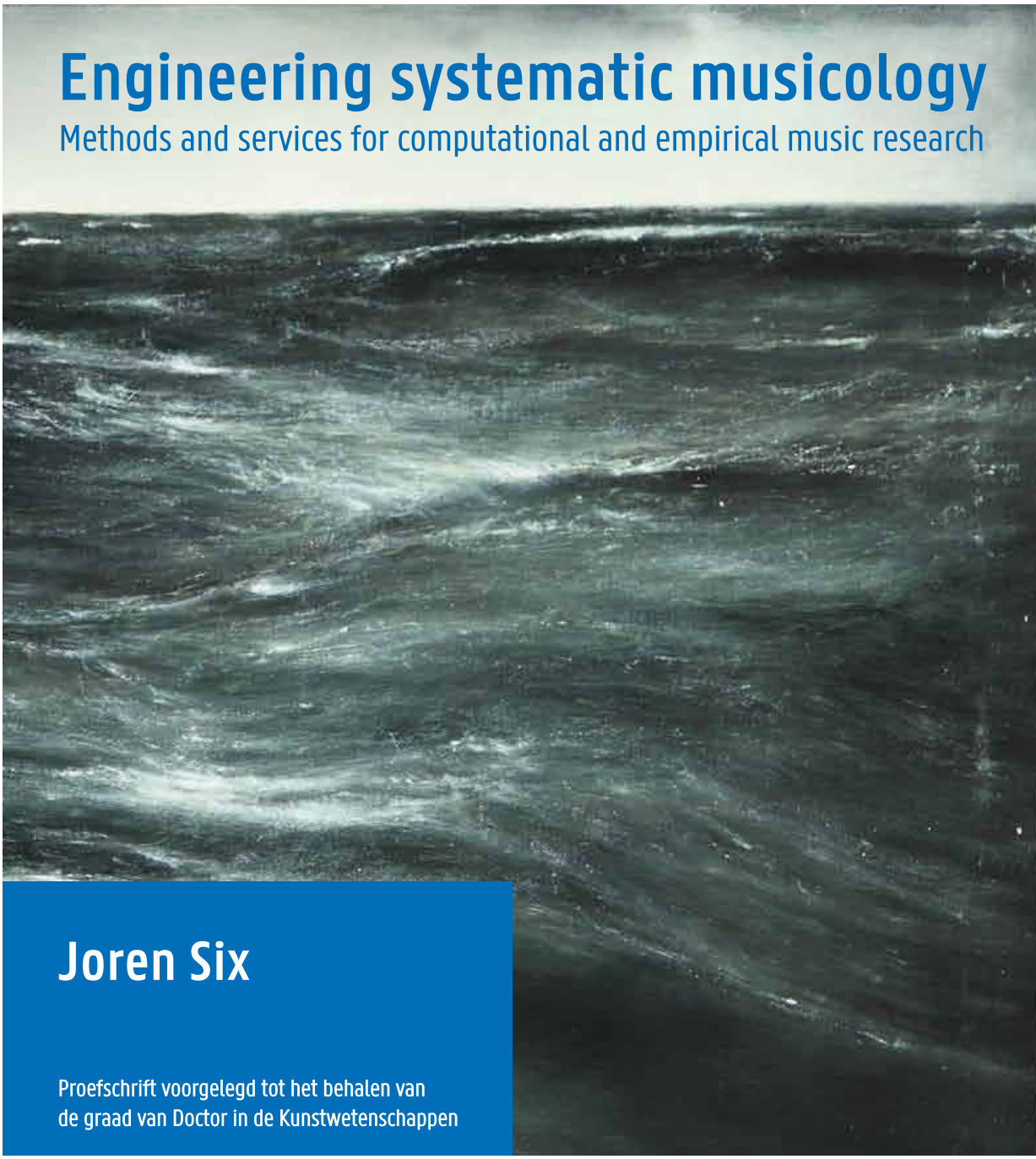
FACULTEIT LETTEREN
EN WIJSBEGEERTE

# Engineering systematic musicology

## Methods and services for computational and empirical music research



Joren Six

Proefschrift voorgelegd tot het behalen van
de graad van Doctor in de Kunstwetenschappen

UNIVERSITEIT
GENT

UNIVERSITEIT GENT

Universiteit Gent
Faculteit Letteren en Wijsbegeerte
Vakgroep Kunst-, Muziek- en Theaterwetenschappen

# Engineering systematic musicology

## Methods and services for computational and empirical music research

---

Joren Six

Proefschrift voorgelegd tot het behalen van
de graad van Doctor in de Kunstwetenschappen
Academisch jaar 2017-2018

UNIVERSITEIT GENT

Universiteit Gent
Faculteit Letteren en Wijsbegeerte
Vakgroep Kunst-, Muziek- en Theaterwetenschappen

| | |
|---|---|
| Promotor: | Prof. dr. Marc Leman |
| Doctoraatsbegeleidingscommissie: | Dr. Olmo Cornelis<br>Dr. Frans Wiering |
| Examencommissie: | Dr. Federica Bressan<br>Dr. Olmo Cornelis<br>Prof. dr. ir. Tijl De Bie<br>Prof. dr. Pieter-Jan Maes<br>Dr. Frans Wiering<br>Dr. Micheline Lesaffre<br>Dr. Luc Nijs |

Proefschrift voorgelegd tot het behalen van
de graad van Doctor in de Kunstwetenschappen
Academisch jaar 2017-2018

Universiteit Gent
Faculteit Letteren en Wijsbegeerte
Vakgroep Kunst-, Muziek- en Theaterwetenschappen, IPEM
De Krook, Miriam Makebaplein 1, B-9000 Gent, België

Kaft: Mer Du Nord, Étude n°1, Thierry De Cordier, 2011

# Acknowledgement

This doctoral dissertation is the culmination of my research carried out at both *IPEM, Ghent University* and the *School of Arts, Ghent*. I have been lucky enough to pursue and combine my interests for both music and computer science in my research. As a trained computer scientist I have been applying my engineering background to problems in systematic musicology. The output of this work has been described in various articles some of which are bundled in this dissertation.

Admittedly, my research trajectory does not follow the straightest path but meanders around several fields. This meandering has enabled me to enjoy various vistas and led me to a plethora of places - physically and intellectually - not easily reached without taking a turn now and again. I think this multi-disciplinary approach prepared me better for a hopefully somewhat stable career in research. I also had the time required to cover a lot of ground. At the *School of Arts, Ghent* I was employed for four years as a scientific employee. At *IPEM, Ghent University* I was given the opportunity to continue my work as a doctoral student again for four years. This allowed me to not only have a broad perspective but also reach the depth required to contribute new knowledge and propose innovative methods.

It is safe to say that without Olmo Cornelis I would not have started this PhD project. Olmo wrote the project proposal which eventually allowed me to start my research career at the School of Arts. The concept of having an engineer next to a humanities scholar was definitely enriching to me and I do hope that the opposite is also somewhat true. His guidance during those first four (and following) years was indispensable. His pointers on music theory, his support with academic writing and ideas on computer aided (ethno)musicology are just a few examples.

Actually, I want to profoundly thank the whole group of colleague researchers at the School of Arts, or what was then known as the Royal Conservatory of Ghent. Undeniably, they had a defining influence on my research and personal life. I fondly remember many discussions in the cellar and at my desk at the Wijnaert. I struggle to think of examples where gossiping, scheming and collusions between a colleague and partner of a colleague could have had a more positive outcome. Indeed, I mean you: Ruth and Clara.

Later on, Marc Leman gave me the opportunity to continue my research at IPEM. I am very grateful to have been offered this privilege. I am quite aware that being able to pursue one's interests by doing research is exactly that: a privilege. IPEM provided fertile ground to further my research. Marc's hands-off approach displays a great

amount of trust in his research staff. This freedom worked especially well for me since it made me self-critical on priorities and planning.

I would also like to acknowledge the IPEM bunch: a diverse collective of great individuals each in their own way. I especially would like to thank Ivan for the many hardware builds and creative ideas for practical solutions. Katrien for taking care of the administrative side of things. Esther for having the patience to listen to me whining about my kids. Jeska and Guy for proofreading this work, all remaining spelling errors are surely introduced while reworking the text afterwards. I would also like to thank all the others for the many discussions at the kitchen table during lunch and generally for being great colleagues.

Furthermore, I am very grateful to the RMCA (Royal Museum for Central Africa), Tervuren, Belgium for providing access to its unique archive of Central African music.

Thanks to my friends and family for the support over the years. I would especially want to thank Will for proofraeding parts of this work and Emilie for pointing me to VEWA: next year will be good. Of course, it is hardly an understatement to claim that this work would not be here without my parents. Thanks for kindling an interest in music, letting me attend piano lessons and for keeping me properly fed and relatively clean, especially in my first years. Thanks also to Bobon for caring for Oscar and Marit, often on short notice in those many small emergency situations. On the topic: I would like to leave Oscar and Marit out of this acknowledgment since they only sabotaged this work, often quite successfully. But they are a constant reminder on the relativity of things and I love them quite unconditionally. Finally, I would like to thank the light of my life, the daydream that accompanies me at night, the mother of my children: Barbara.

# Contents

# Nederlandstalige samenvatting

Een van de grote onderzoeksvragen in systematische muziekwetenschappen is hoe mensen met muziek omgaan. Deze wetenschap onderzoekt bijvoorbeeld hoe effecten van muziek in relatie staan tot specifieke muzikale structuren. Deze fundamentele relatie kan op verschillende manieren benaderd worden. Bijvoorbeeld een perspectief vertrekkende vanuit traditie waarbij muziek aanzien wordt als een fenomeen van menselijke expressie. Een cognitief perspectief is een andere benadering, daarbij wordt muziek gezien als een akoestische informatiestroom gemoduleerd door perceptie, categorisatie, blootstelling en allerhande leereffecten. Een even geldig perspectief is dat van de uitvoerder waarbij muziek voortkomt uit gecoördineerde menselijke interactie. Om een muzikaal fenomeen te begrijpen is een combinatie van (deel)perspectieven vaak een meerwaarde.

Elk perspectief brengt methodes met zich mee die onderzoeksvragen naar concrete musicologische onderzoeksprojecten kan omvormen. Digitale data en software vormen tegenwoordig bijna altijd de kern van deze methodes. Enkele van die algemene methodes zijn: extractie van akoestische kenmerken, classificatie, statistiek, machine learning. Een probleem hierbij is dat het toepassen van deze *empirische en computationele methodes technische oplossingen vraagt.* Het ontwikkelen van deze technische oplossingen behoort vaak niet tot de competenties van onderzoekers die vaak een achtergrond in de zachte wetenschappen hebben. Toegang tot gespecialiseerde technische kennis kan op een bepaald punt noodzakelijk worden om hun onderzoek verder te zetten. Mijn doctoraatsonderzoek situeert zich in deze context.

Ik presenteer in dit werk *concrete technische oplossingen die bijdragen aan de systematische muziekwetenschappen.* Dit gebeurt door oplossingen te ontwikkelen voor meetproblemen in empirisch onderzoek en door implementatie van onderzoekssoftware die computationeel onderzoek faciliteert. Om over de verschillende aspecten van deze oplossingen een overzicht te krijgen worden ze in een vlak geplaatst.

De eerste as van dit vlak contrasteert dienstverlenende oplossingen met oplossingen die methodes *in* de systematische muziekwetenschappen aandragen of aangeven hoe onderzoek kan gebeuren met dienstverlenende oplossingen *voor* de systematische muziekwetenschappen. Die ondersteunen of automatiseren onderzoektaken. De dienstverlenende oplossingen kunnen de omvang van onderzoek vergroten door het eenvoudiger te maken om met grotere datasets aan de slag te gaan. De tweede as in het vlak geeft aan hoe sterk de oplossing leunt op Music Information Retrieval (MIR) technieken. MIR-technieken worden gecontrasteerd met verschillende technieken ter ondersteuning van em-

pirisch onderzoek.

Mijn onderzoek resulteerde in dertien oplossingen die in dit vlak geplaatst worden. De beschrijving van zeven van die oplossingen is opgenomen in dit werk. Drie ervan vallen onder methodes en de resterende vier zijn dienstverlendende (services). Het softwaresysteem Tarsos stelt bijvoorbeeld een methode voor om toohoogtegebruik in muzikale praktijk op grote schaal te vergelijken met theoretische modellen van toonladders. Het softwaresysteem SyncSink is een voorbeeld van een service. Het laat toe om onderzoeksdata te synchroniseren wat het eenvoudiger maakt om meerdere sensorstromen of participanten op te nemen. Andere services zijn TarsosDSP en Panako. TarsosDSP kan kenmerken uit audio halen en Panako is een *acoustic fingerprinting* systeem.

In het algemeen volgen de gepresenteerde oplossingen een reproduceerbare methodologie. Computationeel en MIR onderzoek is niet altijd even makkelijk te reproduceren. In de voorgestelde oplossingen werd aandacht gegeven aan dit aspect. De software werd via open source licenties online geplaatst en de systemen werden zo veel als mogelijk getest met publiek beschikbare data. Dit maakt de processen transparant en verifieerbaar. Het stelt ook anderen in staat om de software te gebruiken, te bekritiseren en te verbeteren.

De belangrijkste bijdragen van dit doctoraatsonderzoek zijn de individuele oplossingen. Met Panako (Six and Leman, 2014) werd een nieuw acoustic fingerprinting algoritme beschreven in de academische literatuur. Vervolgens werden applicaties van Panako toegepast voor het beheer van digitale muziekarchieven. Deze applicaties werden beschreven en getest (Six et al., 2018b). Tarsos (Six et al., 2013) laat toe om toonhoogtegebruik op grote schaal te onderzoeken. Ik heb bijdragen geleverd aan de discussie rond reproduceerbaarheid van MIR onderzoek (Six et al., 2018a). Ook werd een systeem voor verrijkte muziekervaring voorgesteld (Six and Leman, 2017). Naast deze specifieke bijdragen zijn er ook algemene zoals het conceptualiseren van technologische contributies aan de systematische muziekwetenschappen via het onderscheid tussen services en methodes. Als laatste werd het concept *augmented humanities* ook geïntroduceerd als een onderzoekslijn voor verder onderzoek.

# Summary

One of the main research questions of *systematic musicology* is concerned with how people make sense of their musical environment. It is concerned with signification and meaning-formation and relates musical structures to effects of music. These fundamental aspects can be approached from many different directions. One could take a cultural perspective where music is considered a phenomenon of human expression, firmly embedded in tradition. Another approach would be a cognitive perspective, where music is considered as an acoustical signal of which perception involves categorizations linked to representations and learning. A performance perspective where music is the outcome of human interaction is also an equally valid view. To understand a phenomenon combining multiple perspectives often makes sense.

The methods employed within each of these approaches turn questions into concrete musicological research projects. It is safe to say that today many of these methods draw upon digital data and tools. Some of those general methods are feature extraction from audio and movement signals, machine learning, classification and statistics. However, the problem is that, very often, the *empirical and computational methods require technical solutions* beyond the skills of researchers that typically have a humanities background. At that point, these researchers need access to specialized technical knowledge to advance their research. My PhD-work should be seen within the context of that tradition. In many respects I adopt a problem-solving attitude to problems that are posed by research in systematic musicology.

This work *explores solutions that are relevant for systematic musicology.* It does this by engineering solutions for measurement problems in empirical research and developing research software which facilitates computational research. These solutions are placed in an engineering-humanities plane. The first axis of the plane contrasts *services* with *methods*. Methods *in* systematic musicology propose ways to generate new insights in music related phenomena or contribute to how research can be done. Services *for* systematic musicology, on the other hand, support or automate research tasks which allow to change the scope of research. A shift in scope allows researchers to cope with larger data sets which offers a broader view on the phenomenon. The second axis indicates how important Music Information Retrieval (MIR) techniques are in a solution. MIR-techniques are contrasted with various techniques to support empirical research.

My research resulted in a total of thirteen solutions which are placed in this plane. The description of seven of these are bundled in this dissertation. Three fall into the methods category and four in the

services category. For example Tarsos presents a method to compare performance practice with theoretical scales on a large scale. SyncSink is an example of a service. It offers a solution for synchronization of multi-modal empirical data and enables researchers to easily use more streams of sensor data or to process more data from participants. Other services are TarsosDSP and Panako. The former offers real-time feature extraction and the latter an acoustic fingerprinting framework.

Generally, the solutions presented in this dissertation follow a reproducible methodology. Computational research and MIR research is often problematic to reproduce due to code that is not available, copyrights on music which prevent sharing evaluation data-sets and a lack of incentive to spend time on reproducible research. The works bundled here do pay attention to aspects relating to reproducibility. The software is made available under open source licenses and the systems are evaluated using publicly available music as much as possible. This makes processes transparent and systems verifiable. It also allows others, from in and outside academia, to use, criticize and improve the systems.

The main contributions of my doctoral research are found in the individual solutions. Panako (Six and Leman, 2014) contributes a new acoustic fingerprinting algorithm to academic literature. Subsequently applications of Panako for digital music archive management applications were described and evaluated (Six et al., 2018b). Tarsos (Six et al., 2013) facilitates large-scale tone scale use. I have contributed to the discussion on reproducibility and meaningful contributions to MIR (Six et al., 2018a). I have also presented a framework for active listening which enables augmented musical realities (Six and Leman, 2017). Next to these specific contributions the more general contributions include a way to conceptualize contributions to systematic musicology along a methods vs services axis and the concept of *augmented humanities* as a future direction of systematic musicological research.

# Outline

The first chapter outlines the problem and situates my research in the context of systematic musicology, engineering and digital humanities. It also introduces a plane in which solutions can be placed. One axis of this plane contrasts methods and services. The the other axis differentiates between MIR and other techniques. The first chapter continues with a section on the general methodology which covers aspects of reproducibility. It concludes with a summary.

The next two chapters (chapters 2 and 3) bundle seven publications in total. The publications bundled in these chapters only underwent minor cosmetic changes to fit the citation style and layout of this dissertation. Each publication has been subjected to peer review. The two chapters start with an additional introduction that focuses on how the works are placed in the broader framework of the overarching dissertation. Each introduction also contains bibliographical information which mentions co-authors together with information on the journal or conference proceeding where the research was originally published. I have limited the bundled publications to the ones for which I am the first author and which are most central to my research. This means that some works I co-authored are not included which keeps the length in check. Some of those works are Cornelis et al. (2013), Van Dyck and Six (2016) and Van Den Berge et al. (2018). However, they are situated in the plane introduced in chapter one. For a complete list of output, see Appendix A.

The fourth and final chapter offers a discussion together with concluding remarks. The contributions of my research are summarized there as well. Additionally, the term *augmented humanities* is introduced as a way to conceptualize future work. The main text ends with concluding remarks.

Finally, the appendix contains a list of output (Appendix A). Output in the form of software should be seen as an integral part of this dissertation so it is listed as well. The appendix also includes a list of figures, tables and acronyms (Appendix B). The last part of the dissertation lists referenced works.

**Reading guide** As a suggestion on how to best read this thesis I would propose to start with the individual articles (chapters 2 and 3). The articles are self-contained but perhaps it is best to start with chapter 3. The works presented in chapter 3 are used in some of the works in chapter 2. After reading the articles the introduction, which places the works in a context, will be easier to follow. Logically, the conclusion should be read last.

I would also encourage readers to run the described research software and experiment with the supplementary material. During this my research I have kept a research blog. It was updated continuously and includes presentations, lectures, multimedia documents, data sets and so forth. The software that was developed during my research is also available for download there, together with links to source code repositories and documentation. An up-to-date list of research output together with supplementary material (such as evaluation scripts) can be found on the same website. Supplementary material directly related to the dissertation is bundled on:

`http://0110.be/phd`.

# 1 Problem definition and methodology

## 1.1 Problem definition

*'Systematic musicology*[1] *has traditionally been conceived of as an interdisciplinary science, whose aim it is to explore the foundations of music from different points of view, such as acoustics, physiology, psychology, anthropology, music theory, sociology, and aesthetics'* (Leman and Schneider, 1997). It has various sub-disciplines such as music psychology, sociomusicology, music acoustics, cognitive neuroscience of music and the computer sciences of music which, in trun, has a significant overlap with music information retrieval (MIR) as well as sound and music computing.

One of the main research questions of systematic musicology is concerned with how people make sense of their musical environment. It deals with signification and meaning-formation and relates to how music empowers people (Lesaffre et al., 2017, part 5), how relations between musical structures and meaning formation should be understood and which effects music has. These *'fundamental questions are non-historical in nature'*(Duckles and Pasler, 2007) which contrasts systematic musicology with historical musicology.

There are many ways in which the research questions above can be approached or rephrased. For example, it can be approached from a cultural perspective, where music is considered a phenomenon of human expression embedded in tradition, and driven by innovation and creativity. The questions can be approached from a cognitive perspective, where music is considered information, or better: an acoustical signal, of which perception involves particular categorizations, cognitive structures, representations and ways of learning. Or they can be approached from a performance perspective, where music is considered as the outcome of human interactions, sensorimotor predictions and

[1] The term 'systematic musicology' is common in Continental Europe, being under the influence of German music research, but is less used in the UK and US. There, similar approaches are known as 'cognitive musicology', 'empirical musicology', 'systemic musicology' or 'interdisciplinary music studies' (Leman, 2008; Parncutt, 2007).

actions and where cognitive, perceptive processing goes together with physical activity, emotions, and expressive capacities. All these perspectives have their merits and to understand a phenomenon often a multi-perspective is adopted, based on bits and pieces taken from each approach.

The above list of approaches may not be exhaustive, but is only meant to indicate that there are many ways in which musicology approaches the question of meaning formation, signification, and empowerment. Likewise, there are many ways to construct an approach to the study of musical meaning formation which combines several perspectives. A more exhaustive historical overview of the different sides of musicology and the research questions driving (sub)fields is given by Duckles and Pasler (2007), Parncutt (2007) and Leman and Schneider (1997).

Accordingly, the same remark can be made with respect to the methods that turn the above mentioned approaches and perspectives into concrete musicological research projects. It would be possible to list different methods that are used in musicology but it is not my intention to attempt to give such a list, and certainly not an exhaustive list. Instead, what concerns me here, is the level below these research perspectives. One could call it the foundational level of the different methodologies that materialize the human science perspectives.

At this point I believe that it is safe to say that many of the advanced research methodologies in musicology today draw upon digital data and digital tools[2]. To name only a few, digital data ranges from audio/video recordings, motion capture data, to digitally encoded interviews. Analysis methods for this type of data can be assisted by, or even require, digital solutions. Sometimes, they are readily available but very often the digital solutions are only partly available or simply lacking. Almost by definition, there is lack of digital solutions in innovative research environments.

Research in systematic musicology provides a typical example of such an environment. This research tradition is at the forefront of development in which advanced digital data and tools are used and required. Bader (2017); Lesaffre et al. (2017) compiled an up-to-date reference work. Indeed, while the research topics in systematic musicology have kept their typical humanities flavor – notions such as 'expression', 'value', 'intention', 'meaning', 'agency' and so on are quite common – the research methods have gradually evolved in the direction of empirical and computational methods that are typically found in the natural sciences (Honing, 2004). A few examples of such general methods are

[2]Some researchers working in music centered ethnography (like the work by Reed (2016)) almost avoid technology as much as possible. However, I would argue that proper digital tooling could also benefit such methods.

feature extraction from audio and movement signals, machine learning, classification and statistics. This combination of methods gives systematic musicology its inter-disciplinary character.

However, the problem is that, very often, the empirical and *computational methods require technical solutions* beyond the skills of researchers who typically have a humanities background. At that point, researchers need access to specialized technical knowledge to advance their research.

Let me give a concrete example to clarify my point. The example comes from a study about African music, where I collaborated with musicologist and composer dr. Olmo Cornelis on an analysis of the unique and rich archive of audio recordings at the Royal Museum of Central Africa (Cornelis et al., 2005). The research question inquired music scales: have these scales changed over time due to African acculturation to European influences? Given the large number of audio recordings (approximately 35000), it is useful to apply automatic music information retrieval tools that assists the analysis; for example, tools that can extract scales from the recordings automatically, and tools that can compare the scales from African music with scales from European music. Traditionally, such tools are not made by musicologists that do this kind of analysis, but by engineers that provide digital solutions to such a demand. If the engineers do not provide the tools, then the analysis is not possible or extremely difficult and time consuming. However, if the musicologists do not engage with engineers to specify needs in view of a research question, then engineers cannot provide adequate tools. Therefore, both the engineer and the musicologist have to collaborate in *close interaction* in order to proceed and advance research[3] This aspect of close interaction is crucial: successful collaboration thrives on shared understanding and continuous feedback. Simply handing over a set of requirements or demands is a surefire way to achieve unsatisfactory results. Systematic musicology is one of the more successful fields where this type of interdisciplinary collaboration has happened.

This dissertation should be seen within the context of that tradition. In many respects I adopt a problem-solving attitude to problems that are posed by research in systematic musicology. Often the problems themselves are ill-defined. My task, therefore, is to break down ill-defined problems and to offer well-defined solutions. Bridging this gap requires close collaboration with researchers from the humanities and continuous feedback on solutions to gain a deep understanding of the problems at hand. To summarize my research goal in one sentence:

[3]Note that 'the engineer' and 'the musicologist' refer to the traditional role of each not necessarily to different individuals. A musicologist with strong engineering background could assume the role of both or vice versa.

*the goal of my research is to engineer solutions that are relevant for systematic musicology.*

Overall, it is possible to consider this contribution from two different axes. One axis covers the link between engineering *methods* and engineering *services* that are relevant to musicology. The other axis covers the link between different engineering techniques. They either draw on *Music Information Retrieval* techniques or on a number of other techniques here categorized as *techniques for empirical research.* These two axes together define a plane. This plane could be called the engineering-humanities plane since it offers a way to think about engineering contributions *in* and *for* the humanities. Note that the relation between this work and the humanities will be more clearly explained in a section below (see 1.1.4). It allows to situate my solutions, as shown in Figure 1.1.

### 1.1.1 Services versus methods

The vertical axis specifies engineering either a *service* or a *method.* Some solutions have aspects of both services and methods and are placed more in the center of this axis. Vanhoutte (2013, p 120) makes a similar distinction but calls it: computing *for* and computation *in* humanities. Computation for humanities is defined as the *'instrumental use of computing for the sake of humanities'.* The engineering solutions are meant to support or facilitate research, and therefore, computation-for can also be seen as a *service* provided by engineering. Computation-in humanities is defined as that what *'actively contributes to meaning-generation'.* The engineering solutions are meant to be used as part of the research methodology, for example, to gather new insights by modeling aspects of cultural activities, and therefore, the computation-in can be seen as *methods* provided by engineering.

In my PhD I explore this axis actively by building tools, prototypes, and experimental set-ups. Collectively these are called *solutions.* They present *a* solution in a specific context not *the* solution. They are subsequently applied in musicological research. My role is to engineer innovative solutions which support or offer opportunities for new types of research questions in musicology.

A word processor that allows a researcher to easily lay-out and edit a scientific article is an example of a solution situated at the *service-for* side. Nowadays, this seems like a trivial example but employing word processors greatly expedites a key research task: describing research. TeX, the predecessor of the typesetting system used for this dissertation, was invented specifically as a service for the (computer) science community by Knuth (1979). For the interested reader: a technological history of word processor and its effects on literary writing is described

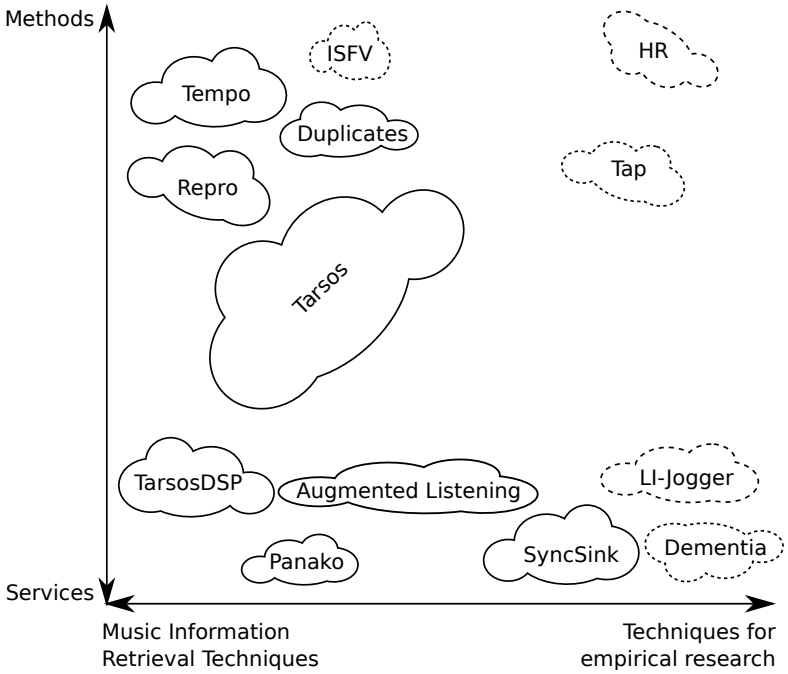



Figure 1.1: The engineering-humanities plane. Different 'solutions' for digital services and tools are situated in a plane defined by two axes. One axis confronts services and methods while the other axis defines techniques. The solutions with dashed lines are not bundled in this dissertation, the others are.

by Kirschenbaum (2016).

The *services-for* are contrasted with *methods-in*. Software situated at the *method-in* side is for example a specialized piece of software that models dance movements and is able to capture, categorize and describe prototypical gestures, so that new insights into that dance repertoire can be generated. It can be seen as a method in the humanities. The distinction can become blurry when solutions appear as *method-in* and as *service-for* depending on the usage context. An example is Tarsos (Figure 1.1, Six et al. (2013), the article on Tarsos is also included in section 2.2). If Tarsos is used to verify a pitch structure it is used as a *service-for*. If research is done on pitch structures of a whole repertoire and generates novel insights it can be seen as *method-in*.

**Engineering solutions become methods-in humanities**

The relationship between computing and humanities, to which musicology is a sub-discipline, has been actively investigated in the digital humanities[4] field. One of the seminal works on this topic is by McCarty (2005, p. 197). In this work McCarty recognized that cultural artifacts are incredibly diverse and multi-faceted. Artifacts here are broadly defined as expressions of a culture such as poems, paintings, songs or even architectural traditions. There can be multiple actors interacting with cultural artifacts in various modalities, and these interactions all contribute to a multi-layered meaning that is attached to those artifacts. However, given the multi-layered meanings, it often happens that research requires a well-defined specific perspective about the artifact. The analysis may then result in a set of discrete components and relations about the artifact, which can be encoded into a digital format and approached with a computational model. At this point humanities and computer science meet as can be seen in Figure 1.2.

Please note that I do not deal with the question whether the specific perspective is relevant for the multi-layered meanings existing in a population of users of those artifacts. What interests me here is how engineering can provide solutions to methods used by scholars studying cultural artifacts, even if these methods cover only part of the multi-layered meaning that is attached to these artifacts.

Three steps can be distinguished to com to a solution. In the first step, the solution does not take into account a specific hardware or software implementation yet. Rather, the goal is to get a clear view on the scholar's approach to the analysis of cultural artifacts. In the second step, then, it is necessary to take into account the inherent

[4]McCarty (2005) prefers the term *humanities computing* over *Digital Humanities*. More about this follows in section 1.1.4.

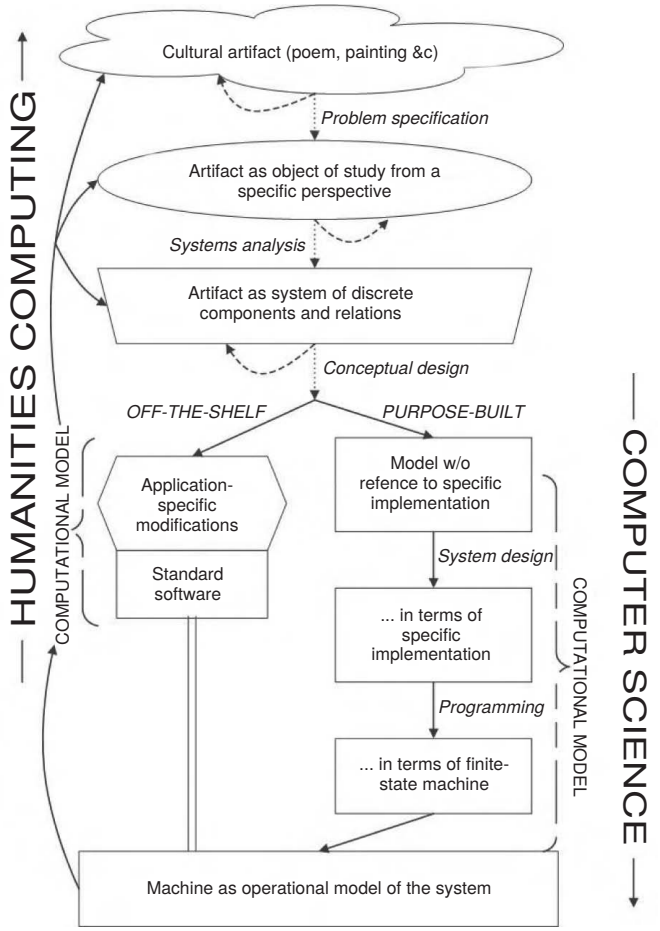



Figure 1.2: A schema that relates humanities computing to computer science. It details the relationship between components that can solve a research question related to cultural artifacts with computational tools. Note that this schema focuses on methods, services are not considered. Copied with permission from McCarty (2005, p. 197)

restrictions of an actual hard- or software implementation. Finally, in the third step, an implementation follows. This results in a working tool, a solution. The solution can be seen as a model of the method. It works as a detailed, (quasi) deterministic model of a methodology.

Rather than building new solutions it is possible that a researcher finds an off-the shelf solution that is readily available. However, it often turns out that a set of thoughtful application-specific modifications may be necessary in order to make the standard piece of software applicable to the study of specific cultural artifacts.

While the above mentioned steps towards an engineering solution may appear as a rather deterministic algorithm, reality is much different. Due to technical limitations the last two steps may strongly influence the 'preceding' step. Or to put a more positive spin to it: the technical possibilities and *opportunities*, may in turn influence the research questions of a researcher in the humanities. To go even one step further, technical solutions can be an incentive for inventing and adopting totally new methods for the study of cultural artifacts. Accordingly, it is this interchange between the technical implementation, modeling and application in and for humanities that forms the central theme of my dissertation.

This view, where technical innovations serve as a catalyst for scientific pursuits in the humanities, reverses the idea that a humanities scholar should find a subservient engineer to implement a computational model. It replaces subservience with an equal partnership between engineers and humanities scholars. The basic argument is that technical innovations often have a profound effect on how research is conducted, and technical solutions may even redirect the research questions that can be meaningfully handled. In this section I have been exclusively dealing with computational solutions for modeling cultural artifacts (*methods-in*). Now it is time to go into detail on *services-for* which may also have a profound effect on research.

**Engineering solutions become services-for humanities**

Engineering solutions for humanities are often related to the automation or facilitation of *research tasks*. Such solutions facilitate research in humanities in general, or musicology in particular, but the solutions cannot be considered methods. The research tasks may concern tasks that perhaps can be done by hand but these are tedious, error-prone and/or time consuming. When automated by soft- and hardware systems, they may be of great help to the researcher so that the focus can be directed towards solving a research question instead of practical matters. A simple example is a questionnaire. When done on paper, a lot of time is needed to transcribe data in a workable format. However,

when filled out digitally, it may be easy to get the data in a workable format.

Solutions that work as services for the humanities often have the power to change the scope of research. Without engineering a solution, a researcher may have been able to analyze a selected subset of artifacts. With engineering solution, a researcher may be able to analyze large quantities of artifacts. Successful services are the result of close collaboration and tight integration. Again, I claim that equal partnership between humanist and engineers is a better model to understand how services materialize in practice.

For example, the pitch analysis tool implemented in Tarsos (Six et al., 2013) can handle the entire collection of 35000 recordings of the KMMA collection. Accordingly, the scope of research changes dramatically. From manual analysis of a small set of perhaps 100 songs, to automatic analysis of the entire collection over a period of about 100 years. This long-term perspective opens up entirely new research questions, such as whether Western influence affected tone-scale use in African music. Note that by offering Tarsos as a service, methods may need to be reevaluated.

Another example in this domain concerns the synchronization of different data streams used in musical performance studies. Research on the interaction between movement and music indeed involves the analysis of multi-track audio, video streams and sensor data. These different data streams need to be synchronized in order to make a meaningful analysis possible. My solution offers an efficient way to synchronize all these data streams (Six and Leman, 2015). This solution saves a lot of effort and tedious alignment work that otherwise has to be done by researchers whose focus is not on the synchronization of media, but on the study of musical performance. The solution is also the result of an analysis of the research needs on the work floor and has been put in practice (Desmet et al., 2017). Again, it enables research on a different scope: a larger number of independent sensor streams with more participants can be easily handled.

To summarize the method-services axis: methods have cultural artifacts at the center and generate new insights, whilst services facilitate research tasks which have the potential to profoundly influence research praxis (e.g. research scope). The other axis in the plane deals with the centrality of MIR-techniques in each solution.

### 1.1.2 MIR-techniques versus techniques for empirical research

The second axis of the engineering-humanities plane specifies an engineering approach. It maps how central music information retrieval

(MIR) techniques are in each solution. The category of techniques for empirical research includes sensor technology, micro-controller programming, analog to digital conversion and scripting techniques. The MIR-techniques turned out to be very useful for work on large music collections, while the techniques in analogue-digital engineering turned out to be useful for the experimental work in musicology.

One of the interesting aspects of my contribution, I believe, is concerned with solutions that are situated in between MIR and tools for experimental work, such as the works described in Six and Leman (2015) and Six and Leman (2017). To understand this, I provide some background to the techniques used along this axis. To get a grasp of the techniques used, it is perhaps best to start with a short introduction to MIR and related fields.

### Symbol-based MIR

The most generally agreed on definition of MIR is given by Downie (2004).

> *"A multidisciplinary research endeavor that strives to develop innovative content-based searching schemes, novel interfaces, and evolving networked delivery mechanisms in an effort to make the world's vast store of music accessible to all."*

Originally the field was mainly dedicated to the analysis of symbolic music data. A music score, encoded in a machine readable way, was the main research object. Starting from the 1960s computers became more available and the terms *computational musicology* and *music information retrieval* were coined. The terms immediately hint at the duality between searching for music - accessibility, information retrieval - and improved understanding of the material: computational musicology. Burgoyne et al. (2016) provides an excellent introduction and historic overview of the MIR research field.

### Signal-based MIR

As computers became more powerful in the mid to late nineties, desktop computers performed better and better on digital signal processing tasks. Combined with advances in audio compression techniques, cheaper digital storage, and accessibility to Internet technologies, this led to vast amounts of digital music and big data collections. The availability of large data sets, in turn, boosted research in MIR but now with musical signals at the center of attention.

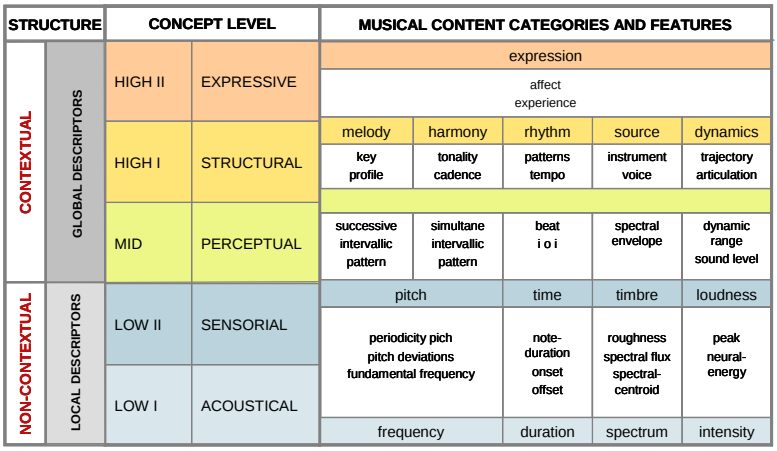



Figure 1.3: Conceptual MIR framework connecting high level concepts and low level features. Reproduced with permission from Lesaffre (2006)

Signal based MIR aims to extract descriptors from musical audio. MIR-techniques are based on low-level feature extraction and on classification into higher-level descriptors. Low-level features contain non-contextual information close to the acoustic domain such as frequency spectrum, duration and energy. Higher level musical content-description focuses on aspects such as timbre, pitch, melody, rhythm and tempo. The highest level is about expressive, perceptual contextual interpretations that typically focus on factors related to movement, emotion, or corporeal aspects. This concept has been captured in a schema by Lesaffre (2006), see Figure 1.3. The main research question in MIR is often on how to turn a set of low-level features into a set of higher level concepts.

For example, harmony detection can be divided into low-level instantaneous frequency detection and a perceptual model that transforms frequencies into pitch estimations. Multiple pitch estimations are integrated over time - contextualized - and contrasted with tonal harmony resulting in harmony estimation. In the taxonomy of Lesaffre (2006), this means going from the acoustical over the sensorial and perceptual to the cognitive or structural level.

Usually, the audio-mining techniques deliver descriptions for large sets of audio. The automated approach is time-saving and often applied to audio collections that are too large to annotate manually. The descriptors related to such large collections and archives provide a basis

for further analysis. In that context, data-mining techniques become useful. The techniques focus on the computational analysis of a large volume of data, using statistical correlation and categorization techniques that look for patterns, tendencies, groupings and changes.

A more extensive overview of the problems that are researched in MIR is given in the the books by Klapuri and Davy (2006), Müller (2015) and Ras and Wieczorkowska (2010). The overview article by Burgoyne et al. (2016) gives more insights in the history of MIR. The proceedings of the yearly ISMIR[5] conferences, the main conference in MIR, give a good sense of the topics that are in vogue.

### MIR and other research fields

Despite the historical close connection between symbolic-based MIR and computational musicology, the link between signal-based MIR and music cognition has not been as strong. At first sight, signal-based MIR and music cognition are both looking at how humans perceive, model and interact with music. However, MIR is more interested in looking for pragmatic solutions for concrete problems and less in explaining processes. Music cognition research, on the other hand, is more interested in explaining psychological and neurological processes. The gap between the fields has been described by Aucouturier and Bigand (2012, 2013).

An example in *instrument recognition* may clarify the issue. In the eyes of a MIR researcher, instrument recognition is a matter of *'applying instrument labels correctly to a music data set'*. Typically, the MIR researcher extracts one or more low-level features from music with the instruments of interest, trains a classifier and applies it to unlabeled music. Finally the MIR researcher shows that the approach improves the current state of the art. Loughran et al. (2008) present such an algorithm, in this case based on MFCC[6]'s. The result is valuable and offers a pragmatic approach to tag a large dataset with instrument labels potentially improving its accessibility. Typically, in this approach, the underlying processes, assumptions or perceptual models are not made explicit. The paper focuses on *results* not on *processes*, arguably it focuses on applied *engineering* and less on explanatory *science*. Similar methodological concerns are raised by Van Balen (2016, p.94).

In contrast, music cognition research tends to approach the same task from a different perspective. The question boils down to *'How are we able to recognize instruments?'*. Neuropsychological experiments

[5]The ISMIR website is http://ismir.net. Proceedings are available as well.

[6]Mel-frequency cepstrum coeficcients (MFCCs) are low level audio features which represent aspects of the short term power spectrum of a sound.

carried out by Omar et al. (2010) suggest how music instrument recognition is processed. The result is valuable and offers useful insights in processes. However, the result does not mention how the findings could be exploited using a computational model.

