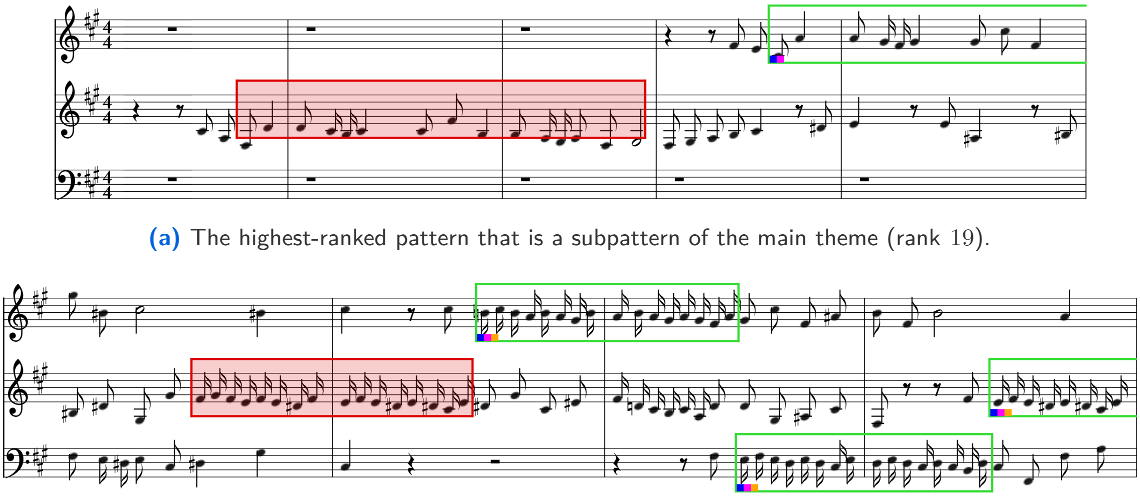


Figure 6.5. The first three measures of the *Fugue No.* 1 from *The Well-Tempered Clavier, Book* 1 by J. S. Bach and the highest-ranked pattern with more than ten notes. According to [14], the theme of this fugue is four notes longer than the depicted pattern. The score was rendered in the MotifViewer. The system currently does not support the visualization of ties and slurs, which is why the tie between the 10th and 11th note of the theme is missing.



(a) The highest-ranked pattern that is a subpattern of the main theme (rank 19).

(b) The highest-ranked pattern of the fugue. The pattern first occurs in measure 36. Except for the missing $G^\sharp$ before the sequence of sixteenth notes, this pattern coincides with the third theme of the fugue [32].

Figure 6.6. Pattern detection results for *Fugue No.* 14 from *The Well-Tempered Clavier, Book* 2. In this example, we searched for patterns containing $10-30$ notes. Bruhn [32] performed a full analysis of all fugues from *The Well-Tempered Clavier*. For this fugue, she identified ten occurrences of the main theme. The best-ranked theme-related pattern detected by our pattern detection method is shown in Figure **(a)**. The number of repetitions and their positions in the piece coincide with those described by Bruhn. In addition to the main theme, the fugue contains two additional themes that appear at a later point in the piece, one of which is depicted in Figure **(b)**. This theme is longer than the main theme and has a higher number of repetitions. Therefore, the employed ranking function assigned this pattern a higher ranking value. In addition, several variations of this theme, e.g., with a shorter length, or starting a few notes earlier or later, were detected as well and most of them received relatively high rankings. Thus, this secondary theme pushed the main theme of the fugue from the list of the ten highest-ranked patterns.

6.4 Graphical User Interfaces

Although a great deal of approaches for computer-aided motivic analysis have been developed, only a few graphical user interfaces have been proposed. However, without appropriate interfaces for accessing, analyzing, and manipulating the automatically detected motif candidates, these algorithms are of little use to musicologists. In [36,38] an interactive visualization is presented. The authors implemented three representations: weight functions, motivic evolution trees, and melodic clustering. All provide information on the detected variations of given patterns. Furthermore, the score of the motif candidates can be accessed. However, no full score visualization where the motifs and their variations are highlighted is available. Collins published a video of a tool using his automatic pattern discovery method [49].[26] With this tool, arbitrary symbolic score data encoded in the Humdrum file format, see Appendix A.1, can be opened for analysis. The detected patterns are ranked and made available for visualization and sonification in a rendered score visualization. To this end, the system employs the music notation software Noteflight.[27] However, the patterns and their occurrences can only be browsed successively and are not made available simultaneously.

Recently, Jekovec et al. [97] proposed a suffix tree-based pattern detection approach. The authors also introduced a graphical user interface for accessing the calculated patterns in Harmonia.[28,29] The presented user interface is already quite comprehensive. The most important functionalities are: visualization of patterns and all occurrences in the score, a ranked list of all detected patterns, access to the suffix tree for the analysis of subpattern relationships, and sonification functionalities for the whole score and individual patterns.

6.4.1 MotifViewer

With the MotifViewer, we propose a Java-based graphical user interface for computer-aided motivic analysis. The system takes symbolic score files in MusicXML, `**kern`, or the MIDI format as input and performs online pattern detection as outlined in Section 6.3. Based on the detection results, the MotifViewer offers an interactive visualization of the patterns, their variations, and their pairwise hierarchical relations. All those patterns, their occurrences, and the whole piece of music are also available for sonification with a MIDI synthesizer.

The MotifViewer interface, see Figure 6.7, is roughly divided into two sections. On the right side, the score material is rendered. The user can choose between a score rendering in classical CPN and a more technical piano roll presentation.[30] The latter might prove useful for users less familiar with CPN as notes are represented by rectangles where the y-position reflects the pitch, the x-position the onset time, and the width of the rectangle is associated with the note duration. In addition, the detected patterns are highlighted in the score. A piece of music usually contains a vast amount of patterns (of different length) and each pattern can in turn reoccur multiple times. To maintain usability, the MotifViewer offers two view modes for patterns of a given length ℓ. In an overview, the beginnings for all patterns of length ℓ are marked in the score, see Figure 6.9. Upon selecting a pattern beginning, the MotifViewer switches to a detailed view showing the full pattern as well as all its occurrences, see Figure 6.11. In this view, the matching types of each occurrence are color-coded. The user can easily switch back and forth between these two views.

[26] `http://www.tomcollinsresearch.net`, February 2013

[27] `http://www.noteflight.com`, February 2013

[28] `http://sourceforge.net/projects/harmoniamusic/`, February 2013

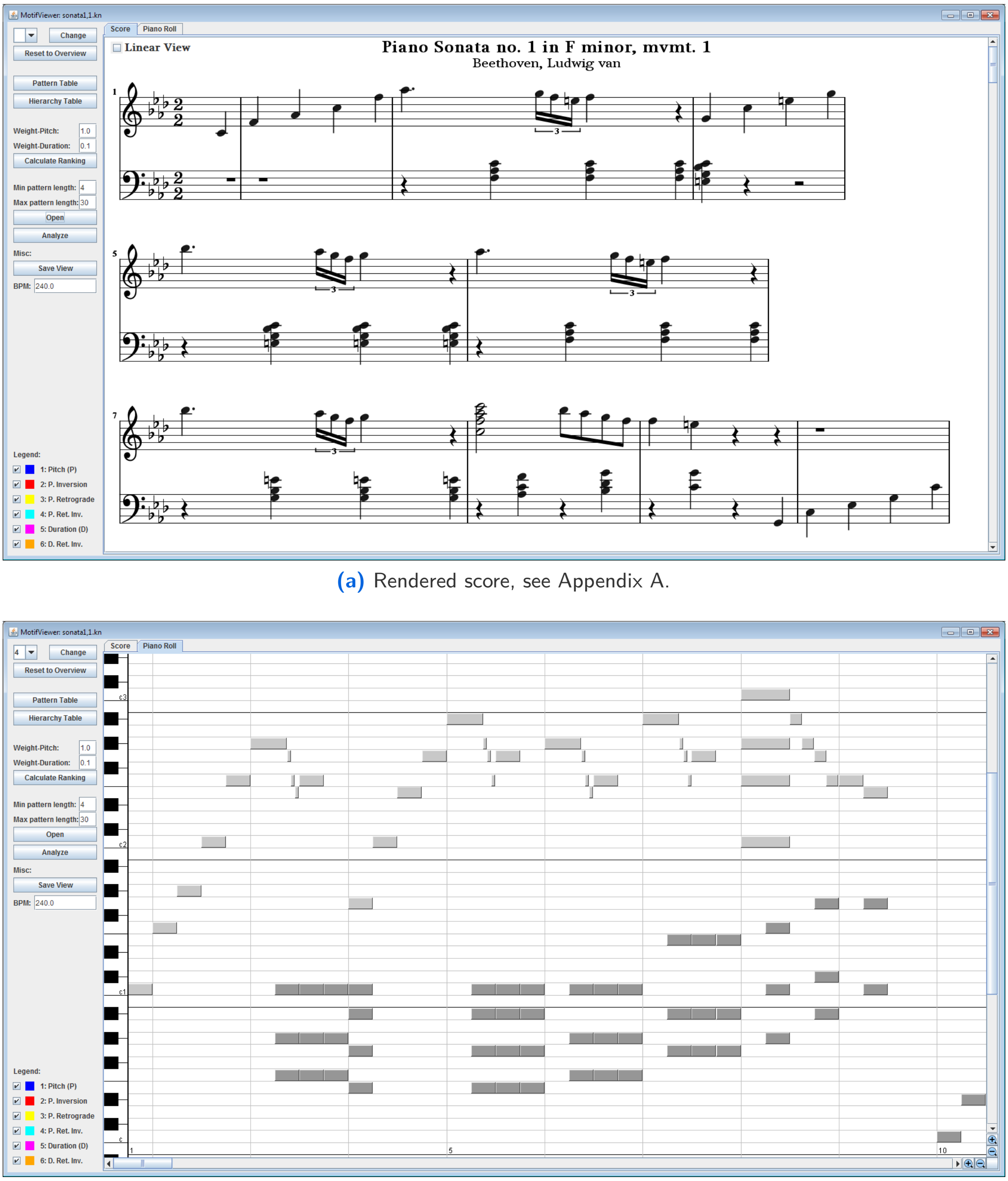
[29] Harmonia is a free music analysis application implemented in Python. By means of the Canorus score editor and LilyPond, MIDI files and MusicXML files can be loaded and rendered as CPN.

[30] CPN is at the moment not available for MIDI files.

On the left side of the MotifViewer front end, a control area is provided, see Figure 6.8. Here, the user can select a file for analysis and manipulate various parameters (e.g., minimal and maximal pattern length and the ranking weights w_m and w_r). Furthermore, several controls allow for an interactive access to the detection result. For example, a different pattern length can be selected to explore the extracted patterns of that length in the score visualization. In addition, the MotifViewer provides a list of all detected patterns (including the calculated pattern ranking) as well as a table describing the pattern hierarchy of the given piece of music, see Figures 6.10 and 6.14. Both are accessed via the control area and can be used to select specific patterns or to explore the exact occurrences of a pattern and a subpattern, see Figure 6.15. Finally, the interface offers the possibility of selecting the types of pattern variations to be visualized. Thus, the user can, e.g., decide to concentrate on melodic and rhythmic repetitions of a pattern while ignoring retrogrades and/or inversions.

A recently added feature of the MotifViewer is the manual selection of note sequence for pattern matching using a modified version of the proposed pattern detection approach, see Figure 6.16. Thus, the user can select a motif by hand and trigger the detection of all its repetitions within the piece of music.

To give an idea of how to perform a motivic analysis with the MotifViewer, Figures 6.7–6.16 provide step-wise examples on how to operate the user interface.



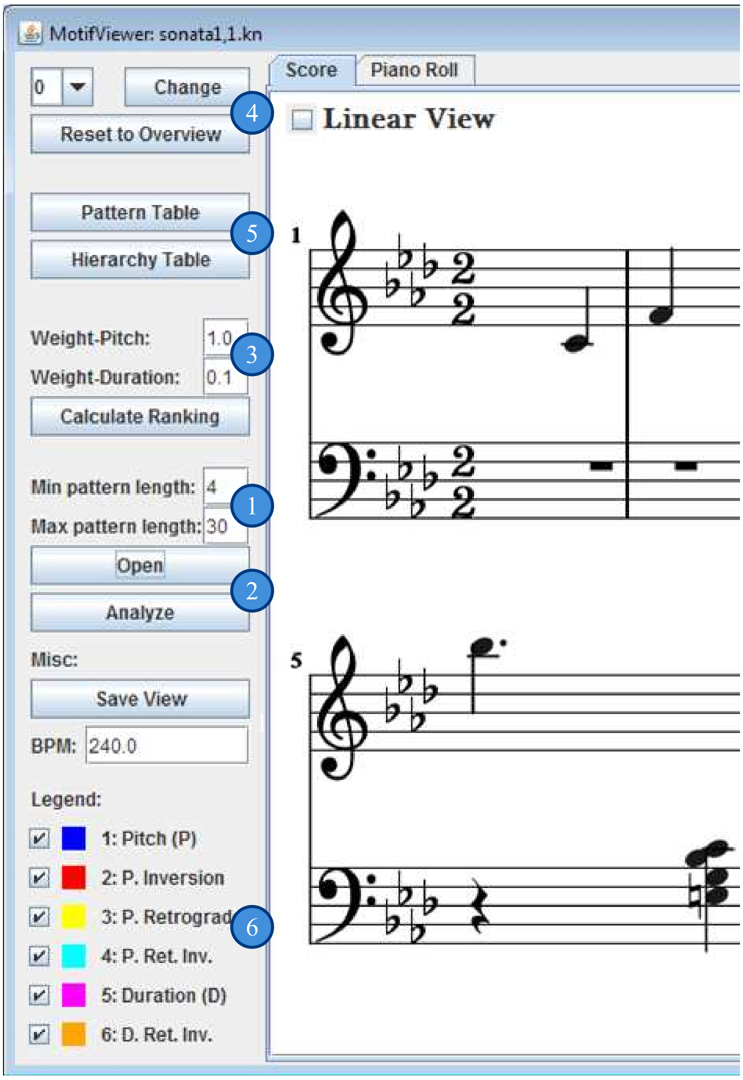
Weight-Pitch: 1.0
Weight-Duration: 0.1
Min pattern length: 4
Max pattern length: 30
BPM: 240.0

- ☑ 🟦 1: Pitch (P)
- ☑ 🟥 2: P. Inversion
- ☑ 🟨 3: P. Retrograde
- ☑ 🟦 4: P. Ret. Inv.
- ☑ 🟪 5: Duration (D)
- ☑ 🟧 6: D. Ret. Inv.

