JOHANNES KEPLER
UNIVERSITY LINZ

Submitted by
Andreas Arzt

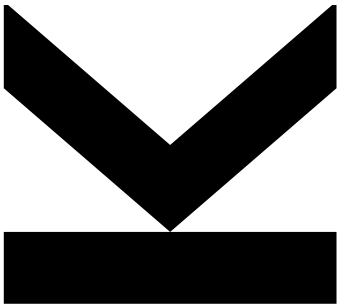
Submitted at
Department of Computational Perception

Supervisor and
First Examiner
Gerhard Widmer

Second Examiner
Meinard Müller

November 2016

Flexible and Robust Music Tracking



Doctoral Thesis

to obtain the academic degree of

Doktor der technischen Wissenschaften

in the Doctoral Program

Technische Wissenschaften

JOHANNES KEPLER
UNIVERSITY LINZ
Altenbergerstraße 69
4040 Linz, Österreich
www.jku.at
DVR 0093696

STATUTORY DECLARATION

I hereby declare that the thesis submitted is my own unaided work, that I have not used other than the sources indicated, and that all direct and indirect sources are acknowledged as references. This printed thesis is identical with the electronic version submitted.

Linz, November 2016

ACKNOWLEDGMENTS

The work described in the thesis was carried out while I was working at the Department of Computational Perception (CP) at Johannes Kepler University in Linz, and at the Austrian Research Institute for Artificial Intelligence (OFAI) in Vienna. I would like to thank my supervisor Gerhard Widmer for giving me the opportunity to work on his teams, and for all the freedom, trust and support over the years.

I would like to thank all my colleagues at the CP and at OFAI, who jointly are responsible for a unique, friendly, relaxed, fun, and at the same time immensely productive and creative work environment. Special thanks go to Harald Frostel and Martin Gasser, for their support regarding implementation frameworks, annotation tools, and their constant criticism of my coding style; Sebastian Flossmann, who meticulously annotated an immense amount of sheet music and performances, of which the thesis took advantage; Thassilo Gadermaier and Maarten Grachten, for sharing the pain of working with MusicXML and MIDI; Sebastian Böck, whose transcription algorithms play an important role in the thesis; Reinhard Sonnleitner, for many fruitful discussions about the latest trends in music fingerprinting; Filip Korzeniowski and Florian Krebs, for sharing their knowledge about probabilistic models with me; Matthias Dorfer, for his insights into deep learning; Tom Collins, for sharing his knowledge about music with me; Peter Knees, for sharing his thoughts about research, culture and arts; and to Claudia Kindermann, the heart and soul of the CP.

Many thanks to Simon Dixon, who back when I was a master's student shared his research code with me, which led to a master's thesis, a first published paper, and in the long run to this doctoral thesis. I would also like to thank Meinard Müller for many interesting discussions at conferences, for agreeing to be an examiner of the thesis, and for helping improving this document.

Special thanks go to Cynthia Liem, who always found time in her incredibly busy schedule to perform and showcase the outcomes of this thesis at conferences and various other events around the world, and to Marcel van Tilburg, for sharing some good sherry and great conversations. I would also like to thank all the great people involved in the Phenicx project, which gave me the opportunity to showcase the music tracker at the wonderful Concertgebouw in Amsterdam. Furthermore, I would like to thank Werner Goebl, for his help with demonstrating the music tracker, and for his valuable feedback.

Overlapping and slightly anonymised thanks go to all my friends, especially the boat trip crew, the Offensee crew, the bavarian curling crew, the cocktail crew, the barbecue crew, the whisky tasting crew, the nightlife crew, and the Bomberman crew. It was a blast.

Last but definitely not least, I would like to thank my amazing family for their never ending love and support.

This research is supported by the City of Linz, the Federal State of Upper Austria, the Austrian Federal Ministry for Transport, Innovation and Technology, the Austrian Science Fund (FWF) under project numbers P19349-N15, TRP109-N23 and Z159, the EU FP7 project Phenicx (grant no. 601166), and the EU ERC project Con Espressione (grant agreement 670035).

ABSTRACT

Computers nowadays are prevalent in all areas of music, from the compositional process to music production, be it in the studio or live on stage. The thesis is concerned with a specific problem in music processing, namely *score following*, also known as *real-time music tracking*. Casually speaking, a music tracking algorithm "listens" to a live performance of music, compares the incoming audio signal to a representation of the score, and "reads" along, i.e. at any given moment it knows the exact position of the musician(s) in the sheet music. This information enables a wide range of applications, e.g. visualisations synchronised to live music, automatic page-turning of the score, and automatic accompaniment.

The focus of the thesis is on robust and flexible music tracking algorithms for Western classical music which overcome the limitations of other existing algorithms. The main contributions of the thesis are

- improved *algorithms*, *features*, and *tempo models*, increasing the robustness and accuracy of music tracking,
- a very robust *multi-agent music tracking approach* which enables robust tracking e.g. of complex orchestral music, and
- fast *music identification algorithms* which enable flexible *any-time music tracking* that is not limited to tracking a single, predefined piece, but works flexibly on a (large) database of sheet music.

In addition to quantitative experiments on diverse datasets, the algorithms described in this thesis were tested in real-world settings — live in front of an audience. This includes demonstrations at scientific conferences and galas, but also the culminating point of this thesis: a live demonstration of robust music tracking technology at the Concertgebouw in Amsterdam. There, our system tracked a live performance of the Alpensinfonie by Richard Strauss and was used to show synchronised visualisations — the sheet music with automatic page turning, artistic videos, and textual information provided by a musicologist — to the audience.

KURZFASSUNG

Computer sind aus der Musik nicht mehr wegzudenken und spielen eine wichtige Rolle während des Kompositionsprozesses, der Produktion und auch während Konzerten live auf der Bühne. Diese Dissertation beschäftigt sich mit einem spezifischen Problem der Musikverarbeitung, dem *Score Following*, auch bekannt als *Music Tracking*, also dem *Verfolgen von Musik*. Dabei „hört" sozusagen ein Musikverfolgungsalgorithmus einem Livekonzert zu, vergleicht das aufgenommene Audiosignal mit einer abstrakten Repräsentation des Notentextes, und „liest" in diesem mit. Das heißt, der Algorithmus kennt zu jedem Zeitpunkt die aktuelle Stelle der Musiker im Notentext. Mithilfe dieser Information lassen sich Applikationen realisieren, die zum Beispiel Visualisierungen automatisch zur Musik synchronisieren, den Notentext automatisch umblättern, oder eine synchronisierte Begleitstimme einspielen.

Der Fokus dieser Dissertation liegt auf flexiblen und robusten Musikverfolgungsalgorithmen für Klassische Musik. Die wichtigsten Beiträge dieser Dissertation sind

- verbesserte *Algorithmen*, *Features* und *Tempomodelle*, die die Robustheit und Genauigkeit des Verfolgungsalgorithmus erhöhen,
- ein robuster *Multi-Agenten Musikverfolgungsalgorithmus*, der auch komplexe Orchestermusik zuverlässig verfolgen kann und
- *Algorithmen, die blitzschnell Musikstücke identifizieren* und als Basis für einen Musikverfolgungsalgorithmus dienen, der flexibles Verfolgen auf Basis einer großen Datenbank an Notentexten realisiert.

Zusätzlich zu quantitativen Experimenten auf vielfältigen Datensammlungen wurden die vorgestellten Algorithmen auch auf wissenschaftlichen Konferenzen und Galas live vor Publikum präsentiert. Der Höhepunkt dieser Dissertation ist aber unzweifelhaft die Demonstration des Musikverfolgungsalgorithmus im Concertgebouw in Amsterdam. Der Algorithmus verfolgte dort eine Aufführung der Alpensinfonie von Richard Strauss und wurde verwendet, um für das Publikum Informationen — den Notentext, künstlerische Videos und informativen Text — zur Musik synchronisiert anzuzeigen.

CONTENTS

Part I

INTRODUCTION AND BACKGROUND

1 INTRODUCTION

Nowadays computers play an important role in all areas of music, from composition to production and live performance. This is obvious for all kinds of popular music, where complete genres are unthinkable without modern technology, or are actually defined by technology, like *electronic music*.

At first sight, for *classical music* the importance of technology might not be equally apparent. Of course, modern developments in classical music which incorporate electronics and computers during composition and/or performance come to mind, with the likes of Karlheinz Stockhausen and Pierre Boulez being some of the main exponents. But Beethoven and technology? An auto-tuned Mozart aria? A synthetic Horowitz playing Chopin at the Musikverein in Vienna?

Actually, even in the world of traditional classical music (and by this vague term I mean the world of famous opera houses and world-renowned concert halls which keep playing basically the same repertoire for decades), one can witness a slow but steady movement towards openness to technology.

For instance, the Vienna State Opera, a traditional opera house with a history dating back to the 1850s, introduced both a live and an on-demand subscription-based streaming service. This service is enriched by the option to use a second screen (e.g. a tablet computer or a mobile phone) to show additional information like multilingual subtitles or historic sheet music, synchronised to the stream[1]. While technically a relatively simple solution was chosen — the synchronisation itself is actually done manually at the concert hall, and the data is transmitted hidden in the audio signal (audio watermarking) — this product shows the opportunities technology can open up to broaden the audience and find new markets for classical music via sensible use of technology.

The technological backbone for this, and especially for more challenging future applications, can be provided by research conducted in the area of *music information retrieval* (MIR) during the last 20 years. This is a vast field, including but not limited to text analysis of web pages and micro blogs to infer musical knowledge (e.g. to find trending artists, or to classify songs into genres), analysis of representations of the sheet music (e.g. to identify the structure of a piece of music, or find common patterns), and audio analysis, both of recordings and of live streams (e.g. to track the beat, or to find out which songs in a database are similar to each other).

1 See http://www.staatsoperlive.com for more information.

The thesis is mainly concerned with the task of *tracking a live performance of music* on-line, in real-time. In the literature, this is also known as *score following*. Casually speaking, the computer is listening to a live performance and reading along in the sheet music (which internally is represented in some abstract way). At any point in time the computer "knows" the musicians' current position in the piece. This information can then be used for a wide range of applications that involve synchronisation of data to music.

The thesis starts with an algorithm that, though being state of the art at that time, was barely working for medium-complex piano music, and ends at the world-famous Concertgebouw in Amsterdam, where the resulting system of this thesis followed a complete orchestra — more than 100 musicians (!) — performing the Alpensinfonie by Richard Strauss. There, it listened to the live performance for about 50 minutes and read along in the sheet music. Based on the output of this system, visualisations were synchronised to the music and presented to the audience. On their tablet computers and mobile phones they could read along in the sheet music (including highlighting of the current bar and automatic page turning), watch artistic videos, visualising the topics of the episodes of this piece, and read informative texts, prepared by a musicologist.

As can be seen by this example, real-time music tracking algorithms are already stable enough to be used in some real-life contexts, and are enabling exciting applications.

1.1 CONTRIBUTIONS OF THIS THESIS

The thesis focuses on three main topics in real-time music tracking.

IMPROVEMENTS TO MUSIC TRACKING VIA ON-LINE TIME WARPING At the heart of the systems and applications presented in this document there is a music tracking algorithm. Based on a representation of the score, it tries to follow a live concert and at any time reports back the current position of the performers in the score. To build advanced systems, this algorithm has to work robustly and also accurately, which is why a part of the thesis is concerned with improving a well-known score following algorithm, proposing better suited features, and extending it with a tempo model. This is documented in Chapter 3.

ROBUST MULTI-AGENT MUSIC TRACKING Although the tracking algorithm is very robust in itself, a new approach was developed in response to an important real-life challenge: tracking a big orchestra playing a complicated piece in a famous concert hall. For this, a multi-agent tracking strategy was developed, which tracks the live

performance using multiple recordings of performances of the same piece as a reference. This work is presented in Chapter 4.

NEAR-INSTANT PIECE IDENTIFICATION AND FLEXIBLE MUSIC TRACKING ON A DATABASE OF SCORES Traditionally, the task of music tracking is defined as following a live performance, from start to end, based on the score of a particular piece. The score is loaded beforehand, the operator hits the start button at the right moment, and the algorithm follows the musician(s), as long as they follow the score closely. The thesis presents a much more flexible approach. Operating on a database of pieces, a piece identification algorithm determines which piece is being played and where in the piece the performers currently are. Then, the system tracks the progress of the musician(s) over time — constantly trying to recognise any jumps, within a piece or to a different piece in the background, and thus being able to follow them in a very flexible way. Chapter 5 documents the progress in this area.

1.2 ORGANISATION OF THE THESIS

The thesis is heavily based on peer-reviewed papers, published during the last few years, which described my contributions in the area of music tracking. It consists of three parts.

PART I: INTRODUCTION AND BACKGROUND **Chapter 1** gives a brief introduction to this thesis. It motivates the work, provides an overview on the contributions of the thesis, its structure, and gives a list of the publications on which the thesis is being based. **Chapter 2** prepares the ground for the following chapters. In particular, Section 2.1 explains the motivation, or "vision", of a *Complete Classical Music Companion*, which guided my work in this area considerably. Section 2.2 briefly describes the data that was gathered and used during the course of this thesis, while Section 2.3 gives some insight into the framework that was used to implement the algorithms presented here. Section 2.4 summarises the related work in a coherent way. In particular, efforts towards music synchronisation and tracking, music identification, and approaches related to the vision of a *Complete Classical Music Companion*, including commercially available products, are presented.

PART II: CONTRIBUTIONS OF THIS THESIS This part is heavily based on peer-reviewed papers and describes the contributions of the thesis in detail. It is structured according to the main contributions, i.e. **Chapter 3** is concerned with improvements of the basic tracking algorithms, **Chapter 4** with the multi-agent tracking approach, and

Chapter 5 with music identification and flexible any-time tracking based on a database of scores.

PART III: LIVE DEMONSTRATIONS AND CONCLUSIONS **Chapter 6** describes the prototypical applications that we implemented based on the research described in this thesis. It also gives an overview of all the demonstrations that were done over the last few years of the standard music tracking algorithm, the multi-agent tracker, and the flexible music companion. Finally, the thesis closes with **Chapter 7**, in which conclusions are presented and possible future work and follow-up projects are outlined.

1.3 MAIN PUBLICATIONS

The following list gives the main publications on which the thesis is based.

- 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)*. Patras, Greece, 2008, pp. 241–245
- Andreas Arzt and Gerhard Widmer. "Simple Tempo Models for Real-time Music Tracking". In: *Proceedings of the Sound and Music Computing Conference (SMC)*. Barcelona, Spain, 2010
- Andreas Arzt and Gerhard Widmer. "Towards Effective 'Any-Time' Music Tracking". In: *Proceedings of the Starting AI Researchers' Symposium (STAIRS)*. Lisbon, Portugal, 2010, pp. 24–36
- Andreas Arzt, Gerhard Widmer, and Simon Dixon. "Adaptive Distance Normalization for Real-time Music Tracking". In: *Proceedings of the European Signal Processing Conference (EUSIPCO)*. Bucharest, Romania, 2012, pp. 2689–2693
- Andreas Arzt, Sebastian Böck, and Gerhard Widmer. "Fast Identification of Piece and Score Position via Symbolic Fingerprinting". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Porto, Portugal, 2012, pp. 433–438
- Andreas Arzt, Gerhard Widmer, and Reinhard Sonnleitner. "Tempo- and Transposition-invariant Identification of Piece and Score Position". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Taipeh, Taiwan, 2014, pp. 549–554
- Andreas Arzt, Harald Frostel, Thassilo Gadermaier, Martin Gasser, Maarten Grachten, and Gerhard Widmer. "Artificial Intelligence in the Concertgebouw". In: *Proceedings of the International*

Joint Conference on Artificial Intelligence (IJCAI). Buenos Aires, Argentina, 2015, pp. 2424–2430

- Andreas Arzt and Gerhard Widmer. "Real-time Music Tracking using Multiple Performances as a Reference". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Málaga, Spain, 2015, pp. 357–363

1.4 ADDITIONAL PUBLICATIONS

The following publications were not considered directly in this thesis, but are related to the topic. This includes publications of which I am not the main author and just contributed in minor ways, and papers that used technology developed in this thesis for a different use case, descriptions of demonstrations, or very recent publications that already go beyond the scope of the thesis and are first steps towards the goals formulated in the conclusion (see Chapter 7).

- Andreas Arzt and Gerhard Widmer. "Robust Real-time Music Tracking". In: *Proceedings of the Vienna Talk on Musical Acoustics (VITA)*. Vienna, Austria, 2010, pp. 5–8
- Harald Frostel, Andreas Arzt, and Gerhard Widmer. "The Vowel Worm: Real-Time Mapping and Visualisation of Sung Vowels in Music". In: *Proceedings of the Sound and Music Computing Conference (SMC)*. Padova, Italy, 2011, pp. 214–219
- Sebastian Böck, Andreas Arzt, Florian Krebs, and Markus Schedl. "Online Real-Time Onset Detection with Recurrent Neural Networks". In: *Proceedings of the International Conference on Digital Audio Effects (DAFx)*. York, United Kingdom, 2012
- Andreas Arzt, Gerhard Widmer, Sebastian Böck, Reinhard Sonnleitner, and Harald Frostel. "Towards a Complete Classical Music Companion". In: *Proceedings of the European Conference on Artificial Intelligence (ECAI)*. Montpellier, France, 2012, pp. 67–72
- Filip Korzeniowski, Florian Krebs, Andreas Arzt, and Gerhard Widmer. "Tracking Rests and Tempo Changes: Improved Score Following with Particle Filters". In: *Proceedings of the International Computer Music Conference (ICMC)*. Perth, Australia, 2013
- Maarten Grachten, Martin Gasser, Andreas Arzt, and Gerhard Widmer. "Automatic Alignment of Music Performances with Structural Differences". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Curitiba, Brazil, 2013, pp. 607–612
- Tom Collins, Andreas Arzt, Sebastian Flossmann, and Gerhard Widmer. "SIARCT-CFP: Improving Precision and the Discovery

of Inexact Musical Patterns in Point-set Representation". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Curitiba, Brazil, 2013, pp. 549–554

- Andreas Arzt, Sebastian Böck, Sebastian Flossmann, Harald Frostel, Martin Gasser, and Gerhard Widmer. "The Complete Classical Music Companion V0.9". In: *Proceedings of the 53rd AES Conference on Semantic Audio*. London, England, 2014
- Andreas Arzt, Sebastian Böck, Sebastian Flossmann, Harald Frostel, Martin Gasser, Cynthia C.S. Liem, and Gerhard Widmer. "The Piano Music Companion". In: *Proceedings of the Conference on Prestigious Applications of Intelligent Systems (PAIS)*. Prague, Czech Republic, 2014, pp. 1221–1222
- Tom Collins, Daniel A. Abrams, Rohan Chandra, Christina Young, Andreas Arzt, and Vinod Menon. "Neural tracking of musical motives revealed by a combination of fMRI and music information retrieval techniques". In: *Proceedings of the International Conference on Music Perception and Cognition (ICMPC)*. Seoul, South Korea, 2014, p. 55
- Andreas Arzt, Cynthia C.S. Liem, and Gerhard Widmer. "A Tempo- and Transposition-invariant Piano Music Companion". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Late Breaking / Demo*. Taipeh, Taiwan, 2014
- Martin Gasser, Andreas Arzt, Thassilo Gadermaier, Maarten Grachten, and Gerhard Widmer. "Classical Music on the Web – User Interfaces and Data Representations". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Málaga, Spain, 2015, pp. 571–577
- Andreas Arzt, Werner Goebl, and Gerhard Widmer. "Flexible Score Following: The Piano Music Companion and Beyond". In: *Proceedings of the Vienna Talk on Musical Acoustics (VITA)*. Vienna, Austria, 2015, pp. 220–223
- Mark Melenhorst, Ron van der Sterren, Andreas Arzt, Agustin Martorell, and Cynthia C.S. Liem. "A Tablet App to Enrich the Live and Post-Live Experience of Classical Concerts". In: *Proceedings of the 3rd ACM International Workshop on Interactive Content Consumption*. Brussels, Belgium, 2015
- Matthias Dorfer, Andreas Arzt, and Gerhard Widmer. "Towards Score Following in Sheet Music Images". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. New York, USA, 2016, pp. 789–795

- Matthias Dorfer, Andreas Arzt, Sebastian Böck, Amaury Durand, and Gerhard Widmer. "Live Score Following on Sheet Music Images". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Late Breaking / Demo*. New York, USA, 2016

- Reinhard Sonnleitner, Andreas Arzt, and Gerhard Widmer. "Landmark-Based Audio Fingerprinting for DJ Mix Monitoring". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. New York, USA, 2016, pp. 185–191

- Rainer Kelz, Matthias Dorfer, Filip Korzeniowski, Sebastian Böck, Andreas Arzt, and Gerhard Widmer. "On the Potential of Simple Framewise Approaches to Piano Transcription". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. New York, USA, 2016, pp. 475–481

- Tom Collins, Andreas Arzt, Harald Frostel, and Gerhard Widmer. "Using Geometric Symbolic Fingerprinting to Discover Distinctive Patterns in Polyphonic Music Corpora". In: *Computational Music Analysis*. Ed. by David Meredith. Springer International Publishing, 2016, pp. 445–474

2

BACKGROUND AND RELATED WORK

Since I started working in the field of score following, I gradually got a clearer vision of how the technology could be used in a much broader sense than it is done traditionally. The classic motivation for score following has always been *automatic accompaniment*: a solo musician plays a piece of music, and the computer takes the role of all the other musicians, possibly a whole orchestra, and accompanies her in the right tempo. The main goal is to provide a smooth, human-like accompaniment that resembles a real-life performance. In the ideal case, the human soloist would not be able to tell if she is performing together with other humans or with a computer (see for example [39, 143] for recent discussions of this topic). The focus of my work in score following has been a different one, partly because of my own background — I am not a musician myself —, and partly because of the background of the department, which focusses more on the analysis of music, and not on the production.

The emphasis of the thesis is on the computer as a musical companion not in the sense of a participating musician, but more in the sense of a support system for performers and listeners, in various situations. Thus, the main task here is the synchronisation of *any kind* of additional information to the on-going live performance. In the simplest case this might be the sheet music, but it might also involve explanatory and/or educational textual information or visual information and artistic videos.

The aim of this chapter is to prepare the ground for the remaining chapters of the thesis, which describes my work in the field of music tracking. As the vision of a "Complete Classical Music Companion" to some extent defined the properties of the algorithms I was aiming for, I will first give a more detailed description of this vision and its ramifications (see Section 2.1). This vision relies on a relatively large collection of data like scores in symbolic form, sheet music, recordings of performances and annotations. In Section 2.2 a short description of the data, and the tools that were built to prepare the data is given. The software infrastructure that was used to implement the prototypes is described in Section 2.3. Then, in Section 2.4, the historic background of score following and related work relevant to the thesis are summarised.

2.1 A VISION OF A COMPLETE CLASSICAL MUSIC COMPANION

In this Section I describe the vision that drove my research during the work on the thesis. This is a long-term goal, and not something that can be fully achieved by the thesis.

The *Complete Classical Music Companion* is an application that is at one's fingertips anytime and anywhere, possibly as an application on a mobile device like a tablet computer. Whatever source of music, be it a live concert, a DVD, a video stream, or radio, whatever piece of classical music, for whatever instrumentation, and whoever the performers are, the companion will detect what it is listening to, inform about the written music, the historical context of the piece, famous interpretations, the specifics of the on-going (live) performance, and guide the user in the listening process.

At any point in time, the companion knows exactly what piece it is listening to, and where in the sheet music the performer(s) are. It will also detect sudden changes, like a switch to a performance of another piece. It can show visualisations synchronised to the music. In the simplest case it can show the sheet music itself, with a marker indicating the current position. While this is already helpful for listeners, more sophisticated visualisations and enrichments are possible, like showing information about the structure of the piece and the most important themes, and giving hints about what to listen for at specific moments. It might also tell you where you can acquire (additional) performances of this piece, or related pieces.

The companion is useful for musicians too. For example, during rehearsal the system can follow the performer(s) and show the sheet music accordingly, even if a section is repeated over and over, mistakes are made, and/or only parts of the score are played. It might even give you feedback about your piano playing, give hints what to improve — all done either fully automatically or also with the involvement of a community. Furthermore, the companion can be used on-stage for fully automatic page turning, either on a screen, or even via a mechanical page turning device that turns the sheet music page at the appropriate time.

The basis of an application as described above is a highly flexible and robust music identification and tracking algorithm. It has to be able to cope with a wide range of situations and contexts. It might be used in concert halls, or in the living room; it might have to identify and follow performances of soloists, quartets or even full symphonic orchestras; it might have to process performances by amateur musicians or professionals, it has to cope with different tempi, be it during rehearsal (like playing a part half the tempo) or on purpose as part of an expressive performance; generally it has to be able to adapt to many different playing styles. Thus, the main focus is on robustness and flexibility, while still providing sufficient tracking accuracy with

minimum need for adaptation, and no prior optimisation or training to a specific piece or performance.

This is very different to the classic motivation for score following: automatic accompaniment. There, the system typically is highly optimised to follow a specific performance. Every delayed triggering of accompaniment events (e.g. by 100 ms) is a big problem, and the triggering of events in the accompaniment ahead of time is even worse — effectively this would mean that the accompaniment drives the performance and the human performer has to react to it. Hence, this thesis is not concerned with problems specific to automatic accompaniment systems (e.g. focus on precision, human-computer interaction models), but with properties more important to the vision described above.

At the time of writing, there is no application available that would even come close to our vision of what a Complete Classical Music Companion should be. The thesis is concerned with the backbone of such an application: the *music identification and tracking technology* on which everything else is based. In this field, the thesis presents substantial progress, including prototypes that were shown live on stage at various occasions.

In particular, the thesis is concerned with

- a robust music tracking algorithm, including a simple tempo model and error recovery capabilities that works on solo piano music as well as on orchestral music,
- a multi-agent tracking approach that adds extra robustness,
- and a flexible tracking system that enables tracking on a complete database of scores for piano music. This is enabled by a music identification algorithm based on fingerprinting that, given a short snippet of audio, makes it possible to identify the underlying piece in a matter of seconds.

The focus on the backbone also means that it will not cover topics related to the extraction of information that is presented to a user in a possible application, i.e. it will not be concerned with such interesting and complicated topics as structure analysis, the identification of themes, harmonic analysis, automatic extraction of background information from the web, and their proper visualisation within a possible application. Hopefully, in the future the work presented in this thesis can be combined with all the efforts in the aforementioned (and other) areas, to make this vision come true.

2.2 DATA COLLECTION FOR THIS THESIS

Having access to appropriate, annotated data is paramount when working in the area of music tracking. To improve and evaluate music tracking and identification algorithms, audio recordings with ground

truth annotations are needed. In the ideal case, given a score the exact timing of each note in the respective audio recordings should be known. As this data is only available in some special circumstances (if the performance was recorded on a computer-controlled instrument), in many cases coarser (manual) annotations e.g. at the beat level or even only the downbeat level have to be used instead.

In the course of working on this thesis a lot of data of pieces and performances of classical music was collected and/or annotated. The data was organised with the help of a relational database. The database started out purely as a helper to evaluate the music tracking algorithms, but it turned into the backbone of many applications in the context of classical music at the department. Thus, the data preparation as well as the development of supporting tools were very much a joint effort with colleagues from the Department of Computational Perception[1] at the Johannes Kepler University and from the Intelligent Music Processing and Machine Learning Group[2] at the Austrian Research Institute for Artificial Intelligence. The most notable contributors are Harald Frostel, Sebastian Flossmann, Maarten Grachten, Martin Gasser, and Thassilo Gadermaier.

While we store a lot more information in the database, for the thesis three main parts are of great importance: the representation of musical scores, the representation of performances, and the mapping between information of these two different kinds of data. This enables the systematic evaluation of music tracking and also music identification algorithms.

2.2.1 *The Data*

For the thesis a variety of datasets of classical music was collected. Most of the data was prepared in the context of a number of projects at the department, while some of the data was annotated specifically for this thesis. For many years the focus of the department was on the analysis of piano music, which explains the large share of piano music in the data collection. In recent years, mainly due to the Phenicx[3] project, the focus slowly shifted to classical music in a broader sense, and there especially to orchestral music.

The database grew gradually, thus earlier papers presented in the thesis will only use a small share of the data that is available now. At the time of writing, the database contains 488 pieces of music with 1,878,898 notes and 4,450 pages of sheet music. To these pieces 1,682 audio recordings of performances are linked. For a part of these recordings, we know the exact timing of each note in the audio (in total 489,982 notes, all of which are connected to events in the scores).

1 http://www.cp.jku.at
2 http://www.ofai.at/research/impml/index.html
3 http://phenicx.upf.edu

For almost all the remaining pieces there are corresponding automatic alignments, some of them rigorously corrected manually (at the time of writing 624 alignments are stored). The most important datasets contained in the database are described below.

MOZART PIANO SONATAS This dataset consists of 13 sonatas by Wolfgang Amadeus MOZART, played by Roland BATIK. The performances were played on a BÖSENDORFER SE 290 computer-controlled piano. Parameters like timing and velocity of each played noted were saved and then synchronised to the original audio recordings (see [140] for details). Thus the exact timing of each note that was played is known, which is the key information for the evaluation of an alignment algorithm. In addition, deadpan MIDI files that can be used as score representations were produced and linked to the symbolic performance information.

CHOPIN ETUDE AND BALLADE This dataset consists of 2 sets of 22 piano recordings each of the Etude in E major Op.10 No. 3, bars 1–21, and the Ballade Op. 38, bars 1–45 by Frédéric CHOPIN. The excerpts were performed by skilled pianists on a BÖSENDORFER SE 290 computer-controlled piano (as above, the exact timing of each note was recorded). For more information on this dataset see [59] (in German) and [142].

CHOPIN PIANO WORKS This large dataset, also called the MAGALOFF CORPUS (see [50, 51]) consists of nearly the complete solo piano works by CHOPIN performed by Nikita MAGALOFF, especially known for his performances of the complete works of Frédéric CHOPIN. The data was collected during a series of 6 concerts in the MOZARTSAAL of the WIENER KONZERTHAUS, which took place between January 16 and May 17, 1989.

The concerts were performed on a BÖSENDORFER SE 290 computer-controlled piano, which means that, as for the MOZART PIANO SONATAS and for CHOPIN ETUDE AND BALLADE, we have very detailed information about the performances. We do not have access to the original audio recordings though, thus we replayed the symbolic performance data on a YAMAHA N2 hybrid piano and recorded the resulting performances. For details on the data collection, the preparation of the score and the alignment of the performance data to the scores see [50].

BEETHOVEN PIANO SONATAS This dataset consists of the complete piano sonatas by BEETHOVEN (32 in total). As well as the CHOPIN PIANO WORKS dataset, this dataset was compiled by Sebastian FLOSSMANN. It contains all the sheet music, the scores in symbolic form, and audio recordings (including symbolic information of key and pedal actions)

of 9 sonatas played by Clemens Zeilinger on a Boesendorfer CEUS computer-controlled piano.

ORCHESTRAL WORKS This data was annotated during the Phenicx project using the tools that were developed in the context of the thesis. The dataset comprises

- Ludwig van Beethoven's Symphonies No. 1, 3, 5, 6 and 9,
- Johannes Brahm's Symphony No. 3,
- Anton Bruckner's Symphonies No. 5 and 9,
- Gustav Mahler's Symphony No. 4,
- Richard Strauss' Alpensinfonie.

For these pieces symbolic scores, images of the sheet music, and performances are available. All the data was connected semi-automatically, and especially the alignment of the performances to the scores (at the downbeat level) involved rigorous manual corrections.

RACHMANINOFF PRELUDE This is a small dataset of 3 performances (by Vladimir Ashkenazy, Andrei Gavrilov and Howard Shelley) of the Prelude Op. 23 No. 5 in G minor by Sergei Rachmaninoff. These performances were manually annotated by the author. Especially during the early stages of developing the tracker this data played a vital role, as the piece contains two big tempo changes (fast – slow – fast) and thus was ideal for developing and testing robust tempo models.

MISCELLANEOUS PIECES The database also includes a number of pieces with varying amount of data (e.g. only the sheet music as an image and in symbolic form, but no performance data or additional annotations). Most of these were used at some point for live demonstrations, but not for formal evaluation purposes, because of lack of ground truth data.

2.2.2 *Score Representations*

The score is the main data needed to track a piece of music. We import the score either from MIDI or MusicXML and store it in a symbolic way. Since sometimes specific versions of a score are needed for tracking purposes, we introduced the notion of a "score variant", which is a version of the original score, but with a different structure. This is important as performers often do not follow the score exactly but for example play additional repetitions or leave out parts. Additionally, we also store information like time signatures or tempo annotations.

ScoreID	NoteID	ChannelID	MidiPitch	StartBeat	StopBeat
161	0	0	55	0.0	0.5
161	1	0	43	0.0	0.5
161	2	0	31	0.0	0.5
161	3	0	62	0.5	0.75
161	4	0	58	0.5	0.75
161	5	0	50	0.5	0.75
161	6	0	55	0.5	0.75
161	7	0	62	0.75	1.0
161	8	0	58	0.75	1.0
161	9	0	55	0.75	1.0
161	10	0	50	0.75	1.0
161	11	0	62	1.0	1.5
161	12	0	58	1.0	1.5
161	13	0	50	1.0	1.5
161	14	0	55	1.0	1.5

Figure 2.1: The minimal information needed to represent the notes within a score. All other information (e.g. the pitch in music notation, velocity information, trills, etc.) is optional and can be stored in additional tables.

Notes can be organised in channels, and an instrument can be assigned to a channel. Thus it is possible to (re-)create a MIDI file from the score, and synthesise it, which is vital for our tracking algorithm. See Figure 2.1 for an example of note entries in the database.