MIR and music cognition can be considered as two prongs of the same fork. Unfortunately the handle seems missing: there are a few exceptions that tried to combine insights into auditory perception with computational modeling and MIR-style feature extraction. One such exception is the IPEM-Toolbox (Leman et al., 2001). While this approach was picked up and elaborated in the music cognition field (Bigand et al., 2014; Koelsch et al., 2007; Marmel et al., 2010), the approach fell on deaf ears in the MIR community. Still, in recent years, some papers claim that it is possible to improve the standards, the evaluation practices, and the reproducibility of MIR research by incorporating more perception-based computational tools. This approach may generate some impact in MIR (Aucouturier and Bigand, 2013; Flexer et al., 2012; Peeters and Fort, 2012; Sturm, 2016).

The specific relation between MIR and systematic musicology is examined in a paper by Leman (2008). It is an elaboration on a lecture given at the University of Cologne in 2003, which had the provocative title *"Who stole musicology?"*. Leman observed that in the early 2000's there was a sudden jump in a number of researchers working on music. While the number of musicology scholars remained relatively small, engineers and neuroscientists massively flocked to music. They offered intricate computational models and fresh views on music perception and performance. Engineers and neuroscientists actively and methodologically contributed to the understanding of music with such advances and in such large numbers that Leman stated "*if music could be better studied by specialized disciplines, then systematic musicology had no longer a value*". However, Leman further argues that there is a value in modern systematic musicology that is difficult to 'steal', which is 'music'. This value (of music) depends on the possibility to develop a trans-disciplinary research methodology, while also paying attention to a strong corporeal aspect that is currently largely ignored by other fields. This aspect includes *"... the viewpoint that music is related to the interaction between body, mind, and physical environment; in a way that does justice to how humans perceive and act in the world, how they use their senses, their feelings, their emotions, their cognitive apparatus and social interactions"*. While this point was elaborated in his book on embodied music cognition (Leman, 2007), it turns out that this holistic approach is still - ten years later - only very rarely encountered in MIR research.

**Computational Ethnomusicology**

It is known that the signal-based methods developed by the MIR community target classical Western music or commercial pop by an overwhelming majority (Cornelis et al., 2010a), whereas the immense diversity in music all over the world is largely ignored[7]. Similarly, in symbolic-based MIR, scholars developed machine readable encodings and retrieval systems focusing almost exclusively on scores with modern Western staff notation, whereas other notation systems or music encoding methods were left unexplored. That means that MIR suffers from an ethnocentric viewpoint: its concepts on rhythm, pitch organization, genres, labels, meta-data taxonomies are deeply rooted into Western music theory and as a result, the MIR tools are not generalizable, or inclusive. For an engineer it means that the available tools are limited and that the demands from a musicologist may be quite remote from the tools that MIR can offer today.

For example, a chroma feature shows the intensity of each of the 12 Western notes at a point in time in a musical piece. Chroma features subsequently imply a tonal organization with an octave divided in 12 equal parts, preferably with $A4$ tuned to 440Hz. Methods that build upon such chroma features perform well on Western music but they typically fail on non-Western music that has other tonal organization. By itself this is no problem, it is simply a limitation of the method (model) and chroma can be adapted for other tonal organizations. However, the limitation is a problem when such a tool would be applied to music that does not have a tonal space that the tool can readily measure. In other words, it is necessary to keep in mind that the toolbox of a MIR researcher is full of methods that make assumptions about music, while these assumptions do not hold universally. Therefore, one should be careful about Western music concepts in standardized MIR methods. They can not be applied equally to other *musics* without careful consideration.

Computational ethnomusicology is a research field in which MIR tools are adopted or re-purposed so that they can provide specialized methods and models for all kinds of musics. For a more detailed discussion on this see page eight of Cornelis (2013). The field aims to provide better access to different musics and to offer a broader view, including video, dance and performance analysis. Tzanetakis et al. (2007) redefine this field[8] and give an overview of works done in the field of computational ethnomusicology. This type of engineering solu-

[7]This improved slightly in recent years (Gómez et al., 2013). For example, many results of the CompMusic project (Serra, 2011), a project with a focus on several music traditions, were presented at ISMIR, the main MIR conference.

[8]One of the earlier definitions is given by Halmos (1978).

tions are much needed by musicologists who manage large collections of digitized historic ethnic music available in several museums (Cornelis et al., 2010b).

Accordingly, re-use of MIR tools demands either a culture specific approach or a general approach. In the first case specific structural elements of the music under analysis are encoded into models. For example in Afro-Cuban music, elements specific to the 'clave' can be encoded so that a better understanding of timing in that type of music becomes possible (Wright et al., 2008). While this solution may limit the applicability to a specific (sub)culture, it also allows deep insights in mid- and high-level concepts of a possible instantiation of music. In that sense, the MIR solutions to harmonic analysis can be seen as a culture specific approach, yielding meaningful results for Western tonal music only.

One of the goals of the MIR solution is to understand elements in music that are universal. In that sense, it corresponds quite well with what systematic musicology is aiming at: finding effects of music on humans, independent of cultures or historical period. Given that perspective, one could say that rhythm and pitch are fundamental elements of such universals in music. They appear almost always across all cultures and all periods of time. Solutions that offer insights in frequently reused pitches for example, may therefore be generally applicable. They have been applied to African, Turkish, Indian and Swedish folk music (Chordia and Rae, 2007; Gedik and Bozkurt, 2010; Six et al., 2013; Sundberg and Tjernlund, 1969). Most often, however, such solutions are limited to low-level musical concepts. The choice seems to boil down to: being low-level and universal, or high-level and culture-specific.

Both culture-specific and what could be called culture-invariant approaches to computational ethnomusicology should further understanding and allow innovative research. Computational ethnomusicology is defined by Tzanetakis et al. (2007) as *"the design, development and usage of computer tools that have the potential to assist in ethnomusicological research."*. This limits the research field to a *service-for* ethnomusicology. I would argue that the *method-in* should be part of this field as well. A view that is shared, in slightly different terms, by Gómez et al. (2013): *"computer models can be 'theories' or 'hypotheses' (not just 'tools'. . . ) about processes and problems studied by traditional ethnomusicologists"*. While computational ethnomusicology is broader in scope, the underlying techniques have a lot in common with standard MIR-techniques which are present also in my own research.

### MIR-techniques

With the history and context of MIR in mind it is now possible to highlight the techniques in my own work. It helps to keep the taxonomy by Lesaffre (2006) in mind (see figure 1.3).

Tarsos (Six et al., 2013) is a typical signal-based MIR-system in that it draws upon low level pitch estimation features and combines those to form higher level insights: pitch and pitch class histograms related to scales and pitch use. Tarsos also offers similarity measures for these higher level features. It allows to define how close two musical pieces are in that specific sense. In the case of Tarsos these are encoded as histogram similarity measures (overlap, intersection). Several means to encode, process and compare pitch interval sets are present as well. Tarsos has a plug-in system to allow to start from any system that is able to extract pitch from audio. By default the TarsosDSP (Six et al., 2014) library is used.

TarsosDSP is a low-level feature extractor. As mentioned previously, it has several pitch extractor algorithms but also includes extraction of onset and beat tracking. It is also capable to extract spectral features and much more. For details see 3.2. It is a robust foundation to build MIR systems on. Tarsos is one example but my work in acoustic fingerprinting also is based on TarsosDSP.

The acoustic fingerprinting work mainly draws on spectral representations (FFT or Constant-Q)[9] and robust peaks in the frequency domain. These peaks are the low level features for this application. The peaks are then combined and indexed in a database which allows efficient lookup. Lookup of small audio queries is one of the quintessential information retrieval tasks. While I have contributed to the academic literature on acoustic fingerprinting with Six and Leman (2014), I have applied this MIR-technique in various less straightforward ways (see Six and Leman (2017) and Six et al. (2018b)). I have also employed it as a technique to support empirical research (Six and Leman, 2015) together with various other techniques.

### Techniques for empirical research

While MIR-techniques form an important component in this research, in this case they are contrasted with various techniques for empirical research. The solutions that appear at the other extreme of the horizontal axis of the plane do not use MIR-techniques. However, there are some solutions for empirical research that do use MIR techniques

[9] Both an Fast Fourier Transform (FFT) and a Constant-Q (Brown and Puckette, 1992) transform are methods to take a time-domain signal and transform it to the frequency domain.

to some degree. These solutions are designed to support specific experiments with particular experimental designs. While the experimental designs can differ a lot, they do share several components that appear regularly. Each of these can present an experimenter with a technical challenge. I can identify five:

1. **Activation** The first component is to present a subject with a stimulus. Examples of stimuli are sounds that need to be presented with millisecond accurate timing or tactile feedback with vibration motors or music modified in a certain way.

2. **Measurement** The second component is the measurement of the phenomenon of interest. It is essential to use sensors that capture the phenomenon precisely and that these do not interfere with the task for the subject. Examples of measurement devices are wearables to capture a musicians movement, microphones to capture sound, video cameras and motion capture systems. Note that a combination of measurement devices is often needed.

3. **Transmission** The third component aims to expose measurements in a usable form. This can involve translation via calibration to a workable unit (Euler angles, acceleration expressed in g-forces, strain-gauge measurements in kg). For example, it may be needed to go from raw sensor readings on a micro-controller to a computer system or from multiple measurement nodes to a central system.

4. **Accumulation** The fourth component deals with aggregating, synchronizing and storage of measured data. For example, it might be needed to capture measurements and events in consistently named text files with time stamps or a shared clock.

5. **Analysis** The final component is analysis of the measured data with the aim to support or disprove a hypothesis with a certain degree of confidence. Often standard (statistical) software suffices, but it might be needed to build custom solutions for the analysis step as well.

These components need to be combined to reflect the experimental design and to form reliable conclusions. Note that each of the components can either be trivial and straightforward or pose a significant challenge. A typical experiment combines available off-the shelf hardware and software solutions with custom solutions which allows successful use in an experimental setting. In innovative experiments it is rare to find designs completely devoid of technical challenges.

For example, take the solution devised for Van Dyck et al. (2017). The study sheds light on the effect of music on heart-rate in rest. It uses musical fragments that are time-stretched to match the subjects' heart rate as stimulus (activation). A heart-rate sensor is the main measurement device (measurement). A micro-controller, in this case an Arduino, takes the sensor values and sends them (transmission) to a computer. These components are controlled by a small program on the computer that initiates each component at the expected time resulting in a research data set (accumulation) that can be analyzed (analysis). In this case the data are 'changes in heart-rate' and 'heart-rate variability' when subjects are in rest or listen to specific music. The technical challenges and contributions were mainly found in high-quality time-stretching of the stimuli (activation) and reliably measuring heart rate at the fingertips (measurement). The other components could be handled with off-the-shelf components. Note that measuring heart rate with chest straps might have been more reliable but this would have been also more invasive, especially for female participants. Aspects of user-friendliness are almost always a concern and even more so if measurement devices interfere with the task: in this setup participants needed to feel comfortable in order to relax.

Another example is the solution called the LI-Jogger (short for Low Impact jogger, see Van Den Berge et al. (2018)). A schema of the set-up can be found in Figure 1.4. This research aims to build a system to lower footfall impact for at-risk amateur runners via biofeedback with music. Foot fall impact is a known risk factor of a common running injury and lowering this impact in turn lowers this risk. Measurement involves measuring peak tibial (shin) acceleration in three dimensions with a high time resolution via a wearable sensor on the runner. Running speed needs to be controlled. This is done via a sonar (as described by Lorenzoni et al. (2017)). Accumulation of the data is a technical challenge as well since footfall measured by the accelerometer needs to be synchronized precisely with other sensors embedded in the sports science lab. To allow this, an additional infra-red (IR) sensor was used to capture the clock of the motion capture system. The MoCap system shares this clock, via wires, with several other measurement devices (force plates). In this case a superfluous stream of measurements was required to allow synchronization. In this project each component is challenging:

1. **Activation**. The biofeedback needs measurements in real-time and modifies music to reflect these measurements in a certain way. This biofeedback is done on a battery-powered wearable device that should hinder the runner as little as possible.

2. **Measurement**. The measurement requires a custom system

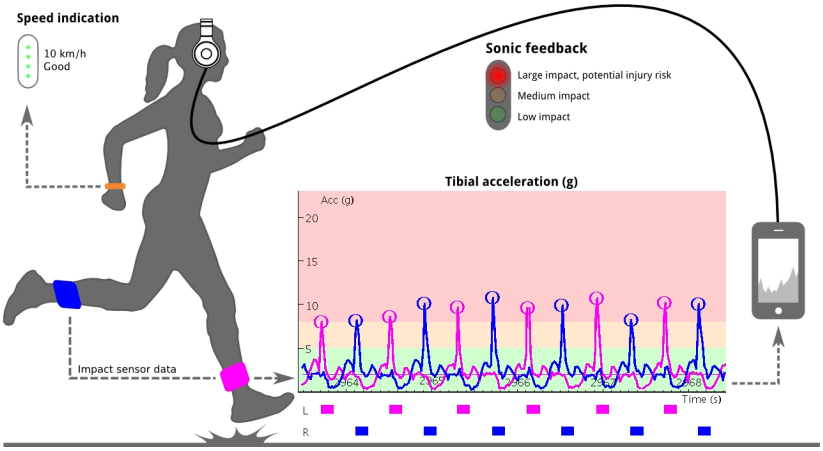



Figure 1.4: LI jogger schema. The aim is to lower footfall impact by providing a runner with musical feedback. Footfalls are measured with accelerometers at each ankle. The acceleration peaks are identified on a wearable computing device (right) which immediately sends sonic feedback to the runner (via headphones). The feedback is simplified here using the traffic lights as 'large', 'medium' or 'low' impact. To control for speed a wristband notifies the runner to speed up, slow down or keep the current pace. Each aspect of the experimental setup requires a customized approach.

with high-quality 3D accelerometers. The accelerometers need to be light and unobtrusive. To allow synchronization and speed control an additional IR-sensor and sonar were needed.

3. **Transmission**. To expose acceleration and other sensor data custom software is required on the transmitting micro-controller and on the receiving system.

4. **Accumulation**. Accumulation requires scripts that use the IR-sensor stream to synchronize acceleration data with a motion capture system and other measurement devices (force-plate, footroll measurement).

5. **Analysis**. Analysis of the multi-modal data is also non-trivial. The article that validates the measurement system (Van Den Berge et al., 2018) also requires custom software to compare the gold-standard force-plate data and acceleration data.

The techniques used in these tools for empirical research are often relatively mundane. However, they do span a broad array of hardware and software technologies that need to be combined efficiently to offer a workable solution. Often there is considerable creativity involved in engineering a cost-effective, practical solution in a limited amount of time. As can be seen from the examples it helps to be able to incorporate knowledge about micro-controllers, sensors, analog/digital conversion, transmission protocols, wireless technologies, data-analysis techniques, off-the-shelf components, scripting techniques, algorithms and data structures.

Now that the two axes of the humanities-engineering plane have been sufficiently clarified it is time to situate my solutions in this plane.

### 1.1.3 Situating my solutions in the humanities-engineering plane

Given the above explanation of the axes, it is now possible to assign a location in this plane for each of the solutions that form the core of my doctoral research (Figure 1.1):

**Tempo** This solution introduces and validates a method to gauge the metric complexity of a musical piece, depending on the level of agreement between automatic beat estimation algorithms. The validation is done by comparing expert human annotations with annotations by a committee of beat estimation algorithms. The solution is based on

MIR-techniques and it is used as a method to get insights into rhythmic ambiguity in sets of music. It is, therefore, placed in the upper left of the engineering-humanities plane. It is described by Cornelis et al. (2013) and not included in this dissertation.

**Tarsos** This solution introduces a method to automatically extract pitch and pitch class histograms from any pitched musical recording. It also offers many tools to process the pitch data. It has a graphical user interface but also a batch processing option. It can be used as on a single recording or on large databases. It covers quite a big area in the plane: depending on the research it can be used as a method for large scale scale analysis or as a service to get insights into pitch use of a single musical piece. Tarsos is situated to the side of MIR-techniques since it depends on techniques as feature extraction. It is described in Six et al. (2013), which is included in chapter 2.2.

**Repro** The main contribution of this work is a reflection on reproducible research methodologies for computational research in general and MIR research in particular. As an illustration a seminal MIR-article is replicated and a reproducible evaluation method is presented. The focus is on methods of computational research and it utilizes MIR-techniques so it is placed at the methods/MIR-techniques side. It is described in Six et al. (2018a), which is included in chapter 2.3.

**Duplicates** This solution presents a method to compare meta-data, reuse segmentation boundaries, improves listening experiences and to merge digital audio. The method is applied to the data set of the RMCA archive which offers new insights into the meta-data quality. The underlying technique is acoustic fingerprinting, a classical MIR-technique. The case-study uses a service provided by Six and Leman (2014). The article (Six et al., 2018b) is included in chapter 2.4.

**TarsosDSP** This digital signal processing (DSP) library is the foundation of Tarsos. TarsosDSP offers many low-level feature extraction algorithms in a package aimed for MIR-researchers, students or developers. It is a piece of work in its own right and is situated on the MIR-techniques/service side. The service is employed in TarsosDSP. It has been used for speech rehabilitation (Bujas et al., 2017), serious gaming contexts (Sandulescu et al., 2015) and human machine interaction (Reyes et al., 2016). The article (Six et al., 2014), is included in chapter 3.2

**Panako** This solution is an acoustic fingerprinting algorithm that allows efficient lookup of small audio excerpts in large reference databases. Panako works even if the audio underwent changes in pitch. It is placed firmly in the MIR side and service side. There are many ways in which Panako can be used to manage large music archives. These different ways are discussed in Bressan et al. (2017b). See Six and Leman (2014), which is included in chapter 3.3.

**Augmented listening** This solution offers a technology for augmented listening experiences. It is effectively proving the tools for a computer-mediated reality. As with a typical augmented reality technology it takes the context of a user and modifies – augments or diminishes – it with additional layers of information. In this work the music playing in the environment of the user is identified with precise timing. This allows to enrich listening experiences. The solution employs MIR-techniques to improve engagement of a listener with the music in the environment. There are, however, also applications to use this in experimental designs for empirical research. So it is a service situated in between MIR-techniques and techniques for empirical research. It is described by Six and Leman (2017), which is included in chapter 3.5.

**SyncSink** This solution presents a general system to synchronize heterogeneous experimental data streams. By adding an audio stream to each sensor stream, the problem of synchronization is reduced to audio-to-audio alignment. It employs MIR-technique to solve a problem often faced in empirical research. The service is placed more to the side of techniques for empirical research . Van Assche (2016) extended the off-line algorithm with a real-time version in his master's thesis. The service is described in Six and Leman (2015), which is included in chapter 3.4.

**Tap** This empirical study compares tapping behaviour when subjects are presented with with tactile, auditory and combined tactile and auditory queues. To allow this, a system is required that can register tapping behaviour and present subjects with stimuli with a very precise timing. The main aim of the study is to present a method and report results on the tapping behaviour by subjects. So while the measurement/stimulus system could be seen as a service, the main contribution lies in the method and results. These are described in Six et al. (2017) which is not included in this dissertation.

**LI-Jogger** This solution is a soft/hardware system that allows immediate feedback of foot-fall impact to allow auditory feedback with

music. It has been used in the previous section as an example (see 1.1.2) where every aspect of an experimental design poses a significant challenge. The innovative aspects are that impact is measured at a high data rate (1000Hz) in three dimensions by a wearable system that is synchronized precisely with other measurement modalities. LI-Jogger supports an empirical research so it is placed right in the plane. It aims to provide a view into how music modifies overground (vs treadmill) running movement but the system itself is a service needed to achieve that goal. This solution is described in (Van Den Berge et al., 2018). A large intervention study that will apply this solution is planned. The results of this study will be described in follow-up articles

**ISFV** This solution includes an analysis and comparison of harmonics in a singing voice while inhaling and exhaling. MIR-techniques are used to gain insights in this performance practice. It is, therefore, situated at the methods/MIR-techniques side of the plan. My contribution was primarily in the data-analysis part. The findings have been described in (Vanhecke et al., 2017), which is not included in this dissertation.

**HR** This solution is a heart-rate measurement system with supporting software to initiate and record conditions and present stimuli to participants. The stimulus is music with modified tempo to match a subjects heart-rate. It has been used in the previous section as an example see 1.1.2. The system made the limited influence of music on heart-rate clear in a systematic way. It is situated in the upper right quadrant. See (Van Dyck et al., 2017).

**Dementia** This solution concerns a hard and software system to measure engagement with live performance versus prerecorded performances for patients with dementia. My contribution for it is in the software that combines various movement sensors and web-cameras that register engagement and in presenting the stimuli. The sensors streams and videos are synchronized automatically. Figure 1.5 shows synchronized video, audio and movement imported in the ELAN software (Wittenburg et al., 2006). It supports empirical research and is placed in the services category. See (Desmet et al., 2017) for a description of the system and the data analysis. The article is not included in this dissertation.

Several of the listed solutions and the projects that use these show similarities to projects from the digital humanities. It is of interest to further dive into this concept and further clarify this overlap.

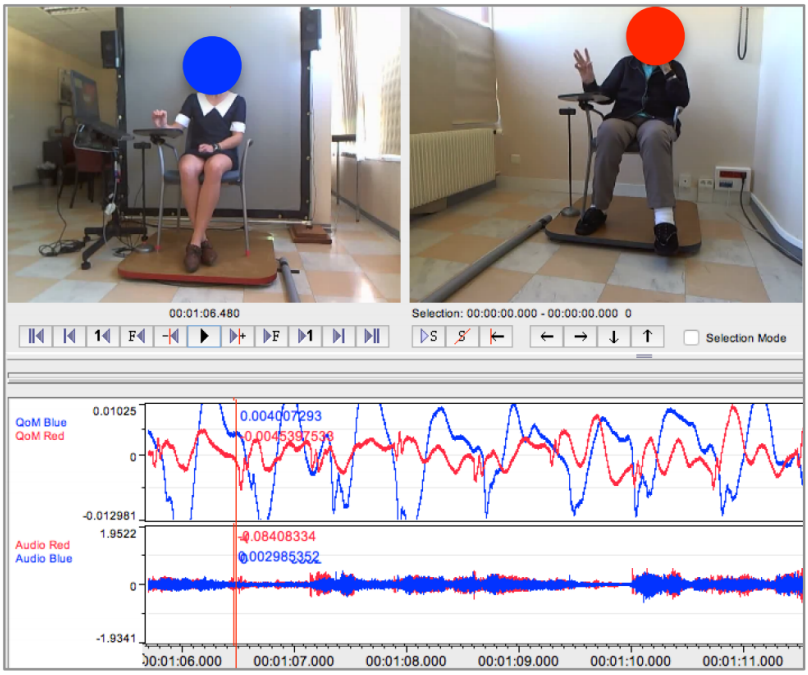



Figure 1.5: Synchronized data of a subject and experimenter visualized in ELAN by Wittenburg et al. (2006). The system registers engagement with prerecorded or live music using multiple cameras and balance boards.

### 1.1.4 Digital humanities

As a final step in my attempt to situate my work in engineering and humanities I want to briefly come back to the concepts used in digital humanities. Digital humanities is an umbrella term for research practices that combine humanities with digital technologies which show much overlap with my activities. There are a number of articles and even books that try to define the digital humanities (Berry, 2012; Gold, 2012; Schreibman et al., 2008, 2015; Terras et al., 2013). There is even disagreement whether it is a research field, "*new modes of scholarship*"(Burdick et al., 2012, p. 122) or a "*set of related methods*"(Schreibman et al., 2015, p. xvii). Warwick et al. (2012) have given up on trying to define digital humanities :*"Since the field is constantly growing and changing, specific definitions can quickly become outdated or unnecessarily limit future potential."*. However, a definition by Kirschenbaum (2012) is presented as bing broadly accepted:

> *"The digital humanities, also known as humanities computing, is a field of study, research, teaching, and invention concerned with the intersection of computing and the disciplines of the humanities. It is methodological by nature and interdisciplinary in scope. It involves investigation, analysis, synthesis and presentation of information in electronic form. It studies how these media affect the disciplines in which they are used, and what these disciplines have to contribute to our knowledge of computing."*

This definition is sufficiently broad that it would also work as a definition for systematic musicology especially in relation to this dissertation. The works bundled here are at the intersection of computing and musicology and have limited contributions to the knowledge of computing. Another relevant observation with respect to this dissertation is the following: "[Digital Humanities is inspired by]...the conviction that computational tools have the potential to transform the content, scope, methodologies, and audience of humanistic inquiry."(Burdick et al., 2012, p. 123). Again, this is similar to what is attempted in this thesis. Moreover, digital humanities projects generally have these common traits which shares much with this dissertation project:

- The projects are **collaborative**. Often engineers and humanities scholars collaborate with a shared aim.
- The nature of the methods requires a **transdisciplinary** approach. Often computer science is combined with deep insights into humanities subjects.

- The main research object is available in the **digital** domain. This immediately imposes a very stringent limitation. The research object and relations between research objects need a practical digital representation to allow successful analysis.

The history of the term can be found in the literary sciences and grew out of 'humanities computing' or 'computing in the humanities' (Berry, 2012). Where originally it was seen as *"a technical support to the work of the 'real' humanities scholars"* (Berry, 2012). The term 'digital humanities' was coined to mark a paradigm shift from merely technical support of humanities research to a "*genuinely intellectual endeavor with its own professional practices, rigorous standards, and exciting theoretical explorations*" (Hayles, 2012). Today most digital humanities scholars are firmly embedded in either library sciences, history or literary science. Due to this history there are only a few explicit links with musicology although the described methods in the digital humanities are sufficiently broad and inclusive.

One of the problems tackled by the digital humanities is the abundance of available digital data. The problem presents itself for example for historians: "*It is now quite clear that historians will have to grapple with* ***abundance, not scarcity****. Several million books have been digitized by Google and the Open Content Alliance in the last two years, with millions more on the way shortly . . . nearly every day we are confronted with a new digital historical resource of almost unimaginable size.*" (Cohen et al., 2008).

As a way to deal with this abundance Moretti (2005) makes the distinction between 'close reading' and 'distant reading' of a literary text. With 'close' meaning attentive critical reading by a scholar while 'distant reading' focuses on 'fewer elements, interconnections, shapes, relations structures, forms and models'. Moretti argues there is an urgent need for distant reading due to she sheer size of the literary field and the problem that only relatively few works are subjected to 'close reading': "*. . . a field this large cannot be understood by stitching together separate bits of knowledge about individual cases [close reading], because . . . it's a collective system, that should be grasped as such, as a whole.*" (Moretti, 2005, p 3–4)

One example of this 'distant reading' approach is given by Reagan et al. (2016). In this article each word is labeled with an emotional value: positive, neutral or negative. Words like 'murder' or 'death' are deemed negative. 'friends' or 'happy' are positive while the neutral category contains words like 'door','apple' or 'lever'. With this simple model a whole corpus is examined and six prototypical story arcs are detected. Examples include the Tragedy ( a downwards fall) and Cinderella (rise - fall - rise). This approach is a good example of

how computational tools can quickly summarize features, how insights can be gathered for a whole system and for which the 'close reading' approach would not work.

To some extend, the field of musicology followed a similar path as literary science. Large amounts of scores have been encoded into digital formats. The number of digitized - or digitally recorded - music recordings easily runs in the millions. Using similar terminology as Moretti (2005), a distinction can be made between 'close listening' and 'distant listening'[10]. 'Close listening' then means an attentive manual analysis of either a score or a recorded piece of music with the purpose to reach a certain research goal. Conversely, 'distant listening' focuses on summarized features, shapes, relations, evolutions with potentially very different research goals in mind. The 'close listening' approach allows detailed analysis but faces the problem that only a very small subset of carefully selected - the exceptional - pieces can be scrutinized. This approach potentially yields a skewed view of the collective body of works. Since little regard is given to the mundane, the exceptional could be regarded as the norm. The 'distant listening' approach offers ways to straighten that skewed view while allowing for research that tracks changes over time. By allowing the analysis of the mundane next to the exceptional, a fairer view is generated on the system as a whole.

To allow this broad, fair view on whole musical collections, this 'distant listening' needs to be as intelligent as possible. The construction of models of music, the selection of features and the corresponding similarity measures need to be done diligently. This is one of the main challenges in the MIR research field.

However, this focus on databases covers only an aspect of musicologists' interest. As mentioned, there is a vast domain of research in musicology that focuses on *music interaction*. In fact, the work at IPEM is known world-wide for establishing a science of interaction (e.g. see Leman (2007, 2016); Lesaffre et al. (2017). Digital humanities are usually associated with fixed items like archives, databases, sets of literary works, historical GIS (Geographical Information System) data which, in my view, is too limited. Therefore, I introduce the term **'augmented humanities'** later on (see 4.2) in order to cover that aspect of my work in which engineering solutions work as an element in a context of human-music interaction, rather than archives and MIR. However, as my dissertation shows, it turns out that MIR techniques are very useful for solving problems in human-music interaction.

First more details are given on the reproducible methodology that is followed in the presented works of this dissertation.

[10]These concepts are elaborated upon in the book by Cook (2013). Chapter five, titled *'close and distant listening'*, is especially relevant.

## 1.2 Methodology

In the works bundled in this dissertation, special efforts have been made to reach a reproducible, verifiable methodology. Reproducibility is one of the corner-stones of scientific methodology. A claim made in a scientific publication should be verifiable and the described method should provide enough detail to allow replication. If the research is carried out on a specific set of data, this data should be available as well. If not for the general public, then at least to peers or - even more limiting - to reviewers of the work. If those basics are upheld, then the work becomes verifiable, reproducible and comparable. It also facilitates improvements by other researchers.

In a journal article (Six et al., 2018a) on this topic I have bundled the main concerns with, and suggestions for, methodological computational research on music. This journal article details the problems with reproducibility in computational research and illustrates this by replicating, in full, a seminal acoustic fingerprinting paper. The main points of that article are repeated here with additional links to the works presented in this dissertation. The aim of this chapter is to strike a balance between providing common methodological background while avoiding too much repeated text.

The ideal, where methods are described in sufficient detail and data is available, is often not reached. From a technical standpoint, sharing tools and data has never been more easy. Reproducibility, however, remains a problem. Especially for Music Information Retrieval research and, more generally, research involving moderately complex software systems. Below, a number of general problems are identified and subsequently discussed how these are handled in the presented works.

### 1.2.1 Intellectual property rights and sharing research code

Journal articles and especially conference papers have a limited space for detailed descriptions of methods or algorithms. For moderately complex systems there are numerous parameters and edge cases which are glossed over in textual descriptions. This makes articles readable and the basic method intelligible, but those details need to be expounded somewhere otherwise too much is left to assumptions and perhaps missing shared understanding. The ideal place for such information is well documented, runnable code. Unfortunately, *intellectual property rights* by universities or research institutions often limit researchers to freely distribute code.

In the works presented in this dissertation attention has been given

to sharing research code. As Ghent University is more and more striving for open-access publications and even working on a policy and support for sharing research data. Until today there is, however, little attention for research software. Arguably, it makes little sense to publish only part of research in the open - the textual description - and keeping code and data behind closed doors. Especially if the research is funded by public funds. A clear stance on **intellectual property rights** of research code would help researchers.

A possibility to counter this problem is to use Open Source licenses. These licenses "allow software to be freely used, shared and modified"[11]. In my own works I have benefited a lot from Open Source code. This code allowed me to focus on the task at hand and not spending too much time on basic house-keeping code. Examples include statistical libraries, command line argument parsing code, FFT libraries and so forth. During the development of Tarsos and TarsosDSP (Six et al., 2013, 2014) a lot of code released under the GPL (General Public License) license was re-purposed or translated from other projects to benefit Tarsos. A total of 18 sources are listed in the readme of Tarsos and TarsosDSP. The GPL license requires that modifications are made public under the same conditions as well. This viral aspect of GPL has the advantage that it becomes impossible to keep code behind closed doors. The GPL requires an open methodology which fits well with research software and prototypes. If code is released in a reasonable way, adhering to the sustainable software philosophy (Crouch et al., 2013), the processes or models encoded in the software become verifiable, transparent and ready for improvement or external contributions.

On GitHub, a code repository hosting service, TarsosDSP has 14 contributers and has been forked more than 200 times. This means that 14 people contributed to the main code of TarsosDSP and that there are about *200 different flavors* of TarsosDSP. These flavors are at some point split - forked - from the main code and are developed by people with a slightly different focus. On GitHub more than 100 bug reports are submitted. This example highlights another benefit from opening source code. The code is tested by others, bugs are reported and some even contribute fixes and improvements.

The fact that GPL was used also made it possible to further improve the Tarsos software myself after my transition from the School of Arts, Ghent to Ghent University. A process that could have been difficult if the code was not required to be released under GPL. Panako and SyncSink (Six and Leman, 2014, 2015) also build upon GPL licensed software and are therefore released under a similar license.

[11] See http://opensource.org/licenses for an overview of differences between several open source licenses.

### 1.2.2 Copyrights on music and sharing research data

To make computational research reproducible both the source code and the data that was used need to be available. **Copyrights on music** make it hard to share music freely. Redistribution of historic field-recordings in museum archives is even more problematic. Due to the nature of the recordings, the copyright status is often unclear. Clearing the status of tracks involves international, historical copyright laws and multiple stakeholders such as performers, soloists, the museum, the person who performed the field recording and potentially a publisher that already published parts on an LP. The rights of each stakeholder need to be carefully considered while at the same time they can be hard to identify due to a lack of precise meta-data and the passage of time. I see two ways to deal with this:

1. *Pragmatic versus Ecological* or Jamendo *vs* iTunes. There is a great deal of freely available music published under various creative commons licenses. Jamendo for example contains half a million creative commons-licensed[12] tracks which are uniquely identifiable and can be downloaded via an API. Much of the music that can be found there is recorded at home with limited means. It only contains a few professionally produced recordings. This means that systems can behave slightly differently on the Jamendo set when compared with a set of commercial music. What is gained in pragmatism is perhaps lost in ecological validity. Whether this is a problem depends very much on the research question at hand.

2. *Audio versus Features.* Research on features extracted from audio does not need audio itself, if the features are available this can suffice. There are two large sets of audio features: the million song data set by Bertin-Mahieux et al. (2011) and Acousticbrainz[13], described by Porter et al. (2015). Both ran feature extractors on millions of commercial tracks and have an API to query or download the data. Unfortunately the source code of the feature extractors used in the Million Song data set is not available. Additionally, the features are only described until a certain level of detail. This makes the dataset a black box and, in my eyes, unfit for decent reproducible science. Indeed, due to internal reorganizations and mergers the API and the data become less and less

[12] Creative commons (https://creativecommons.org) provides a set of licenses to distribute works under clear conditions.

[13] Acousticbrainz can be found at https://acousticbrainz.org and is an initiative by the Music Technology Group, UPF Universitat Pompeu Fabra, Barcelona

available. The science build on this set is on shaky ground. Fortunately Acousticbrainz is completely transparent. It uses well documented, open source software (Bogdanov et al., 2013) and the feature extractors are reproducible. The main shortcoming of this approach is that only a curated set of features is available and if another feature is needed, then you are out of luck. Adding a feature is far from trivial, since even Acousticbrainz has no access to all audio: they rely on crowd-sourced feature extraction.

In my own work an acoustic fingerprinting evaluation methodology was developed using music from Jamendo for Panako (Six and Leman, 2014). The exact same methodology was copied by Sonnleitner and Widmer (2016) and elaborated on by myself (Six et al., 2018a) with a focus on methodological aspects. For acoustic fingerprinting the Jamendo dataset fits since it does offer a large variability in genres and is representative in those cases.

For the research in collaboration with the Museum for Central Africa (Cornelis et al., 2013; Six et al., 2013, 2014) the issue of unclear copyrights and ethical questions on sharing field recordings is pertinent. Indeed, the copyright status of most recordings is unclear. Ethical questions on the conditions in which the recordings were made and to what extent the recorded musicians gave informed consent for further use can be raised. The way this research was done follows the second guideline. The research is done on extracted features and only partially on audio. These features are not burdened by copyright issues and can be shared freely. More specifically, a large set of features were extracted by placing a computer at the location of the museum and processing each field recording. This method enables the research to be replicated – starting from the features – and verified.

### 1.2.3 Publication culture and providing incentives

Output by researchers is still mainly judged by the number of articles they publish in scientific journals or conferences. Other types of output are not valued as much. The **incentive** to put a lot of work in documenting, maintaining and publishing reproducible research or supplementary material is lacking. This focus on publishing preferably novel findings in journal articles probably affects the research conducted. It drives individual researchers - consciously or unconsciously - to further their careers by publishing underpowered small studies in stead of furthering the knowledge in their fields of research (Higginson and Munafò, 2016).

A way forward is to provide an **incentive** for researchers to make

their research reproducible. This requires a mentality shift. Policies by journals, conference organizers and research institutions should gradually change to require reproducibility. There are a few initiatives to foster reproducible research, specifically for music informatics research. The 53rd Audio Engineering Society (AES) conference had a price for reproducibility. Six et al. (2014) was submitted to that conference and subsequently acknowledged as reproducibility-enabling. ISMIR 2012 had a tutorial on *"Reusable software and reproducibility in music informatics research"* but structural attention for this issue at ISMIR seems to lack. As one of the few places Queen Mary University London (QMUL) seems to have continuous attention to the issue and researchers are trained in software craftsmanship. They also host a repository for software dealing with sound at `http://soundsoftware.ac.uk.` and offer a yearly workshop on "Software and Data for Audio and Music Research":

> The third SoundSoftware.ac.uk one-day workshop on "Software and Data for Audio and Music Research" will include talks on issues such as robust software development for audio and music research, **reproducible research in general**, management of research data, and open access.[14]

Another incentive to spend time documenting and publishing research software is already mentioned above: code is reviewed by others, bugs are submitted and some even take the time to contribute to the code by fixing bugs or extending functionality. A more indirect incentive is that it forces a different approach to writing code. Quick and dirty hacks are far less appealing if one knows beforehand that the code will be out in the open and will be reviewed by peers. Publishing code benefits the reuseablity, modularity, clarity and longevity and software quality in general. It also forces one to think about installability, buildability and other aspects that make software sustainable (Crouch et al., 2013; Jackson et al., 2011).

In my work the main incentive to publish code is to make the tools and techniques available and attempt to put these in the hands of end-users. While the aim is not to develop end-user software, establishing a feedback loop with users can be inspiring and even drive further research. In an article I co-authored with a number of colleagues, a possible approach is presented to fill the gap between research software and end-users (de Valk et al., 2017). One of the hurdles is to make users - in this case managers of archives of ethnic music - aware of the available tools and techniques. In two other articles I (co-)authored (Bressan et al., 2017b; Six et al., 2018b) exactly this is done: it details

[14] `http://soundsoftware.ac.uk/soundsoftware2014` - March 2017

how archives can benefit from a mature MIR technology - in this case acoustic fingerprinting.

### 1.2.4 Research software versus end-user software

The distinction between end-user ready software, a (commercial) product, and useful contributions to a field in the form of research software may not be entirely clear. One of the outputs of computational research is often research software. This software should focus on novel methods encoded in software and not on creating end-user software. Perhaps it is of interest to stress the differences. Below an attempt follows to make this distinction by focusing on several aspects of software.

- **Transparency**. The processes encoded in end-user software are not necessarily transparent. End-user software can be used effectively as a *black box*. The outcome - what you can achieve with the software - is more important than how it is done. Research software should focus exactly on making the *process transparent* while getting tasks done.

- **Complexity**. The complexity of research software should not be hidden, it should be clear which parameters there are and how parameters change results. Researchers can be expected to put effort in getting to know details of software they use. This is again different in end-user software where ease-of-use and intuitiveness matter.

- **Openness**. Researchers should be able to improve, adapt and experiment with the software. It stands to reason that research software should be open and allow, even encourage, improvement. Source control and project management websites such as GitHub and SoundSoftware.ac.uk facilitate these kinds of interactions. For end-user software this may not be a requirement.

Note that these characteristics of research software do not necessarily prevent such software from being applied as-is in practice. The research software Panako (Six and Leman, 2014), which serves as an experimental platform for acoustic fingerprinting algorithms, is being used by Musimap and the International Federation of the Phonographic Industry (IFPI). Tarsos (Six et al., 2013) has an attractive easy-to-use graphical interface and is being used in workshops and during lectures by students. SyncSink (Six and Leman, 2015) exposes many parameters to tweak but can and is effectively used by researchers to synchronize research data. So some research software can be used as-is.

Conversely, there is also transparent and open end-user software available that does not hide its complexity such as the relational database system PostgreSQL. This means that the characteristics (transparency, complexity, openness) are not exclusive to research software but they are, in my view, requirements for good research software. The focus should be on demonstration of a process while getting (academic) tasks done and not on simply getting tasks done. This can be found, for example, in the 'mission statement' of TarsosDSP:

> TarsosDSP is a Java library for audio processing. Its aim is to provide an easy-to-use interface to practical music processing algorithms implemented as simply as possible ... The library tries to hit the sweet spot between being capable enough to get real tasks done but compact and simple enough to serve as a demonstration on how DSP algorithms work.

This distinction between 'real' and 'academic' tasks is of interest. In 'real' tasks practical considerations with regard to computational load, scalability and context need to be taken into account. This is much less the case for 'academic' tasks: there the process is the most important contribution, whereas performance and scalability may be an afterthought. For example, research software which encodes a multi-pitch estimation algorithm able to score much better than the current state of the art software. If this algorithm has an enormous computational load and it takes many hours to process only a couple of seconds of music, this still is a valid contribution: it shows how accurate multi pitch estimation can be. However, it is completely unpractical to use if thousands of songs need to be processed. The system serves an academic purpose but can not be used for 'real' tasks. A fingerprinting system that has desirable features but can only handle a couple of hundred reference items is another example. The previously mentioned publication by de Valk et al. (2017) which I co-authored deals with this topic and gives several examples of research software capable enough to be used effectively by archivists to manage digital music archives.

To summarize: in my work research software packages form a considerable type of output. These systems serve an academic purpose in the first place but if they can be used for 'real' tasks then this is seen as an added benefit. I have identified common problems with reproducibility in computational research and have strived to make my own contributions reproducible by publishing source code and evaluating systems with publicly available data sets as much as possible. This makes my contributions verifiable and transparent but also allows others to use, criticize and improve these systems.

## 1.3 Summary

To contextualize my research I have given a brief overview of the interdisciplinary nature of the systematic musicology research field. The main point is that advanced research practices almost always involve challenging technological problems not easily handled by researchers with a background in humanities. The problems concern dealing with various types of digital data, computational models and complex experimental designs. I have set myself the task to *engineer solutions that are relevant for systematic musicology.* I have created such solutions and these are placed in a plane. One axis of that plane goes from methods to services. The other axis contrasts MIR-technologies with technologies for empirical research. These concepts are delineated and contextualized. A total of thirteen solutions explore this plane and are briefly discussed. The solutions have many attributes also found in *digital humanities* research projects. The link with the digital humanities was made explicit as well.

The solutions presented in my work aims to follow a *reproducible methodology.* Problems with reproducible computational research were identified and the importance of reproducibility was stressed. It was explained how my own research strifes for this ideal: source code that implements solutions is open sourced and solutions are evaluated with publicly available data as much as possible. Finally a distinction was clarified between research prototypes and end-user ready software.

The following two chapters contain several publications that have been published elsewhere. They are self-contained works which means that some repetition might be present. The next chapter deals with publications that describe methods. The chapter that bundles publications describing services follows. For each chapter an additional introduction is included.