(a) Rendered score, see Appendix A.

(b) Piano roll like visualization of the score.

Figure 6.7. The MotifViewer user interface after loading the `**kern` file of the *Piano Sonata No. 1* by L. v. Beethoven (source [102]).



MotifViewer: sonata1,1.kn

0

Weight-Pitch: 1.0

Weight-Duration: 0.1

Min pattern length: 4

Max pattern length: 30

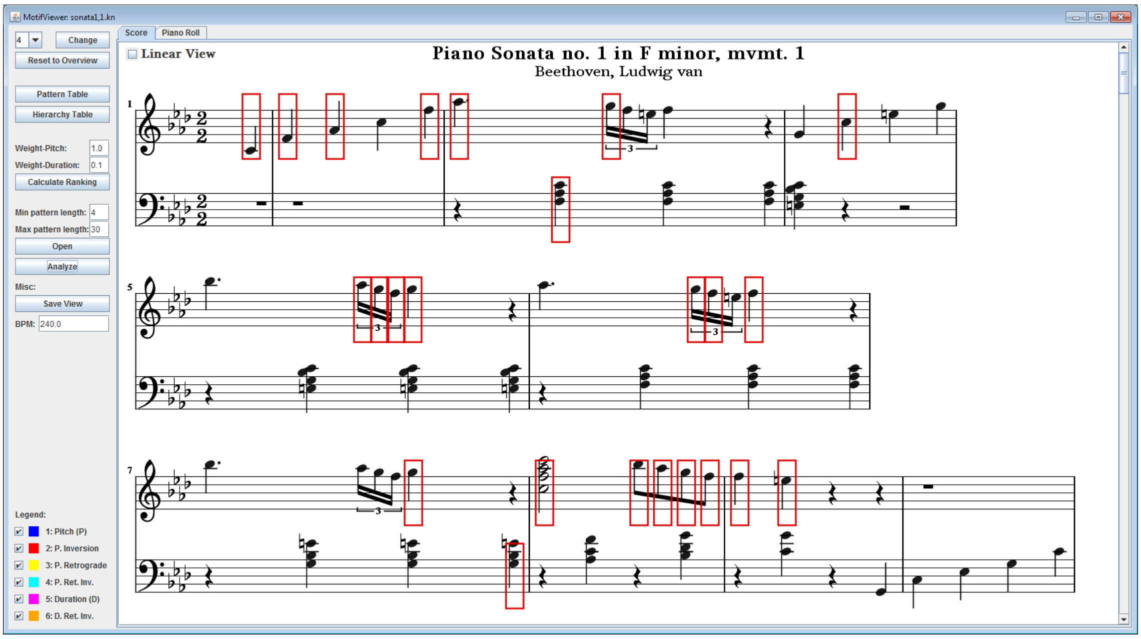
Misc:

BPM: 240.0

Legend:

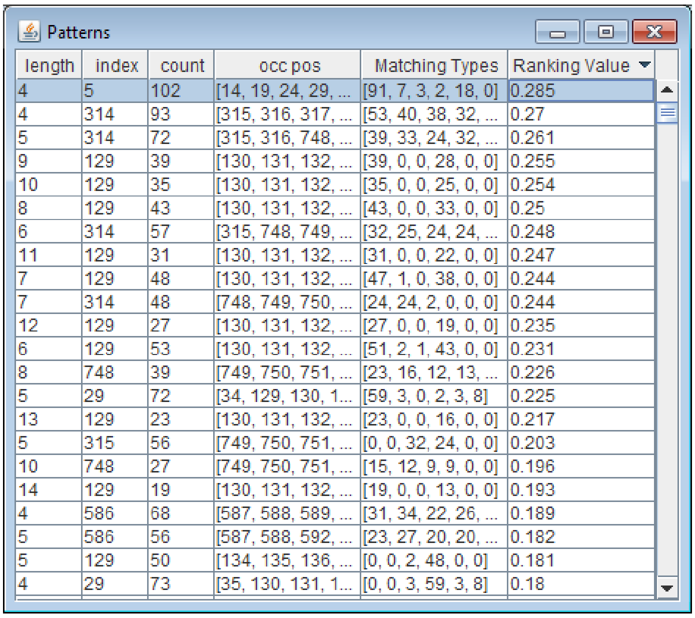
- ☑ 🟦 1: Pitch (P)
- ☑ 🟥 2: P. Inversion
- ☑ 🟨 3: P. Retrograd
- ☑ 🟦 4: P. Ret. Inv.
- ☑ 🟪 5: Duration (D)
- ☑ 🟧 6: D. Ret. Inv.

Figure 6.8. Close-up of the control area on the left of the MotifViewer interface. Here, the user can change various settings and control the pattern visualization. We will now walk through the individual control elements: **(1)** before starting the pattern detection, the user can change the pattern-length range using these textfields; **(2)** buttons for opening a score file in `**kern`, MIDI, or MusicXML format and starting the pattern detection; **(3)** control elements to change the weights of the ranking function, see Equation 6.1, and to recalculate the ranking values of the already detected patterns; **(4)** drop-down list to change the pattern length selected for visualization; **(5)** buttons to open the pattern table depicted in Figure 6.10 and the hierarchy table from Figure 6.14; **(6)** checkboxes for all supported types of motivic variations. Only patterns that have at least one occurrence of one of the selected types are displayed. Furthermore, only occurrences of the selected variation types are visualized in the score.



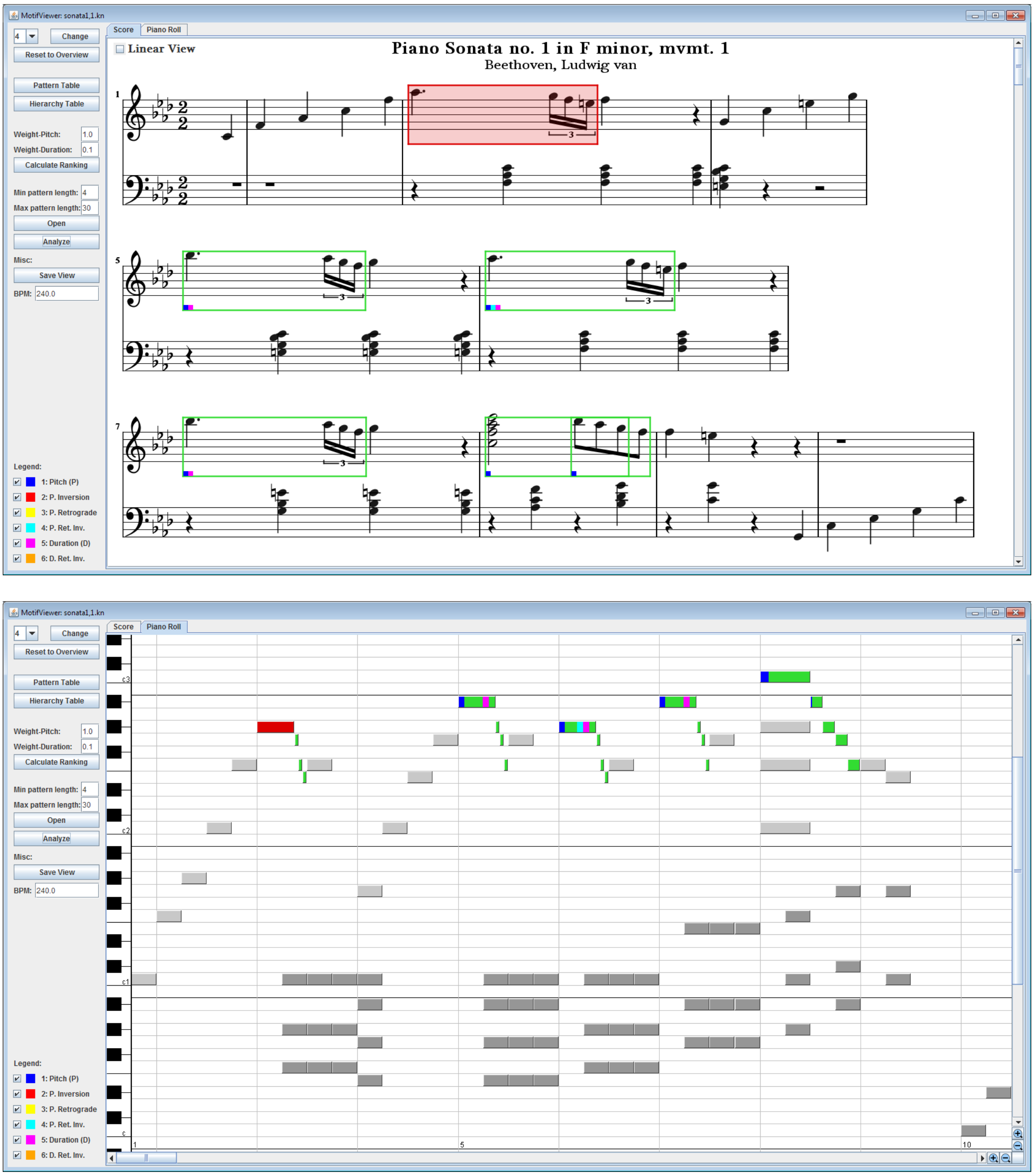
4
Weight-Pitch: 1.0
Weight-Duration: 0.1
Min pattern length: 4
Max pattern length: 30
BPM: 240.0

Figure 6.9. After performing a pattern detection, the minimal available pattern length is used for the visualization. In this example, the minimal pattern length is four. In the score view, the **start positions** of all length-four patterns are indicated by a red rectangle. This view allows the user to get an overview of all patterns of a certain length.



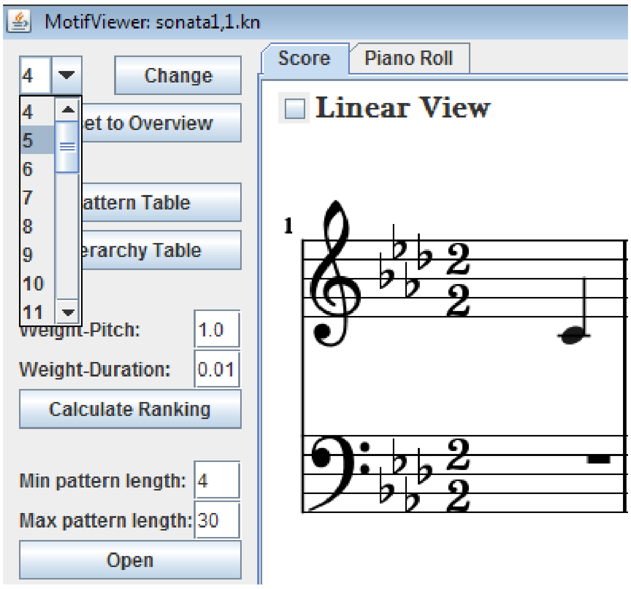
length	index	count	occ pos	Matching Types	Ranking Value
4	5	102	[14, 19, 24, 29, ...	[91, 7, 3, 2, 18, 0]	0.285
4	314	93	[315, 316, 317, ...	[53, 40, 38, 32, ...	0.27
5	314	72	[315, 316, 748, ...	[39, 33, 24, 32, ...	0.261
9	129	39	[130, 131, 132, ...	[39, 0, 0, 28, 0, 0]	0.255
10	129	35	[130, 131, 132, ...	[35, 0, 0, 25, 0, 0]	0.254
8	129	43	[130, 131, 132, ...	[43, 0, 0, 33, 0, 0]	0.25
6	314	57	[315, 748, 749, ...	[32, 25, 24, 24, ...	0.248
11	129	31	[130, 131, 132, ...	[31, 0, 0, 22, 0, 0]	0.247
7	129	48	[130, 131, 132, ...	[47, 1, 0, 38, 0, 0]	0.244
7	314	48	[748, 749, 750, ...	[24, 24, 2, 0, 0, 0]	0.244
12	129	27	[130, 131, 132, ...	[27, 0, 0, 19, 0, 0]	0.235
6	129	53	[130, 131, 132, ...	[51, 2, 1, 43, 0, 0]	0.231
8	748	39	[749, 750, 751, ...	[23, 16, 12, 13, ...	0.226
5	29	72	[34, 129, 130, 1...	[59, 3, 0, 2, 3, 8]	0.225
13	129	23	[130, 131, 132, ...	[23, 0, 0, 16, 0, 0]	0.217
5	315	56	[749, 750, 751, ...	[0, 0, 32, 24, 0, 0]	0.203
10	748	27	[749, 750, 751, ...	[15, 12, 9, 9, 0, 0]	0.196
14	129	19	[130, 131, 132, ...	[19, 0, 0, 13, 0, 0]	0.193
4	586	68	[587, 588, 589, ...	[31, 34, 22, 26, ...	0.189
5	586	56	[587, 588, 592, ...	[23, 27, 20, 20, ...	0.182
5	129	50	[134, 135, 136, ...	[0, 0, 2, 48, 0, 0]	0.181
4	29	73	[35, 130, 131, 1...	[0, 0, 3, 59, 3, 8]	0.18

Figure 6.10. List of all detected patterns. The table can be accessed through the according button in the control area. The individual columns denote the length of a pattern, its start position, the number of detected repetitions, the number of repetitions per motivic variation type, and the ranking value. By double-clicking an entry, the according pattern and all its repetitions are visualized. Here, the list was sorted by the ranking value and the first, highest-ranked pattern was selected for visualization, see Figure 6.11.



4
Weight-Pitch: 1.0
Weight-Duration: 0.1
Min pattern length: 4
Max pattern length: 30
BPM: 240.0

Figure 6.11. Detailed view of the highest-ranked pattern of the *Piano Sonata No. 1* visualized in CPN and as piano roll. The original pattern of length four is highlighted in red. Repetitions are indicated in green. In the score, the lower left corner of each marker color-codes the types of motivic variations that are met by the respective repetition. In the piano roll view, the first note of each repetition contains the information on the variation types.