The general idea of music tracking is to follow a live performance played in any tempo — e.g. a beginner might play the piece in half the tempo that a professional performer might choose. Still, providing tracking algorithms with a reasonable base tempo will improve the tracking results drastically. In our datasets the tempo is generally set such that a synthesised version of the score will roughly have the same duration as a "mean" performance of the piece in question. As many kinds of classical music (and especially piano music from the romantic era, which makes up a big part of the data) can exhibit huge variations in tempo, this base tempo in many cases is still very different to the local tempo of a specific part. Tempo deviations up to a factor of two or three are quite common. For most pieces we only use a single tempo annotation for the whole piece, such that the duration of a MIDI file of the score roughly corresponds to the average duration of a real performance. Only in cases where a big change in tempo occurs, and this is also notated in the sheet music, we might add additional annotations.

In addition, for visualisation purposes (and for making sure visually that the tracking is working fine), we also store the sheet music in the form of images. Pixel coordinates in the images are then linked to either single notes or at least bars in the symbolic score data (see Figure 2.2).

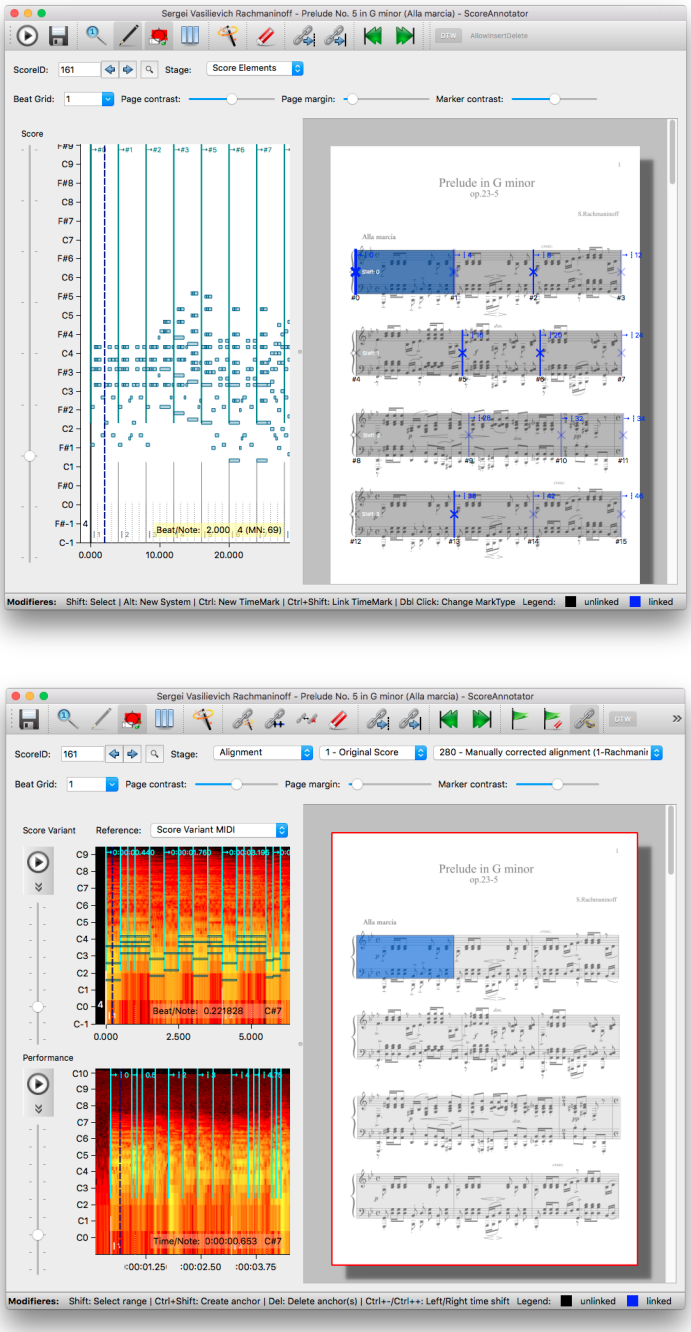


Figure 2.2: These screenshots show one of the annotation tools developed for the music database. For the thesis it was used to produce the mapping of time points in the symbolic score representation to areas in the score images (top), and to align a symbolic score and a performance (bottom).

PerformanceID	PerformanceAnnotationID	PerformanceNoteID	Onset	Offset	Velocity	MidiPitch
98	1	0	0.0	0.038	76	60
98	1	1	0.01	0.109	42	36
98	1	2	0.045	0.145	11	48
98	1	3	0.555	0.588	73	55
98	1	4	0.584	0.645	46	31
98	1	5	1.119	1.158	75	60
98	1	6	1.139	1.192	52	36
98	1	7	1.144	1.193	38	48
98	1	8	1.662	1.704	57	31
98	1	9	1.668	1.702	75	55
98	1	10	1.68	1.733	39	43
98	1	11	1.937	1.97	67	36
98	1	12	1.942	1.975	60	48
98	1	13	1.95	1.984	67	60
98	1	14	2.258	2.293	71	55
98	1	15	2.277	2.32	52	43

Figure 2.3: The minimal information needed to represent the symbolic note information of a performance. More information can be stored in additional tables.

2.2.3 *Performance Representations*

In the database, performances are stored and linked to the score representations (and the specific score variant, see above). Currently the database contains more than 1,600 performances. For many of these performances not only an audio recording, but also annotations are stored, which makes this data very useful for a range of tasks in music information retrieval (e.g. score following, beat tracking, structure analysis, etc.).

If available, not only the audio file, but also symbolic information about the performance (i.e. the exact timing of each note) is stored (see Figure 2.3).

2.2.4 *Mapping between Score and Performance Information*

To evaluate the algorithms presented in this thesis a mapping between time points in the score and time points in the performance is needed. Given this information, the output of the algorithms can be compared to the ground truth at the mapped time points.

The annotations were collected from different sources and in different levels of detail. For four important datasets (Mozart Sonatas, Chopin Etude and Ballade, Chopin Solo Piano and Beethoven Sonatas) we have exact information for each note in the score, in the performance, and their connection, i.e. which score note corresponds to each note played in the performance (see Figure 2.4). This also includes notes that should have been played, but were left out, and notes that were played although they do not occur in the score. This

ID	PerformanceID	PerformanceAnnotationID	PerformanceNoteID	ScoreID	ScoreVariantID	ScoreVariantNoteID
7045	98	1	0	3	11	2
7046	98	1	2	3	11	1
7047	98	1	1	3	11	0
7048	98	1	3	3	11	5
7049	98	1	4	3	11	3
7050	98	1	5	3	11	8
7051	98	1	7	3	11	7
7052	98	1	6	3	11	6
7053	98	1	9	3	11	11
7054	98	1	10	3	11	10
7055	98	1	8	3	11	9
7056	98	1	13	3	11	14
7057	98	1	12	3	11	13

Figure 2.4: Sample mappings of notes in a performance to notes in a score. This connection is possible because of very detailed performance data, recorded on a computer-controlled piano.

AlignmentBeatLevelID	ScoreBeat	PerformanceTime
26	-1.0	0.952941176470588
26	0.0	1.12941176470588
26	3.0	1.65882352941176
26	6.0	2.18823529411765
26	9.0	2.71764705882353
26	12.0	3.14352941176471
26	15.0	3.62823529411765
26	18.0	4.08588235294118
26	21.0	4.61529411764706
26	24.0	5.10470588235294
26	27.0	5.59705882352941
26	30.0	6.00352941176471
26	33.0	6.53294117647059
26	36.0	7.01117647058824
26	39.0	7.51176470588235
26	42.0	8.04117647058824
26	45.0	8.45058823529412
26	48.0	8.96

Figure 2.5: Sample mappings of time points in a performance to time points in a score. These alignments are typically prepared in a semi-automatic way via music synchronisation techniques and manual corrections.

kind of data is invaluable for detailed in-depth evaluations of music alignment and tracking algorithms.

If information in such a detail is not available, we have to rely on semi-automatically prepared ground truth data, at varying level of detail. In the annotation tool we support the generation of such alignment in a convenient way (see Figure 2.2). Generally, it is advisable to prepare alignments via a combination of automatic alignments and manual corrections, i.e. first compute an automatic alignment, manually correct obvious mistakes, re-compute the automatic alignment using these manual alignment points as fixed points, again manually correct mistakes, and so on. This process can be repeated until the desired level of detail is reached. In most cases it is very difficult to actually align performances to scores at the note level with sufficient confidence, thus normally we resort to either bar or beat-level for alignment experiments. An example can be seen in Figure 2.5.

2.3 IMPLEMENTATION FRAMEWORK

The algorithms and systems presented in the thesis were implemented in C++, based on a real-time audio processing framework called FLOWER (first used in [57]). This framework is developed and maintained by Martin GASSER[4] at the AUSTRIAN RESEARCH INSTITUTE FOR ARTIFICIAL INTELLIGENCE[5]. Similar frameworks include CLAM [1], MAX[6], PURE DATA[7] and AURA [37]. FLOWER is heavily inspired by CLAM [1], but restricts itself to an easy-to-use and embeddable software framework. It runs on WINDOWS, MACOS, LINUX, and on mobile devices (IOS and ANDROID).

FLOWER is a modular, multi-rate processing framework, based on the idea of a data processing graph consisting of relatively simple and reusable plug-ins. Plug-ins are written in C++ by implementing a simple plug-in API. For this thesis many plug-ins that were written by colleagues (or in co-operations with colleagues for other tasks) were reused — for example a mixer, a short-time Fourier transform, chroma features, and so on. In addition, a substantial number of plug-ins was developed by the author solely for this thesis. This includes the feature computation, music tracking, multi-agent tracking and fingerprinting plugins (see Chapters 3, 4 and 5).

The main tasks of the core framework are (1) allocation/deallocation and generally management of plug-ins, (2) the construction/destruction of processing graphs and (3) scheduling and dispatching of nodes in a processing graph. The scheduling algorithm is derived from work originally presented in the field of embedded real-time systems [2].

4 http://www.ofai.at/~martin.gasser/
5 http://www.ofai.at
6 https://cycling74.com/products/max/
7 https://puredata.info

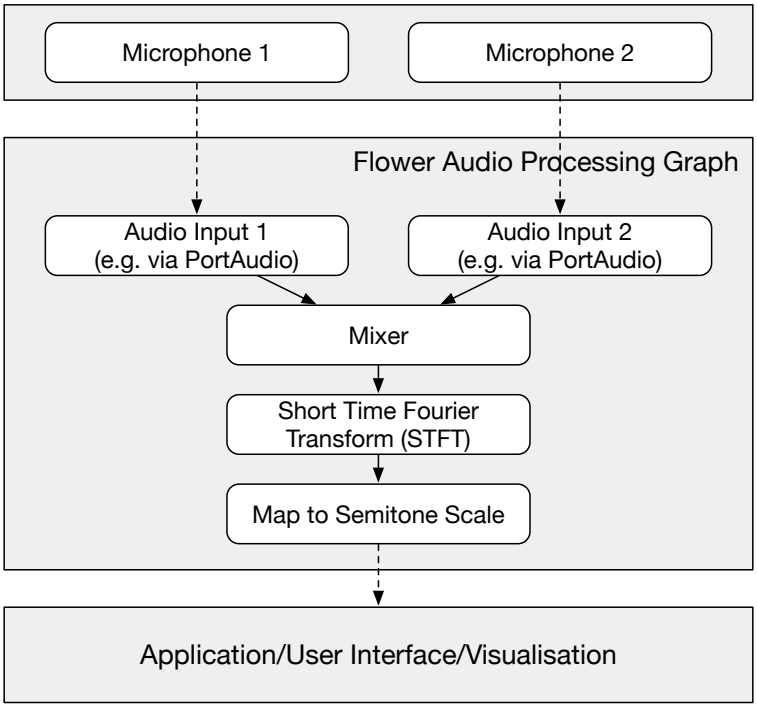


Figure 2.6: A simple Flower application. Two audio streams are first mixed into a mono signal, a short-time Fourier transform is computed, which then is mapped onto the semi-tone scale.

Furthermore, Flower supports parallel processing via the concept of zones (derived from Aura [37]) — sub-graphs that are executed by a dedicated thread. Communication between zones is implemented via lock-free FIFO queues.

To get a better idea, a simple graph composed of Flower plug-ins that computes a semitone spectrum can be seen in Figure 2.6. Here, two microphones record the signal. The data enters the network via two input plug-ins. These plug-ins are connected to a mixer that combines the stereo input into a mono signal. The mixer can also take care of the synchronisation of both input streams by blocking until there is data present in both inputs. The mixed signal is sent to another plug-in that performs a short-time Fourier transform. Then, the resulting spectrum is mapped onto the semitone scale. The data then can be handed over to an application (via a FIFO buffer), running in a separate thread to make sure that it does not block the audio processing network.

All the plug-ins are easily configurable and reusable components. For example, the STFT processing can be set up with different window sizes, hop sizes, window functions, and so on. The Flower framework takes care of the correct processing order of all the plug-ins in the processing graph.

Flower also supports multi-threading. Figure 2.7 shows an example of a graph that is very similar to the first one, except that both

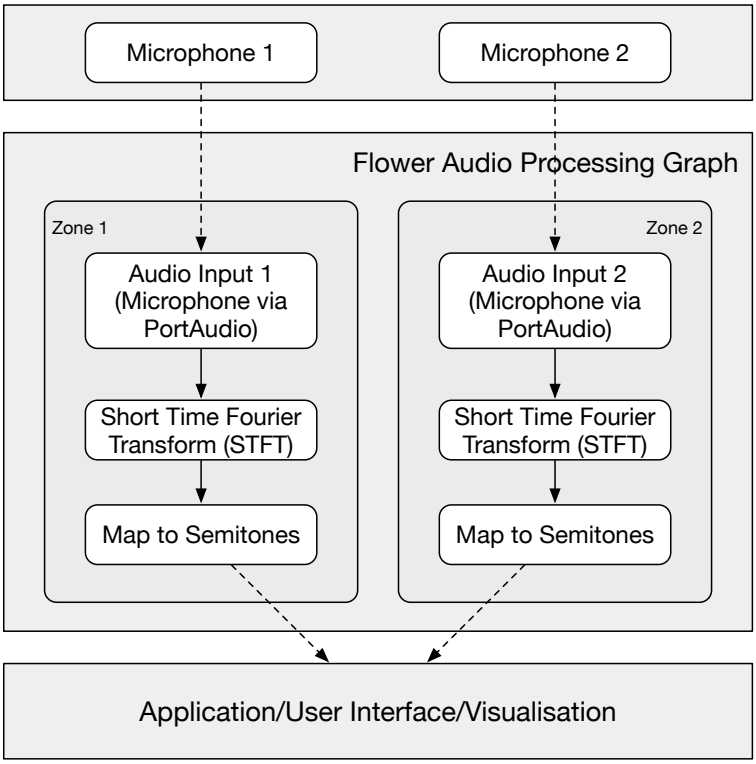


Figure 2.7: FLOWER provides a very simple API to build multi-threaded audio processing applications. The data of both microphones enters the processing graph synchronously in zone 1, and is processed in two separate zones (threads) in parallel.

input channels are processed separately (i.e. not mixed into one mono channel). In Zone 1, only the two input plug-ins remain, which take care of the microphone input. This zone is driven by the incoming audio. zone 2 and 3 are running in separate threads and process the two channels individually in parallel, as the data arrives. Between zone 1 and 2 and zone 1 and 3, small FIFO buffers take care that no data is lost. In the end, an application can request the data from both zones, and e.g. visualise the output. Again, FLOWER takes care of the correct execution order and also timing of the plug-ins within each zone, and the communication between the zones.

This framework simplified the implementation of the systems presented in this thesis immensely, as it takes care of many important aspects of the design of a real-time music processing system. Probably the most important feature of FLOWER for this thesis is its parallel processing capability. Especially for the design of the multi-agent and the any-time tracking systems, which involve multiple tracking algorithms running in parallel, this proved to be invaluable. As long as the application is easily separable into blocks that are supposed to run in parallel, building clean, multi-threaded applications in FLOWER is very convenient.

2.4 RELATED WORK

In this section a brief summary of research related to this thesis is given. In particular, the topic of music synchronisation — on-line as well as off-line. Furthermore, a summary of related work in the area of music identification will be given, as far as it is relevant for the content of this thesis, and approaches and available applications are presented that are related to the "vision" of this thesis.

2.4.1 *Music Tracking*

Score following algorithms that listen to a musical performance and at any time report the current position in the musical score, originated in the 1980s (see [36, 136]). These early approaches were not based on audio processing and could only follow certain (monophonic) instruments based on the live performance in symbolic form (e.g. by using the fingering information of the flute or key-presses on a keyboard). Most early algorithms were based on string matching techniques. The main motivation of these early approaches was automatic accompaniment (i.e. the automatic synchronisation of an accompaniment to a live performance of a soloist). Nowadays, there is still research in symbolic score following, with most approaches being based on stochastic models (see e.g. [126] for well-known, and [95, 96] for recent approaches). As this thesis is solely concerned with score following based on an audio signal as input, I will not go into more detail here.

With the arrival of more powerful computing hardware in the early 1990s the focus shifted towards score following based on the audio signal directly, albeit at first still limited to monophonic music. Pitch detectors were used to transcribe the live performance into symbolic form and used as input to a string matching algorithm. A widely used example of this approach is presented in [111].

In score following, there generally are two sources of uncertainty. First and foremost, an (expressive) performance of a piece of music will always differ in some ways from what is written in the sheet music. Performers will play some notes differently, be it on purpose or by mistake. Events that might be written as happening synchronously (like notes in a chord) will be played at least in a slightly asynchronous manner. The tempo will vary throughout the piece, and so on. Also, due to the nature of an audio signal, its interpretation (be it via a pitch detector or via some lower level features) will add further uncertainty. Thus, a natural choice to model the problem of score following is via statistical approaches.

The first stochastic model was presented in [65], while the formulation of the problem via a hidden Markov model (HMM) proved to be seminal (see [114]). Since then, a number of stochastic approaches, based on HMMs [29, 99, 100] and other state space models [46, 79],

conditional random fields [123, 144] and particle filtering [74, 86, 103] were presented. Arguably the most influential and most widely used systems are ANTESCOFO[8] and MUSICPLUSONE[9].

MUSICPLUSONE (see [115, 117, 118]) is based on a two-level architecture. The score following task is performed by an HMM. The output (i.e. begin times of notes) is fed to a switching Kalman filter that tries to model musical timing (and can be trained for a specific performer). The resulting system is able to anticipate actions by the performer and to control the accompaniment accordingly.

A related system is IRCAM's ANTESCOFO (see [30]). Here, the main difference is that score position and tempo are computed within a coupled stochastic model (a hidden hybrid Markov/semi Markov framework). The coupled tempo/audio inference model adaptively updates its duration models during the performance, which means that it does not need off-line training or parameter tweaking. Recent developments of this system include considerations about the internal modelling of time [33] and temporal accuracy regarding the underlying implementation of audio and event processing [43].

In recent years, some publications tackled very specific challenges using stochastic methods, like the improvement of the tracking results in the presence of heavy use of the sustain pedal when following a piano performance [79], or how to model (long) rests and tempo changes [74]. Some papers also considered the problem of structural changes to a piece, i.e. approaches were developed that can cope with a performer who spontaneously skips a repeat or adds a repeat (see [97, 104, 134]). This is a major topic in this thesis as well, and will be discussed in Chapter 5.

For the task of automatic accompaniment (which is outside of the scope of this thesis), a very important aspect is the way the computer and human performers interact during a performance. This goes far beyond score following in the traditional sense and includes anticipation of events, but also interaction with the human performer in terms of expressive timing and articulation (see [39, 143]).

Recent work has also seen deep learning applied to music tracking. The main advantage of the approach presented in [45] is that the tracking algorithm actually works directly on the sheet music, without any need for an intermediate representation. This is still very early work, but in long run it could make music tracking much more accessible for wider audiences: the musician could simply take a picture of the sheet music, and the algorithm is ready to follow the performance.

The main alternative to stochastic approaches is to use an on-line variant of the dynamic time warping algorithm (DTW). The thesis is heavily based on algorithms related to DTW, both off-line and on-line. Thus, these algorithms will be described in more detail in

8 http://repmus.ircam.fr/antescofo and http://www.antescofo.com
9 http://music.informatics.indiana.edu/~craphael/music_plus_one/

separate sections. The basic (off-line) DTW algorithm and its use in music synchronisation will be described in Section 2.4.2. In [41], an on-line version of DTW was presented, on which the on-line music tracker used in this thesis is based. This algorithm is described in Section 3.2 (Chapter 3 is solely concerned with improvements to this algorithm). On a side note, it can actually be shown that DTW is a special case of an HMM (see [32]), which is the main stochastic music synchronisation algorithm.

Comparisons between all these approaches are very difficult for multiple reasons. First and foremost, these approaches are optimised for specific kinds of music. A score following system for monophonic music or only slightly polyphonic music will be very different to an algorithm that is supposed to track orchestral music, and a tracker for piano music from the romantic era has to deal with different challenges than a tracker for 20th century classical music. In addition, also the task the music tracker is used for plays an important role. For example, for automatic accompaniment systems the main goal is to provide a smooth and natural accompaniment (thus in this case qualitative evaluations with human performers might be more important than quantitative experiments). To this end, the tracker has to be careful not to trigger events too early, as this would disturb the soloist. Optimisation for a specific piece, including adaption to a specific style of performance, and good detection (and anticipation) models for events are paramount. This is very different if the goal is a robust tracker for a wide range of classical music that works out of the box for many different styles of interpretation, and that is used to synchronise visual information to the live music. There, it is not as big of a problem if some of the information is shown a bit too early.

While there are still many open research questions, real-time score following is already used in real-world applications. The most prominent example is the abovementioned Antescofo system, which is actively used by professional musicians to synchronise a performance (mostly solo instruments or small ensembles) with computer realised performance elements — and was already featured in performances in venues like the Royal Albert Hall[10]. The main focus of this program is on modern classical music by composers like Boulez, Cage and Stockhausen, and thus actually is coupled with a synchronous programming language for musical composition.

There also have been two attempts of bringing music tracking technology to big concert halls. In [110], an application of music tracking in the context of the Philadelphia Orchestra is described, and in Chapter 4 of the thesis the demonstration of our music tracking system in the Concertgebouw in Amsterdam is presented. There, music tracking algorithms are used to present information (e.g. the

10 http://www.bbc.co.uk/events/evrmbp

sheet music and other visualisations as well as textual information) synchronised to the live performance to the audience.

Finally, TONARA[11] is an application based on music tracking, which runs on a tablet computer and is focused on persons studying the piano. The music tracker presented in this thesis also has been ported to iOS, but not been released as an application yet (see Chapter 6).

2.4.2 *Music Synchronisation with Dynamic Time Warping*

In Section 2.4.1 research on music tracking (which can be seen as online music synchronisation) was reviewed. In the off-line (non-causal) case, very similar techniques are applicable. Although stochastic approaches are a viable option (see for example [22, 116]), I will focus on the technique that was mainly used for this thesis: dynamic time warping (DTW).

Originally developed in the field of speech recognition [112], dynamic time warping (DTW) is a general technique to find an optimal alignment between two sequences. Other typical applications are gesture recognition [58] and handwriting recognition [119, 137]. The basic idea of DTW is to warp the input sequences in such a way that they match each other as closely as possible. To this day, it plays a very important role in music processing. It is used for such diverse tasks as *music synchronisation* (which is one of the main topics of this thesis), *cover song identification* [128] and *dance pattern recognition* [108]. In music processing, DTW is generally used to either 1) find corresponding time points in two sequences (e.g. the position in the score that corresponds to some time point in the audio), or 2) compute the similarity of two sequences (for example to find out if two audio files are based on the same piece of music). For a very detailed overview, both on DTW in general, including formal definitions, and on the use of DTW in music processing, I refer the reader to [88, 112].

The goal of DTW is to compare two sequences $U = (u_1, \ldots, u_m)$ and $V = (v_1, \ldots, v_n)$. In the general case, U and V are feature sequences, sampled at equidistant points in time. To compare two features, a local cost function is needed. By computing the cost function for each feature pair from U and V, a cost matrix d can be obtained. Given these costs, the goal of DTW is to find an alignment between U and V, with minimal overall costs. Intuitively, such an alignment is represented by a path P that follows "valleys" of low costs within the local cost matrix. The overall costs are the sum of all costs at points on this path.

Typically, several constraints are placed on the path P:

- P is bounded by the ends of both sequences
- P is monotonic

11 http://tonara.com

- P is continuous

Often, additional global path constraints are used with the goal to reduce complexity, such as the Sakoe-Chiba bound [124] which constrains the path to lie within a fixed distance of the diagonal or the Itakura parallelogram [69] which constrains the path to lie within a parallelogram around the diagonal of the matrix.

There are a number of local step constraints which can be used for the computation of the minimum cost path. The simplest and most common one (see [112] for more options) is defined by Eq. 2.1.

$$D(i,j) = d(i,j) + \min \left\{ \begin{array}{c} D(i,j-1) \\ D(i-1,j) \\ D(i-1,j-1) \end{array} \right\} \tag{2.1}$$

This recursion can be computed in quadratic time by linear programming. D is called the accumulated cost matrix, where $D(i,j)$ is the cost of the minimum cost path from $(1,1)$ to (i,j). $D(1,1) = d(1,1)$. In the end, the minimum cost path is extracted by following the recursion backwards from $D(m,n)$.

A common use case for DTW is as a measure of similarity of sequences, while discarding differences in timing. The distance of two sequences is directly given by the value $D(m,n)$.

For this thesis, the warping path itself is even more important: this path represents a mapping from time points in sequence U to time points in sequence V. If e.g. U represents an audio recording of a performance, and V a representation of a musical score, then DTW can be used to compute for *any point in time* of the audio recording the respective *position in the score*.

Generally, this process is called music alignment and exists in three closely related flavours. *Audio-to-audio alignment* is concerned with aligning two audio recordings of the same piece to each other. This is the most straightforward case, as both sequences are in the same domain, and the same kind of feature extraction can be applied. *Audio-to-Score alignment* is the task of aligning a recording of a performance (audio) to symbolic representation of the musical score (symbolic information). In this case either comparable features are computed directly from the symbolic information and the audio recording (see e.g. [38]), or the score is transfered into an audio representation (via a MIDI synthesiser), which means that again the same kind of feature computation can be applied to both sequences. In this way, the audio-to-score alignment task is effectively treated as an audio-to-audio alignment task, with some extra information available. Lastly, *Audio-to-Sheet-Music alignment* is concerned with aligning a recording of a performance to an image of the sheet music. There exists preliminary work that tries to do this directly, without explicitly transferring the

data of both sequences into a common domain (see [45]), but so far the main approaches are based on using optical music recognition algorithms to first transfer the image information into the symbolic domain, and to try to align this (noisy) information via typical audio-to-score alignment approaches (see e.g. [54, 77, 135]).

Thus, although music synchronisation spans multiple domains, the techniques and features that are used are often very similar. DTW was introduced to the problem of music synchronisation in [101] (in this case the task was audio-to-score alignment). This implementation follows exactly the standard definition of DTW, and used the simplest local continuity constraint. As features expected peaks in the spectrum are modelled from the pitches in the score and compared to the audio signal.

Features

For music synchronisation, the representation of the sequences to be aligned plays an important role. Ideally these features are both discriminative and robust against noise. A natural choice are features based on a series of short-time spectra, although early approaches tried to use more sparse mid-level features, to reduce the memory complexity [93, 101].

The most common low-level representation for music synchronisation are chroma features, also known as pitch class profiles (see e.g. [38, 68, 92]). In music, a pitch class is the set of all pitches that are a whole number of octaves apart. For example, the pitch class "C" consists of all the "Cs" in all octaves. The idea of chroma features is to aggregate spectral information that relates to pitch classes. Thus, a chroma feature is a 12 dimensional vector and indicates the strength of each of the 12 pitch classes in an analysis frame (computed e.g. via a short time Fourier transform or a constant Q transform). There are a number of different ways of computing chroma features (see e.g. [47, 82, 90, 106]) and variations based on this representation (see e.g. [49, 92]).

Other feature representations that have been tried for music synchronisation include mel frequency cepstral coefficients (MFCCs) (especially for performance-to-performance alignment, see e.g. [61]), onset-based features [42, 49], and spectral patterns [23]. Comparisons of features for music synchronisation can be found in [61, 70].

Complexity

Plain DTW has a time and space complexity that is proportional to the product of the lengths of the input sequences. This is problematic for processing music, especially for longer pieces. A common feature resolution for audio alignment is 50 Hz, thus computing alignments for recordings longer than a few minutes quickly becomes intractable.

A common method to speed up DTW and to reduce its memory usage is presented in [125]. The idea is to reduce the feature resolution and compute the optimal alignment on this coarse representation. Then, the resolution is increased step by step, while each time only the area close to the optimal alignment at the prior resolution level is considered. In [94] this approach was adapted for music alignment. Subsequently, this approach was refined to work in heavily memory-restricted environments (see [109], using ideas presented in [80]).

A different approach is presented in [72]. Based on the observation that one path in the matrix cannot cross another one, the amount of memory is significantly reduced. Via backtracking "fusion points" are found, which determine segments of the optimal path. After finding a fusion point it is sufficient to store the path found up to this point and to clear all the previous data. Then the calculation is continued, treating this fusion point as the beginning of the sequences. As this is an optimal approach the same results as with standard DTW are guaranteed.

This thesis is based on an on-line tracking algorithm that can also be used to compute off-line alignments (see [42], also summarised in Section 3.2). This approach has linear time and space complexity, based on a greedy forward path that is computed in a causal way. Assuming that the forward path always stays close to the actual minimal cost path, the algorithm will return the exact same results as standard DTW. The downside of this approach is the dependency on the forward path computation, which for in some extreme cases might fail, resulting in sub-optimal alignments.

Structure

In music, structure is a very important concept that can be analysed on multiple levels. For music synchronisation, structure is important in the sense that performers might deviate from the piece as it is notated in the sheet music and e.g. omit a repetition or insert additional parts. If a performance with such omissions or insertions is aligned to the score via a naive method, the process is confused by the missing or additional parts. The result is an alignment that for some parts of the piece might be correct, but for the omitted and inserted parts, as well as their surroundings, the output is unpredictable.

To manually take care of these differences in structure by adapting the score to a specific performance is a laborious task (although for our data collection (see Section 2.2) we actually do this). There have been a few approaches that try to automatically detect differences in structure. An early method based on a dynamic programming technique similar to DTW is presented in [89]. There, an intuitive way is described to automatically analyse a similarity matrix between two sequences (e.g. two different audio recordings of the same piece), and

to detect and extract paths that represent partial matches between the two sequences while ignoring gaps in between them.

An interesting approach is presented in [53], which is specifically designed for score representations that stem from automatic analysis of sheet music via optical music recognition (OMR). Amongst other shortcoming, OMR algorithms often do not reliably recognise repetition signs, volta brackets, coda markers, fine markers, and so on. In this paper an adapted version of DTW, called JumpDTW, was presented that explicitly allows for jumps to other blocks during the alignment process. This also includes adaptations that remove the constraints for the performance to have to start at the first bar, and to end in the last bar (e.g. when in the presence of a "da capo", the performance would actually end at the block marked by "fine"). A downside of this approach is that it assumes access to information about the borders of blocks — for the case discussed in the paper this information can be gained during the OMR step, but if the sequence is already present in symbolic form (e.g. as a MIDI file), this method is not directly applicable.

In [61] an approach is presented which does not rely on explicit information on where jumps might occur. Here, the dynamic time warping algorithm is replaced by a close relative, the Needleman-Wunsch algorithm, which allows for skips during the alignment process. Given a "complete" score, the algorithm can cope with performers skipping some parts. But the algorithm is unable to correctly align an inserted repetition, or cope with any other changes of the structure that involve jumps "backwards" in score time.

Improving the Alignment Accuracy

Some approaches have been presented that take a computed alignment and then try to refine the results. Examples include interpolation strategies, to increase the resolution of the alignment [48], the use of an onset detection function to explicitly look for the exact begin time around an aligned point [84], the use of image processing techniques to find exact begin times of notes in the spectrogram [85], and methods that try to determine anchor notes that are correctly aligned with a high probability, and revise the notes in-between them [98].

An alternative approach is presented in [139], which tries to improve the alignment process by jointly aligning multiple recordings of the same piece. This is implemented as an extension to the DTW algorithm, and increases the alignment robustness (especially for recordings that are difficult to align) and accuracy. In the thesis a similar approach is used to improve the robustness and the accuracy of on-line music tracking (see Chapter 4).

Conclusion

In the thesis, DTW is used both for off-line and on-line music synchronisation. Off-line DTW is mainly used to automatically preprocess data, extract tempo curves or transfer temporal annotations between performances. The music tracker used throughout this thesis is based on the above-mentioned on-line variant of DTW [42]. In Chapter 3, this algorithm is presented and a number of improvements are proposed. Then, Chapters 4 and 5 are based on this algorithm, and present a robust multi-agent music tracking approach and a very flexible music tracking system working on a database of scores.

2.4.3 *Music Identification*

Music identification is a very broad area (see e.g. [64] for an overview). In this thesis music identification plays an important role in the attempts to make music tracking more flexible. The classic approach to music tracking is to tell the algorithm first which piece the performer will play. Then, the performer plays this piece, starting in the beginning and following the structure of the piece until the end. As our goal was to build a more flexible system, which can follow the performer even when leaving out arbitrary parts or jumping to an entirely different piece, we needed a method to detect these actions as quickly as possible and query the corresponding data from the database in real-time.

A common identification problem is *cover song detection*, which has the goal to identify different versions of one and the same song in a collection of music (see e.g. [127]). This technique is aimed mostly at popular music. Algorithms performing this task have to cope with large variations in-between songs, like instrumentation, tempo, timing, style and structure. The more difficult problems of finding music based on the hummed melody (*query by humming*, see e.g. [105]) or based on a tapped version of the rhythm (*query by tapping*, see e.g. [107]) could be seen as special sub-problems of cover song detection.