# 2 Methods

## 2.1 Introduction

This chapter bundles three articles that are placed in the methods category of the humanities-engineering plane depicted in Figure 1.1. The focus of the papers is to present and apply methods which can yield new insights:

- 2.2 – Six, J., Cornelis, O., and Leman, M. (2013). Tarsos, a modular platform for precise pitch analysis of Western and non-Western music. *Journal of New Music Research*, 42(2):113–129.
- 2.4 – Six, J., Bressan, F., and Leman, M. (In press – 2018). Applications of duplicate detection in music archives: From metadata comparison to storage optimisation - The case of the Belgian Royal Museum for Central Africa. In *Proceedings of the 13th Italian Research Conference on Digital Libraries (IRCDL 2018)*
- 2.3 – Six, J., Bressan, F., and Leman, M. (In press – 2018b). A case for reproduciblity in MIR. Replication of 'a highly robust audio fingerprinting system'. *Transactions of the International Society for Music Information Retrieval (TISMIR).*

The first paper (Six et al., 2013) describes Tarsos. It details a method to for the extraction, comparison and analysis of pitch class histograms on a large scale. The method is encoded in a software system called Tarsos. Tarsos features a graphical user interface to analyze a single recording quickly and an API to allow analysis of many recordings. The final parts of the paper give an example of extraction and matching scales for hundreds of recordings of the Makam tradition effectively contrasting theoretical models with (historical) performance practice. This serves as an illustration how models can be contrasted with practice on a large scale.

The second paper (Six et al., 2018b) presents a method to find duplicates in large music archives. It shows how duplicate detection technology can be employed to estimate the quality of meta-data and to contrast meta-data of an original with a duplicate. As a case study, the method is applied to the data set of the RMCA .

Reproducibility is the main topic of the third work (Six et al., 2018a). It describes the problems with reproducibility in computational research and MIR research. These problems are illustrated by

replicating a seminal acoustic fingerprinting paper. While the results of the replication come close to the originally published results and the main findings are solidified, there is a problematic unexplained discrepancy, an unknown unknown. The main contribution of the paper lays in the method which describes how new insights in MIR can be described in a sustainable manner.

The article by Cornelis et al. (2013) describes a method to automatically estimate the rhythmic complexity of a piece of music by using a set of beat tracking algorithms. The method is validated by comparing the results of the set of algorithms with human expert annotations. I co-authored the article and it could have been bundled here as well but I chose to limit bundled works to articles for which I serve as the main author.

## 2.2 Tarsos, a modular platform for precise pitch analysis of western and non-western music

Six, J., Cornelis, O., and Leman, M. (2013). Tarsos, a modular platform for precise pitch analysis of Western and non-Western music. *Journal of New Music Research*, 42(2):113–129.

### Abstract

*This paper presents Tarsos, a modular software platform used to extract and analyze pitch organization in music. With Tarsos pitch estimations are generated from an audio signal and those estimations are processed in order to form musicologically meaningful representations. Tarsos aims to offer a flexible system for pitch analysis through the combination of an interactive user interface, several pitch estimation algorithms, filtering options, immediate auditory feedback and data output modalities for every step.*

*To study the most frequently used pitches, a fine-grained histogram that allows up to 1200 values per octave is constructed. This allows Tarsos to analyze deviations in Western music, or to analyze specific tone scales that differ from the 12 tone equal temperament, common in many non-Western musics.*

*Tarsos has a graphical user interface or can be launched using an* API *- as a batch script. Therefore, it is fit for both the analysis of individual songs and the analysis of large music corpora. The interface allows several visual representations, and can indicate the scale of the piece under analysis. The extracted scale can be used immediately to tune a* MIDI *keyboard that can be played in the discovered scale.*

*These features make Tarsos an interesting tool that can be used for musicological analysis, teaching and even artistic productions.*

### 2.2.1 Introduction

In the past decennium, several algorithms became available for extracting pitch from audio recordings (Clarisse et al., 2002; de Cheveigné and Hideki, 2002; Klapuri, 2003). Pitch extraction tools are prominently used in a wide range of studies that deal with analysis, perception and retrieval of music. However, up to recently, less attention has been paid to tools that deal with distributions of pitch in music.

The present paper presents a tool, called Tarsos, that integrates existing pitch extraction tools in a platform that allows the analysis of pitch distributions. Such pitch distributions contain a lot of information, and can be linked to tunings, scales, and other properties of musical performance. The tuning is typically reflected in the distance between pitch classes. Properties of musical performance may relate to pitch drift within a single piece, or to influence of enculturation (as it is the case in African music culture, see Moelants et al. (2009)). A major feature of Tarsos is concerned with processing audio-extracted pitches into pitch and pitch class distributions from which further properties can be derived.

Tarsos provides a modular platform used for pitch analysis - based on pitch extraction from audio and pitch distribution analysis - with a flexibility that includes:

- The possibility to focus on a part of a song by selecting graphically displayed pitch estimations in the melograph.
- A zoom function that allows focusing on global or detailed properties of the pitch distribution.
- Real-time auditory feedback. A tuned MIDI synthesizer can be used to hear pitch intervals.
- Several filtering options to get clearer pitch distributions or a more discretized melograph, which helps during transcription.

In addition, a change in one of the user interface elements is immediately propagated through the whole processing chain, so that pitch analysis becomes easy, adjustable and verifiable.

This paper is structured as follows. First, we present a general overview of the different processing stages of Tarsos, beginning with the low level audio signal stage and ending with pitch distributions and their musicological meaning. In the next part, we focus on some case studies and give a scripting example. The next part elaborates on the musical aspects of Tarsos and refers to future work. The fifth and final part of the main text contains a conclusion.

### The Tarsos platform

Figure 2.1 shows the general flow of information within Tarsos. It starts with an audio file as input. The selection of a pitch estimation algorithm leads to a pitch estmations, which can be represented in different ways. This representation can be further optimized, using different types of filters for peak selection. Finally, it is possible to produce an audio output of the obtained results. Based on that output, the analysis-representation-optimization cycle can be refined. All steps contain data that can be exported in different formats. The obtained pitch distribution and scale itself can be saved as a scala file which in turn can be used as input, overlaying the estimation of another audio file for comparison.

In what follows, we go deeper into the several processing aspects, dependencies, and particularities. In this section we first discuss how to extract pitch estimations from audio. We illustrate how these pitch estimations are visualized within Tarsos. The graphical user interface is discussed. The real-time and output capabilities are described, and this section ends with an explanation about scripting for the Tarsos API. As a reminder: there is a manual available for Tarsos at: `http://0110.be/tag/JNMR`.

### Extracting pitch estimations from audio

Prior to the step of pitch estimation, one should take into consideration that in certain cases, audio preprocessing can improve the subsequent analysis within Tarsos. Depending on the source material and on the research question, preprocessing steps could include noise reduction, band-pass filtering, or harmonic/percussive separation (Nobutaka et al., 2010). Audio preprocessing should be done outside of the Tarsos tool. The, optionally preprocessed, audio is then fed into Tarsos and converted to a standardized format[1].

The next step is to generate pitch estimations. Each selected block of audio file is examined and pitches are extracted from it. In figure 2.2, this step is located between the input and the signal block phases. Tarsos can be used with external and internal pitch estimators. Currently, there is support for the polyphonic MAMI pitch estimator (Clarisse et al., 2002) and any VAMP plug-in (Cannam et al., 2010) that generates pitch estimations. The external pitch estimators are platform dependent and some configuration needs to be done to get them working. For practical purposes, platform independent implementations of two

[1] The conversion is done using FFMPEG, a cross platform command line tool to convert multimedia files between formats. The default format is PCM WAV with 16 bits per sample, signed, little endian, 44.1kHz. Furthermore, all channels are down mixed to mono.

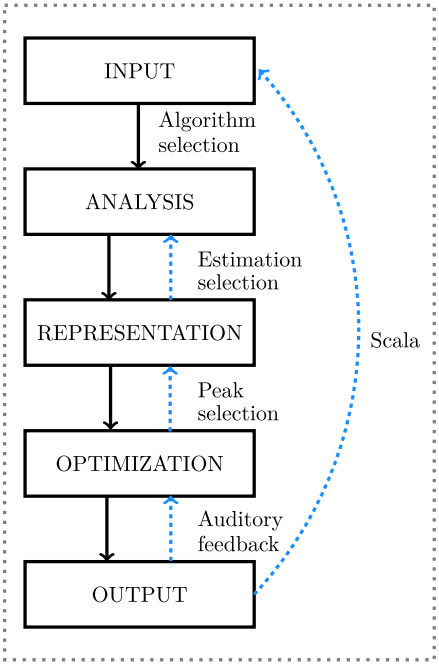



Figure 2.1: The main flow of information within Tarsos.

pitch detection algorithms are included, namely, YIN (de Cheveigné and Hideki, 2002) and MPM (McLeod and Wyvill, 2005). They are available without any configuration. Thanks to a modular design, internal and external pitch detectors can be easily added. Once correctly configured, the use of these pitch modules is completely transparent, as extracted pitch estimations are transformed to a unified format, cached, and then used for further analysis at the symbolic level.

### Visualizations of pitch estimations

Once the pitch detection has been performed, pitch estimations are available for further study. Several types of visualizations can be created, which lead, step by step, from pitch estimations to pitch distribution and scale representation. In all these graphs the *cent* unit is used. The cent divides each octave into 1200 equal parts. In order to use the cent unit for determining absolute pitch, a reference frequency of 8.176Hz has been defined[2], which means that 8.176Hz equals 0 cents, 16.352Hz equals 1200 cents and so on.

A first type of visualization is the melograph representation, which is shown in Figure 2.3. In this representation, each estimated pitch is plotted over time. As can be observed, the pitches are not uniformly distributed over the pitch space, and form a clustering around 5883 cents.

A second type of visualization is the pitch histogram, which shows the pitch distribution regardless of time. The pitch histogram is constructed by assigning each pitch estimation in time to a bin between 0 and 14400[3] cents, spanning 12 octaves. As shown in Figure 2.4, the peak at 5883 cents is now clearly visible. The height of a peak represents the total number of times a particular pitch is estimated in the selected audio. The pitch range is the difference between the highest and lowest pitch. The graph further reveals that some peaks appear every 1200 cents, or every octave.

A third type of visualization is the pitch *class* histogram, which is obtained by adding each bin from the pitch histogram to a corresponding modulo 1200 bin. Such a histogram reduces the pitch distribution to one single octave. A peak thus represents the total duration of a pitch *class* in a selected block of audio. Notice that the peak at 5883 cents in the pitch histogram (Figure 2.4) now corresponds to the peak at 1083 cents in the pitch class histogram (Figure 2.6).

It can also be useful to select only filter pitch estimations that make up the pitch class histogram. The most obvious 'filter' is to select

[2]See Appendix 2.2.5 for a discussion about pitch representation in cents and the seemingly arbitrary reference frequency of 8.176Hz.

[3]14400 absolute cents is equal to 33488Hz, well above human hearing.

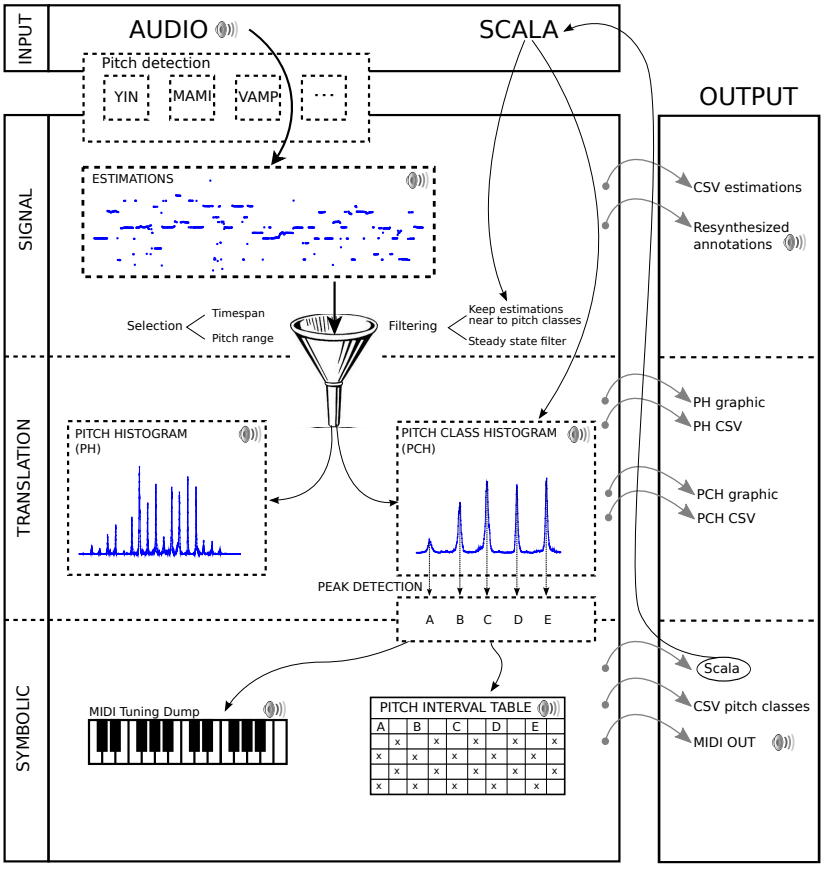



Figure 2.2: Detailed block diagram representing all components of Tarsos, from input to output, from signal level to symbolic level. All additional features (selection, filtering, listening) are visualized (where they come into play). Each step is described into more detail in Chapter 3.

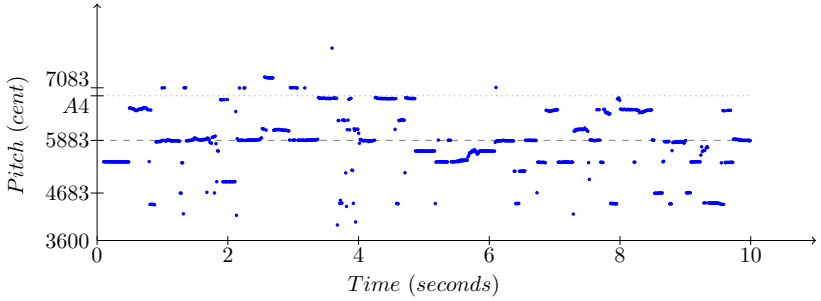



Figure 2.3: A melograph representation. Estimations of the first ten seconds of an Indonesian Slendro piece are shown. It is clear that pitch information is horizontally clustered, e.g. the cluster around 5883 cents, indicated by the dashed horizontal line. For reference a dotted horizontal line with $A4$, 440Hz is also present.

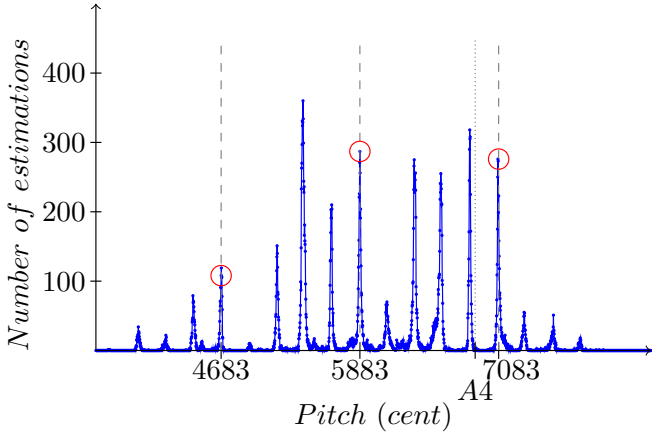



Figure 2.4: A pitch histogram with an Indonesian Slendro scale. The circles mark the most estimated pitch classes. The dashed vertical lines show the same pitch class in different octaves. A dotted vertical line with $A4$, 440Hz, is used as a reference for the diapason.

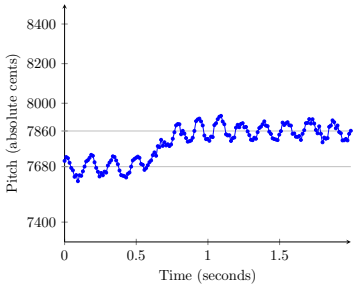
only an interesting timespan and pitch range. The distributions can be further manipulated using other filters and peak detection. The following three filters are implemented in Tarsos:

The first is an *estimation quality* filter. It simply removes pitch estimations from the distribution below a certain quality threshold. Using YIN, the quality of an estimation is related to the periodicity of the block of sound analyzed. Keeping only high quality estimations should yield clearer pitch distributions.

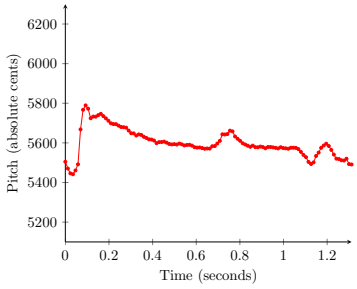
The second is called a *near to pitch class filter*. This filter only allows pitch estimations which are close to previously identified pitch classes. The pitch range parameter (in cents) defines how much ornamentations can deviate from the pitch classes. Depending on the music and the research question, one needs to be careful with this - and other - filters. For example, a vibrato makes pitch go up and down - pitch modulation - and is centered around a pitch class. Figure 2.5(a) gives an example of Western vibrato singing. The melograph reveals the ornamental singing style, based on two distinct pitch classes. The two pitch classes are hard to identify with the histogram 2.5(c) but are perceptually there, they are made clear with the dotted gray line. In contrast, figure 2.5(b) depicts a more continuous glissando which is used as a building block to construct a melody in an Indian raga. For these cases, Krishnaswamy (2004b) introduced the concept of two-dimensional 'melodic atoms'. (In Henbing and Leman (2007) it is shown how elementary bodily gestures are related to pitch and pitch gestures.) The histogram of the pitch gesture Figure 2.5(d) suggests one pitch class while a fundamentally different concept of tone is used. Applying the near to pitch class filter on this type of music could result into incorrect results. The goal of this filter is to get a clearer view on the melodic contour by removing pitches between pitch classes, and to get a clearer pitch class histogram.

The third filter is a *steady state filter*. The steady state filter has a time and pitch range parameter. The filter keeps only consecutive estimations that stay within a pitch range for a defined number of milliseconds. The default values are 100ms within a range of 15 cents. The idea behind it is that only 'notes' are kept and transition errors, octave errors and other short events are removed.

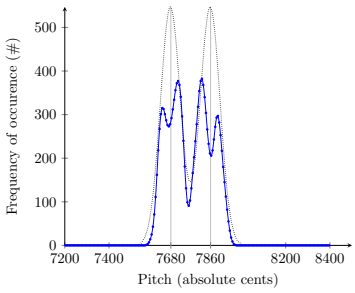
Once a selection of the estimations are made or, optionally, other filters are used, the distribution is ready for peak detection. The peak detection algorithm looks for each position where the derivative of the histogram is zero, and a local height score is calculated with the formula in (2.1). The local height score $s_w$ is defined for a certain window $w$, $\mu_w$ is the average height in the window, $\sigma_w$ refers to the standard deviation of the height in the window. The peaks are ordered by their score and iterated, starting from the peak with the highest score. If peaks are



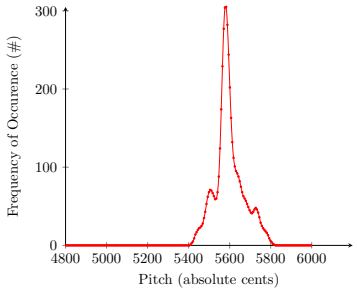

(a) Western vibrato melograph.




(b) Indian raga pitch gesture.




(c) Western vibrato pitch histogram.




(d) Indian raga pitch gesture histogram.

Figure 2.5: Visualization of pitch contours of Western and Indian singing; notice the fundamentally different concept of tone. In the western example two distinct pitches are used, they are made clear with the dotted gray lines. In Figure 2.5(c) two dotted gray curves are added, they represent the two perceived pitches.

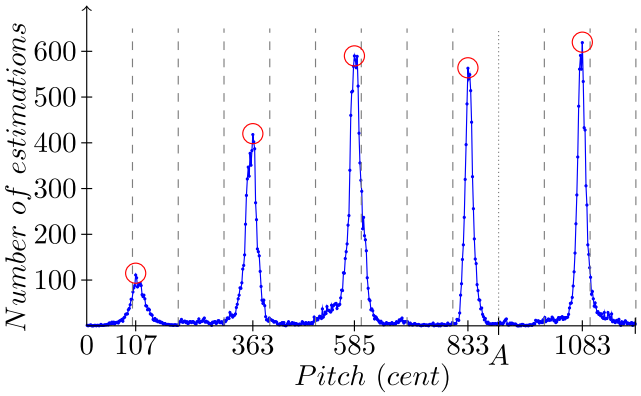



Figure 2.6: A pitch class histogram with an Indonesian Slendro scale. The circles mark different pitch classes. For reference, the dashed lines represent the Western equal temperament. The pitch class $A$ is marked with a dotted line.

found within the window of the current peak, they are removed. Peaks with a local height score lower than a defined threshold are ignored. Since we are looking for pitch classes, the window $w$ wraps around the edges: there is a difference of 20 cent between 1190 cent and 10 cent.

$$s_w = \frac{height - \mu_w}{\sigma_w} \tag{2.1}$$

Figure 2.7 shows the local height score function applied to the pitch class histogram shown in Figure 2.6. The desired leveling effect of the local height score is clear, as the small peak at 107 cents becomes much more defined. The threshold is also shown. In this case, it eliminates the noise at around 250 cents. The noise is caused by the small window size and local height deviations, but it is ignored by setting threshold $t$. The performance of the peak detection depends on two parameters, namely, the window size and the threshold. Automatic analysis either uses a general preset for the parameters or tries to find the most stable setting with an exhaustive search. Optionally gaussian smoothing can be applied to the pitch class histogram, which makes peak detection more straightforward. Manual intervention is sometimes needed, by fiddling with the two parameters a user can quickly browse through several peak detection result candidates.

Once the pitch classes are identified, a pitch class interval matrix can be constructed. This is the fourth type of representation, which is shown in Table 2.1. The pitch class interval matrix represents the

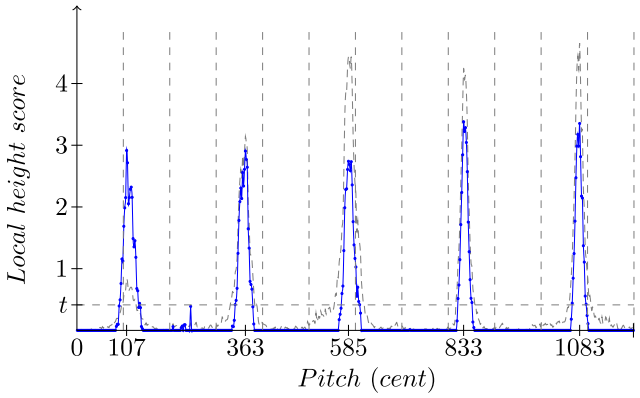



Figure 2.7: A local height score function used to detect peaks in a pich class histogram. Comparing the original histogram of figure 2.6 with the local height score shows the leveling effect of the local height score function. The dashed vertical lines represents the Western equal temperament, the dashed horizontal line the threshold $t$.

| P.C. | 107 | 364 | 585 | 833 | 1083 |
|---|---|---|---|---|---|
| 107 | 0 | 256 | 478 | 726 | 976 |
| 364 | 944 | 0 | 221 | 470 | 719 |
| 585 | 722 | 979 | 0 | 248 | **498** |
| 833 | 474 | 730 | 952 | 0 | 250 |
| 1083 | 224 | 481 | **702** | 950 | 0 |

Table 2.1: Pitch classes (P.C.) and pitch class intervals, both in cents. The same pentatonic Indonesian Slendro is used as in figure 2.6. A prefect fifth and its dual, a perfect fourth, are marked by a bold font.

found pitch classes, and shows the intervals between the pitch classes. In our example, a perfect fourth[4], a frequency ratio of 4/3 or 498 cent, is present between pitch class 585 and 1083. This means that a perfect fifth, a frequency ratio of $\frac{2/1}{4/3} = 3/2$ or $1200 - 498 = 702$ cent, is also present[5].

[4]The perfect fourth and other musical intervals are here used in their physical meaning. The physical perfect fourth is sometimes called just fourth, or perfect fourth in just intonation.

[5]See Appendix 2.2.5 to see how ratios translate to cent values.

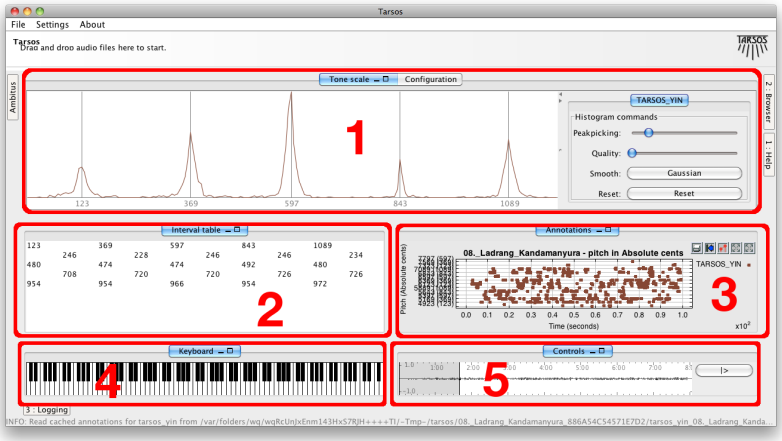



Figure 2.8: A screenshot of Tarsos with 1 a pitch class histogram, 2 a pitch class interval table, 3 a melograph with pitch estimations, 4 a MIDI keyboard and 5 a waveform.

### The interface

Most of the capabilities of Tarsos are used through the graphical user interface (Figure 2.8). The interface provides a way to explore pitch organization within a musical piece. However, the main flow of the process, as described above, is not always as straightforward as the example might suggest. More particularly, in many cases of music from oral traditions, the peaks in the pitch class histogram are not always well-defined (see Section 2.2.3). Therefore, the automated peak detection may need manual inspection and further manual fine-tuning in order to correctly identify a songs' pitch organization. The user interface was designed specifically for having a flexible environment where all windows with representations communicate their data. Tarsos has the attractive feature that all actions, like the filtering actions mentioned in Section 2.2.1, are updated for each window in real-time.

One way to closely inspect pitch distributions is to select only a part of the estimations. In the block diagram of Figure 2.2, this is represented by the funnel. Selection in time is possible using the waveform view (Figure 2.8-5). For example, the aim could be a comparison of pitch distributions at the beginning and the end of a piece, to reveal whether a choir lowered or raised its pitch during a performance (see Section 2.2.3 for a more elaborate example).

Selection in pitch range is possible and can be combined with a selection in time using the melograph (Figure 2.8-3). One may se-

lect the melodic range such as to exclude pitched percussion, and this could yield a completely different pitch class histogram. This feature is practical, for example when a flute melody is accompanied with a low-pitched drum and when you are only interested in flute tuning. With the melograph it is also possible to zoom in on one or two notes, which is interesting for studying pitch contours. As mentioned earlier, not all music is organized by fixed pitch classes. An example of such pitch organization is given in Figure 2.5(b), a fragment of Indian music where the estimations contain information that cannot be reduced to fixed pitch classes.

To allow efficient selection of estimations in the time and frequency, they are stored in a kd-tree (Bentley, 1975). Once such a selection of estimations is made, a new pitch histogram is constructed and the pitch class histogram view (Figure 2.8-1) changes instantly.

Once a pitch class histogram is obtained, peak detection is a logical next step. With the user interface, manual adjustment of the automatically identified peaks is possible. New peak locations can be added and existing ones can be moved or deleted. In order to verify the pitch classes manually, it is possible to click anywhere on the pitch class histogram. This sends a MIDI-message with a pitch bend to synthesize a sound with a pitch that corresponds to the clicked location. Changes made to the peak locations propagate instantly throughout the interface.

The pitch class interval matrix (Figure 2.8-2) shows all new pitch class intervals. Reference pitches are added to the melograph and MIDI tuning messages are sent (see Section 2.2.1). The pitch class interval matrix is also interactive. When an interval is clicked on, the two pitch classes that create the interval sound at the same time. The dynamics of the process and the combination of both visual and auditory clues makes manually adjusted, precise peak extraction, and therefore tone scale detection, possible. Finally, the graphical display of a piano keyboard in Tarsos allows us to play in the (new) scale. This feature can be executed on a computer keyboard as well, where notes are projected on keys. Any of the standard MIDI instruments sounds can be chosen.

It is possible to shift the pitch class histogram up- or downwards. The data is then viewed as a repetitive, octave based, circular representation. In order to compare scales, it is possible to upload a previously detected scale (see Section 2.2.1) and shift it, to find a particular fit. This can be done by hand, exploring all possibilities of overlaying intervals, or the best fit can be suggested by Tarsos.

### Real-time capabilities

Tarsos is capable of real-time pitch analysis. Sound from a microphone can be analyzed and immediate feedback can be given on the played or sung pitch. This feature offers some interesting new use-cases in education, composition, and ethnomusicology.

For educational purposes, Tarsos can be used to practice singing quarter tones. Not only the real time audio is analyzed, but also an uploaded scale or previously analyzed file can be listened to by clicking on the interval table or by using the keyboard. Singers or string players could use this feature to improve their intonation regardless of the scale they try to reach.

For compositional purposes, Tarsos can be used to experiment with microtonality. The peak detection and manual adjustment of pitch histograms allows the construction of any possible scale, with the possibility of setting immediate harmonic and melodic auditory feedback. Use of the interval table and the keyboard, make experiments in interval tension and scale characteristics possible. Musicians can tune (ethnic) instruments according to specific scales using the direct feedback of the real-time analysis. Because of the MIDI messages, it is also possible to play the keyboard in the same scale as the instruments at hand.

In ethnomusicology, Tarsos can be a practical tool for direct pitch analysis of various instruments. Given the fact that pitch analysis results show up immediately, microphone positions during field recordings can be adjusted on the spot to optimize measurements.

### Output capabilities

Tarsos contains export capabilities for each step, from the raw pitch estimations until the pitch class interval matrix. The built-in functions can export the data as comma separated text files, charts, TEX-files, and there is a way to synthesize estimations. Since Tarsos is scriptable there is also a possibility to add other export functions or modify the existing functions. The API and scripting capabilities are documented on the Tarsos website: `http://0110.be/tag/JNMR`.

For pitch class data, there is a special standardized text file defined by the Scala[6] program. The Scala file format has the `.scl` extension. The Scala program comes with a dataset of over 3900 scales ranging from historical harpsichord temperaments over ethnic scales to scales used in contemporary music. Recently this dataset has been used to find the universal properties of scales (Honingh and Bod, 2011). Since Tarsos can export scala files it is possible to see if the star-convex

[6]See `http://www.huygens-fokker.org/scala/`

structures discussed by Honingh and Bod (2011) can be found in scales extracted from real audio. Tarsos can also parse Scala files, so that comparison of theoretical scales with tuning practice is possible. This feature is visualized by the upwards Scala arrow in Figure 2.2. When a scale is overlaid on a pitch class histogram, Tarsos finds the best fit between the histogram and the scala file.

A completely different output modality is MIDI. The MIDI Tuning Standard defines MIDI messages to specify the tuning of MIDI synthesizers. Tarsos can construct `Bulk Tuning Dump`-messages with pitch class data to tune a synthesizer enabling the user to play along with a song in tune. Tarsos contains the Gervill synthesizer, one of the very few (software) synthesizers that offer support for the MIDI Tuning Standard. Another approach to enable users to play in tune with an extracted scale is to send pitch bend messages to the synthesizer when a key is pressed. Pitch bend is a MIDI-message that tells how much higher or lower a pitch needs to sound in comparison with a standardized pitch. Virtually all synthesizers support pitch bend, but pitch bends operate on MIDI-channel level. This makes it impossible to play polyphonic music in an arbitrary tone scale.

### Scripting capabilities

Processing many audio files with the graphical user interface quickly becomes tedious. Scripts written for the Tarsos API can automate tasks and offer a possibility to utilize Tarsos' building blocks in entirely new ways. Tarsos is written in Java, and is extendable using scripts in any language that targets the JVM (Java Virtual Machine) like JRuby, Scala[7] and Groovy. For its concurrency support, concise syntax and seamless interoperability with Java, the Scala programming languages are used in example scripts, although the concepts apply to scripts in any language. The number of applications for the API is only limited by the creativity of the developer using it. Tasks that can be implemented using the Tarsos API are for example:

**Tone scale recognition:** given a large number of songs and a number of tone scales in which each song can be brought, guess the tone scale used for each song. In section 2.2.2 this task is explained in detail and effectively implemented.

**Modulation detection:** this task tries to find the moments in a piece of music where the pitch class histogram changes from one stable state to another. For western music this could indicate a change

[7]Please do not confuse the general purpose Scala programming language with the tool to experiment with tunings, the Scala program.

of mode, a modulation. This task is similar as the one described by Lesley Mearns (2011). With the Tarsos API you can compare windowed pitch histograms and detect modulation boundries.

**Evolutions in tone scale use:** this task tries to find evolutions in tone scale use in a large number of songs from a certain region over a long period of time. Are some pitch intervals becoming more popular than others? This is done by Moelants et al. (2009) for a set of African songs.

**Acoustic Fingerprinting:** it is theorized by Tzanetakis et al. (2002) that pitch class histograms can serve as an acoustic fingerprint for a song. With the building blocks of Tarsos: pitch detection, pitch class histogram creation and comparison this was put to the test by Six and Cornelis (2012a).

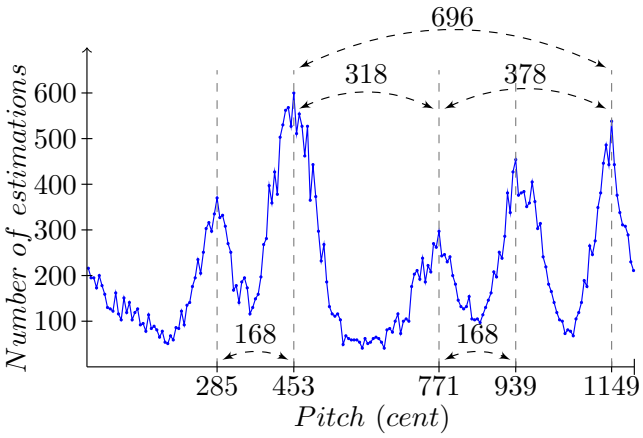
The article by Tzanetakis et al. (2002) gives a good overview of what can be done using pitch histograms and, by extension, the Tarsos API. To conclude: the Tarsos API enables developers to quickly test ideas, execute experiments on large sets of music and leverage the features of Tarsos in new and creative ways.

### 2.2.2 Exploring the capabilities of Tarsos through case studies

In what follows, we explore Tarsos' capabilities using case studies in non-Western music. The goal is to focus on problematic issues such as the use of different pitch extractors, music with pitch drift, and last but not least, the analysis of large databases.

#### Analysing a pitch histogram

We will first consider the analysis of a song that was recorded in 1954 by missionary Scohy-Stroobants in Burundi. The song is performed by a singing soloist, Léonard Ndengabaganizi. The recording was analysed with the YIN pitch detection method and a pitch class histogram was calculated: it can be seen in Figure 2.9. After peak detection on this histogram, the following pitch intervals were detected: 168, 318, 168, 210, and 336 cents. The detected peaks and all intervals are shown in an interval matrix (see Figure 2.9). It can be observed that this is a pentatonic division that comprises small and large intervals, which is different from an equal tempered or meantone division. Interestingly, the two largest peaks define a fifth interval, which is made of a pure minor third (318 cents) and a pure major third (378 cents) that lies between the intervals $168 + 210 = 378$ cents). In addition, a mirrored




(a) Pitch Class histogram.

| P.C. | 285 | 453 | 771 | 939 | 1149 |
|---|---|---|---|---|---|
| 285 | 0 | *168* | 486 | 654 | 864 |
| 453 | 1032 | 0 | *318* | 486 | **696** |
| 771 | 714 | 882 | 0 | *168* | *378* |
| 939 | 546 | 714 | 1032 | 0 | 210 |
| 1149 | 336 | 504 | 822 | 990 | 0 |

(b) Pitch class intervals.

Figure 2.9: This song uses an unequally divided pentatonic tone scale with mirrored intervals 168-318-168, indicated on the pitch class histogram. Also indicated is a near perfect fifth consisting of a pure minor and pure major third.

set of intervals is present, based on 168-318-168 cents. This phenomena is also illustrated by Figure 2.9.

### Different pitch extractors

However, Tarsos has the capability to use different pitch extractors. Here we show the difference between seven pitch extractors on a histogram level. A detailed evaluation of each algorithm cannot be covered in this article but can be found in the cited papers. The different pitch extractors are:

- YIN (de Cheveigné and Hideki, 2002) (YIN) and the McLeod Pitch Method (MPM), which is described by (McLeod and Wyvill, 2005), are two time-domain pitch extractors. Tarsos contains a platform independent implementation of the algorithms.

| | YIN | MPM | Schmitt | FHC | SC | MAMI 1 | MAMI 6 |
|---|---|---|---|---|---|---|---|
| YIN | 1.00 | **0.81** | *0.41* | 0.65 | 0.62 | 0.69 | 0.61 |
| MPM | **0.81** | 1.00 | 0.43 | 0.67 | 0.64 | 0.71 | 0.63 |
| Schmitt | *0.41* | 0.43 | 1.00 | 0.47 | 0.53 | 0.42 | 0.56 |
| FHC | 0.65 | 0.67 | 0.47 | 1.00 | 0.79 | 0.67 | 0.66 |
| SC | 0.62 | 0.64 | 0.53 | 0.79 | 1.00 | 0.65 | 0.70 |
| MAMI 1 | 0.69 | 0.71 | 0.42 | 0.67 | 0.65 | 1.00 | 0.68 |
| MAMI 6 | 0.61 | 0.63 | 0.56 | 0.66 | 0.70 | 0.68 | 1.00 |
| Average | 0.69 | 0.70 | *0.55* | 0.70 | 0.70 | 0.69 | 0.69 |

Table 2.2: Similarity matrix showing the overlap of pitch class histograms for seven pitch detection methods. The similarities are the mean of 2484 audio files. The last row shows the average of the overlap for a pitch detection method.

- Spectral Comb (SC), Schmitt trigger(Schmitt) and Fast Harmonic Comb (FHC) are described by Brossier (2006). They are available for Tarsos through VAMP-plugins (Cannam, 2008);

- MAMI 1 and MAMI 6 are two versions of the same pitch tracker. MAMI 1 only uses the most present pitch at a certain time, MAMI 6 takes the six most salient pitches at a certain time into account. The pitch tracker is described by Clarisse et al. (2002).

Figure 2.10 shows the pitch histogram of the same song as in the previous section, which is sung by an unaccompanied young man. The pitch histogram shows a small tessitura and wide pitch classes. However, the general contour of the histogram is more or less the same for each pitch extraction method, five pitch classes can be distinguished in about one-and-a-half octaves, ranging from 5083 to 6768 cent. Two methods stand out. Firstly, MAMI 6 detects pitch in the lower and higher regions. This is due to the fact that MAMI 6 always gives six pitch estimations in each measurement sample. In this monophonic song this results in octave - halving and doubling - errors and overtones. Secondly, the Schmitt method also stands out because it detects pitch in regions where other methods detect a lot less pitches, e.g. between 5935 and 6283 cent.

Figure 2.11 shows the pitch class histogram for the same song as in Figure 2.10, now collapsed into one octave. It clearly shows that it is hard to determine the exact location of each pitch class. However, all histogram contours look similar except for the Schmitt method, which results in much less well defined peaks. The following evaluation shows that this is not only the case.

In order to be able to gain some insight into the differences between

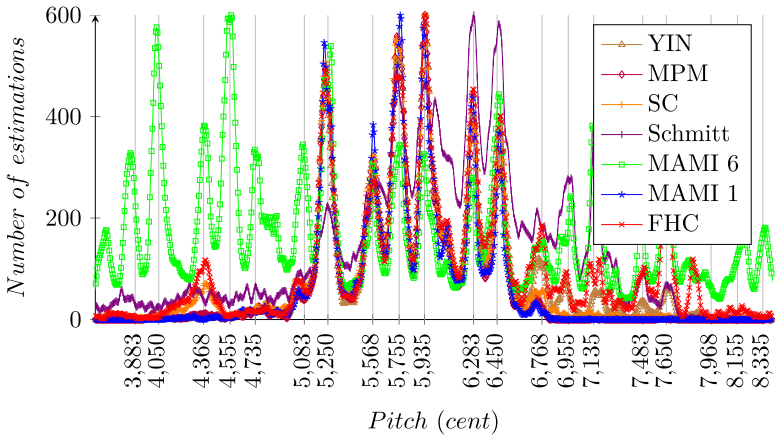



Figure 2.10: Seven different pitch histograms of a traditional Rwandese song. Five pitch classes repeat every octave. The Schmitt trigger (Schmitt) results in much less well defined peaks in the pitch histogram. MAMI 6 detects much more information to be found in the lower and higher regions, this is due to the fact that it always gives six pitch estimations, even if they are octave errors or overtones.

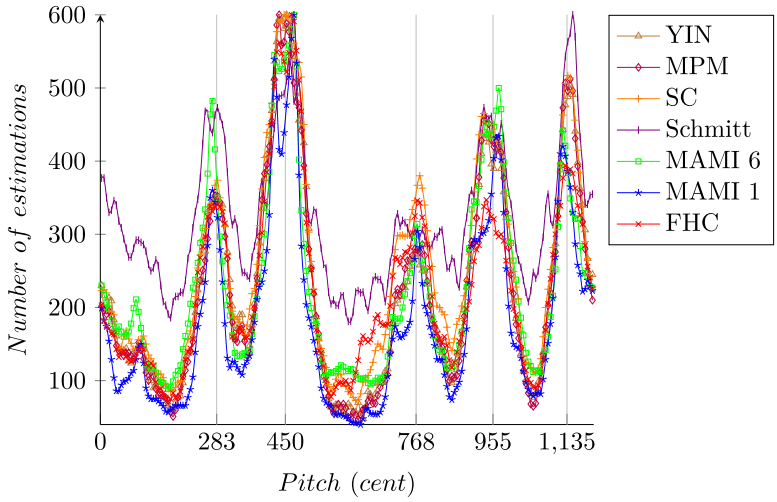



Figure 2.11: Seven different pitch class histograms of a traditional Rwandese song. Five pitch classes can be distinguished but is clear that it is hard to determine the exact location of each pitch class. The Schmitt trigger (Schmitt) results in a lot less well defined peaks in the pitch class histogram.

the pitch class histograms resulting from different pitch detection methods, the following procedure was used: for each song in a data set of more than 2800 songs - a random selection of the music collection of the Belgian Royal Museum of Central Africa (RMCA) - seven pitch class histograms were created by the pitch detection methods. The overlap - a number between zero and one - between each pitch class histogram pair was calculated. A sum of the overlap between each pair was made and finally divided by the number of songs. The resulting data can be found in Table 2.2. Here histogram overlap or intersection is used as a distance measure because Gedik and Bozkurt (2010) show that this measure works best for pitch class histogram retrieval tasks. The overlap $c(h_1, h_2)$ between two histograms $h_1$ and $h_2$ with $K$ classes is calculated with equation 2.2. For an overview of alternative correlation measures between probability density functions see Cha (2007).

$$c(h_1, h_2) = \frac{\sum_{k=0}^{K-1} min(h_1(k), h_2(k))}{max(\sum_{k=0}^{K-1} h_1(k), \sum_{k=0}^{K-1} h_2(k))} \tag{2.2}$$

The table 2.2 shows that there is, on average, a large overlap of 81%, between the pitch class histograms created by YIN and those by MPM. This can be explained by the fact that the two pitch extraction algorithms are very much alike: both operate in the time-domain with autocorrelation. The table also shows that Schmitt generates rather unique pitch class histograms. On average there is only 55% overlap with the other pitch class histogram. This performance was already expected during the analysis of one song (above).

The choice for a particular pitch detection method depends on the music and the analysis goals. The music can be monophonic, homophonic or polyphonic, different instrumentation and recording quality all have influence on pitch estimators. Users of Tarsos are encouraged to try out which pitch detection method suits their needs best. Tarsos' scripting API - see section 2.2.2 - can be helpful when optimizing combinations of pitch detection methods and parameters for an experiment.