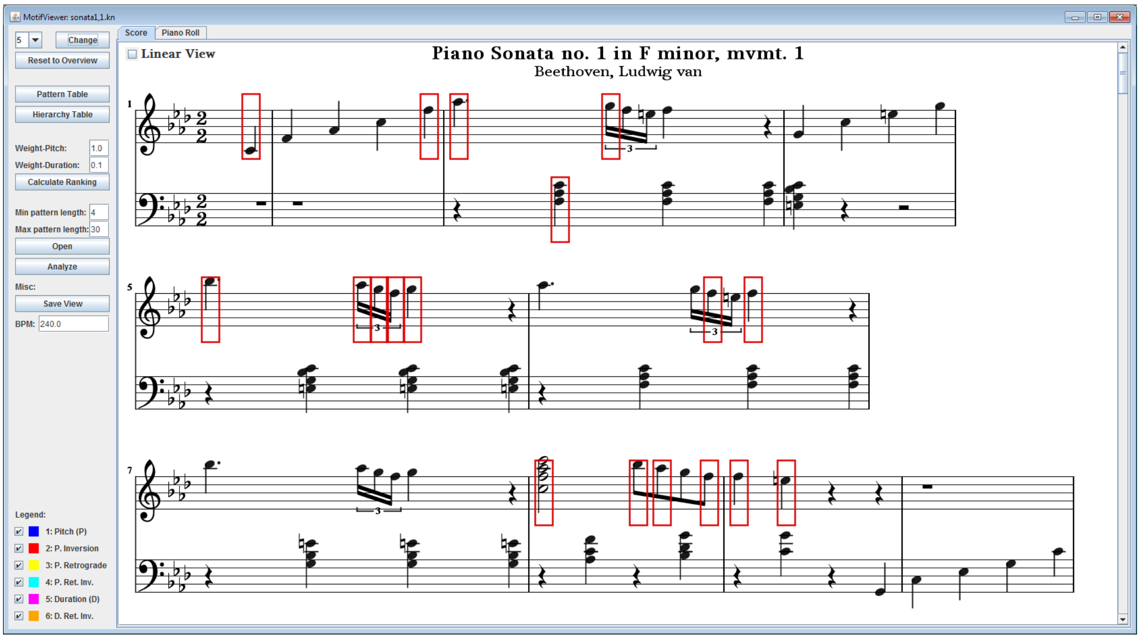
Weight-Pitch: 1.0

Weight-Duration: 0.01

Min pattern length: 4

Max pattern length: 30

Figure 6.12. Selecting a different pattern length (here, length five) for visualization. After verifying the selection with the *Change* button, the overview with the start positions of all patterns of the chosen length is created. For our example, this results in the view depicted in Figure 6.13a.



MotifViewer: sonata1,1.kn

5

Weight-Pitch: 1.0

Weight-Duration: 0.1

Min pattern length: 4

Max pattern length: 30

BPM: 240.0

Legend:

☑ 🟦 1: Pitch (P)

☑ 🟥 2: P. Inversion

☑ 🟨 3: P. Retrograde

☑ 🟦 4: P. Ret. Inv.

☑ 🟪 5: Duration (D)

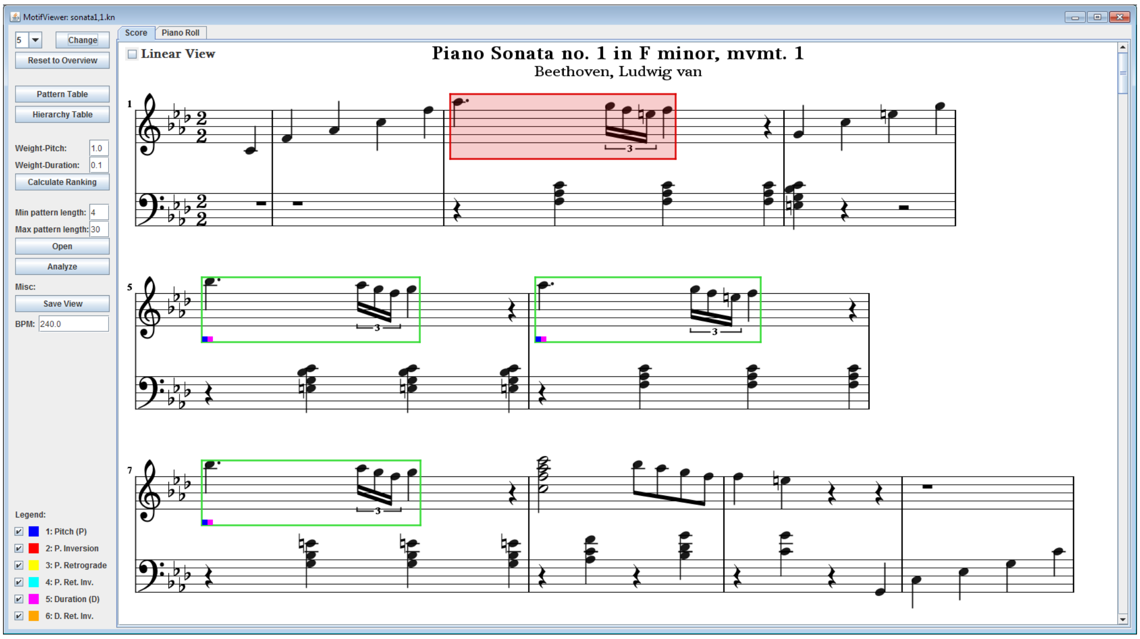
☑ 🟧 6: D. Ret. Inv.

□ Linear View

Piano Sonata no. 1 in F minor, mvmt. 1

Beethoven, Ludwig van

(a) Overview of the start positions of all length-five patterns. By clicking on a start position (in this example, the one indicated by the blue arrow), the detailed visualization of that pattern and its occurrences is created, see Figure **(b)**.



MotifViewer: sonata1,1.kn

5

Weight-Pitch: 1.0

Weight-Duration: 0.1

Min pattern length: 4

Max pattern length: 30

BPM: 240.0

Legend:

☑ 🟦 1: Pitch (P)

☑ 🟥 2: P. Inversion

☑ 🟨 3: P. Retrograde

☑ 🟦 4: P. Ret. Inv.

☑ 🟪 5: Duration (D)

☑ 🟧 6: D. Ret. Inv.

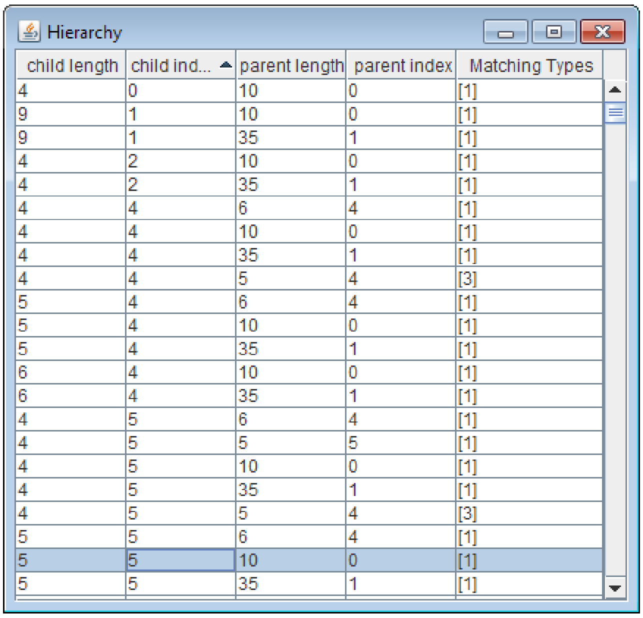
□ Linear View

Piano Sonata no. 1 in F minor, mvmt. 1

Beethoven, Ludwig van

(b) Detailed view of the length-five pattern selected from the pattern overview in Figure **(a)**.

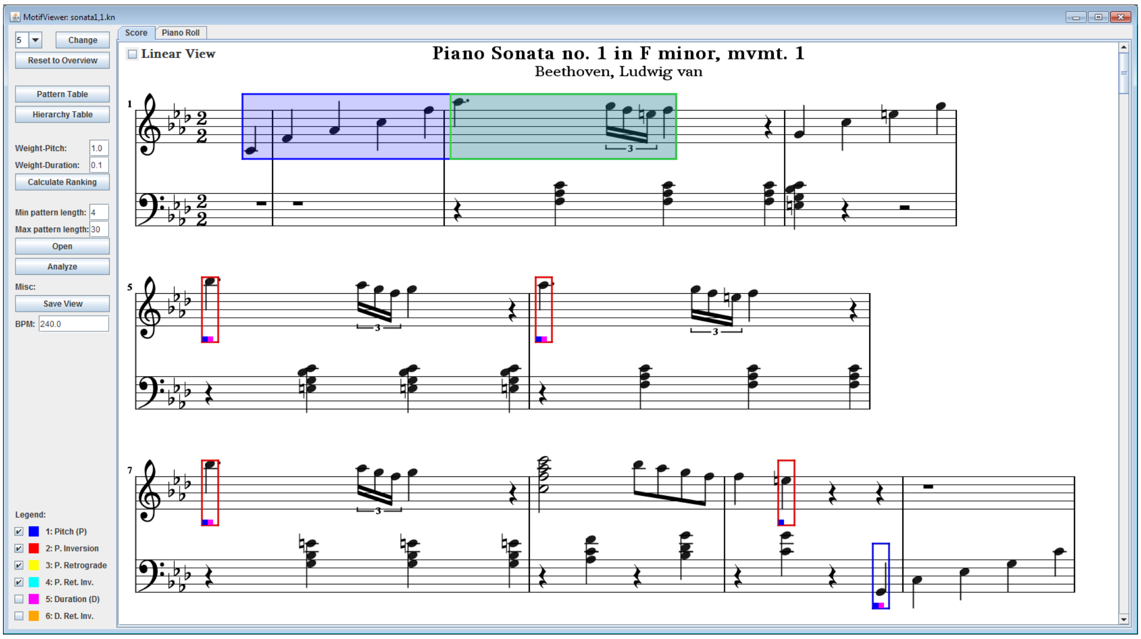
Figure 6.13. Example for manually selecting a pattern for close inspection.



Hierarchy

child length	child ind... ▲	parent length	parent index	Matching Types
4	0	10	0	[1]
9	1	10	0	[1]
9	1	35	1	[1]
4	2	10	0	[1]
4	2	35	1	[1]
4	4	6	4	[1]
4	4	10	0	[1]
4	4	35	1	[1]
4	4	5	4	[3]
5	4	6	4	[1]
5	4	10	0	[1]
5	4	35	1	[1]
6	4	10	0	[1]
6	4	35	1	[1]
4	5	6	4	[1]
4	5	5	5	[1]
4	5	10	0	[1]
4	5	35	1	[1]
4	5	5	4	[3]
5	5	6	4	[1]
5	5	10	0	[1]
5	5	35	1	[1]

Figure 6.14. The pattern hierarchy table can be accessed via the control area. Each row represents a child-parent relation and states the length and start position of both patterns as well as the types of motivic variations that both meet. Upon selecting an entry, the two patterns and their occurrences are visualized, see Figure 6.15.



Weight-Pitch: 1.0

Weight-Duration: 0.1

Min pattern length: 4

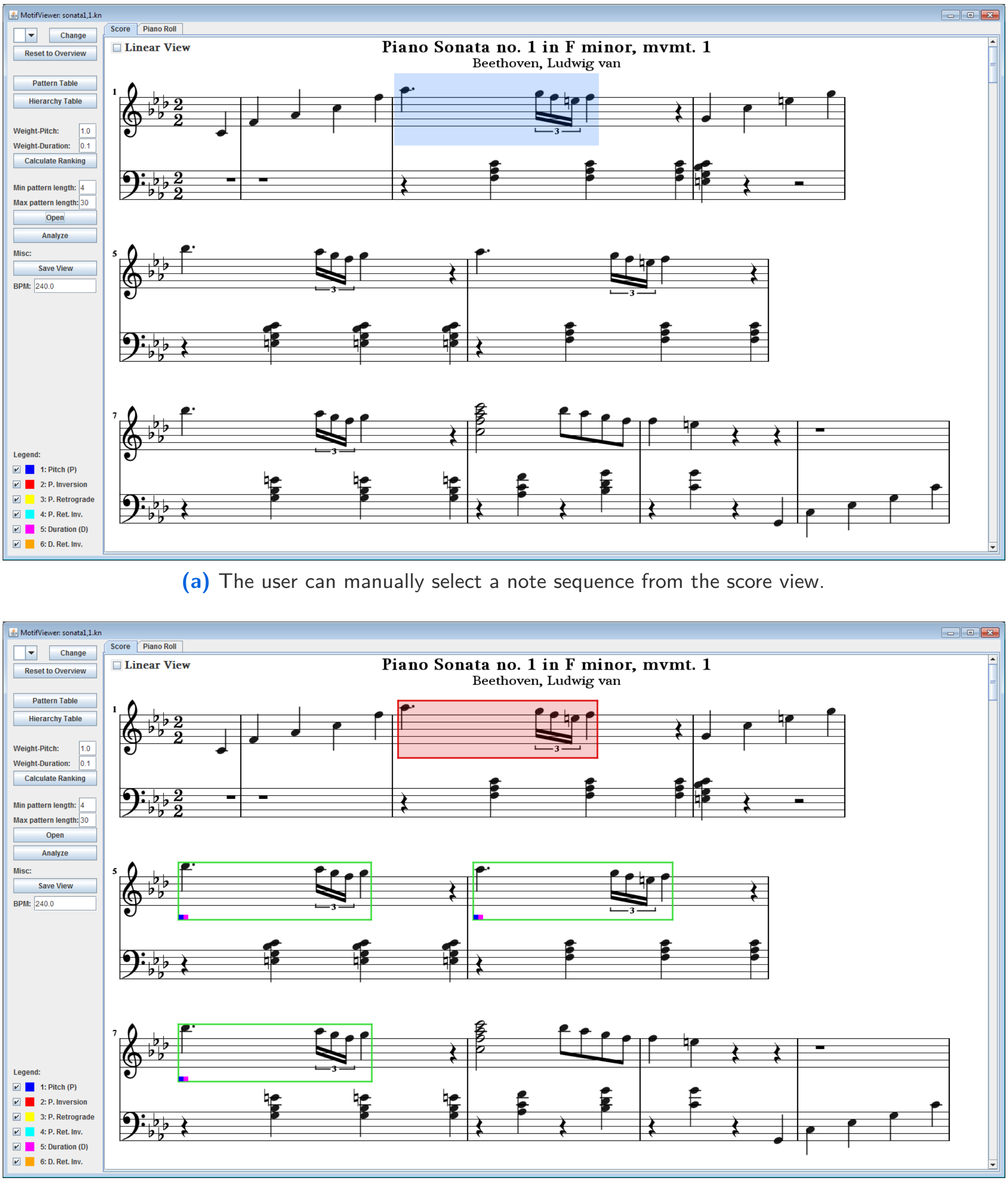
Max pattern length: 30

BPM: 240.0

Legend:

- ☑ 1: Pitch (P)
- ☑ 2: P. Inversion
- ☑ 3: P. Retrograde
- ☑ 4: P. Ret. Inv.
- □ 5: Duration (D)
- □ 6: D. Ret. Inv.