A related problem is the fast and robust identification of *exact replicas* of audio recordings, possibly distorted in some ways (e.g., compression artefacts, noise). For this problem, which is commonly called *audio fingerprinting*, industry-strength algorithms exist and are in every day use in commercial applications (e.g. the well-known service Shazam[12], which is based on [138]). For an overview on early work regarding audio fingerprinting see [21]. Recent work (see e.g. [18, 113, 129, 132]) mainly focuses on making fingerprinting algorithms more robust to transformation in the time-scale (replay speed of the audio) and the frequency scale (transpositions).

12 http://www.shazam.com

In the thesis, we are interested in a problem that to some extent is related to both cover song identification and fingerprinting: *score identification*. Given a *performance* of a piece of music, we are interested in automatically identifying the underlying *score* on which the performance is based. More concretely, we are interested in an algorithm that, given a short snippet of audio material (e.g. the most recent five seconds of an on-going live performance), finds the corresponding score in a database, and computes the exact position in that score.

This is a multi-modal problem, and could either be solved in the symbolic domain (score), the audio domain (performance), or via some suitable mid-level representation to which both kinds of data can be transferred. As the transformation of a symbolic score into the audio domain is relatively straightforward — a synthesiser can be used to render a "machine-like" low-quality audio version of the score —, it seems natural to try to solve this task via *audio matching* (see Section 2.4.2 above). A typical implementation is described in [92]. The downsides of approaches in the audio domain are that relatively large query sizes are needed (around 20 seconds) and that it is computationally expensive, despite attempts to improve on this via sophisticated indexing strategies (see [62, 75]).

Instead, we followed a different approach and tried to solve this problem in the symbolic domain. We developed a new method that relies on a (noisy) transcription of the performance and then uses a tempo- and transposition-invariant fingerprinting algorithm to retrieve the correct piece and score position in a matter of seconds (see Chapter 5).

2.4.4 *Efforts related to a Complete Classical Music Companion*

Currently, there is no system available that covers all the aspects that were mentioned in the vision of a *Complete Classical Music Companion*. Nonetheless, there already are interesting prototypes and applications available that cover a few of the ideas included in our vision. A combination of these approaches with the music identification and tracking technologies described in the thesis would seem to be a very promising endeavour.

The simplest example are sheet music viewers on tablet computers, like the HENLE LIBRARY[13] or the BÄRENREITER STUDY SCORE READER[14]. Actually, it would be fairly easy to turn these applications into more "intelligent" companions by integrating music identification and tracking technologies.

13 http://www.henle-library.com/en/

14 https://www.baerenreiter.com/programm/digitale-medien/baerenreiter-study-score-reader-app/

The PHENICX project[15] (see [60]) was an EU FP7 project that ran from 2013 to 2016. Its main aim was to "innovate the classical music experience", with a strong focus on the concert experience. I was involved in this project, with my main task being to build a reliable live music tracking algorithm for complex orchestral music. This tracking algorithm was used to synchronise information for the audience to the live music, like the sheet music, artistic videos and notes prepared by a musicologist. The most important results of the project are shown collectively in an online demonstrator[16], which can be seen as interactive and dynamic program notes that guide, educate and entertain the concertgoer before, during and after the concert. A related application called LIVENOTE[17] was developed by the PHILADELPHIA ORCHESTRA, which is solely focused on the live performance. The music tracking system behind this application is described in [110] (which is very similar to the basic tracking algorithm this thesis is based on — see [41] and Section 3.2).

Both of these projects demand a lot of (time-consuming) manual preparation, and are focused on specific performances. This also means that both systems would fail if something unexpected were to happen, e.g. a repetition that is left out. Despite their inflexibility, the way in which information is presented to the user is very much in the spirit of the vision of a Complete Classical Music Companion.

There also exist a number of applications and projects that provide a similar experience off-line, i.e. for recorded performances. TOUCHPRESS[18] has published polished applications that allow the user to explore and enjoy a number of famous pieces of classical music (e.g. BEETHOVEN'S 9TH or the LISZT Sonata in B minor) and get to know famous performers (e.g. the JUILLIARD STRING QUARTET). These applications offer features like synchronised scores, seamless switching between performances, helpful visualisations, background information, and so on.

A similar application is published by the ROYAL CONCERTGEBOUW ORCHESTRA for the IPAD. The RCO EDITIONS[19] is an interactive magazine with new editions released six times per year. It contains concert recordings, expert commentary, articles, and graphics. Some of the concert recordings are enriched in the sense that multi-modal information is synchronised to the recording. We were involved in the synchronisation process, which is done semi-automatically using our off-line audio-to-score alignment algorithms. This on-going collaboration is also an outcome of the PHENICX project.

The SYNCPLAYER [52, 76] is an off-line system for multi-modal music presentation (e.g. the audio, sheet music, lyrics), including a basic

15 http://phenicx.upf.edu
16 http://beta.phenicx.com
17 http://livenote.philorch.org
18 http://www.touchpress.com
19 https://www.concertgebouworkest.nl/en/all-editions

mechanism for music identification. Later on, in the context of the PROBADO[20] project, this system was extended for big collections of musical data (see [34, 35]). A more recent example is the project FREISCHÜTZ DIGITAL[21] [122]. This is an extensive collection of data, tools, and visualisations, including efforts to present multi-modal information about music in a unified and informative way. In this project, different media types like images of historic and modern sheet music, images of historic texts of the libretti, and audio recordings are linked to each other and made browsable on an on-line platform. Available demonstrators also include a network representation of topics and concepts of the piece, a music player with synchronised sheet music, and tools with which it is possible to explore the microphone setup during recording, and to focus on specific instruments.

20 http://www.probado.de/en_home.html
21 http://freischuetz-digital.de

Part II

CONTRIBUTIONS OF THE THESIS

3

IMPROVEMENTS TO MUSIC TRACKING VIA ON-LINE TIME WARPING

PRELIMINARIES This chapter describes work on music tracking via on-line time warping, which was published between 2008 and 2012. In particular, the chapter is based on the papers

[15] 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)*. Patras, Greece, 2008, pp. 241–245

[11] Andreas Arzt and Gerhard Widmer. "Simple Tempo Models for Real-time Music Tracking". In: *Proceedings of the Sound and Music Computing Conference (SMC)*. Barcelona, Spain, 2010

[16] Andreas Arzt, Gerhard Widmer, and Simon Dixon. "Adaptive Distance Normalization for Real-time Music Tracking". In: *Proceedings of the European Signal Processing Conference (EUSIPCO)*. Bucharest, Romania, 2012, pp. 2689–2693

At this early stage of my research the data collection was still relatively small. Consequently, the algorithms were evaluated on a growing data collection, which explains the need to describe the data for each of the papers individually.

In this chapter the algorithms and their evaluations are described as they were presented in the original papers. While this is not mentioned in the papers explicitly, they are actually based on two different implementations of the music tracker. In the beginning, up until 2011, the algorithms were implemented in JAVA. Due to concerns regarding the real-time capability, I re-implemented the complete tracker in C++ using the FLOWER framework (see Section 2.3). Basically, this can be seen as a fresh start. During this re-write many improvements and optimisations regarding memory usage and runtime were made. This lead to slight inconsistencies between the older JAVA version and the new C++ implementation, while the general performance of the system stayed the same. Footnotes will be used to comment on these inconsistencies, and the chapter will close with a few remarks regarding developments and changes that occurred after these papers were published.

CONTRIBUTIONS Unless stated otherwise, the research in this chapter was conducted by the author of the thesis. The original OLTW

algorithm was provided by Simon Dixon. This algorithm was extended and later re-implemented from scratch by the author of the thesis. Both Gerhard Widmer and Simon Dixon were involved in the writing process of the respective papers.

3.1 INTRODUCTION

The goal of real-time music tracking is to follow a musical performance on-line and at any time report the current position in the score. For this task many different approaches have been proposed, which are summarised in Section 2.4.1. In the thesis, this task is tackled via on-line audio-to-audio alignment. That is, rather than trying to transcribe the incoming audio stream into discrete notes and aligning the transcription to the score, a MIDI version of the given score is first converted into a sound file by using a software synthesiser. The result is a "machine-like", low-quality rendition of the piece with constant tempo. Due to the information stored in the MIDI file, the time of every event (e.g. note onsets) in this rendition is known. Then, an on-line music alignment algorithm is used to align the live performance to the "score audio".

In the thesis, a music tracking algorithm based on on-line time warping [41] is used, which will be summarised in Section 3.2. In its original form, this algorithm becomes unstable in the case of mistakes of the performer, increased pedal usage and polyphonic mixes of many different instruments. Moreover, it is very sensitive to differences in tempo between the performance and the score. In extreme cases it would get lost completely. Thus, the main lines of work here were to

- generally improve the robustness and the capability to recover after tracking errors (Section 3.3),
- introduce a tempo model that makes the algorithm applicable in the case of tempo differences between the score and the performance, both globally and locally (Section 3.4), and
- improve the features on which the tracking is based (Section 3.5).

The remainder of this thesis is based on the outcome of this chapter: a robust music tracking algorithm that reliably tracks a wide range of classical music.

3.2 THE ORIGINAL ON-LINE TIME WARPING ALGORITHM

In [41], an algorithm for the online alignment of two audio streams based on dynamic time warping (DTW) was presented (see Section 2.4.2 for a description of the DTW algorithm and its uses in music

processing). The important differences between this algorithm and standard DTW are linear time and space complexity, and the fact that the alignment is computed incrementally.

Formally, this on-line time warping algorithm (OLTW) works as follows. Given two sequences $U = (u_1, \ldots, u_m)$ and $V = (v_1, \ldots, v_n)$, an alignment between U and V is a path $P = (P_1, \ldots, P_l)$ (through a cost matrix) where each P_k is an ordered pair (i_k, j_k) such that $(i, j) \in P$ means that the points u_i and v_j are aligned. P is constrained to be monotonic and continuous. An $m \times n$ matrix represents a cost matrix $d(i, j)$ which assigns costs to the alignment of each pair (u_i, v_j). The cost of a path P is the sum of the local alignment costs along the path. The $m \times n$ accumulated cost matrix D is computed using the recursion:

$$D(i,j) = \min \left\{ \begin{array}{c} D(i,j-1) + d(i,j)w_a \\ D(i-1,j) + d(i,j)w_b \\ D(i-1,j-1) + d(i,j)w_c \end{array} \right\} \tag{3.1}$$

The weights w_a, w_b and w_c can be set to give some positive or negative bias towards specific step directions. Here, it is set to $w_a = w_b = 1$ and $w_c = 2$, which ensures that there is no bias for either step type (i.e. a diagonal step has the same weight as one horizontal and one vertical step combined). $D(i, j)$ is the cost of the minimum cost path from $(1, 1)$ to (i, j), $D(1, 1) = d(1, 1)$.

So far, this formulation is equivalent to the classic DTW algorithm. The difference is that based on this formulation, the on-line time warping algorithm computes a quasi-optimal solution (a *"forward path"*) by incrementally constructing this cost matrix in real time. During the initial phase, as long as fewer than $s = 500$ elements of each series have been processed, columns and rows are calculated alternately and the path follows the diagonal of the matrix. Calculating a row (column) means incrementing the pointer to the next element of the respective time series, calculating the new local distances, and updating the accumulated cost matrix D by using Formula 3.1.

After this initial phase the number of cells to be calculated is given by a search width parameter $c = 500$, e.g. for a new column i the local distances $d(i, j - (c - 1)), d(i, j - (c - 2)), \ldots, d(i, j)$ are calculated, where j is the index of the current row. The calculation of the minimum cost paths using Formula 3.1 is restricted to using only calculated cells. In this way, only a sub-band of the cost matrix of constant width is computed (see Figure 3.1), which reduces time and space complexity from quadratic to linear.

To decide if a row or a column should be computed (i.e. which of the two time series to advance), for each cell in the current row j and column i the normalised minimum path cost is found. The normalisation is done via the number of steps it takes from $(1, 1)$ to

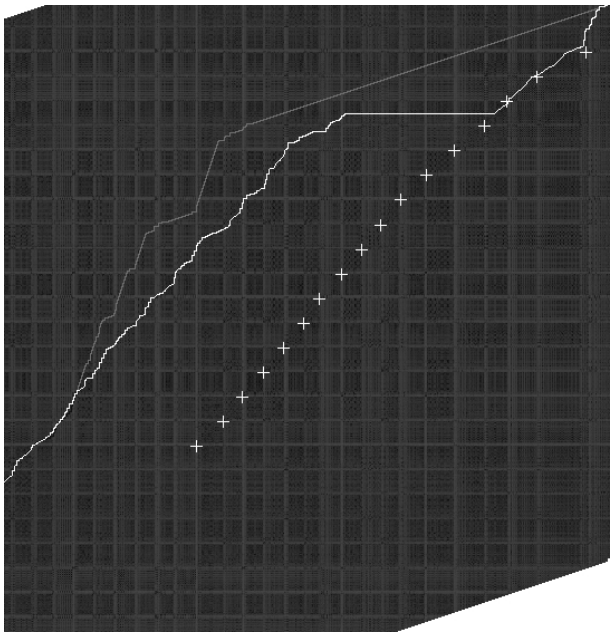


Figure 3.1: Part of a cost matrix (note that not the complete matrix, but only a sub-band around the diagonal is computed). Note the areas in the upper left and lower right corners, illustrating the constrained path computation around the forward path. This particular situation shows the system reacting to an additional bar of music (not present in the score) erroneously played by the pianist. The live performance is on the x axis, the score representation on the y axis. Crosses show the correct note onsets according to the score. The grey path is calculated by Dixon's original OLTW algorithm, the white path is the result of the tracker including the backward-forward strategy. Note how the algorithm effectively "waits" for the pianist (the horizontal segment) after having noticed the error. This is made possible by the backward-forward approach (see text).

the respective cell (i.e. the sum of the indices of the cell in question), to ensure that shorter paths are not necessarily preferred over longer ones. If the minimum cost occurs in the current position (i, j) both a new row and column are calculated. If this occurs elsewhere in row j a new row is calculated and if this occurs elsewhere in column i a new column is calculated. If one time series has been incremented more than $MaxRunCount = 3$ times, the other series is incremented. This embodies the assumption that a given performance will not be more than 3 times faster or slower than the reference score, and prevents the alignment algorithm from "running away" too far.

In addition to the forward computation, it is also possible to compute a "backward path" via following the recursion in Formula 3.1 backwards. This is equivalent to the normal DTW path computation (limited to the computed cells).

The audio streams to be aligned are represented as sequences of analysis frames, using a low-level spectral representation computed via a windowed FFT of the signal with a hamming window of size 46ms and a hop size of 20ms. The data is mapped into 84 frequency bins which are spread linearly up to 370Hz and logarithmically above, with semitone spacing, and then normalised to sum up to 1. In order to emphasise *note onsets* — the most important indicators of musical timing — only the increase in energy in each bin relative to the previous frame is stored. The cost of aligning two such 84-dimensional vectors is computed as the Manhattan distance between the two vectors, normalised by a logarithmically weighted sum of their norms[1].

3.3 IMPROVEMENT 1: RECONSIDERING PAST DECISIONS

The OLTW algorithm has a tendency to "run away" in the score, and then has a hard time to recover. It is unable to actually wait for the performer, due to constraints placed on the path computation. Similarly, if the algorithm were to fall behind, it would have no way to catch up quickly. These constraints are necessary mainly because the path computation happens in a greedy way. At each step the decision to take a step in the score or to stay at the current position is entirely based on local information, and only indirectly (via the accumulated costs) on past information. To weaken these constraints, we introduced a way of explicitly re-considering past decisions [15]. This approach uses the present hypothesis plus the information from which it was constructed, in order to re-check past decisions and then, in turn, uses the revised decisions to improve the present hypothesis.

More precisely, the method works as follows: After every two frames of the live input a smoothed backward path is computed, starting at the current position (i, j) of the forward path. By following this path b steps backwards on the y-axis (the score) one gets a new point which lies with a high probability nearer to the globally optimal alignment than the corresponding point of the forward path (because this backward computation takes into account information from the "future" that was not available when computing the original forward path). Starting at this new point another forward path is computed until a border of the current matrix (either column i or row j) is reached. If this new path ends in (i, j) again, this can be seen as a confirmation of the current position. If the path ends in a column $k < i$, new rows are calculated until the current column i is reached again. If the path ends in a row $l < j$, the calculation of new rows is stopped until the current row j is reached. We call this method the "backward-forward" approach. In our specific implementation, two different backtracking lengths are used: after 4 short backtrackings of length $b = 10$ a longer one of length $b = 50$ is performed.

1 Both [41] and [15] incorrectly stated that the Euclidean distance was used.

ID	COMPOSER	PIECE NAME	# PERF.	DATA TYPE
CB	Chopin	Ballade Op. 38 No. 1 (exc.)	22	Match
CE	Chopin	Etude Op. 10 No. 3 (exc.)	22	Match

Table 3.1: The data set used for the evaluation of our real-time music tracking system.

The main effect of this strategy is *increased robustness against tempo changes* and *improved error tolerance*. If there are extreme tempo changes in the performance, or the performer makes large errors — plays wrong notes and repeats or omits a whole bar — the backward-forward strategy permits the system to correct the error faster by waiting for the musician or jumping forward in the score. A situation where the system "waits" for the performer to catch up after a serious mistake is shown in Figure 3.1.

In addition to this extension, a few minor details of the algorithm were changed. We set $w_a = w_b = 1.3$ and $w_c = 2$, which introduces some bias towards diagonal steps. Together with the backward-forward approach, this stabilised the algorithm such that we could set $MaxRunCount = 6$. Furthermore, we found using such a long initialisation phase to be unnecessary, and set $s = 50$, which effectively starts the actual tracking after slightly more than one second[2].

3.3.1 *Evaluation*

A quantitative evaluation requires correct reference alignments. For practical reasons, the systematic experiments were performed off-line. The results are the same as for on-line alignment, except for a small latency that would occur in real-time processing. In the following we refer to Dixon's original OLTW algorithm, which serves as a reference, as D, and to the new algorithm that uses the backward-forward idea as A1.

The algorithms were evaluated on two sets of 22 piano recordings (see Table 3.1). The audio recordings were aligned to synthesised score audio files with constant tempo. As the computer-monitored piano that was used for the recordings also stores the precise note onset times ("match files"), the alignment error could then be calculated. For more information on the data see Section 2.2.

As Tables 3.2 and 3.3 show, A1 outperforms D by far. The mixture of the backward-forward algorithm and the slightly loosened constraints generally improves the tracking accuracy. The only minor problem

2 In [15] we also introduced a very naive on-line method for actually searching explicitly for note onsets, which worked well on this set of test data. On more complicated music (e.g. with piano music which makes more use of the sustain pedal, higher degrees of polyphony or a mix of different instruments) this approach failed to work. Hence, it was discarded in later papers (and will not be discussed here).

	BALLADE		ETUDE	
ERR. (SEC)	D	A1	D	A1
≤ 0.05	**52.1%**	48.0%	**53.8%**	53.2%
≤ 0.10	70.9%	**72.6%**	68.3%	**75.7%**
≤ 0.15	77.2%	**80.0%**	74.2%	**82.0%**
≤ 0.20	83.3%	**86.6%**	80.1%	**87.7%**
≤ 0.25	85.8%	**89.1%**	83.2%	**90.8%**
≤ 0.30	88.1%	**91.2%**	86.0%	**93.4%**
≤ 0.35	89.3%	**92.7%**	87.0%	**94.6%**
≤ 0.40	90.9%	**93.7%**	89.2%	**96.0%**
≤ 0.45	91.6%	**94.4%**	90.1%	**96.7%**
≤ 0.50	92.5%	**94.9%**	91.1%	**97.6%**
≤ 1.0	97.0%	**97.3%**	96.5%	**99.5%**

Table 3.2: Real-time alignment results shown as cumulative frequencies of errors of matching pairs of notes for the original OLTW algorithm (D) and the improved version (A1) on the Etude and the Ballade. The results are based on the alignment of 3564 notes in the Etude and 4422 notes in the Ballade.

ID	D	A1
CB	85.8%	**89.1%**
CE	83.2%	**90.8%**

Table 3.3: Real-time alignment results. Here, the percentage of notes is shown that was aligned with an error smaller or equal 250 ms (for convenience — evaluations in the thesis will often use this evaluation measure).

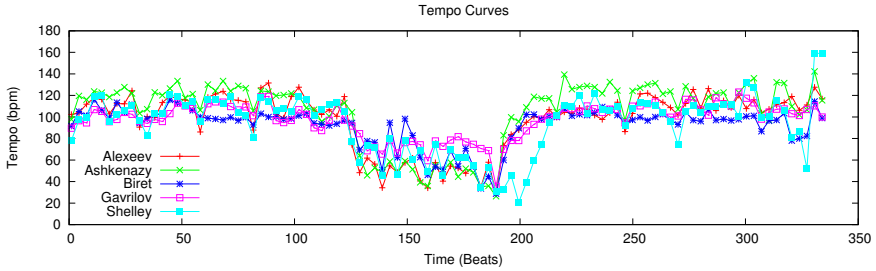


Figure 3.2: Tempo curves (at the level of quarter notes) automatically extracted from 5 different commercial recordings of the Prelude Op. 23 No. 5 by RACHMANINOFF. Note especially the slow-down around beat 130 and the subsequent speed-up around beat 190 and the generally big differences in timing between the performances.

is that the proposed method tends to "overshoot" a little bit, which explains the results in the first line in Table 3.3.

3.4 IMPROVEMENT 2: TEMPO MODELS

Tempo models are one of the most important building blocks of a music tracking algorithm. After all, the task of a music tracker can be seen as adapting the tempo of the score representation to the live performance. The better the tempo model, and thus the prediction of future events, the easier this task becomes. In the ideal case, the tempo model predicts the times of future events almost perfectly, and the tracking algorithm only has to check for their occurrence and adjust the timing slightly to minor prediction errors. The problem for a music tracking algorithm is that for most kinds of (classical) music, tempo is one of the main parameters a performer will vary to achieve an expressive performance of a piece. As can be seen in Figure 3.2, these changes are considerable within a performance, and also between performances of the same piece. Thus, predicting the timing, even of the next few events, is a very difficult task, and sometimes impossible without additional information.

Our work on tempo models [11] was motivated by the observation that the closer the tempo of the score representation is to the actual live performance, the better the tracking works. Thus, what we actually tried was not to design a system which predicts events and checks for their occurrence, but a simple extension to the on-line time warping algorithm that adapts the score feature sequence on the fly to match the tempo of the performance as closely as possible. We came up with two versions of tempo models, one being very simple and only based on the current alignment context, and the second one taking into account external information via automatically processed performances of the same piece.

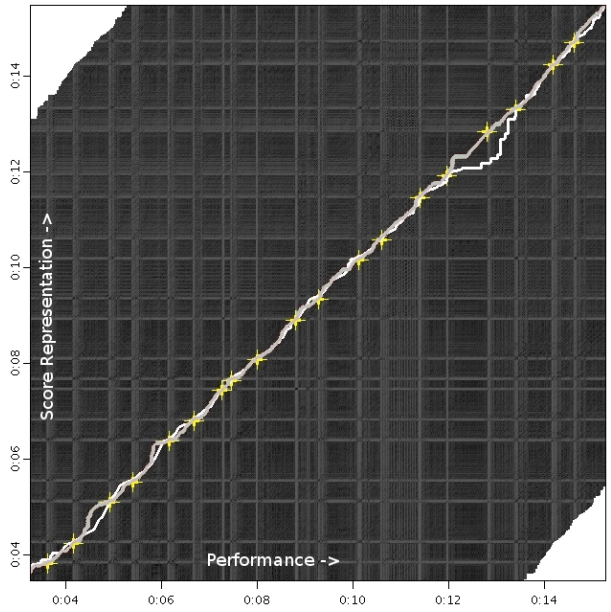


Figure 3.3: Illustration of the music tracking algorithm, showing the iteratively computed forward path (white), the much more accurate backward path (grey, also catching the one onset that the forward path misaligned), and the correct note onsets (yellow crosses, annotated beforehand). In the background the local alignment costs for all pairs of cells are displayed. Also note the white areas in the upper left and lower right corners, illustrating the constrained path computation around the forward path.

3.4.1 *Computation of the Current Tempo*

The computation of the current tempo of the performance (relative to the score representation) is based on a constantly updated backward path starting in the current position of the forward calculation. As the backward path, in contrast to the forward path, which has to make its decisions on-line, has perfect information about the performance — at least up to the current position in the performance —, it is much more accurate and reliable than the forward path (see Figure 3.3).

Intuitively, the slope of a backward path represents the relative tempo differences between the score representation and the actual performance. Given a perfect alignment, the slope between the last two onsets would give a very good estimation of the current tempo. But as the correctness of the alignment of these last onsets generally is quite uncertain, one has to discard the last few onsets and use a larger window over more note onsets to come up with a reliable tempo estimation.

In particular, our tempo computation algorithm uses a method described in [91]. It is based on a rectified version of the backward alignment path, where the path between note onsets is discarded and the onsets (known from the score representation) are instead linearly

connected. In this way, possible instabilities of the alignment path between onsets (as, e.g. between the 2^{nd} and 3^{rd} onset in the lower left corner in Figure 3.3) are smoothed away.

After computing this path, the $n = 20$ most recent note onsets which lie at least one second in the past are selected, and the local tempo for each onset is computed by considering the slope of the rectified path in a window with size three seconds centred on the onset. This results in a list v of length n of relative tempo deviations from the score representation. Finally, an estimate of the current relative tempo t is computed using Equation 3.2, which emphasises more recent tempo developments while not discarding older tempo information completely, for robustness considerations.

$$t = \frac{\sum_{i=1}^{n}(v_i * i)}{\sum_{i=1}^{n} i} \tag{3.2}$$

Of course, due to the simplicity of the procedure and especially the fact that only information older than one second is used, this tempo estimation method can recognise tempo changes only with some delay. However, the computation is very fast, which is important for real-time applications, and it proved very useful for the task we have in mind.

3.4.2 *Feeding Tempo Information to the OLTW algorithm*

Based on the observation that both the alignment accuracy and the robustness directly depend on the similarity between the tempo of the performance and the score representation, we now use the current tempo estimate to alter the score representation on the fly, stretching or compressing it to match the tempo of the performance as closely as possible. This is done by altering the sequence of feature vectors representing the score audio. The relative tempo is directly used as the probability to compress or extend the sequence by either adding new vectors or removing vectors.

More precisely, after every incoming frame from the live performance, and before the actual path computation, the current relative tempo t is computed as given above, where $t = 1$ means that the live performance and the score representation currently are in the exact same tempo and $t > 1$ means that the performance is faster than the score representation. The current position in the score p is given by the forward path and thus coincides with the index of the last processed frame of the score representation. If a newly computed random number r between 0 and 1 is larger than t (or $\frac{1}{t}$ if $t > 1$) an alteration step takes place. If $t > 1$, a feature vector is removed from the score representation by replacing $p + 1$ and $p + 2$ with a mean vector of $p + 1$ and $p + 2$. And if $t < 1$, a new feature vector, computed as the mean of p and $p + 1$ is inserted next into the sequence between

p and $p+1$. As our system is based on features emphasising note onsets, score feature vectors representing onsets (which are known from the score) are not duplicated, as more (and wrong) onsets would be introduced to the score representation. In such cases the alteration process is postponed until the next frame. Furthermore, to avoid that the system could get stuck at one frame, alterations may take place at most three times in a row.

3.4.3 *"Learning" Tempo Deviations From Different Performers*

As will be shown later in Section 3.4.4, the introduction of this very simple tempo model — simply using the current estimated tempo to stretch/compress the reference score audio — already leads to considerably improved tracking results. But especially at phrase boundaries with huge changes in tempo (e.g. a slow-down or a speed-up by a factor of two is not uncommon, see also Figure 3.2) the above-mentioned delay in the recognition of tempo changes still results in large alignment errors. Furthermore, such tempo changes are very hard to catch instantly, even with more reactive tempo models. To cope with this problem we came up with an automatic and very general way to provide the system with information about possible ways in which a performer might shape the tempo of the piece.

First we extract tempo curves from various different performances (audio recordings) of the piece in question. Again, as for the real-time tempo estimation, this is done completely automatically using the method described in [91] (see Section 3.4.1), but as the whole performance is known beforehand and the tempo analysis can be done off-line there is now no need for further smoothing of the tempo computation. These tempo curves (see Figure 3.2) are directly imported into our real-time tracking system.

We use this additional information during the tracking process to compute a tempo estimate based not only on tracking information about the last couple of seconds, but also on similarities to other known performances.

More precisely, as before, after every iteration of the path computation algorithm the list v containing tempo information at note onsets is updated based on the backward path and the above-mentioned local tempo computation method. But now the tempo curve of the live performance over the last $w = 50$ onsets, again located at least one second in the past, is compared to the previously stored tempo curves at the same position. To do this all n tempo curves are first normalised to represent the same mean tempo over these w onsets as the live performance. The Euclidean distances between the curve of the live performance and the stored curves are computed. These distances are inverted and normalised to sum up to 1, thus now representing the similarity to the tempo curve of the live performance.

Based on the stored tempo curves our system can now estimate the tempo at the current position. As the current position should be somewhere between the last aligned onset o_j and the onset o_{j+1} to be aligned next, we compute the current tempo t according to Equation 3.3, where t_{i,o_j} and $t_{i,o_{j+1}}$ represent the (scaled) tempo information of curve i at onset o_j and o_{j+1} respectively, and s_i is the similarity value of tempo curve i to the current performance context.

$$t = \frac{\sum_{i=1}^{n}[(t_{i,o_j} + t_{i,o_{j+1}})s_i]}{2} \tag{3.3}$$

Intuitively, the tempo is estimated as the mean of the tempo estimates at these two onsets, which in turn are computed as a weighted sum of the (scaled) tempi in the stored performance curves, with each curve contributing according to its local similarity to the current performance. Please note that this approach somewhat differs from typical ways of training a score follower to follow a particular performance. We are not feeding the system with rehearsal data by a particular musician, but with many different ways of how to perform the piece in question, as the analysed performances may be by different performers and differ heavily in their interpretation style. The system then decides on-line at every iteration how to weigh the curves, effectively selecting a mixture of the curves which represents the current performance best.

3.4.4 *Evaluation*

The accuracy of our system was thoroughly tested on various pieces of music (see Table 3.4), with very well known musicians like Vladimir Horowitz, Vladimir Ashkenazy and Daniel Barenboim amongst the performers. While we currently focus on classical piano music, to show the independence of specific instruments we also tested our system on an Oboe Sonata by Mozart and the 1^{st} Movement of the 5^{th} Symphony by Beethoven.

As for the evaluation reference alignments of the performances are needed, Table 3.4 also indicates how the ground truth data was prepared. For CB and CE we have access to very accurate data about every note onset, as these were recorded on a computer-monitored grand piano. For the performances of the three movements of Mozart's Sonata KV279 (MS) the evaluation is based on exact information about every beat time, which was manually compiled. The evaluation of the other pieces is based on off-line alignments produced by our system, which are in general much more precise than on-line alignments. We are well aware that this information is not guaranteed to be entirely accurate, but we manually checked the alignments for obvious errors and are quite confident that the results based on these alignments are reasonable, especially as evaluations of CB, CE and MS based on these

ID	COMPOSER	PIECE NAME	# PERF.	DATA TYPE
BF	Bach	Fugue BMV847	7	Alignment
BS	Beethoven	5^{th} Sym., 1^{st} Mov,	5	Alignment
CB	Chopin	Ballade Op. 38 No. 1 (exc.)	22	Match
CE	Chopin	Etude Op. 10 No. 3 (exc.)	22	Match
CW	Chopin	Waltz Op. 34 No. 1	8	Alignment
MO1	Mozart	Oboe Quartet KV370 Mov. 1	5	Alignment
MO3	Mozart	Oboe Quartet KV370 Mov. 3	5	Alignment
MS1	Mozart	Sonata KV279 Mov. 1	5	Beats
MS2	Mozart	Sonata KV279 Mov. 2	5	Beats
MS3	Mozart	Sonata KV279 Mov. 3	5	Beats
RP	Rachmaninoff	Prelude Op. 23 No. 5	5	Alignment
SI	Schubert	Impromptu D935 No. 2	12	Alignment

Table 3.4: The data set used for the evaluation of our real-time tracking system. Note that besides piano performances this also includes two movements of an oboe quartet (oboe, violin, viola, cello) and a symphony for orchestra.

alignments led to very similar numbers compared to the evaluation on the correct reference alignments.

For all pieces we used audio files synthesised from publicly available "flat" MIDI files with fixed tempo as score representation, only the MIDI representing the Beethoven Symphony contained sparse tempo annotations.

The evaluation took the form of a leave one out cross-validation. Every performance in our data set (Table 3.4) was aligned with three algorithms: the system as introduced in Section 3.3 above; the system including the simple tempo model (Section 3.4.1); and the tempo model that has access to a set of possible performance strategies (Section 3.4.3). For the latter, all recordings pertaining to the given piece were used except, of course, for the performance currently being aligned. The result, for each performance and each algorithm, is a set of events with detection times in milliseconds.

The evaluation measure is defined after [31]. The percentage of correctly aligned notes with respect to some threshold is the main performance measure for a real-time music tracking system. We follow [31] and accept a note as correctly aligned if the computed time differs from the actual onset time by not more than 250 ms.