### Shifted pitch distributions

Several difficulties in analysis and interpretation may arise due to pitch shift effects during musical performances. This is often the case with a capella choirs. Figure 2.13 shows a nice example of an intentionally raised pitch, during solo singing in the Scandinavian Sami culture. The short and repeated melodic motive remains the same during the entire song, but the pitch raises gradually ending up 900 cents higher than the beginning. Retrieving a scale for the entire song is in this case irrelevant, although the scale is significant for the melodic motive. Figure

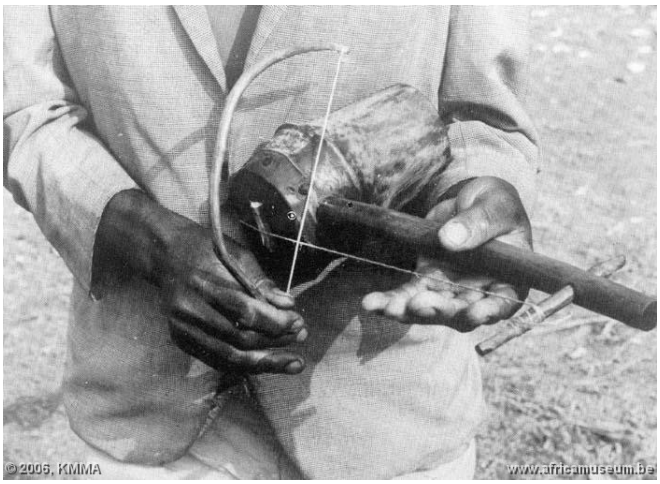



Figure 2.12: The iningidi, a type of African fiddle. To play the instrument, the neck is held in the palm of the left hand so that the string can be stopped using the second phalanx of the index, middle and ring fingers. Consequently, a total of four notes can be produced.

2.14 shows an example where scale organization depends on the characteristics of the instrument. This type of African fiddle, the iningidi, does not use a soundboard to shorten the strings. Instead the string is shortened by the fingers that are in a (floating) position above the string: an open string and three fingers give a tetratonic scale. Figure 2.12 shows an iningidi being played. This use case shows that pitch distributions for entire songs can be misleading, in both cases it is much more informative to compare the distribution from the first part of the song with the last part. Then it becomes clear how much pitch shifted and in what direction.

Interesting to remark is that these intervals have more or less the same distance, a natural consequence of the distance of the fingers, and that, consequently, not the entire octave tessitura is used. In fact only 600 cents, half an octave, is used. A scale that occurs typically in fiddle recordings, that rather can be seen as a tetrachord. The open string (lowest note) is much more stable than the three other pitches that deviate more, as is shown by the broader peaks in the pitch class histogram. The hand position without soundboard is directly related to the variance of these three pitch classes. When comparing the second minute of the song with the seventh, one sees a clear shift in pitch, which can be explained by the fact the musician changed the hand position a little bit. In addition, another phenomena can be observed,

namely, that while performing, the open string gradually loses tension, causing a small pitch lowering which can be noticed when comparing the two fragments. This is not uncommon for ethnic music instruments.

### Tarsos' scripting applied to Makam recognition

In order to make the use of scripting more concrete, an example is shown here. It concerns the analysis of Turkish classical music. In an article by Gedik and Bozkurt (2010), pitch histograms were used for - amongst other tasks - makam[8] recognition. The task was to identify which of the nine makams is used in a specific song. With the Tarsos API, a simplified, generalized implementation of this task was scripted in the Scala programming language. The task is defined as follows:

> For a small set of tone scales $T$ and a large set of musical performances $S$, each brought in one of the scales, identify the tone scale $t$ of each musical performance $s$ automatically.

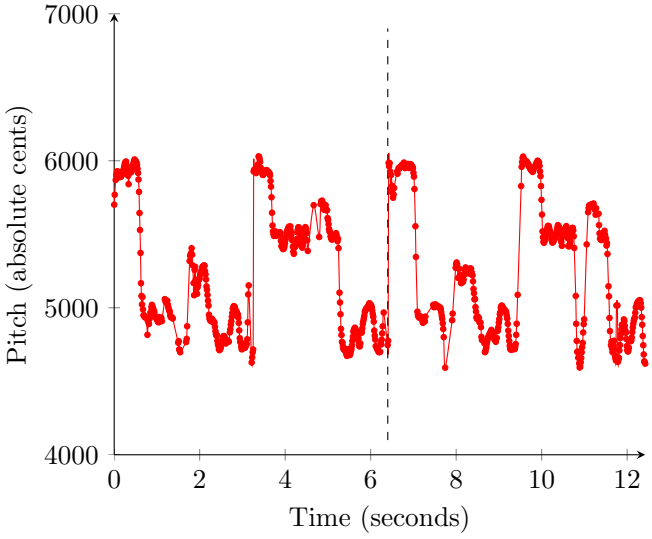
**Algorithm 1** Tone scale recognition algorithm

```
1:  T ← constructTemplates()
2:  S ← fetchSongList()
3:  for all s ∈ S do                                  ▷ For all songs
4:      O ← {}                                        ▷ Initialize empty hash
5:      h ← constructPitchClassHisto(s)
6:      for all t ∈ T do                              ▷ For all templates
7:          o ← calculateOverlap(t, h)
8:          O[s] ← o                                  ▷ Store overlap in hash
9:      end for
10:     i ← getFirstOrderedByOverlap(O))
11:     write s "is brought in tone scale" i
12: end for
```

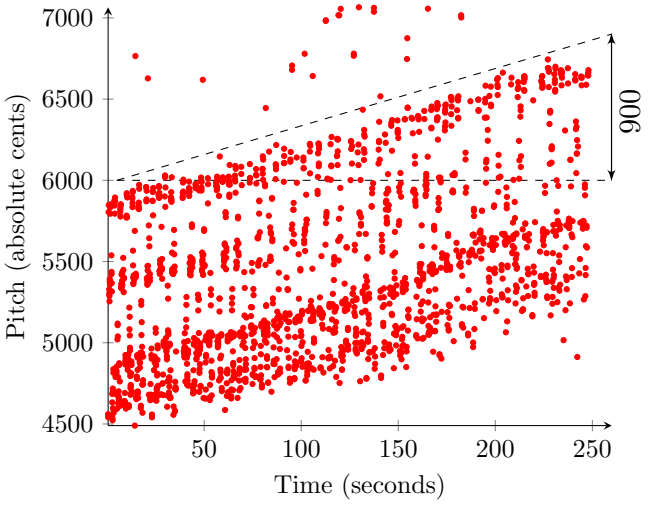
An example of makam recognition can be seen in Figure 2.15. A theoretical template - the dotted, red line - is compared to a pitch class histogram - the solid, blue line - by calculating the maximum overlap between the two. Each template is compared with the pitch class histogram, the template with maximum overlap is the guessed makam. Pseudocode for this procedure can be found in Algorithm 1.

To construct the tone-scale templates theoretical descriptions of those tone scales are needed, for makams these can be found in Gedik

[8] A makam defines rules for a composition or performance of classical Turkish music. It specifies melodic shapes and pitch intervals.




(a) Motive




(b) Contour

Figure 2.13: An a capella song performed by Nils Hotti from the Sami culture shows the gradual intentional pitch change during a song. The melodic motive however is constantly repeated (here shown twice).

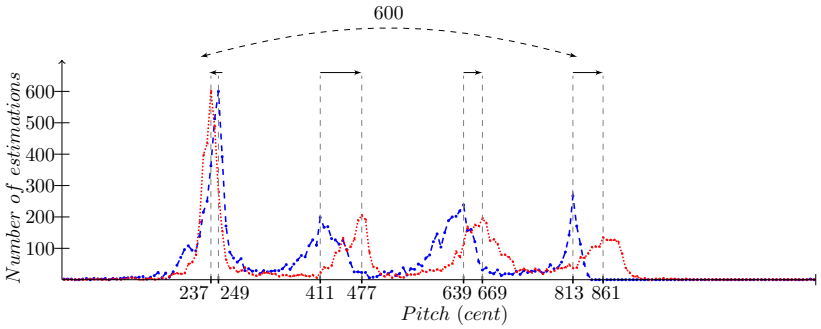



Figure 2.14: Histogram of an African fiddle song. The second minute of the song is represented by the dashed line, the seventh minute is represented by the dotted line. The lowest, most stable pitch class is the result of the open string. It lost some tension during the piece and started to sound lower. This is in sharp contrast with the other pitch classes that sound higher, due to a change in hand position.

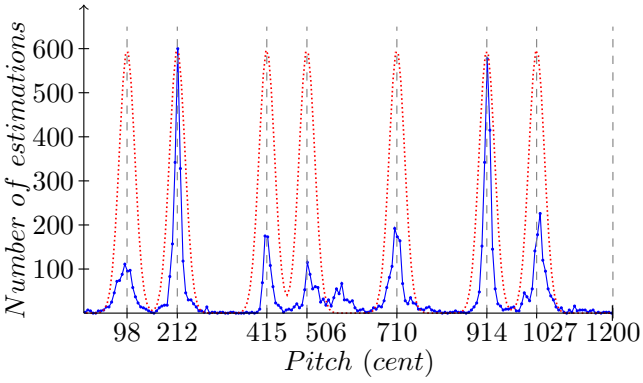



Figure 2.15: The solid, blue line is a pitch class histogram of a Turkish song brought in the makam Hicaz. The dotted, red line represents a theoretical template of that same Hicaz makam. Maximizing the overlap between a theoretical and an actual pitch class histogram suggests which makam is used.

| **Makam** | **Pitch classes (in cents)** | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Hicaz | 113 | | | | 384 | | 498 | 701 | 792 | | | 996 | |
| Huseyni | | 181 | | 294 | | | 498 | 701 | | | 883 | 996 | |
| Huzzam | 113 | | | | 316 | | 430 | | 701 | 812 | | | 1109 |
| Kurdili Hicazar | 90 | | | 294 | | | 498 | 701 | 792 | | | 996 | |
| Nihavend | | | 203 | 294 | | | 498 | 701 | 792 | | | 996 | |
| Rast | | | 203 | | 384 | | 498 | 701 | | | 905 | | 1086 |
| Saba | | 181 | | 294 | | 407 | | 701 | 792 | | | 996 | |
| Segah | 113 | | | | 316 | | 498 | 701 | | 815 | | | 1109 |
| Ussak | | 181 | | 294 | | | 498 | 701 | 792 | | | 996 | |

Table 2.3: The nine makams used in the recognition task.

**Listing 2.1: Template construction**

```
1 val makams = List(  "hicaz","huseyni","huzzam","
      kurdili_hicazar",
            "nihavend","rast","saba","segah","ussak")

  var theoreticKDEs = Map[java.lang.String,
      KernelDensityEstimate]()
  makams.foreach{ makam =>
6   val scalaFile =  makam + ".scl"
    val scalaObject = new ScalaFile(scalaFile);
    val kde = HistogramFactory.createPichClassKDE(scalaObject
        ,35)
    kde.normalize
    theoreticKDEs = theoreticKDEs + (makam -> kde)
11 }
```

and Bozkurt (2010). The pitch classes are converted to cent units and listed in Table 2.3. An implementation of *constructTemplates*() in Algorithm 1 can be done as in Listing 2.1. The capability of Tarsos to create theoretical tone scale templates using Gaussian kernels is used, line 8. Line 7 shows how a scala file containing a tone-scale description is used to create an object with the same information, the height is normalized.

The $calculateOverlap(t, h)$ method from line 7 in Algorithm 1 is pitch invariant: it shifts the template to achieve maximum overlap with respect to the pitch class histogram. Listing 2.2 contains an implementation of the matching step. First a list of audio files is created by recursively iterating a directory and matching each file to a regular expression. Next, starting from line 4, each audio file is processed. The internal implementation of the YIN pitch detection algorithm is used on the audio file and a pitch class histogram is created (line 6,7). On line 10, normalization of the histogram is activated, to make the correlation calculation meaningful. In line 11 to line 15 the created histogram from the audio file is compared with the templates calculated beforehand (in Listing 2.1). The results are stored, ordered and eventually printed on line 19.

The script ran on Bozkurts data set with Turkish music: with this straightforward algorithm it is possible to correctly identify 39% of makams in a data set of about 800 songs. The results for individual makam recognition vary between 76% and 12% depending on how distinguishable the makam is. If the first three guesses are evaluated, the correct makam is present in 75% of the cases. Obviously, there is room for improvement by using more domain knowledge. A large improve-

**Listing 2.2: Makam Recognition**

```
val directory = "/home/user/turkish_makams/"
val audio_pattern = ".*.(mp3|wav|ogg|flac)"
val audioFiles = FileUtils.glob(directory,audio_pattern,true
    ).toList

audioFiles.foreach{ file =>
  val audioFile = new AudioFile(file)
  val detectorYin = PitchDetectionMode.TARSOS_YIN.
      getPitchDetector(audioFile)
  val annotations = detectorYin.executePitchDetection()
  val actualKDE = HistogramFactory.createPichClassKDE(
      annotations,15);
  actualKDE.normalize
  var resultList = List[Tuple2[java.lang.String,Double]]()
  for ((name, theoreticKDE) <- theoreticKDEs){
      val shift = actualKDE.shiftForOptimalCorrelation(
          theoreticKDE)
      val currentCorrelation = actualKDE.correlation(
          theoreticKDE,shift)
      resultList =  (name -> currentCorrelation) ::
          resultList
  }
  //order by correlation
  resultList = resultList.sortBy{_._2}.reverse
  Console.println(file + " is brought in tone scale " +
      resultList(0)._1)
}
```

| **Makam** | **Songs (#)** | **Correct (#)** | **Correct (%)** |
|---|---|---|---|
| Kurdili Hicazar | 91 | 32 | 35.16% |
| Huseyni | 64 | 8 | 12.50% |
| Nihavend | 75 | 27 | 36.00% |
| Segah | 111 | 43 | 38.74% |
| Saba | 81 | 50 | 61.73% |
| Huzzam | 62 | 15 | 24.19% |
| Rast | 118 | 23 | 19.49% |
| Ussak | 102 | 54 | 52.94% |
| Hicaz | 68 | 52 | 76.47% |
| **Total** | 772 | 304 | 39.69% |

Table 2.4: Results of the makam recognition task, using theoretical intervals, on the Bozkurt data set with Turkish music.

ment can be made by taking into account the duration of each pitch class in each template. Bozkurt does this by constructing templates by using the audio itself. A detailed failure analysis falls outside the scope of this article. It suffices to say that practical tasks can be scripted successfully using the Tarsos API.

### 2.2.3 Musicological aspects of Tarsos

Tarsos is a tool for the analysis of pitch distributions. For that aim, Tarsos incorporates several pitch extraction modules, has pitch distribution filters, audio feedback tools, and scripting tools for batch processing of large databases of musical audio. However, pitch distributions can be considered from different perspectives, such as ethnographical studies of scales (Schneider, 2001), theoretical studies in scale analysis (Sethares, 2005), harmonic and tonal analysis (Krumhansl, 1990; Krumhansl and Shepard, 1979), and other structural analysis approaches to music (such as set theoretical and Schenkerian). Clearly, Tarsos does not offer a solution to all these different approaches to pitch distributions. In fact, seen from the viewpoint of Western music analysis, Tarsos is a rather limited tool as it doesn't offer harmonic analysis, nor tonal analysis, nor even statistical analysis of pitch distributions. All of this should be applied together with Tarsos, when needed. Instead, what Tarsos provides is an intermediate level between pitch extraction (done by pitch extractor tools) and music theory. The major contribution of Tarsos is that it offers an easy to use tool for pitch distribution analysis that applies to all kinds of music, including Western and non-Western. The major contribution of Tarsos, so to speak, is that it offers pitch distribution analysis without imposing a music theory. In what follows, we explain why such tools are needed and why they are useful.

#### Tarsos and Western music theoretical concepts

Up to recently, musical pitch is often considered from the viewpoint of a traditional music theory, which assumes that pitch is stable (e.g. vibrato is an ornament of a stable pitch), that pitch can be segmented into tones, that pitches are based on octave equivalence, and that octaves are divided into 12 equal-sized intervals of each 100 cents, and so on. These assumptions have the advantage that music can be reduced to symbolic representations, a written notation, or notes, whose structures can be studied at an abstract level. As such, music theory has conceptualized pitch distributions as chords, keys, modes, sets, using a symbolic notation.

So far so good, but tools based on these concepts may not work for many nuances of Western music, and especially not for non-Western

music. In Western music, tuning systems have a long history. Proof of this can be found in tunings of historical organs, and in tuning systems that have been explored by composers in the 20th century (cf. Alois Haba, Harry Partch, Ivo Darreg, and Lamonte Young). Especially in non-Western classical music, pitch distributions are used that radically differ from the Western theoretical concepts, both in terms of tuning, as well as in pitch occurrence, and in timbre. For example, the use of small intervals in Arab music contributes to nuances in melodic expression. To better understand how small pitch intervals contribute to the organization of this music, we need tools that do no assume octave divisions in 12 equal-sized intervals (see Gedik and Bozkurt (2010)). Other types of music do not have octave equivalence (cf. the Indonesian gamalan), and also some music work with modulated pitch. For example, Henbing and Leman (2007) describe classical Chinese guqin music which uses tones that contain sliding patterns (pitch modulations), which form a substantial component of the tone and consider it as a succession of prototypical gestures. Krishnaswamy (2004a,b) introduces a set of 2D melodic units, melodic atoms, in describing Carnatic (South-Indian classical) music. They represent or synthesize the melodic phrase and are not bound by a scale type. Hence, tools based on Western common music theoretical conceptions of pitch organization may not work for this type of music.

Oral musical traditions (also called ethnic music) provide a special case since there is no written music theory underlying the pitch organization. An oral culture depends on societal coherence, interpersonal influence and individual musicality, and this has implications on how pitch gets organized. Although oral traditions often rely on a peculiar pitch organization, often using a unique system of micro-tuned intervals, it is also the case that instruments may lack a fixed tuning, or that tunings may strongly differ from one instrument to the other, or one region to the other. Apparently, the myriad of ways in which people succeed in making sense out of different types of pitch organization can be considered as cultural heritage that necessitates a proper way of documentation and study (Cornelis et al., 2010b; Moelants et al., 2009).

Several studies attempt at developing a proper approach to pitch distributions. Gómez and Bonada (2008) look for pitch gestures in European folk music as an additional aspect to pitch detection. Moving from tone to scale research, Chordia and Rae (2007) acknowledges interval differences in Indian classical music, but reduces to a chromatic scale for similarity analysis and classification techniques. Sundberg and Tjernlund (1969) developed, already in 1969, an automated method for extracting pitch information from monophonic audio for assembling the scale of the spilåpipa by frequency histograms. Bozkurt (2008); Gedik

and Bozkurt (2010) build a system to classify and recognize Turkish maqams from audio files using overall frequency histograms to characterize the maqams scales and to detect the tonic centre. Maqams contain intervals of different sizes, often not compatible with the chromatic scale, but partly relying on smaller intervals. Moelants et al. (2007) focuses on pitch distributions of especially African music that deals with a large diversity of irregular tuning systems. They avoid a priori pitch categories by using a quasi-continuous rather than a discrete interval representation. In Moelants et al. (2009) they show that African songs have shifted more and more towards Western well temperament from 1950s to 1980s.

To sum up, the study of pitch organization needs tools that go beyond elementary concepts of the Western music theoretical canon (such as octave equivalence, stability of tones, equal temporal scale, and so on). This is evident from the nuances of pitch organization in Western music, in non-Western classical music, as well as in oral music cultures. Several attempts have been undertaken, but we believe that a proper way of achieving this is by means of a tool that combines audio-based pitch extraction with a generalized approach to pitch distribution analysis. Such a tool should be able to automatically extract pitch from musical audio in a culture-independent manner, and it should offer an approach to the study of pitch distributions and its relationship with tunings and scales. The envisioned tool should be able to perform this kind of analysis in an automated way, but it should be flexible enough to allow a musicologically grounded manual fine-tuning using filters that define the scope at which we look at distributions. The latter is indeed needed in view of the large variability of pitch organization in music all over the world. Tarsos is an attempt at supplying such a tool. One the one hand, Tarsos tries to avoid music theoretical concepts that could contaminate music that doesn't subscribe the constraints of the Western music theoretical canon. On the other hand, the use of Tarsos is likely to be too limited, as pitch distributions may further draw upon melodic units that may require an approach to segmentation (similar to the way segmented pitch relates to notes in Western music) and further gestural analysis (see the references to the studies mentioned above).

### Tarsos pitfalls

The case studies from section 2.2.2 illustrate some of the capabilities of Tarsos as tool for the analysis of pitch distributions. As shown Tarsos offers a graphical interface that allows a flexible way to analyse pitch, similar to other editors that focus on sound analysis (Sonic Visualizer, Audacity, Praat). Tarsos offers support for different pitch extractors, real-time analysis (see section 2.2.1), and has numerous output capa-

bilities (See section 2.2.1). The scripting facility allows us to use of Tarsos' building blocks in unique ways efficiently.

However, Tarsos-based pitch analysis should be handled with care. The following three recommendations may be taken into account: First of all, one cannot extract scales without *considering the music itself.* Pitch classes that are not frequently used, won't show up clearly in a histogram and hence might be missed. Also not all music uses distinct pitch classes: the Chinese and Indian music traditions have been mentioned in this case. Because of the physical characteristics of the human voice, voices can glide between tones of a scale, which makes an accurate measurement of pitch not straightforward. It is recommended to zoom in on the estimations in the melograph representation for a correct understanding.

Secondly, analysis of *polyphonic recordings should be handled with care* since current pitch detection algorithms are primarily geared towards monophonic signals. Analysis of homophonic singing for example may give incomplete results. It is advisable to try out different pitch extractors on the same signal to see if the results are trustworthy.

Finally, Schneider (2001) recognizes the use of "pitch categories" but warns that, especially for complex inharmonic sounds, *a scale is more than a one dimensional series of pitches* and that spectral components need to be taken into account to get better insights in tuning and scales. Indeed, in recent years, it became clear that the timbre of tones and the musical scales in which these tones are used, are somehow related (Sethares, 2005). The spectral content of pitch (i.e. the timbre) determines the perception of consonant and dissonant pitch intervals, and therefore also the pitch scale, as the latter is a reflection of the preferred melodic and harmonic combinations of pitch. Based on the principle of minimal dissonance in pitch intervals, it is possible to derive pitch scales from spectral properties of the sounds and principles of auditory interference (or critical bands). Schwartz and Purves (2004) argue that perception is based on the disambiguation of action-relevant cues, and they manage to show that the harmonic musical scale can be derived from the way speech sounds relate to the resonant properties of the vocal tract. Therefore, the annotated scale as a result of the precise use of Tarsos, does not imply the assignment of any characteristic of the music itself. It is up to the user to correctly interprete of a possible scale, a tonal center, or a melodic development.

**Tarsos - future work**

The present version of Tarsos is a first step towards a tool for pitch distribution analysis. A number of extensions are possible.

For example, given the tight connection between timbre and scale,

it would be nice to select a representative tone from the music and transpose it to the entire scale, using a phase vocoder. This sound sample and its transpositions could then be used as a sound font for the MIDI synthesizer. This would give the scale a more natural feel compared to the general MIDI device instruments that are currently present.

Another possible feature is tonic detection. Some types of music have a well-defined tonic, e.g. in Turkish classical music. It would make sense to use this tonic as a reference pitch class. Pitch histograms and pitch class histograms would then not use the reference frequency defined in appendix 2.2.5 but a better suited, automatically detected reference: the tonic. It would make the intervals and scale more intelligible.

Tools for comparing two or more scales may be added. For example, by creating pitch class histograms for a sliding window and comparing those with each other, it should be possible to automatically detect modulations. Using this technique, it should also be possible to detect pitch drift in choral, or other music.

Another research area is to extract features on a large data set and use the pitch class histogram or interval data as a basis for pattern recognition and cluster analysis. With a time-stamped and geo-tagged musical archive, it could be possible to detect geographical or chronological clusters of similar tone scale use.

On the longer term, we plan to add representations of other musical parameters to Tarsos as well, such as rhythmic and instrumental information, temporal and timbral features. Our ultimate goal is to develop an objective albeit partial view on music by combining those three parameters within an easy to use interface.

### 2.2.4 Conclusion

In this paper, we have presented Tarsos, a modular software platform to extract and analyze pitch distributions in music. The concept and main features of Tarsos have been explained and some concrete examples have been given of its usage. Tarsos is a tool in full development. Its main power is related to its interactive features which, in the hands of a skilled music researcher, can become a tool for exploring pitch distributions in Western as well as non-Western music.

### 2.2.5 Appendix A - pitch representation

Since different representations of pitch are used by Tarsos and other pitch extractors this section contains definitions of and remarks on different pitch and pitch interval representations.

For humans the perceptual distance between 220Hz and 440Hz is the same as between 440Hz and 880Hz. A pitch representation that takes this logarithmic relation into account is more practical for some purposes. Luckily there are a few:

**MIDI Note Number**

The MIDI standard defines note numbers from 0 to 127, inclusive. Normally only integers are used but any frequency $f$ in Hz can be represented with a fractional note number $n$ using equation 2.3.

$$n = 69 + 12\log_2(\frac{f}{440}) \tag{2.3}$$

$$n = 12 \times \log_2(\frac{f}{r}) \; ; \; r = \frac{440}{2^{(69/12)}} = 8.176\text{Hz} \tag{2.4}$$

Rewriting equation 2.3 to 2.4 shows that MIDI note number 0 corresponds with a reference frequency of 8.176Hz which is $C_{-1}$ on a keyboard with $A_4$ tuned to 440Hz. It also shows that the MIDI standard divides the octave in 12 equal parts.

To convert a MIDI note number $n$ to a frequency $f$ in Hz one of the following equations can be used.

$$f = 440 \times 2^{(n-69)/12} \tag{2.5}$$

$$f = r \times 2^{(n/12)} \text{ with } r = 8.176\text{Hz} \tag{2.6}$$

Using pitch represented as fractional MIDI note numbers makes sense when working with MIDI instruments and MIDI data. Although the MIDI note numbering scheme seems oriented towards western pitch organization (12 semitones) it is conceptually equal to the cent unit which is more widely used in ethnomusicology.

**Cent**

von Helmholtz and Ellis (1912) introduced the nowadays widely accepted cent unit. To convert a frequency $f$ in Hz to a cent value $c$ relative to a reference frequency $r$ also in Hz.

$$c = 1200 \times \log_2(\frac{f}{r}) \tag{2.7}$$

With the same reference frequency $r$ equations 2.7 and 2.4 differ only by a constant factor of exactly 100. In an environment with

pitch representations in MIDI note numbers and cent values it is practical to use the standardized reference frequency of 8.176Hz.

To convert a frequency $f$ in Hz to a cent value $c$ relative to a reference frequency $r$ also in Hz.

$$f = r \times 2^{(c/1200)} \tag{2.8}$$

**Savart & Millioctaves**

Divide the octave in 301.5 and 1000 parts respectively, which is the only difference with cents.

### Pitch Ratio Representation

Pitch ratios are essentially pitch intervals, an interval of one octave, 1200 cents equal to a frequency ratio of 2/1. To convert a ratio $t$ to a value in cent $c$:

$$c = \frac{1200 \ln(t)}{\ln(2)} \tag{2.9}$$

The natural logarithm, the logarithm base $e$ with $e$ being Euler's number, is noted as ln. To convert a value in cent $c$ to a ratio $t$:

$$t = e^{\frac{c \ln(2)}{1200}} \tag{2.10}$$

Further discussion on cents as pitch ratios can be be found in appendix B of Sethares (2005). There it is noted that:

> There are two reasons to prefer cents to ratios: Where cents are added, ratios are multiplied; and it is always obvious which of two intervals is larger when both are expressed in cents. For instance, an interval of a just fifth, followed by a just third is $(3/2)(5/4) = 15/8$, a just seventh. In cents, this is $702 + 386 = 1088$. Is this larger or smaller than the Pythagorean seventh 243/128? Knowing that the latter is 1110 cents makes the comparison obvious.

### Conclusion

The cent unit is mostly used for pitch interval representation while the MIDI key and Hz units are used mainly to represent absolute pitch. The main difference between cent and fractional MIDI note numbers is the standardized reference frequency. In our software platform Tarsos we use the exact same standardized reference frequency of 8.176Hz which enables us to use cents to represent absolute pitch and it makes

conversion to MIDI note numbers trivial. Tarsos also uses cents to represent pitch intervals and ratios.

### 2.2.6 Appendix B - audio material

Several audio files were used in this paper to demonstrate how Tarsos works and to clarify musical concepts. In this appendix you can find pointers to these audio files.

The thirty second excerpt of the musical example used throughout chapter 2.2.1 can be downloaded on `http://tarsos.0110.be/tag/JNMR` and is courtesy of: WERGO/Schott Music & Media, Mainz, Germany, `www.wergo.de` and Museum Collection Berlin. Ladrang Kandamanyura (slendro pathet manyura) is track eight on Lestari - *The Hood Collection, Early Field Recordings from Java* - `SM 1712 2`. It is recorded in 1957 and 1958 in Java.

The yoiking singer of Figure 2.13 can be found on a production released on the label Caprice Records in the series of Musica Sveciae Folk Music in Sweden. The album is called Jojk CAP 21544 CD 3, Track No 38 Nila, hans svager/His brother-in-law Nila.

The API example (section 2.2.2) was executed on the data set by Bozkurt. This data set was also used in (Gedik and Bozkurt, 2010). The Turkisch song brought in the makam Hicaz from Figure 2.15 is also one of the songs in the data set.

For the comparison of different pitch trackers on pitch class histogram level (section 2.2.2) a subset of the music collection of the Royal Museum for Central Africa (RMCA, Tervuren, Belgium) was used. We are grateful to the RMCA for providing access to its unique archive of Central African music. A song from the RMCA collection was also used in section 2.2.2. It has the tape number MR.1954.1.18-4 and was recorded in 1954 by missionary Scohy-Stroobants in Burundi. The song is performed by a singing soloist, Léonard Ndengabaganizi. Finally the song with tape number MR.1973.9.41-4, also from the collection of the RMCA, was used to show pitch shift within a song (Figure 2.14). It is called *Kana nakunze* and is recorded by Jos Gansemans in Mwendo, Rwanda in the year 1973.

## 2.3 A case for reproduciblity in MIR. Replication of 'a highly robust audio fingerprinting system'

Six, J., Bressan, F., and Leman, M. (In press – 2018a). A case for reproduciblity in MIR. Replication of 'a highly robust audio fingerprinting system'. *Transactions of the International Society for Music Information Retrieval (TISMIR)*

## Abstract

*This article makes a case for reproducibility in MIR research. Claims made in many MIR publications are hard to verify due to the fact that (i) often only a textual description is made available and code remains unpublished – leaving many implementation issues uncovered; (ii) copyrights on music limit the sharing datasets; and (iii) incentives to put effort into reproducible research – publishing and documenting code and specifics on data – is lacking.*

*In this article the problems around reproducibility are illustrated by replicating a MIR work. The system and evaluation described in 'A Highly Robust Audio Fingerprinting System' is replicated as closely as possible. The replication is done with several goals in mind: to describe difficulties in replicating the work and subsequently reflect on guidelines around reproducible research. Added contributions are the verification of the reported work, a publicly available implementation and an evaluation method that is reproducible.*

**Keywords:** Replication, Acoustic fingerprinting, Reproducibility.

### 2.3.1 Introduction

Reproducibility is one of the corner-stones of scientific methodology. A claim made in a scientific publication should be verifiable and the described method should provide enough detail to allow replication, "reinforcing the transparency and accountability of research processes" (Levin et al., 2016, p.129). The Open Science movement has recently gained momentum among publishers, funders, institutions and practicing scientists across all areas of research. It is based on the assumption that promoting "openness" will foster equality, widen participation, and increase productivity and innovation in science. *Re-usability* is a keyword in this context: data must be "useful rather than simply available" (Levin et al., 2016, p.133), with a focus on facilitating the advancement of knowledge based on previous work (spot check to avoid repeating work rather than on verifying the correctness of previous work.
From a technical standpoint, sharing tools and data has never been easier. Reproducibility, however, remains a problem. Especially for Music Information Retrieval (MIR) research and, more generally, research involving complex software systems. This problem has several causes:

- Journal articles and especially conference papers have limited space for detailed descriptions of methods or algorithms. Even for only moderately complex systems there are numerous parameters, edge cases and details which are glossed over in textual descriptions. This makes articles readable and the basic method intelligible, but those details need to be expounded somewhere. The ideal place for such details is well documented, runnable code. Unfortunately, **intellectual property rights** by universities or research institutions often limit researchers to freely distribute their code. This is problematic since it leaves the ones reproducing the work guessing for details. It makes replicating a study prohibitively hard. Even if there is code available it is often not documented well or it is very hard to make it actually run and reproduce results.

- **Copyrights on music** make it hard to share music freely. MIR research often has commercial goals and focuses on providing access to commercial, popular music. It is sensible to use commercial music while doing research as well. Unfortunately this makes it potentially very expensive to reproduce an experiment: all music needs to be purchased again and again by researchers reproducing the work.

The original research also needs to uniquely identify the music used, which is challenging if there are several versions, re-issues or recordings of a similarly titled track. Audio fingerprinting techniques allow us to share unique identifiers[9] but in practice this is rarely done. If annotations are part of the data set, a description of how the annotations were made, and by whom, alleviates methodological problems with reusing said data sets. A list of best practices specifically for MIR corpora is given by Peeters and Fort (2012).

Redistribution of historical field-recordings in museum archives is even more problematic. Due to the nature of the recordings, copyright status is often unclear. Clearing the status of such tracks involves international, historical copyright laws and multiple stakeholders such as performers, soloists, the museum, the person who performed the field recording and potentially a publisher that already published parts on an LP. The rights of each stakeholder need to be carefully considered, while at the same time being difficult to identify due to a lack of precise meta-data and awing also to the passage of time. While it is possible to clear a few tracks, it quickly becomes an insurmountable obstacle to clear a representative set of recordings for the research community. For very old recordings where copyright is not a problem, sometimes there are ethical issues related to public sharing: some Australian indigenous music, for instance, is considered very private and not meant to be listened to by others[10]. Recordings of women performing Torah chants is another example of sensitive material not trivially shared.

- The evaluation of research work (and most importantly of researchers) is currently based on the number of articles published in ranked scientific journals or conferences. Other types of scientific products are not valued as much. The advantage of investing resources in documenting, maintaining and publishing reproducible research and supplementary material is not often obvious in the effort of prioritising and strategising research outputs (Levin et al., 2016, p.130). Short-lived project funding is also a factor that directs attention of researchers to short-term output (publications), and not to long-term aspects of reproducible contributions to a field. In short there is **no incentive** to spend much time on non-textual output.

[9]The music meta-data service Musicbrainz, for example, uses Chromaprint to assign a unique identifier to a recording, based on the audio content.

[10]The Australian government described best practices in "Guidelines for Ethical Research in Australian Indigenous Studies".

Reproducing works is not an explicit tradition in computer science research. In the boundless hunt to further the state-of-the-art there seems no time or place for a sudden standstill and reflection on previous work. Implicitly, however, there seems to be a lot of replication going on. It is standard practice to compare results of a new method with earlier methods (baselines) but often it is not clear whether authors reimplemented those baselines or managed to find or modify an implementation. It is also not clear if those baselines are verified for correctness by reproducing the results reported in the original work. Moreover, due to a lack of standardized datasets, approaches are often hard to compare directly.

If all goes well, exact replications do not contain any new findings and therefore they may be less likely to get published. A considerable amount of work that risks remaining unpublished is not a proposition many researchers are looking forward to, expressing the tension between acting for the good of the community and their own (Nosek et al., 2015).

In the social sciences the reproducibility project illustrated that the *results* of many studies could not be successfully reproduced (Collaboration et al., 2015; Pashler and Wagenmakers, 2012) mainly due to small sample sizes and selection bias, a finding that was also demonstrated in a special issue in Musicae Scientiae on *Replication in music psychology* (Fischinger, 2013). In these replicated studies the main problem did not lay in replicating methods.

For research on complex software systems (MIR) it is expected that the replicated results will closely match the original if the method can be accurately replicated and if the data are accessible. But those two conditions are hard to meet. The replication problem lies exactly in the difficulties in *replicate the method and to access the data.* Once method and data are available, a statistical analysis on the behavior of deterministic algorithms is inherently less problematic than on erratic humans. Sturm (2012) showed that even if data and method are available, replication can be challenging if the problem is ill-defined and the test data contains inconsistencies.

Even if there is little doubt on the accuracy of reported results, the underlying need for replication remains. First of all, it checks if the problem is well-defined. Secondly, it tests if the method is described well and in fine enough detail. Thirdly ,it tests if the data used are described well and accessible. And finally results are also confirmed. It serves basically to *check if proper scientific methodology is used* and solidifies the original work.

**Open Science and MIR**

Open Science doesn't come as a set of prescriptive rules, but rather as a set of principles centred around the concept of "openness", with (i) theoretical, (ii) technological/practical and (iii) ethical implications. Each scientific community needs to identify how Open Science applies to its own domain, developing "the infrastructures, algorithms, terminologies and standards required to disseminate, visualise, retrieve and re-use data" (Leonelli, 2016, p.5). A general survey on Open Science policies in the field of MIR has never been performed, so an overview of their current application and their specific declination is not clearly defined. However, the members of this community have an implicit understanding of their own methods and their common practices to spread their materials and outputs, making it possible to lay out some fixed points. For example, methods, runnable annotated code and data sets, are key to MIR reproducibility. Often research serves precisely to introduce an improvement, a variation or an extension to an existing algorithm. When the algorithm is not available, it needs to be re-created in order to implement the modification – which is not only resource consuming, but never gives the guarantee that the re-created code matches the antecedent down to the last detail (Mesirov, 2010; Peng, 2011). Data sets are also very important, and they should be made available – if not for the general public, at least for peers and/or for reviewers.

Implementing Open Science policies in their full potential would change the face of science practice as we know it today. But its achievement requires a capillar change in how we understand our day-to-day research activities and how we carry them out, and right now we are in a situation where most researchers endorse openness yet "struggle to engage in community-oriented work because of the time and effort required to format, curate, and make resources widely available" Leonelli and Ankeny (2015). At the same time, the adoption of Open Science policies is encouraged but not mandatory and the "variety of constraints and conditions relevant to the sharing of research materials" creates "confusion and disagreement" among researchers (Levin et al., 2016, p.130). A recent survey of biomedical researchers in the United Kingdom Levin et al. (2016) identified 9 external factors that affect the practice of Open Science, including the existence (or lack) of repositories and databases for data, materials, software and models; the credit system in academic research; models and guidelines for intellectual property; collaborations with industrial partners, as well as attempts at commercialization and the digital nature of research. These constraints are generally applicable across scientific domains, thus including MIR – where the aspect of commercialization emerges much

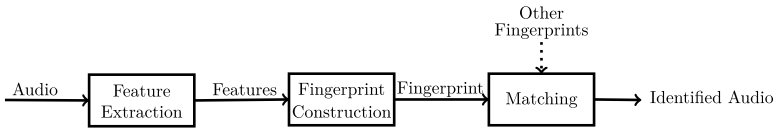



Figure 2.16: A generalized audio fingerprinter scheme. Audio is fed into the system, features are extracted and fingerprints constructed. The fingerprints are consecutively compared with a database containing the fingerprints of the reference audio. The original audio is either identified or, if no match is found, labeled as unknown.

earlier in the research workflow, at the level of music collections that need to be purchased.

So thinking of Open Science in MIR, where systematic support of reproducibility is but one of the possible applications, is an invitation to think about openness in relation to "all components of research, including data, models, software, papers, and materials such as experimental samples" (Levin et al., 2016, p.132). An important and cross-domain side aim of Open Science is also to show the importance of "encouraging critical thinking and ethical reflection among the researchers involved in data processing practices" (Leonelli, 2016, p.3). Open Science is not only about materials and platforms, but about people: the 'social' is not merely 'there' in science: "it is capitalised upon and upgraded to become an instrument of scientific work" (Knorr-Cetina, 1999, p.29).

### Replicating an acoustic fingerprinting system

This work replicates an acoustic fingerprinting system. This makes it one of the very few reported replication articles in Music Information Retrieval. Sturm (2012); Sturm and Noorzad (2012) also replicated MIR systems. They replicated two musical genre classification systems to critically review the systems and to challenge the reported high performances. Our aim is to highlight the reproducibility aspects of a milestone acoustic fingerprinting paper and to provide an illustration of good research practices. In doing so we will also provide an implementation to the research community, as well as solidifying the original acoustic fingerprinting research.

An acoustic fingerprint is a condensed representation of audio that can be matched reliably and quickly with a large set of fingerprints extracted from reference audio. The general acoustic fingerprinting system process is depicted in Figure 2.16. A short query is introduced in the system. Fingerprints are extracted from the query audio and

subsequently compared with a large set of fingerprints in the reference database. Finally, either a match is found or it is reported that it is not present in the database. Such acoustic fingerprint systems have many use-cases such as digital rights management, identifying duplicates (Cotton and Ellis, 2010; Six et al., 2018b), audio synchronization (Six and Leman, 2015) or labeling untagged audio with metadata (Bressan et al., 2017b).

The requirements for an acoustic fingerprinting system are described by Cano et al. (2005). They need to be granular, robust, reliable and economic in terms of storage requirements and computational load while resolving a query. Granular means that only a short fragment is needed for identification. Robustness is determined by various degradations a query can be subjected to while remaining recognizable. Degradations can include additional noise, low-quality encoding, compression, equalization, pitch-shifting and time-stretching. The ratios between true/false positives/negatives determine the reliability of the system. To allow potentially millions of reference items, an economy in terms of storage space is needed. Finally, resolving a query needs to be economic in terms of computational load. The weight of each requirement can shift depending on the context: if only a couple of hundred items end up in the reference database, the low storage space requirement is significantly relaxed.

Acoustic fingerprinting is a well researched MIR topic and over the years several efficient acoustic fingerprinting methods have been introduced (Allamanche, 2001; Coover and Han, 2014; Ellis et al., 2011; Haitsma and Kalker, 2002; Herre et al., 2002; Wang, 2003). These methods perform well even with degraded audio quality and with industrial sized reference databases. Some systems are able to recognize audio even when pitch-shifts are present (Bellettini and Mazzini, 2008; Fenet et al., 2011; Ouali et al., 2014; Ramona and Peeters, 2013) but without allowing for time-scale modification. Others systems are designed to handle both pitch and time-scale modification at the same time. Some systems only support small (Malekesmaeili and Ward, 2013; Zhu et al., 2010) datasets, others relatively large ones (Six and Leman, 2014; Sonnleitner and Widmer, 2016; Wang and Culbert, 2009).

This work replicates and critically reviews an acoustic fingerprinting system by Haitsma and Kalker (2002). The ISMIR proceedings article is from 2002 and it is elaborated upon by an article in the Journal of New Music Research (Haitsma and Kalker, 2003). The paper was chosen for several reasons:

1. It is widely cited: the ISMIR paper is cited more than 750 times and more than 250 times since 2013 according to Google Scholar. This indicates that it is *relevant* and still relevant today. A recent

study, for example, improved the system by replacing the FFT with a filter bank (Plapous et al., 2017). Another study (Coover and Han, 2014) improved its robustness against noise.

2. It is a paper that has a very *prototypical* structure which presents and evaluates a MIR system. The system, in this case, is an acoustic fingerprinting system. Replicating this work, in other words, should be similar to replicating many others.

3. The described algorithm and the evaluation method are *only moderately complex and self-contained*. They only depend on regularly available tools or methods. Note that this reason is symptomatic of the reproducibility problem: some papers are borderline impossible to replicate.

#### Contributions

The contributions of this article are either generally applicable or specifically about the replicated work. The specific contributions are the verification of the results described by Haitsma and Kalker (2002) and a solidification of the work. A second contribution lies in a publicly available, verifiable, documented implementation of the method of that paper[11]. Another contribution is the reproducible evaluation framework. The more general contributions are a further problematization of reproducibility in MIR and guidelines to make MIR work reproducible.