Figure 6.15. Visualization of the subpattern relationship selected through the hierarchy table in Figure 6.14. The length-five pattern starting at position six is a subpattern of the length-10 pattern starting at position one. This hierarchical relationship corresponds to the example in Figure 6.1 on page 103. To keep the view clear, occurrences of the subpattern are visualized by marking only the first note red. Equally, the beginnings of superpattern occurrences are marked in blue.



Weight-Pitch: 1.0
Weight-Duration: 0.1
Min pattern length: 4
Max pattern length: 30
BPM: 240.0

(a) The user can manually select a note sequence from the score view.

(b) Result of the manual pattern matching for the note sequence highlighted in Figure **(a)**.

Figure 6.16. Manual motif selection and pattern matching in the MotifViewer interface.

7 Conclusions

The long-standing efforts of the Multimedia Signal Processing Group at the University of Bonn to create the digital music library system PROBADO MUSIC inspired this thesis. In addition to making substantial contributions to the developed interfaces for document management and access, we addressed newly encountered issues in music synchronization and audio matching and proposed an extension of the PROBADO MUSIC functionality by computer-aided motivic analysis.

Currently, PROBADO MUSIC provides access to scanned score documents and audio recordings. Furthermore, the lyrics available from the score are extracted and presented separately. In the course of this thesis, we demonstrated the extensibility of the library system to support further document types using the example of video recordings. In the future, the inclusion of more music document types would be interesting. Just like the music players proposed in [10, 12], the system could provide access to programs, CD covers, pictures, or texts (e.g., critical reviews or musicological analyses). Furthermore, these documents can be aligned with the audio and score material, for example, to show pictures from the second act of Mozart's *The Magic Flute* while a corresponding recording is playing.

During the preparation of a music collection for its presentation with the PROBADO MUSIC system, one encounters several tasks that currently cannot or only in parts be automatized. While some might only be required in case of unsatisfactory synchronization results, e.g., the correction of jump instructions, others are mandatory. In our experience, the most important and also most time-consuming of them is the segmentation of the music documents and the work identification for the individual segments. In his thesis, Christian Fremerey proposes an innovative procedure that segments and identifies the content of scanned score documents, provided that for each contained piece of music there already exists a correctly assigned audio recording, see [73, 76] and Section 4.2.2. As an extension of this work, it would be great to avoid the required first manual mapping or to support this task as far as possible by using the information available from the music documents themselves. For audio recordings, the back of CD covers – if available – can provide some information on the content of the audio tracks. Similarly, ID3 tags might contain some meta data about title, composer, and artist. In the score, the beginning of a new work is in the majority of cases marked by an indented staff system. Furthermore, some textual information on the piece is provided. Our results for the OCR-based reconstruction of transposition information, see Section 4.3, strongly suggest that OCR can be used to detect and interpret this text. While this might be a straightforward task for human subjects,

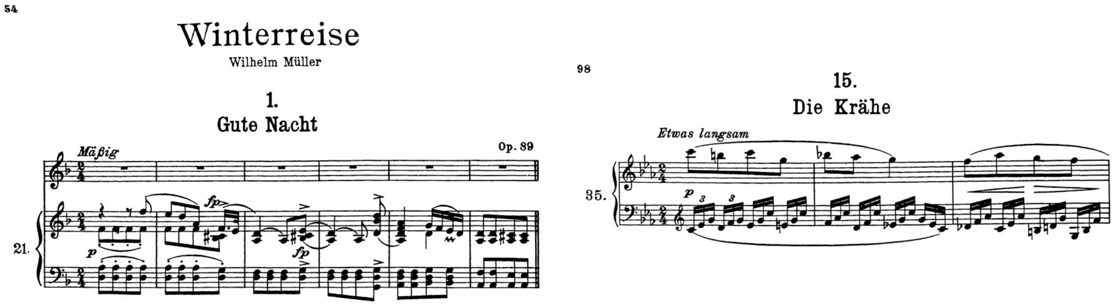


Figure 7.1. Example excerpts of sheet music scans showing the title headings of the first and the 15th song of the song cycle *Winterreise* by F. Schubert (publisher *C. F. Peters*). While for the first song the name of the parent work and an opus number are stated, for the song *Die Krähe* only the child work title is made available. Furthermore, this particular example might result in a misunderstanding. The title heading states the name Wilhelm Müller, who is the author of the used poems and not the composer.

achieving the same computationally is far from trivial. This is mainly due to the lack of unity in the input data. One obstacle is the language. The names of some composers, such as P. I. Tchaikovsky, are spelled differently in different languages. In German, for example, Tchaikovsky is usually spelled Tschaikowski and in Italian it is Čajkovskij. Equally, the titles might differ (e.g., German original title: *Die Zauberflöte*, English: *The Magic Flute*, French: *La Flûte enchantée*, Italian: *Il flauto magico*). Next, we give a score example to explain some further issues to be dealt with when creating a computational approach to segmentation and identification. In Figure 7.1, the beginnings of two songs from the song cycle *Winterreise* by F. Schubert are depicted. For the first song, both the parent work title and the title of the song are available. In contrast, in the second example only the song title is given. Therefore, one has to remember the parent work information. Furthermore, none of the title headings contain the composer's name. The composer, therefore, has to be deduced from the work title or from the cover page of the score book. It is also important to note that different work catalogs can exist. In the given example, the opus identifier "`Op. 89`" is used. However, for music by Schubert a more widespread catalog is the "Deutsch-Verzeichnis" which assigns the identifier "`D 911`" to the *Winterreise*. Despite all these issues – not to mention the question of how to correctly deduce which piece of information is given in a detected string – pursuing this approach would constitute a very interesting future direction, and we feel certain that satisfactory results can be achieved.

After introducing Probado Music, we focused on sheet music-audio synchronization. We have seen that due to the complexity of CPN and the quality of the score scans the OMR results are usually far from perfect. Additionally, all OMR systems known to us lack the capability of detecting and interpreting transposing instruments properly. We have first shown that without this information, the alignment accuracy is significantly reduced and afterwards proposed a strategy for recovering the missing information. As some editors employ compressed notation, we also had to recover the instrumentation of the score. By means of this information, new instrument-based applications could be developed. Instead of always highlighting the current measure for the whole staff system, users could use the instrumentation information and decide to focus on one particular instrument, see Figure 7.2. By further adding methods for score-informed source separation, as proposed by Ewert and Müller [69, 71], the functionality could be further extended. Thus, users could use the system in two settings: First, listen to their own voice and, second, play

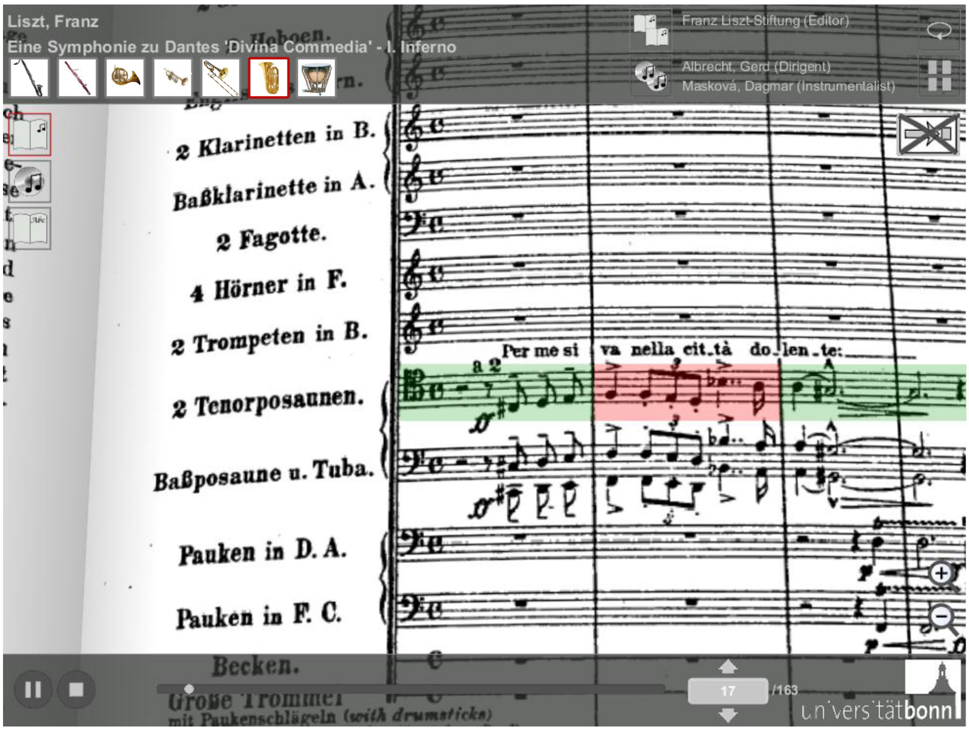

Liszt, Franz
Eine Symphonie zu Dantes 'Divina Commedia' - I. Inferno
Franz Liszt-Stiftung (Editor)
Albrecht, Gerd (Dirigent)
Maskova, Dagmar (Instrumentalist)
2 Klarinetten in B.
Baßklarinette in A.
2 Fagotte.
4 Hörner in F.
2 Trompeten in B.
2 Tenorposaunen.
Baßposaune u. Tuba.
Pauken in D. A.
Pauken in F. C.
Becken.
Große Trommel
mit Paukenschlägeln (with drumsticks)
a 2
Per me si va nella cit.tà do.len.te:
17
/163
universitätbonn

Figure 7.2. Design study on how to select and highlight only individual voices in the score visualization of Probado Music. A first prototype of this functionality has already been implemented and tested.

their own voice accompanied by the recording (where their own voice is muted). To further improve the alignment accuracy in the case of transposing instruments, some currently disregarded scenarios have to be addressed: First, the transposition can change over the course of the piece of music and such a change can also occur in the middle of a system, see Figure 2.10 on page 14. Second, in some editions the instrument labels of some instruments are omitted even in the event of an altered instrumentation, see Figure 7.3. Here, human readers are capable of implicitly identifying the correct instrument-staff mapping by using braces, instrument groupings, and their knowledge from the previous systems.

Next, we looked into the audio matching features of Probado Music. Despite the size of the music collection and the real-life demands, we attempted to replace the previously used diagonal matching approach by the more flexible SSDTW-based audio matching technique. As the higher flexibility came at the cost of significantly longer response times, some kind of speed-up had to be performed. However, a big disadvantage of SSDTW is its incompatibility with common indexing approaches. In our specific application scenario in the context of Probado Music, however, we managed to design and implement an index-based approach that uses SSDTW. Here, we took advantage of the fact that the queries are not arbitrary audio snippets, but extracts from the Probado Music document collection itself (i.e., intra-collection queries). In this scenario, we were able to split the collection into equal-sized overlapping segments and to precompute their respective retrieval results using SSDTW. Storing these matches in appropriate index structures then enabled us to efficiently recombine them at runtime.

To relax the restriction on intra-collection queries, the system could easily be extended by an audio identification step [86, 206]. In such a preprocessing step, a given external audio

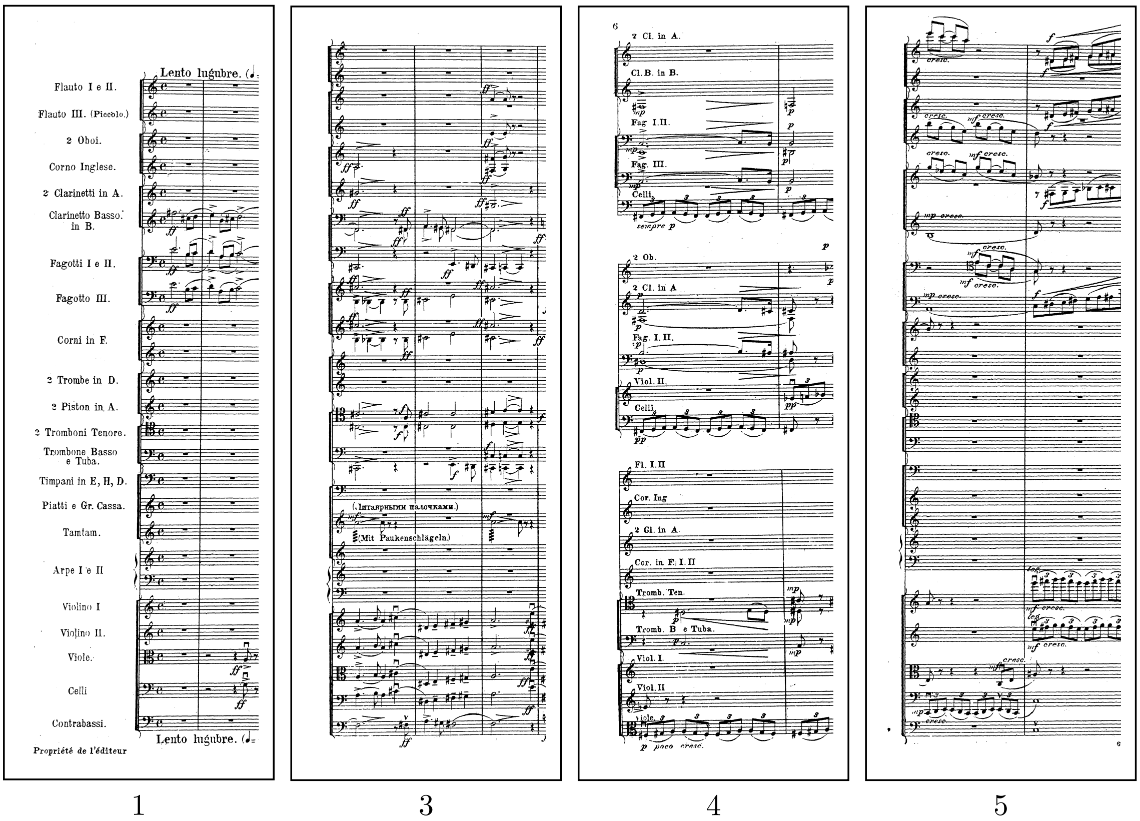


Figure 7.3. The system beginnings of pages $1, 3, 4$, and 5 from the *Manfred Symphony, Op.* 58 by P. I. Tchaikovsky (publisher: *P. Jurgenson*). In terms of instrumentation, page 2 (not depicted) is equal to pages 1 and 3. The example shows a compression of the staff system on page 4. On page 5 the instrumentation returns to the original instrumentation from page 1 without providing any textual labels. Here, the reader has to deduce the instrumentation from prior knowledge.

snippet could be tested for membership with the collection. If this is the case, the according audio snippet from the collection could be used instead (to benefit from intra-collection search). Otherwise, the system could fall back on diagonal matching (accepting inferior results) or classical SSDTW-based audio matching (accepting long response times). The presented experiments in combination with the audio identification performances reported in the literature suggest that this approach would still yield response times that are orders of magnitude better than those of classical SSDTW-based audio matching (for queries that are part of the collection).