Table 3.5 summarises the results. Clearly, both tempo models lead to large improvements in tracking accuracy for pieces played with a lot of expressive freedom, especially for the SCHUBERT Impromptu (SI), the RACHMANINOFF Prelude (RP) and the CHOPIN Waltz (CW). Nonetheless these kinds of music still pose a great challenge to real-time tracking systems. As the results for the BEETHOVEN Symphony (BS) show, our system can also cope quite well with orchestral music

ID	NONE	SIMPLE	LEARNED
BF	97.3%	97.8%	**98.1%**
BS	84.1%	85.0%	**86.1%**
CB	89.1%	90.0%	**90.1%**
CE	90.8%	91.6%	**93.2%**
CW	72.4%	83.7%	**88.1%**
MO1	85.0%	93.0%	**93.1%**
MO3	81.6%	92.1%	**93.0%**
MS1	96.4%	**96.7%**	96.2%
MS2	80.2%	86.2%	**88.7%**
MS3	96.1%	96.7%	**97.9%**
RP	68.2%	82.9%	**85.2%**
SI	58.2%	76.4%	**79.9%**

Table 3.5: Real-time alignment results for all three evaluated systems. NONE is the system without a tempo model, SIMPLE uses a tempo model solely based on data of the current live performance, and LEARNED uses a tempo model using data from other performances of the same piece.

and does not depend on specific instruments. This is also supported by the results on the Oboe Quartet (MO). As expected, the results for pieces with less extreme tempo deviations were improved to a much smaller extent. Further investigation showed that as intended, the "learned" tempo curves guide the alignment path more accurately and more reactively during huge tempo changes (i.e. at phrase boundaries).

3.5 IMPROVEMENT 3: BETTER FEATURES FOR MUSIC TRACKING

Over the years there have been a number of studies on features for music alignment via dynamic time warping, comparing and evaluating features like (compressed versions of) the plain spectrogram, mappings to different kinds of logarithmic scales like the semitone scale, chromagrams, onset-emphasised features, and Mel-frequency cepstral coefficients (see Section 2.4.2).

Some of these comparisons were focused on performance to performance alignment, others on performance to score alignment. This makes a difference, as some features, like MFCCs, tend to work better on signals that are more "similar", like two performances on the same piece (compared to a performance and a synthesised version of a score). Other features, like chroma features, generalise better and are suitable for aligning a real performance to a score synthesised in a "mechanical" way with some sound font.

This section is based on [16]. The goal of this paper was to try to find a suitable feature representation for our music tracker. Note that

this paper is already based on the new C++ implementation, which accounts for slight inconsistencies with respect to the previous two sections.

The basic setup is as follows. In order to align a live performance to a score, suitable feature representations are needed. As mentioned above, we actually treat this task as an audio-to-audio alignment problem, with additional knowledge about the score audio file (e.g. the exact timing of each note onset). Thus, we first convert a MIDI version of the score into a sound file using a software synthesiser. Then, both streams to be aligned are represented as sequences of analysis frames, computed via a short-time Fourier transform (STFT) of the signal with a hamming window of size 92ms and a hop size of 23ms[3]. Then each frame resulting from the STFT is mapped onto a musically more meaningful representation better suited for the task of audio alignment.

A natural musically motivated choice for representing the tonal/harmonic content is to map the data into frequency bins with semitone spacing. As we would like this representation to be invariant to dynamic variations, we normalise each vector to sum up to 1. We will refer to this representation as *Normalised Semitone features* (NS).

For features representing the note onsets we again map the STFT data to the semitone scale but now only store the increase in energy in each bin relative to the previous frame. As described in [49] for chroma onset features we now first take a suitable logarithm of the values in the vector, motivated by the logarithmic sensation of sound in humans, and then normalise each vector by the maximum norm in a fixed window around this vector. While in the original paper this window is centred on the vector to be normalised, we had to shift the window to only use data up to the current vector to make it computable on-line. We will refer to this representation as *Locally Adaptive Normalised Semitone Onset features* (LNSO).

Most recent audio alignment systems are based on different variants of chroma-based harmonic features (e.g. [68, 94, 98]). In general chroma vectors consist of 12 elements per time frame, corresponding to pitch classes; their values are computed by mapping the frequency bins of the STFT to the 12 pitch classes and summing up the energies. There also exist more sophisticated ways of computing chroma features, and for this paper we are in fact using the method described in [47] (for a collection of various chroma implementation see [90]). Again each vector finally is normalised to sum up to 1 to make it invariant to dynamic variations (*Normalised Chroma features* (NC)).

It is also possible to compute chroma onset features by a mapping of the LNSO features described earlier to the chroma representation, as recently done in [49]. The authors refer to this representation as

3 Note that in the previous two sections a window size of 46 ms and a hop size of 20 ms was used

ID	COMPOSER	PIECE NAME	# PERF.	DATA TYPE
CE	Chopin	Etude Op. 10 No. 3 (excerpt)	22	Match
CB	Chopin	Ballade Op. 38 No. 1 (excerpt)	22	Match
MS	Mozart	1^{st} Mov. of Sonatas KV279, KV280, KV281, KV282, KV283, KV284, KV330, KV331, KV332, KV333, KV457, KV475, KV533	1	Match
RP	Rachmaninoff	Prelude Op. 23 No. 5	3	Manual

Table 3.6: The data set used for the evaluation of the different features for music tracking.

Locally Adaptive Normalised Chroma Onset features (LNCO). Additionally, as this introduces a desirable property for off-line (= backward) audio alignment they also introduce an extra temporal decay, resulting in *Decaying* LNCO features (DLNCO), which should not be favourable for the on-line task in question. Nonetheless we will also evaluate the effect of this decay in our on-line (= forward) audio alignment algorithm.

Finally, having defined a number of possible feature representations, a function determining the alignment cost of two frames (distance between 2 frames) is needed. Here, we will use the L_1 distance:

$$d(I, J) = \sum_{k=1}^{n} |I_k - J_k| \tag{3.4}$$

where I and J are either semitone or chroma frames.

3.5.1 *Adaptive Distance Normalisation*

One problem with using the aforementioned features based on onset information (LNSO, LNCO and DLNCO) directly is that they are only normalised relative to their local context within their audio streams. There can be huge differences in onset strength between the two audio streams, especially when the score audio stream is generated from a deadpan MIDI file without loudness information (= with the same velocity for every note). In contrast to this, there are all kinds of variations in dynamics in the live performance. Take for example a piano performance in which the performer emphasises the melody while playing the accompaniment very softly, or blurs the onsets by using the sustain pedal, as is often the case. Then the correct alignment of the louder melody notes would lead to minimal distances, but there

would occur substantial alignment costs for each of the accompanying notes, possibly leading to alignment errors.

A simple solution to this problem is to compute a normalised distance d_n of two frames by dividing d by the sum of their L_1-norms.

$$d_n(I,J) = \frac{d(I,J)}{|I|_1 + |J|_1} \tag{3.5}$$

Evidently this simple approach has its drawbacks. It introduces a lot of noise to the distance matrix by heavily up-scaling small distances between frames with low energy in them. Still this normalisation step greatly improves the alignment results and actually makes the semitone onset features useable — alignments based on the unnormalised distances got lost most of the time.

To get rid of this up-scaling effect, we introduce a weight describing the "onsetness" of the two frames involved. When chosen correctly, this weight can be seen as a dampening factor: avoiding the scale-up effect for small numbers while still normalising the distance when enough energy is involved. Recall that due to the locally adaptive normalisation step the L_1-norm of each frame is a measure of its onsetness, ranging from 0 to 1. Thus the mean of the L_1-norms of two frames is a natural measure of their combined onsetness. Based on experimental results we chose to apply a suitable function to this value, leading to the desired dampening effect and resulting in the normalised and weighted distance d_{nw}.

$$d_{nw}(I,J) = d_n(I,J) * \sqrt[4]{\frac{|I|_1 + |J|_1}{2}} \tag{3.6}$$

It is important to note that the main point of the formula above is not the application of exactly the 4^{th} root — this function merely gave the best results in our evaluation, but only by a very small margin. We achieved very similar results with other functions (the square root, cubic root or also a function based on the logarithm), as long as the function fulfilled the intended dampening task described above.

Results

The performance of each feature configuration was thoroughly tested on various pieces of piano music (see Table 3.6). This table also indicates how the ground truth data was prepared, where "match" means that we have access to very accurate data about every note onset, as these were recorded on a computer-monitored grand piano. For more information on the datasets see Section 2.2. The Tables 3.7 and 3.8 show the percentage of correctly aligned notes for the different configurations mentioned in the text. A note is accepted as correctly aligned if the computed time differs from the actual onset time not

ID	NS^{dn}	NC^{dn}
CE	82.01%	**87.78%**
CB	75.04%	**79.97%**
MS	90.19%	**91.20%**
RP	75.61%	**81.45%**

Table 3.7: Real-time alignment results for the harmonic features, i.e. the *Normalised Semitone* (NS) and the *Normalised Chroma* (NC) features.

ID	CE	CB	MS	RP
LNSO^{d}	9.68%	5.80%	1.20%	2.28%
LNSO^{dn}	91.93%	91.79%	**97.41%**	78.71%
LNSO^{dnw}	**96.09%**	**95.61%**	93.76%	**86.25%**
LNCO^{d}	43.66%	28.92%	18.23%	20.77%
LNCO^{dn}	92.78%	86.74%	90.28%	42.67%
LNCO^{dnw}	**95.85%**	**93.72%**	**90.91%**	**71.77%**
DLNCO^{d}	77.01%	65.97%	48.18%	**33.73%**
DLNCO^{dn}	89.46%	79.05%	79.14%	2.91%
DLNCO^{dnw}	**93.26%**	**85.27%**	**85.18%**	23.56%

Table 3.8: Real-time alignment results for the onset features for the *Locally Adaptive Normalised Semitone Onset* (LNSO), *Locally Adaptive Normalised Chroma Onset* (LNSO), and the *Decaying Locally Adaptive Chroma Onset* features, combined with different normalisation methods.

more than 250 ms (see also [31] for more details on the evaluation of real-time audio-to-score alignment systems).

Regarding the harmonic features, the suitability of the NC features for audio alignment purposes is well established (see e.g. [70]) and again confirmed by our experiments (see Table 3.7). In contrast to that the performance of the LNSO features[4], which work far better than the related LNCO features, may come as a bit of a surprise (see Table 3.8). It seems that when it comes to modelling onsets the mapping to the chroma scale destroys crucial information (the absolute height of the onsets). As also shown in this table, the normalisation step for onset features is indispensable. While with the unnormalised distances the tracker in many cases gets completely lost or at least produces a lot of errors, both normalised versions lead to robust and accurate alignments, the weighted one even clearly outperforming the NC features.

Interestingly, the basic DLNCO features outperformed the LNSO and LNCO features. But while the normalisation process also has a positive influence on the DLNCO features in general (interestingly, the Rachmaninoff Prelude is an exception) they do not benefit to the same extent. When comparing the evaluation runs using the distance normalisation process, the LNSO features are clearly preferable at least for on-line trackers such as ours.

3.5.2 *Mixing Chroma and Onset Information*

Having evaluated their accuracy when used individually, it seems reasonable to try to combine the two presented feature types (harmonic and onset) into one feature set, something which has already been suggested by [49] in a different (off-line) alignment setting. There the authors mix NC features with the DLNCO features described above by simply computing 2 distinct local cost matrices, and finally summing up both matrices to get a distance measure which accounts for both types of information.

We will now combine the best features of both classes according to our evaluation runs in the same fashion. For this we picked the NC features (see Table 3.7) and the LNSO features with the distance normalization procedure described above (see Table 3.8). Thus as the total distance d_{tot} of 2 frames we get:

$$d_{tot}(I,J) = d_{nw}^{LNSO}(I,J) + d_{n}^{NC}(I,J) \tag{3.7}$$

4 Conceptually, the configuration LNSO^{dnw} is very similar to the features and the distance calculation that were used in Sections 3.3 and 3.4 above. The main difference are the use of triangular filters instead of simple mappings and the use of the root function instead of a function based on the logarithm. Further differences are the use of a larger window size (92 ms vs. 46 ms) and a different hop size (23 ms vs20 ms).

ID	NC^{dn}	LNSO^{dnw}	$\text{NC}^{dn}+\text{LNSO}^{dnw}$
CE	87.78%	96.09%	**96.13%**
CB	79.97%	95.61%	**96.38%**
MS	91.20%	93.76%	**98.20%**
RP	81.45%	86.25%	**93.73%**

Table 3.9: Real-time alignment results for the single best features *Normalised Chroma* (NC) and *Locally Adaptive Normalised Semitone Onset* (LNSO), as well as their combination.

ERR. (SEC)	NC^{dn}	LNSO^{dnw}	$\text{NC}^{dn}+\text{LNSO}^{dnw}$
≤ 0.05	35.53%	44.69%	**46.24%**
≤ 0.10	67.64%	86.95%	**89.00%**
≤ 0.15	78.23%	90.49%	**94.13%**
≤ 0.20	83.75%	92.42%	**96.03%**
≤ 0.25	87.12%	93.32%	**96.93%**
≤ 0.30	89.75%	94.05%	**97.57%**
≤ 0.35	91.65%	94.58%	**97.96%**
≤ 0.40	92.91%	94.99%	**98.28%**
≤ 0.45	93.98%	95.36%	**98.47%**
≤ 0.50	94.77%	95.66%	**98.71%**
≤ 1.0	98.16%	97.44%	**99.59%**

Table 3.10: Real-time alignment results for the single best features and their combination on the whole test set (79,178 notes in total) shown as cumulative frequencies of errors of matching pairs of notes.

We also experimented with an additional weighting of the distances such that in case of onsets $d_{nw}^{LNSO}(I,J)$ is dominant and $d_{n}^{NC}(I,J)$ otherwise, but — despite some promising results with some of the weaker features — we did not achieve further improvements by this strategy.

Results

As expected, these combined features, which complement one another in a natural way, lead to a substantial increase in alignment accuracy and robustness compared to their individual results (see Table 3.9). The combination outperformed each configuration with single features we tested for this paper.

Table 3.10 gives a more in-depth comparison of the combined features to the individual ones based on the cumulative frequency of errors. It again confirms that in this natural combination the NC features mainly add robustness (i.e. these features show fewer extreme errors larger than 1 second than the LNSO features). On the other hand the LNSO features greatly improve the accuracy (e.g. 86.95%

of the notes are aligned with an error smaller or equal 0.1 seconds, compared to 67.64% when using the NC features).

3.6 CONCLUSIONS

Although the original algorithm [41] was presented more than 10 years ago, and even the improvements date back around 5 years, the method described in this section still is a competitive music tracking algorithm. The main strength of the algorithm is its versatility and robustness. Given a good representation of the score (in the sense of "sounding similar to a real performance"), this algorithm works well even for very complex symphonic music (see Chapter 4). On the downside, it is not ideally suited for sparse, monophonic music as it has no model of duration other than the implicit encodings in the score representation. It has no real expectation of note lengths and thus a hard time at coping with pauses. Casually speaking, the more note onsets are being played, the easier is the task of this tracking algorithm.

3.6.1 *Current State of the Music Tracker*

Over the years, the tracking algorithm has changed in a few minor points. The features stayed basically the same, although we now do zero-padding of the hamming window to a total size of 184 ms. Then, in our current implementation we actually perform the backward-forward procedure after every new frame of the live performance. Furthermore, we compute the backward path as a smoothed version of two kinds of backward paths — one starting in the cell that currently holds the minimum costs, another one starting in the current position hypothesis of the tracker (which are not necessarily the same) — which are further smoothed over time. Furthermore, instead of on absolute times in seconds we base the backtracking procedure on the note context: each time the backward path is followed backwards in time for 50 note onsets, and then the path computation is restarted from there. Regarding the tempo models, we only changed the number of onsets to use for the tempo computation from 20 to 30.

3.7 PROTOTYPICAL IMPLEMENTATION

Figure 3.4 shows a conceptual sketch of the music tracker. As described in this chapter, the algorithm takes a live audio stream as input and aligns it to an internal representation of the score (which is represented as an audio stream too). The tracker is based on on-line time warping, which has been provided with a tempo model. The output is the current position in the score, which enabled a number of applications (e.g. automatic page turning for musicians).

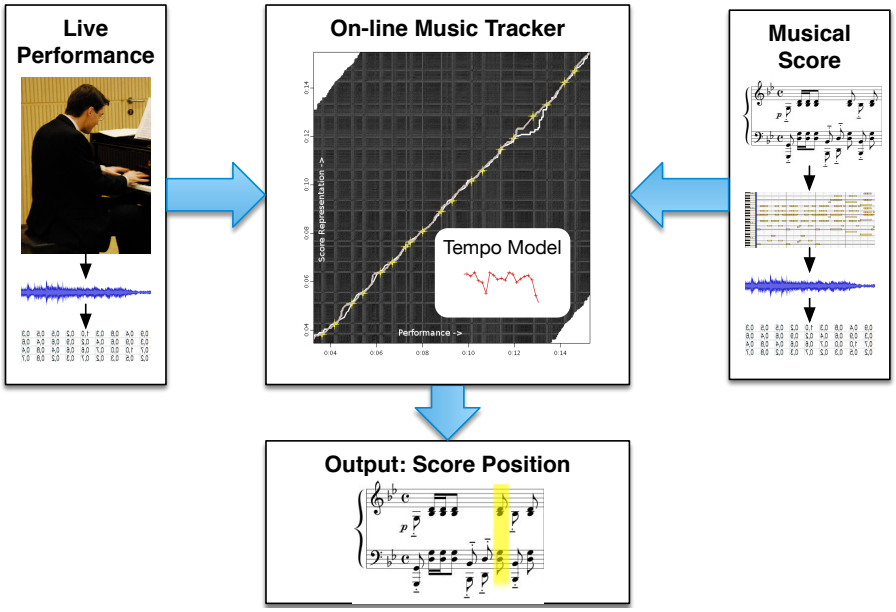


Figure 3.4: A conceptual sketch of the music tracker. Picture of Werner GOEBL (c) Clemens CHMELAR

Figure 3.5 shows the current state of the music tracker as it is implemented in the modular FLOWER framework (see Section 2.3). The live audio stream is captured via a microphone. Raw features are computed via a number of processings (IFGRAM, IFEXTRACTOR, CHROMAIF and STMSP). These basically are chroma features and spectral features mapped to the semitone scale. They are then further processed by MUSICTRACKINGFEATURES, computing NC^{dn} and $LNSO^{dnw}$ features as described in Section 3.5 above. These are handed over to the MUSIC-TRACKER processing, which implements the tracking algorithm. This processing also reads in all the necessary data (the score representation in symbolic and in feature form) before the actual tracking begins. The output of the MUSIC TRACKER processing is an estimate of the current position in the score.

Besides the evaluations presented in this section, we also demonstrated this algorithm live in front of audiences at a number of occasions (even in combination with an automatic page turning device). We mainly demonstrated this algorithm with piano music, although it is perfectly capable of following other kinds of classical music. A list of public showings, including a video, can be found in Chapter 6.

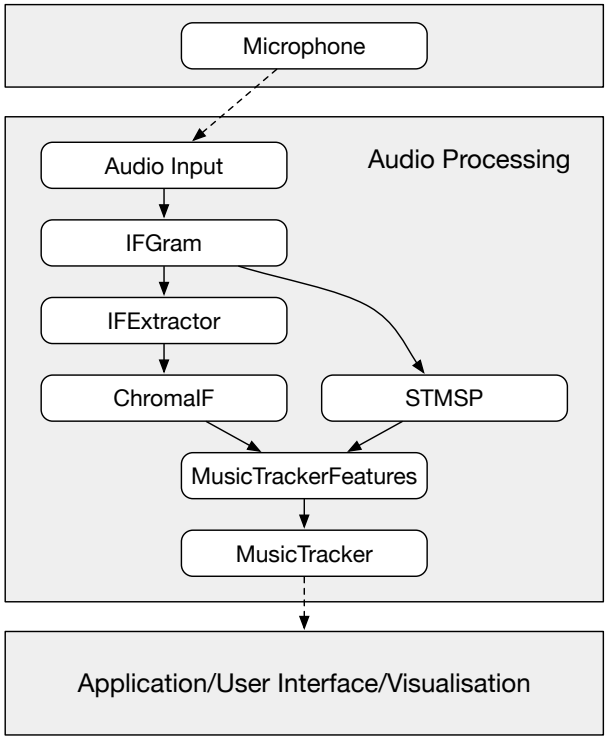


Figure 3.5: The music tracker based on on-line time warping, as it is implemented via the FLOWER framework.

4

ROBUST MULTI-AGENT MUSIC TRACKING

PRELIMINARIES This chapter describes work on a novel music tracking approach based on simultaneous tracking of the live performance via multiple recordings of performances of the same piece. In particular, this chapter is based on

[13] Andreas Arzt and Gerhard Widmer. "Real-time Music Tracking using Multiple Performances as a Reference". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Málaga, Spain, 2015, pp. 357–363

[7] Andreas Arzt, Harald Frostel, Thassilo Gadermaier, Martin Gasser, Maarten Grachten, and Gerhard Widmer. "Artificial Intelligence in the Concertgebouw". In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. Buenos Aires, Argentina, 2015, pp. 2424–2430

This work was motivated by the task of tracking complicated symphonic music, which poses some unique challenges. On a personal note, the two papers above also mark two personal achievements. For "Real-time Music Tracking using Multiple Performances as a Reference" we received the best paper award at the ISMIR conference. But more importantly, the paper "Artificial Intelligence in the Concertgebouw" describes how we successfully applied our tracking algorithm during a performance of Richard Strauss' Alpensinfonie at the Concertgebouw in Amsterdam, which was a remarkable challenge.

CONTRIBUTIONS Unless stated otherwise, the research in this chapter was conducted by the author of the thesis. Gerhard Widmer was involved in the writing process. Harald Frostel, Martin Gasser, Maarten Grachten and Thassilo Gadermaier helped with data preparation and setup for the application of the multi-agent tracking algorithm at the Concertgebouw. Martin Gasser was responsible for the client-server infrastructure that was used to distribute the data to the tablet computers in the audience [56].

4.1 INTRODUCTION

The performance of a music tracker depends considerably on the quality of the underlying score representation. If a score differs from the actual performance (e.g. missing notes, missing voices, different tempo, differences due to different editions), the tracker will have a

hard time following the performers. In addition, as we are treating music tracking as an audio-to-audio synchronisation task, the quality of the synthesised audio version of the score plays a vital role. For piano music, producing synthesised audio files of sufficient quality is feasible. High quality sound fonts are available, and the piano produces sounds in a very controlled way, compared to many other instruments. There is a limited and well-defined number of pitches. The sound is produced by a hammer that strikes the strings with a specific speed, and the main parameter that can be varied by the performer is the speed which controls the loudness. In addition, the performer has some additional control of the timbre via the pedals.

In contrast to this, e.g. a violin player has much more direct control of the way the sound is produced. The strings can be bowed or picked, the performer can produce basically any pitch, might apply vibrato, or add special harmonics by only touching a string lightly, can control the timbre via various bowing techniques, and so on. This complexity makes it much harder to produce an appropriate audio file representing the score for music tracking. While for piano this is an automatic process (the result is very mechanic, but sufficient for our purposes), for many other instruments a sound engineer would have to spend a considerable amount of time to produce usable output. Thus, tracking an orchestra, which produces a complex mix of sounds of instruments with very different properties, is a special challenge. While in theory it is possible to generate an audio representation of the score that is suitable as a basis for our tracker, it clearly is impracticable and very inflexible.

Thus, the goal of the work presented in this chapter was to find an automatic way of producing high quality features for music tracking. The basic idea of our approach is to use other performances of the same piece as a tracking reference. We propose to automatically prepare the necessary symbolic data via off-line alignment, and present a multi-agent approach based on a number of these automatically preprocessed performances that substantially improves the tracking results (see Section 4.2, which is based on [13]). This multi-agent approach was then applied in a similar scenario to a very challenging task. At the Concertgebouw in Amsterdam we had the opportunity to track the world-famous Concertgebouw Orchestra, which was playing the Alpensinfonie by Richard Strauss. A description of this challenge is given in Section 4.3, which is based on [7].

4.2 MUSIC TRACKING USING MULTIPLE PERFORMANCES AS A REFERENCE

The music tracking algorithm presented in Chapter 3 is based on the idea of aligning the live performance to a synthesised audio version of the score. As could be seen, this generally works well, especially

ID	COMPOSER	PIECE NAME	# PERF.	DATA TYPE
CE	Chopin	Etude Op. 10 No. 3 (excerpt)	22	Match
CB	Chopin	Ballade Op. 38 No. 1 (excerpt)	22	Match
MS	Mozart	1^{st} Mov. of Sonatas KV279, KV280, KV281, KV282, KV283, KV284, KV330, KV331, KV332, KV333, KV457, KV475, KV533	1	Match
RP	Rachmaninoff	Prelude Op. 23 No. 5	3	Manual
B3	Beethoven	Symphony No. 3	1	Manual
M4	Mahler	Symphony No. 4	1	Manual

Table 4.1: The evaluation data set.

for piano music – but it turned out to be problematic for complex polyphonic orchestral music due to the quality of the automatically generated "score audio".

In this section, which is based on [13], an alternative approach is presented. Instead of using the symbolic score directly, we propose to first automatically align a recording of another performance of the same piece to the score. Then, we use this automatically annotated "score performance" as the new score representation for the on-line tracking process. The motivation for this is twofold. Firstly, we expect the quality of the features to be higher than if they were computed from a synthesised version of the score. Secondly, in a performance a lot of intricacies are encoded that are missing in the symbolic score, including changes in (local) tempo and loudness. In this way we implicitly also take care of special events like trills, which normally are insufficiently represented in a symbolic score representation.

As will be seen in this section, this approach proves to be promising, but the results also depend heavily on which performance was chosen as a reference. To improve the robustness we further propose a multi-agent approach (inspired by [139], where a related strategy was applied to off-line audio alignment), which does not depend on a single performance as a reference, but takes multiple "score performances" and aligns the live performance to all these references simultaneously. The output of all agents is combined to come up with the current position in the score. As will be shown in the evaluation, this extension stabilises our approach and increases the alignment accuracy.

4.2.1 *Data Description*

Compared to Chapter 3, the data collection was expanded with two Symphonies by BEETHOVEN and MAHLER (see Table 4.1). These two pieces were the main target of this new approach. Again, the table also indicates how the ground truth was compiled. For the Ballade and Etude by CHOPIN, and for the MOZART Piano Sonatas we have access to accurate data about every note onset that was played, as these were recorded on a computer-monitored grand piano. For the Prelude by RACHMANINOFF as well as for the Symphonies by BEETHOVEN and MAHLER we have to rely on manually annotated performances (at the note level for the Prelude and at the downbeat level for the two Symphonies). See Section 2.2 for more information about the data collection.

Furthermore, we collected a number of additional performances of the pieces in our dataset. For these we do not have any annotations, and their sole purpose is to be processed fully automatically. These will act as replacements for the symbolic scores. We collected seven additional performances for each piece in the dataset. We made an exception for the excerpts of the Ballade and the Etude by CHOPIN, as we already have 22 performances of those. We thus reused these performances accordingly, randomly selected seven additional performances for each performance in the evaluation set, and treated them in the same way as the other additional data (i.e. we did not use any part of the ground truth, everything was computed automatically when they were used as a "score performance"). We also took care not to use additional performances of the same performer(s) that occur in our evaluation set.

4.2.2 *Standard Music Tracking Based on a Symbolic Score Representation*

The methods described here are based on the music tracking approach as detailed in Chapter 3. First, we re-evaluated the algorithm on the expanded data set (see Table 4.2[1]). The results are shown as proportion of correctly aligned pairs of time points (note times or downbeat times, respectively) for different error tolerances (in seconds). For instance, the first number in the first row means that for the CHOPIN Etude the alignment was performed for 33% of the "annotation time points" (i.e. notes or downbeats) with an error smaller than or equal to 0.05 seconds.

The goal is to improve on these results, both regarding tracking accuracy and, especially, robustness (i.e. reduce the amount of big mistakes made by the music tracker). As can be seen, the algorithm

1 The differences between these results and the ones presented in Section 3.5 are explained by small optimisations and changes of parameter settings of the algorithm (see Section 3.6.1).

ERROR	CE	CB	MZ	RP	B3	M4
≤ 0.05 s	0.33	0.33	0.55	0.45	0.42	0.23
≤ 0.25 s	0.96	0.92	0.97	0.90	0.84	0.71
≤ 0.50 s	0.99	0.96	0.98	0.96	0.91	0.83
≤ 0.75 s	1	0.98	0.99	0.98	0.94	0.87
≤ 1.00 s	1	0.98	0.99	0.98	0.95	0.91

Table 4.2: Results for the *on-line tracking algorithm*. The results are shown as proportion of correctly aligned pairs of time points (note times or downbeat times, respectively) for different error tolerances (in seconds).

works particularly well on the piano pieces, but shows problems with the two symphonies. A reason for this is that it is relatively easy to synthesise piano pieces from MIDI in acceptable quality, but it is much harder to do this automatically for orchestral pieces.

4.2.3 *Music Tracking via a Single Performance as a Reference*

As we are effectively treating the task of music tracking as an on-line audio-to-audio alignment task, we can actually use any annotated audio recording of a performance as a score representation. Using a real performance as a "score" has some advantages.

Firstly, an audio file synthesised from a deadpan MIDI file may sound bad compared to a real performance, thus also the features are of relatively low quality (i.e. they differ sometimes quite heavily from the features computed from the live performance we want to track). Despite obvious differences between performances, their respective features tend to be more similar to each other. This is especially true for orchestral pieces, which often include instruments that are hard to synthesise in high quality (or at least this would demand expensive sound fonts and a lot of effort by a trained audio engineer).

Secondly, a performance implicitly encodes a lot of information that is missing in the symbolic score. This includes detailed information about tempo, loudness and articulation. Again we want to stress that of course performances differ from each other quite heavily, but compared to the differences between a performance and an audio synthesised from the MIDI file, these differences are small.

There is also one big disadvantage: the symbolic information linking time points in the audio to beat times in the score, which we get for free when we use a MIDI file as the basis for the score audio, is missing. Thus, this information needs to be generated. There are two possible ways to do that: (1) by manual annotation, which can be very laborious, or (2) by automatic off-line alignment of the performance to the score — which is the option we decided on, as we are interested

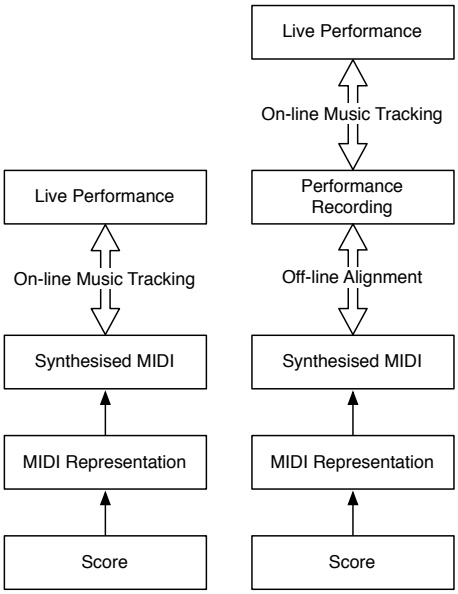


Figure 4.1: Standard music tracking (left) vs. music tracking via an off-line aligned reference performance (right).

in an automatic method to improve tracking results (see Section 4.2.3 below).

Figure 4.1 shows a sketch of the intended setup. On the left, standard music tracking is shown, where the live performance is aligned to the symbolic score (via a synthesised audio). On the right, another performance is first aligned to the symbolic score. This performance is then used as the new reference in the on-line alignment process.

Off-line Alignment

To use a performance as a "score" we have to generate the necessary symbolic information, linking time points in the audio to beat times in the score. As we are interested in a fully automatic way to improve the tracking results, we decided to use off-line audio alignment to align the "score performance" to the symbolic score, which gives us the needed mapping as a result. As off-line audio alignment is far more accurate than on-line tracking, our intuition was that the increase in feature quality outweighs the error introduced by the off-line alignment process.

The off-line alignment is computed with the music tracking algorithm from Section 4.2.2, with the only difference being that in the end we compute the backward path, as it is done in the standard DTW algorithm. As this path is based on more information (i.e. it is computed in a non-causal way), the results are generally much more accurate than in the on-line case. Of course any off-line audio-to-score alignment algorithm could be used for this task (see Section 2.4.2 for a discussion of work related to this task).

ERROR	CE	CB	MZ	RP	B3	M4
≤ 0.05 s	0.92	0.87	0.93	0.75	0.54	0.38
≤ 0.25 s	0.99	0.97	0.99	0.97	0.93	0.86
≤ 0.50 s	1	0.97	1	0.99	0.96	0.94
≤ 0.75 s	1	0.98	1	0.99	0.97	0.97
≤ 1.00 s	1	0.98	1	1	0.98	0.98

Table 4.3: Results for the *off-line alignments*. The results are shown as proportion of correctly aligned pairs of time points (note times or downbeat times, respectively) for different error tolerances (in seconds).

ERROR	CE	CB	MZ	RP	B3	M4
≤ 0.05 s	0.39	0.35	0.52	0.25	0.35	0.27
≤ 0.25 s	0.98	0.96	0.97	0.87	0.85	0.80
≤ 0.50 s	0.99	0.97	0.99	0.97	0.93	0.92
≤ 0.75 s	1	0.98	0.99	0.99	0.95	0.95
≤ 1.00 s	1	0.98	1	1	0.97	0.96

Table 4.4: Results for *on-line music tracking* based on a *single off-line aligned performance as a reference*. The results are shown as proportion of correctly aligned pairs of time points (note times or downbeat times, respectively) for different error tolerances (in seconds).

Just to get a rough idea of how much error will be introduced by the off-line alignment, we ran an experiment on our test data and aligned it to the symbolic scores (later on, off-line alignments of the additional recordings will be used, but we expect a similar behaviour). Unsurprisingly, the results show that there is a gap between the results of the off-line approach (see Table 4.3) and the on-line music tracking approach (see Table 4.2). As we will use the off-line algorithm during data preparation, we strongly expect that the higher quality of the features and the additional information encoded in the performances will outweigh the error that is introduced during this step.

Thus, we aligned all the additional performances from Section 4.2.1 to the respective symbolic scores, resulting in performances with linked symbolic information. In the following sections, we will use these performances as new references ("score performances") for the music tracking algorithm.