The paper continues with introducing the method that is replicated and the problems encountered while replicating it. Subsequently the same is done for the evaluation. To ameliorate problems with respect to replicability in the original evaluation an alternative evaluation is proposed. The results are compared and finally a discussion follows where guidelines are proposed.

### 2.3.2 Method

As with most acoustic fingerprinting systems this method consists of a fingerprint extraction step and a search strategy. In the terminology of Figure 2.16 this would be the feature extraction/fingerprint construction step and the matching step.

#### Fingerprint extraction

The fingerprint extraction algorithm is described in more detail in section 4.2 of Haitsma and Kalker (2002) but is summarized here as well.

[11] The implementation part of Panako, an acoustic fingerprinting framework. Panako is available at `http://github.com/JorenSix/Panako`

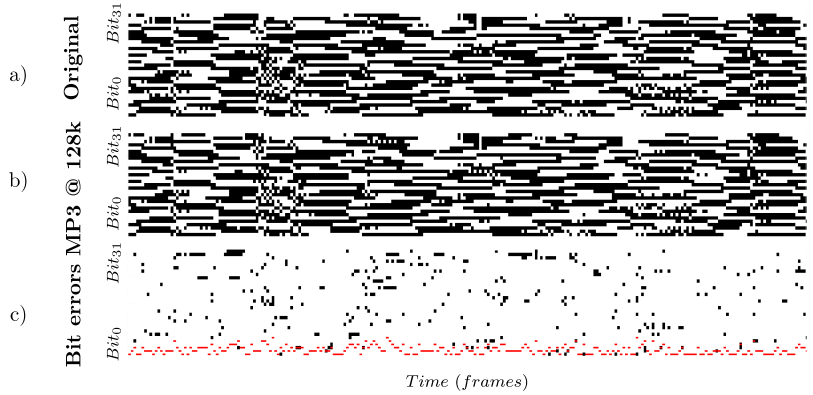



Figure 2.17: a) Fingerprint block of original music clip, (b) fingerprint block of a compressed version, (c) the difference between a and b showing the bit errors in black. The hamming distance or the number of bit errors is indicated in red.

First of all the input audio is resampled to about 5000Hz. On the resampled signal a Hamming windowed FFT with a length of 2048 samples is taken every 64 samples - an overlap of 96.7%. In the FFT output only 33 logarithmically spaced bins between $300Hz$ to $2000Hz$ in the magnitude spectrum are used. The energy of frequency band $m$ at frame index $n$ is called $E(n,m)$. Finally, a fingerprint $F(n,m)$ is constructed using the $E(n,m)$ with the following formula:

$$v = \text{E(n,m)-E(n,m+1)-(E(n-1,m)-E(n-1,m+1))}$$

$$F(n,m) = \begin{cases} 1 \text{ if } v > 0 \\ 0 \text{ if } v \leq 0 \end{cases}$$

Since the last frequency band is discarded - there is no $m+1$ for the last band - only 32 of the original 33 values remain. Every FFT frame is reduced to a 32bit word. Figure 2.17 shows a three second audio fragment comparing the original a) with a 128kb/s CBR MP3 encoded version. b) the difference between the two. c) shows the distance measure for this acoustic fingerprinting system: the number of places where the two binary words differ (in red in Figure 2.17). This distance measure is also known as the Hamming distance or the bit error rate BER.

Figure 2.18 provides a bit more insights into the BER in two cases. In the first case a high quality query, a 128 kb/s CBR encoded MP3, is compared with the reference and only a small number of bits change. Note that there are quite a few places where the BER is zero. The

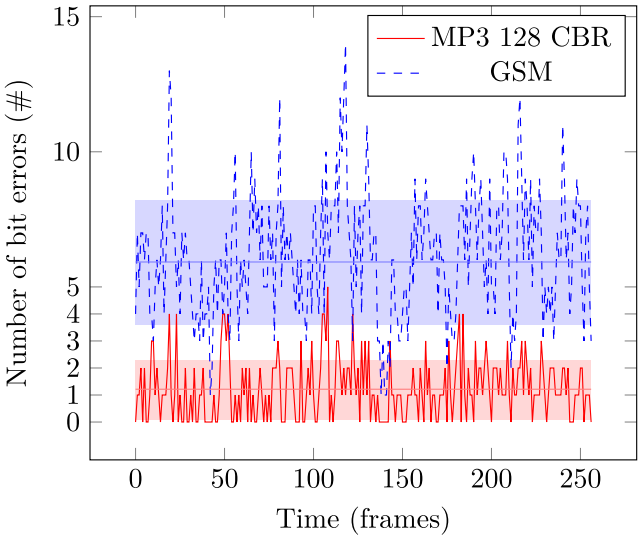



Figure 2.18: Bit errors per fingerprint for the 128kb/s CBR encoded MP3 and the GSM encoded version of the same three seconds of audio. Both are compared to the original uncompressed version. The average and standard deviation are indicated.

other case uses a low quality GSM codec. The BER, in this case, is always above zero.

**Replication** The original paper includes many details about chosen parameters. It defines an FFT size, window function and sample rate, which is a good start. Unfortunately the parameters are not consistently used throughout the paper. Twice $11.6ms$ is reported as the FFT step size and twice $11.8ms$. In the replicated implementation an $11.6ms$ step size is used ($11.6ms = 64/5500Hz$)[12]. While small changes to these parameters probably only have limited effect on the overall performance of the system it is exactly *these ambiguities that make an exact replication impossible*. The replication, in other words, includes various assumptions on the original system which may be false. However, even under these conditions the general behavior of the system may still remain intact. Results are expected to differ but only up until a certain unknown degree.

We further argue that even a detailed, consistent textual description

[12]Note that a sample rate of $5.5kHz$ is used and not, as reported in the original paper, $5kHz$, to end up at the correct step size. The follow up journal article Haitsma and Kalker (2003) does mention $5.5Hz$

of an algorithm always leaves some wiggle room for different interpretations (Peng, 2011). Only if source code is available with details on which system - software and hardware - the evaluation is done can an exact replication become feasible. The source code could also include bugs that perhaps have an effect on the results. Bugs will, by definition, not be described as such in a textual description.

This strengthens the case that source code should be an integral part of a scientific work. If interested in further details of the new implementation readers are referred to the source code in the supplementary material[13]. There, it becomes clear at which precision the FFT was calculated, how exactly downsampling was done, the precision of the windowing function, and so forth.

### Search strategy

The basic principle of the search strategy is a nearest neighbor search in Hamming space. For each fingerprint extracted from a query, a list of near neighbors is fetched which ideally includes a fingerprint from the matching audio fragment. The actual matching fragment will be present in most lists of near neighbors. A naive approach would compute the Hamming distance between a query fingerprint and each fingerprint in the reference database. This approach can be improved with algorithms that evaluate only a tiny fraction of the reference database but yield the same retrieval rates. The details of the search strategy are much less critical then the parameters of the fingerprint extraction step. As long as the nearest neighbor search algorithm is implemented correctly the only difference will be the speed at which a query is resolved.

The search strategies' main parameter is the supported Hamming distance. With an increased Hamming distance $d$ more degraded audio can be retrieved but the search space quickly explodes. For $b$ the number of bits the search space equals:

$$\sum_{k=d}^{k-2|k\geq 0} \frac{b!}{k!(b-k)!} = \sum_{k=d}^{k-2|k\geq 0} \binom{b}{k} \tag{2.11}$$

A good search strategy strikes a balance between query performance and retrieval rate. The search strategy from the original work does this by keeping track of which bits of a fingerprint are uncertain. The uncertain bits are close to the threshold. It assigns each bit with a value from 1 to 32 that describes confidence in the bit, with 1 being the least reliable bit and 32 the most reliable. Subsequently, a search is done for

[13]The implementation is also available as a part of Panako, see `http://github.com/JorenSix/Panako`

the fingerprint itself and for fingerprints which are permutations of the original with one or more uncertain bits toggled. To strike that balance between performance and retrieval rate the number of bits need to be chosen. If the three least reliable bits are toggled, this generates $2^3$ permutations. This is much less than flipping 3 bits randomly in the 32bit fingerprint:

$$\sum_{k=3}^{k-2|k\geq 0} \binom{33}{3} = 5489$$

Once a matching fingerprint is found the next step is to compare a set of fingerprints of the query with the corresponding set of fingerprints in the reference audio. The Hamming distance for each fingerprint pair is calculated. If the sum of the distances is below a threshold then it is declared a match, otherwise the search continues until either a match is found or until the query is labeled as unknown. The parameters are determined experimentally in the original work: 256 fingerprints are checked and the threshold for the Hamming distance is 2867bits. So from a total of $256 \times 32bits$, 2867 or about 35% are allowed to be different.

**Replication** The implementation is done with two hash tables. A lookup table with fingerprints as key and a list of $(identifier, offset)$ pairs as value. The identifier refers uniquely to a track in the reference database. The offset points precisely to the time at which the fingerprint appears in that track. The second hash table has an identifier as key and an array of fingerprints as value. Using the offset, the index in the fingerprint array can be determined. Subsequently, the previous 256 fingerprints from the query can be compared with the corresponding fingerprints in the reference set and a match can be verified.

Implementing this search strategy is relatively straightforward.

### 2.3.3 Evaluation

The evaluation of the system is done in two ways. First we aim to replicate the original evaluation and match the original results as closely as possible to validate the new implementation and the original work. The original evaluation is not easily replicated since it uses copyrighted evaluation audio with ambiguous descriptions, a data set that is not available or described and modifications that are only detailed up until a certain degree.

The second evaluation is fully replicable: it uses freely available evaluation audio, a data set with creative commons music and modifications that are encoded in scripts. Interested readers are encouraged to replicate the results in full.

**Replication of the original evaluation**

The evaluation of the original system is done on four short excerpts from commercially available tracks:

> *'We selected four short audio excerpts (Stereo, 44.1kHz, 16bps) from songs that belong to different musical genres: "O Fortuna" by Carl Orff, "Success has made a failure of our home" by Sinead o'Connor, "Say what you want" by Texas and "A whole lot of Rosie" by AC/DC.'*

Unfortunately it fails to mention which excerpts where used or even how long these excerpts were. The selection does have an effect on performance. If, for instance, a part with little acoustic information is selected versus a dense part different results can be expected. It also fails to mention which edition, version or release is employed which is especially problematic with the classical piece: many performances exist with varying lengths and intensities. The paper also mentions a reference database of 10 000 tracks but fails to specify which tracks it contains. The fact that only one excerpt from each song is used for evaluation makes the selection critical which is problematic by itself. Reporting an average performance with standard deviations would have been more informative.

To evaluate the robustness of the system each short excerpt is modified in various ways. The modifications to the query are described well but there is still room for improvement. For example it is not mentioned how time-scale modification is done: there are different audible artifacts - i.e. different results - when a time or frequency domain method for time-scale modification is used. The description of the echo modification seems to have been forgotten while dry, wetness or delay length parameters definitely have a large effect on sound and subsequent results.

To summarize: essential information is missing to replicate the results exactly. The next best thing is to follow the basic evaluation method which can be replicated by following various clues and assumptions. To this end the previously mentioned four tracks were bought from a digital music store (7digital, see table 2.5). Two were available in a lossless format and two in a high quality MP3 format (320 kb/s CBR). The test data set can not be freely shared since commercial music is used which, again, hinders replicability.

The original evaluation produces two tables. The first documents the bit error rates (BER, Table 2.6. It compares the fingerprints extracted from a reference recording with those of modified versions. If all bits are equal the error rate is zero. If all bits are different then the error rate is one. Comparison of random fingerprints will result in

| Identifier | Track | Format |
|---|---|---|
| 56984036 | Sinead | 320kbs MP3 |
| 52740482 | AC \DC | 16-bit/44.1kHz FLAC |
| 122965 | Texas | 320kbs MP3 |
| 5917942 | Orff | 16-bit/44.1kHz FLAC |

Table 2.5: Tracks bought from 7digital music store with 7digital identifier and format information.

a bit error rate of around 0.5. The original article suggests that 256 fingerprints (about three seconds of audio) are compared and the average is reported. Experimentally the original article determines that a BER of 0.35 or less is sufficient to claim that two excerpts are the same with only a very small chance of yielding a false positive. The BER evaluation has been replicated but due to the fact that the excerpts are not identical and the modifications also deviate slightly the replicated BER values differ. However if the original and replicated results are compared using a Pearson correlation there is a very strong linear relation $r(58) = 0.92, p < 0.001$. This analysis suggests that the system behaves similarly for the various modifications. The analysis left out the white noise condition, which is an outlier. The replicated modification probably mixed more noise into the signal than the original. Some modifications could not be successfully replicated either because they are not technically relevant (cassette tape, real media encoding) or the method to do the modification was unclear (GSM C/I).

A second table (Table 2.7) shows how many of 256 fingerprints could be retrieved in two cases. The first case tries to find only the exact matches in the database. The reported number shows how many of the 256 fingerprints point to the matching fingerprint block in the database. If all fingerprints match the maximum of 256 is reported. In the second case the 10 most unreliable bits are flipped resulting in 1024 fingerprints which are then matched with the database. In both cases only one correct hit is needed to successfully identify an audio excerpt.

The original results are compared with the replicated results with a Pearson correlation. The exact matching case shows a strong linear correlation $r(62) = 0.66, p < 0.001$ and the case of 10 flipped bits show similar results $r(62) = 0.67, p < 0.001$. This suggests that the system behaves similarly considering that the audio excerpts, the modifications and implementation include differences and various assumptions had to be made.

| | Texas | | Sinead | | Orff | | AC/DC | |
|---|---|---|---|---|---|---|---|---|
| **Modification** | **Original** | **Replication** | **Original** | **Replication** | **Original** | **Replication** | **Original** | **Replication** |
| MP3@128Kbps | 0.081 | 0.055 | 0.085 | 0.077 | 0.078 | 0.056 | 0.084 | 0.035 |
| MP3@32Kbps | 0.096 | 0.097 | 0.106 | 0.115 | 0.174 | 0.100 | 0.133 | 0.089 |
| Real@20Kbps | 0.159 | / | 0.138 | / | 0.161 | / | 0.21 | / |
| GSM | 0.168 | 0.194 | 0.144 | 0.211 | 0.16 | 0.217 | 0.181 | 0.187 |
| GSM C/I = 4dB | 0.316 | / | 0.247 | / | 0.286 | / | 0.324 | / |
| All-pass filtering | 0.018 | 0.020 | 0.015 | 0.032 | 0.019 | 0.033 | 0.027 | 0.010 |
| Amp. Compr. | 0.113 | 0.010 | 0.07 | 0.027 | 0.052 | 0.033 | 0.073 | 0.014 |
| Equalization | 0.066 | 0.025 | 0.045 | 0.024 | 0.048 | 0.023 | 0.062 | 0.013 |
| Echo Addition | 0.139 | 0.132 | 0.148 | 0.145 | 0.157 | 0.118 | 0.145 | 0.109 |
| Band Pass Filter | 0.024 | 0.031 | 0.025 | 0.034 | 0.028 | 0.030 | 0.038 | 0.017 |
| Time Scale +4% | 0.2 | 0.279 | 0.183 | 0.283 | 0.202 | 0.302 | 0.206 | 0.301 |
| Time Scale –4% | 0.19 | 0.263 | 0.174 | 0.277 | 0.207 | 0.281 | 0.203 | 0.294 |
| Linear Speed +1% | 0.132 | 0.189 | 0.102 | 0.193 | 0.172 | 0.214 | 0.238 | 0.181 |
| Linear Speed -1% | 0.26 | 0.177 | 0.142 | 0.199 | 0.243 | 0.201 | 0.196 | 0.177 |
| Linear Speed +4% | 0.355 | 0.434 | 0.467 | 0.461 | 0.438 | 0.551 | 0.472 | 0.470 |
| Linear Speed -4% | 0.47 | 0.425 | 0.438 | 0.500 | 0.464 | 0.510 | 0.431 | 0.464 |
| Noise Addition | 0.011 | 0.042 | 0.011 | 0.122 | 0.009 | 0.273 | 0.036 | 0.027 |
| Resampling | 0 | 0.000 | 0 | 0.000 | 0 | 0.004 | 0 | 0.000 |
| D/A A/D | 0.111 | / | 0.061 | / | 0.088 | / | 0.076 | / |

Table 2.6: Replication of bit error rates (BER) for different kinds of signal degradations. The original results and replicated results are reported.

| | Orff | | Sinead | | Texas | | AC/DC | |
|---|---|---|---|---|---|---|---|---|
| **Modification** | **Original** | **Replication** | **Original** | **Replication** | **Original** | **Replication** | **Original** | **Replication** |
| MP3@128Kbps | 17, 170 | 150, 226 | 20, 196 | 59, 111 | 23, 182 | 94, 166 | 19, 144 | 144, 207 |
| MP3@32Kbps | 0, 34 | 44, 123 | 10, 153 | 14, 63 | 13, 148 | 20, 56 | 5, 61 | 29, 87 |
| Real@20Kbps | 2, 7 | / | 7, 110 | / | 2, 67 | / | 1, 41 | / |
| GSM | 1, 57 | 2, 6 | 2, 95 | 0, 1 | 1, 60 | 0, 5 | 0, 31 | 4, 16 |
| GSM C/I = 4dB | 0, 3 | / | 0, 12 | / | 0, 1 | / | 0, 3 | / |
| All-pass filtering | 157, 240 | 170, 244 | 158, 256 | 161, 226 | 146, 256 | 166, 251 | 106, 219 | 191, 245 |
| Amp. Compr. | 55, 191 | 145, 222 | 59, 183 | 98, 156 | 16, 73 | 169, 247 | 44, 146 | 183, 241 |
| Equalization | 55, 203 | 161, 236 | 71, 227 | 220, 126 | 34, 172 | 126, 193 | 42, 148 | 171, 227 |
| Echo Addition | 2, 36 | 53, 70 | 12, 69 | 37, 73 | 15, 69 | 68, 112 | 4, 52 | 73, 102 |
| Band Pass Filter | 123, 225 | 169, 237 | 118, 253 | 149, 193 | 117, 255 | 110, 186 | 80, 214 | 159, 241 |
| Time Scale +4% | 6, 55 | 43, 72 | 7, 68 | 53, 54 | 16, 70 | 57, 123 | 6, 36 | 66, 118 |
| Time Scale –4% | 17, 60 | 57, 107 | 22, 77 | 53, 57 | 23, 62 | 54, 118 | 16, 44 | 60, 108 |
| Linear Speed +1% | 3, 29 | 2, 6 | 18, 170 | 2, 16 | 3, 82 | 3, 22 | 1, 16 | 8, 35 |
| Linear Speed -1% | 0, 7 | 0, 8 | 5, 88 | 2, 16 | 0, 7 | 1, 22 | 0, 8 | 4, 16 |
| Linear Speed +4% | 0, 0 | 0, 0 | 0, 0 | 0, 0 | 0, 0 | 0, 0 | 0, 1 | 0, 0 |
| Linear Speed -4% | 0, 0 | 0, 0 | 0, 0 | 0, 0 | 0, 0 | 0, 0 | 0, 0 | 0, 0 |
| Noise Addition | 190, 256 | 30, 73 | 178, 255 | 0, 9 | 179, 256 | 23, 101 | 114, 255 | 99, 167 |
| Resampling | 255, 256 | 253, 256 | 255, 256 | 239, 256 | 254, 256 | 254, 256 | 254, 256 | 253, 256 |
| D/A A/D | 15, 149 | / | 38, 229 | / | 13, 114 | / | 31, 145 | / |

Table 2.7: Replication of hits in the database for different kinds of signal degradations. First number indicates the hits for using only the 256 sub-fingerprints to generate candidate positions. Second number indicates hits when 1024 most probable candidates for every sub-fingerprint are also used

### A replicable evaluation

The original evaluation has several problems with respect to replicability. It uses commercial music but fails to mention which exact audio is used both in the reference database as for the evaluation. The process to generate modification is documented but still leaves room for interpretation. There are also other problems: The evaluation depends on the selection of only four audio excerpts.

The ideal acoustic fingerprinting system evaluation depends on the use-case. For example, the evaluation method described by Ramona et al. (2012) focuses mainly on broadcast monitoring and specific modifications that appear when broadcasting music over the radio. The SyncOccur corpus (Ramona and Peeters, 2013) also focuses on this use-case. An evaluation of an acoustic fingerprinting system for DJ-set monitoring (Sonnleitner et al., 2016) or sample identification (Van Balen et al., 2012) needs another approach. These differences in focus lead to a wide variety of evaluation techniques for systems which makes them hard to compare. The evaluation described here evaluates a fingerprint system for (re-encoded) duplicate detection with simple degradations[14]. The evaluation is fully replicable and requires only open source software. The replicable evaluation is similar to the procedure used already by Sonnleitner and Widmer (2016) and Six and Leman (2014).

The evaluation is done as follows. Using a script, available as supplementary material, 10,100 tracks creative commons licensed musical tracks are downloaded from Jamendo. Jamendo is a music sharing service. 10,000 of these tracks are then added to the reference database. The remaining 100 are not. The script provides a list of Jamendo track identifiers that are used in the reference database. Using another script, 1100 queries are selected at random[15]. The queries are divided into 1000 from the items that are in the database and 100 from items that are not present in the reference database. This is to check true negatives. A query is three seconds long and starts at 30 seconds into the song. Each query is modified automatically using the modifications described above. This modification process is also automatized with SoX, a command line audio editor. Subsequently these queries are matched with the reference database. The main parameters of the evaluation are the amount of unreliable bits to flip and the threshold when a match is accepted. The number of unreliable bits to flip was set to 10. If less than 2867 bits are different between 256 subsequent 32bits fingerprints then the match is accepted.

[14] If complex but repeatable audio degradations are needed the Audio Degradation MatLab toolbox by Mauch and Ewert (2013) can be of interest.

[15] The random function uses a fixed seed so that the evaluation can be either repeated exactly or, when given a different seed, verified with another set

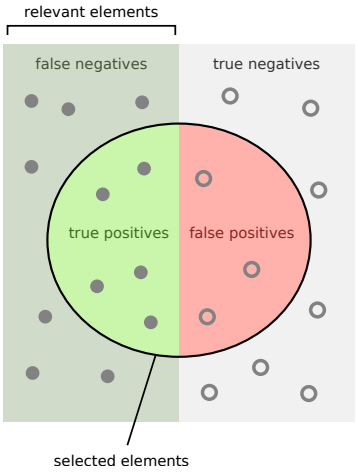



Figure 2.19: A graphical reminder on true and false positives and negatives.
Modified version of of a CC BY-SA 4.0 image by wikimedia user Walber.

| | |
|---|---|
| Sensitivity | $TP/(TP+FN)$ |
| Specificity | $TN/(TN+FP)$ |
| Precision | $TP/(TP+FP)$ |
| Accuracy | $(TP+TN)/(TP+FP+FN+TN)$ |

Table 2.8: The evaluation measures that are dependent on $TP$, $FP$, $TN$ and $FN$.

Once the results are available each query is checked if it is either a true positive $TP$, false positive $FP$, true negative $TN$ or false negative $FN$. Figure 2.19 is a graphical reminder on these measures. Next to $TP$, $FP$, $TN$ and $FN$ sensitivity, specificity, precision and accuracy are calculated as well. Table 2.8 gives the relation between these measures.

Table 2.9 summarizes the results. As expected the system's specificity and precision is very high. The few cases where a false positive is reported is due to audio duplicates in the reference database. The reference database does contain a few duplicate items where audio is either completely the same or where parts of another track are sampled. Note that the evaluation is done on the track level, the time offset is not taken into account. Since exact repetition is not uncommon, especially in electronic music, a query can be found at multiple, equally correct time offsets. If the system returns the correct track identifier with an

| | TP | TN | FP | FN | Sensitivity | Specificity | Accuracy | Precision | Avg. dist. (bits) |
|---|---|---|---|---|---|---|---|---|---|
| **MP3@128Kbps** | 90.53% | 9.18% | 0.09% | 0.19% | 99.79% | 98.99% | 99.72% | 99.90% | 4.72 ± 1.64 |
| **MP3@32Kbps** | 89.97% | 9.18% | 0.19% | 0.66% | 99.28% | 98.00% | 99.16% | 99.79% | 5.73 ± 1.72 |
| **All-pass filtering** | 90.35% | 9.18% | 0.00% | 0.47% | 99.48% | 100.00% | 99.53% | 100.00% | 5.09 ± 1.72 |
| **Amp. Compr.** | 90.44% | 9.18% | 0.00% | 0.37% | 99.59% | 100.00% | 99.63% | 100.00% | 5.26 ± 1.79 |
| **Band Pass Filter** | 90.63% | 9.18% | 0.09% | 0.09% | 99.90% | 98.99% | 99.81% | 99.90% | 5.13 ± 1.75 |
| **Echo Addition** | 86.14% | 9.27% | 0.19% | 4.40% | 95.14% | 98.02% | 95.41% | 99.78% | 7.12 ± 1.53 |
| **Equalization** | 90.63% | 9.18% | 0.00% | 0.19% | 99.79% | 100.00% | 99.81% | 100.00% | 5.25 ± 1.78 |
| **GSM** | 42.92% | 9.28% | 0.19% | 47.61% | 47.41% | 98.02% | 52.20% | 99.57% | 9.02 ± 1.17 |
| **Resampling** | 90.43% | 9.19% | 0.09% | 0.28% | 99.69% | 98.99% | 99.62% | 99.90% | 4.95 ± 1.68 |
| **Linear Speed -4%** | 0.00% | 9.27% | 0.00% | 90.73% | 0.00% | 100.00% | 9.27% | / | / |
| **Linear Speed -1%** | 75.66% | 9.27% | 0.19% | 14.89% | 83.56% | 98.02% | 84.93% | 99.75% | 8.41 ± 1.46 |
| **Linear Speed +1%** | 79.40% | 9.27% | 0.28% | 11.05% | 87.78% | 97.06% | 88.67% | 99.65% | 7.65 ± 1.41 |
| **Linear Speed +4%** | 0.00% | 9.27% | 0.00% | 90.73% | 0.00% | 100.00% | 9.27% | / | / |
| **Time Scale -4%** | 76.50% | 9.27% | 0.28% | 13.95% | 84.58% | 97.06% | 85.77% | 99.63% | 10.20 ± 0.88 |
| **Time Scale +4%** | 88.30% | 9.27% | 0.19% | 2.25% | 97.52% | 98.02% | 97.57% | 99.79% | 9.72 ± 1.00 |
| **Noise Addition** | 87.83% | 9.27% | 0.19% | 2.72% | 97.00% | 98.02% | 97.10% | 99.79% | 5.60 ± 1.98 |

Table 2.9: Results on a dataset of 10k songs with 1000 queries per modification. The average Hamming distance between a modified fingerprint of 32 bits and the matching reference is reported ± one standard deviation.

unexpected offset then it is still counted as a true positive.

The sensitivity and accuracy of the system goes down when the average bit errors per fingerprint approaches the threshold of 10 erroneous bits. True positives for GSM encoded material are only found about half of the time. The average Hamming distance in bits for queries with a changed time scale of ±4% is higher than the GSM encoded queries while accuracy is much higher. This means that for the GSM encoded material the reliability information is not reliable: the 10 least reliable bits are flipped but still the original fingerprint is not found for about half of the queries.

There are some discrepancies between these results and the reported results in the original study. The average Hamming distance between queries and reference is higher in the new evaluation. This is potentially due to the use of 128kbs MP3's during the evaluation. The original material is decoded to store in the reference database and the queries are re-encoded after modification. Another discrepancy is related to the GSM encoded queries: the original results seem to suggest that all GSM encoded queries would yield a true positive (see table 2.7). This was not achieved in the replication. Whether this is due to incorrect assumptions, different source material, the evaluation method or other causes is not clear.

### 2.3.4 Discussion

As statistical comparison showed, the replicated system behaves generally in a similar way as the originally described system. On top of that an alternative, reproducible, evaluation showed that following the system's design allows for functional acoustic fingerprinting. There are however unexplained discrepancies between both systems especially concerning the GSM modification. It is worrisome that it is impossible to pinpoint the source of these discrepancies since neither the original evaluation material, evaluation method, nor implementation are available. While there is no guarantee that the replication is bug free, at least the source can be checked.

All in all, the results are quite similar to the original. As stated in the introduction replication of results should be expected to pose no problem. It is, however, the replication of methods and accessibility of data that makes replication prohibitively time-consuming. This could be alleviated with releasing research code and data. While the focus of the MIR community should remain on producing novel techniques to deal with musical information and not on producing end-user ready software, it would be beneficial for the field to keep sustainable software aspects in mind when releasing research prototypes. Aspects such as those identified by Jackson et al. (2011) where a distinction is made be-

tween usability (documentation, installability,...) and maintainability (identity, copyright, accessibility, interoperability,...).

### 2.3.5 Conclusion

Intellectual property rights, copyrights on music and a lack of incentive pose a problem for reproducibility of MIR research work. There are, however, ways to deal with these limiting factors and foster reproducible research. We see a couple of work-arounds and possibilities, which are described below.

As universities are striving more and more for open-access publications there could be a similar movement for data and code. After all, it makes little sense to publish only part of the research in the open (the textual description) while keeping code and data behind closed doors. Especially if the research is funded by public funds. In Europe, there is an ambition to make all scientific articles freely available by 2020 and to achieve optimal reuse of scientific data[16], though research software seems to have been forgotten in this directive. A change in attitude towards releasing more software for research institutions and public funded universities is needed. A good starting point would be updating publication policies to include software together with a clear stance on **intellectual property rights**.

**Copyrights on music** make it hard to share music freely. We see two ways to deal with this:

1. *Pragmatic vs Ecological* or Jamendo *vs* iTunes. There is a great deal of freely available music published under various creative commons licenses. Jamendo for example contains half a million cc-lisenced tracks which are uniquely identifiable and can be download via an API. Much of the music that can be found there is recorded at home with limited means. It only contains a few professionally produced recordings. This means that systems can behave slightly differently on the Jamendo set when compared with a set of commercial music. What is gained in pragmatism is perhaps lost in ecological validity. Whether this is a problem depends very much on the research question at hand. In the evaluation proposed here Jamendo was used (similarly to Six and Leman (2014); Sonnleitner and Widmer (2016)) since it does offer a large variability in genres and is representative for this use-case.

2. *Audio vs Features.* Research on features extracted from audio does not need audio itself: if the features are available this can

[16] All European scientific articles to be freely accessible by 2020 - Europe makes a definitive choice for open access by 2020 - 27 May 2016 - Michiel Hendrikx

> suffice. There are two large sets of audio features. The million song data set by Bertin-Mahieux et al. (2011) and Acousticbrainz, described by Porter et al. (2015). Both ran feature extractors on millions of commercial tracks and have an API to query or download the data. Unfortunately the source of the feature extractors used in the Million Song data set are not available and only described up until a certain level of detail which makes it a black box and, in my eyes, unfit for decent reproducible science. Indeed, due to internal reorganizations and mergers the API and the data have become less and less available. The science build on the million song dataset is on shaky ground. Fortunately Acousticbrains is completely transparent. It uses well documented, open source software (Bogdanov et al., 2013) and the feature extractors are reproducible. The main shortcoming of this approach is that only a curated set of features is available. If another feature is needed, then you are out of luck. Adding a feature is far from trivial, since even Acousticbraiz has no access to all audio: they rely on crowdsourced feature extraction.

Providing an **incentive** for researchers to make their research reproducible is hard. This requires a mentality shift. Policies by journals, conference organizers and research institutions should gradually change to require reproducibility. There are a few initiatives to foster reproducible research, specifically for music informatics research. The 53rd Audio Engineering Society (AES) conference had a price for reproducibility. ISMIR 2012 had a tutorial on *"Reusable software and reproducibility in music informatics research"* but structural attention for this issue at ISMIR seems to lack. There is, however, a yearly workshop organized by Queen Mary University London (QMUL) on "Software and Data for Audio and Music Research":

> The third SoundSoftware.ac.uk one-day workshop on "Software and Data for Audio and Music Research" will include talks on issues such as robust software development for audio and music research, **reproducible research in general**, management of research data, and open access.[17]

At QMUL there seems to be continuous attention to the issue and researchers are trained in software craftsmanship[18]

In this article we problematized reproducibility in MIR and illustrated this by replicating an acoustic fingerprinting system. While similar results were obtained there are unexplained and *unexplainable dis-*

[17] `http://soundsoftware.ac.uk/soundsoftware2014` - March 2017

[18] They also host a repository for software dealing with sound at `http://soundsoftware.ac.uk`.

*crepancies* due to the fact that the original data, method and evaluation is only partly available and assumptions need to be made. We proposed an alternative, reproducible, evaluation and extrapolated general guidelines aiming to improve reproducibility of MIR research in general.

## 2.4 Applications of duplicate detection in music archives: From metadata comparison to storage optimisation
### The case of the Belgian Royal Museum for Central Africa

Six, J., Bressan, F., and Leman, M. (In press – 2018). Applications of duplicate detection in music archives: From metadata comparison to storage optimisation - The case of the Belgian Royal Museum for Central Africa

## Abstract

*This work focuses on applications of duplicate detection for managing digital music archives. It aims to make this mature music information retrieval (MIR) technology better known to archivists and provide clear suggestions on how this technology can be used in practice. More specifically applications are discussed to complement meta-data, to link or merge digital music archives, to improve listening experiences and to re-use segmentation data. To illustrate the effectiveness of the technology a case study is explored. The case study identifies duplicates in the archive of the Royal Museum for Central Africa, which mainly contains field recordings of Central Africa. Duplicate detection is done with an existing Open Source acoustic fingerprinter system. In the set, 2.5% of the recordings are duplicates. It is found that meta-data differs dramatically between original and duplicate showing that merging meta-data could improve the quality of descriptions. The case study also shows that duplicates can be identified even if recording speed is not the same for original and duplicate.*

**Keywords** MIR applications, documentation, collaboration, digital music archives

### 2.4.1 Introduction

Music Information Retrieval (MIR) technologies have a lot of untapped potential in the management of digital music archives. There seems to be several reasons for this. One is that MIR technologies are simply not well known to archivists. Another reason is that it is often unclear how MIR technology can be applied in a digital music archive setting. A third reason is that considerable effort is often needed to transform a potentially promising MIR research prototype into a working solution for archivists as end-users.

In this article we focus on duplicate detection. It is an MIR technology that has matured over the last two decades for which there is usable software available. The aim of the article is to describe several applications for duplicate detection and to encourage the communication about them to the archival community. Some of these applications might not be immediately obvious since duplicate detection is used indirectly to complement meta-data, link or merge archives, improve listening experiences and it has opportunities for segmentation. These applications are grounded in experience with working on the archive of the Royal Museum for Central Africa, a digitised audio archive of which the majority of tracks are field recordings from Central Africa.

### 2.4.2 Duplicate detection

The problem of duplicate detection is defined as follows:

> *How to design a system that is able to compare every audio fragment in a set with all other audio in the set to determine if the fragment is either unique or appears multiple times in the complete set. The comparison should be robust against various artefacts.*

The artefacts in the definition above include noise of various sources. This includes imperfections introduced during analog-to-digital (A/D) conversion. Artefacts resulting from mechanical defects, such as clicks from gramophone discs or magnetic tape hum. Detecting duplicates should be possible when changes in volume, compression or dynamics are introduced as well.

There is a distinction to be made between *exact*, *near* and *far* duplicates by Orio (2016). Exact duplicates contain the exact same information, near duplicates are two tracks with minor differences e.g. a lossless and lossy version of the same audio. Far duplicates are less straightforward. A far duplicate can be an edit where parts are added to the audio – e.g. a radio versus an album edit with a solo added. Live versions or covers of the same song can also be regarded as a far

duplicate. A song that samples an original could again be a far duplicate. In this work we focus on duplicates which contain the *same recorded material* from the original. This includes samples and edits but excludes live versions and covers.

The need for duplicate detection is there since, over time, it is almost inevitable that duplicates of the same recording end up in a digitised archive. For example, an original field recording is published on an LP, and both the LP as the original version get digitised and stored in the same lot. It is also not uncommon that an archive contains multiple copies of the same recording because the same live event was captured from two different angles (normally on the side of the parterre and from the orchestra pit), or because before the advent of digital technology, copies of degrading tapes were already being made on other tapes. Last but not least, the chance of duplicates grows exponentially when different archives or audio collections get connected or virtually merged, which is a desirable operation and one of the advantages introduced by the digital technology (see 2.4.2).

From a technical standpoint and using the terminology by Cano et al. (2005) a duplicate detector needs to have the following requirements:

- It needs to be capable to mark duplicates without generating false positives or missing true positives. In other words **precision and recall** need to be acceptable.
- It should be capable to operate on large archives. It should be **efficient**. Efficient here means quick when resolving a query and efficient on storage and memory use when building an index.
- Duplicates should be marked as such even if there is noise or the speed is not kept constant. It should be **robust** against various modifications.
- Lookup for short audio fragments should be possible, the algorithm should be **granular**. A resolution of 20 seconds or less is beneficial.

Once such system is available, several applications are possible. Orio (2016) describes many of these applications as well, but, notably, the application of re-use of segmentation boundaries is missing.

**Duplicate detection for complementing meta-data.** Being aware of duplicates is useful to **check or complement meta-data**. If an item has richer meta-data than a duplicate, the meta-data of the duplicate can be integrated. With a duplicate detection technology conflicting meta-data between an original and a duplicate can be resolved

or at least flagged. The problem of conflicting meta-data is especially prevalent in archives with ethnic music where often there are many different spellings of names, places and titles. Naming instruments systematically can also be very challenging.

**Duplicate detection to improve the listening experience.** When multiple recordings in sequence are marked as exact duplicates, meaning they contain the exact same digital information, this **indicates inefficient storage use**. If they do not contain exactly the same information it is possible that either the same analog carrier was accidentally digitised twice or there are effectively two analogue copies with the same content. To **improve the listening experience** the most qualitative digitised version can be returned if requested, or alternatively to assist philological research all the different versions (variants, witnesses of the archetype) can be returned.

**Duplicate detection for segmentation.** It potentially solves **segmentation** issues. When an LP is digitised as one long recording and the same material has already been segmented in an other digitisation effort, the segmentation boundaries can be reused. Also duplicate detection allows to identify when different segmentation boundaries are used. Perhaps an item was not segmented in one digitisation effort while a partial duplicate is split and has an extra meta-data item – e.g. an extra title. Duplicated detection allows re-use of segmentation boundaries or, at the bare minimum, indicate segmentation discrepancies.

**Duplicate detection for merging archives.** Technology makes it possible to **merge or link digital archives** from different sources – e.g. the creation of a single point of access to documentation from different institutions concerning a special subject; the implementation of the "virtual re-unification" of collections and holdings from a single original location or creator now widely scattered (IFLA - Audiovisual and Multimedia Section, 2002, p.11). More and more digital music archives 'islands' are bridged by efforts such as Europeana Sounds. Europeana Sounds is a European effort to standardise meta-data and link digital music archives. The EuropeanaConnect/DISMARC Audio Aggregation Platform provides this link and could definitely benefit from duplicate detection technology and provide a view on unique material.

If duplicates are found in one of these merged archives, all previous duplicate detection applications come into play as well. How similar is the meta-data between original and duplicate? How large is the

difference in audio quality? Are both original and duplicate segmented similarly or is there a discrepancy?

**Robustness to speed change**

Duplicate detection robust to speed changes has an important added value. When playback (or recording) speed changes from analogue carriers, both tempo and pitch change accordingly. Most people are familiar with the effect of playing a 33 rpm LP at 45 rpm. But the problem with historic archives and analogue carriers is more subtle: the speed at which the tape gets digitised might not match the original recording speed, impacting the resulting pitch. Often it is impossible to predict with reasonable precision when the recording device was defective, inadequately operated, or when the portable recorder was slowly running out of battery.

So not only it is nearly impossible to make a good estimation of the original non-standard recording speed, but it might not be a constant speed at all, it could actually fluctuate 'around' a standard speed. This is also a problem with wax cylinders, where there are numerous speed indications but they are not systematically used – if indications are present at all. In the impossibility to solve this problem with exact precision, a viable approach, balancing out technical needs and philological requirements, is normally to transfer the audio information at standard speed with state-of-the-art perfectly calibrated machinery. The precision of the A/D transfer system in a way compensates for the uncertainty of the source materials. We still obtain potentially sped-up or slowed-down versions of the recording, but when the original context in which the recording was produced can be reconstructed, it is possible to add and subtract quantities from the digitised version because that is exactly known (and its parameters ought to be documented in the preservation meta-data).If the playback speed during transfer is tampered, adapted, guessed, anything that results in a non-standard behaviour in the attempt of matching the original recording speed, will do nothing but add uncertainty to uncertainty, imprecision to imprecision.

An additional reason to digitise historical audio recordings at standard speed and with state-of-the-art perfectly calibrated machinery, is that by doing so, the archive master (IASA-TC 04, 2004) will preserve the information on the fluctuations of the original. If we are to "save history, not rewrite it" (Boston, 1998), then our desire to "improve" the quality of the recording during the process of A/D conversion should be held back. Noises and imperfections present in the source carrier bear witness to its history of transmission, and as such constitute part of the historical document. Removing or altering any of these ele-

ments violates basic philological principles (Bressan et al., 2017a) that should be assumed in any act of digitisation which has the ambition to be culturally significant. The output of a process where sources have been altered (with good or bad intention, consciously or unconsciously, intentionally or unintentionally, or without documenting the interventions) is a *corpus* that is not authentic, unreliable and for all intents and purposes useless for scientific studies. Therefore, in the light of what has been said so far, the problem of speed fluctuation is structural and endemic in historical analogue sound archives, and cannot be easily dismissed. Hence the crucial importance of algorithms that treat this type of material to consider this problem and operate accordingly.

### 2.4.3 Acoustic fingerprinting

Some possible applications of duplicate detection have been presented in the previous section, now we see how they can be put into practice. It is clear that naively comparing every audio fragment – e.g. every five seconds – with all other audio in an archive quickly becomes impractical, especially for medium-to-large size archives. Adding robustness to speed changes to this naive approach makes it downright impossible. An efficient alternative is needed and this is where *acoustic fingerprinting techniques* comes into play, a well researched MIR topic.

The aim of acoustic fingerprinting is to generate a small representation of an audio signal that can be used to reliably identify identical, or recognise similar, audio signals in a large set of reference audio. One of the main challenges is to design a system so that the reference database can grow to contain millions of entries. Over the years several efficient acoustic fingerprinting methods have been introduced (Ellis et al., 2011; Haitsma and Kalker, 2002; Orio, 2016; Wang, 2003). These methods perform well, even with degraded audio quality and with industrial sized reference databases. However, these systems are not designed to handle duplicate detection when speed is changed between the original and duplicate. For this end, fingerprinting system robust against speed changes are desired.

Some fingerprinting systems have been developed that take pitch-shifts into account (Bellettini and Mazzini, 2008; Fenet et al., 2011; Ramona and Peeters, 2013) without allowing time-scale modification. Others are designed to handle both pitch and time-scale modification (Malekesmaeili and Ward, 2013; Zhu et al., 2010). The system by Zhu et al. (2010) employs an image processing algorithm on an auditory image to counter time-scale modification and pitch-shifts. Unfortunately, the system is computationally expensive, it iterates the whole database to find a match. The system by Malekesmaeili and Ward (2013) allows extreme pitch-shifting and time-stretching, but has the same problem.

The ideas by both Six and Leman (2014); Sonnleitner and Widmer (2014) allow efficient duplicate detection robust to speed changes. The systems are built mainly with recognition of original tracks in DJ-sets in mind. Tracks used in DJ-sets are manipulated in various ways and often speed is changed as well. The problem translates almost directly to duplicate detection for archives. The respective research articles show that these systems are efficient and able to recognise audio with a $\pm 30\%$ speed change.

Only (Six and Leman, 2014) seems directly applicable in practice since it is the only system for which there is runnable software and documentation available. It can be downloaded from `http://panako.be` and has been tested with datasets containing tens of thousands of tracks on a single computer. The output is data about duplicates: which items are present more than once, together with time offsets.