Through personal correspondence, we recently learned of new efforts to improve the runtime of DTW using LBFs [160]. Other than previous approaches the authors propose to use multiple LBFs in combination with an early abandoning technique. Thereby, the number of match candidates is successively and very quickly reduced. In a preprint, the authors also suggest the applicability of their techniques to matching tasks [159]. Other than SSDTW, their approach performs DTW between the query and segments from the database that have the same size as the query. To handle global distortions that influence the total length of the sequence, a uniform scaling approach is included. Thus, instead of one query, multiple linearly stretched versions of the query are retrieved. While evaluations demonstrate the efficiency of their approach, the applicability to music documents – in terms of result quality – remains to be assessed.

In the last part of the thesis, we investigated symbolic intra-document pattern detection for application in computer-aided motivic analysis. We suggested a novel string-based approach that considers several different types of common motivic variations and is above that capable of detecting hierarchical relations between patterns of different length. An essential next step is the detection of the remaining common motivic variations, such as rhythmic and melodic augmentation/diminution. We suggest using an appropriate ranking strategy to first select the most-promising motif candidates. Subsequently, pattern matching techniques tailored to the specific features of the so far unacknowledged motivic variations could be applied to these candidates.

Our long-term goal is the inclusion of the developed pattern detection techniques and the corresponding front end into the Probado Music system. In contrast to the currently used `**kern` and MusicXML files, the symbolic information created via OMR can contain errors. Thus, the employed pattern detection approach should be extended to allow for some fuzziness in the input data.

A Loading and Rendering Symbolic Score in the MotifViewer

The MotifViewer interface introduced in Section 6.4.1 is capable of importing several different symbolic score formats, which are then rendered in CPN. In this chapter, we briefly introduce the supported file formats, namely the Humdrum `**kern` format and MusicXML. Afterwards, some details on the score rendering process are presented.

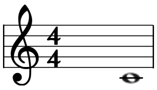
A.1 Humdrum Files

David Huron developed the software system Humdrum with the intention of assisting music researchers. Humdrum consists of two parts: the Humdrum Toolkit and the Humdrum Syntax. The toolkit provides a set of software tools that help with posing and answering questions about music. Given this toolkit for computer-aided music research, the Humdrum Syntax provides a sequential grammar to store and represent all sorts of music-related information. Examples are score material, analysis results, piano fingerings, dance steps, and concert programs. There exist approximately 20 pre-defined representation schemes in the Humdrum Syntax. The most commonly known and used representation is the `**kern` format, which can represent a variety of music notations, among others, CPN, medieval square notation, and Indian tabla notation. In the remainder of this section, we focus on introducing some details on the `**kern` representation of scores in CPN. For more details on the Humdrum Toolkit and the Humdrum Syntax, we refer to the Humdrum user guide [92].

CPN can be viewed as a table where the rows contain the score information for the individual instruments – or rather voices – and the columns represent successive moments in time. The `**kern` format employs the same layout, only flipped 90 degrees. A very simple `**kern` example representing a single whole note of pitch C4 is depicted in Figure A.1. Individual notes are represented by strings consisting of an integer, a (possibly repeated) character and – if required – an accidental sign (`"#"` for sharps and `"-"` for flats). The integer encodes the duration of the note with `"1"` for a whole note, `"2"` for a half note, `"4"` for a quarter note, etc. The pitch is encoded by chroma and tone height. The character represents the chroma class, whereas its frequency determines the octave (or tone height). The middle C (C4) is represented using the lower-case letter `"c"` while an upper-case `"C"` designates C3. Starting from those, successive higher and lower octaves are then designated by letter repetition, for example, `"eee"` for E6 and `"BB"` for B2. In case of a rest, the character `"r"`

```
**kern
*clefG2
*k[]
*M4/4
=-
1c/
==
*-
```

(a) **kern code.



(b) Score in CPN.

Figure A.1. **kern example by which one measure in $4/4$ time with a whole note on middle C (C4) is represented. The score was rendered with the MotifViewer.

is employed instead. As each column represents one voice, the onset time of a note is indirectly encoded via its position in the data stream. In polyphonic pieces of music, a place holder represented by a single period character (".") helps with correctly aligning the time events. In combination with a note or a rest, the period character, however, indicates a dotted duration. At this point, we would like to remark that with the **kern format the representation of syntactic information of a score is addressed. Thus, for example, visual and orthographic information, which might be specific to a certain rendition of the score, is not considered. One result is a duplicate statement of accidentals in case of key signatures. More precisely, a note entry always explicitly describes the exact pitch of the note, regardless of a possible key signature. In addition to the note elements, basic information such as clef ("*clefG2"), time signature ("*M3/4"), key signature ("*k[b-]"), measure boundaries ("="), and repetitions ("||:", ":||", and ":||:") can be described with the **kern syntax. Furthermore, stem directions, beaming information, slurs, phrases, embellishments, and grace notes have encodings as well. In Figure A.2 a more complex example of a score for two voices and its visualization in CPN is depicted. For a more detailed description of the **kern encoding of CPN, we refer to Chapters 2 and 6 of the Humdrum user guide [92].

A.2 MusicXML

MusicXML is an open XML-based file format for the representation of score in CPN.[1] The goal was the creation of a common standard for the interchange of scores, in particular between different score notation programs, and today the majority of notation programs (e.g., Finale, Sibelius, Cubase, SharpEye, capella-scan, LilyPond) support the import and export of score material in MusicXML.

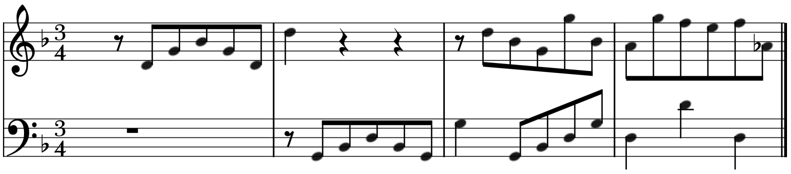
Initially, MusicXML was an XML updating of the MuseData format [45] where some key concepts of the Humdrum Syntax have been added. From there, it was significantly extended to also support contemporary popular music and to turn it into a distribution and interchange format.

In contrast to **kern, the layout does not directly reflect the structure of a score. Instead, XML elements and their nesting reflect the structure. This is why most score elements like

[1] http://www.makemusic.com/, February 2013

```
**kern    **kern
*staff2   *staff1
*clefF4   *clefG2
*k[b-]    *k[b-]
*M3/4     *M3/4
=1-       =1-
2.r       8r
.         8d/L
.         8g/
.         8b-/
.         8g/
.         8d/J
=2        =2
8r        4dd\
8GG/L     .
8BB-/     4r
8D/       .
8BB-/     4r
8GG/J     .
=3        =3
4G\       8r
.         8dd\L
8GG/L     8b-\
8BB-/     8g\
8D/       8gg\
8G/J      8b-\J
=4        =4
4D\       8a\L
.         8gg\
4d\       8ff\
.         8ee\
4D\       8ff\
.         8a-\J
=5        =5
*-        *-
```

(a) **kern code.



(b) Rendering result of the MotifViewer.

Figure A.2. Example of a two-staff score encoding in the **kern syntax. Other than in CPN, the score is laid out vertically and each column represents an individual staff/voice. In this example, stem directions ("/" and "\") and beamings ("L" and "J") are explicitly indicated.

parts, measures, notes and even most of the attributes (e.g., key signature, time signature, clef, pitch, duration) are represented as elements and XML attributes are scarcely used. There exist two root document types which differ in their approach towards structuring the score material. MusicXML files that conform with the `<score-wise>` type contain parts which each contain measures, whereas `<time-wise>` MusicXML descriptions contain measures, which are made up of parts. By courtesy, MusicXML provides two XSLT stylesheets to convert back and forth between those two document types.

We refrain from discussing the MusicXML format in detail and instead only point out some important aspects. In Figure A.3 the MusicXML description of the one-note example shown in Figure A.1b is provided. The example illustrates the structuring of the score into parts that contain measures which in turn contain notes. Just like the `**kern` format, a pitch representation through chroma class (`<step>`) and tone height (`<octave>`) was chosen. Furthermore, note durations are represented by integer values and optionally by a string `<type>` description as well. Onset times are deduced implicitly by the position in the description. In contrast to `**kern`, MusicXML can contain information on the score layout, such as the size of the score pages and of the music symbols and their position. For further details, we refer the interested reader to the MusicXML tutorial [165].

A.3 Score Rendering

Music notation – also referred to as *music engraving* – is the art of drawing music with the purpose of mechanical reproduction. At the end of the 20th century the traditional *plate engraving*[2] was successively replaced by computer software designed for the same purpose. Music engraving software is also known as *scorewriter* or *music notation software*. While the techniques for creating engravings and reproducing music notation changed, the challenges in attempting the creation of an appealing score representation remain the same. Computer programs that provide a visualization of music score (*score rendering*) have to face these issues as well even though their usage scenario is different. The `**kern` format does not supply any layout information for score rendering. Equally, layout information in MusicXML is only optional and thus often not available in the files. Thus, to provide a visualization of a piece of music in the MotifViewer, a score layout has to be derived from the given information. In the following, we will briefly discuss the employed score rendering method. In the process, we point out some particular challenges and discuss the chosen approaches for solving them.

The MotifViewer user interface allows a flexible adaptation of the font size and the size of the score rendering area. Therefore, the zoom factor as well as the current width of the score have to be considered throughout the layout process. For the majority of music symbols freely available SVG files from Wikimedia Commons[3] are used, see Figure A.4. In addition, lines for the staves, measure boundaries, stems, and beams as well as the numbers of the time signature and tuplets are rendered directly in Java. For a given score, its layout is determined measure after measure. We will therefore first describe how the symbols

[2] In plate engraving music was reproduced onto a zinc or pewter plate in a mirror image. For fixed symbols, like clefs and note heads, dies were prepared while variable symbols, such as beams and slurs, were engraved by hand [205, Music Engraving].