Tracking Based on a Single Aligned Performance

Given the automatically computed "score performances", we can now use them in the tracking process as shown in Figure 4.1. In this experiment, each performance from the evaluation set is aligned to

ERROR	CE	CB	MZ	RP	B3	M4
≤ 0.05 s	0.39	0.35	0.58	0.19	0.44	0.32
≤ 0.25 s	0.99	0.98	0.99	0.92	0.90	0.84
≤ 0.50 s	1	0.98	1	1	0.95	0.94
≤ 0.75 s	1	0.98	1	1	0.96	0.96
≤ 1.00 s	1	0.99	1	1	0.97	0.97

Table 4.5: Results for the *multi-agent tracking* approach based on a *set of off-line aligned performances as a reference*. The results are shown as proportion of correctly aligned pairs of time points (note times or downbeat times, respectively) for different error tolerances (in seconds).

the score via each respective "score performance", resulting in seven on-line alignments for each performance.

The results are given in Table 4.4 and should be compared to the numbers in Table 4.2. As can be seen, the general trend is an improvement in robustness, especially for the complex orchestral pieces (e.g. the percentage of aligned downbeats with an error smaller than 250 ms increased from 71% to 80% for the MAHLER symphony).

Unfortunately, the results also proved to be unstable. Some performances are more similar (or at least easier to align) to each other, which also results in good tracking results — but the use of some of the "score performances" led to results that were worse than our basic approach. A closer look at the positions where tracking errors occurred showed that some of them happened at the same points in time over all alignments of the piece — basically showing that some parts are harder to track than others. But there were also many alignment errors that occurred only for one or two of the "score performances", but not for the others. This led us to the idea to combine individual on-line alignments in such a way that it would smooth out these errors.

4.2.4 *Music Tracking via a Set of Performances as Reference*

The analysis of the results from Section 4.2.3 above showed that a combination of a number of on-line alignments might further improve the tracking results. Here, we propose a simple multi-agent strategy (see Figure 4.2 for an illustration). During a live concert n trackers run in parallel and each tracker tries to align the incoming live performance to its score representation, each producing its own, independent hypothesis of the current position in the score. Finally, the hypotheses are combined to form one collective hypothesis of the music tracking system.

Many different ways of combining the hypotheses would be possible, e.g. based on voting or on the current alignment error of the individual

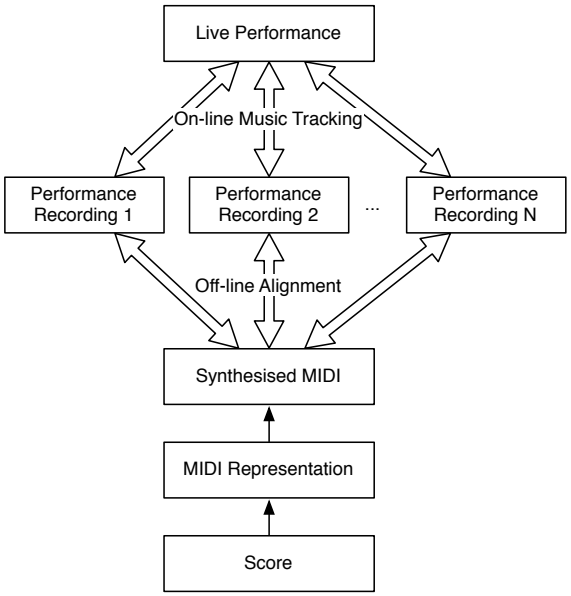


Figure 4.2: Multi-agent tracking based on off-line aligned performances as a reference.

trackers. Here, we decided on a very simple method: taking the median of the positions that are returned by the individual trackers. The reasoning behind this is that trackers tend to make mistakes in both directions — i.e. "running ahead" (reporting events to early), and "lagging behind" (reporting events with some delay) — with about the same frequency. Thus, trackers that stay safely in the middle of the pack tend to give a robust estimate of the position in the score.

Furthermore, using the median also means that as long as $\frac{n}{2} + 1$ trackers stay close to the actual position, the system would still come up with a reasonable position estimate — while this is not directly reflected in the evaluation results, this extra robustness is convenient when the tracking algorithm is used in real-world applications. Further strategies to increase the robustness are possible, like the automatic replacement of trackers that got lost, but were not used in our experiments.

For the evaluation, we set $n = 7$, as this was a good trade-off between robustness and computation time (seven on-line alignments can still be easily computed in real-time on a conventional consumer laptop). The results given in Table 4.5 show that our approach is working well. Errors of more than 1 second are rare, and the multi-agent approach even improved the alignment accuracy for all pieces (with the exception of the Prelude by Rachmaninoff).

4.2.5 *Discussion*

The main goal of our approach was to increase the robustness of the algorithm, i.e. to decrease the frequency of large errors and to make

PIECE	OFF-LINE	STANDARD	VIA 1	VIA 7
CE	99.06%	95.62%	97.92%	**98.78%**
CB	97.13%	92.10%	96.00%	**97.93%**
MZ	99.35%	96.88%	97.46%	**99.04%**
RP	96.62%	90.14%	87.47%	**92.47%**
B3	92.88%	83.67%	85.04%	**89.55%**
M4	86.74%	71.15%	80.06%	**83.66%**

Table 4.6: Comparison of the results (error tolerance 250 ms). The results are shown as percentage of matching pairs of time points (note times or downbeat times, respectively). For instance, the first number in the first row means that for the CHOPIN Etude the off-line alignment was performed for 99.06% of the notes with an error smaller than or equal to 0.25 seconds. The results of the *off-line* alignment algorithm are only shown for comparison. *Standard* refers to the on-line music tracker using a synthesised MIDI as a reference (see Section 4.2.2), *Via 1* to the tracker using a single "score performance" as a reference, *Via 7* to the multi-agent approach based on seven trackers.

sure that the tracker does not get lost, even when following difficult orchestral pieces. For convenience, we give a summary of the results (see Table 4.6) based on a common measure in the evaluation of music tracking algorithms: the percentage of notes that were aligned with an error less than or equal to 250 ms (see [31]). As can be seen, the multi-agent approach based on automatically aligned reference performances improves the results heavily — in fact for CB the results of the on-line alignment even surpassed the off-line alignment. For the results on the CHOPIN data (CE and CB) one has to take into account that we used 22 performances which were recorded by different performers, but still on the same piano and with the same recording setup, which will have a positive influence on the alignment results. Still, as the remaining results show, even when completely unrelated performances of the same piece were used as references, the alignment results improved drastically.

Especially for the orchestral pieces (B3 and M4), we can see that our intuition proved to be correct: the error introduced by the off-line alignment had a lot less impact than the better quality of the features and the additional tempo and loudness information provided by the performances. The multi-agent approach proved to be very effective regarding the increase in robustness. It smoothes out some of the bigger errors that occur when using just a single performance as a score reference.

4.3 ARTIFICIAL INTELLIGENCE IN THE CONCERTGEBOUW

The multi-national European research project PHENICX[2] provided us with the unique opportunity (and challenge) to demonstrate our score following technology in the context of a live symphonic concert [7]. The general goal of the project was to develop technologies that enrich the experience of classical music concerts. In the experiment to be described, this was done by using the music tracker to control the transmission and display of additional visual and textual information, synchronised to the live performance on stage (a similar system was presented in [110] in the context of the PHILADELPHIA ORCHESTRA). The user interface and the visualisations were provided by our project partner VIDEODOCK[3].

The event took place on February 7th, 2015, in the CONCERTGEBOUW in Amsterdam. The ROYAL CONCERTGEBOUW ORCHESTRA, conducted by Semyon BYCHKOV, performed the ALPENSINFONIE ("Alpine Symphony") by Richard STRAUSS. This concert was part of a series called ESSENTIALS, during which technology developed within the project could be tested in a real-life concert environment. All the tests during this concert series had to be as non-intrusive as possible. For the demonstration during the concert in question, a test audience of about 30 people was provided with tablet computers and placed in the rear part of the concert hall to not disturb the regular concert goers.

The setup was as follows. Two microphones were placed a few meters above the conductor, picking up the music, but also a lot of noise, e.g. coughing in the audience and noise made by orchestra members, and a fair amount of reverberation from the hall. In a control room behind the scenes a regular consumer laptop was receiving the audio signal and feeding it to a music tracking algorithm, computing at any point during the performance the current position in the score. This information was sent to the tablets of the test audience and triggered pre-prepared visualisations at the appropriate times. The audience could choose between three different kinds of synchronised visualisations: the sheet music (with synchronised highlighting of the current bar, and automatic page turning), textual information and explanations, and an artistic video, visualising the story of the symphony (which is "program music" *par excellence*). Two pictures with impressions from the live setup are shown in Figure 4.3.

This specific application of music tracking poses some unique challenges. Most importantly, so far the focus of music tracking has mostly been on solo or small ensemble music, like solo violin or flute, solo piano or string quartets, and not to a full sized orchestra (according to STRAUSS' notes the optimal size of the orchestra for the ALPENSINFONIE is 129 or more musicians!). This level of polyphony and of variety of

2 http://phenicx.upf.edu
3 http://videodock.com

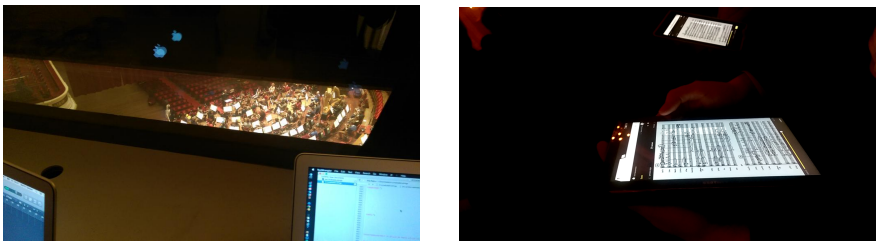


Figure 4.3: Left: view from the control room onto the stage (during orchestra rehearsal); right: synchronised score display in the audience during the concert.

instruments has to be considered when choosing the internal representation of the score and the features used during the on-line alignment process. Furthermore, this piece challenges the music tracking algorithm with a vast range of musical situations: very quiet and slow parts without a clear melodic line (only a sound texture), very sparse parts with long pauses, energetic, loud and fast parts and even solo sections.

Ideally, the tracker has to do equally well in all these situations, or at least well enough to not get lost completely. Thus, the main focus of our music tracking algorithm is placed on robustness. It actually does not matter much if an event is detected with a short delay, but it is very important that the algorithm does not get lost during this long piece (a typical performance takes about 50 minutes and contains no breaks).

4.3.1 *The Score: Data Representation*

To make the live tracking possible some internal representation of the musical score is needed. Normally the starting point is a symbolic representation of the score, which we then semi-automatically connect to images of the sheet music, for visualisation purposes (most of the time we do this at the bar level). As a basis for the tracking process, we synthesise the symbolic score and either use it directly (see Chapter 3), or base the tracking on the multi-agent approach described above (see Section 4.2).

For the Alpensinfonie we ran into problems with this approach. First of all, it is far from easy to get good quality symbolical representations for orchestral pieces. In the case of the Alpensinfonie we found some MIDI files on the internet, but in the end all of them turned out to be unusable because of grave mistakes and missing parts. We also contacted a music publishing house, but they could not provide us with a symbolic version for this particular piece. In theory one could try to scan a printed score — those are of course readily available — and try to convert it to a symbolic representation. Unfortunately, optical music recognition (OMR) algorithms are still

not good enough to cope with the complexity of an orchestral score fully automatically, and the manual correction of their output would take an immense amount of time.

Thus, we decided to try a different approach and did not use a symbolic representation at all. Instead, we directly used a recording of the same piece as the basis for the music tracking. This version — we selected a performance by the ROYAL CONCERTGEBOUW ORCHESTRA from 2007, conducted by Mariss JANSONS — is manually annotated beforehand, so that the timing of each downbeat (the first beat in a bar, and thus the start time of a new bar) is known and the performance can be used as a direct replacement of the score (all the visual information shown to the audience is timed at the bar level).

We strongly believe that from a practical point of view this is the best approach for tracking music of this complexity (regarding number of instruments / polyphony). As discussed in Section 4.2, the resulting score features are of very high quality, while the amount of time spent on annotating the performance (about 12 hours) was acceptable — especially compared to the amount of time it would have taken to either repair one of the MIDI files or produce a digital version of the score from scratch.

The absence of symbolic scores in the system also means that theoretically it can be used for *any* piece of music for which a recording exists. This immensely extends the repertoire for which music tracking can be used.

Another important point is that the amount of annotations actually depends on the specific usage scenario. We decided to show the sheet music synchronised at the bar level, and thus needed to annotate the timing of every downbeat. As the piece consists of 1154 bars, we had to find each of these points in the audio file. Then we linked all the remaining information (the text and the videos) to these time points. Had we decided to only turn the pages automatically, the annotation work would have been reduced to about 190 time points (160 pages plus about 30 additional events for the videos and textual information).

The downside of this approach is that without the symbolic score there is no information about specific notes. While this is not important for our task, it might be important if the computer's role is a more proactive one and predicts the timing of certain events before they are being played, or makes use of the symbolic information to actively take part in the performance (e.g., by synthesising an accompaniment).

4.3.2 *Following Live Orchestral Music: Tracking Algorithm*

For the tracking itself we relied on our multi-agent approach as described in Section 4.2.4. We collected six more performances of the ALPENSINFONIE and aligned them to the manually annotated perfor-

CONDUCTOR	ORCHESTRA	YEAR	DUR.	DATA TYPE
Jansons	Royal Concertgebouw Orch.	2007	52:51	Manual
Haitink	London Symphony Orch.	2008	50:20	Alignment
Previn	Philadelphia Orch.	1980	49:09	Alignment
Karajan	Berlin Philharmonic Orch.	1980	51:05	Alignment
Luisi	Staatskapelle Dresden	2007	50:42	Alignment
Haitink	Royal Concertgebouw Orch.	1985	49:29	Alignment
Järvi	Royal Scottish National Orch.	1987	49:33	Alignment

Table 4.7: Performances annotated to be used as alignment basis ("score representations")

mance via off-line audio alignment (see Table 4.7 for an overview on the data), to produce the information about the location of the downbeats. This means that these six additional "score performances" were produced without any additional manual effort. Figure 4.4 shows a sketch of the multi-agent tracker as it was used at the concert.

4.3.3 *The Event: Live Tracking in the Concertgebouw*

The event on February 7, 2015 in the CONCERTGEBOUW was a big success. The tracking went smoothly and there were no glitches, only some minor inaccuracies. An obvious mistake happened at the quiet section in the beginning (see Figure 4.5). The sound texture here essentially consists of a very soft and repeating pattern. In cases like this the trackers sometimes tend to "wait", because they try to align newly incoming instances of the pattern to past positions in the score (that also represent the same pattern). This resulted in a perceived delay of roughly one bar, for a period of about five bars. As soon as the texture changed and more distinct sounds could be recognised, the trackers recovered quickly. There were no further noticeable problems and in the end all of the trackers could follow the whole concert, and there was never any concern that the system might fail.

The general opinion amongst the project staff and the test audience was that the tracking worked very well and the accuracy was more than sufficient to trigger the visualisation in time. Only a few inaccuracies were noticed.

4.3.4 *Evaluation*

For a quantitative post-hoc evaluation we annotated a recording of the live concert. The results of the evaluation are presented in Tables 4.8 and 4.9. To see the effect of the multi-tracking approach, we also simulated the tracking of the performance via the single annotated "score performance". As can be seen, there are only slight differences

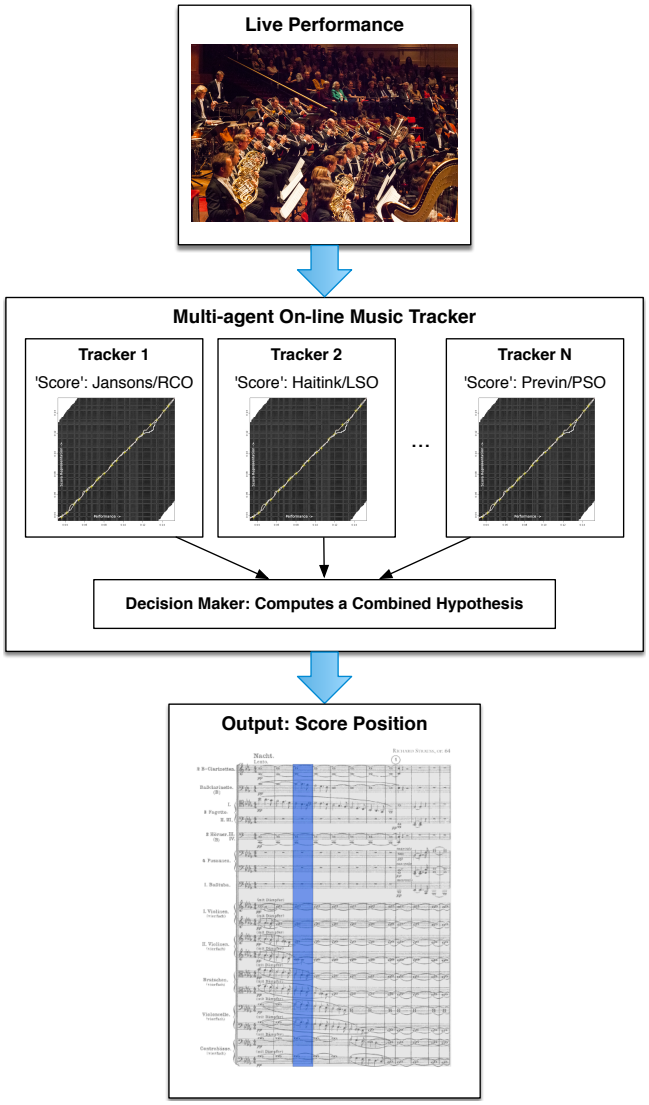


Figure 4.4: The Multi-agent Tracker. The live input is fed to N independent instances of the tracking algorithm. Each aligns the input to its own score representation, based on different performances of the same piece. Then, the individual hypotheses are combined and the estimate of the current position in the score is returned. Source of the sheet music: IMSLP (http://imslp.org), Score 20542. Picture of the Concertgebouw Orchestra (c) Christina Chouchena / Royal Concertgebouw Orchestra.

in the results for the single tracker and the multi-agent approach. Keeping in mind that the goal of the multi-agent approach was to increase the robustness — as long as $\frac{n}{2} + 1$ trackers stay close to the actual position, the system would still come up with a reasonable position estimate —, this is a good result: extra robustness and a slight increase in accuracy were achieved without any extra manual efforts as the additional data was prepared by automatic methods.

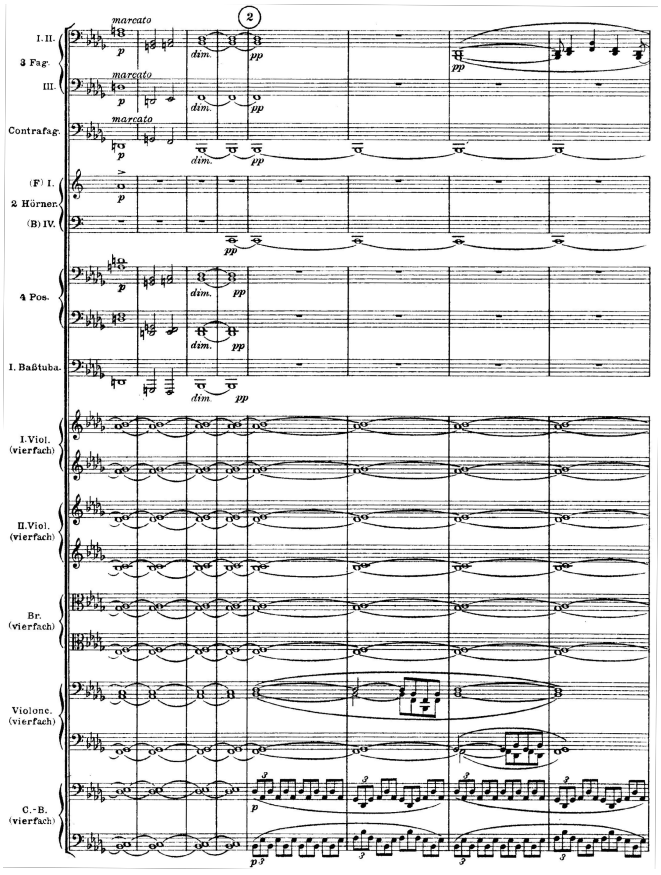


Figure 4.5: Excerpt from the score. This part is played very slowly and softly (note the *p* and *pp* dynamic markings), without a distinct melody (sustained notes in the strings, horns and the contrabassoon). The triplet figures in the bass section are so soft that they don't stand out but add to the overall sound texture. Source of the sheet music: IMSLP (`http://imslp.org`), Score 20542.

The results were more than sufficient for the task in question. The median error for the multi-tracking approach is about 0.1 seconds. Only in very rare cases did the tracker make major mistakes. Specifically the section already discussed above (see Figure 4.5) still causes problems, culminating in a maximum error of 5.38 seconds at bar 24 (which translates to about 1.5 bars, as this part has a relatively slow tempo). The extent of the problem was not as apparent during the concert itself, also because even for humans it is very hard to follow the orchestra during this part.

ERR. (SEC)	SINGLE	MULTI-AGENT
≤ 0.25	78.25%	**81.80%**
≤ 0.50	92.20%	**93.24%**
≤ 0.75	95.57%	**96.44%**
≤ 1.00	97.49%	**98.01%**

Table 4.8: Real-time alignment results for the *single tracker* and the *multi-agent tracker*, shown as cumulative frequencies of errors of matching pairs of downbeats. For instance, the first number in the first row means that the single tracker aligned 78.25% of the downbeats with an error smaller than or equal to 0.25 seconds.

	SINGLE	MULTI-AGENT
Average Error	0.20 sec.	0.19 sec.
Standard Dev.	0.35 sec.	0.36 sec.
First Quartile	0.06 sec.	0.05 sec.
Median Error	0.11 sec.	0.10 sec.
Third Quartile	0.22 sec.	0.19 sec.
Maximum Error	5.33 sec.	5.38 sec.

Table 4.9: Real-time alignment results for the *single tracker* and the *multi-agent tracker* on the ALPENSINFONIE.

4.4 CONCLUSIONS

In this chapter a robust approach to real-time music tracking was presented. Instead of tracking directly on a symbolic score representation, we first use off-line alignment to match other performances of the piece in question to the symbolic score. We then use these performances as our new score representation, which results in high quality features, and implicitly also adds extra information about how this piece generally is performed. Together with a multi-agent tracking strategy, which smoothes out most of the major errors, we achieve increased robustness and also increase the accuracy of the live tracking, especially for complex orchestral music. We also reported on a successful real-world test of our algorithm in a famous concert hall.

4.5 PROTOTYPICAL IMPLEMENTATION

A very general sketch of the prototype has already been shown in Figure 4.4 above. Figure 4.6 shows the implementation of the multi-agent tracker within the FLOWER framework. The feature computation stage is the same as for the normal tracker. The only difference is that instead of one tracking algorithm, multiple instances are used,

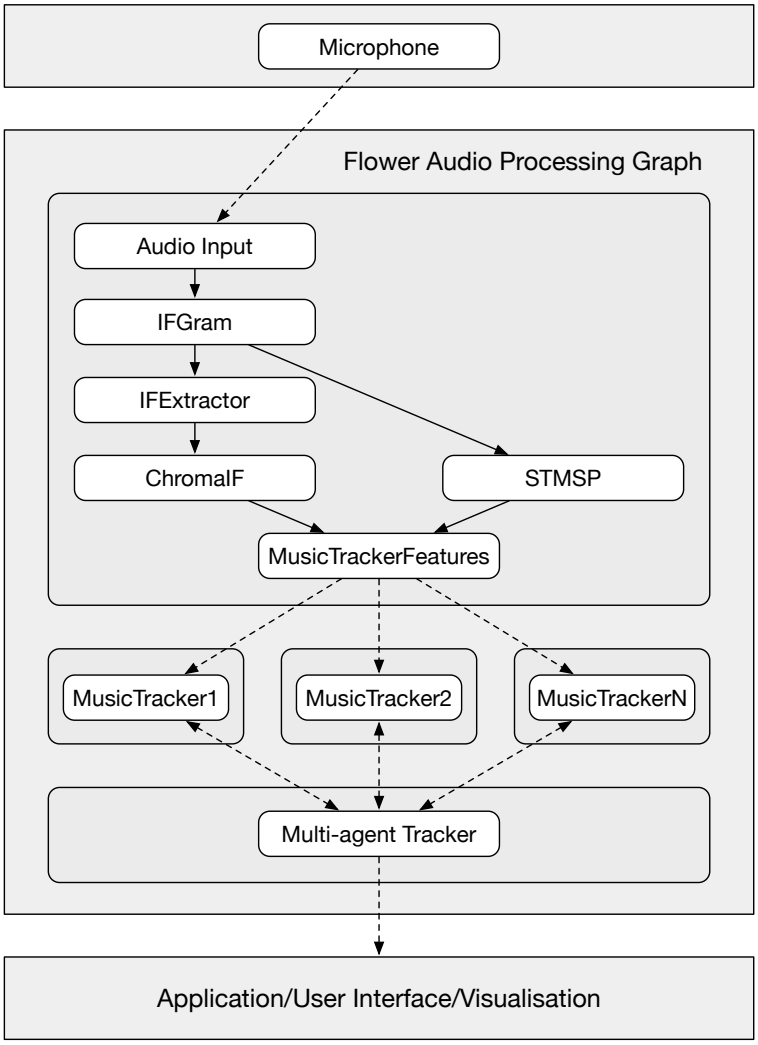


Figure 4.6: The multi-agent tracker, as used e.g. at the Concertgebouw for tracking orchestral music.

each running in a separate zone (thread). For most of the experiments, including the live demonstration at the Concertgebouw, seven trackers are running in parallel, each using its own score representation based on a performance of the piece in question. The outputs of the individual trackers are collected by the Multi-agent Tracker to compute a combined hypothesis about the current position in the score. In addition, the Multi-agent Tracker processing can also directly influence individual trackers and e.g. reset them to a sensible position if they clearly got lost.

This setup was successfully employed multiple times for public demonstrations, mostly in the context of orchestral music (see Section 6.2 for descriptions of further live demonstrations, as well as videos of these experiments).

5

NEAR-INSTANT PIECE IDENTIFICATION AND FLEXIBLE MUSIC TRACKING ON A DATABASE OF SCORES

PRELIMINARIES This chapter describes a flexible music tracking algorithm that is able to follow the performer not only based on a single score, but on a complete database of scores. To do so, the tracker needs a way of identifying the piece that is being played, and the position within that piece. Here, work on a fingerprinting algorithm is described that tries to solve this task.

The main contributions of this chapter were published in

[6] Andreas Arzt, Sebastian Böck, and Gerhard Widmer. "Fast Identification of Piece and Score Position via Symbolic Fingerprinting". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Porto, Portugal, 2012, pp. 433–438

[17] Andreas Arzt, Gerhard Widmer, and Reinhard Sonnleitner. "Tempo- and Transposition-invariant Identification of Piece and Score Position". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Taipeh, Taiwan, 2014, pp. 549–554

Preliminary work that will only be briefly described can be found in

[15] 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)*. Patras, Greece, 2008, pp. 241–245

[12] Andreas Arzt and Gerhard Widmer. "Towards Effective 'Any-Time' Music Tracking". In: *Proceedings of the Starting AI Researchers' Symposium (STAIRS)*. Lisbon, Portugal, 2010, pp. 24–36

CONTRIBUTIONS Unless stated otherwise, the research in this chapter was conducted by the author of the thesis. The transcription algorithm was developed (in PYTHON) by Sebastian BÖCK [20], who also trained the models used in the presented experiments. The author of the thesis re-implemented the necessary parts of the algorithm in C++. Reinhard SONNLEITNER contributed the idea of the verification step, which was adapted to the use case described in this chapter. Gerhard WIDMER was involved in the writing process and gave valuable suggestions regarding the evaluation procedures used in this chapter.

5.1 INTRODUCTION

Generally, the underlying assumption for the task of music tracking (see Section 2.4.1) — and also for many cases of off-line music synchronisation (see Section 2.4.2) — is that the performers follow a fixed score (the sheet music), and deviate from this score only in some minor details. Mistakenly or deliberately, they may leave out a few notes or play a few additional notes that are not in the score. Often, they will not strictly follow tempo and loudness annotations as given in the sheet music. But almost all music tracking algorithms in the literature assume that regarding note content and structure the performers follow the sheet music closely, i.e. start at the beginning, play the piece as written down in the sheet music, without additional repetitions, without skipping parts, and end with the final notes.

In many real-life situations this view of music tracking is too restrictive. For example, when practising a piece of music, the performer might repeat parts over and over, or skip sections and jump to another part that needs to be practised. Even in concert situations performers sometimes ignore repeated sections. Music tracking algorithms that are not aware of these structural deviations are useless in these situations and will inevitably get lost. For approaches to music tracking that are to some extent able to cope with structural differences within a single piece see e.g. [97, 104, 134]. The same problem for off-line alignment is discussed in Section 2.4.2.

In addition, if a musician at home is sitting down at the piano, just playing a single piece, it might be quite inconvenient to first set everything up, look for the piece in the database, press the start button and finally start playing. Furthermore, sometimes the musician may not even know the exact name of the piece, and just remember the first few notes, so she would have to browse the sheet music and actively look for the piece in question. Instead, it would be much more convenient to just start playing, and after the first few notes the piece in question comes up on the screen. Then, the system follows the performance and automatically turns the pages (or scrolls the sheet music).

In this chapter an approach is presented that makes music tracking much more flexible and can cope with all the situations mentioned above. The techniques presented here can form the backbone of a Complete Classical Music Companion (see Section 2.1), which is of use not only for performers, but also for listeners of classical music. For this flexible approach, we coined the term "any-time music tracking": an algorithm that is constantly listening to its environment, always ready to react to music, to identify what it is listening to and to act accordingly and track the progress of the performer(s) over time.

The development of this approach can be followed over the last few years. In [15] a first effort was presented to at least be able to cope

with left-out or added repetitions, based on manual annotations of the possible paths through the piece. Later on, this was extended to deal with arbitrary jumps, but still within a single piece [12] (see Section 5.2 below). By developing a tempo-invariant version of fingerprinting [6] we were finally able to extend this approach to work on a database of scores. In addition, we developed a version that can even cope with arbitrary transpositions [17]. These two approaches are closely related, and jointly described in Section 5.3.

5.2 EARLY APPROACHES

A simple, inflexible approach to detecting deviations from the intended structure is described in [15]. The general idea is as follows (for an illustration see also Figure 5.1). In addition to the score (in symbolic form and as computed features), structural information is given. In the simplest case, this information consists of manually annotated section borders, all repeat signs, and similar annotations (e.g. *da capo al fine*). The goal is to detect possible skips and repetitions at these predetermined positions.

The tracking starts at the beginning of the piece. When it reaches the position of one of the provided annotations (the repeat sign (2) in Figure 5.1), according to this model there are three possible paths the performer can follow: the performer repeats the section in question, i.e. jumps back to (1) in the score, or she could ignore the repeat sign and just continue with the next section (2), or skip an entire section, i.e. jump to (3) and continue from there (which would be very unusual in this particular case).

To detect which course of action the performer takes, two additional instances of the tracking algorithm are initialised, which try to align the incoming live performance to the different score contexts. Thus, trackers are positioned at (1), (2), and (3) in Figure 5.1. The trackers then try to follow the live performance. After a few seconds, the alignment costs of the three trackers are compared. Generally, the tracker that followed the same path in the score as the performer will have the lowest alignment costs and thus it can be detected which decision the performer actually made.

This idea works very well, and all the following approaches are based on it. Thus, the main challenge was to get rid of the need for predefined structural information and to provide the system with ways to cope with arbitrary deviations.

In [12] we described a slightly more flexible method. The static structural information is replaced by a relatively crude algorithm that provides the system with a set of hypotheses that are in turn then checked by instances of the tracking algorithm.