The idea behind Panako is relatively simple. Audio enters the system and is transformed into a spectral representation. In the spectral domain peaks are identified. Some heuristics are used to detect only salient, identifiable peaks and ignore spectral peaks in areas with equal energy – e.g. silent parts. Once peaks are identified, these are bundled to form triplets. Valid triplets only use peaks that are near both in frequency as in time. For performance reasons a peak is also only used in a limited number of triplets. These triplets are the fingerprints that are hashed and stored and ultimately queried for matches.

Exact hashing makes lookup fast but needs to be done diligently to allow retrieval of audio with modified speed. A fingerprint together with a fingerprint extracted from the same audio but with modified speed can be seen in Figure 2.20. While absolute values regarding time change, ratios remain the same: $\frac{\Delta t_1}{\Delta t_2} = \frac{\Delta t'_1}{\Delta t'_2}$. The same holds true for the frequency ratios. This information is used in a hash. Next to the hash, the identifier of the audio is stored together with the start time of the first spectral peak.

Lookup follows a similar procedure: fingerprints are extracted and hashes are formed. Matching hashes from the database are returned and these lists are processed. If the list contains an audio identifier multiple times and the start times of the matching fingerprints align in time accounting for an optional linear scaling factor then a match is found. The linear time scaling factor is returned together with the match. An implementation of this system was used in the case study.

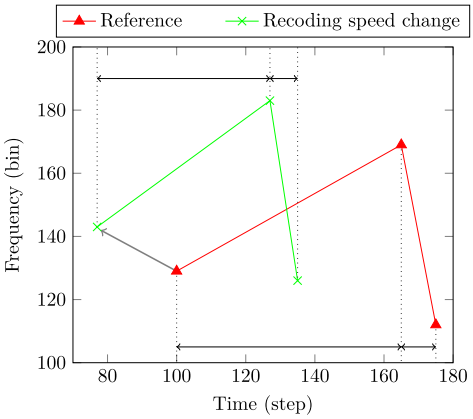



Figure 2.20: The effect of speed modification on a fingerprint. It shows a single fingerprint extracted from reference audio (red, triangle) and the same fingerprint extracted from audio after recording speed modification (green, ×).

### 2.4.4 The sound archive of the Royal Museum for Central Africa: a case study

The Royal Museum for Central Africa, Tervuren, Belgium preserves a large archive with field recordings mainly from Central Africa. The first recordings were made on wax cylinders in the late 19th century and later on all kinds of analogue carriers were used from various types of gramophone discs to sonofil. During a digitisation project called DEKKMMA (digitisation of the Ethnomusicological Sound Archive of the Royal Museum for Central Africa) (Cornelis et al., 2005) the recordings were digitised. Due to its history and size it is reasonable to expect that duplicates be found in the collection. In this case study we want to identify the duplicates, quantify the similarity in meta-data between duplicates and report the number of duplicates with modified speed. Here it is not the aim improve the data itself, this requires specialists with deep knowledge on the archive to resolve or explain (meta-data) conflicts: we mainly want to illustrate the practical use of duplicate detection.

With the Panako (Six and Leman, 2014) fingerprints of 35,306 recordings of the archive were extracted. With the default parameters of Panako this resulted in an index of 65 million fingerprints for 10 million seconds of audio or 6.5 fingerprints per second. After indexing, each recording was split into pieces of 25 seconds with 5 seconds overlap, this means a granularity of 20 seconds. Each of those pieces

(10,000,000 s / 20 s = 500,000 items) was compared with the index and resulted in a match with itself and potentially one or more duplicates. After filtering out identical matches, 4,940 fragments of 25 seconds were found to be duplicates. The duplicate fragments originated from 887 unique recordings. This means that 887 recordings (2.5%) were found to be (partial) duplicates. Thanks to the efficient algorithm, this whole process requires only modest computational power. It was performed on an Intel Core2 Quad CPU Q9650 @ 3.00GHz, with 8GB RAM, introduced in 2009.

Due to the nature of the collection, some duplicates were expected. In some cases the collection contains both the digitised version of a complete side of an analogue carrier as well as segmented recordings. Eighty duplicates could be potentially be explained in this way thanks to similarities in the recording identifier. In the collection recordings have an identifier that follows a scheme:

```
collection_name.year.collection_id.subidentifier-track
```

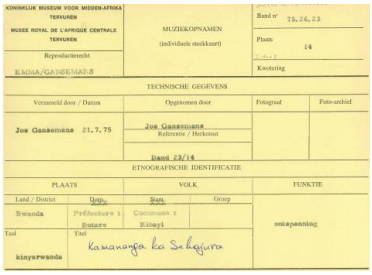
If a track identifier contains A or B it refers to a side of an analog carrier (cassette or gramophone disc). For example, the following pair of recordings `MR.1979.7.1-A1` and `MR.1979.7.1-A6` suggest that `A1` contains the complete side and A6 is track 6 on that side. The following duplicate pair suggests that the same side of a carrier has been digitised twice but stored with two identifiers: `MR.1974.23.3-A` and `MR.1974.23.3-B`. Unfortunately this means that one side is probably not digitised.

The 800 other duplicates do not have similar identifiers and lack a straightforward explanation. These duplicates must have been accumulated over the years. Potentially duplicates entered in the form of analogue copies in donated collections. It is clear that some do not originate from the same analog carrier when listening to both versions. The supplementary material contains some examples. Next, we compare the meta-data difference between original and duplicate.

### Differences in meta-data

Since the duplicates originate from the same recorded event, to original and duplicate should have identical or very similar meta-data describing their content. This is unfortunately not the case. In general, meta-data implementation depends on the history of an institution. In this case the older field-recordings are often made by priests or members of the military who did not follow a strict methodology to describe the

(a) Filing cabinet in the museum




(b) Main part of meta-data on file. Some fields use free, handwritten text (e.g. title) others a pre-defined list which are stamped (e.g. language)

Figure 2.21: Meta-data on file

musical audio and its context. Changes in geographical nomenclature over time, especially in Africa, is also a confounding factor (Cornelis et al., 2010a). There is also a large amount of vernacular names for musical instruments. The lamellophone for example is known as Kombi, Kembe, Ekembe, Ikembe Dikembe and Likembe (Cornelis et al., 2010a) to name only a few variations. On top of that, the majority of the Niger-Congo languages are tonal (Yoruba, Igbo, Ashanti, Ewe) which further limits accurate, consistent description with a western alphabet. These factors, combined with human error in transcribing and digitising information, results in an accumulation of inaccuracies. Figure 2.21 shows the physical meta-data files. If there are enough duplicates in an archive, duplicate detection can serve as a window on the quality of meta-data in general.

Table 2.10 show the results of the meta-data analysis. For every duplicate a pair of meta-data elements is retrieved and compared. They are either empty, match exactly or differ. Some pairs match quite well but not exactly. It is clear that the title of the original *O ho yi yee yi yee* is very similar to the title of the duplicate *O ho yi yee yie yee*. To capture such similarities as well, a fuzzy string match algorithm based on Sørensen–Dice coefficients is employed. When comparing the title of an original with a duplicate, only 19% match. If fuzzy matches are included 30% match. The table makes clear titles often differ while country is the most stable meta-data field. It also makes clear that the overall quality of the meta-data leaves much to improve. To correctly merge meta-data fields requires specialist knowledge - is it *yie* or *yi* - and individual inspection. This falls outside the scope of this case study.

| Field | Empty | Different | Exact match | Fuzzy or exact match |
|---|---|---|---|---|
| Identifier | 0.00% | 100.00% | 0.00% | 0.00% |
| Year | 20.83% | 13.29% | 65.88% | 65.88% |
| People | 21.17% | 17.34% | 61.49% | 64.86% |
| Country | 0.79% | 3.15% | 96.06% | 96.06% |
| Province | 55.52% | 5.63% | 38.85% | 38.85% |
| Region | 52.03% | 12.16% | 35.81% | 37.95% |
| Place | 33.45% | 16.67% | 49.89% | 55.86% |
| Language | 42.34% | 8.45% | 49.21% | 55.74% |
| Functions | 34.12% | 25.34% | 40.54% | 40.54% |
| Title | 42.23% | 38.40% | 19.37% | 30.18% |
| Collector | 10.59% | 14.08% | 75.34% | 86.71% |

Table 2.10: Comparison of pairs of meta-data fields for originals and duplicates. The field is either empty, different or exactly the same. Allowing fuzzy matching shows that fields are often similar but not exactly the same.

| Original title | Duplicate title |
|---|---|
| Warrior dance | Warriors dance |
| Amangbetu Olia | Amangbetu olya |
| Coming out of walekele | Walekele coming out |
| Nantoo | Yakubu Nantoo |
| O ho yi yee yi yee | O ho yi yee yie yee |
| Enjoy life | Gently enjoy life |
| Eshidi | Eshidi (man's name) |
| Green Sahel | The green Sahel |
| Ngolo kele | Ngolokole |

Table 2.11: Pairs of titles that match only when using a fuzzy match algorithm.

#### Speed modifications

In our dataset only very few items with modified speed have been detected. For 98.8% of the identified duplicates the speed matches exactly between original and duplicate. For the remaining 12 identified duplicates speed is changed in a limited range, from -5% to +4%. These 12 pieces must have multiple analogue carriers in the archive. Perhaps copies were made with recording equipment that was not calibrated; or if the live event was captured from multiple angles, it is possible that the calibration of the original recorders was not consistent. There is a number of reasons why a digitised archive ends up containing copies of the same content at slightly different speeds, but it is normally desirable that the cause for this depends on the attributes of the recordings *before* digitisation, and it is not introduced *during* the digitisation process. Our case study shows that duplicates can be successfully detected even when speed is modified. How this is done is explained in the following section.

### 2.4.5 De-duplication in practice

In this section, the practical functioning of Panako is described. The Panako acoustic fingerprinting suite is Java software and needs a recent Java Runtime. The Java Runtime and TarsosDSP (Six et al., 2014) are the only dependencies for the Panako system, no other software needs to be installed. Java makes the application multi-platform and compatible with most software environments. It has a command-line interface, users are expected to have a basic understanding of their command line environment.

Panako contains a deduplicate command which expects either a list of audio files or a text file that contains the full path of audio files

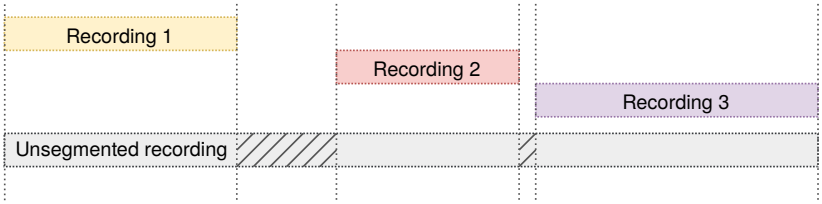



Figure 2.22: Reuse of segmentation boundaries. The recording 1, 2 and 3 are found in a long unsegmented track. The segmentation boundaries (dotted lines) can be reused. Some parts in the unsegmented track remain unlabeled (the parts with diagonal lines).

separated by newlines. This text file approach is more practical on large archives. After running the deduplicate program a text file will contain the full path of duplicate files together with the time at which the duplicate audio was detected.

Several parameters need to be set for a successful de-duplication. The main parameters determine the granularity level, allowed modifications and performance levels. The granularity level determines the size of the audio fragments that are used for de-duplication. If this is set to 20 seconds instead of 10, then the number of queries is, obviously, halved. If speed is expected to be relatively stable, a parameter can be set to limit the allowed speed change. The performance can be modified by choosing the number of fingerprints that are extracted per second. The parameters determine several trade-offs between query speed, storage size, and retrieval performance. The default parameters should have the system perform reasonably effectively in most cases.

The indirect applications of linking meta-data is dependent on organization of the meta-data of the archive but has some common aspects. First, the audio identifiers of duplicates are arranged in original/duplicate pairs. Subsequently, the meta-data of these pairs is retrieved from the meta-data store (e.g. a relational database system). Finally, the meta-data element pairs are compared and resolved. The last step can use a combination of rules to automatically merge meta-data and manual intervention when a meta-data conflict arises. The manual intervention requires analysis to determine the correct meta-data element for both original and duplicate.

Reuse of segmentation boundaries needs similar custom solutions. However, there are again some commonalities in reuse of boundaries. First, audio identifiers from the segmented set are identified within the unsegmented set resulting in a situation as in 2.22. The identified segment boundaries can subsequently be reused. Finally, segments are labeled. Since these tasks are very dependent on file formats, database

types, meta-data formats and context in general it is hard to offer a general solutions. This means that while the duplicate detection system is relatively user friendly and ready to use, applying it still needs a software developer but not, and this is crucial, an MIR specialist.

### 2.4.6 Conclusions

In this paper we described possible applications of duplicate detection techniques and presented a practical solution for duplicate detection in an archive of digitised audio of African field recordings. More specifically applications were discussed to complement meta-data, to link or merge digital music archives, to improve listening experiences and to re-use segmentation data. In the case study on the archive of the Royal Museum of Central Africa we were able to show that duplicates can be successfully identified. We have shown that the meta-data in that archive differs significantly between original and duplicate. We have also shown that duplicate detection is robust to speed variations.

The archive used in the case study is probably very similar to many other archives of historic recordings and similar results can be expected. In the case study we have shown that the acoustic fingerprinting software Panako is mature enough for practical application in the field today. We have also given practical instructions on how to use the software. It should also be clear that all music archives can benefit from this technology and we encourage archives to experiment with duplicate detection since only modest computing power is needed even for large collections.

# 3 Services

## 3.1 Introduction

This chapter bundles four articles that are placed more towards the services region of the humanities-engineering plane depicted in Figure 1.1. They offer designs and implementations of tools to support certain research tasks. The four works bundled here are:

1. 3.2 – Six, J., Cornelis, O., and Leman, M. (2014). TarsosDSP, a real-time audio processing framework in Java. In *Proceedings of the 53rd AES Conference (AES 53rd)*. The Audio Engineering Society.

2. 3.3 – Six, J. and Leman, M. (2014). Panako - A scalable acoustic fingerprinting system handling time-scale and pitch modification. In *Proceedings of the 15th ISMIR Conference (ISMIR 2014)*, pages 1–6.

3. 3.4 – Six, J. and Leman, M. (2015). Synchronizing Multimodal Recordings Using Audio-To-Audio Alignment. *Journal of Multimodal User Interfaces*, 9(3):223–229.

4. 3.5 – Six, J. and Leman, M. (2017). A framework to provide fine-grained time-dependent context for active listening experiences. In *Proceedings of the AES Conference on Semantic Audio 2017*. AES.

The first (Six et al., 2014) describes a software library which originated as a set of pitch estimation algorithms for Tarsos (Six et al., 2013). Over the years more and more audio processing algorithms were added to the library and to highlight its inherent value it was separated from Tarsos and made into a reusable component. It was presented with a poster presentation at an Audio Engineering Society conference in London. At the conference TarsosDSP was acknowledged as a 'reproducibility-enabling work' which is in my view a requirement for valuable services to research communities. TarsosDSP was picked up in research (Lv et al., 2015; Reyes et al., 2016; Sandulescu et al., 2015) and in many interactive music software projects.

The second (Six and Leman, 2014) work details a description of a novel acoustic fingerprinting algorithm. It can be placed in the services category since it offers a publicly verifiable implementation of this new

algorithm together with several baseline algorithms. Next to the implementation, a reproducible evaluation methodology is described as well. The code to run the evaluation is open as well. This can be seen as a second service to the community. This approach has been moderately successful since Panako was already used multiple times as a baseline to compare with other, newer systems (Sonnleitner et al., 2016; Sonnleitner and Widmer, 2014; Tsai, 2016). It was presented at the ISMIR conference of 2014 in Taipei, Taiwan during a poster presentation.

The third (Six and Leman, 2015) work included in this chapter is a journal article that describes a way to synchronize heterogeneous research data geared towards music and movement research. Next to the description there is also a verifiable implementation publicly available. It falls into the services category since synchronizing data before analysis is a problemetic research task many researchers deal with. The method is especially applicable for research on interaction between movement and music since this type of research needs many different measurement devices and wearable sensors that are not easily synchronized. It has been used to sync, amongst others, the dataset of Desmet et al. (2017).

Finally the fourth and last work (Six and Leman, 2017) presents meta-data to a user synchronized with music in its environment. Metadata is broadly defined as meaning any type of additional information stream that enriches the listening experience (for example lyrics, light effects or video). This technology could be employed as a service to support a research task as well: for example if during an experiment dancers need to be presented with tactical stimuli this technology could be used. Essentially it is an augmented reality technology or more generally a technology to establish a computer-mediated reality.

## 3.2 TarsosDSP, a real-time audio processing framework in Java

Six, J., Cornelis, O., and Leman, M. (2014). TarsosDSP, a real-time audio processing framework in Java. In *Proceedings of the 53rd AES Conference (AES 53rd)*. The Audio Engineering Society.

### Abstract

*This paper presents TarsosDSP, a framework for real-time audio analysis and processing. Most libraries and frameworks offer either audio analysis and feature extraction or audio synthesis and processing. TarsosDSP is one of only a few frameworks that offers both analysis, processing and feature extraction in real-time, a unique feature in the Java ecosystem.*

*The framework contains practical audio processing algorithms, it can be extended easily, and has no external dependencies. Each algorithm is implemented as simple as possible thanks to a straightforward processing pipeline. TarsosDSP's features include a resampling algorithm, onset detectors, a number of pitch estimation algorithms, a time stretch algorithm, a pitch shifting algorithm, and an algorithm to calculate the Constant-Q. The framework also allows simple audio synthesis, some audio effects, and several filters.*

*The Open Source framework is a valuable contribution to the MIR-Community and ideal fit for interactive MIR-applications on Android.*

| Name | Ref | Extr. | Synth | R-T | Tech |
|---|---|---|---|---|---|
| Aubio | Brossier (2006) | True | False | True | C |
| **CLAM** | Amatriain et al. (2002) | True | True | True | C |
| **CSL** | Pope and Ramakrishnan (2003) | True | True | True | C++ |
| Essentia | Bogdanov et al. (2013) | True | False | True | C++ |
| Marsyas | Tzanetakis and Cook (1999) | True | True | False | C++ |
| **SndObj** | Lazzarini (2001) | True | True | True | C++ |
| Sonic Visualizer | Cannam et al. (2006) | True | False | False | C++ |
| STK | Scavone and Cook (2005) | False | True | True | C++ |
| Tartini | McLeod (2009) | True | False | True | C++ |
| YAAFE | Mathieu et al. (2010) | True | False | False | C++ |
| Beads | Merz (2011) | False | True | True | Java |
| JASS | Van Den Doel and Pai (2001) | False | True | True | Java |
| jAudio | McEnnis et al. (2005) | True | False | False | Java |
| Jipes | | True | False | False | Java |
| jMusic | Brown and Sorensen (2000) | False | True | False | Java |
| JSyn | Burk (1998) | False | True | True | Java |
| Minim | Mills III et al. (2010) | False | True | True | Java |
| **TarsosDSP** | | True | True | True | Java |

Table 3.1: Notable audio processing frameworks. Only a few frameworks offer real-time feature extraction and audio synthesis capabilities. According to the research by the authors, in the Java ecosystem only TarsosDSP offers this capability.

### 3.2.1 Introduction

Frameworks or libraries[1] for audio signal processing can be divided into two categories.The first category offers audio analysis and feature extraction. The second category offers audio synthesis capabilities. Both types may or may not operate in real-time. Table 3.1 shows a partial overview of notable audio frameworks. It shows that only a few frameworks offer real-time feature extraction combined with synthesis capabilities. To the best of the authors' knowledge, TarsosDSP is unique in that regard within the Java ecosystem. The combination of real-time feature extraction and synthesis can be of use for music education tools or music video games. Especially for development on the Android platform there is a need for such functionality.

TarsosDSP also fills a need for educational tools for Music Information Retrieval. As identified by Gómez (2012), there is a need for comprehensible, well-documented MIR-frameworks which perform useful tasks on every platform, without the requirement of a costly software package like Matlab. TarsosDSP serves this educational goal, it

[1]The distinction between library and framework is explained by Amatriain (2005). In short, a framework is an abstract specification of an application whereby analysis and design is reused, conversely when using a (class) library code is reused but a library does not enforce a design.

has already been used by several master students as a starting point into music information retrieval (Benavides, 2012; Stubbe, 2013; Wager, 2011).

The framework tries to hit the sweet spot between being capable enough to get real tasks done, and compact enough to serve as a demonstration for beginning MIR-researchers on how audio processing works in practice. TarsosDSP therefore targets both students and more experienced researchers who want to make use of the implemented features.

After this introduction a section about the design decisions made follows, then the main features of TarsosDSP are highlighted. Chapter four is about the availability of the framework. The paper ends with a conclusion and future work.

### 3.2.2 Design decisions

To meet the goals stated in the introduction a couple of design decisions were made.

#### Java based

TarsosDSP was written in Java to allow portability from one platform to another. The automatic memory management facilities are a great boon for a system implemented in Java. These features allow a clean implementation of audio processing algorithms. The clutter introduced by memory management instructions, and platform dependent `ifdef`'s typically found in C++ implementations are avoided. The Dalvik Java runtime enables to run TarsosDSP's algorithms unmodified on the Android platform. Java or C++ libraries are often hard to use due to external dependencies. TarsosDSP has no external dependencies, except for the standard Java Runtime. Java does have a serious drawback, it struggles to offer a low-latency audio pipeline. If real-time low-latency is needed, the environment in which TarsosDSP operates needs to be optimized, e.g. by following the instructions by Juillerat et al. (2007).

#### Processing pipeline

The processing pipeline is kept as simple as possible. Currently, only single channel audio is allowed, which helps to makes the processing chain extremely straightforward[2]. A schematic representation can be found in Figure 3.1. The source of the audio is a file, a microphone, or an optionally empty stream. The `AudioDispatcher` chops incoming audio in blocks of a requested number of samples, with a de-

[2]Actually multichannel audio is accepted as well, but it is automatically downmixed to one channel before it is send through the processing pipeline

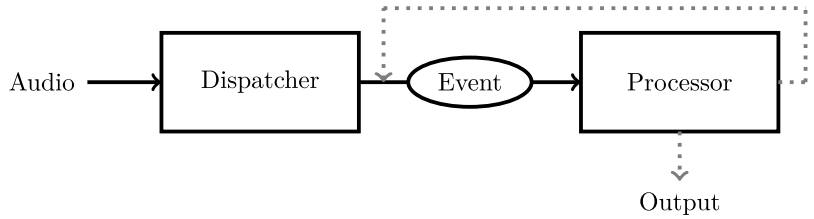



Figure 3.1: A schematic representation of the TarsosDSP processing pipeline. The incoming audio (left) is divided into blocks which are encapsulated in Event objects by the Dispatcher. The event objects flow through one or more Processor blocks, which may or may not alter the audio and can generate output (e.g. pitch estimations). Dotted lines represent optional flows.

fined overlap. Subsequently the blocks of audio are scaled to a `float` in the range `[-1,1]`. The wrapped blocks are encapsulated in an `AudioEvent` object which contains a pointer to the audio, the start time in seconds, and has some auxiliary methods, e.g. to calculate the energy of the audio block. The `AudioDispatcher` sends the `AudioEvent` through a series of `AudioProcessor` objects, which execute an operation on audio. The core of the algorithms are contained in these `AudioProcessor` objects. They can e.g. estimate pitch or detect onsets in a block of audio. Note that the size of a block of audio can change during the processing flow. This is the case when a block of audio is stretched in time. For more examples of available `AudioProcessor` operations see section 3.2.2. Figure 3.2 shows a processing pipeline. It shows how the dispatcher chops up audio and how the `AudioProcessor` objects are linked. Also interesting to note is line 8, where an anonymous inner class is declared to handle pitch estimation results. The example covers filtering, analysis, effects and playback. The last statement on line 23 bootstraps the whole process.

### Optimizations

TarsosDSP serves an educational goal, therefore the implementations of the algorithms are kept as pure as possible, and no obfuscating optimizations are made. Readability of the source code is put before its execution speed, if algorithms are not quick enough users are invited to optimize the Java code themselves, or look for alternatives, perhaps in another programming language like C++. This is a rather unique feature of the TarsosDSP framework, other libraries take a different approach. jAudio (McEnnis et al., 2005) and YAAFE (Mathieu et al.,

```
01: //Get an audio stream from the microphone, chop it in blocks
02: // of 1024 samples, no overlap (0 samples)
03: AudioDispatcher d = AudioDispatcher.fromDefaultMicrophone(1024, 0);
04: float sampleRate = 44100;//The sample rate
05: //High pass filter, let everything pass above 110Hz
06: AudioProcessor highPass = new HighPass(110,sampleRate);
07: d.addAudioProcessor(highPass);
08: //Pitch detection, print estimated pitches on standard out
09: PitchDetectionHandler printPitch = new PitchDetectionHandler() {
10:         @Override
11:         public void handlePitch(PitchDetectionResult result,
12:                 AudioEvent event) {
13:                 System.out.println(result.getPitch());
14:         }
15: };
16: PitchEstimationAlgorithm algo = PitchEstimationAlgorithm.YIN; //use YIN
17: AudioProcessor pitchEstimator = new PitchProcessor(algo, sr,1024,printPitch);
18: d.addAudioProcessor(pitchEstimator);
19: //Add an audio effect (delay)
20: d.addAudioProcessor(new DelayEffect(0.5,0.3,sr));
21: //Mix some noise with the audio (synthesis)
22: d.addAudioProcessor(new NoiseGenerator(0.3));
23: //Play the audio on the loudspeakers
24: d.addAudioProcessor(new AudioPlayer(
25:                 new AudioFormat(sampleRate, 16, 1, true,true)));
26: d.run();//starts the dispatching process
```

Figure 3.2: A TarsosDSP processing PipeLine. Here, pitch estimation on filtered audio from a microphone sample session is done in real-time. A delay audio effect is added and some noise is added to the audio before it is played back. The example covers filtering, analysis, audio effects, synthesis and playback.

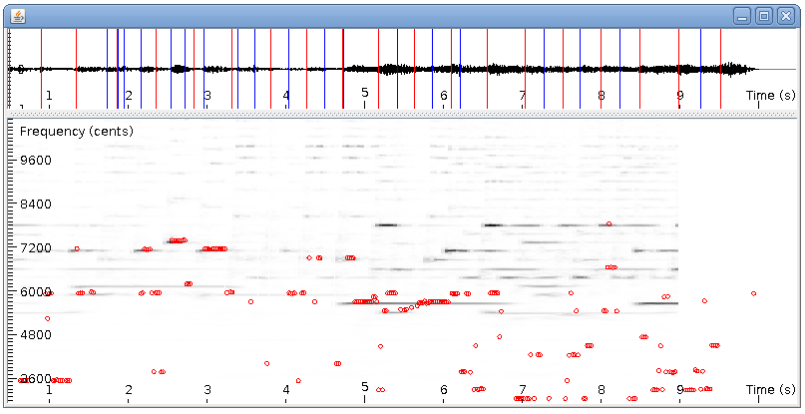



Figure 3.3: A visualization of some of the features that can be extracted using TarsosDSP: a waveform (top panel, black), onsets (top panel, blue), beats (top panel, red), a Constant-Q spectrogram (bottom panel, gray), and pitch estimations (bottom panel, red). The source code for the visualisation is part of the TarsosDSP distribution as well.

2010) for example reuse calculations for feature extraction, this makes algorithms more efficient, but also harder to grasp. Other libraries still, like SoundTouch[3], carry a burden by being highly optimized - with assembler code - and by having a large history. These things tend to contribute to less readable code, especially for people new in the field.

### Implemented Features

In this chapter the main implemented features are highlighted. Next to the list below, there are boiler-plate features e.g. to adjust gain, write a wav-file, detect silence, following envelope, playback audio. Figure 3.3 shows a visualization of several features computed with TarsosDSP.

- TarsosDSP was originally conceived as a library for pitch estimation: it contains several pitch estimators: YIN (de Cheveigné and Hideki, 2002), MPM (McLeod and Wyvill, 2005), AMDF (Ross et al., 1974)[4], and an estimator based on dynamic wavelets (Larson and Maddox, 2005). There are two YIN implementations, one remains within the comforts of the time domain, the other calculates convolution in the frequency domain[5].

- Two onset detectors are provided. One described in (Barry et al., 2005), and the one used by the BeatRoot system (Dixon, 2001).

- The WSOLA time stretch algorithm (Verhelst and Roelands, 1993), which allows to alter the speed of an audio stream without altering the pitch is included. On moderate time stretch factors - 80%-120% of the original speed - only limited audible artifacts are noticeable.

- A resampling algorithm based on (Smith and Gosset, 1984) and the related open source resample software package[6].

- A pitch shifting algorithm, which allows to change the pitch of audio without affecting speed, is formed by chaining the time-stretch algorithm with the resample algorithm.

- As examples of audio effects, TarsosDSP contains a delay and flanger effect. Both are implemented as minimalistic as possible.

[3] `http://www.surina.net/soundtouch/` SoundTouch, by Olli Parviainen, is an open-source audio processing library.

[4] Partial implementation provided by Eder Souza

[5] The YIN FFT implementation was kindly contributed by Matthias Mauch

[6] Resample 1.8.1 can be found on the digital audio resampling home page and is maintained by Julius O. Smith `https://ccrma.stanford.edu/~jos/resample/`,

- Several IIR-filters are included. A single pass and four stage low pass filter, a high pass filter, and a band pass filter.

- TarsosDSP also allows audio synthesis and includes generators for sine waves and noise. Also included is a Low Frequency Oscillator (LFO) to control the amplitude of the resulting audio.

- A spectrum can be calculated with the inevitable FFT or using the provided implementation of the Constant-Q (Brown and Puckette, 1992) transform.

### 3.2.3 Example applications

To show the capabilities of the framework, seventeen examples are built. Most examples are small programmes with a simple user interface, showcasing one algorithm. They do not only show which functionality is present in the framework, but also how to use those in other applications. There are example applications for time stretching, pitch shifting, pitch estimation, onset detection, and so forth. Figure 3.4 shows an example application, featuring the pitch shifting algorithm.

TarsosDSP is used by Tarsos (Six et al., 2013) a software tool to analyze and experiment with pitch organization in non-western music. It is an end-user application with a graphical user interface that leverages a lot of TarsosDSP's features. It can be seen as a showcase for the framework.

### 3.2.4 Availability and license

The source code is available under the GPL license terms at GitHub: `https://github.com/JorenSix/TarsosDSP`.
Contributions are more than welcome. TarsosDSP releases, the manual, and documentation can all be found at the release directory which is available at the following url:
`http://0110.be/releases/TarsosDSP/`.
Nightly builds can be found there as well. Other downloads, documentation on the example applications and background information is available on:
`http://0110.be`
Providing the source code under the GPL license makes sure that derivative works also need to provide the source code, which enables reproducibility.

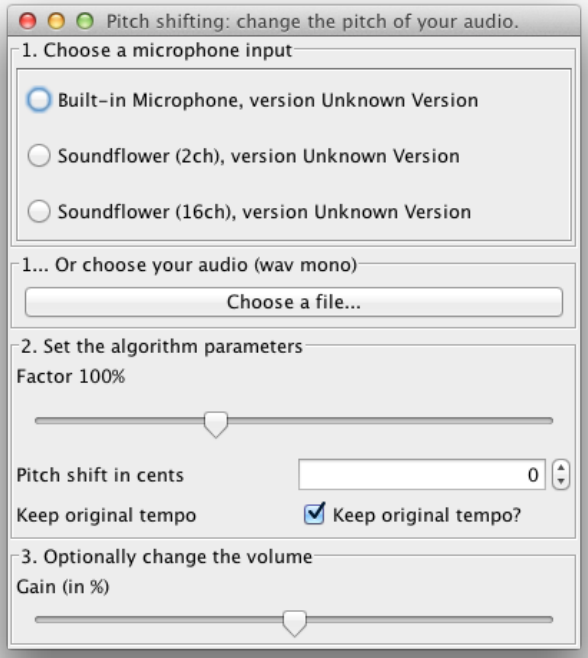



Figure 3.4: A TarsosDSP example application. Most algorithms implemented in TarsosDSP have a demo application with a user interface. Here, the capabilities of a pitch shifting algorithm are shown.

### 3.2.5 Conclusion

In this paper TarsosDSP was presented. An Open Source Java library for real time audio processing without external dependencies. It allows real-time pitch and onset extraction, a unique feature in the Java ecosystem. It also contains algorithms for time stretching, pitch shifting, filtering, resampling, effects, and synthesis. TarsosDSP serves an educational goal, therefore algorithms are implemented as simple and self-contained as possible using a straightforward pipeline. The library can be used on the Android platform, as a back-end for Java applications or stand alone, by using one of the provided example applications. After two years of active development it has become a valuable addition to the MIR-community.

## 3.3 Panako - a scalable acoustic fingerprinting system handling time-scale and pitch modification

Six, J. and Leman, M. (2014). Panako - A scalable acoustic fingerprinting system handling time-scale and pitch modification. In *Proceedings of the 15th ISMIR Conference (ISMIR 2014)*, pages 1–6.

### Abstract

*This paper presents a scalable granular acoustic fingerprinting system. An acoustic fingerprinting system uses condensed representation of audio signals, acoustic fingerprints, to identify short audio fragments in large audio databases. A robust fingerprinting system generates similar fingerprints for perceptually similar audio signals. The system presented here is designed to handle time-scale and pitch modifications. The open source implementation of the system is called Panako and is evaluated on commodity hardware using a freely available reference database with fingerprints of over 30,000 songs. The results show that the system responds quickly and reliably on queries, while handling time-scale and pitch modifications of up to ten percent.*

*The system is also shown to handle GSM-compression, several audio effects and band-pass filtering. After a query, the system returns the start time in the reference audio and how much the query has been pitch-shifted or time-stretched with respect to the reference audio. The design of the system that offers this combination of features is the main contribution of this paper.*

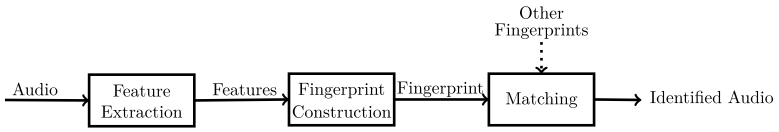



Figure 3.5: **A generalized audio fingerprinter scheme**. Audio is fed into the system, features are extracted and fingerprints constructed. The fingerprints are consecutively compared with a database containing the fingerprints of the reference audio. The original audio is either identified or, if no match is found, labeled as unknown.

### 3.3.1 Introduction

The ability to identify a small piece of audio by comparing it with a large reference audio database has many practical use cases. This is generally known as *audio fingerprinting* or *acoustic fingerprinting*. An acousic fingerprint is a condensed representation of an audio signal that can be used to reliably identify identical, or recognize similar, audio signals in a large set of reference audio. The general process of an acoustic fingerprinting system is depicted in Figure 3.5. Ideally, a fingerprinting system only needs a short audio fragment to find a match in large set of reference audio. One of the challenges is to design a system in a way that the reference database can grow to contain millions of entries. Another challenge is that a robust fingerprinting should handle noise and other modifications well, while limiting the amount of false positives and processing time (Cano et al., 2005). These modifications typically include dynamic range compression, equalization, added background noise and artifacts introduced by audio coders or A/D-D/A conversions.

Over the years several efficient acoustic fingerprinting methods have been introduced (Allamanche, 2001; Ellis et al., 2011; Haitsma and Kalker, 2002; Wang, 2003). These methods perform well, even with degraded audio quality and with industrial sized reference databases. However, these systems are not designed to handle queries with modified time-scale or pitch although these distortions can be present in replayed material. Changes in replay speed can occur either by accident during an analog to digital conversion or they are introduced deliberately.

Accidental replay speed changes can occur when working with physical, analogue media. Large music archive often consist of wax cylinders, magnetic tapes and gramophone records. These media are sometimes digitized using an incorrect or varying playback speed. Even when cali-

brated mechanical devices are used in a digitization process, the media could already have been recorded at an undesirable or undocumented speed. A fingerprinting system should therefore allow changes in replay speed to correctly detect duplicates in such music archives.

Deliberate time-scale manipulations are sometimes introduced as well. During radio broadcasts, for example, songs are occasionally played a bit faster to make them fit into a time slot. During a DJ-set pitch-shifting and time-stretching are present almost continuously. To correctly identify audio in these cases as well, a fingerprinting system robust against pitch-shifting and time-stretching is desired.

Some fingerprinting systems have been developed that take pitch-shifts into account (Bellettini and Mazzini, 2008; Fenet et al., 2011; Ramona and Peeters, 2013) without allowing time-scale modification. Others are designed to handle both pitch and time-scale modification (Malekesmaeili and Ward, 2013; Zhu et al., 2010). The system by Zhu et al. (2010) employs an image processing algorithm on an auditory image to counter time-scale modification and pitch-shifts. Unfortunately, the system is computationally expensive, it iterates the whole database to find a match. The system by Malekesmaeili and Ward (2013) allows extreme pitch- shifting and time-stretching, but has the same problem. To the best of our knowledge, a description of a practical acoustic fingerprinting system that allows substantial pitch-shift and time-scale modification is nowhere to be found in the literature. This description is the main contribution of this paper.

### 3.3.2 Method

The proposed method is inspired by three works. Combining key components of those works results in a design of a granular acoustic fingerprinter that is robust to noise and substantial compression, has a scalable method for fingerprint storage and matching, and allows time-scale modification and pitch-shifting.

Firstly, the method used by Wang (2003) establishes that local maxima in a time-frequency representation can be used to construct fingerprints that are *robust to quantization effects, filtering, noise and substantial compression.* The described exact-hashing method for *storing and matching fingerprints has proven to be very scalable.* Secondly, Arzt et al. (2012) describe a method to align performances and scores. Especially interesting is the way how triplets of events are used to search for performances with different timings. Thirdly, The method by Fenet et al. (2011) introduces the idea to extract fingerprints from a Constant-Q (Brown and Puckette, 1992) transform, a time-frequency representation that has a constant amount of bins for every octave. In their system a *a fingerprint remains constant when a pitch-shift oc-*

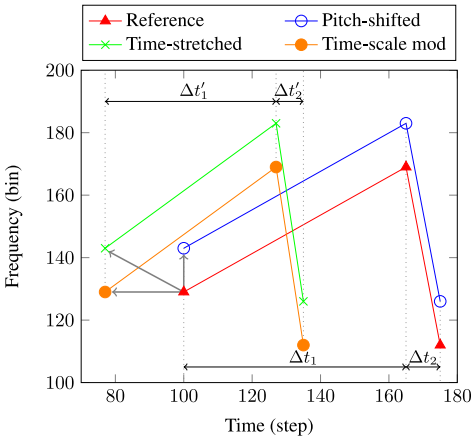



Figure 3.6: The effect of time-scale and pitch modifications on a fingerprint. It shows a single fingerprint extracted from reference audio (red, triangle) and the same fingerprint extracted from audio after pitch-shifting (blue, clear circle), time-stretching (orange, full circle) and time-scale modification (green, x).

*curs*. However, since time is encoded directly within the fingerprint, the method does not allow time-scale modification.

Considering previous works, the method presented here uses local maxima in a spectral representation. It combines three event points, and takes time ratios to form time-scale invariant fingerprints. It leverages the Constant-Q transform, and only stores frequency differences for pitch-shift invariance. The fingerprints are designed with an exact hashing matching algorithm in mind. Below each aspect is detailed.

### Finding local maxima

Suppose a time-frequency representation of a signal is provided. To locate the points where energy reaches a local maximum, a tiled two-dimensional peak picking algorithm is applied. First the local maxima for each spectral analysis frame are identified. Next each of the local maxima are iterated and put in the center of a tile with $\Delta T \times \Delta F$ as dimensions. If the local maximum is also the maximum within the tile it is kept, otherwise it is discarded. Thus, making sure only one point is identified for every tile of $\Delta T \times \Delta F$. This approach is similar to the ones by Fenet et al. (2011); Wang (2003). This results in a list of event points each with a frequency component $f$, expressed in bins, and a

time component $t$, expressed in time steps. $\Delta T$ and $\Delta F$ are chosen so that there are between 24 and 60 event points every second.

A spectral representation of an audio signal has a certain granularity; it is essentially a grid with bins both in time as in frequency. When an audio signal is modified, the energy that was originally located in one single bin can be smeared over two or more bins. This poses a problem, since the goal is to be able to locate event points with maximum energy reliably. To improve reliability, a post processing step is done to refine the location of each event point by taking its energy and mixing it with the energy of the surrounding bins. The same thing is done for the surrounding bins. If a new maximum is found in the surroundings of the initial event point, the event point is relocated accordingly. Effectively, a rectangular blur with a $3 \times 3$ kernel is applied at each event point and its surrounding bins.

Once the event points with local maximum energy are identified, the next step is to combine them to form a fingerprint. A fingerprint consists of three event points, as seen in Figure 3.6. To construct a fingerprint, each event point is combined with two nearby event points. Each event point can be part of multiple fingerprints. Only between 8 and 20 fingerprints are kept every second. Fingerprints with event points with the least cumulative energy are discarded. Now that a list of fingerprints has been created a method to encode time information in a fingerprint hash is needed.

**Handling time stretching: event triplets**

Figure 3.6 shows the effect of time stretching on points in the time-frequency domain. There, a fingerprint extracted from reference audio (Fig.3.6, red, triangle) is compared with a fingerprint from time stretched audio (Fig.3.6, orange, full circle). Both fingerprints are constructed using three local maxima $e_1, e_2, e_3$ and $e'_1, e'_2, e'_3$ . While the frequency components stay the same, the time components do change. However, the ratios between the time differences are constant as well. The following equation holds[7]:

$$\frac{t_2 - t_1}{t_3 - t_1} = \frac{t'_2 - t'_1}{t'_3 - t'_1} \tag{3.1}$$

With event point $e_n$ having a time and frequency component $(t_n, f_n)$ and the corresponding event points $e'_n$ having the components $(t'_n, f'_n)$. Since $t_3 - t_1 \geq t_2 - t_1$, the ratio always resolves to a number in the range $]0, 1]$. This number, scaled and rounded, is a component of the eventual fingerprint hash (an approach similar to Arzt et al. (2012)).

[7] It is assumed that the time stretch factor is constant in the time interval $t'_3 - t'_1$. A reasonable assumption since $t'_3 - t'_1$ is small.

Now that a way to encode time information, indifferent of time-stretching, has been found, a method to encode frequency, indifferent to pitch-shifting is desired.

### Handling pitch-shifts: constant-q transform

Figure 3.6 shows a comparison between a fingerprint from pitch shifted audio (blue, clear circle) with a fingerprint from reference audio (red, triangle). In the time-frequency domain pitch shifting is a vertical translation and time information is preserved. Since every octave has the same number of bins (Brown and Puckette, 1992) a pitch shift on event $e_1$ will have the following effect on it's frequency component $f_1$, with $K$ being a constant, $f_1' = f_1 + K$. It is clear that the difference between the frequency components remains the same, before and after pitch shifting: $f_1 - f_2 = (f_1' + K) - (f_2' + K)$ (Fenet et al., 2011). In the proposed system three event points are available, the following information is stored in the fingerprint hash: $f_1 - f_2; f_2 - f_3; \tilde{f}_1; \tilde{f}_3$

The last two elements, $\tilde{f}_1$ and $\tilde{f}_3$ are sufficiently coarse locations of the first and third frequency component. They are determined by the index of the frequency band they fall into after dividing the spectrum into eight bands. They provide the hash with more discriminative power but also limit how much the audio can be pitch-shifted, while maintaining the same fingerprint hash.

### Handling time-scale modification

Figure 3.6 compares a fingerprint of reference audio (Fig.3.6, red, triangle) with a fingerprint from the same audio that has been sped up (Fig.3.6, green, x). The figure makes clear that speed change is a combination of both time-stretching and pitch-shifting. Since both are handled in with the previous measures, no extra precautions need to be taken. The next step is to combine the properties into a fingerprint that is efficient to store and match.