[3] *Category: SVG musical notation*, `http://commons.wikimedia.org/wiki/Category:SVG_musical_notation`, February 2013.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE score-partwise PUBLIC
  "-//Recordare//DTD MusicXML 3.0 Partwise//EN"
  "http://www.musicxml.org/dtds/partwise.dtd">
<score-partwise version="3.0">
  <part-list>
    <score-part id="P1">
      <part-name>Music</part-name>
        </score-part>
  </part-list>
  <part id="P1">
    <measure number="1">
      <attributes>
        <divisions>1</divisions>
          <key>
            <fifths>0</fifths>
          </key>
          <time>
            <beats>4</beats>
            <beat-type>4</beat-type>
          </time>
          <clef>
            <sign>G</sign>
            <line>2</line>
          </clef>
      </attributes>
      <note>
        <pitch>
          <step>C</step>
          <octave>4</octave>
        </pitch>
        <duration>4</duration>
        <type>whole</type>
      </note>
    </measure>
  </part>
</score-partwise>
```

Figure A.3. MusicXML code for the one-note example depicted in Figure A.1b.

in a measure are properly placed and subsequently talk about the proper positioning of measures. A measure has assigned to it a clef, a time signature, and if required a key signature. This information is usually placed at the beginning of a measure. However, it is only visualized for the first measure in a line or if one of the three was changed compared to the previous measure. In the latter event, only the modified property is shown. Next, the note material is placed in the measure. It is important to note that the horizontal position of a note in the measure reflects its onset time. Therefore, we defined a variable `quarter_note_spacing` whereby the distance between two quarter notes is determined. Using the onset time of a note (in quarters) and this variable, the position of the note in the measure can be determined. It is obvious, that the quarter-note spacing has to be large, if a lot of short note durations exist. In contrast, long values allow for a smaller spacing. Therefore, the individual quarter-note spacing of each measure is calculated as a

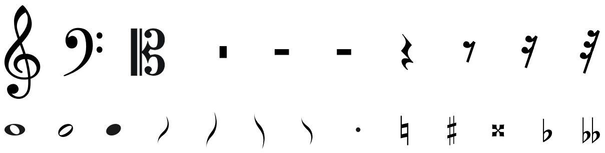


Figure A.4. Table of all music symbol images used by the MOTIFVIEWER (source Wikimedia Commons). Lines such as the staff lines, measure lines, beams, and stems, and text (time signatures and tuplet identifiers) are rendered directly in Java.

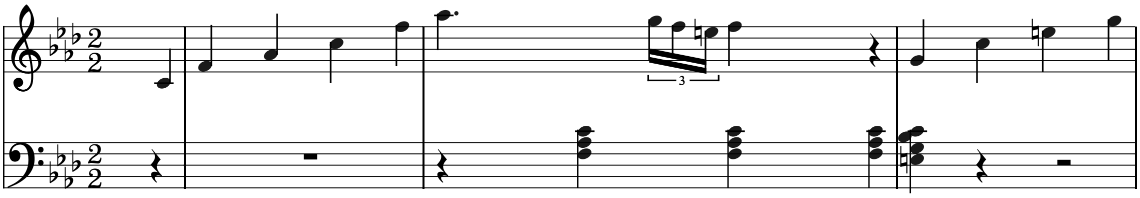


Figure A.5. Example of adjusted quarter-note spacings for different note durations. Measures two and four are distinctly shorter than measure three. In addition, in measure three all notes starting from the E4 in the first staff are slightly shifted to accommodate the natural symbol.

function of the shortest note value in the measure, see Figure A.5. With this approach, accidentals and clefs in the middle of a measure might end up overlapping with the note objects. We consider this case by always checking for overlaps if an accidental or a clef has to be drawn. In the event of an overlap, all subsequent symbols in the measure are appropriately shifted to the right. For polyphonic pieces, a measure spans multiple staves, which have to obey a common horizontal spacing. Therefore, the previously described approach for the placement of the score symbols has to consider all staves simultaneously.

On the note level, several music engraving rules should be observed to yield an appealing visualization: First, stem directions need to be determined if they were not specified by the symbolic score file. It is a common convention that all notes on or above the middle line of a staff are stemmed down, whereas notes below the middle are stemmed up [144]. Furthermore, the stem length has to be checked. Usually, a stem spans one octave. For chords, the length is calculated from the note that is closest to the end of the stem. If a note is placed on ledger lines, the stem has to be extended to touch the middle line of the staff. Another exception to be considered are chords containing an interval of a second. Regularly placed, the two notes concerned would overlap. The *Standard Music Notation Practice* of the Music Publishers Association of the United States [144] states that these cases are to be resolved by placing the lower note on the left and the upper note on the right. Remaining notes are then placed according to the stem: For upstem the lower note is placed in correct relation, whereas for downstems the upper note has the correct placing, see Figure A.6.

Concluding, we want to briefly comment on the placing of the measures. After determining the appropriate positions for all symbols contained in a measure, its width can be deduced as well. If adding the measure to the previous measure in the score would exceed the score width, the current measure is moved into a new line. In this case, the measure needs to be extended to also show its time signature, key signature, and clef. Afterwards, all note

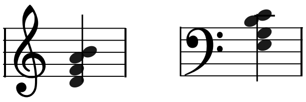


Figure A.6. Example of chord spacings in the MotifViewer. If the chord contains a second interval, the lower note is placed on the left and the upper note in the interval is placed on the right. Depending on the stem direction, further notes are placed in correct relation to the stem.

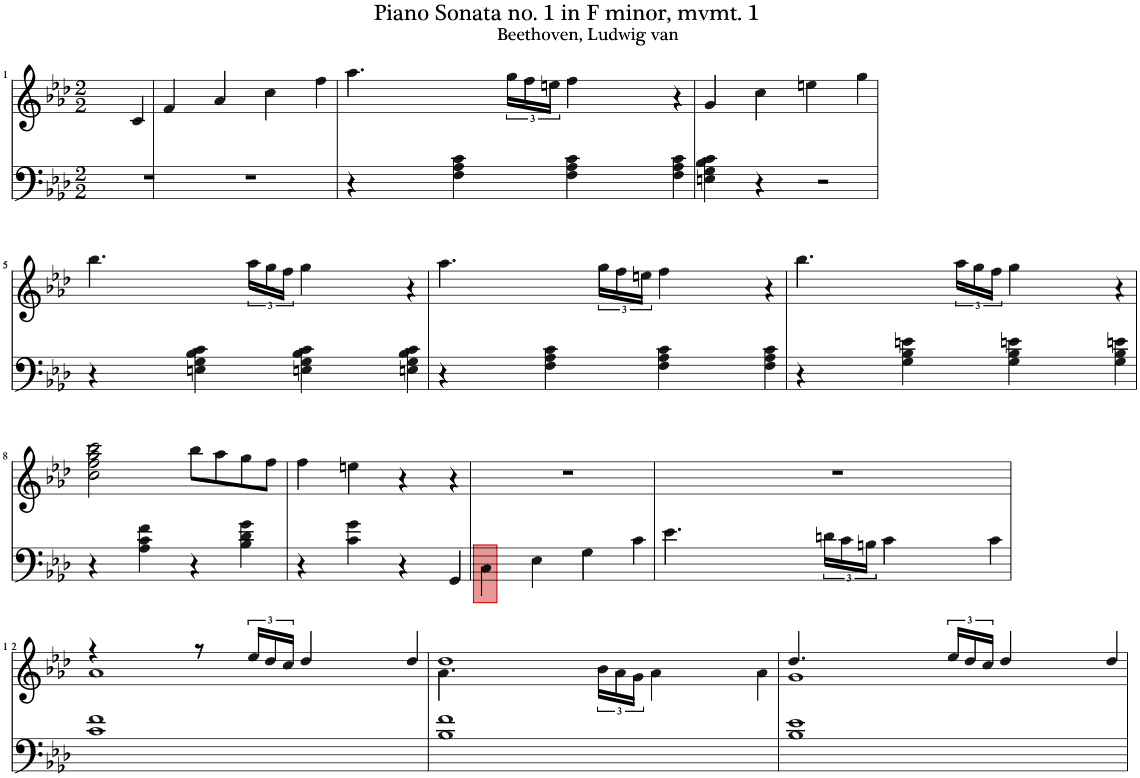


Figure A.7. Score rendering result in the MotifViewer showing the first 14 measures of Beethoven's *Piano Sonata No. 1*. Note that according to the stated score rendering conventions, in measure nine the C3 in the second voice should have an upstem. However, the used **kern representation explicitly notated a downstem for that note.

symbols in the measure have to be shifted accordingly as well. Following this rule, we obtain a left-aligned score, see Figure A.7.

The reader may have noticed that at the current time the score visualization produced by the MotifViewer lacks some elements like ties, slurs, ornamentations, trills, staccatos, and dynamics. While these symbols are important for producing a proper performance of the notated piece of music, they are of lower relevance when it comes to pattern detection and motivic analysis. In future developments of the system, these shortcomings will be addressed and support of the mentioned score symbols will gradually be added.

Bibliography

[1] Kamil Adiloğlu, Thomas Noll, and Klaus Obermayer. A paradigmatic approach to extract the melodic structure of a musical piece. *Journal of New Music Research*, 35(3):221–236, 2006.

[2] Kamil Adiloğlu and Klaus Obermayer. Finding subsequences of melodies in musical pieces. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 483–487, Barcelona, Spain, 2005.

[3] Rakesh Agrawal, King-Ip Lin, Harpreet S. Sawhney, and Kyuseok Shim. Fast similarity search in the presence of noise, scaling, and translation in time-series databases. In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*, pages 490–501, Zurich, Switzerland, 1995.

[4] Teppo E. Ahonen, Kjell Lemström, and Simo Linkola. Compression-based similarity measures in symbolic, polyphonic music. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 91–96, Miami, FL, USA, 2011.

[5] Günter Altmann. *Musikalische Formenlehre*. Schott, Mainz, Germany, eighth edition, 2001.

[6] American Memory Project-Sheet. Band music from the civil war era. `http://memory.loc.gov/ammem/cwmhtml/cwmhome.html` (accessed: January 2013).

[7] Archival Sound Recordings Project. British Library Sounds. `http://sounds.bl.uk` (accessed: January 2013).

[8] Andreas Arzt, Gerhard Widmer, and Simon Dixon. Automatic page turning for musicians via real-time machine listening. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, pages 241–245, Patras, Greece, 2008.

[9] Jean-Julien Aucouturier and Francois Pachet. Improving timbre similarity: How high's the sky? *Journal of Negative Results in Speech and Audio Sciences*, 1(1), 2004.

[10] Denis Baggi, Adriano Baratè, Goffredo Haus, and Luca Andrea Ludovico. NINA—navigating and interacting with notation and audio. In *Proceedings of the International Workshop on Semantic Media Adaptation and Personalization (SMAP)*, pages 134–139, Washington, DC, USA, 2007. IEEE Computer Society.

[11] David Bainbridge, John Thompson, and Ian H. Witten. Assembling and enriching digital library collections. In *Proceedings of the ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL)*, pages 323–334, Houston, TX, USA, 2003.

[12] Adriano Baratè, Goffredo Haus, Luca A. Ludovico, and Davide A. Mauro. IEEE 1599 for live musical and theatrical performances. *Journal of Multimedia*, 7(2):170–178, 2012.

[13] Adriano Baratè and Luca Andrea Ludovico. New frontiers in music education through the IEEE 1599 standard. In *Proceedings of the International Conference on Computer Supported Education (CSEDU)*, pages 145–151, Porto, Portugal, 2012.

[14] Harold Barlow and Sam Morgenstern. *A Dicitonary of Musical Themes–Revised Edition*. Faber and Faber, 1983.

[15] Mark A. Bartsch and Gregory H. Wakefield. Audio thumbnailing of popular music using chroma-based representations. *IEEE Transactions on Multimedia*, 7(1):96–104, 2005.

[16] Bavarian State Library. `http://www.bsb-muenchen.de/index.php` (accessed: January 2013).

[17] Pierfrancesco Bellini, Ivan Bruno, Paolo Nesi, and Marius B. Spinu. Execution and synchronisation of music score pages and real performance audios. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pages 125–128, Lausanne, Switzerland, 2002.

[18] Juan Pablo Bello and Jeremy Pickens. A robust mid-level representation for harmonic content in music signals. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, London, UK, 2005.

[19] Jon L. Bentley and Thomas A. Ottmann. Algorithms for reporting and counting geometric intersections. *Computers, IEEE Transactions on*, 28(9):643–647, 1979.

[20] René Berndt, Ina Blümel, Michael Clausen, David Damm, Jürgen Diet, Dieter Fellner, Christian Fremerey, Reinhard Klein, Frank Krahl, Maximilian Scherer, Tobias Schreck, Irina Sens, Verena Thomas, and Raoul Wessel. The PROBADO project – approach and lessons learned in building a digital library system for heterogeneous non-textual documents. In *Proceedings of the European Conference on Digital Libraries (ECDL)*, pages 376–383, Glasgow, UK, 2010.

[21] René Berndt, Ina Blümel, and Raoul Wessel. PROBADO3D – towards an automatic multimedia indexing workflow for architectural 3D models. In *Proceedings of the International Conference on Electronic Publishing (ELPUB)*, pages 79–88, Helsinki, Finland, 2010.

[22] René Berndt, Ina Blümel, and Raoul Wessel. PROBADO3D - new ways of indexing and experiencing architectural 3D databases. In *Proceedings of FOCUS K3D Conference on Semantic 3D Media and Content*, pages 89–90, Sophia Antipolis - Méditerranée, France, February 2010.

[23] René Berndt, Harald Krottmaier, Sven Havemann, and Tobias Schreck. The PROBADO-framework: Content-based queries for non-textual documents. In *Proceedings of the International Conference on Electronic Publishing (ELPUB)*, pages

485–500, Milan, Italy, 2009.

[24] Wallace Berry. *Form in Music.* Prentice Hall, Upper Saddle River, NJ, USA, second edition, 1985.

[25] William P. Birmingham, Kevin O'Malley, Jon W. Dunn, and Ryan Scherle. V2V: A second variation on query-by-humming. In *Proceedings of the ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL)*, pages 380–380, Washington, DC, USA, 2003.

[26] William P. Birmingham, Bryan Pardo, Colin Meek, and Jonah Shifrin. The MusArt music-retrieval system: An overview. *D-Lib Magazine*, 8(2):73–81, 2002.

[27] Karl Blessinger. *Grundzüge der musikalischen Formenlehre.* Engelhorn, Stuttgart, Germany, 1926.

[28] Ina Blümel, Jürgen Diet, and Harald Krottmaier. Integrating multimedia repositories into the PROBADO-framework. In *Proceedings of the International Conference on Digital Information Management (ICDIM)*, pages 178–183, London, UK, 2008.

[29] Jan Brase and Ina Blümel. Information supply beyond text: non-textual information at the German National Library of Science and Technology (TIB) – challenges and planning. *Interlending & Document Supply*, 38(2):108–117, 2010.

[30] British Library. Turning the pages. `http://portico.bl.uk/onlinegallery/ttp/ttpbooks.html` (accessed: January 2013).

[31] Judith C. Brown. Calculation of a constant Q spectral transform. *Journal of the Acoustical Society of America*, 89(1):425–434, 1991.

[32] Siglind Bruhn. J. S. Bachs Wohltemperiertes Klavier: Analyse und Gestaltung. `http://edition-gorz.de/bruhn4.html` (accessed: January 2013).

[33] Esben Paul Bugge, Kim Lundsteen Juncher, Brian Søborg Mathiasen, and Jakob Grue Simonsen. Using sequence alignment and voting to improve optical music recognition from multiple recognizers. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 405–410, Miami, FL, USA, 2011.

[34] Juan José Burred. Genetic motif discovery applied to audio analysis. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 361–364, Kyoto, Japan, 2012.

[35] Chantal Butau and Guerimo Mazzola. From contour similarity to motivic topologies. *Musicae Scientiae*, 4(2):125–149, 2000.

[36] Chantal Buteau. Topological spaces of motives of Brahms Op. 51 No1. `http://recherche.ircam.fr/equipes/repmus/mamux/Buteau.pdf` (accessed: January 2013), 2008. International Workshop on Computational Music Analysis.

[37] Chantal Buteau and Guerino Mazzola. Motivic analysis according to Rudolph Réti: formalization by a topological model. *Journal of Mathematics and Music*, 2(3):117–134, 2008.

[38] Chantal Buteau and John Vipperman. Melodic clustering within motivic spaces: Visualization in OpenMusic and application to Schumann's Träumerei. In *Mathe-*

matics and Computation in Music, volume 37 of *Communications in Computer and Information Science*, pages 59–66. Springer, 2009.

[39] Donald Byrd, William Guerin, Megan Schindele, and Ian Knopke. OMR evaluation and prospects for improved OMR via multiple recognizers. Technical report, Indiana University, Bloomington, IN, USA, 2010.

[40] Donald Byrd and Megan Schindele. Prospects for improving OMR with multiple recognizers. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 41–46, Victoria, Canada, 2006.

[41] Emilios Cambouropoulos. Musical parallelism and melodic segmentation: A computational approach. *Music Perception*, 23(3):249–268, 2006.

[42] Emilios Cambouropoulos, Maxime Crochemore, Costas Iliopoulos, Laurent Mouchard, and Yoan Pinzon. Algorithms for computing approximate repetitions in musical sequences. *International Journal of Computer Mathematics*, 79(11):1135–1148, 2002.

[43] Emilios Cambouropoulos, Maxime Crochemore, Costas S. Iliopoulos, Manal Mohamed, and Marie-France Sagot. All maximal-pairs in step-leap representation of melodic sequence. *Information Science*, 177(9):1954–1962, 2007.

[44] Michael Casey, Christophe Rhodes, and Malcolm Slaney. Analysis of minimum distances in high-dimensional musical spaces. *IEEE Transactions on Audio, Speech & Language Processing*, 16(5):1015–1028, 2008.

[45] Center for Computer Assisted Research in the Humanities, Stanford University. MuseData. `http://www.musedata.org` (accessed: January 2013).