In this system music tracking is performed on two different levels. The first level is using low-resolution audio features (frame size 600 ms

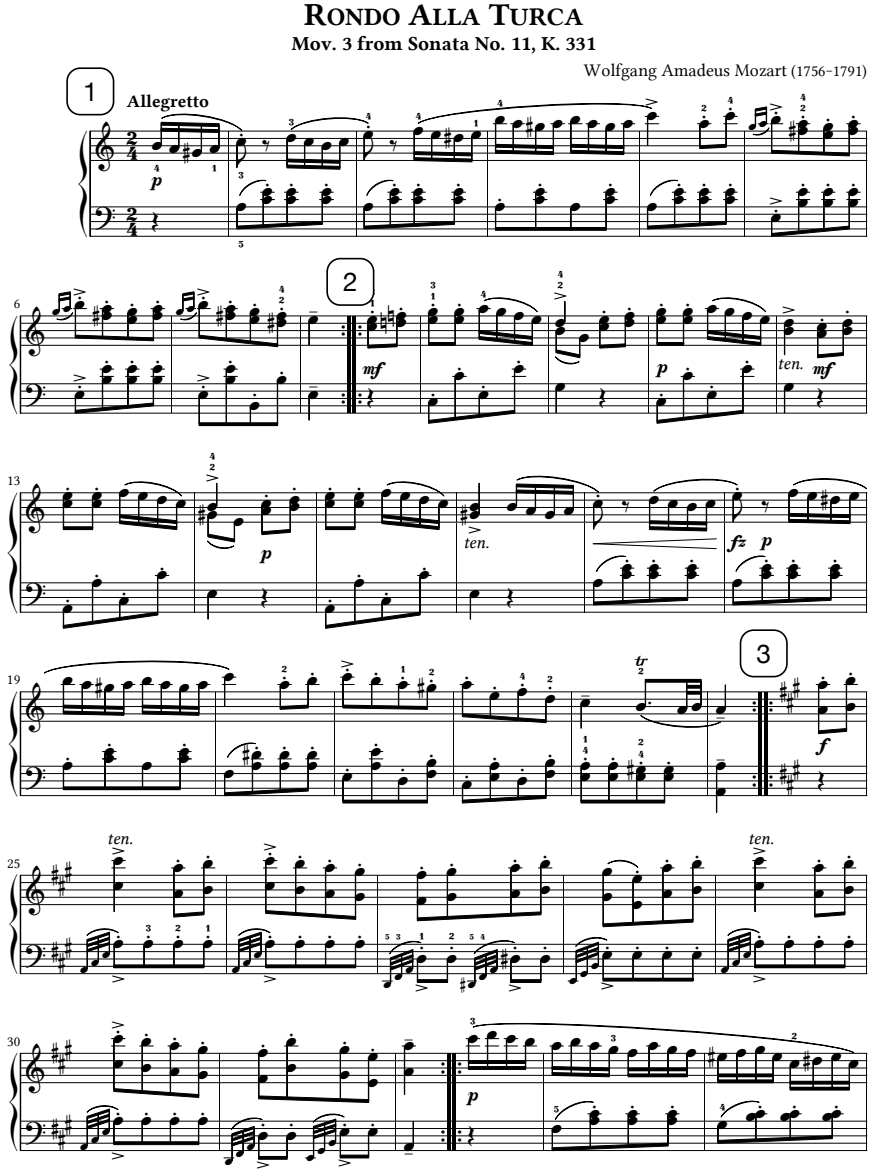


Figure 5.1: The first page of MOZART'S ALLA TURCA (Piano Sonata No. 11 in A major, K. 331, 3^{rd} movement), Source: IMSLP (http://imslp.org), Score 286253.

and hop size 300 ms) that are computed by aggregating the normal (high resolution) features. These low resolution features are computed for the score and for the live audio. A full distance matrix is computed, containing the distances of the live audio features to all score features. Based on this coarse resolution, for every incoming frame backward alignments starting in every position in the score are computed (via a computationally cheap, greedy local search algorithm). The result is a list of score positions, ordered by their alignment costs to the current context of the live performance. This effectively replaces the static information about possible "forks" in the sheet music and enables the detection of arbitrary structural changes.

The second level works on high-resolution features (frame size 46 ms and hop size 20 ms) and comprises a fixed number of instances of the tracking algorithm that take the hypotheses of the coarse tracking level and follow them over some time (as described above). Again, the real score position is determined by comparing their alignment costs over the last few seconds. The system is helped by an automatically computed structure model based on a self similarity matrix of the score that identifies equivalent positions (parts with the same note content), such that the system avoids tracking the same content via multiple trackers.

While working well on single pieces, which mostly have a duration of up to 10 minutes, it is obvious that for a collection of pieces this is not feasible. Firstly, the time and space complexity for the coarse tracking are too high, as they are linear in the database size. Secondly, even if the computation could be sped up via, e.g. some indexing techniques, preliminary experiments showed that the approach is not discriminative enough to produce good hypotheses on even a medium sized database (about 20 pieces). Due to these shortcomings, the idea to solve the identification problem in the audio domain was abandoned (thus the description here is rather brief – more details can be found in [12]). In the next section a different approach to this problem is presented, based on music transcription and a tempo- and pitch-invariant fingerprinting algorithm. Although it is motivated by the idea of using it in combination with a music tracker, this algorithm is a general approach to retrieve matching sheet music for short audio queries.

5.3 FAST IDENTIFICATION OF PIECE AND SCORE POSITION

Efficient systems for content-based music retrieval are a major topic in music information retrieval research. An important sub-task is the problem of music identification. This is a multi-faceted task, ranging from identifying noisy version of the same recording (via *audio fingerprinting*) to finding a song based on the hummed melody (*query by humming*). A brief summary is given in Section 2.4.3.

Here, we consider the task of *score identification*: given a short audio query, the goal is to identify the musical score it is based on. For example, if we present an audio excerpt of Vladimir Horowitz playing Chopin's Nocturne Op. 55 No. 1 to the system, it should return the name and data of the piece (Nocturne Op. 55 No. 1 by Chopin) rather than the data of the specific performance. Moreover, we are also interested in the exact position within the respective score. Accordingly, the database for this task does not contain audio recordings (as would be the case for most music identification tasks), but symbolic representations of musical scores (i.e., to identify the piece being played, the system only uses the symbolic score and has no information about the specific performance by Horowitz in the database). Especially for classical music, this is not a trivial task, because performances of the same piece generally differ heavily in terms of tempo, expressive timing, and other performance aspects, and the sheet music is by far under-specifying the way a piece is supposed to be played.

In the literature, the main method to solve this task is based on *audio matching* (see e.g., [92]). In this case the score is first transformed into an audio file (or a suitable mid-level representation). Then, an audio matching algorithm, commonly based on dynamic programming techniques (also see Section 2.4.2), retrieves all excerpts which musically correspond to a short query clip from a database. The downside of audio matching is that in general these methods are very slow (i.e. not suitable for the real-time task we have in mind). To cope with the computational costs, [75] presented clever indexing strategies that greatly reduce the computation time. Still, due to the coarse feature resolution, relatively large query sizes are needed. The audio fingerprinting approach presented in [63] suffers from the same limitation (see also Section 2.4.3).

Here, we describe a *symbolic* approach to score identification. Instead of creating an audio representation of the score we create the database directly from the symbolic score information. Then we transform the audio query into a symbolic representation — a list of note onset times with their respective pitches — and use a symbolic fingerprinting algorithm to find matching positions in the score database, inspired by the algorithm described in [138]. This process is very fast, can be used in real-time, on-line applications, and yields a high recognition rate (as can be seen in Section 5.5.3). Note that this algorithm involves music (audio) transcription — which is still an unsolved problem in the general case — as a query preprocessing step. For this system a state of the art music transcription system for piano music is used, which, despite many errors, provides transcriptions of sufficient quality for the robust symbolic fingerprinting algorithm. Still, this also means that this system currently only works for piano music (due to a lack of training data for other instruments).

DATA	PIECES	SCORE NOTES	PERF. NOTES	DURATION
Chopin Piano Works	154	325,263	326,501	9:38:36
Mozart Piano Sonatas	13	42,049	42,095	1:23:56
Additional Pieces	159	574,926	–	–
Total	326	942,238		

Table 5.1: Database and test set overview. In the database, all the pieces are included. As we only have performances aligned to the scores for the CHOPIN and the MOZART corpus, only these are included in the test set to query the database.

5.4 THE DATASET

For the evaluation of the score (and score position) identification task a ground truth is required. As for music tracking, exact alignments of performances (recordings) of classical music to their respective scores are needed, such that we know exactly when each note given in the score is actually played in the performance. Here, we use the datasets MOZART PIANO SONATAS and CHOPIN PIANO WORKS, which are described in Section 2.2. For these datasets both symbolic and audio information are available. To build the score database, the symbolic music is converted to MIDI files with a constant tempo such that the overall duration of the file is comparable to a common performance of the piece.

In addition to these two datasets, the score database includes the complete BEETHOVEN piano sonatas, two symphonies by BEETHOVEN, and various other piano pieces. To this data there is no ground truth available, but this is irrelevant since they are not actively queried in our evaluation runs (it is solely there to provide more diversity and to make the task even harder for our algorithm). See Table 5.1 for an overview of the complete dataset.

5.5 TEMPO-INVARIANT FINGERPRINTING

For solving the task of score identification, we propose a symbolic fingerprinting approach (see Figure 5.2), based on the ideas of the well-known SHAZAM[1] audio identification algorithm [138]. From the symbolic scores local, tempo-invariant, and discriminative fingerprints are extracted and stored in a hash table. For querying the database the query is transcribed, resulting in a (noisy) list of notes. This list is then processed in the same way as the symbolic score representations, resulting in a set of performance fingerprints. Then, this set is compared to the score fingerprint database to identify the corresponding

1 http://www.shazam.com

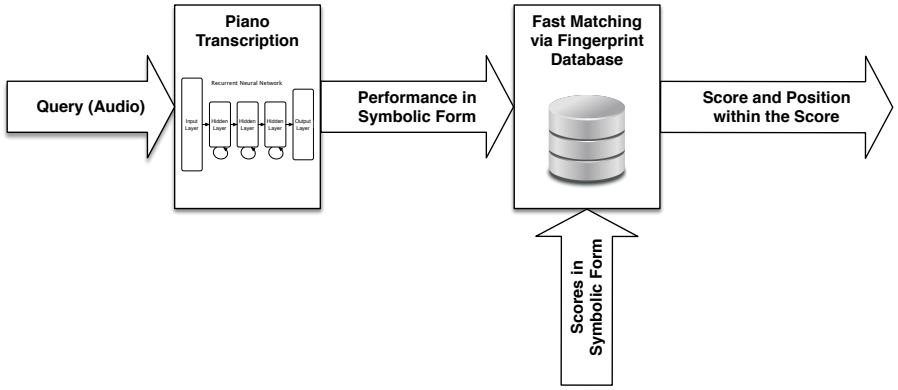


Figure 5.2: Overview of the proposed solution to score identification.

score (including the exact position within that score). Note that the tempo-invariance of this method is based on the assumption that the tempo *within* a query is relatively stable, i.e. it is mainly invariant to global tempo. For short queries of a few seconds, this assumption holds in the majority of cases. An approach taking care of local tempo differences, which allows for using this approach with longer queries, is presented in Section 5.7.

5.5.1 *Building the Score Fingerprint Database*

Before actually processing queries, the score database has to be built. Here, deadpan MIDI files are used as the basis for the score database. The duration of these MIDI files is comparable to the duration of a "typical" performance of the respective piece, but without encoded timing variations. From these files a simple ordered list of note events is extracted where for each note event the exact time in seconds and the pitch as MIDI note number is stored.

Next, for each piece fingerprint tokens are generated. In contrast to [138] we create them from three successive events according to some constraints (also see Figure 5.3), to make them tempo-invariant. Given a fixed event e we pair it with the first n_1 events with a distance of at least d seconds "in the future" of e. This results in n_1 event pairs. For each of these pairs we then repeat this step and again pair them with the n_2 future events with a distance of at least d seconds. This finally results in $n_1 * n_2$ event triplets. In our experiments, we used the values $d = 0.05$ seconds and $n_1 = n_2 = 5$. Also inspired by [138] we further constrain the pair creation steps to notes which are at most two octaves apart.

Given such a triplet consisting of the events e_1, e_2 and e_3 the time difference $td_{1,2}$ between e_1 and e_2 and the time difference $td_{2,3}$ between e_2 and e_3 are computed. To get a tempo independent fingerprint token we compute the time difference ratio of the time differences: $tdr = \frac{td_{2,3}}{td_{1,2}}$. This finally leads to a fingerprint token $[pitch_1 : pitch_2 : pitch_3 : tdr]$:

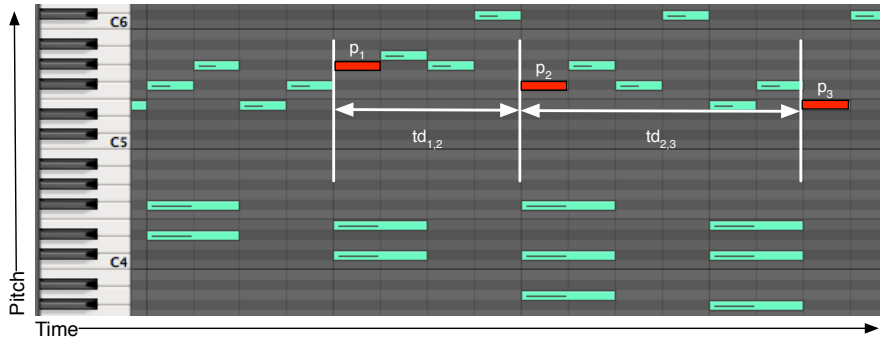


Figure 5.3: Fingerprint token generation (example of one generated token).

pieceID : *time* : $td_{1,2}$, where the hash key $[pitch_1 : pitch_2 : pitch_3 : tdr]$ can be stored in a 32 bit integer. Both *pieceID* and *time* (of the first event e_1 in the triplet) are needed to identify the corresponding position later on. The purpose of storing $td_{1,2}$ in the fingerprint token will be explained in the description of the search process itself (see Section 5.5.2 below).

The result of the score preprocessing is our score fingerprint database; a container of fingerprint tokens which provides quick access to the tokens via hash keys.

5.5.2 *Querying the Database*

Querying the database for matches via a short audio snippet is a two-step process. First, the query is transcribed into a list of note pitches with timestamps. Then, this list is converted into fingerprint tokens, which are used to query the database of score fingerprint tokens for consecutive sequences of matches.

Preprocessing: Transcribing the Query

Before querying the database, the query (an audio snippet of a performance) has to be transformed into a symbolic representation. The algorithm we use to transcribe musical note onsets from an audio signal is described in [20], which exhibits state of the art performance for this task. It uses a recurrent neural network to simultaneously detect the pitches and the onsets of the notes (see Figure 5.4 for an illustration of the algorithm). The author of the thesis was not involved in the development of this algorithm.

For its input, a discretely sampled audio signal is split into overlapping blocks before it is transferred to the frequency domain with two parallel Short-Time Fourier Transforms (STFT). Two different window lengths have been chosen to achieve both a good temporal precision and a sufficient frequency resolution for the transcription of the notes. Phase information of the resulting complex spectrogram is discarded and only the logarithm of the magnitude values is used for further

processing. To reduce the dimensionality of the input vector for the neural network, the spectrogram representation is filtered with a bank of filters whose frequencies are equally spaced on a logarithmic frequency scale and are aligned according to the MIDI pitches. The attack phase of a note onset is characterized by a rise of energy, thus the first order differences of the two spectrograms are used as additional inputs to the neural network.

The neural network consists of a linear input layer with 324 units, three bidirectional fully connected recurrent hidden layers, and a regression output layer with 88 units, which directly represent the MIDI pitches. Each of the hidden layers uses 88 neurons with hyperbolic tangent activation function. The use of bidirectional hidden layers enables the system to better model the context of the notes, which show a very characteristic envelope during their decay phase.

The network is trained with supervised learning and early stopping. The network weights are initialised with random values following a Gaussian distribution with mean 0 and standard deviation 0.1. Standard gradient descent with backpropagation of the errors is used for training. The network was trained on a collection of 281 piano pieces (this dataset is independent from any other data used in the thesis) recorded on various pianos, virtual and real (seven different synthesisers, an upright YAMAHA DISKLAVIER, and a BÖSENDORFER SE grand piano).

To make the transcriber applicable also in on-line scenarios, instead of preprocessing the whole piece of audio at a time, the signal is split into blocks of 11 frames centred around the actual frame. The use of 11 frames is a trade-off between keeping the system's ability to model the context of the notes and to keep the introduced delay at a minimum. In the current system the constant lag caused by the query preprocessing amounts to about 210 ms.

Table 5.2 shows the on-line transcription results for the complete test set described in Section 5.4. A note is considered to have been discovered correctly if its position is detected within a detection window of given size around the annotated ground truth position. As can be seen in the table, the results are far from perfect (though they are very good, considering the state of the art). If the proposed fingerprinting system is used in an off-line scenario, the use of an off-line transcription algorithm is an option to slightly improve the results.

Querying the Database

The transcription of the query results in a list of note pitches with timestamps. This list is then processed in the same way as described in Section 5.5.1 above to produce query tokens. Of course in this case no piece ID is known and furthermore each query starts at time 0. These query fingerprint tokens are used to query the database. The

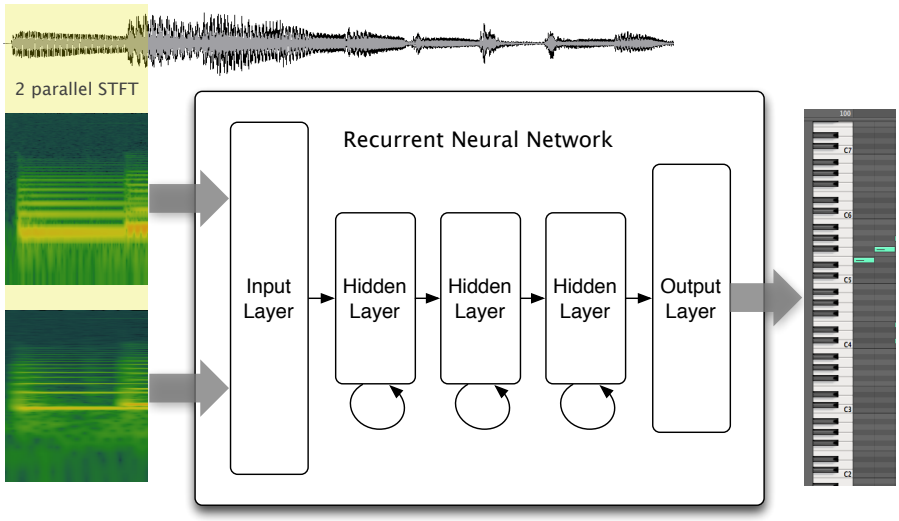


Figure 5.4: The transcription system, based on a recurrent neural network.

DETECTION WINDOW	PRECISION	RECALL	F-MEASURE
20 ms	0.586	0.489	0.533
40 ms	0.812	0.678	0.739
60 ms	0.851	0.710	0.774
80 ms	0.864	0.720	0.786
100 ms	0.869	0.725	0.790

Table 5.2: Results of the on-line transcription algorithm, for different detection window sizes.

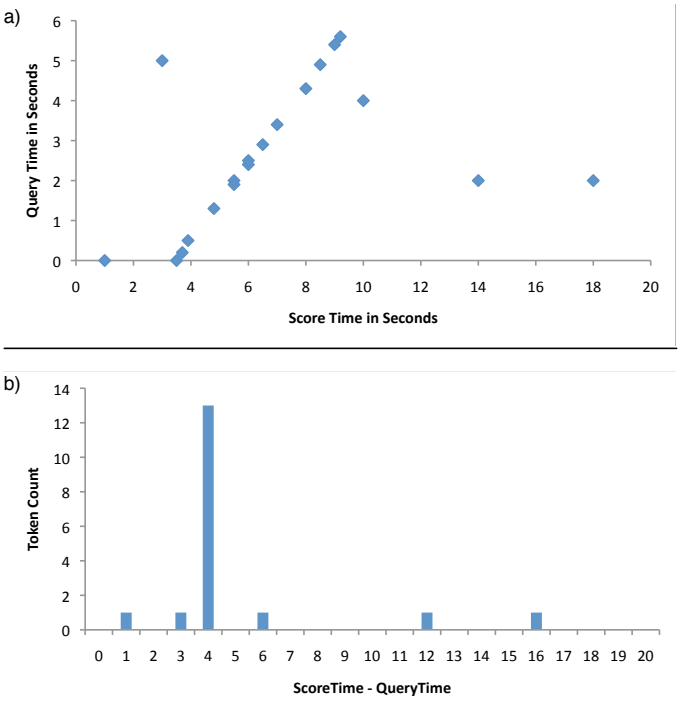


Figure 5.5: a) scatter plot of matching tokens and b) computed histogram for diagonal identification

method described below is again very much inspired by the audio fingerprinting method proposed in [138].

The general idea is to find regions in the score fingerprint database which share a continuous sequence of tokens with the query. To do so first all the score tokens which match the query tokens are extracted from the database. When plotted as a scatter plot against their respective time stamps (see Figure 5.5a) matches will be indicated by (rough) diagonals (i.e., these indicate that the query tokens match the score tokens over a period of time). As identifying these diagonals directly would be computationally expensive, we instead use a simpler method described in [138]. This is based on histograms (one for each piece in the score database, with a time resolution of one second) into which the matched tokens are sorted in a way such that peaks appear at the start points of these diagonals (i.e., the start point of a query, see Figure 5.5b). This is achieved by computing the bin to sort the token into as the difference between the time of the score token and time of the query token. The complete process will be explained in more detail below.

For each of the *query tokens* with $[qpitch_1 : qpitch_2 : qpitch_3 : qtdr] : qtime : qtd_{1,2}$ the following process is repeated. First, matching tokens are extracted from the score fingerprint database via the hash key. To allow for local tempo differences we permit the normalized time difference to be within $\frac{1}{4}$ of $qtdr$. This normally results in a large number of *score tokens* $[spitch_1 : spitch_2 : spitch_3 : stdr] : spieceID : stime : std_{1,2}$. Unfortunately, directly sorting these tokens into bin

QUERY LENGTH IN NOTES	10	15	20	25
CORRECT PIECE AS TOP MATCH	0.6	0.82	0.88	**0.91**
CORRECT PIECE MRR	0.68	0.86	0.91	**0.93**
CORRECT POSITION AS TOP MATCH	0.53	0.72	0.77	**0.79**
CORRECT POSITION MRR	0.60	0.79	0.83	**0.85**
MEAN QUERY LENGTH IN SECONDS	1.47	2.26	3.16	3.82
MEAN QUERY EXECUTION TIME (SEC.)	0.02	0.06	0.11	0.16

Table 5.3: Results for different query sizes of the *tempo-invariant* piece and score position identification algorithm on the test database at the piece level (upper half) and at the score position level (lower half). Each estimate is based on 2500 random audio queries. For both categories the percentage of correct detections at rank 1 and the mean reciprocal rank (MRR) are given. Additionally, the mean length of the query in seconds and the mean execution time for a query is shown.

$round(stime - qtime)$ of the histogram *spieceID* does not necessarily make sense because of the query possibly having a different tempo than expected by the score.

As an illustration let us assume a slower tempo for the query than for the respective score. Then the diagonal in Figure 5.5a would be steeper and when computing the bins via $round(stime - qtime)$ the first few tokens may fall into the correct bins. But soon the tokens, despite belonging to the same score position, would get sorted into lower bins instead.

Thus, we first try to adapt the timing by estimating the tempo difference between the *score token* and the *query token*. First we compute the tempo ratio of both tokens $r = \frac{std_{1,2}}{qtd_{1,2}}$ and then adapt the time of the query event when computing the bin to sort the token into: $bin = round(stime - qtime * r)$. This assumes that the local tempo is relatively stable, which is true for most short queries (see Section 5.7 for an extension, making this approach applicable to longer queries).

We now have a number of histograms, one for each score in the fingerprint database, and need a way of deciding on the most probable score position(s) (and, by implication, the most probable piece), for the query. We did experiments with different methods of computing the matching score but in the end simply taking the number of tokens in each bin as the score produced the best results.

5.5.3 *Evaluation*

An evaluation of the transcription stage (query preprocessing) was already presented in Section 5.5.2 above. As Table 5.2 shows, the results

of this stage are rather noisy. Still, the quality of the transcription is sufficient to be used with our robust fingerprinting technique.

The data for the evaluation of the identification task is summarised in Section 5.4. We tested the fingerprinting algorithms with different query lengths: 10, 15, 20 and 25 notes (automatically transcribed from the audio query). For each of the query lengths, we generated 2500 queries by picking random points in the performances of our test database, and used them as input for the proposed algorithms. Duplicate retrieval results (i.e. positions that have the exact same note content; also, duplicate piece IDs for the experiments on piece-level) are removed from the result set.

For the evaluation a score position X is considered correct if it marks the beginning (+/- 1.5 seconds) of a score section that is identical in note content to the "real" score situation corresponding to the query audio segment. We check the note content over a time span the length of the query, but at least 20 notes. This can be established as we have the correct alignment between performance time and score positions. The complex definition is necessary because musical pieces may contain repeated sections or phrases, and it is impossible for the system (or anyone else, for that matter) to guess the "true" one out of a set of identical passages matching the current performance snippet, given just that performance snippet as input. We acknowledge that a measurement of musical time in a score in terms of seconds is rather unusual. But as the MIDI tempos in our database are set in a meaningful way, this seemed the best decision to make errors comparable over different pieces, with different time signatures — it would not be very meaningful to, e.g. compare errors in bars or beats over different pieces.

Table 5.3 shows the results of the tempo-invariant fingerprinting algorithm on our dataset. Here, we present results for two categories: correctly identified pieces, and correctly identified piece and position in the score. For both categories we give the percentage of correct results at rank 1 and the mean reciprocal rank (MRR). The MRR is a standard measure for this kind of task, and for a sample of queries Q is computed as

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i} \tag{5.1}$$

where $rank_i$ refers to the rank position of the first relevant document for the i-th query.

As can be seen, even queries of only a length of 10 notes lead to a surprising number of correct position identifications, and already for a query length of 20 notes (which corresponds to a mean query duration of 3.16 seconds) the correct piece is returned as top match in almost 90% of the cases.

To show the tempo independence of our method not only via experiments, but also qualitatively, we performed a non-systematic experiment with data from different sources — videos by amateurs and by professional pianists, with differing recording qualities (including noisy old recordings and noisy amateur recordings) — for which we have no ground truth data. The general impression is that the system works well in these scenarios, but of course the performance worsens in the presence of noise. A video demonstration of the system is available on-line[2] (note especially the identification capabilities for heavily sped-up and slowed-down performances).

5.6 ADDING TRANSPOSITION INVARIANCE

In the algorithm described above, the pitches in the hash keys are represented as absolute values. Thus, if a performer decides to transpose a piece by an arbitrary number of semi-tones, any identification attempt by the algorithm must fail.

To overcome this problem, a simple, *relative* representation of the pitch values can be used, which makes the algorithm invariant to linear transpositions. Instead of using three absolute pitch values, we replace them by two differences, $pd_1 = pitch_2 - pitch_1$ and $pd_2 = pitch_3 - pitch_2$, resulting in a hash key $[pd_1 : pd_2 : tdr]$. For use in Section 5.6.1 below we additionally store $pitch_1$, the absolute pitch of the first note, in the token value.

In every other aspect the algorithm works in the same way as the purely tempo-invariant version described above. Of course this kind of transposition invariance cannot come free of cost, as the resulting fingerprints will not be as discriminative as before. This has two important direct consequences: (1) the retrieval accuracy will suffer, and (2) for every query a lot more matching tokens are found in the database, thus the runtime for each query increases (see Section 5.6.2).

5.6.1 *De-noising the Results: Token Verification*

To compensate for the loss in discriminative power we propose an additional step before accepting a database token as a match to the query. The general idea is taken from [78] and was first used in a music context by [131]. It is based on a verification step for each returned token that looks at the context within the query and the context at the returned position the database.

Each *score token* that was returned in response to a *query token* can be used to *project the query* (i.e. the notes identified from the query audio snippet by the transcription algorithm) to the possibly matching position in the score indicated by the *score token*. The intuition then is that at true matching positions we will find a majority of the notes

2 http://www.cp.jku.at/people/arzt/thesis/instant_music_identification.mp4

QUERY LENGTH IN NOTES	10	15	20	25
CORRECT PIECE AS TOP MATCH	0.30	0.40	**0.41**	0.40
CORRECT PIECE MRR	0.36	0.47	**0.50**	0.49
CORRECT POSITION AS TOP MATCH	0.23	**0.33**	0.32	0.32
CORRECT POSITION MRR	0.29	0.40	**0.41**	0.40
MEAN QUERY LENGTH IN SECONDS	1.47	2.26	3.16	3.82
MEAN QUERY EXECUTION TIME IN (SEC.)	0.10	0.32	0.62	0.91

Table 5.4: Results for different query sizes of the proposed *tempo- and transposition-invariant* piece and score position identification algorithm on the test database *without the verification step*.

from the query at their expected positions in the score. This will permit us to more reliably decide if the match of hash keys is a false positive or an actual match.

To do this, we need to compute the pitch shift and the tempo difference between the query and the potential position in the database. The pitch shift is computed as the difference of the $pitch_1$ of the matching *query token* and *score token*. The difference in tempo is computed as the ratio of $td_{1,2}$ of the two tokens. This information can now in turn be used to compute the expected time and pitch for each query note at the current score position hypothesis. We actually do not do this for the whole query, but only for a window of $w = 10$ notes, centred at the event e_1 of the query, and we exclude the notes e_1, e_2 and e_3 from this list (as they were already used to come up with the match in the first place).

We now take these w notes and check if they appear in the database as would be expected. In this search we are strict on the pitch value, but allow for a window of ± 100 ms with regards to the actual time in the database. If we can confirm that a certain percentage of notes from the query appears in the database as expected (in the experiments we used 0.8), we finally accept the query token as an actual match.

As this approach is computationally expensive, we actually compute the results in two steps: we first do "normal" fingerprinting without the verification step and only keep the top 5% of the results. We then perform the verification step on these results only and recompute the scores. On our dataset this effectively more than halves the computation time.

5.6.2 *Evaluation*

Again, as the basis for the evaluation the data presented in Section 5.4 is used. We evaluated the tempo- and transposition-invariant fingerprinting method both without (see Table 5.4) and with (see Table

QUERY LENGTH IN NOTES	10	15	20	25
CORRECT PIECE AS TOP MATCH	0.43	0.63	0.71	**0.75**
CORRECT PIECE MRR	0.49	0.69	0.76	**0.79**
CORRECT POSITION AS TOP MATCH	0.33	0.51	0.57	**0.60**
CORRECT POSITION MRR	0.41	0.59	0.66	**0.69**
MEAN QUERY LENGTH IN SECONDS	1.47	2.26	3.16	3.82
MEAN QUERY EXECUTION TIME IN (SEC.)	0.12	0.38	0.72	1.09

Table 5.5: Results for different query sizes of the proposed *tempo- and transposition-invariant* piece and score position identification algorithm on the test database *with the verification step*.

5.5) the verification step. In these tables, each estimate is based on 2500 random audio queries. The upper half shows recognition results on the piece level, the lower half on the score position level. For both categories the percentage of correct detections at rank 1 and the mean reciprocal rank (MRR) are given. Additionally, the mean length of the query in seconds and the mean execution time for a query is shown.

As expected, the use of pitch-invariant fingerprints without additional verification causes a big decrease in retrieval accuracy (compare Table 5.4 with Table 5.3). Furthermore, the loss in discriminative power of the fingerprint tokens also results in an increased number of tokens returned for every query, which has a direct influence on the runtime of the algorithm (last row in Table 5.4). The proposed verification step solves this problem, at least to some extent, and in our opinion makes the approach usable. On the downside, the runtime increases slightly.

We also tried to use the verification step with the original tempo-invariant algorithm but were not able to improve on the retrieval results. At least on our test data the tempo-invariant fingerprints are discriminative enough to mostly avoid false positives.

5.7 PROCESSING LONG QUERIES

The fingerprinting method described so far is mainly concerned with invariance regarding the *global tempo*. When applying this algorithm to our database with longer queries, *local tempo changes* (i.e. tempo changes within the query) prove to be problematic, because they break the cheap histogram approach that is used to determine continuous regions of matching tokens.

Instead of using computationally much more expensive methods for determining these regions, we propose to split longer queries into shorter ones and track the results of these sub-queries over time. This is known as shingling [24, 62]. Here, it is based on the assumption that in short queries the tempo is (quasi) stationary, and that a few exceptions

will not break the tracking algorithm we use. In our implementation, we split each query into sub-queries with a window size of $w = 15$ notes and a hop size of $h = 5$ notes and then feed each sub-query to the fingerprinter individually.

Each result of a sub-query (but at most the top 100 positions that are returned) is in turn fed to an on-line position hypothesis tracking algorithm. In our current proof-of-concept implementation we use a simple on-line rule-based multi-agent approach, inspired by the beat-tracking algorithm described in [40]. For a purely off-line retrieval task a non-causal algorithm will lead to even better results.

The basic idea is to create virtual agents for positions in the result sets. Each agent has a current hypothesis of the piece, the position within the piece and the tempo, and a score based on the results of the sub-queries. The agents are updated, if possible, with newly arriving data. In doing so, agents that represent positions that successively occur in result sets will accumulate higher scores than agents that represent positions that only occurred once or twice by chance, and are most probably false positives.

More precisely, we iterate over all sub-queries and perform the following steps in each iteration:

NORMALISE SCORING OF POSITION First the scores of the positions in the result set of the sub-query are normalised by dividing them by their median. This ensures that each iteration has approximately the same influence on the tracking process.

UPDATE AGENTS For every agent, we look for a matching position in the result set of the sub-query (i.e. a position that approximately fits the extrapolated position of the agent, given the old position, the tempo, and the elapsed time). The position, the tempo and the score of the agent are updated with the new data from the matching result of the sub-query. If we do not find a matching position in the result set, we update the agent with a score of 0, and the extrapolated position is taken as the new hypothesis. If a matching position is found, the accumulated score is updated in a fashion such that scores from further in the past have a smaller impact than more recent ones. Each agent has a ring buffer s of size 50, in which the scores of the individual sub-queries are being stored. The accumulated score of the agent is then calculated as $score_{acc} = \sum_{i=1}^{50} \frac{s_i}{1+\log i}$, where s_1 is the most recent score.

CREATE AGENTS Each sub-query result that was not used to update an existing agent is used to initialise a new agent at the respective score position (i.e. in the first iteration up to 100 agents are created).

	NO TRACKING		TRACKING	
QUERY LENGTH IN NOTES	50	100	50	100
CORRECT PIECE AS TOP MATCH	0.95	0.96	0.98	**1**
CORRECT PIECE MRR	0.97	0.98	0.99	**1**
CORRECT POSITION AS TOP MATCH	0.78	0.73	0.87	**0.88**
CORRECT POSITION MRR	0.85	0.81	0.89	**0.90**
MEAN QUERY LENGTH IN SECONDS	7.62	15.03	7.62	15.03
MEAN QUERY EXEC. TIME (SEC.)	0.42	0.92	0.49	1.08

Table 5.6: Results of the proposed tracking algorithm on the test database for the *tempo-invariant algorithm*. Each estimate is based on 2500 random audio queries.