### Fingerprint hash

A fingerprint with a corresponding hash needs to be constructed carefully to maintain aforementioned properties. The result of a query should report the amount of pitch-shift and time-stretching that occurred. To that end, the absolute value of $f_1$ and $t_3 - t_1$ is stored, they can be used to compare with $f_1'$ and $t_3' - t_1'$ from the query. The time offset at which a match was found should be returned as well, so $t_1$ needs to be stored. The complete information to store for each fingerprint is:

$$\left(f_1 - f_2; f_2 - f_3; \tilde{f}_1; \tilde{f}_3; \frac{t_2 - t_1}{t_3 - t_1}\right); t_1; f_1; t_3 - t_1; id \tag{3.2}$$

The hash, the first element between brackets, can be packed into a $32bit$ integer. To save space, $f_1$ and $t_3 - t_1$ can be combined in one $32bit$ integer. An integer of $32bit$ is also used to store $t_1$. The reference audio identifier is also a $32bit$ identifier. A complete fingerprint consists of $4 \times 32bit = 128bit$. At eight fingerprints per second a song of four minutes is reduced to $128bit \times 8 \times 60 \times 4 = 30kB$. An industrial size data set of one million songs translates to a manageable $28GB$[8].

### Matching algorithm

The matching algorithm is inspired by Wang (2003), but is heavily modified to allow time stretched and pitch-shifted matches. It follows the scheme in Figure 3.5 and has seven steps.

1. Local maxima are extracted from a constant-Q spectrogram from the query. The local maxima are combined by three to form fingerprints, as explained in Sections 3.3.2, 3.3.2 and 3.3.2.

2. For each fingerprint a corresponding hash value is calculated, as explained in Section 3.3.2.

3. The set of hashes is matched with the hashes stored in the reference database, and each exact match is returned.

4. The matches are iterated while counting how many times each individual audio identifier occurs in the result set.

5. Matches with an audio identifier count lower than a certain threshold are removed, effectively dismissing random chance hits. In practice there is almost always only one item with a lot of matches, the rest being random chance hits. A threshold of three or four suffices.

6. The residual matches are checked for alignment, both in frequency and time, with the reference fingerprints using the information that is stored along with the hash.

7. A list of audio identifiers is returned ordered by the amount of fingerprints that align both in pitch and frequency.

[8]Depending on the storage engine used, storage of fingerprints together with an index of sorts introduces a storage overhead. Since the data to store is small, the index can be relatively large.

In step six, frequency alignment is checked by comparing the $f_1$ component of the stored reference with $f'_1$, the frequency component of the query. If, for each match, the difference between $f_1$ and $f'_1$ is constant, the matches align.

Alignment in time is checked using the reference time information $t_1$ and $t_3 - t_1$, and the time information of the corresponding fingerprint extracted from the query fragment $t'_1$, $t'_3 - t'_1$. For each matching fingerprint the time offset $t_o$ is calculated. The time offset $t_o$ resolves to the amount of time steps between the beginning of the query and the beginning of the reference audio, even if a time modification took place. It stands to reason that $t_o$ is constant for matching audio.

$$t_o = t_1 - t'_1 \times \frac{(t_3 - t_1)}{(t'_3 - t'_1)} \tag{3.3}$$

The matching algorithm also provides information about the query. The time offset tells at which point in time the query starts in the reference audio. The time difference ratio $(t_3 - t_1)/(t'_3 - t'_1)$ represents how much time is modified, in percentages. How much the query is pitch-shifted with respect to the reference audio can be deduced from $f'_1 - f_1$, in frequency bins. To convert a difference in frequency bins to a percentage the following equation is used, with $n$ the number of cents per bin, $e$ Eulers number, and $ln$ the natural logarithm: $e^{((f'_1 - f_1) \times n \times ln(2)/1200)}$

The matching algorithm ensures that random chance hits are very uncommon, the number of false positives can be effectively reduced to zero by setting a threshold on the number of aligned matches. The matching algorithm also provides the query time offset and the percentage of pitch-shift and time-scale modification of the query with respect to the reference audio.

### 3.3.3 Results

To test the system, it was implemented in the Java programming language. The implementation is called Panako and is available under the *GNU Affero General Public License* on `http://panako.be`. The DSP is also done in Java using a DSP library by Six et al. (2014). To store and retrieve hashes, Panako uses a key-value store. Kyoto Cabinet, BerkeyDB, Redis, LevelDB, RocksDB, Voldemort, and MapDB were considered. MapDB is an implementation of a storage backed B-Tree with efficient concurrent operations (Lehman and Yao, 1981) and was chosen for its simplicity, performance and good Java integration. Also, the storage overhead introduced when storing fingerprints on disk is minimal. Panako is compared with Audfprint by Dan Ellis, an implementation of a fingerprinter system based on Wang (2003).

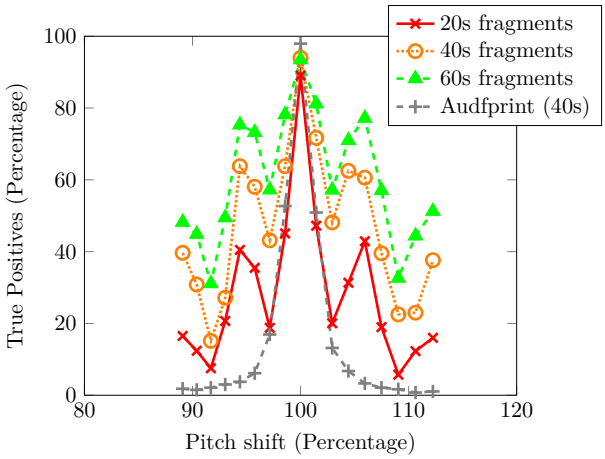



Figure 3.7: True positive rate after pitch-shifting. Note the fluctuating effect caused by the Constant-Q frequency bins.

The test data set consists of freely available music downloaded from Jamendo[9]. A reference database of about 30,000 songs, about $10^6$ seconds of audio, was created. From this data set random fragments were selected, with a length of 20, 40 and 60 seconds. Each fragments was modified 54 times. The modifications included: pitch-shifting ($-200$ tot 200 cents in steps of 25 cents), time-stretching ($-16\%$ to $+16\%$, in steps of 2%), time-scale modification ($-16\%$ to $+16\%$, in steps of 2%), echo, flanger, chorus and a band-pass filter[10]. Another set of fragments were created from audio not present in the reference database, in order to measure the number of correctly unidentified fragments. In total $3(durations) \times 600(excerpts) \times 54(modifications) = 97,200$ fragments were created.

Each fragment is presented to both Panako and Audfprint and the detection results are recorded. The systems are regarded as binary classifiers of which the amount of true positives ($TP$), false positives ($FP$), true negatives ($TN$) and false negatives ($FN$) are counted. During the experiment with Panako no false positives ($FP$) were detected. Also, all fragments that are not present in the reference database were rejected

[9] `http://jamendo.com` is a website where artists share their work freely, under various creative commons licenses. To download the data set used in this paper, and repeat the experiment, please use the scripts provided at `http://panako.be`.

[10] The effects were applied using SoX, a command line audio editor. The scripts used to generate the queries can be found at the website `http://panako.be`

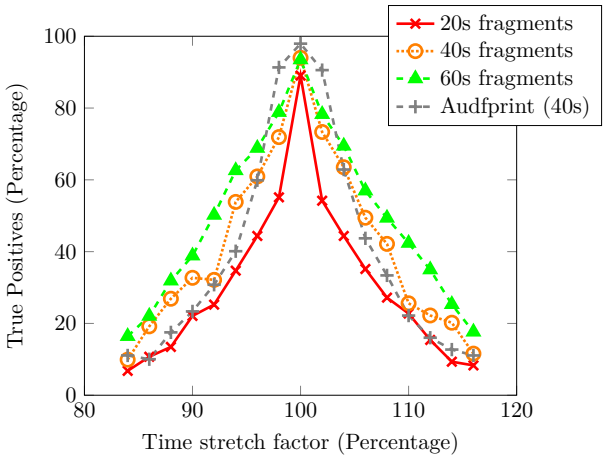



Figure 3.8: The true positive rate after time-stretching.

correctly ($TN$). So Panako's specificity is $TN/(TN + FP) = 100\%$. This can be explained by the design of the matching algorithm. A match is identified as such if a number of hashes, each consisting of three points in a spectrogram, align in time. A random match between hashes is rare, the chances of a random match between consecutively aligned hashes is almost non-existent, resulting in 100% specificity.

The sensitivity $TP/(TP + FN)$[11] of the system, however, depends on the type of modification on the fragment. Figure 3.7 shows the results after pitch-shifting. It is clear that the amount of pitch-shift affects the performance, but in a fluctuating pattern. The effect can be explained by taking into account the Constant-Q bins. Here, a bin spans 33 cents, a shift of $n \times 33/2$ cents spreads spectral information over two bins, if $n$ is an odd number. So performance is expected to degrade severely at $\pm 49.5$ cents (3%) and $\pm 148.5$ cents (9%) an effect clearly visible in figure 3.7. The figure also shows that performance is better if longer fragments are presented to the system. The performance of Audfprint, however, does not recover after pitch-shifts of more than three percent.

Figure 3.8 shows the results after time stretching. Due to the granularity of the time bins, and considering that the step size stays the

[11] Erratum: The sensitivy was previously mistakenly defined as $FP/(TP + FN)$. An error that slipped through peer review.

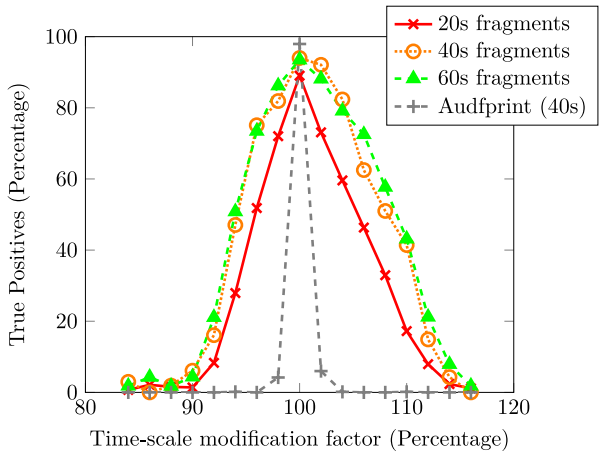



Figure 3.9: True positive rate after time-scale modification

same for each query type, time modifications have a negative effect on the performance. Still, a more than a third of the queries is resolved correctly after a time stretching modification of 8%. Performance improves with the length of a fragment. Surprisingly, Audfprint is rather robust against time-stretching, thanks to the way time is encoded into a fingerprint.

Figure 3.9 shows the results after time-scale modification. The performance decreases severely above eight percent. The figure shows that there is some improvement when comparing the results of 20s fragments to 40s fragments, but going from 40s to 60s does not change much. Audiofprint is unable to cope with time-scale modification due to the changes in both frequency and time.

In Figure 3.10, the results for other modifications like echo, chorus, flanger, tremolo, and a band pass filter can be seen. The parameters of each effect are chosen to represent typical use, but on the heavy side. For example the echo effect applied has a delay line of 0.5 seconds and a decay of 30%. The system has the most problems with the chorus effect. Chorus has a blurring effect on a spectrogram, which makes it hard for the system to find matches. Still it can be said that the algorithm is rather robust against very present, clearly audible, commonly used audio effects. The result of the band pass filter with a center of 2000Hz is especially good. To test the systems robustness to severe audio compression a test was executed with GSM-compressed

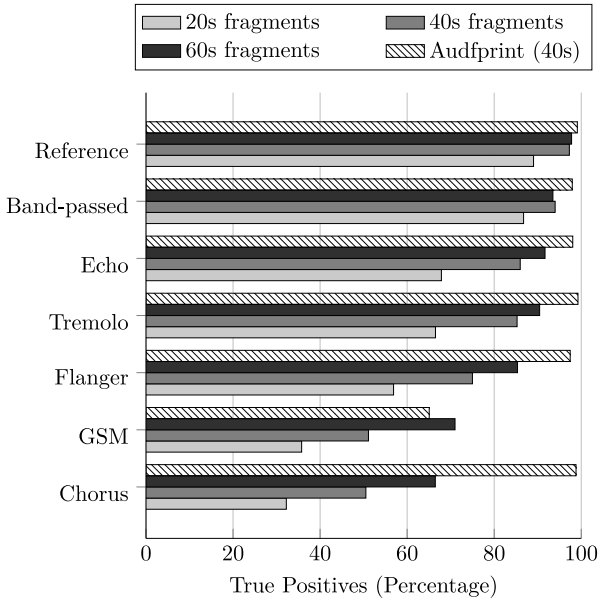



Figure 3.10: Effect of several attacks on true positive rate.

queries. The performance on 20s fragments is about 30% but improves a lot with query length, the 60s fragment yields 65%. The results for Audfprint show that there is room for improvement for the performance of Panako.

A practical fingerprinting system performs well, in terms of speed, on commodity hardware. With Panako extracting and storing fingerprints for 25s of audio is done in one second using a single core of a dated processor[12]. The test data set was constructed in $30,000 \times 4 \times 60s/25 =$ 80 processor hours. Since four cores were used, it took less than a full day. After the feature extraction, matching a $40s$ query with the test database with $30,000$ songs is done within $75ms$. The complete matching process for a $40s$ fragment takes about one second. Monitoring multiple streams in real-time poses no problem for the system. Building a fingerprint dataset with Audfprint is faster since fingerprints are extracted from an FFT which is less demanding than a Constant-Q transform. The matching step performance, however, is comparable.

Failure analysis shows that the system does not perform well on music with spectrograms either with very little energy or energy evenly spread across the range. Also extremely repetitive music, with a spec-

[12]The testing machine has an Intel Core2 Quad CPU Q9650 @ 3.00GHz introduced in 2009. The processor has four cores.

trogram similar to a series of dirac impulses, is problematic. Also, performance drops when time modifications of more than 8% are present. This could be partially alleviated by redesigning the time parameters used in the fingerprint hash, but this would reduce the discriminative power of the hash.

### 3.3.4 Conclusion

In this paper a practical acoustic fingerprinting system was presented. The system allows fast and reliable identification of small audio fragments in a large set of audio, even when the fragment has been pitch-shifted and time-stretched with respect to the reference audio. If a match is found the system reports where in the reference audio a query matches, and how much time/frequency has been modified. To achieve this, the system uses local maxima in a Constant-Q spectrogram. It combines event points into groups of three, and uses time ratios to form a time-scale invariant fingerprint component. To form pitch-shift invariant fingerprint components only frequency differences are stored. For retrieval an exact hashing matching algorithm is used.

The system has been evaluated using a freely available data set of 30,000 songs and compared with a baseline system. The results can be reproduced entirely using this data set and the open source implementation of Panako. The scripts to run the experiment are available as well. The results show that the system's performance decreases with time-scale modification of more than eight percent. The system is shown to cope with pitch-shifting, time-stretching, severe compression, and other modifications as echo, flanger and band pass.

To improve the system further the constant-Q transform could be replaced by an efficient implementation of the non stationary Gabor transform. This is expected to improve the extraction of event points and fingerprints without effecting performance. Panako could also benefit from a more extensive evaluation and detailed comparison with other techniques. An analysis of the minimum , most discriminative, information needed for retrieval purposes could be especially interesting.

## 3.4 Synchronizing multimodal recordings using audio-to-audio alignment An application of acoustic fingerprinting to facilitate music interaction research

Six, J. and Leman, M. (2015). Synchronizing Multimodal Recordings Using Audio-To-Audio Alignment. *Journal of Multimodal User Interfaces*, 9(3):223–229.

### Abstract

*Research on the interaction between movement and music often involves analysis of multi-track audio, video streams and sensor data. To facilitate such research a framework is presented here that allows synchronization of multimodal data. A low cost approach is proposed to synchronize streams by embedding ambient audio into each data-stream. This effectively reduces the synchronization problem to audio-to-audio alignment.*

*As a part of the framework a robust, computationally efficient audio-to-audio alignment algorithm is presented for reliable synchronization of embedded audio streams of varying quality. The algorithm uses audio fingerprinting techniques to measure offsets. It also identifies drift and dropped samples, which makes it possible to find a synchronization solution under such circumstances as well. The framework is evaluated with synthetic signals and a case study, showing millisecond accurate synchronization.*

### 3.4.1 Introduction

During the past decades there has been a growing interest in the relation between music and movement, an overview of ongoing research is given by Godøy and Leman (2010). This type of research often entails the analysis of data from various (motion) sensors combined with multi-track audio and video recordings. These multi-modal signals need to be synchronized reliably and precisely to allow successful analysis, especially when aspects on musical timing are under scrutiny.

Synchronization of heterogeneous sources poses a problem due to large variability in sample rate and due to the latencies introduced by each recording modality. For example it could be the case that accelerometer data (sampled at 100Hz) needs to be synchronized with both multi-track audio (recorded at $48kHz$) and two video streams, captured using webcams at 30 frames per second.

Several methods have been proposed to address synchronization problems when recording multi-modal signals. The most straightforward approach to solve this problem is to route a master clock signal through each device and synchronize using this pulse. Jaimovich and Knapp (2010) show a system where an SMPTE signal serves as a clock for video cameras and other sensor as well. In a system by Hochenbaum and Kapur (2012) a clock signal generated with an audio card is used to synchronize OSC, MIDI, serial data and audio. A drawback of this approach is that every recording modality needs to be fitted with a clock signal input. When working with video this means that expensive cameras are needed that are able to control shutter timing with a sync port. Generally available webcams do not have such functionality. The EyesWeb system (Camurri et al., 2004) has similar preconditions.

An other approach is to use instantaneous synchronization markers in data-streams. In audio streams such marker could be a hand clap. A bright flash could be used in a video stream. These markers are subsequently employed to calculate timing offsets and synchronize streams either by hand or assisted by custom software. This method does not scale very well to multiple sensor streams and does not cope well with drift or dropped samples. Some types of sensor streams are hard to manipulate with markers e.g. ECG recordings, which prohibits the use of this method. Although the method has drawbacks it can be put to use effectively in controlled environments, as is shown by Bannach et al. (2009); Gowing et al. (2011).

In this article a novel low-cost approach is proposed to synchronize streams by embedding ambient audio into each stream. With the stream and ambient audio being recorded synchronously, the problem of mutual synchronization between streams is effectively reduced to audio-to-audio alignment. As a second contribution of this paper, a

robust, computationally efficient audio-to-audio alignment algorithm is introduced. The algorithm extends audio fingerprinting techniques with a cross-covariance step. It offers precise and reliable synchronization of audio streams of varying quality. The algorithm proposes a synchronization solution even if drift is present or when samples are dropped in one of the streams.

### 3.4.2 Audio-to-audio alignment

There are several requirements for the audio-to-audio alignment algorithm. First and foremost, it should offer reliable and precise time-offsets to align multiple audio-streams. Offsets should be provided not only at the start of the stream but continuously in order to spot drifting behavior or dropped samples. The algorithm needs to be computationally efficient so it can handle multiple recordings of potentially several hours long. Highly varying signal quality should not pose a problem for the alignment algorithm: the algorithm should be designed to reliably match a stream sampled at a low bit-rate and sample rate with a high-fidelity signal.

The requirements are similar to those of acoustic fingerprinting algorithms. An acoustic fingerprinting algorithm uses condensed representations of audio signals, acoustic fingerprints, to identify short audio fragments in large audio databases. A robust fingerprinting algorithm generates similar fingerprints for perceptually similar audio signals, even if there is a large difference in quality between the signals. Wang (2003) describes such algorithm. Wang's algorithm is able to recognize short audio fragments reliably even if the audio has been subjected to modifications like dynamic range compression, equalization, added background noise and artifacts introduced by audio coders or A/D-D/A conversions. The algorithm is computationally efficient, relatively easy to implement and yields precise time-offsets. All these features combined make it a good candidate for audio-to-audio alignment. This is recognized by others as well since variations on the algorithm have been used to identify multiple video's of an event (Cotton and Ellis, 2010) and to identify repeating acoustic events in long audio recordings (Ogle and Ellis, 2007). The patent application for the algorithm by Wang and Culbert (2002) mentions that it could be used for precise synchronization, unfortunately it is not detailed how this could be done. Below a novel post-processing step to achieve precise synchronization is proposed.

Shrestha et al. (2007) describes another approach to audio-to-audio synchronization. This algorithm offers an accuracy of +-11ms in the best case and does not cope with drift or dropped samples. Two elements that are improved upon in the algorithm proposed below.

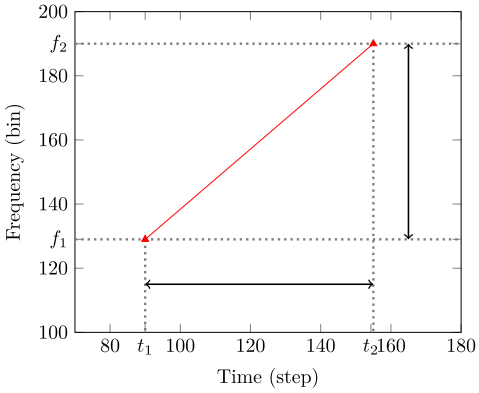



Figure 3.11: A fingerprint consists of two peaks in a time-frequency representation.

The algorithm works as follows. First audio is transformed to the time-frequency domain. In the time-frequency domain peaks are extracted. Peaks have a frequency component $f$ and a time component $t$. The frequency component is expressed as an FFT bin index and the time component as an analysis frame index. The peaks are chosen to be spaced evenly over the time-frequency plane. A combination of two nearby peaks are combined to form one fingerprint, as shown in Figure 3.11. The fingerprints of the reference audio are stored in a hashtable with the key being a hash of $f_1$, $\Delta t$, $\Delta f$. Also stored in the hashtable, along with the fingerprint, is $t_1$, the absolute time of the first peak. Further details can be found in Six and Leman (2014); Wang (2003); Wang and Culbert (2002).

For audio that needs to be synchronized with a reference, fingerprints are extracted and hashed in the exact same way. Subsequently, the hashes that match with hashes in the reference hashtable are identified. For each matching hash, a time offset is calculated between the query and the reference audio, using the auxiliary information stored in the hashtable $t_1$. If query and reference audio match, an identical time offset will appear multiple times. Random chance matches do not have this property. If a match is finally found, the time offset is reported. The matching step is visualized in Figure 3.12. In Figure 3.12 several matching fingerprints are found. For two of those matching fingerprints the time offset is indicated using the dotted lines. The figure also makes

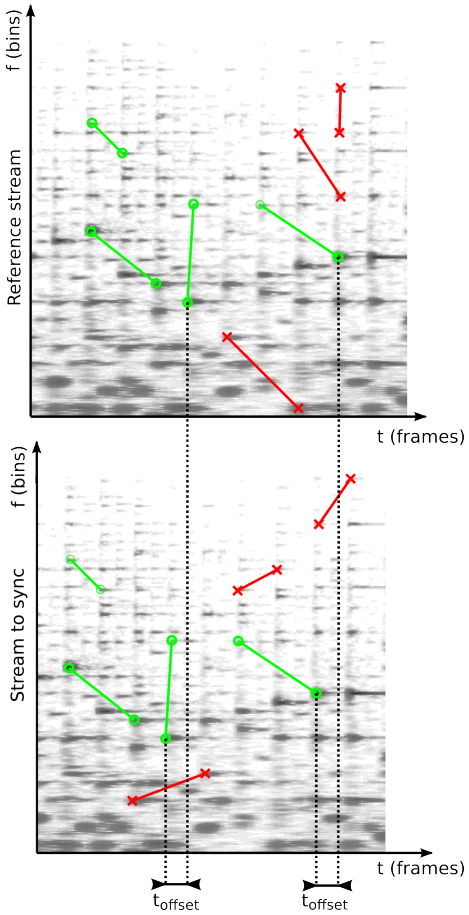



Figure 3.12: Two streams of audio with fingerprints marked. Some fingerprints are present in both streams (green, O) while others are not (red, ×). Matching fingerprints have the same offset ($t_{offset}$), indicated by the dotted lines.

clear that the method is robust to noise or audio that is only present in one of the streams. Since only a few fingerprints need to match with the same offset, the other fingerprints - introduced by noise or other sources - can be safely discarded (the red fingerprints in Figure 3.12).

In this algorithm time offsets are expressed using the analysis frame index. The time resolution of such audio frame is not sufficient for precise synchronization. [13] To improve time resolution cross-covariance between audio blocks of the reference $r$ audio and query $q$ is calculated. For each time shift $i$ the following is done: $(r \star q)_i = \sum_j r_j^* \, q_{i+j}$, with $j$ the size of the audio block. The time shift $i$ with the highest covariance determines the best match between the two signals. By adding the time shift, expressed in samples, to the audio block index times the audio block size, a sample accurate time offset can be determined.

The cross-covariance calculation step is not efficient $\mathcal{O}(i * j^2)$, so it should be done for the least amount of audio blocks possible. Since it is known from before at which audio block indexes similar audio is present - at audio blocks with matching offsets - in the reference and query, the calculation can be limited to only those audio blocks. The number of cross-covariance calculations can be further reduced by only calculating covariances until agreement is reached.

Until now there have been no precautions to deal with drift or dropped samples. To deal with drift the algorithm above is expanded to allow multiple time-offsets between the audio that needs to be synchronized and a reference. During the iteration of fingerprint hash matches, a list of matching fingerprints is kept for each offset. If list of matching fingerprints reaches a certain threshold the corresponding offset is counted as a valid. Drift can then be identified since many, gradually increasing or decreasing, offsets will be reported. If samples are dropped, two distinct offsets will be reported. The time at which samples are dropped or drift occurs is found in the list of matching fingerprints for an offset. The time information of the first and last fingerprint match mark the beginning and end of a sequence of detected matches.

### 3.4.3 Synchronizing data streams

With a working audio-to-audio alignment strategy in place, a setup for multimodal recording should include audio streams for each sensor stream. While building a setup it is of utmost importance that the sensor-stream and the corresponding audio-stream are in check. For a video recording this means that AV-sync needs to be guaranteed. To

[13] If for example audio with an 8000Hz sample rate is used and each analysis frame is 128 samples, then time resolution is limited to 16ms.

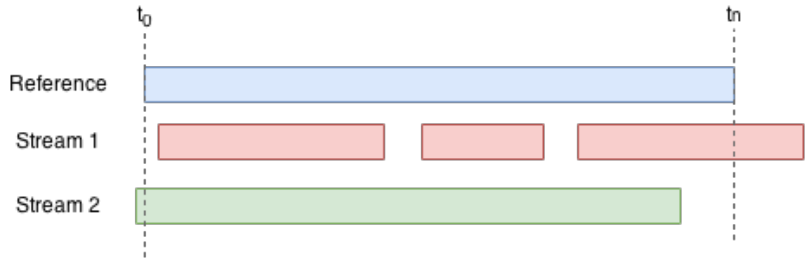



Figure 3.13: A reference stream (blue) can be synchronized with streams one and two. It allows a workflow where streams are started and stopped (red) or start before the reference stream (green).

make sure that analog sensors are correctly synchronized with audio the use of a data acquisition module is advised. Generally these modules are able to sample data at sufficiently high sampling rates and precision. The main advantage is that such module generally has many analog input ports and by recording the audio via the same path it is guaranteed to be in sync with the sensor streams. In the setup detailed below (Section 3.4.4), for example, an USB data acquisition module with 16 inputs, 16 bit resolution and maximum sample rate of 200kHz is used.

To accommodate data-streams with an inherently low sampling - 8kHz or less - rate a method can be devised that does not sample the audio at the same rate as the data-stream. In the setup detailed below an accelerometer (Section 3.4.4) is sampled at 345Hz by dividing the audio into blocks of 128 samples, and only measuring one acceleration for 128 audio samples. Since the audio is sampled at 44.1kHz the data is indeed sampled at $44100Hz/128 = 345Hz$. Depending on the type of data-stream and audio-recording device, other sampling rates can be achieved by using other divisors. Conversely, high data sampling rates can be accommodated by using a multiplier instead of a divisor.

Note that during such recording session a lot of freedom concerning the workflow is gained. While one stream is recording continuously stopping and starting other recording modalities can be done without affecting sync. The order at which recording devices are started also has no effect on synchronization. This is in stark contrast with other synchronization methods. In Figure 3.13 this is shown.

Once all data is synchronized analysis can take place. If video analysis is needed, a tool like ELAN (Wittenburg et al., 2006) can be used. If

audio and sensor-streams are combined without video, Sonic Visualizer (Cannam et al., 2006) is helpful to check mutual alignment. To store and share multimodal data RepoVIZZ (Mayor et al., 2013) is useful.

### 3.4.4 Results

To test the audio-to-audio alignment algorithm, it was implemented in Java. A user-friendly interface, called SyncSink[14], supporting drag-and-drop was developed as well. The SyncSink tool is depicted in Figure 3.14. The implementation of the algorithm is based on the open-source acoustic fingerprinting software Panako (Six and Leman, 2014) and TarsosDSP (Six et al., 2013, 2014). Panako contains several modules that can be reused: the automatic conversion of audio in various formats to PCM, the efficient FFT implementation and the spectral peak detection algorithm which uses a two dimensional min-max filter. The algorithm works best with an FFT of 512 points, and a step size of 128 samples for a sampling rate of 8000Hz. At least 7 fingerprints need to match before a synchronization solution is presented. With this parameter configuration, the worst synchronization accuracy is equal to $8000Hz/128 = 16ms$. Thanks to the cross covariance step, the best case synchronization is sample accurate. It is limited to $1/8000Hz = 0.125ms$.

To measure the accuracy of the time-offsets that are reported by the algorithm the following experiment is done. For each audio file in a dataset a random snippet of ten seconds is copied. The ten seconds of audio is stored together with an accurate representation of the offset at which it starts in the reference audio. Subsequently the snippet is aligned with the reference audio and the actual offset is compared with the offset reported by the algorithm. To make the task more realistic the snippet is GSM 06.10 encoded. The GSM 06.10 encoding is a low-quality 8KHz, 8-bit encoding. This degradation is done to ensure that the algorithm reports precise time-offsets even when high-fidelity signals - the reference audio files - are aligned with low-quality audio - the snippets.

The procedure described above was executed for a thousand snippets of ten seconds. For 17 an incorrect offset was found due to identically repeating audio. It could be said that these offsets yield an alternative, but equally correct, alignment. For 10 snippets no alignment was found. For the remaining 973 snippets the offsets were on

[14]SyncSink is included into the GPL'd Panako project available at `http://panako.be`

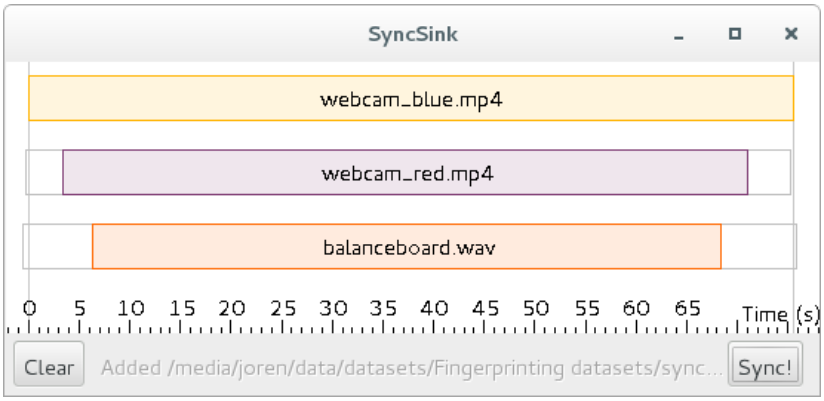



Figure 3.14: A user-friendly interface to synchronize media and data files. First a reference media-file is added using drag-and-drop. The audio steam of the reference is extracted and plotted on a timeline as the topmost box. Subsequently other media-files are added. The offsets with respect to the reference are calculated and plotted. CSV-files with timestamps and data recorded in sync with a stream can be attached to a respective audio stream. Finally, after pressing `Sync!`, the data and media files are modified to be exactly in sync with the reference.

average 1.01ms off, with a standard deviation of 2.2ms. The worst case of 16ms (128/8000Hz) was reported once. Note that a delay in audio from 1-14ms affects spatial localization, a 15-34ms delay creates a chorus/flanger like effect and starting from 35ms discrete echos can be heared.

To get an idea how quickly the algorithm returns a precise offset, the runtime was measured. Four reference audio files were created, each with a duration of one hour and each with a corresponding query file. The queries consist of the same audio but GSM encoded and with a small time-offset. A query performance of on average 81 times real-time is reached on modest computing hardware[15] when synchronizing the reference with query streams.

This method was used in a study with dementia patients. The study aimed at measuring how well participants can synchronize to a musical stimulus. A schematic of the system can be found in Figure 3.15. The system has the following components:

[15] An Intel Core2 Quad CPU Q9650 @ 3.00GHz x 4 was used, with 8GB memory. A CPU that entered the market late 2008.

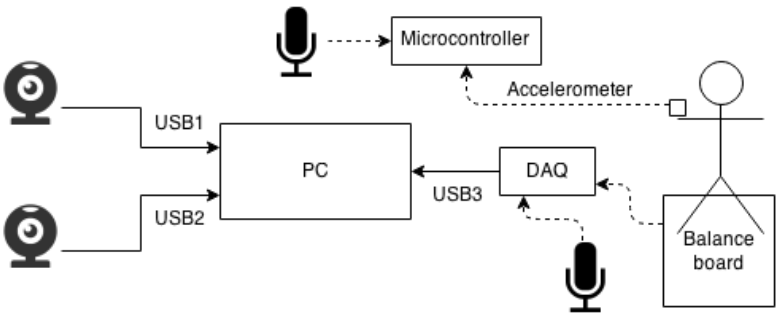



Figure 3.15: Multimodal recording system diagram. Each webcam has a microphone and is connected to the pc via USB. The dashed arrows represent analog signals. The balance board has four analog sensors but these are simplified to one connection in the schematic. The analog output of the microphones is also recorded through the DAQ. An analog accelerometer is connected with a microcontroller which also records audio.

- A balance board equiped with an analog pressure sensors at each corner.
- Two HD-webcams (Logitech C920), recording the balance board and ambient audio using the internal microphones.
- An electret microphone (CMA-4544PF-W) with amplifier circuit (MAX4466).
- A data acquisition module with analog inputs. Here an Advantech USB4716 DAQ was used. It has 16 single-ended inputs with 16-bit resolution and is able to sample up to 200 kHz.
- A wearable microcontroller with an electret microphone, MicroSD-card slot and an analog accelerometer (MMA7260Q) attached to it. Here we used the Teensy 3.1 with audio shield. It runs at 96MHz and has enough memory and processing power to handle audio sampled at 44.1kHz in real-time. The microcontroller can be seen in Figure 3.16.

The microcontroller shown in Figure 3.16 was programmed to stream audio data sampled at 44.1kHz to the SD-card in blocks of 128 samples.

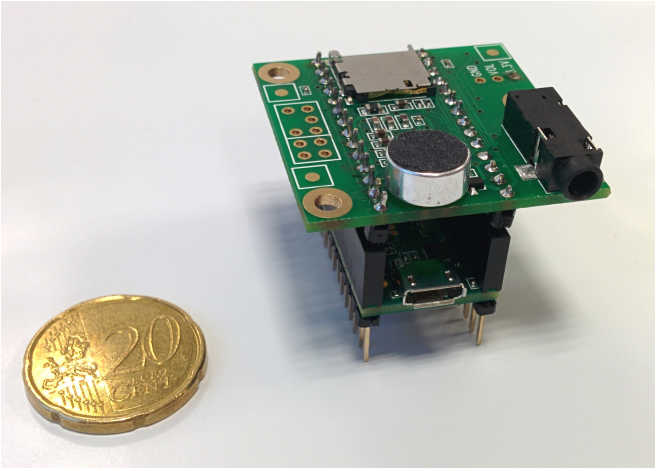


Figure 3.16: A microcontroller fitted with an electret microphone and a microSD card slot. It can record audio in real-time together with sensor data.

Using the same processing pipeline the instantaneous acceleration was measured for each block of audio. This makes sure that the measurements and audio stay in sync even if there is a temporary bottleneck present in the pipeline. During the recording session this showed to be of value: due to a slow micro SD-card[16] audio samples were dropped twice in a recording session of 30 seconds. Thanks to the audio alignment algorithm it became clear that this happened after 3.1 seconds and after 23.9s. The sensor stream from zero to 3.1 seconds could not be matched reliably nor could the data from 23.9s to 30s. The bulk of the acceleration measurements, between 3.1 and 23.9 seconds, could however be synchronized correctly with the other data streams. A feat that would have been difficult using other synchronization means.

Once all data was transferred to a central location mutual time offsets were calculated automatically. Subsequently the files were trimmed in order to synchronize them. In practice this means chopping off a part of the video or audio file (using a tool like ffmpeg) and modifying the data files accordingly. The data of this recording session and the software used is available at `http://0110.be/syncsink`. The next step is to analyse the data with tools like ELAN (Wittenburg et al.,

[16]To support real-time recording writing a 512 byte buffer should be faster than $256/44100Hz = 5.802ms$, on average.

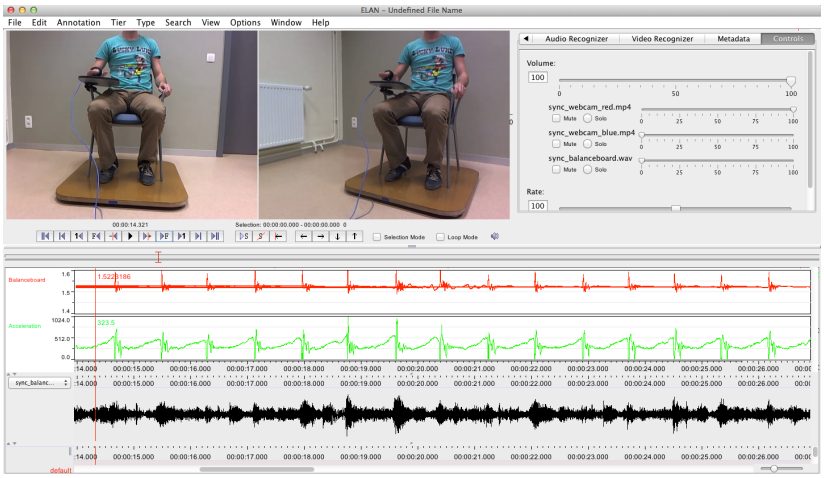



Figure 3.17: The synchronized data from the two webcams, accelerometer and balanceboard in ELAN. From top to bottom the synchronized streams are two video-streams, balance-board data (red), accelerometer-data (green) and audio (black).

2006) or Sonic Visualizer (Cannam et al., 2006) (Figure 3.17 and 3.18) or any other tool. The analysis itself falls outside the scope of this paper.

### 3.4.5 Discussion

Since ambient audio is used as a synchronization clock signal the speed of sound needs to be taken into account. If microphones are spread out over a room the physical latency quickly adds up to 10ms (for 3.4m) or more. If microphone placement is fixed this can be taken into account. If microphone placement is not fixed it should be determined if the precision that the proposed method can provide is sufficient for the measurement.

The proposed method should not be seen as a universal synchronization solution but provides a method that can fit some workflows. Combining audio-to-audio alignment with other synchronization methods is of course possible. If for example motion capture needs to be synchronized with other sources, the motion capture clock pulse can be routed through a device that records the clock together with ambient audio making it possible to sync with other modalities. The same could be done for an EEG system and clock. This setup would make it

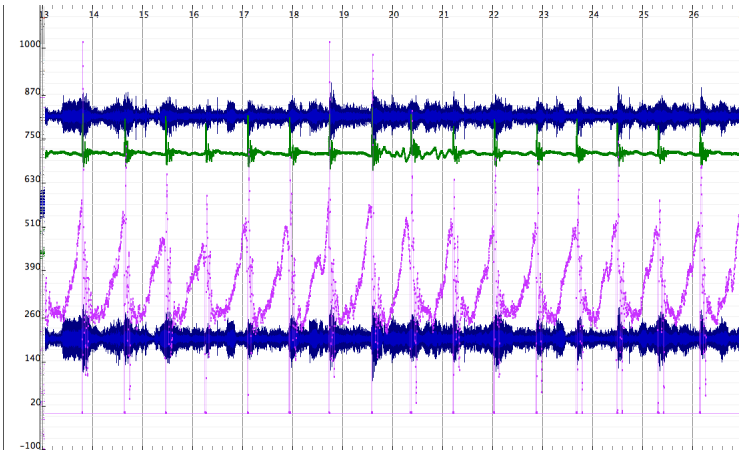



Figure 3.18: Synchronized streams in Sonic Visualizer. Here you can see two channel audio synchronized with accelerometer data (top, green) and balanceboard data (bottom, purple).

possible to sync EEG with motion capture data, an otherwise difficult task. Combining the method with other synchronization approaches - e.g. synchronization markers - is also possible.

The current system is presented as a post-processing step but if the latencies of each recording system are relatively stable then there is potential to use the approach *real-time*. It would work as follows: each recording device would start streaming both audio and data. After about ten seconds audio-to-audio alignment can be done and the mutual offsets can be determined. Once this information is known the sensor data can be buffered and released in a timely manner to form one fused synchronized sensor data stream. The overall latency of stream is then at best equal to the recording modality with the largest latency. While streaming data, the audio-to-audio alignment should be repeated periodically to check or adapt offsets of sensor streams.

### 3.4.6 Conclusion

An efficient audio-to-audio alignment algorithm was presented and used effectively to synchronizing recordings and linked data streams. The algorithm is based on audio fingerprinting techniques. It finds a rough offset using fingerprints and subsequently refines the offset with a cross-covariance step. During synthetic benchmarks an average synchronization accuracy of 1.1ms was reached with a standard deviation of 2.2ms. A query performance of 81 times real-time is reached on modest com-

puting hardware when synchronizing two streams. A case study showed how the method is used in research practice. The case study combines recordings from two webcams and a balance board together with acceleration data recorded using a microcontroller which were all synchronized reliably and precisely.

## 3.5 A framework to provide fine-grained time-dependent context for active listening experiences

Six, J. and Leman, M. (2017). A framework to provide fine-grained time-dependent context for active listening experiences. In *Proceedings of the AES Conference on Semantic Audio 2017*. AES.

### Abstract

*This work presents a system that is able to provide a user with fine-grained time-dependent context while listening to recorded music. By utilizing acoustic fingerprinting techniques the system recognizes which music is playing in the environment and also determines an exact playback position. This makes it possible to provide context at exactly the right time. It can be used to augment listening experiences with lyrics, scores, tablature or even music videos.*

*To test the concept, a prototype has been built that is able to give feedback that coincides with the beat of music playing in the user's environment. The system is evaluated with respect to timing and is able to respond to beats within 16 ms on average.*

### 3.5.1 Introduction

The ability to identify which music is playing in the environment of a user has several use cases. After a successful recognition meta-data about the music is immediately available: artist, title, album. More indirect information can also be made available: related artist, upcoming concerts by the artist or where to buy the music. Such systems have been in use for more than a decade now.

A system that is able to not only recognize the music, but also determine a sufficiently precise playback time opens new possibilities. It would allow to show lyrics, scores or tablature in sync with the music. If the time resolution is fine enough it would even allow to play music videos synced to the environment. In this work a design of such system is proposed. The paper focuses on yet another type of time-dependent contextual data: beat lists. A prototype is developed that provides feedback exactly on the beat for the following three reasons:

1. For its *inherent value*. Humans are generally able to track musical beat and rhythm. Synchronizing movement with perceived beats is a process that is natural to most. Both processes develop during early childhood (Hannon and Trehub, 2005). However, some humans are unable to follow musical beat. They fall into two categories. The first category are people that suffer from hearing impairments which have difficulties to perceive sound in general and music in particular. Especially users of cochlear implants that were early-deafened but only implanted during adolescence or later have difficulties following rhythm (Fuller et al., 2013; Timm et al., 2014). In contrast, post-lingually deafened CI users show similar performance as normal hearing persons (McDermott, 2004). The second category are people suffering from beat deafness (Lebrun et al., 2012; Phillips-Silver et al., 2011). Beat deafness is a type of congenital amusia which makes it impossible to extract music's beat. Both groups could benefit from a technology that finds the beat in music and provides tactile or visual feedback on the beat.