[46] Shih-Chuan Chiu, Man-Kwan Shan, Jiun-Long Huang, and Hua-Fu Li. Mining polyphonic repeating patterns from music data using bit-string based approaches. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pages 1170 –1173, New York, NY, USA, 2009.

[47] Taemin Cho, Ron J. Weiss, and Juan Pablo Bello. Exploring common variations in state of the art chord recognition systems. In *Proceedings of the Sound and Music Computing Conference (SMC)*, pages 1–8, Barcelona, Spain, 2010.

[48] Michael Clausen, Roland Engelbrecht, Dirk Meyer, and Jürgen Schmitz. Proms: A web-based tool for searching in polyphonic music. In *International Symposium on Music Information Retrieval*, 2000.

[49] Tom Collins. *Improved methods for pattern discovery in music, with applications in automated stylistic composition.* PhD thesis, The Open University, 2011.

[50] Tom Collins, Jeremy Thurlow, Robin Laney, Alistair Willis, and Paul H. Garthwaite. A comparative evaluation of algorithms for discovering translational patterns in baroque keyboard works. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 3–8, Utrecht, The Netherlands, 2010.

[51] Arshia Cont. ANTESCOFO: anticipatory synchronization and control of interactive parameters in computer music. In *Proceedings of the International Computer Music Conference (ICMC)*, Belfast, Northern Ireland, 2008.

[52] Arshia Cont. A coupled duration-focused architecture for real-time music-to-score alignment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(6):974–987, 2010.

[53] Timothy Crist. The Motive. `http://www.clt.astate.edu/tcrist/theory2/motive.pdf` (accessed: January 2013).

[54] Antonello D'Aguanno and Giancarlo Vercellesi. Automatic music synchronization using partial score representation based on IEEE 1599. *Journal of Multimedia*, 4(1):19–24, 2009.

[55] David Damm. *A Digital Library Framework for Heterogeneous Music Collections—from Document Acquisition to Cross-modal Interaction.* PhD thesis, University of Bonn (in preparation), 2013.

[56] David Damm, Christian Fremerey, Frank Kurth, Meinard Müller, and Michael Clausen. Multimodal presentation and browsing of music. In *Proceedings of the International Conference on Multimodal Interfaces (ICMI)*, pages 205–208, Chania, Crete, Greece, 2008.

[57] David Damm, Christian Fremerey, Verena Thomas, Michael Clausen, Frank Kurth, and Meinard Müller. A digital library framework for heterogeneous music collections – from document acquisition to cross-modal interaction. *International Journal on Digital Libraries*, 12(2-3):53–71, 2012.

[58] David Damm, Frank Kurth, Christian Fremerey, and Michael Clausen. A concept for using combined multimodal queries in digital music libraries. In *Proceedings of the European Conference on Digital Libraries (ECDL)*, pages 261–272, Kanoni, Corfu, Greece, 2009.

[59] Ivan Damnjanovic, Josh Reiss, and Dan Barry. Enabling access to sound archives through integration, enrichment, and retrieval. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pages 1597–1598, Hannover, Germany, 2008.

[60] Roger B. Dannenberg. An on-line algorithm for real-time accompaniment. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 193–198, Paris, France, 1984.

[61] Roger B. Dannenberg and Masataka Goto. Music structure analysis from acoustic signals. In David Havelock, Sonoko Kuwano, and Michael Vorländer, editors, *Handbook of Signal Processing in Acoustics*, volume 1, pages 305–331. Springer, New York, NY, USA, 2008.

[62] Roger B. Dannenberg and Ning Hu. Pattern discovery techniques for music audio. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 63–70, Paris, France, 2002.

[63] Steven B. Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *Readings in Speech Recognition*, pages 65–74, 1990.

[64] Jürgen Diet and Frank Kurth. The Probado music repository at the Bavarian State Library. In *Proceedings of the International Conference on Music Information*

Retrieval (ISMIR), pages 501–504, Vienna, Austria, 2007.

[65] Karin Dressler and Sebastian Streich. Tuning frequency estimation using circular statistics. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 357–360, Vienna, Austria, 2007.

[66] Jon W. Dunn, Donald Byrd, Mark Notess, Jenn Riley, and Ryan Scherle. Variations2: Retrieving and using music in an academic setting. *Communications of the ACM, Special Issue: Music information retrieval*, 49(8):53–48, 2006.

[67] Daniel P. W. Ellis and Graham. E. Poliner. Identifying 'cover songs' with chroma features and dynamic programming beat tracking. In *Proeedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Honolulu, HI, USA, 2007.

[68] European Union. Europeana. `http://www.europeana.eu/portal` (accessed: January 2013).

[69] Sebastian Ewert. *Signal Processing Methods for Music Synchronization, Audio Matching, and Source Separation.* PhD thesis, University of Bonn, 2012.

[70] Sebastian Ewert and Meinard Müller. Estimating note intensities in music recordings. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 385–388, Prague, Czech Republic, 2011.

[71] Sebastian Ewert and Meinard Müller. Score-informed source separation. In Meinard Müller, Masataka Goto, and Markus Schedl, editors, *Multimodal Music Processing*, volume 3 of *Dagstuhl Follow-Ups*, pages 73–94. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 2012.

[72] Christos Faloutsos, M. Ranganathan, and Yannis Manolopoulos. Fast subsequence matching in time-series databases. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*, pages 419–429, Minneapolis, MN, USA, 1994.

[73] Christian Fremerey. *Automatic Organization of Digital Music Documents – Sheet Music and Audio.* PhD thesis, University of Bonn, 2010.

[74] Christian Fremerey, Michael Clausen, Sebastian Ewert, and Meinard Müller. Sheet music-audio identification. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 645–650, Kobe, Japan, 2009.

[75] Christian Fremerey, Meinard Müller, and Michael Clausen. Handling repeats and jumps in score-performance synchronization. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 243–248, Utrecht, The Netherlands, 2010.

[76] Christian Fremerey, Meinard Müller, Frank Kurth, and Michael Clausen. Automatic mapping of scanned sheet music to audio recordings. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 413–418, Philadelphia, PA, USA, September 2008.

[77] Hiromasa Fujihara, Masataka Goto, Jun Ogata, and Hiroshi G. Okuno. LyricSynchronizer: Automatic synchronization system between musical audio signals and

lyrics. *IEEE Journal of Selected Topics in Signal Processing*, 5(6):1252–1261, 2011.

[78] Ichiro Fujinaga. Optical Music Recognition Bibliography. `http://ddmal.music.mcgill.ca/wiki/Optical_Music_Recognition_Bibliography` (accessed: January 2013).

[79] Lora L. Gingerich. A technique for melodic motivic analysis in the music of Charles Ives. *Music Theory Spectrum*, 8:75–93, 1986.

[80] Emilia Gómez. *Tonal Description of Music Audio Signals*. PhD thesis, UPF Barcelona, 2006.

[81] Google Inc. Google Books. `http://books.google.com` (accessed: January 2013).

[82] Markus Gorski. Formenlehre: Die Fuge. `http://www.lehrklaenge.de/PHP/Formenlehre/FugeBWV867.php` (accessed: January 2013).

[83] Masataka Goto. A chorus-section detecting method for musical audio signals. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 437–440, Hong Kong, China, 2003.

[84] Masataka Goto. AIST Annotation for the RWC Music Database. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 359–360, Victoria, Canada, 2006.

[85] Peter Grosche and Meinard Müller. Toward characteristic audio shingles for efficient cross-version music retrieval. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 473–476, Kyoto, Japan, 2012.

[86] Peter Grosche, Meinard Müller, and Joan Serrà. Audio content-based music retrieval. In Meinard Müller, Masataka Goto, and Markus Schedl, editors, *Multimodal Music Processing*, volume 3 of *Dagstuhl Follow-Ups*, pages 157–174. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2012.

[87] Edward W. Guo. International Music Score Library Project. `http://imslp.org/wiki` (accessed: January 2013), 2006.

[88] Andrew Hankinson, Laurent Pugin, and Ichiro Fujinaga. Interfaces for document representation in digital music libraries. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 39–44, Kobe, Japan, 2009.

[89] Christoph Hempel. *Neue Allgemeine Musiklehre*. Schott Music, Mainz, Germany, eighth edition, 2008.

[90] Jia-Lien Hsu, Chih-Chin Liu, and Arbee L. P. Chen. Discovering nontrivial repeating patterns in music data. *Multimedia, IEEE Transactions on*, 3(3):311–325, 2001.

[91] Ning Hu, Roger B. Dannenberg, and George Tzanetakis. Polyphonic audio matching and alignment for music retrieval. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, NY, USA, 2003.

[92] David Huron. Humdrum user guide. `http://www.musiccog.ohio-state.edu/Humdrum/guide.toc.html` (accessed: January 2013).

[93] Werner Icking. Werner Icking Music Archive. `http://icking-music-archive.org` (accessed: January 2013).

[94] IFLA Study Group on the Functional Requirements of Bibliographic Records. Functional Requirements for Bibliographic Records; Final Report. Saur, Munich, 1998. available at `http://www.ifla.org/files/assets/cataloguing/frbr/frbr.pdf` (accessed: January 2013).

[95] Isabella Stewart Gardener Museum. Music Library. `http://www.gardnermuseum.org/music/listen/music_library` (accessed: January 2013).

[96] Cihan Isikhan and Giyasettin Oszcan. A survey of melody extraction techniques for music information retrieval. In *Proceedings of the Conference on Interdisciplinary Musicology*, Thessaloniki, Greece, 2008.

[97] Matevž Jekovec, Janez Demšar, and Andrej Brodnik. Computer aided melodic analysis using suffix tree. In *Proceedings of the International Computer Music Conference Conference (ICMC)*, pages 563–566, Ljubljana, Slovenia, 2012.

[98] John Hopkins University. The Lester S. Levy Collection of Sheet Music. `http://levysheetmusic.mse.jhu.edu` (accessed: January 2013).

[99] Brewster Kahle. Internet Archive. `http://archive.org/index.php` (accessed: January 2013).

[100] Michael Kennedy and Joyce Bourne. Oxford Music Online. `http://www.oxfordmusiconline.com` (accessed:January 2013).

[101] Eamonn Keogh and Chotirat Ann Ratanamahatana. Exact indexing of dynamic time warping. *Knowledge and Information Systems*, 7(3):358–386, 2005.

[102] KernScores. `http://kern.ccarh.org` (accessed: January 2013).

[103] Anssi Klapuri and Manuel Davy, editors. *Signal Processing Methods for Music Transcription.* Springer, New York, 2006.

[104] Verena Konz and Meinard Müller. A cross-version approach for harmonic analysis of music recordings. In Meinard Müller, Masataka Goto, and Markus Schedl, editors, *Multimodal Music Processing*, volume 3 of *Dagstuhl Follow-Ups*, pages 53–71. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 2012.

[105] Clemens Kühn. *Formenlehre der Musik.* Bärenreiter, Kassel, Germany, eighth edition, 2007.

[106] Frank Kurth, David Damm, Christian Fremerey, Meinard Müller, and Michael Clausen. A framework for managing multimodal digitized music collections. In *Proceedings of the European Conference on Digital Libraries (ECDL)*, pages 334–345, Aarhus, Denmark, 2008.

[107] Frank Kurth and Meinard Müller. Efficient index-based audio matching. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):382–395, 2008.

[108] Frank Kurth, Meinard Müller, Christian Fremerey, Yoon-Ha Chang, and Michael Clausen. Automated synchronization of scanned sheet music with audio recordings. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*,

pages 261–266, Vienna, Austria, 2007.

[109] Christian Landone, Joseph Harrop, and Josh Reiss. Enabling access to sound archives through integration, enrichment and retrieval: the EASAIER project. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 159–160, Vienna, Austria, 2007.

[110] Olivier Lartillot. Discovering musical patterns through perceptive heuristics. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 89–96, Washington, DC, USA, 2003.

[111] Olivier Lartillot. Efficient extraction of closed motivic patterns in multi-dimensional symbolic representations of music. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 191–198, London, UK, 2005.

[112] John Lawter and Barry Moon. Score following in open form compositions. In *Proceedings of the International Computer Music Conference (ICMC)*, Ann Arbor, MI, USA, 1998.

[113] Andreas Lehmann. Automatisiertes Motivsuchen in Musikwerken im MusicXML Format. Diploma thesis, Humboldt-Universiät zu Berlin, 2009.

[114] Heinrich Lemacher and Hermann Schroeder. *Formenlehre der Musik*. Hans Gerig Verlag, Köln, Germany, fourth edition, 1974.

[115] Kjell Lemström and Mika Laitinen. Transposition and time-warp invariant geometric music retrieval algorithms. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, Barcelona, Spain, 2011.

[116] Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

[117] Lewis Music Library. Inventions of note sheet music collection. `http://libraries.mit.edu/music/sheetmusic` (accessed: January 2013).

[118] Beth Logan. Mel frequency cepstral coefficients for music modeling. In *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)*, Plymouth, MA, USA, 2000.

[119] Gareth Loy. *Musimathics, Volume 1: The Mathematical Foundations of Music*. The MIT Press, 2006.

[120] Luca A. Ludovico. IEEE 1599: a multi-layer approach to music description. *Journal of Multimedia*, 4(1):9–14, 2009.

[121] Laura Macy. Grove Music Online. `http://www.oxfordmusiconline.com` (accessed: January 2013), 2001.

[122] Alan Mardsen. Recognition of variations using automatic Schenkerian reduction. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 501–506, Utrecht, The Netherlands, 2010.

[123] Matthias Mauch and Simon Dixon. Simultaneous estimation of chords and musical context from audio. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1280–1289, 2010.

[124] Guerino Mazzola. Mathematical music theory - status quo 2000, 2001.

[125] Guerino Mazzola. *The topos of music.* Birkhäuser, 2002.

[126] Richard A. Medina, Lloyd A. Smith, and Deborah R. Wagner. Content-based indexing of musical scores. In *Proceedings of the Joint Conference on Digital Libraries (JCDL)*, pages 18–26, Houston, TX, USA, 2003.

[127] Collin Meek and William P. Birmingham. Thematic extractor. In *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)*, Bloomington, IN, USA, 2001.

[128] David Meredith. Point-set algorithms for pattern discovery and pattern matching in music. In Tim Crawford and Remco C. Veltkamp, editors, *Content-Based Retrieval*, number 06171 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2006. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany.

[129] David Meredith, Kjell Lemström, and Geraint A. Wiggins. Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music. *Journal of New Music Research*, 31(4):321–345, 2002.

[130] Ulrich Michels. *dtv-Atlas Musik - Band 1: Systematischer Teil.* Deutscher Taschenbuch Verlag, Munich, Germany, 22nd edition, 2008.

[131] Nicola Montecchio and Arshia Cont. A unified approach to real time audio-to-score and audio-to-audio alignment using sequential Montecarlo inference techniques. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 193–196, Prague, Czech Republic, 2011.