	NO TRACKING		TRACKING	
QUERY LENGTH IN NOTES	50	100	50	100
CORRECT PIECE AS TOP MATCH	0.81	0.79	0.92	**0.98**
CORRECT PIECE MRR	0.85	0.82	0.94	**0.99**
CORRECT POSITION AS TOP MATCH	0.64	0.59	0.77	**0.83**
CORRECT POSITION MRR	0.72	0.66	0.82	**0.86**
MEAN QUERY LENGTH IN SECONDS	7.62	15.03	7.62	15.03
MEAN QUERY EXEC. TIME (SEC.)	2.71	6.11	3.21	7.09

Table 5.7: Results of the proposed tracking algorithm on the test database for the *tempo- and pitch-invariant algorithm*. Each estimate is based on 2500 random audio queries.

REMOVE OBSOLETE AGENTS Finally, agents with low scores are removed. In our implementation we simply remove agents that are older than 10 iterations and are not part of the current top 25 agents.

At each point in time the agents are ordered by $score_{acc}$ and can be seen as hypotheses about the current position in the database of pieces. Thus, in the case of a single long query, the agents with the highest accumulated scores are returned in the end. In an on-line scenario, where an audio stream is constantly being monitored by the fingerprinting system, the current top hypotheses can be returned after each performed update (i.e. after each processed sub-query).

5.7.1 *Evaluation*

As before, the evaluation is performed on the data described in Section 5.4. Tables 5.6 and 5.7 give the results on slightly longer queries for the original *tempo-invariant* and the *tempo- and transposition-invariant* algorithm. For the category "No Tracking", the query was fed directly to the fingerprinting algorithm. For "Tracking", the queries were split into sub-queries with a window size of 15 notes and a hop size of 5 notes, and the individual results were tracked by our proof-of-concept *multi-agent* approach. Each estimate is based on 2500 random audio queries. The upper half shows recognition results on the piece level, the lower half on the score position level. For both categories the percentage of correct detections at rank 1 and the mean reciprocal rank (MRR) are given. Additionally, the mean length of the query in seconds and the mean execution time for a query is shown.

As can be seen, for the detection of the exact position in the score, using no tracking, the results based on queries with length 100 notes are worse than those for queries with only 50 notes, i.e. more information leads to worse results. This is caused by local tempo changes within the query, which break the histogram approach for finding sequences of matching tokens.

As shown on the right hand side for both fingerprinting types, the approach of splitting longer queries into shorter ones and tracking the results takes care of this problem. Please note that for the tracking approach we check if the position hypotheses after the last tracking step match the correct position in the score. Thus, as this is an on-line algorithm, we are not interested in the start position of the query in the score, but in the endpoint, i.e. if the query was tracked successfully, and the correct *current* position is returned. Even the causal approach leads to a high percentage of correct results with both the original and the tempo- and pitch-invariant fingerprinting algorithm. Most of the remaining mistakes happen because (very) similar parts within one and the same piece are confused.

5.8 CONCLUSIONS

The proposed algorithms are useful in a wide range of applications. As a retrieval algorithm it enables fast and robust (inter- and intra-document) searching and browsing in large collections of musical scores and corresponding performances. Furthermore, we believe that the algorithm is not limited to retrieval tasks in classical music, but may be of use for cover version identification in general, and possibly many other tasks. For example, it was already successfully applied in the field of symbolic music processing to find repeating motifs and sections in complex musical scores [27].

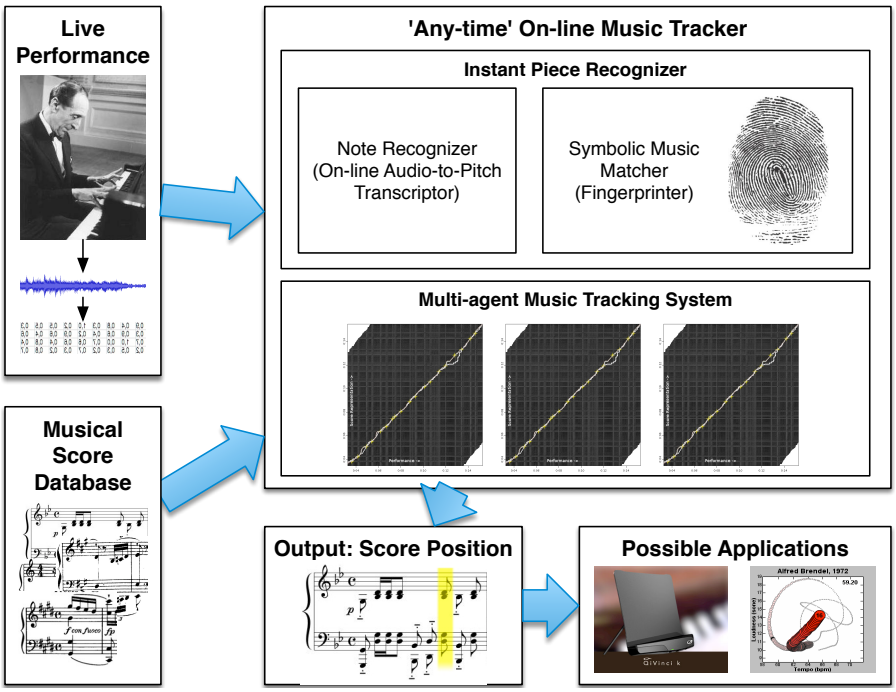


Figure 5.6: The Any-time Music Tracker. Picture of Werner GOEBL (c) Clemens CHMELAR. The page turner was formerly produced by Qidenus (http://qidenus.com), the "performance worm" was produced by Werner GOEBL

In the context of this thesis the algorithm is used an on-line music tracking scenario:. the fingerprinting algorithm is used to retrieve probable score positions from a database of scores, which are then used as input to an 'any-time' music tracking system (see Section 5.9).

5.9 PROTOTYPICAL IMPLEMENTATION

In combination with the music tracking algorithm, we built a very flexible system that is able to recognise arbitrary pieces of classical piano music, identify the position in the score and track the progress of the performer. A sketch of this system is shown in Figure 5.6. The system is provided with scores in symbolic form as well as audio feature for the tracking. The live performance is transcribed and processed by the fingerprinter. Multiple instances of the tracking algorithm try to align the live audio to the score positions suggested by the fingerprinter. The system determines a final hypothesis via the alignment costs. At any time it outputs a position in some score in the database.

The implementation with the help of the FLOWER framework is sketched in 5.7. The audio input is processed in two threads in parallel, computing features for the tracking and for the music transcription modules. The RECURRENT NEURAL NETWORK transcribes the incoming

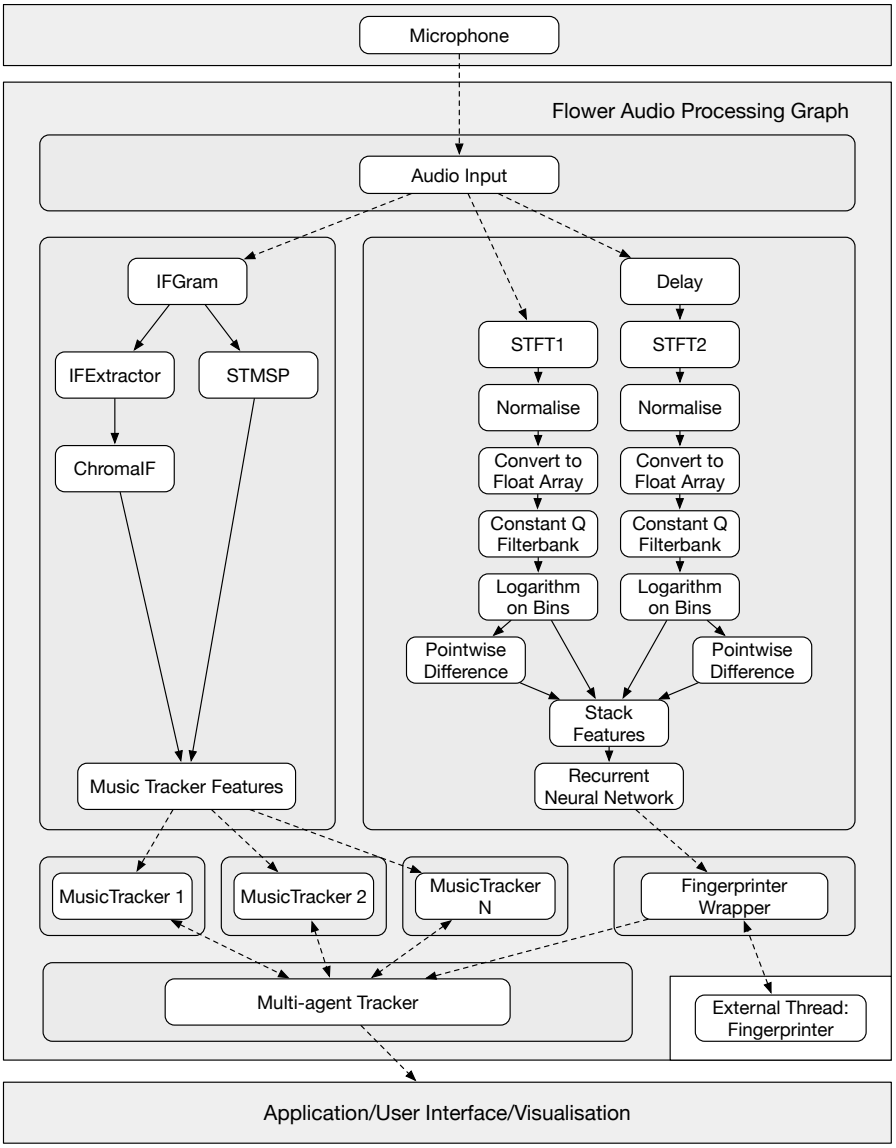


Figure 5.7: The Any-time Tracker, as implemented in the Flower framework.

audio stream and hands the data over to a processing that wraps the fingerprinter, which has access to a fingerprint database computed from the scores. The fingerprinter is running as an external thread, receives newly transcribed notes and returns position hypotheses. The MULTI-AGENT TRACKER processing is acting as the central unit that receives data from the fingerprinter and manages the instances of the music tracking algorithm accordingly (MUSIC TRACKER 1 to MUSIC TRACKER N, each running in a separate thread). It receives the position hypotheses and uses them to initialise new instances of the music tracking algorithm at the respective position. It also receives the positions from the trackers, including their alignment costs over the last couple of seconds. Based on this information, it decides which trackers to reset, which trackers to keep, and also which tracker to trust at any given time. The position of this tracker is returned as the current score position in the whole database of scores. We have demonstrated this system at various occasions. A full list, including links to videos of live demonstrations, can be found in Chapter 6. A short demonstration video is also available at `http://www.cp.jku.at/people/arzt/thesis/fingerprinting_transposition.mp4`.

Part III

LIVE DEMONSTRATIONS AND CONCLUSIONS

6 REAL-LIFE APPLICATIONS

In the previous chapters research towards a complete classical music companion was presented. During the work on this thesis the focus was not only on research, but to some extent also on bringing this technology to life and presenting it in live music environments.

In Section 6.1, three different algorithm configurations are described that we used for the live demonstration. Then, Section 6.2 lists the live showings of the prototypes presented in chronological order.

6.1 TRACKING ALGORITHMS USED FOR LIVE DEMONSTRATIONS

Over the years, we demonstrated three different kinds of systems, at different stages and in various settings.

MUSIC TRACKER The *music tracker* was the starting point for all demonstrations. It is based on the work described in Chapter 3 (for notes regarding the implementation see Section 3.7 for a description of the implementation). In contrast to the other demonstration prototypes, this one was at first implemented in Java, and then later on, around 2010, re-implemented in the Flower framework. Although theoretically usable for other kinds of music too, it was used exclusively for demonstrations with live piano performances.

MULTI-AGENT TRACKER The *multi-agent tracker* is based on the work described in Chapter 4 (for notes regarding the implementation see Section 4.5). It is designed as a more robust version of the *music tracker*. The tracking is based on a number of automatically preprocessed performances of the piece in question. While also applicable for piano or other kinds of classical music, this system was developed with orchestral music in mind and was used in real concert settings.

PIANO COMPANION The *piano companion* is based on the work described in Chapter 5 (for notes regarding the implementation see Section 5.9). This system listens to a live piano performance, detects the piece and the position within the piece in a matter of seconds and then tracks the performer over time. It continues to track multiple hypotheses in the background and is able to detect any jumps within a piece or to other pieces.

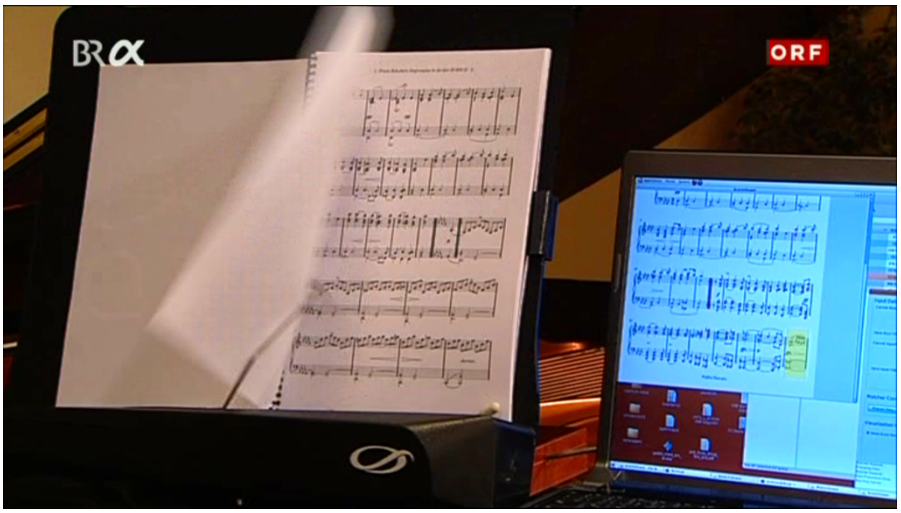


Figure 6.1: An impression of the page turner as shown in the documentary.

6.2 LIVE DEMONSTRATIONS

The following list gives an overview of live demonstrations featuring technology that was developed during the course of this thesis.

TV Documentary: Stil und Interpretation - Fingerprints der Musik, Music Tracker, first shown on BR Alpha in January 2010

The first prototype was featured in a TV documentary that was produced in summer 2009 (see Figure 6.1). Gerhard WIDMER, the supervisor of this thesis, is shown playing the piano (Impromptu D.935 No.2 by Franz SCHUBERT) while the score follower is tracking the progress. This is visualised on screen as a marker in the sheet music, and also via a page turning device that was connected to the computer and thus was able to turn the pages for the pianist automatically at the appropriate moments. The page turning device was developed by the Viennese company QIDENUS[1], and usually is controlled manually by the musician via a pedal. We built a small adapter to connect it to the computer via USB (see Figure 6.2). This system was used to demonstrate intelligent music technology to different audiences, ranging from school kids to fellow researchers.

Start-Wittgenstein Gala, Vienna, Music Tracker, March 2010

The START-WITTGENSTEIN GALA is an event organised by the Austrian funding agency FWF[2], where the winners of Austria's most prestigious

1 http://www.qidenus.com; The press release for the first public demonstration of their page turning device is available at http://www.ots.at/presseaussendung/OTS_20060801_OTS0110/ (in German).

2 https://www.fwf.ac.at

Figure 6.2: The QIDENUS QIVINCI page turner (left) and the adapter we built to connect it to the computer via USB (right).

research awards are celebrated. During this event Gerhard WIDMER, who was one of the two winners of the WITTGENSTEIN AWARD that year, gave a talk that included a demonstration of the score following technology. Concert pianist Veronika TRISKO played SCHUBERT's Impromptu D899 No. 3, with the music tracker following the progress. The page turning device was connected to the computer and turned the pages for the pianist automatically at the appropriate moments (see Figure 6.3 for an impression of the performance).

Vienna Talk on Music Acoustics, Vienna, Music Tracker, October 2010

This was the first showing of our score following technology at a scientific conference. During the keynote given by Gerhard WIDMER ("On the use of intelligent computational methods for music performance analysis"), Werner GOEBL played the Intermezzo of the FASCHINGSSCHWANK Op. 26 by Robert SCHUMANN, with the computer tracking the progress live and showing the position in the score. Again, the page turning device was included in the demonstration (see Figure 6.4). A video of this performance is available on-line[3].

ICT Conference of the European Union, Piano Companion, November 2013

At the ICT 2013, in the context of the EU FP7 PHENICX project[4] the piano companion was shown for the first time. In addition to demonstrations at the exhibition booth, the piano companion was featured in two live shows on a stage in the conference venue. At this occasion Cynthia LIEM (see Figure 6.5), who is both a professional pianist and a researcher in music information retrieval at TU DELFT, played the piano.

3 http://www.cp.jku.at/people/arzt/thesis/pageturner_werner_goebl.mov
4 http://phenicx.upf.edu

Figure 6.3: Veronika Trisko demonstrating the Automatic Page Turner at the Start-Wittgenstein Gala 2010. Picture (c) FWF / Hans Schubert

Figure 6.4: Werner Goebl demonstrating the Automatic Page Turner at the Vienna Talk on Music Acoustics 2010.

Figure 6.5: Cynthia LIEM. Picture (c) Marco BORGGREVE

The setup was as follows: The computer has knowledge of a large database of piano scores, including all the 32 BEETHOVEN piano sonatas, most of CHOPIN's solo piano works, most of the MOZART piano sonatas. At that time, the total database consisted of more than 1,000,000 notes. Cynthia LIEM's repertoire covers large parts of the database. To demonstrate the capabilities of the piano music companion, she switched between different pieces, jumping to any position, which the piano music companion recognised within a few seconds and visualised the correct position in the sheet music. This was done in a controlled environment with predefined, rehearsed jumps, but also without a safety net, by handing out sheet music to the audience and letting them decide on which piece she is supposed to play. Short clips of this demonstration are available on-line[5].

AES Semantic Audio Conference, Piano Companion, London, January 2014

At the AES Semantic Audio Conference the piano companion was demonstrated in front of a scientific audience (again with the help of Cynthia LIEM). In the audience there were a few (amateur-)pianists, who were given the opportunity to try the piano companion themselves. At the conference, we received the best demonstration award.

5 http://www.cp.jku.at/people/arzt/thesis/vilnius.mp4 and http://www.cp.jku.at/people/arzt/thesis/decomposing_mozart.mov

Figure 6.6: Werner Goebl and Andreas Arzt (background) demonstrating the Piano Companion at the OFAI Jubelfeier in December 2014. Picture (c) Thomas Grill

Prestigious Applications of Intelligent Systems Conference, held in conjunction with the European Conference on Artificial Intelligence, Music Tracker and Piano Companion, Prague, August 2014

At the ECAI 2014 both the music tracker, including the automatic page turner, and the piano companion were demonstrated. At this occasion Gerhard Widmer and Cynthia Liem played the piano. The contribution was awarded the best demonstration award at the conference.

International Society for Music Information Retrieval Conference, Piano Companion, Taipei, November 2014

The International Symposium on Music Information Retrieval Conference (ISMIR) is the main conference of the music information retrieval community. At the demo session, the piano companion was presented. This is a very informal session which gave fellow researchers the opportunity to try the companion themselves.

OFAI 30th Anniversary Celebrations, Piano Companion, Vienna, December 2014

The Austrian Research Institute for Artificial Intelligence (OFAI) celebrated its 30th anniversary with a gala in the great festival hall of the Austrian Academy of Sciences. The piano companion was demonstrated with Werner Goebl at the piano (see 6.6).

Concert by the Concertgebouw Orchestra under Semyon Bychkov, Multi-agent Tracking, Amsterdam, February 2015

The "adventure" at the CONCERTGEBOUW definitely was the highlight during the work on the thesis. During a regular concert, the multi-agent tracking algorithm listened to the live performance of a full orchestra playing the ALPENSINFONIE by Richard STRAUSS, and computed the position in the sheet music. This information was used to provide a test audience with synchronised information on their mobile phones and tablets (the sheet music, textual information which was prepared by a musicologist, and artistic videos telling the story of the piece). A detailed description of this event can be found in Section 4.3. Short videos of this concert and of a related internal test at the CONCERTGEBOUW are avalable on-line[6].

Science Days at TU Delft, Delft, Piano Companion, May 2015

A demo of the piano companion was included in a talk about music and technology by Cynthia LIEM at the A DAY OF WONDER festival at TU DELFT.

IMAGINE Conference, Vienna, Piano Companion, June 2015

At IMAGINE 2015 in Vienna I was invited to give a talk about music tracking technology, including live demos of the music tracker, as well as the piano companion, with Werner GOEBL at the piano.

International Joint Conference on Artificial Intelligence, Piano Companion, Buenos Aires, July 2015

At the opening celebrations of the IJCAI the piano companion was presented as a special music act (with Cynthia LIEM on the piano). The demonstration included both the music tracker and the piano companion demo, including interaction with the audience.

ICT Conference of the European Union, Multi-agent Tracker, Lisbon, October 2015

At the ICT conference some of the outcomes of the PHENICX project (which was mainly concerned with orchestral music) were presented. For practical reasons, at this occasion the orchestra was "emulated" on the piano (i.e. we used the multi-agent tracker in the same way as with orchestral music, but with piano as input).

Cynthia LIEM played a piano version of the overture of BEETHOVEN's THE CREATURES OF PROMETHEUS. The multi-agent tracker followed her

6 http://www.cp.jku.at/people/arzt/thesis/alpensinfonie_1.mp4, http://www.cp.jku.at/people/arzt/thesis/alpensinfonie_2.mp4, http://www.cp.jku.at/people/arzt/thesis/alpensinfonie_3.mp4 and http://www.cp.jku.at/people/arzt/thesis/chabrier.mp4

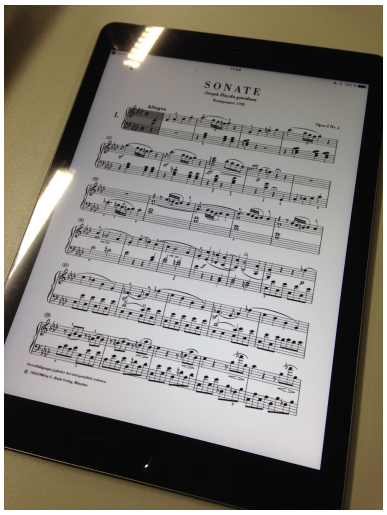


Figure 6.7: The music tracker on the iPad (Score (c) HENLE VERLAG)

performance and controlled visualisations prepared by our project partner, the MUSIC TECHNOLOGY GROUP at UNIVERSITAT POMPEU FABRA (UPF).

International Symposium on Music Information Retrieval Conference, Multi-agent Tracker, Malaga, October 2015

At ISMIR 2015 in Malaga, an event was organised that gave scientists and musicians the opportunity to present their work to the public. During this event the PHENICX project, together with a Spanish youth orchestra, presented the orchestra tracking technology as it was used in the CONCERTGEBOUW Amsterdam earlier that year. This time, visualisations by the project partner UPF were synchronised to the music via the multi-agent tracker. The synchronisations included the sheet music in multiple forms, an activity map of the orchestra and gestures by the conductor, and were projected on a big screen located above the orchestra.

Public Demonstration of the Phenicx Project, Piano Companion and Multi-agent Tracker, Barcelona, March 2016

Towards the end of the PHENICX project, a final event was organised to present the results of the project to the general public. At this occasion, the piano companion, including audience interaction, and the tracking and visualisation of orchestral music (again the orchestra had to be "emulated" by a piano) were shown.

Addendum: Music Tracking on the iPad

For demonstration purposes the music tracker was also ported to iOS, the operating system of APPLE's mobile devices. This was done by Martin GASSER (see Figure 6.7).

7 CONCLUSIONS AND OUTLOOK

Music tracking algorithms are already robust enough to be used in real life applications. This has been demonstrated by a number of research groups and companies. Within the PHENICX[1] project we applied music tracking at the CONCERTGEBOUW. The PHILADELPHIA ORCHESTRA[2] has used a similar algorithm during some of their concerts, and automatic accompaniment systems have already been showcased in concert halls such as the Royal Albert Hall[3]. There are also first attempts at building interactive piano tutors that help beginners with learning the piano[4].

The thesis contributes to this field by making music tracking more robust and more flexible. Starting from a simple and fragile algorithm, systems are built that work well with complex romantic piano music and orchestral music. In addition, methods are presented that take music tracking to the next level and enable following the performer flexibly based on a big database of piano music. To lift the limitation of the presented music identification system to piano music, and make it work with arbitrary instruments and higher levels of polyphony (e.g. orchestral music) still remains an open point. Essentially, the limitation is not caused by the algorithm itself, but by the absence of appropriate training data for the underlying audio transcription algorithm.

The context of the thesis is the "vision" of a Complete Classical Music Companion, a "support system" for performers and listeners that flexibly detects music and provides synchronised information. While the thesis presents algorithms and technology that can be used as a basis for such an application, it does not tackle problems like data preparation and the presentation of meaningful visualisations to the user.

Especially data preparation remains a big problem, as even just preparing the sheet music for tracking is a very laborious task. Only a small percentage of the sheet music is available in symbolic form. For music publishers it is time consuming, error prone and without any financial gain to redo the typesetting of most pieces of classical music and thus they continue to use the old printing plates as long as possible. And, even if clean, digitally typeset sheet music is available, the preparation of the data for a synchronisation algorithm is not an automatic process, as many problems can arise that might need manual intervention (e.g. conversion errors, non-standard annotations of transpositions of instruments, transpositions that are not annotated).

1 http://phenicx.upf.edu
2 http://livenote.philorch.org
3 http://www.bbc.co.uk/events/evrmbp
4 http://tonara.com

The most flexible and promising way forward actually seems to be to try to work with images of sheet music directly. Unfortunately, optical music recognition [120] is still not at a level where it can read complex scores reliably enough. Recently, we published a deep learning approach to music tracking that works directly on sheet music (see [44, 45]). This bypasses the complex and laborious task of score data preparation. This is still early work, and the tracking and identification abilities are not comparable to the complex systems presented in this thesis. In the future, we plan to improve this approach and make it work reliably on more complex music. It is a data-driven approach, and thus requires large amounts of (annotated) training and testing data, which is not readily available. Hence, besides advances regarding the algorithm itself, the collection and preparation of sufficient high quality training data will be paramount. We plan to create this data by collecting large amounts of sheet music, symbolic information (if available) and performances, and aligning these kinds of data to each other. Essentially, this is cross-modal, large scale multiple sequence alignment, and we hope that we can improve on the state of the art of the individual alignment tasks by leveraging the information we gain with each new datapoint and each additional alignment.

Regarding the Complete Classical Music Companion, there is also the problem of preparing the data for visualisation purposes. Showing the sheet music is relatively straight forward, but presenting more sophisticated information — like a visualisation of the structure of a piece, a summary of important motifs, or a characterisation of a performance — needs in-depth annotations of a piece of music. At the time of writing, there are no algorithms that are able to extract complex musical concepts like these reliably from the score or from the audio signal of a performance (see for example [141] for a discussion of what computers so far do *not* understand in music). From an application standpoint, this can be solved via manual annotations by experts, possibly combined with a crowdsourcing approach — just covering the, e.g. 1,000 most "important" pieces of classical music in more depth (i.e. in addition to showing the synchronised score) would result in a very useful program. From the research point of view, for extracting such complex concepts data-driven methods seem to be most promising, and we hope that we can support this research by creating parts of the necessary data via our music synchronisation technologies.

BACK MATTER

BIBLIOGRAPHY

[1] Xavier Amatriain, Pau Arumi, and David Garcia. "CLAM: A Framework for Efficient and Rapid Development of Cross-platform Audio Applications". In: *Proceedings of the ACM International Conference on Multimedia*. Santa Barbara, USA, 2006, pp. 951–954.

[2] Pau Arumi. "Real-time Multimedia Computing on Off-The-Shelf Operating Systems: From Timeliness Dataflow Models to Pattern Languages". PhD thesis. Barcelona, Spain: Universitat Pompeu Fabra, 2009.

[3] Andreas Arzt. "Score Following with Dynamic Time Warping: An Automatic Page Turner". MA thesis. Vienna, Austria: Vienna University of Technology, 2008.

[4] Andreas Arzt, Sebastian Böck, Sebastian Flossmann, Harald Frostel, Martin Gasser, Cynthia C.S. Liem, and Gerhard Widmer. "The Piano Music Companion". In: *Proceedings of the Conference on Prestigious Applications of Intelligent Systems (PAIS)*. Prague, Czech Republic, 2014, pp. 1221–1222.

[5] Andreas Arzt, Sebastian Böck, Sebastian Flossmann, Harald Frostel, Martin Gasser, and Gerhard Widmer. "The Complete Classical Music Companion V0.9". In: *Proceedings of the 53rd AES Conference on Semantic Audio*. London, England, 2014.

[6] Andreas Arzt, Sebastian Böck, and Gerhard Widmer. "Fast Identification of Piece and Score Position via Symbolic Fingerprinting". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Porto, Portugal, 2012, pp. 433–438.

[7] Andreas Arzt, Harald Frostel, Thassilo Gadermaier, Martin Gasser, Maarten Grachten, and Gerhard Widmer. "Artificial Intelligence in the Concertgebouw". In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. Buenos Aires, Argentina, 2015, pp. 2424–2430.

[8] Andreas Arzt, Werner Goebl, and Gerhard Widmer. "Flexible Score Following: The Piano Music Companion and Beyond". In: *Proceedings of the Vienna Talk on Musical Acoustics (VITA)*. Vienna, Austria, 2015, pp. 220–223.

[9] Andreas Arzt, Cynthia C.S. Liem, and Gerhard Widmer. "A Tempo- and Transposition-invariant Piano Music Companion". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Late Breaking / Demo*. Taipeh, Taiwan, 2014.

[10] Andreas Arzt and Gerhard Widmer. "Robust Real-time Music Tracking". In: *Proceedings of the Vienna Talk on Musical Acoustics (VITA)*. Vienna, Austria, 2010, pp. 5–8.

[11] Andreas Arzt and Gerhard Widmer. "Simple Tempo Models for Real-time Music Tracking". In: *Proceedings of the Sound and Music Computing Conference (SMC)*. Barcelona, Spain, 2010.

[12] Andreas Arzt and Gerhard Widmer. "Towards Effective 'Any-Time' Music Tracking". In: *Proceedings of the Starting AI Researchers' Symposium (STAIRS)*. Lisbon, Portugal, 2010, pp. 24–36.

[13] Andreas Arzt and Gerhard Widmer. "Real-time Music Tracking using Multiple Performances as a Reference". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Málaga, Spain, 2015, pp. 357–363.

[14] Andreas Arzt, Gerhard Widmer, Sebastian Böck, Reinhard Sonnleitner, and Harald Frostel. "Towards a Complete Classical Music Companion". In: *Proceedings of the European Conference on Artificial Intelligence (ECAI)*. Montpellier, France, 2012, pp. 67–72.

[15] 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)*. Patras, Greece, 2008, pp. 241–245.

[16] Andreas Arzt, Gerhard Widmer, and Simon Dixon. "Adaptive Distance Normalization for Real-time Music Tracking". In: *Proceedings of the European Signal Processing Conference (EUSIPCO)*. Bucharest, Romania, 2012, pp. 2689–2693.

[17] Andreas Arzt, Gerhard Widmer, and Reinhard Sonnleitner. "Tempo- and Transposition-invariant Identification of Piece and Score Position". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Taipeh, Taiwan, 2014, pp. 549–554.

[18] Shumeet Baluja and Michele Covell. "Waveprint: Efficient Wavelet-based Audio Fingerprinting". In: *Pattern Recognition* 41.11 (2008), pp. 3467–3480.

[19] Sebastian Böck, Andreas Arzt, Florian Krebs, and Markus Schedl. "Online Real-Time Onset Detection with Recurrent Neural Networks". In: *Proceedings of the International Conference on Digital Audio Effects (DAFx)*. York, United Kingdom, 2012.

[20] Sebastian Böck and Markus Schedl. "Polyphonic piano note transcription with recurrent neural networks". In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Kyoto, Japan, 2012, pp. 121–124.

[21] Pedro Cano, Eloi Batlle, Ton Kalker, and Jaap Haitsma. "A Review of Algorithms for Audio Fingerprinting". In: *Proceedings of the IEEE International Workshop on Multimedia Signal Processing (MMSP)*. St. Thomas, Virgin Islands, USA, 2002, pp. 169–173.

[22] Pedro Cano, Alex Loscos, and Jordi Bonada. "Score-Performance Matching Using HMMs". In: *Proceedings of the International Computer Music Conference ICMC*. Beijing, China, 1999.

[23] J. J. Carabias-Orti, F. J. Rodriguez-Serrano, P. Vera-Candeas, N. Ruiz-Reyes, and F. J. Canãdas-Quesada. "An Audio to Score Alignment Framework using Spectral Factorization and Dynamic Time Warping". In: *Proceedings of the International Society for Music Information Retrieval Conference*. Málaga, Spain, 2015, pp. 742–748.

[24] Michael A. Casey and Malcolm Slaney. "Song Intersection by Approximate Nearest Neighbor Search". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Victoria, Canada, 2006, pp. 144–149.

[26] Tom Collins, Daniel A. Abrams, Rohan Chandra, Christina Young, Andreas Arzt, and Vinod Menon. "Neural tracking of musical motives revealed by a combination of fMRI and music information retrieval techniques". In: *Proceedings of the International Conference on Music Perception and Cognition (ICMPC)*. Seoul, South Korea, 2014, p. 55.

[27] Tom Collins, Andreas Arzt, Sebastian Flossmann, and Gerhard Widmer. "SIARCT-CFP: Improving Precision and the Discovery of Inexact Musical Patterns in Point-set Representation". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Curitiba, Brazil, 2013, pp. 549–554.

[28] Tom Collins, Andreas Arzt, Harald Frostel, and Gerhard Widmer. "Using Geometric Symbolic Fingerprinting to Discover Distinctive Patterns in Polyphonic Music Corpora". In: *Computational Music Analysis*. Ed. by David Meredith. Springer International Publishing, 2016, pp. 445–474.

[29] Arshia Cont. "Realtime Audio to Score Alignment for Polyphonic Music Instruments, using Sparse Non-Negative Constraints and Hierarchical HMMS". In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Toulouse, France, 2006, pp. 245–248.

[30] Arshia Cont. "A Coupled Duration-Focused Architecture for Real-Time Music-to-Score Alignment". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.6 (2010), pp. 974–987.

[31] Arshia Cont, Diemo Schwarz, Norbert Schnell, and Christopher Raphael. "Evaluation of Real-Time Audio-to-Score Alignment". In: *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*. Vienna, Austria, 2007, pp. 315–316.

[32] Stephen S. Cox. "Speech and Language Processing". In: ed. by C. Wheddon and R. Linggard. London, UK: Chapman & Hall, Ltd., 1990. Chap. Hidden Markov Models for Automatic Speech Recognition: Theory and Application, pp. 209–230.

[33] Philippe Cuvillier and Arshia Cont. "Coherent time modeling of Semi-Markov models with application to real-time audio-to-score alignment". In: *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*. Reims, France, 2014, pp. 1–6.

[34] 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)*. Chania, Crete, Greece, 2008, pp. 205–208.

[35] 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". In: *International Journal on Digital Libraries: Special Issue on Music Digital Libraries* 12.2-3 (2012), pp. 53–71.

[36] Roger B. Dannenberg. "An On-Line Algorithm for Real-Time Accompaniment". In: *Proceedings of the International Computer Music Conference (ICMC)*. Paris, France, 1984, pp. 193–198.

[37] Roger B. Dannenberg. "Aura II: Making Real-Time Systems Safe for Music". In: *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*. Hamamatsu, Japan, 2004, pp. 132–137.

[38] Roger B. Dannenberg and Ning Hu. "Polyphonic Audio Matching for Score Following and Intelligent Audio Editors". In: *Proceedings of the International Computer Music Conference (ICMC)*. San Francisco, USA, 2003, pp. 27–34.

[39] Roger B. Dannenberg, Zeyu Jin, Nicolas E. Gold, Octav-Emilian Sandu, Praneeth N. Palliyaguru, Andrew Robertson, Adam Stark, and Rebecca Kleinberger. "Human-Computer Music Performance: From Synchronized Accompaniment to Musical Partner". In: *Proceedings of the Sound and Music Computing Conference (SMC)*. Stockholm, Sweden, 2013, pp. 277–283.

[40] Simon Dixon. "Automatic extraction of tempo and beat from expressive performances". In: *Journal of New Music Research* 30.1 (2001), pp. 39–58.

[41] Simon Dixon. "An On-Line Time Warping Algorithm for Tracking Musical Performances". In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. Edinburgh, Scotland, 2005, pp. 1727–1728.

[42] Simon Dixon and Gerhard Widmer. "MATCH: A Music Alignment Tool Chest". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. London, UK, 2005, pp. 492–497.

[43] Pierre Donat-Bouillud, Jean-Louis Giavitto, Arshia Cont, Nicolas Schmidt, and Yann Orlarey. "Embedding native audio-processing in a score following system with quasi sample accuracy". In: *Proceedings of the International Computer Music Conference (ICMC)*. Utrecht, Netherlands, 2016, pp. 478–484.

[44] Matthias Dorfer, Andreas Arzt, Sebastian Böck, Amaury Durand, and Gerhard Widmer. "Live Score Following on Sheet Music Images". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Late Breaking / Demo*. New York, USA, 2016.

[45] Matthias Dorfer, Andreas Arzt, and Gerhard Widmer. "Towards Score Following in Sheet Music Images". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. New York, USA, 2016, pp. 789–795.

[46] Zhiyao Duan and Brian Pardo. "A state space model for on-line polyphonic audio-score alignment". In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Prague, Czech Republic, 2011, pp. 197–200.

[47] Daniel P.W. Ellis and Graham E. Poliner. "Identifying 'Cover Songs' with Chroma Features and Dynamic Programming Beat Tracking". In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Vol. 4. Honolulu, Hawaii, USA, 2007.

[48] Sebastian Ewert and Meinard Müller. "Refinement Strategies for Music Synchronization". In: Lecture Notes in Computer Science 5493 (2008), pp. 147–165.

[49] Sebastian Ewert, Meinard Müller, and Peter Grosche. "High Resolution Audio Synchronization Using Chroma Onset Features". In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Taipei, Taiwan, 2009, pp. 1869–1872.

[50] Sebastian Flossmann. "Expressive Performance Rendering with Probabilistic Models - Creating, Analyzing, and Using the Magaloff Corpus". PhD thesis. Linz, Austria: Johannes Kepler University, 2010.

[51] Sebastian Flossmann, Werner Goebl, Maarten Grachten, Bernhard Niedermayer, and Gerhard Widmer. "The Magaloff project: An interim report". In: *Journal of New Music Research* 39.4 (2010), pp. 363–377.

[52] Christian Fremerey, Frank Kurth, Meinard Müller, and Michael Clausen. "A Demonstration of the SyncPlayer System". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Vienna, Austria, 2007, pp. 131–132.

[53] Christian Fremerey, Meinard Müller, and Michael Clausen. "Handling Repeats and Jumps in Score-Performance Synchronization". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Utrecht, The Netherlands, 2010, pp. 243–248.

[54] Christian Fremerey, Meinard Müller, Frank Kurth, and Michael Clausen. "Automatic Mapping of Scanned Sheet Music to Audio Recordings". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Philadelphia, USA, 2008, pp. 413–418.

[55] Harald Frostel, Andreas Arzt, and Gerhard Widmer. "The Vowel Worm: Real-Time Mapping and Visualisation of Sung Vowels in Music". In: *Proceedings of the Sound and Music Computing Conference (SMC)*. Padova, Italy, 2011, pp. 214–219.

[56] Martin Gasser, Andreas Arzt, Thassilo Gadermaier, Maarten Grachten, and Gerhard Widmer. "Classical Music on the Web – User Interfaces and Data Representations". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Málaga, Spain, 2015, pp. 571–577.

[57] Martin Gasser, Arthur Flexer, and Gerhard Widmer. "Streamcatcher: Integrated Visualization of Music Clips and Online Audio Streams". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Philadelphia, USA, 2008, pp. 205–210.

[58] Dariu M. Gavrila and Larry S. Davis. "Towards 3-D Model-based Tracking and Recognition of Human Movement". In: *Proceedings of the IEEE International Workshop on Face and Gesture Recognition*. Zurich, Switzerland, 1995, pp. 272–277.

[59] Werner Goebl. "Numerisch-klassifikatorische Interpretationsanalyse mit dem Boesendorfer Computerfluegel". MA thesis. Vienna, Austria: University of Vienna, 1999.

[60] Emilia Gómez, Maarten Grachten, Alan Hanjalic, Jordi Janer, Sergi Jorda, Carles F. Julia, Cynthia Liem, Agustin Martorell, Markus Schedl, and Gerhard Widmer. "PHENICX: Performances as Highly Enriched aNd Interactive Concert Expe-

riences". In: *Proceedings of the Sound and Music Computing Conference (SMC)*. Stockholm, Sweden, 2013, pp. 681–688.

[61] Maarten Grachten, Martin Gasser, Andreas Arzt, and Gerhard Widmer. "Automatic Alignment of Music Performances with Structural Differences". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Curitiba, Brazil, 2013, pp. 607–612.

[62] 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)*. Kyoto, Japan, 2012.

[63] Peter Grosche and Meinard Müller. "Toward Musically-Motivated Audio Fingerprints". In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Kyoto, Japan, 2012, pp. 93–96.

[64] Peter Grosche, Meinard Müller, and Joan Serrà. "Audio Content-Based Music Retrieval". In: *Multimodal Music Processing*. Ed. by Meinard Müller, Masataka Goto, and Markus Schedl. Vol. 3. Dagstuhl Follow-Ups. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2012, pp. 157–174.

[65] Lorin Grubb and Roger B. Dannenberg. "A Stochastic Method of Tracking a Vocal Performer". In: *Proceedings of the International Computer Music Conference (ICMC)*. Thessaloniki, Greece, 1997.

[68] 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, USA, 2003.

[69] F. I. Itakura. "Minimum prediction residual principle applied to speech recognition". In: *IEEE Transactions on Acoustics Speech and Signal Processing* 23.1 (1975), pp. 52–72.

[70] Cyril Joder, Slim Essid, and Gaël Richard. "A comparative study of tonal acoustic features for a symbolic level music-to-score alignment". In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Dallas, USA, 2010.

[72] Hagen Kaprykowsky and Xavier Rodet. "Globally Optimal Short-Time Dynamic Time Warping, Application to Score to Audio Alignment". In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Toulouse, France, 2006, pp. 249–252.

[73] Rainer Kelz, Matthias Dorfer, Filip Korzeniowski, Sebastian Böck, Andreas Arzt, and Gerhard Widmer. "On the Potential of Simple Framewise Approaches to Piano Transcription". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. New York, USA, 2016, pp. 475–481.

[74] Filip Korzeniowski, Florian Krebs, Andreas Arzt, and Gerhard Widmer. "Tracking Rests and Tempo Changes: Improved Score Following with Particle Filters". In: *Proceedings of the International Computer Music Conference (ICMC)*. Perth, Australia, 2013.

[75] Frank Kurth and Meinard Müller. "Efficient Index-Based Audio Matching". In: *IEEE Transactions on Audio, Speech, and Language Processing* 16.2 (2008), pp. 382–395.

[76] Frank Kurth, Meinard Müller, David Damm, Christian Fremerey, Andreas Ribbrock, and Michael Clausen. "SyncPlayer - An Advanced System for Multimodal Music Access". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. London, UK, 2005, pp. 381–388.

[77] 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 Society for Music Information Retrieval Conference (ISMIR)*. Vienna, Austria, 2007, pp. 261–266.

[78] Dustin Lang, David W. Hogg, Keir Mierle, Michael Blanton, and Sam Roweis. "Astrometry.net: Blind astrometric calibration of arbitrary astronomical images". In: *The Astronomical Journal* 139.5 (2010), p. 1782.

[79] Bochen Li and Zhiyao Duan. "Score Following for Piano Performances with Sustain-Pedal Effects". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Málaga, Spain, 2015, pp. 469–475.

[80] Robert Macrae and Simon Dixon. "Accurate Real-time Windowed Time Warping". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Utrecht, The Netherlands, 2010, pp. 423–428.

[82] Matthias Mauch and Simon Dixon. "Approximate note transcription for the improved identification of difficult chords". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Utrecht, The Netherlands, 2010, pp. 135–140.

[83] Mark Melenhorst, Ron van der Sterren, Andreas Arzt, Agustin Martorell, and Cynthia C.S. Liem. "A Tablet App to Enrich the Live and Post-Live Experience of Classical Concerts". In: *Proceedings of the 3rd ACM International Workshop on Interactive Content Consumption*. Brussels, Belgium, 2015.

[84] Yoram Meron and Keikichi Hirose. "Automatic alignment of a musical score to performed music". In: *Acoustical Science & Technology* 22.3 (2001), pp. 189–198.

[85] Marius Miron, Julio José Carabias-Orti, and Jordi Janer. "Audio-to-Score Alignment at Note Level for Orchestral Recordings". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Taipei, Taiwan, 2014, pp. 125–130.

[86] 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)*. Prague, Czech Republic, 2011, pp. 193–196.

[88] Meinard Müller. *Fundamentals of Music Processing*. Springer Verlag, 2015.

[89] Meinard Müller and Daniel Appelt. "Path-Constrained Partial Music Synchronization". In: *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Las Vegas, USA, 2008, pp. 65–68.

[90] Meinard Müller and Sebastian Ewert. "Chroma Toolbox: MATLAB Implementations For Extracting Variants of Chroma-Based Audio Features". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Miami, USA, 2011, pp. 215–220.

[91] Meinard Müller, Verena Konz, Andi Scharfstein, Sebastian Ewert, and Michael Clausen. "Towards Automated Extraction of Tempo Parameters from Expressive Music Recordings". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Kobe, Japan, 2009, pp. 69–74.

[92] Meinard Müller, Frank Kurth, and Michael Clausen. "Audio Matching via Chroma-Based Statistical Features". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. London, UK, 2005, pp. 288–295.

[93] Meinard Müller, Frank Kurth, and Tido Röder. "Towards an Efficient Algorithm for Automatic Score-to-Audio Synchronization". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Barcelona, Spain, 2004, pp. 365–372.

[94] Meinard Müller, Henning Mattes, and Frank Kurth. "An Efficient Multiscale Approach to Audio Synchronization". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Victoria, Canada, 2006, pp. 192–197.

[95] Eita Nakamura, Philippe Cuvillier, Arshia Cont, Nobutaka Ono, and Shigeki Sagayama. "Autoregressive Hidden Semi-Markov Model of Symbolic Music Performance for Score Following". In: *Proceedings of the International Society for Music Information Retrieval Conference*. Málaga, Spain, 2015, pp. 392–398.

[96] Eita Nakamura, Nobutaka Ono, Yasuyuki Saito, and Shigeki Sagayama. "Merged-Output Hidden Markov Model for Score Following of MIDI Performance with Ornaments, Desynchronized Voices, Repeats and Skips". In: *Joint Proceedings of the International Computer Music Conference (ICMC) and the Sound and Music Computing Conference (SMC)*. Athens, Greece, 2014, pp. 1185–1192.

[97] Tomohiko Nakamura, Eita Nakamura, and Shigeki Sagayama. "Real-Time Audio-to-Score Alignment of Music Performances Containing Errors and Arbitrary Repeats and Skips". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 24.2 (2016), pp. 329–339.

[98] Bernhard Niedermayer and Gerhard Widmer. "A Multi-pass Algorithm for Accurate Audio-to-Score Alignment". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Utrecht, The Netherlands, 2010, pp. 417–422.

[99] Nicola Orio and François Déchelle. "Score Following Using Spectral Analysis and Hidden Markov Models". In: *Proceedings of the International Computer Music Conference (ICMC)*. Havana, Cuba, 2001.

[100] Nicola Orio, Serge Lemouton, Diemo Schwarz, and Norbert Schnell. "Score Following: State of the Art and New Developments". In: *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*. Montréal, Canada, 2003, pp. 36–41.

[101] Nicola Orio and Diemo Schwarz. "Alignment of Monophonic and Polyphonic Music to a Score". In: *Proceedings of the International Computer Music Conference ICMC*. Havana, Cuba, 2001.

[103] Takuma Otsuka, Kazuhiro Nakadai, Toru Takahashi, Tetsuya Ogata, and Hiroshi G. Okuno. "Real-Time Audio-to-Score Alignment using Particle Filter for Co-player Music Robots". In: *EURASIP Journal on Advances in Signal Processing* 2011.2011:384651 (2011).

[104] 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)*. Pittsburgh, USA, 2005, pp. 1018–1023.

[105] Bryan Pardo, Jonah Shifrin, and William Birmingham. "Name that Tune: A Pilot Study in Finding a Melody from a Sung Query". In: *Journal of the American Society for Information Science and Technology* 55.4 (2004), pp. 283–300.

[106] Geoffroy Peeters. "Chroma-based estimation of musical key from audio-signal analysis". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Victoria, Canada, 2006, pp. 115–120.

[107] Geoffrey Peters, Caroline Anthony, and Michael Schwartz. "Song Search and Retrieval by Tapping". In: *Proceedings of the National Conference on Artificial Intelligence (AAAI)*. Pittsburgh, USA, 2005, pp. 1696–1697.

[108] Henning Pohl and Aristotelis Hadjakos. "Dance Pattern Recognition using Dynamic Time Warping". In: *Proceedings of the Sound and Music Computing Conference (SMC)*. Barcelona, Spain, 2010.

[109] Thomas Prätzlich, Jonathan Driedger, and Meinard Müller. "Memory-Restricted Multiscale Dynamic Time Warping". In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Shanghai, China, 2016, pp. 569–573.

[110] Matthew Prockup, David Grunberg, Alex Hrybyk, and Youngmoo E. Kim. "Orchestral Performance Companion: Using Real-Time Audio to Score Alignment". In: *IEEE Multimedia* 20.2 (2013), pp. 52–60.

[111] Miller S. Puckette and Cort Lippe. "Score Following in Practice". In: *Proceedings of the International Computer Music Conference (ICMC)*. San Jose, USA, 1992.

[112] Lawrence Rabiner and Bing-Hwang Juang. *Fundamentals of Speech Recognition*. Prentice Hall Signal Processing Series, 1993.

[113] Mathieu Ramona and Geoffroy Peeters. "AudioPrint: an efficient audio fingerprint system based on a novel cost-less synchronization scheme". In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Vancouver, Canada, 2013, pp. 818–822.

[114] Christopher Raphael. "Automatic Segmentation of Acoustic Musical Signals Using Hidden Markov Models". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21.4 (1999), pp. 360–370.

[115] Christopher Raphael. "A Bayesian Network for Real-Time Musical Accompaniment". In: *Advances in Neural Information Processing Systems (NIPS)*. Vancouver, Canada, 2001, pp. 1433–1439.

[116] Christopher Raphael. "Aligning music audio with symbolic scores using a hybrid graphical model". In: *Machine Learning* 65.2-3 (2006), pp. 389–409.

[117] Christopher Raphael. "Current Directions with Music Plus One". In: *Proceedings of the Sound and Music Computing Conference (SMC)*. Porto, Portugal, 2009, pp. 71–76.

[118] Christopher Raphael. "Music Plus One and Machine Learning". In: *Proceedings of the International Conference on Machine Learning (ICML)*. Haifa, Israel, 2010, pp. 21–28.

[119] Toni M. Rath and R. Manmatha. "Word Image Matching Using Dynamic Time Warping". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Madison, USA, 2003, pp. 521–527.

[120] Ana Rebelo, Ichiro Fujinaga, Filipe Paszkiewicz, Andre R. S. Marcal, Carlos Guedes, and Jaime S. Cardoso. "Optical music recognition: state-of-the-art and open issues". In: *International Journal of Multimedia Information Retrieval* 1.3 (2012), pp. 173–190.

[122] Daniel Röwenstrunk, Thomas Prätzlich, Thomas Betzwieser, Meinard Müller, Gerd Szwillus, and Joachim Veit. "Das Gesamtkunstwerk Oper aus Datensicht – Aspekte des Umgangs mit einer heterogenen Datenlage im BMBF-Projekt "Freischütz Digital"". In: *Datenbank-Spektrum* 15.1 (2015), pp. 65–72.

[123] Shinji Sako, Ryuichi Yamamoto, and Tadashi Kitamura. "Ryry: A Real-Time Score-Following Automatic Accompaniment Playback System Capable of Real Performances with Errors, Repeats and Jumps". In: (2014), pp. 134–145.

[124] Hiroaki Sakoe and Seibi Chiba. "Dynamic programming algorithm optimization for spoken word recognition". In: *IEEE Transactions on Acoustics Speech and Signal Processing* 26.1 (1978), pp. 43–49.

[125] Stan Salvador and Philip Chan. "Toward accurate dynamic time warping in linear time and space". In: *Intelligent Data Analysis* 11.5 (2007), pp. 561–580.

[126] Diemo Schwarz, Nicola Orio, and Norbert Schnell. "Robust Polyphonic Midi Score Following with Hidden Markov Models". In: *Proceedings of the International Computer Music Conference (ICMC)*. Miami, USA, 2004.

[127] Joan Serrà, Emilia Gómez, and Perfecto Herrera. "Audio cover song identification and similarity: background, approaches, evaluation and beyond". In: *Advances in Music Information Retrieval*. Ed. by Z. W. Ras and A. A. Wieczorkowska. Vol. 274. Studies in Computational Intelligence. Berlin, Germany: Springer, 2010. Chap. 14, pp. 307–332.

[128] Joan Serrà, Emilia Gómez, Perfecto Herrera, and Xavier Serra. "Chroma Binary Similarity and Local Alignment Applied to Cover Song Identification". In: *IEEE Transactions on Audio, Speech, and Language Processing* 16 (2008), pp. 1138–1151.

[129] Joren Six and Marc Leman. "Panako - A Scalable Acoustic Fingerprinting System Handling Time-Scale and Pitch Modification". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Taipei, Taiwan, 2014, pp. 259–264.

[130] Reinhard Sonnleitner, Andreas Arzt, and Gerhard Widmer. "Landmark-Based Audio Fingerprinting for DJ Mix Monitoring". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. New York, USA, 2016, pp. 185–191.

[131] Reinhard Sonnleitner and Gerhard Widmer. "Quad-Based Audio Fingerprinting Robust to Time and Frequency Scaling". In: *Proceedings of the International Conference on Digital Audio Effects (DAFx)*. Erlangen, Germany, 2014, pp. 173–180.

[132] Reinhard Sonnleitner and Gerhard Widmer. "Robust Quad-based Audio Fingerprinting". In: *IEEE/ACM Transactions on Audio, Speech and Language Processing* 24.3 (2016), pp. 409–421.

[134] 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)*. Esbjerg, Denmark, 2004, pp. 211–219.

[135] Verena Thomas, Christian Fremerey, Meinard Müller, and Michael Clausen. "Linking Sheet Music and Audio - Challenges and New Approaches". In: *Multimodal Music Processing*. Ed. by Meinard Müller, Masataka Goto, and Markus Schedl. Vol. 3. Dagstuhl Follow-Ups. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2012, pp. 1–22.

[136] Barry Vercoe. "The Synthetic Performer in the Context of Live Performance". In: *Proceedings of the International Computer Music Conference (ICMC)*. Paris, France, 1984, pp. 199–200.

[137] Alessandro Vinciarelli. "A survey on off-line Cursive Word Recognition". In: *Pattern Recognition* 35.7 (2002), pp. 1433–1446.

[138] Avery Wang. "An Industrial Strength Audio Search Algorithm". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Baltimore, Maryland, USA, 2003, pp. 7–13.

[139] Siying Wang, Sebastian Ewert, and Simon Dixon. "Robust Joint Alignment of Multiple Versions of a Piece of Music". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Taipei, Taiwan, 2014, pp. 83–88.

[140] Gerhard Widmer. "Discovering simple rules in complex data: A meta-learning algorithm and some surprising musical discoveries". In: *Artificial Intelligence* 146.2 (2003), pp. 129–148.

[141] Gerhard Widmer. "Getting Closer to the Essence of Music: The Con Espressione Manifesto". In: *ACM Trans. Intell. Syst. Technol.* 8.2 (2016), 19:1–19:13.

[142] Gerhard Widmer, Simon Dixon, Werner Goebl, Elias Pampalk, and Asmir Tobudic. "In search of the Horowitz factor". In: *AI Magazine* 24.3 (2003), pp. 111–130.

[143] Guangyu Xia, Yun Wang, Roger B. Dannenberg, and Geoffrey Gordon. "Spectral Learning for Expressive Interactive Ensemble Music Performance". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Málaga, Spain, 2015, pp. 816–822.

[144] Ryuichi Yamamoto, Shinji Sako, and Tadashi Kitamura. "Robust on-line algorithm for real-time audio-to-score alignment based on a delayed decision and anticipation framework". In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Vancouver, Canada, 2013, pp. 191–195.

CURRICULUM VITAE

ANDREAS ARZT

PARTICULARS

EDUCATION

Technical University of Vienna — Vienna, Austria
M. S. in Computer Science — *February 2006 - February 2008*
With Distinction

University of Vienna — Vienna, Austria
BSc. in Computer Science — *October 2002 - February 2006*

CURRENT STATUS

Austrian Resident, Citizen of Austria.

RESEARCH INTERESTS

My main interests in research are in the area of (real-time) music understanding, which can be seen as a multi-faceted combination of digital signal processing, artificial intelligence (mainly machine learning), and musicology.

ACADEMIC HONORS

- Best Paper Award for "Real-time Music Tracking using Multiple Performances as a Reference" by A. Arzt and G. Widmer at International Society for Music Information Retrieval Conference (ISMIR), Málaga, Spain, 2015.
- Best Demo Award for the "The Piano Music Companion" by A. Arzt, S. Böck, S. Flossmann, H. Frostel, M. Gasser, C.C.S. Liem, and G. Widmer at the 8th International Conference on the Prestigious Applications of Intelligent Systems (PAIS) in conjunction with the 21st European Conference on Artificial Intelligence (ECAI), Prague, Czech Republic, 2014.
- Best Demo Award for "The Complete Classical Music Companion V0.9" by A, Arzt, S. Böck, S. Flossmann, H. Frostel, M. Gasser, and G. Widmer at the AES 53rd International Conference on Semantic Audio, London, UK, 2014.

RESEARCH EXPERIENCE

- **Austrian Research Institute for Artificial Intelligence (OFAI), Vienna, Austria**, July 2016 - Present.
  Research Topics: Methods to collect, annotate and analyse large corpora of classical music.
- **Austrian Research Institute for Artificial Intelligence (OFAI), Vienna, Austria**, September 2014 - December 2015.
  Research Topics: Live Tracking of Orchestral Music (including a live event at the Concertgebouw in Amsterdam).
- **Research Assistant and University Assistant, Department of Computational Perception, Johannes Kepler University, Linz, Austria**, July 2009 - June 2016.
  Research Topics: Music Synchronisation, Real-time Music Tracking, Music Identification, Pattern Discovery in Music

TEACHING EXPERIENCE

- **KV Special Topics Audio and Music Processing.** Summer 2011, Summer 2012, Summer 2013, Summer 2014 and Summer 2015 at Johannes Kepler University. This course is an introduction to the automatic analysis of music via computer algorithms.
- **KV Special Topics Artificial Intelligence in Media, Art and Society.** Summer 2014 and Summer 2015 at Johannes Kepler University. This class consists of lectures, guest lectures, and screenings that focus on current trends in media, art, and society that are related to Artificial Intelligence.
- **UE Algorithmen und Datenstrukturen 1.** Summer 2014 at Johannes Kepler University. This is the accompanying lab course to the lecture VO Algorithmen und Datenstrukturen 1, which teaches basic techniques in algorithms and data structures.
- **UE Artificial Intelligence.** Winter 2010, Winter 2011, Winter 2012 and Winter 2013 at Johannes Kepler University. This is the accompanying lab course to the lecture VO Artificial Intelligence. The goal is to improve the students' understanding of the material and prepare them for the exam.
- **SE Seminar in Computational Engineering.** Winter 2013 at Johannes Kepler University. Research seminar.
- **PR Project in Computational Engineering.** Winter 2013 at Johannes Kepler University.

PAPERS

1. 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)*. Patras, Greece, 2008, pp. 241–245
2. Andreas Arzt and Gerhard Widmer. "Simple Tempo Models for Real-time Music Tracking". In: *Proceedings of the Sound and Music Computing Conference (SMC)*. Barcelona, Spain, 2010
3. Andreas Arzt and Gerhard Widmer. "Towards Effective 'Any-Time' Music Tracking". In: *Proceedings of the Starting AI Researchers' Symposium (STAIRS)*. Lisbon, Portugal, 2010, pp. 24–36
4. Andreas Arzt and Gerhard Widmer. "Robust Real-time Music Tracking". In: *Proceedings of the Vienna Talk on Musical Acoustics (VITA)*. Vienna, Austria, 2010, pp. 5–8
5. Harald Frostel, Andreas Arzt, and Gerhard Widmer. "The Vowel Worm: Real-Time Mapping and Visualisation of Sung Vowels in Music". In: *Proceedings of the Sound and Music Computing Conference (SMC)*. Padova, Italy, 2011, pp. 214–219
6. Andreas Arzt, Gerhard Widmer, Sebastian Böck, Reinhard Sonnleitner, and Harald Frostel. "Towards a Complete Classical Music Companion". In: *Proceedings of the European Conference on Artificial Intelligence (ECAI)*. Montpellier, France, 2012, pp. 67–72
7. Andreas Arzt, Gerhard Widmer, and Simon Dixon. "Adaptive Distance Normalization for Real-time Music Tracking". In: *Proceedings of the European Signal Processing Conference (EUSIPCO)*. Bucharest, Romania, 2012, pp. 2689–2693
8. Andreas Arzt, Sebastian Böck, and Gerhard Widmer. "Fast Identification of Piece and Score Position via Symbolic Fingerprinting". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Porto, Portugal, 2012, pp. 433–438
9. Sebastian Böck, Andreas Arzt, Florian Krebs, and Markus Schedl. "Online Real-Time Onset Detection with Recurrent Neural Networks". In: *Proceedings of the International Conference on Digital Audio Effects (DAFx)*. York, United Kingdom, 2012
10. Filip Korzeniowski, Florian Krebs, Andreas Arzt, and Gerhard Widmer. "Tracking Rests and Tempo Changes: Improved Score Following with Particle Filters". In: *Proceedings of the International Computer Music Conference (ICMC)*. Perth, Australia, 2013
11. Maarten Grachten, Martin Gasser, Andreas Arzt, and Gerhard Widmer. "Automatic Alignment of Music Performances with Structural Differences". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Curitiba, Brazil, 2013, pp. 607–612
12. Tom Collins, Andreas Arzt, Sebastian Flossmann, and Gerhard Widmer. "SIARCT-CFP: Improving Precision and the Discovery of Inexact

Musical Patterns in Point-set Representation". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Curitiba, Brazil, 2013, pp. 549–554

13. Andreas Arzt, Gerhard Widmer, and Reinhard Sonnleitner. "Tempo- and Transposition-invariant Identification of Piece and Score Position". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Taipeh, Taiwan, 2014, pp. 549–554

14. Andreas Arzt, Cynthia C.S. Liem, and Gerhard Widmer. "A Tempo- and Transposition-invariant Piano Music Companion". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Late Breaking / Demo*. Taipeh, Taiwan, 2014

15. Andreas Arzt, Sebastian Böck, Sebastian Flossmann, Harald Frostel, Martin Gasser, and Gerhard Widmer. "The Complete Classical Music Companion V0.9". In: *Proceedings of the 53rd AES Conference on Semantic Audio*. London, England, 2014

16. Andreas Arzt, Sebastian Böck, Sebastian Flossmann, Harald Frostel, Martin Gasser, Cynthia C.S. Liem, and Gerhard Widmer. "The Piano Music Companion". In: *Proceedings of the Conference on Prestigious Applications of Intelligent Systems (PAIS)*. Prague, Czech Republic, 2014, pp. 1221–1222

17. Tom Collins, Daniel A. Abrams, Rohan Chandra, Christina Young, Andreas Arzt, and Vinod Menon. "Neural tracking of musical motives revealed by a combination of fMRI and music information retrieval techniques". In: *Proceedings of the International Conference on Music Perception and Cognition (ICMPC)*. Seoul, South Korea, 2014, p. 55

18. Andreas Arzt, Werner Goebl, and Gerhard Widmer. "Flexible Score Following: The Piano Music Companion and Beyond". In: *Proceedings of the Vienna Talk on Musical Acoustics (VITA)*. Vienna, Austria, 2015, pp. 220–223

19. Andreas Arzt, Harald Frostel, Thassilo Gadermaier, Martin Gasser, Maarten Grachten, and Gerhard Widmer. "Artificial Intelligence in the Concertgebouw". In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. Buenos Aires, Argentina, 2015, pp. 2424–2430

20. Andreas Arzt and Gerhard Widmer. "Real-time Music Tracking using Multiple Performances as a Reference". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Málaga, Spain, 2015, pp. 357–363

21. Martin Gasser, Andreas Arzt, Thassilo Gadermaier, Maarten Grachten, and Gerhard Widmer. "Classical Music on the Web – User Interfaces and Data Representations". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. Málaga, Spain, 2015, pp. 571–577

22. Mark Melenhorst, Ron van der Sterren, Andreas Arzt, Agustin Martorell, and Cynthia C.S. Liem. "A Tablet App to Enrich the Live and Post-Live Experience of Classical Concerts". In: *Proceedings of the 3rd ACM International Workshop on Interactive Content Consumption*. Brussels, Belgium, 2015

23. Rainer Kelz, Matthias Dorfer, Filip Korzeniowski, Sebastian Böck, Andreas Arzt, and Gerhard Widmer. "On the Potential of Simple Framewise Approaches to Piano Transcription". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. New York, USA, 2016, pp. 475–481
24. Reinhard Sonnleitner, Andreas Arzt, and Gerhard Widmer. "Landmark-Based Audio Fingerprinting for DJ Mix Monitoring". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. New York, USA, 2016, pp. 185–191
25. Tom Collins, Andreas Arzt, Harald Frostel, and Gerhard Widmer. "Using Geometric Symbolic Fingerprinting to Discover Distinctive Patterns in Polyphonic Music Corpora". In: *Computational Music Analysis*. Ed. by David Meredith. Springer International Publishing, 2016, pp. 445–474
26. Matthias Dorfer, Andreas Arzt, Sebastian Böck, Amaury Durand, and Gerhard Widmer. "Live Score Following on Sheet Music Images". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Late Breaking / Demo*. New York, USA, 2016
27. Matthias Dorfer, Andreas Arzt, and Gerhard Widmer. "Towards Score Following in Sheet Music Images". In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. New York, USA, 2016, pp. 789–795

THESIS

1. Andreas Arzt. "Score Following with Dynamic Time Warping: An Automatic Page Turner". MA thesis. Vienna, Austria: Vienna University of Technology, 2008

OTHER REPORTS

1. Andreas Arzt, Werner Goebl, Dominik Schnitzer and Gerhard Widmer. "Assessment nationaler Nischen in Bezug auf EU-Aktivitäten am Beispiel Computational Perception", *Studie im Auftrag des österreichischen Bundesministeriums für Verkehr, Innovation und Technologie*, 2009.

TALKS

INVITED TALKS

- "Demonstration of the Piano Music Companion at the Opening Ceremony", *International Joint Conference on Artificial Intelligence (IJCAI)*, Buenos Aires, Argentina, July 2015.
- "Echtzeit-Musik-Erkennng Live-Demo (Real-time Music Recognition Live Demo)", *IMAGINE*, Vienna, Austria, June 2015.