[132] Nicola Montecchio and Nicola Orio. A discrete filter bank approach to audio to score matching for polyphonic music. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 495–500, Kobe, Japan, 2009.

[133] Meinard Müller. *Information Retrieval for Music and Motion.* Springer Verlag, 2007.

[134] Meinard Müller and Daniel Appelt. Path-constrained partial music synchronization. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 65–68, Las Vegas, Nevada, USA, 2008.

[135] Meinard Müller and Michael Clausen. Transposition-invariant self-similarity matrices. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 47–50, Vienna, Austria, 2007.

[136] Meinard Müller and Sebastian Ewert. Joint structure analysis with applications to music annotation and synchronization. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 389–394, Philadelphia, PA, USA, 2008.

[137] Meinard Müller and Sebastian Ewert. Towards timbre-invariant audio features for harmony-based music. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):649–662, 2010.

[138] Meinard Müller and Sebastian Ewert. Chroma Toolbox: MATLAB implementations for extracting variants of chroma-based audio features. In *Proceedings of the Interna-*

tional Conference on Music Information Retrieval (ISMIR), pages 215–220, Miami, FL, USA, 2011.

[139] Meinard Müller and Nanzhu Jiang. A scape plot representation for visualizing repetitive structures of music recordings. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 97–102, Porto, Portugal, 2012.

[140] Meinard Müller and Frank Kurth. Towards structural analysis of audio recordings in the presence of musical variations. *EURASIP Journal on Advances in Signal Processing*, 2007(1), 2007.

[141] Meinard Müller, Frank Kurth, and Michael Clausen. Audio matching via chroma-based statistical features. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 288–295, London, UK, 2005.

[142] Meinard Müller, Frank Kurth, David Damm, Christian Fremerey, and Michael Clausen. Lyrics-based audio retrieval and multimodal navigation in music collections. In *Proceedings of the European Conference on Digital Libraries (ECDL)*, pages 112–123, Budapest, Hungary, 2007.

[143] Armando Muscariello, Guillaume Gravier, and Frédéric Bimbot. Variability tolerant audio motif discovery. In *Proceedings of the International Multimedia Modeling Conference on Advances in Multimedia Modeling (MMM)*, pages 275–286, Sophia-Antipolis, France, 2009.

[144] Music Publishers' Association of the United States. Standard music notation practice. `http://icking-music-archive.org/lists/sottisier/notation.pdf` (accessed: January 2013).

[145] Music Scanning. Sharpeye. `http://www.music-scanning.com/sharpeye.html` (accessed: January 2013).

[146] Mutopia Project. Music free to download, print out, perform and distribute. `http://www.mutopiaproject.org` (accessed: January 2013).

[147] Jean-Jaques Nattiez. *Music and Discourse: Toward a Semiology of Music.* Princeton University Press, Princeton, NJ, USA, 1990.

[148] New Zealand Digital Library Project. Greenstone. `http://www.greenstone.org` (accessed: January 2013).

[149] Bernhard Niedermayer. Improving accuracy of polyphonic music-to-score alignment. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 585–590, Kobe, Japan, 2009.

[150] Franz Nistl. Correct tuning pitch of many international orchestras. `http://members.aon.at/fnistl/index.html` (accessed: January 2013).

[151] Panagiotis Papapetrou, Vassilis Athitsos, Michalis Potamias, George Kollios, and Dimitrios Gunopulos. Embedding-based subsequence matching in time series databases. *ACM Transactions on Database Systems (TODS)*, 36(3), 2011.

[152] Bryan Pardo and William P. Birmingham. Modeling form for on-line following of musical performances. In *Proceedings of the National Conference on Artificial*

Intelligence (AAAI), pages 1018–1023, Pittsburgh, PA, USA, 2005.

[153] Jouni Paulus, Meinard Müller, and Anssi Klapuri. Audio-based music structure analysis. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 625–636, Utrecht, The Netherlands, 2010.

[154] Alberto Pinto. Multi-model music content description and retrieval using IEEE 1599 XML standard. *Journal of Multimedia*, 4(1):30–39, 2009.

[155] Dennis Howard Pruslin. *Automatic Recognition of Sheet Music.* PhD thesis, Massachusetts Institute of Technology, 1966.

[156] Quaero. `http://www.quaero.org` (accessed: January 2013).

[157] Quaero. Exalead. `http://www.exalead.com/search/` (accessed: January 2013).

[158] Lawrence Rabiner and Bing-Hwang Juang. *Fundamentals of Speech Recognition.* Prentice Hall Signal Processing Series, 1993.

[159] Thanawin Rakthanmanon, Bilson Campana, Abdullah Mueen, Gustavo Batista, Brandon Westover, Qiang Zhu, Jesin Zakaria, and Eamonn Keogh. Addressing big data time series: Mining trillions of time series subsequences under dynamic time warping (preprint). 2012.

[160] Thanawin Rakthanmanon, Bilson Campana, Abdullah Mueen, Gustavo Batista, Brandon Westover, Qiang Zhu, Jesin Zakaria, and Eamonn Keogh. Searching and mining trillions of time series subsequences under dynamic time warping. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 262–270, Beijing, China, 2012.

[161] Christopher Raphael. Music Plus One: A system for flexible and expressive musical accompaniment. In *Proceedings of the International Computer Music Conference (ICMC)*, Havana, Cuba, 2001.

[162] Christopher Raphael. A hybrid graphical model for aligning polyphonic audio with musical scores. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 387–394, Barcelona, Spain, 2004.

[163] Christopher Raphael. Music Plus One and machine learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, Haifa, Israel, 2010.

[164] Erwin Ratz. *Einführung in die Musikalische Formenlehre.* Universal Edition, third edition, 1973.

[165] Recordare. Musicxml 3.0 tutorial. `http://downloads2.makemusic.com/MusicXML/musicxml-tutorial.pdf` (accessed: January 2013).

[166] Xiaona Ren, Lloyd A. Smith, and Richard A. Medina. Discovery of retrograde and inverted themes for indexing musical scores. In *Proceedings of the Joint Conference on Digital Libraries (JCDL)*, pages 252–253, Tuscon, AZ, USA, 2004.

[167] Pierre-Yves Rolland. FlExPat: Flexible extraction of sequential patterns. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pages 481–488, San Jose, CA, USA, 2001.

[168] Pierre-Yves Rolland and Jean-Gabriel Ganascia. Pattern detection and discovery: The case of music data mining. In David Hand, Niall Adams, and Richard Bolton, editors, *Pattern Detection and Discovery*, volume 2447 of *Lecture Notes in Computer Science*, pages 69–87. Springer Berlin / Heidelberg, 2002.

[169] Azriel Rosenfeld and John L. Pfaltz. Sequential operations in digital picture processing. *Journal of the ACM*, 13(4):471–494, 1966.

[170] Joe Cheri Ross, Vinutha T. P., and Preeti Rao. Detecting melodic motifs from audio for Hindustani classical music. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 193–198, Porto, Portugal, 2012.

[171] Stanley Sadie, editor. *The New Grove Dictionary of Music and Musicians (second edition)*. Macmillan, London, UK, 2001.

[172] Franz Sauter. *Die tonale Musik*. Books on Demand, third edition, 2010.

[173] Franz Sauter. "Motiv" – Online-Musiklexikon. `http://www.tonalemusik.de/musiklexikon.htm` (accessed: January 2013), 2010.

[174] Markus Schäfer. Hochwertige automatische Extraktion von Gesangstext aus Notenbänden und mediensynchrone Darstellung. Diploma thesis, University of Bonn, 2012.

[175] Heinz-Christian Schaper. *Musikform compact*. Schott Music, Mainz, Germany, 1982.

[176] Arnold Schönberg. *Style and Idea*. University of California Press, 1975.

[177] Robert Schumann. *Musikalische Haus- und Lebensregeln*. Schuberth & Co., Leipzig, Germany, 1860.

[178] Roger Scruton. *The Aesthetics of Music*. Oxford University Press, Oxford, Great Britain, 1999.

[179] Alexander Sheh and Daniel P. W. Ellis. Chord segmentation and recognition using EM-trained hidden Markov models. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Baltimore, MD, USA, 2003.

[180] Roger N. Shepard. Circularity in judgments of relative pitch. *Journal of the Acoustic Society of America*, 36(12):2346–2353, 1964.

[181] Robert Ståhlbrand, Erik Helling, and Joffrey Wallaart. Piano Society. `http://pianosociety.com` (accessed: January 2013).

[182] Leon Stein. *Structure and Style*. Summy-Birchard, third edition, 1979.

[183] Richard Stöhr. *Formenlehre der Musik*. Kistner & Siegel, Leipzig, Germany, 1933.

[184] Atsuhiro Takasu, Takashi Yanase, Teruhito Kanazawa, and Jun Adachi. Music structure analysis and its application to theme phrase extraction. In *Proceedings of the Conference on Research and Advanced Technology for Digital Libraries (ECDL)*, pages 92–105, Paris, France, 1999.

[185] Mevlut Evren Tekin, Christina Anagnostopoulou, and Yo Tomita. Towards an intelligent score following system: Handling of mistakes and jumps encountered during piano practicing. In *Proceedings of the International Symposium on Computer*

Music Modeling and Retrieval (CMMR), pages 211–219, Pisa, Italy, 2005.

[186] Hiroko Terasawa, Malcolm Slaney, and Jonathan Berger. The thirteen colors of timbre. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 323–326, New Paltz, NY, USA, 2005.

[187] The Electronic Dictionary of Musical Themes. `http://www.multimedialibrary.com/barlow/index.asp` (accessed: January 2013).

[188] The Juilliard School. Juilliard Manuscript Collection. `http://www.juilliardmanuscriptcollection.org` (accessed: January 2013).

[189] Verena Thomas and Michael Clausen. MotifViewer: Hierarchical pattern detection. In *Proceedings of the International Computer Music Conference Conference (ICMC)*, pages 555–558, Ljubljana, Slovenia, 2012.

[190] Verena Thomas, David Damm, Christian Fremerey, Michael Clausen, Frank Kurth, and Meinard Müller. PROBADO music: A multimodal online music library. In *Proceedings of the International Computer Music Conference Conference (ICMC)*, pages 289–292, Ljubljana, Slovenia, 2012.

[191] Verena Thomas, Sebastian Ewert, and Michael Clausen. Fast intra-collection audio matching. In *Proceedings of the ACM Workshop on Music Information Retrieval with User-Centered and Multimodal Strategies (MIRUM)*, pages 1–6, Nara, Japan, 2012.

[192] Verena Thomas, Christian Fremerey, David Damm, and Michael Clausen. SLAVE: a score-lyrics-audio-video-explorer. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 717–722, Kobe, Japan, 2009.

[193] Verena Thomas, Christian Fremerey, Sebastian Ewert, and Michael Clausen. Notenschrift-Audio Synchronisation komplexer Orchesterwerke mittels Klavierauszug. In *Proceedings of the Deutsche Jahrestagung für Akustik (DAGA)*, pages 191–192, Berlin, Germany, 2010.

[194] Verena Thomas, Christian Fremerey, Meinard Müller, and Michael Clausen. Linking sheet music and audio - challenges and new approaches. In Meinard Müller, Masataka Goto, and Markus Schedl, editors, *Multimodal Music Processing*, volume 3 of *Dagstuhl Follow-Ups*, pages 1–22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 2012.

[195] Verena Thomas, Christian Wagner, and Michael Clausen. OCR-based post-processing of OMR for the recovery of transposing instruments in complex orchestral scores. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 411–416, Miami, FL, USA, 2011.

[196] Alexandra L. Uitdenbogerd and Justin Zobel. Manipulation of music for melody matching. In *Proceedings of the ACM International Conference on Multimedia*, pages 235–240, New York, NY, USA, 1998.

[197] Union der deutschen Akademien der Wissenschaften. Neue Mozart-Ausgabe. `http://www.nma.at` (accessed: January 2013).

[198] University of Chicago Library. Chopin Early Editions. `http://chopin.lib.uchicago.edu` (accessed: January 2013).

[199] University of Oxford. Digital Image Archive of Medieval Music. `http://www.diamm.ac.uk` (accessed: January 2013).

[200] University of Rochester Libraries. UR Research–Sibley Music Library. `https://urresearch.rochester.edu/viewInstitutionalCollection.action?collectionId=63` (accessed: January 2013).

[201] U.S. Library of Congress. World Digital Library. `http://www.wdl.org/en/` (accessed: January 2013).

[202] Barry Vercoe. The synthetic performer in the context of live performance. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 199–200, Paris, France, 1984.

[203] Vladimir Viro. Peachnote: Music score search and analysis platform. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 359–362, Miami, FL, USA, 2011.

[204] Christian Wagner. OCR based postprocessing of OMR results in complex orchestral scores – Which (transposing) instrument corresponds in which staff? Diploma thesis, University of Bonn, 2011.

[205] Jimmy Wales and Larry Sanger. Wikipedia. `http://en.wikipedia.org` (accessed: January 2013), 2001.

[206] Avery Wang. An industrial strength audio search algorithm. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 7–13, Baltimore, MD, USA, 2003.

[207] Ye Wang, Min-Yen Kan, Tin Lay Nwe, Arun Shenoy, and Jun Yin. LyricAlly: Automatic synchronization of textual lyrics to acoustic music signals. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(2):338–349, 2008.

[208] Wienbibliothek im Rathaus. Schubert-Autographe. `http://www.schubert-online.at` (accessed: January 2013).

[209] Ian H. Witten, Rodger J. Mcnab, Stefan J. Boddie, and David Bainbridge. Greenstone: A comprehensive open-source digital library software system. In *Proceedings of the ACM International Conference on Digital Libraries (ACM DL)*, pages 113–121, San Antonio, TX, USA, 2000.

[210] Guangyu Xia, Dawen Liang, Roger B. Dannenberg, and Mark J. Harvilla. Segmentation, clustering, and display in a personal audio database for musicians. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 139–144, Miami, FL, USA, 2011.

[211] Cheng Yang. Efficient acoustic index for music retrieval with various degrees of similarity. In *Proceedings of the ACM International Conference on Multimedia (ACMMM)*, pages 584–591, Juan les Pins, France, 2002.

[212] Yunyue Zhu and Dennis Shasha. Warping indexes with envelope transforms for query by humming. In *Proceedings of the ACM International Conference on Management of data (ACM SIGMOD)*, pages 181–192, San Diego, CA, USA, 2003.

[213] Wieland Ziegenrücker. *ABC Musik. Allgemeine Musiklehre.* Breitkopf und Härtel, Wiesbaden, Germany, 1982.

[214] Eberhard Zwicker and Hugo Fastl. *Psychoacoustics, facts and models.* Springer Verlag, New York, NY, US, 1990.

List of Abbreviations

BPM	beats per minute
CC	connected component
CD	Compact Disc
CENS	chroma energy normalized statistics
CPN	common practice notation
CRP	chroma DCT-reduced log pitch
DCT	discrete cosine transform
DTW	dynamic time warping
Hz	hertz
LBF	lower bounding function
LSH	locality sensitive hashing
MFCC	mel-frequency cepstral coefficient
MIDI	Musical Instrument Digital Interface
MIR	Music Information Retrieval
OCR	optical character recognition
OMR	optical music recognition
PFCC	pitch-frequency cepstral coefficient
SSDTW	subsequence dynamic time warping
STMSP	short-time mean-square power
XML	Extensible Markup Language

Index