2. For *evaluation purposes*. Using discrete events - the beats - makes evaluation relatively straightforward. It is a matter of comparing the expected beat timing with the timing of the feedback event.

3. For *pragmatic reasons*. The contextual data - the beat lists - are available or can be generated easily. There are a few options to extract beat timestamps. The first is to manually annotate beat information for each piece of music in the reference database. It is the most reliable method, but also the most laborious. The second option is to use a state of the art beat tracking algorithms e.g.

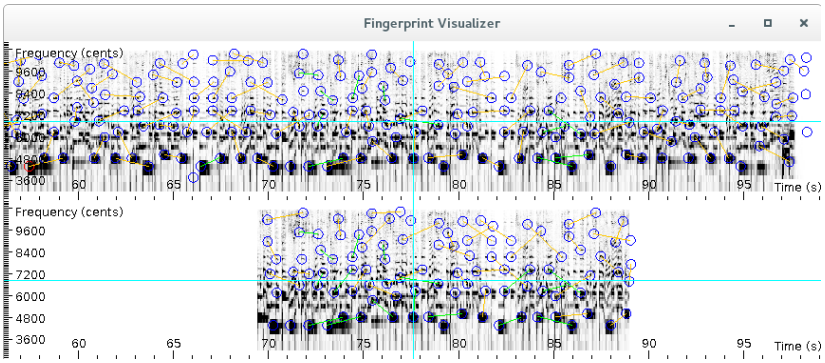



Figure 3.19: Two streams of audio with fingerprints marked. The reference audio is the stream on top, the bottom stream is the possibly noisy query. Some fingerprints are present in both streams (green) while others are not (yellow). Matching fingerprints have the same relative time offset in the query with respect to the reference.

the one available in Essentia (Bogdanov et al., 2013). The third option is to request beat timestamps from specialized web services. The AcousticBrainz project (Porter et al., 2015) provides such a service. AcousticBrainz[17] currently has information for more than two million songs. It is similar to the Echo Nest API (Bertin-Mahieux et al., 2011) but AcousticBrainz' inner workings are well documented and completely transparent. In the prototype AcousticBrainz is used to provide beat timings. If the audio is not present in AcousticBrainz, the beat timings are extracted using the previously mentioned beat-tracker and stored locally.

The following sections present the system and its evaluation. The paper ends with the discussion and the conclusions.

### 3.5.2 System overview

A system that provides time-dependent context for music has several requirements. The system needs to be able to recognize audio or music being played in the environment of the user together with a precise time-offset. It also needs contextual, time-dependent information to provide to the user e.g. lyrics, scores, music video, tablature or triggers. The information should be prepared beforehand and stored in a

[17]Information on the AcousticBrainz project is available at its website: `http://acousticbrainz.org/`

repository. The system also needs an interface to provide the user with the information to enhance the listening experience. As a final soft requirement, the system should preferably minimize computational load and resources so it could be implemented on smartphones.

Acoustic fingerprinting algorithms are designed to recognize which music is playing in the environment. The algorithms use condensed representations of audio signals to identify short audio fragments in vast audio databases. A well-designed fingerprinting algorithm generates similar fingerprints for perceptually similar audio signals, even if there is a large difference in quality between the signals. Wang (2003) describes such algorithm. The algorithm is based on pairs of spectral peaks which are hashed and compared with the peaks of reference audio. Wang's algorithm is able to recognize short audio fragments reliably even in the presence of background noise or artifacts introduced by A/D or D/A conversions. The algorithm is computationally efficient, relatively easy to implement and yields precise time-offsets. All these features combined make it a good algorithm to detect which music is being played and to determine the time-offset precisely. Figure 3.20 shows fingerprints extracted from two audio streams using Panako (Six and Leman, 2014), an implementation of the aforementioned algorithm. The reference audio is in the top, the query in the bottom. Using fingerprints that match (in green), the query is aligned with the reference audio.

For the prototype, the complete process is depicted in Figure 3.20. A client uses its microphone to register sounds in the environment. Next, fingerprints are extracted from the audio stream. The fingerprints are sent to a server. The server matches the fingerprints with a reference database. If a match is found, a detailed time-offset between the query and the reference audio is calculated. Subsequently, the server returns this time-offset together with a list of beat timestamps. Using this information the client is able to generate feedback events that coincide with the beat of the music playing in the users environment. This process is repeated to make sure that the feedback events remain in sync with the music in the room. If the server fails to find a match, the feedback events stop.

With the list of beat events available the system needs to generate feedback events. The specific feedback mode depends on the use case. Perhaps it suffices to accentuate the beat using an auditory signal: e.g. by a loud sound with a sharp attack. A bright flash on each beat could also help some users. Haptic feedback can be done with vibration motors attached to the wrist using a bracelet. A commercially available wireless tactile metronome - the Soundbrenner Pulse - lends itself well for this purpose. A combination of feedback modes could prove beneficial since multisensory feedback can improve sensorimotor

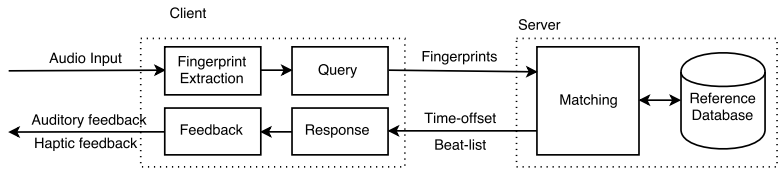



Figure 3.20: Schema of a client/server architecture to match an audio query and recieve synchronized feedback on the beat.

synchronization (Elliott et al., 2010; Six et al., 2017).

### 3.5.3 Evaluation

The evaluation makes clear how synchronized context can be delivered to ambient audio or music. The evaluation quantifies the time-offset between the beats - annotated beforehand - and the time of the feedback event that should correspond with a beat. For an evaluation of the underlying fingerprinting algorithm readers are referred to the evaluation by Wang (2003).

The evaluation procedure is as follows: a device plays a piece of music. The device also knows and updates the current playback position accurately in real-time[18]. A client listens to this audio, extracts fingerprints and sends these to a server (see 3.20). If the server finds a match, a time-offset is returned together with a list of beat events. The client uses the beat-list and offset to send events that are in sync with the beat to the playback device. The playback device responds with an accurate representation of the current playback position. Finally, the reported playback position is compared with the beat-list. The difference between the beats and the feedback events are used as a synchronicity measure.

To counter problems arizing with soft real-time thread scheduling and audio output latency the evaluation was done using a microcontroller with a hard real-time scheduler and low-latency audio playback. An extra benifit is that timing measurements are very presice. Cheap, easily programmable microcontrollers come with enough computing

[18]In a conventional computing environment this is not trivial. Audio travels throug layers of software abstractions or mixers before it arrives at the audio hardware output. At each point it can potentially be buffered. The time reported by a software audio player and the actual audio being played by the speakers differs by the total audio output latency which quickly amounts to over 20ms and is only constant if carfully configured.

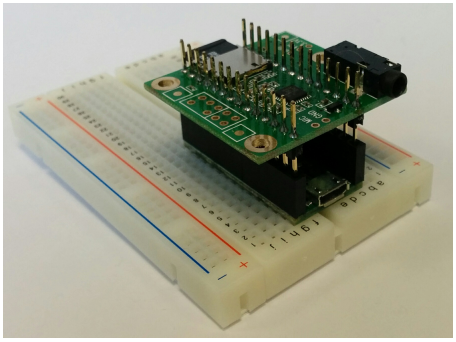


Figure 3.21: Teensy with Audio Board. The microcontroller is capable of high-quality low-latency audio playback from an SD-card and precise timing measurement.

power these days to handle high-quality audio. One such device is the Axoloti: a microcontroller for audio applications that can be programmed using a patcher environment. Another is the Teensy equiped with a Teensy Audio Board. Here we use a Teensy 3.2 with Audio Board for audio playback. It supports the Arduino environment which makes it easy to program it as a measuring device. It has an audio-output latency of about 1 ms. It is able to render 16 bit audio sampled at 44100 Hz and is able to read PCM-encoded audio from an SD-card. In the experimental setup, the Teensy is connected to a Behringer B2031 active speaker.

Audio enters the client by means of the built in microphone. The client part of Figure 3.20 is handled by a laptop: a late 2010 Macbook Air, model A1369 running Mac OS X 10.10. Next the audio is fed into the fingerprinting system. The system presented here is based on Panako (Six and Leman, 2014). The source of Panako is available online and it is implemented in Java 1.8. Panako was modified to allow a client/server architecture. The client is responsible for extracting fingerprints. A server matches fingerprints and computes time offsets. The server also stores a large reference database with fingerprints together with beat positions. The client and server communicate via a web-service using a JSON protocol. JSON is a standard that describes a way to encode data, it allows communication between computers.

When the time-offsets and the beat list are available on the client feedback events are generated. To evaluate the system the feedback events are sent over a USB-serial port to the Teensy. The Teensy

| BPM | Artist - Title |
|---|---|
| 82 | Arctic Monkeys - *Brianstorm* |
| 87 | Pendulum - *Propane Nightmares* |
| 90 | Ratatat - *Mirando* |
| 91 | C2C - *Arcades* |
| 95 | Hotei - *Battle Without Honor or Humanity* |
| 95 | Skunk Anansie - *Weak* |
| 100 | Really Slow Motion - *The Wild Card* |
| 105 | Muse - *Panic Station* |
| 108 | P.O.D. - *Alive* |
| 111 | Billie - *Give Me the Knife* |
| 121 | Daft Punk - *Around The World* |
| 128 | Paul Kalkbrenner - *Das Gezabel de Luxe* |
| 144 | Panda Dub - *Purple Trip* |
| 146 | Digicult - *Out Of This World* |
| 153 | Rage Against the Machine - *Bombtrack* |
| 162 | Pharrell Williams - *Happy* |

Table 3.2: The dataset used during the evaluation has a wide BPM range and a stable easy to follow audible beat.

replies over the serial port with the current playback position of the audio. The playback position is compared with the expected position of the beat and the difference is reported in milliseconds. Negative values mean that the feedback-event came before the actual audible beat, whereas positive values mean the opposite. The system is tested using a the data set presented in Table 3.2. It features music in a broad BPM range with a clear, relatively stable beat.

### 3.5.4 Results

The results are depicted in Figure 3.22. The system responds on average 16 ms before the beat. This allows feedback events to be perceived together with the beat. Depending on the tempo (BPM) of the music and type of feedback it might be needed to schedule events later or even sooner. This can be done by adapting the latency parameter which modifies the timing of the scheduled feedback events. However, there is a large standard deviation of 42 ms. The current system is limited to an accuracy of 32 ms: the size of a block of 256 audio samples, sampled at 8000 Hz. The block size used in the fingerprinter. In Figure 3.22 each histogram bin is 32 ms wide and centered around -16 ms. The results show that the system is able to recognize the correct audio block but is sometimes one block off. The main issue here is the unpredictable

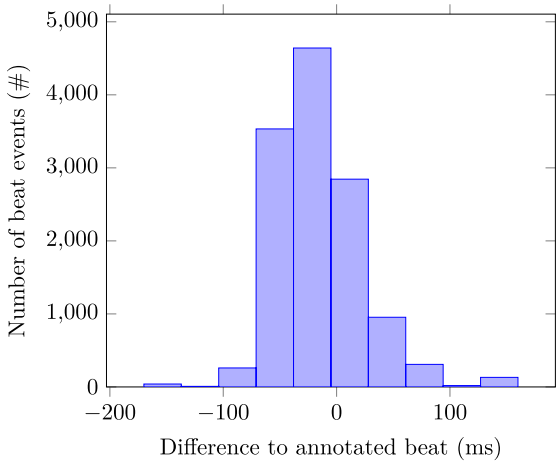



Figure 3.22: The difference (in ms) between feedback events and beats. The feedback events arrive 16ms before the beat to allow to perceive beats together with feedback. The data is centered around the average and bins of 32ms are used, the precision of the system.

nature of scheduling in Java: Java threads are not guaranteed to start with predictable millisecond accurate timing. Garbage collection can cause even larger delays. Larger delays are due to incorrectly recognized audio. Repetition in music can cause the algorithm to return an incorrect absolute offset which makes the beat drift. The results, however, do show that the concept of the system is very promising and can deliver timing dependent context.

In its current state the system listens to 12 seconds of audio and sends the fingerprints of those 12 seconds to the server. The state is reset after that. The system does not employ use-case dependent heuristics. If it is known beforehand that the user will most likely listen to full tracks the current state, time-offset and beat-lists could be reused intelligently to improve accuracy, especially in the case of repeating audio. This could also be optimized by running a real-time onset detector and correlating the detected onsets list with the beat list returned by the server, this would however make the system more computationally expensive.

### 3.5.5 Discussion

The proposed system only supports recorded music. Supporting live music is challenging but could be done. The MATCH algorithm (Dixon

and Widmer, 2005) for example supports tracking live performances in real time via dynamic time warping. The basic song structure however needs to be kept intact during the live rendition, otherwise the pre-computed contextual data becomes useless. Another challenge is DJ-sets. Although recorded music is used, during DJ-sets the tempo of the music is often modified to match a previous piece of music. To support such situations a more involved acoustic fingerprinting algorithm is needed. Currently there are two algorithms described in the literature that report both time-offsets and tempo modifications accurately (Six and Leman, 2014; Sonnleitner and Widmer, 2016).

Repetition is inherent in music. Especially in electronic music the same exact audio can be repeated several times. A fingerprinting algorithm that only uses a short audio excerpt could, in such cases, return an incorrect absolute offset. To alleviate this problem, context could also be taken into account. Also the type of data returned needs to be considered. Lyrics could be incorrect while tablature or beats could still be aligned correctly since they do not depend as much on an absolute offset.

Since the system uses a computationally inexpensive algorithm, it can be executed on a smartphone. The implementation used here is compatible with Android since it depends only on two Android compatible libraries (Six et al., 2014; Six and Leman, 2014). If only a small reference music library is used, all server components of the system could be moved to the smartphone. An app that offers aligned music videos for the music for one album could run all components easily on a smartphone without the need for an external server.

For the prototype a database with pre-computed beat-position is created *off-line* using all the acoustic information of the song. However, it is possible to determine beat positions with a real-time beat tracking algorithm by Goto (2001). Unfortunately, this poses several problems. Beat-tracking involves an important predictive and reflective element. To correctly model beats based on a list of onsets extracted in real-time, musical context is needed. This context may simply not be available. Another issue is that the computational load for a beat-tracking systems is often high. Müller (2015) gives an overview of beat tracking techniques which are challenging to implement on smartphones. A third problem is that feedback needs to be provided before the actual acoustic beat is perceived by the user. Tactile feedback e.g. takes around 35 ms to be processed (Elliott et al., 2010). Feedback based on a real-time beat-tracker - which introduces a latency by itself - would be always late. Generating feedback based on real-time beat-tracking algorithms is impractical especially in the context of smartphones with low-quality microphones and restricted computational resources.

To further develop the prototype into an assistive technology, more

fundamental research is needed to pinpoint the optimal type of feedback for user groups. The early deafened late implanted CI user group is recognized as an *'understudied clinical population'* (Fuller et al., 2013) for which models on auditory rhythm perception are underdeveloped. Insights into tactile or multi-modal rhythm perception for this specific group seem to be lacking from the academic literature. There is however a study that suggests that multi-sensory cues improve sensorimotor synchronization (Elliott et al., 2010). In Sowiński and Dalla Bella (2013) another fundamental issue is raised. In the study two participants seem to be able to perceive small timing deviations in audio but are unable to move accordingly. As the authors put it *"This mismatch of perception and action points toward disrupted auditory-motor mapping as the key impairment accounting for poor synchronization to the beat"*. The question remains whether this holds for tactile-motor mappings especially in the late implanted CI user-group.

### 3.5.6 Conclusion

A system was described that employs acoustic fingerprinting techniques to provide augmented music listening experience. A prototype was developed that provides feedback synchronized with music being played in the environment. The system needs a dataset with fingerprints from reference audio and pre-computed beat-lists. Since it offers fine-grained context awareness it can show lyrics, scores, visuals, aligned music videos or other meta-data that enrich the listening experience. The system can also be used to trigger events linked to audio during e.g. a theater performance.

# 4 Discussion and conclusion

This chapter starts with a list of contributions. It also provides a place for discussing limitations and future work by introduction of the term augmented humanities. It finishes with concluding remarks.

## 4.1 Contributions

General contributions of this doctoral thesis are the way to think about technical contributions to a field using the concepts of methods and services. The specific contributions of this doctoral research project are found in the individual articles.

The main contribution of Tarsos (Six et al., 2013) is that it lowered the barrier to allow musicologist and students to quickly extract pitch class histograms. Accessibility to an easy-to-use analysis tool allows it to be a stepping stone to automated methods for large scale analysis which are also available with Tarsos. The underlying DSP library (Six et al., 2014) became a stand-alone service and has been used in research and interactive music applications.

The contribution of the Panako (Six and Leman, 2014) acoustic fingerprinting system is threefold. First it features a novel algorithm for acoustic fingerprinting that is able to match queries with reference audio even if pitch shifting and time-stretching is present. Although similarities to a previously patented method (Wang and Culbert, 2009) were pointed out after publication it remains a valuable contribution to the academic literature. The second contribution lies in the publicly available and verifiable implementation of the system and three other baseline algorithms (Haitsma and Kalker, 2002; Six and Cornelis, 2012b; Wang, 2003), which serve as a service to the MIR community. The final contribution is the reproducible evaluation method which has been copied by Sonnleitner and Widmer (2016) and is part of the focus of Six et al. (2018a). To make the technology and capabilities of Panako better known it has been featured in articles targeting archivist and library science communities (Bressan et al., 2017b; Six et al., 2018b).

Six et al. (2018a) contributed to the discussion in computational research and challenges concerning reproducibility. It summarizes the common problems and illustrates these by replicating – in full – a seminal acoustic fingerprinting paper. It also proposes ways to deal with reproducibility problems and applies these for the reproduced work.

Six and Leman (2015) describe a novel way to synchronize multimodal research data. The main contribution of the work is the described method and the implementation to reduce the problem of synchronization of data to audio to audio alignment. The method was used in practice by Desmet et al. (2017) and extended with a real-time component during a master thesis project by Van Assche (2016).

Six and Leman (2017), finally enable augmented musical realities. The musical environment of a user can be enriched with all types of meta-data by using the blue-prints of the system described in this article.

Another set of contributions are the solutions that enabled empirical studies. Development of various hard- and software systems with considerable technical challenges facilitated research on various aspects of interaction with music. These solutions are featured in articles by Desmet et al. (2017); Six et al. (2017); Van Den Berge et al. (2018); Van Dyck et al. (2017); Vanhecke et al. (2017). Each of these assisted in a technical solution for one or more components typically present of an empirical research project: activation, measurement, transmission, accumulation or analysis. For more detail on each of these please see 1.1.3.

## 4.2 From digital humanities towards augmented humanities

To hint at the limitations of my current research and immediately offer perspectives on future work the augmented humanities in introduced.

In my research on MIR and archives and in the digital humanities in general, cultural objects are often regarded as static, immutable objects. An example of this is the instrument collection of the Royal museum of Central Africa in Tervuren, Belgium (RMCA). There, thousands of musical instruments are conserved and studied. These physical objects are cataloged according to sound producing mechanism or physical characteristics and digitized via photographs by the MIMO Consortium[1]. The digital archive with field recordings is seen in a similar way. These recordings are historical records, the witnesses of a certain event, set in stone. This immediately brings us to an important limitation of much of my research. When only audio is considered, there are a few well-defined but limited research questions that can be handled using these static objects.

[1]See http://www.mimo-international.com/MIMO for the website of the MIMO consortium.

However, cultural artifacts typically thrive on interaction, interaction between performer, public, time, space, context, body and expressive capabilities. An unplayable musical instrument studied as a static 'thing' loses its interactive dynamics. The instrument is very much part of an interactive system between an expert performer, listeners and context. When listening in silence with headphones to a musical record of a participatory act of music making, it loses much of its meaning. The static witness of this act only captures a ghost of its 'true multi-dimensional self'. Other research questions can be answered when this interactive aspect is re-introduced.

This distinction between static and interactive could be summarized as a path from digital humanities to augmented humanities. The augmented humanities could re-introduce, augment or diminish interactive behavior to test certain hypothesis. The interactive behavior could be elicited or, conversely, prevented or sabotaged to gain insights into this behavior. The cultural object would be seen as an actor in this model. To interfere or mediate interactions augmented reality technology could be re-purposed. Perhaps some examples can clarify this vision.

The D-Jogger system by Moens et al. (2014) could be seen as rudimentary augmented humanities research. It is a system for runners which are listening to specific music while running. The music is chosen to have a tempo in the range of the runners number of steps per minute. The system employs the tendency many runners have to align their footfalls with the musical beats. So far, music is still an immutable actor in this interaction. Runners simply align their actions with the music. However, if the music is modified dynamically to match its tempo with the runners, a system with two active actors appears. Both are drawn to one another. The interaction can then be elicited by allowing the music to always sync to the runner or the interaction can be prevented: the beat would then never align with a footfall. The runner can also be sped-up by first allowing an alignment and dynamically increasing the musical tempo slightly. The point is that music stops being an immutable, a static but becomes an actor that offers a view on the interactive system. Modifying the interaction sheds light on which musical features activate or relax movement (Leman et al., 2013), the coupling strength between runner and musical tempo and the adaptive behavior of a runner.

A study by Coorevits and Moelants (2016) studies musical scores of baroque dance music. For these scores tempo indications are not present. The metronome and precise indications on the score of the preferred tempo only appeared later in the romantic period. To infer a tempo from a static score is very challenging. A more interactive approach is to play the music in different tempi and let people with experience in dancing on baroque music move to the music and subse-

quently determine an optimal tempo. In this example again the static cultural object – a musical score – is transformed into a malleable (with respect to tempo) interaction between dancers and performers.

Historically, MIR technologies are developed to describe and query large databases of music (Burgoyne et al., 2016). However, MIR techniques can be used in interactive settings as well. For example, in (Six and Leman, 2017) acoustic fingerprinting techniques are used to evoke interaction. This system is capable to present the user with an augmented musical reality or more generally a computer-mediated reality. It allows to augment the music in a user's environment with additional layers of information. In the article a case is built around a cochlear implant user and dance music playing in his or her environment. The system emphasizes musical beat with a tactile pulse in sync with the music environment. This allows users with limited means of tracking musical beats to synchronize movement to beats.

To further this type of research, in which musical realities are constructed and interaction is modified, requires three aspects. First, insights are needed into the core topics of the humanities and hypothesis on how humans develop meaningful, sense-giving, interactions with their (musical) environment (many can be found in Leman (2016)). Secondly, a technological aspect is required to allow a laboratory setting in which environments can be adjusted (augmented) and responses recorded. The third aspect concerns the usability of these new technologies, it should be clear how convincing this augmented reality which relates to ecological validity.

The ArtScience Lab at the Krook in Ghent together with the usability laboratories also at the Krook, provide the right infrastructure to be able to show the potential of an augmented humanities approach. With (Six and Leman, 2017) I have already taken a step in this direction and would like to further continue in this direction.

## 4.3 Conclusion

I have introduced a way to organize solutions for problems in systematic musicology by mapping them on a plane. One axis contrasts methods with services while the other axis deals with techniques: MIR-techniques versus techniques for empirical research. By engineering prototypes, tools for empirical research and software systems I have explored this plane and contributed solutions that are relevant for systematic musicology. More specifically I have contributed to tone scale research, acoustic fingerprinting, reproducibility in MIR and to several empirical research projects.

I have aimed for a reproducible methodology by releasing the source

code of software systems under open source licenses and have evaluated systems with publicly available music when possible.

To illustrate the limitations of my current research and to propose a direction of future work I have proposed the term augmented humanities where hypothesis on musical interactions are tested by interfering in these interactions. Augmented reality technologies offer opportunities to do this in a convincing manner.

# A List of output

## A.1 List of peer-reviewed journal articles

Articles in chronological order. Bold means included in the dissertation.

**Six, J., Cornelis, O., and Leman, M. (2013). Tarsos, a modular platform for precise pitch analysis of Western and non-Western music. *Journal of New Music Research*, 42(2):113–129**

Cornelis, O., Six, J., Holzapfel, A., and Leman, M. (2013). Evaluation and recommendation of pulse and tempo annotation in ethnic music. *Journal of New Music Research (JNMR)*, 42(2)

**Six, J. and Leman, M. (2015). Synchronizing Multimodal Recordings Using Audio-To-Audio Alignment. *Journal of Multimodal User Interfaces*, 9(3):223–229**

Vanhecke, F., Moerman, M., Desmet, F., Six, J., Daemers, K., Raes, G.-W., and Leman, M. (2017). Acoustical properties in Inhaling Singing: a case-study. *Physics in Medicine*

Van Dyck, E., Six, J., Soyer, E., Denys, M., Bardijn, I., and Leman, M. (2017). Adopting a music-to-heart rate alignment strategy to measure the impact of music and its tempo on human heart rate. *Musicae Scientiae*, 0(0):10

**Six, J., Bressan, F., and Leman, M. (In press – 2018a). A case for reproduciblity in MIR. Replication of 'a highly robust audio fingerprinting system'. *Transactions of the International Society for Music Information Retrieval (TISMIR)***

Van Den Berge, P., Six, J., Gerlo, J., De Clerq, D., and Leman, M. (Submitted - 2018). Real-time measures of tibial acceleration as biofeedback on impact intensity during overground running. *Journal of Biomechanics*, 0(0)

Maes, P.-J., Lorenzoni, V., Moens, B., Six, J., Bressan, F., Schepers, I., and Leman, M. (Submitted - 2018). Embodied, participatory sense-

making in digitally-augmented music practices: Theoretical principles and the artistic case "soundbikes". *Critical Arts*

## A.2 List of peer-reviewed conference contributions

Articles in bold are included in the dissertation. The type of presentation is included at the end of each item.

Six, J. and Cornelis, O. (2011). Tarsos - a Platform to Explore Pitch Scales in Non-Western and Western Music. In *Proceedings of the 12th International Symposium on Music Information Retrieval (ISMIR 2011)* – Oral presentation

Six, J. and Viro, V. (2011). Peachnote Piano: Making MIDI Instruments Social and Smart Using Arduino, Android and Node.js. In *Late breaking Demos at the 12th International Symposium on Music Information Retrieval (ISMIR 2011)* – Demo presentation

Six, J. and Cornelis, O. (2012a). A Robust Audio Fingerprinter Based on Pitch Class Histograms - Applications for Ethnic Music Archives. In *Proceedings of the Folk Music Analysis conference (FMA 2012)* – Poster presentation

Cornelis, O. and Six, J. (2012). Towards the Tangible: Mircotonal Scale Exploration in Central-African Music. In *Proceedings of the Analytical Approaches to World Music Conference, AAWM 2012* – Poster presentation

Six, J. and Cornelis, O. (2013). Computer Assisted Transcription of Ethnic Music. In *Proceedings of the 2013 Folk Music Analysis conference (FMA 2013)* – Poster presentation

**Six, J., Cornelis, O., and Leman, M. (2014). TarsosDSP, a real-time audio processing framework in Java. In *Proceedings of the 53rd AES Conference (AES 53rd).* The Audio Engineering Society – Poster presentation**

**Six, J. and Leman, M. (2014). Panako - A scalable acoustic fingerprinting system handling time-scale and pitch modification. In *Proceedings of the 15th ISMIR Conference (ISMIR***

***2014)*, pages 1–6 – Poster presentation**

Leroi, A., Mauch, M., Savage, P., Benetos, E., Bello, J., Pantelli, M., Six, J., and Weyde, T. (2015). The Deep History of Music Project. In *Proceedings of the 5th International Folk Music Analysis Workshop (FMA 2015)* – Poster presentation

Van Dyck, E., Six, J., and Leman, M. (2016). The relaxing effect of tempo on music-aroused heart rate. In *Proceedings of the 14th International Conference for Music Perception and Cognition (ICMPC 2014)* – Poster presentation

Desmet, F., Lesaffre, M., Six, J., Ehrlé, N., and Samson, S. (2017). Multimodal analysis of synchronization data from patients with dementia. In *Proceedings of the 25th Anniversary Conference of the European Society for the Cognitive Sciences of Music (ESCOM 2017)* – Poster presentation

Six, J., Arens, L., Demoor, H., Kint, T., and Leman, M. (2017). Regularity and asynchrony when tapping to tactile, auditory and combined pulses. In *Proceedings of the 25th Anniversary Conference of the European Society for the Cognitive Sciences of Music (ESCOM 2017)* – Oral presentation

Bressan, F., Six, J., and Leman, M. (2017b). Applications of duplicate detection: linking meta-data and merging music archives. The experience of the IPEM historical archive of electronic music. In *Proceedings of 4th International Digital Libraries for Musicology workshop (DLfM 2017)*, page submitted, Shanghai (China). ACM Press – Oral presentation

de Valk, R., Volk, A., Holzapfel, A., Pikrakis, A., Kroher, N., and Six, J. (2017). Mirchiving: Challenges and opportunities of connecting mir research and digital music archives. In *Proceedings of the 4th International Digital Libraries for Musicology workshop (DLfM 2017)* – Poster presentation

**Six, J. and Leman, M. (2017). A framework to provide fine-grained time-dependent context for active listening experiences. In *Proceedings of the AES Conference on Semantic Audio 2017*. AES – Poster presentation**

**Six, J., Bressan, F., and Leman, M. (In press – 2018b). Applications of duplicate detection in music archives: From meta-**

**data comparison to storage optimisation - The case of the Belgian Royal Museum for Central Africa. In *Proceedings of the 13th Italian Research Conference on Digital Libraries (IRCDL 2018)* – Presentation type undecided**

## A.3 List of lectures and discussions

Panel discussion, 2012: "*Technological challenges for the computational modelling of the world's musical heritage*" ", Folk Music Analysis Conference 2012 – FMA 2012, organizers: Polina Proutskova and Emilia Gomez, Seville, Spain

Guest lecture, 2012: Non-western music and digital humanities, for: "*Studies in Western Music History: Quantitative and Computational Approaches to Music History*", MIT., Boston, U.S.

Guest lecture, 2011: "*Presenting Tarsos, a software platform for pitch analysis*", Electrical and Electronics Eng.Dept. IYTE, Izmir, Turkey

Workshop 2017: "*Computational Ethnomusicology – Methodologies for a new field*", Leiden, The Netherlands

Guest lectures, A002301 (2016–2017 and 2017-2018) "*Grondslagen van de muzikale acoustica en sonologie*" – Theory and Practice sessions together with dr. Pieter-Jan Maes, UGent

## A.4 List of software systems

This dissertation has output in the form of scientific, peer reviewed articles but also considerable output in the form of research software systems and source code. This section lists research software with a small descriptions and links where source code and further information can be found.

### A.4.1 Panako: an acoustic fingerprinting framework

Panako is an extendable acoustic fingerprinting framework. The aim of acoustic fingerprinting is to find small audio fragments in large audio databases. Panako contains several acoustic fingerprinting algorithms to make comparison between them easy. The main Panako algorithm uses key points in a Constant-Q spectrogram as a fingerprint to allow pitch-shifting, time-stretching and speed modification. The aim of Panako is to serve as a platform for research on Acoustic Fingerprinting Systems wile striving to be applicable in small to medium scale situations.

| | |
|---|---|
| Described in | Six and Leman (2014) |
| Lisence | AGPL |
| Repository | `https://github.com/JorenSix/Panako` |
| Downloads | `https://0110.be/releases/Panako`[1] |
| Website | `https://panako.be` |

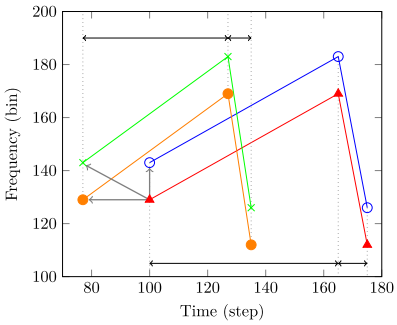



Figure A.1: A fingerprint as a combination of three spectral peaks in Panako and various modifications.

### A.4.2 Tarsos: software for pitch analysis

Tarsos is a software tool to analyze and experiment with pitch organization in all kinds of musics. Most of the analysis is done using pitch histograms and octave reduced pitch class histograms. Tarsos has an intuitive user interface and contains a couple of command line programs to analyze large sets of music.

| | |
|---|---|
| Described in | Six et al. (2013) and Six and Cornelis (2011) |
| Lisence | AGPL |
| Repository | `https://github.com/JorenSix/Tarsos` |
| Downloads | `https://0110.be/releases/Tarsos`[1] |
| Website | `https://0110.be/tags/Tarsos` |

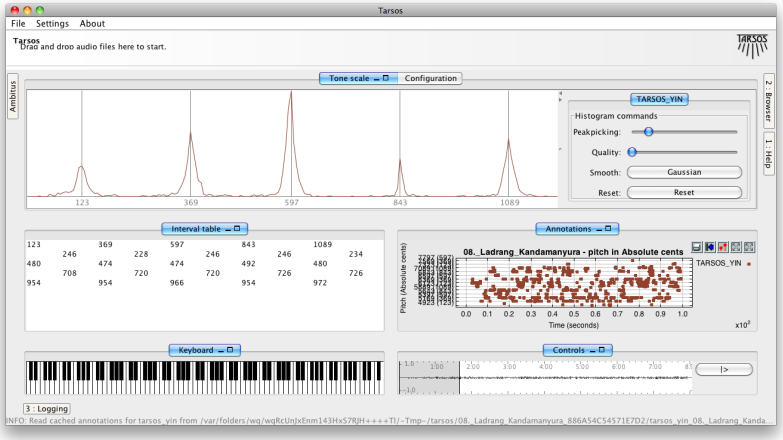



Figure A.2: Pitch analysis with Tarsos

### A.4.3 TarsosDSP: a Java library for audio processing

TarsosDSP is a Java library for audio processing. Its aim is to provide an easy-to-use interface to practical music processing algorithms implemented, as simply as possible, in pure Java and without any other external dependencies. TarsosDSP features an implementation of a percussion onset detector and a number of pitch detection algorithms: YIN, the Mcleod Pitch method and "Dynamic Wavelet Algorithm Pitch Tracking" algorithm. Also included is a Goertzel DTMF decoding algorithm, a time stretch algorithm (WSOLA), resampling, filters, simple synthesis, some audio effects, a pitch shifting algorithm and wavelet filters.

| | |
|---|---|
| Described in | Six et al. (2014) |
| Lisence | GPL |
| Repository | `https://github.com/JorenSix/TarsosDSP` |
| Downloads | `https://0110.be/releases/TarsosDSP` |
| Website | `https://0110.be/tags/TarsosDSP` |

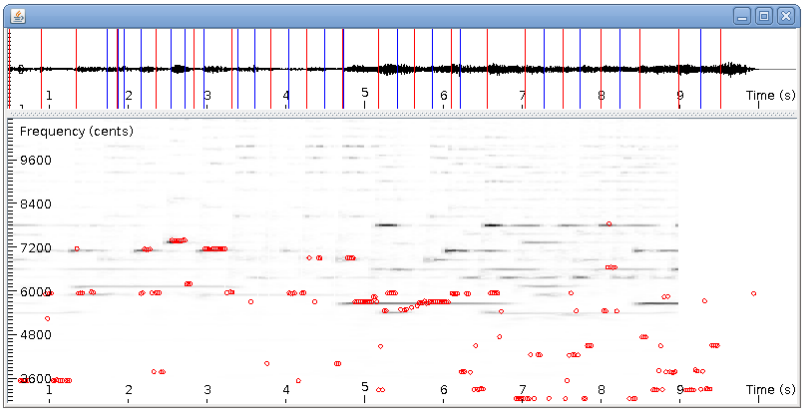



Figure A.3: Visualization of audio features extracted with TarsosDSP.

### A.4.4 SyncSink: synchronize multimodal data

SyncSink is able to synchronize video and audio recordings of the same event. As long as some audio is shared between the multimedia files a reliable synchronization solution will be proposed. SyncSink is ideal to synchronize video recordings of the same event by multiple cameras or to align a high definition audio recording with a video recording (with less qualitative audio).

SyncSink is also used to facilitate synchronization of multimodal research data e.g. to research the interaction between movement and music.

| | |
|---|---|
| Described in | Six and Leman (2015) |
| Lisence | AGPL |
| Repository | `https://github.com/JorenSix/Panako`[1] |
| Downloads | `https://0110.be/releases/Panako`[1] |
| Website | `http://panako.be`[1] |

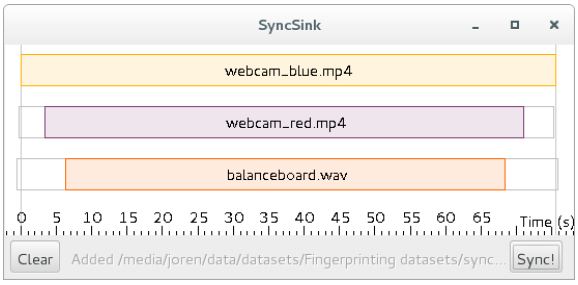



Figure A.4: Screenshot of SyncSink synchronizing three streams.

### A.4.5 TeensyDAQ: an application to visualize analog input signals on a Teensy

TeensyDAQ is a Java application to quickly visualize and record analog signals with a Teensy micro-controller and some custom software. It is mainly useful to quickly get an idea of how an analog sensor reacts to different stimuli. Some of the features of the TeensyDAQ:

Visualize or sonify up to five analog signals simultaneously in real-time. Capture analog input signals with sampling rates up to 8000Hz. Record analog input to a CSV-file and, using drag-and-drop, previously recorded CSV-files can be visualized. While a capture session is in progress you can going back in time and zoom, pan and drag to get a detailed view on your data.

| | |
|---|---|
| Lisence | GPL |
| Repository | `https://github.com/JorenSix/TeensyDAQ` |
| Downloads | `https://0110.be/releases/TeensyDAQ`[1] |

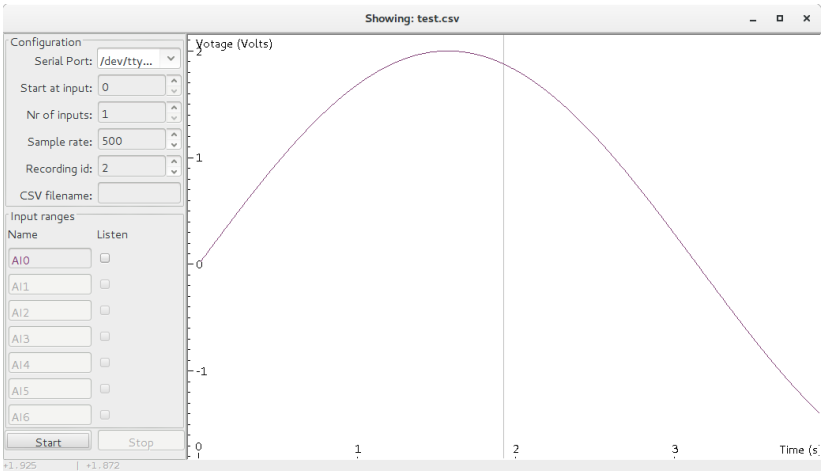



Figure A.5: Screenshot of TeensyDAQ with a light sensor reading.

# B List of figures, tables and acronyms

## B.1 List of figures

## B.2 List of tables

## B.3 List of acronyms

| | |
|---|---|
| **AES** | Audio Engineering Society |
| **AGPL** | Affero General Public License |
| **AMDF** | Average Magnitude Difference |
| **API** | Application Programmers Interface |
| **BER** | Bit Error Rate |
| **BPM** | Beats Per Minute |
| **CBR** | Constant Bit Rate |
| **CC** | Creative Commons |
| **CNRS** | Centre Nationnal de la Recherche Scientifique |
| **CREM** | Centre de Recherche en Ethnomusicologie |
| **CSV** | Comma separated values |
| **DAQ** | Data Aquisition |
| **DB** | Database |
| **DEKKMMA** | Digitalisatie van het Etnomusicologisch Klankarchief van het Koninklijk Museum voor Midden-Afrika |
| **DSP** | Digital Signal Processing |
| **DTMF** | Dual Tone – Multi Frequency |
| **ECG** | Electrocardiogram |
| **EEG** | Electroencephalography |
| **ESCOM** | European Society for the Cognitive Sciences of Music |
| **FFT** | Fast Fourier Transform |
| **FMA** | Folk Music Analysis |
| **FN** | False Negative |
| **FP** | False Positive |
| **FPS** | Frames Per Second |
| **FWO** | Fonds Wetenschappelijk Onderzoek |
| **GNU** | GNU Not Unix |
| **GPL** | General Public License |
| **GPU** | Graphical Processing Unit |
| **GSM** | Global System for Mobile Communication |
| **HD** | Hard Disk |
| **HR** | Heart Rate |
| **IASA** | International Association of Sound and Audiovisual Archives |
| **IRCDL** | Italian Research Conference on Digital Libraries |
| **ISMIR** | International Society of Music Information Retrieval |
| **JNMR** | Journal of New Music Research |
| **JVM** | Java Virtual Machine |
| **KMMA** | Koninklijk Museum voor Midden Afrika |
| **LFO** | Low Frequency Oscillator |
| **MFCC** | Mel-Frequency Cepstral Coefficients |

| | |
|---|---|
| **MIDI** | Musical Instrument Digital Interface |
| **MIR** | Music Information Retrieval |
| **MIMO** | Musical Instruments Museum Online |
| **MIREX** | Music Information Retrieval Evaluation eXchange |
| **MPM** | Mcleod Pitch Method |
| **PCH** | Pitch Class Histogram |
| **PCM** | Pulse Code Modulation |
| **QMUL** | Queen Mary University London |
| **RMCA** | Royal Museum for Central Africa |
| **TN** | True Negative |
| **TP** | True Positive |
| **USB** | Univeral Serial Bus |
| **WSOLA** | Waveform Similarity and OverLap Add |
| **XML** | eXtensible Markup Language |
| **XSD** | XML Schema Definition |
| **XSL** | XML Stylesheet Language |
| **YAAFE** | Yet Another Audio Feature Extractor |

# References


Allamanche, E. (2001). Content-based identification of audio material using mpeg-7 low level description. In *Proceedings of the 2nd International Symposium on Music Information Retrieval (ISMIR 2001)*.

Amatriain, X. (2005). An object-oriented metamodel for digital signal processing with a focus on audio and music.

Amatriain, X., Arumí, P., and Ramírez, M. (2002). CLAM, Yet Another Library for Audio and Music Processing? In *Proceedings of 17th Annual ACM Conference on Object-Oriented Programming, Systems, Languages and Applications (OOPSLA 2002)*.

Arzt, A., Böck, S., and Widmer, G. (2012). Fast identification of piece and score position via symbolic fingerprinting. In Gouyon, F., Herrera, P., Martins, L. G., and Müller, M., editors, *Proceedings of the 13th International Symposium on Music Information Retrieval (ISMIR 2012)*, pages 433–438.

Aucouturier, J.-J. and Bigand, E. (2012). Mel cepstrum & ann ova: The difficult dialog between mir and music cognition. In Gouyon, F., Herrera, P., Martins, L. G., and Müller, M., editors, *Proceedings of the 13th International Symposium on Music Information Retrieval (ISMIR 2012)*, pages 397–402. FEUP Edições.

Aucouturier, J.-J. and Bigand, E. (2013). Seven problems that keep mir from attracting the interest of cognition and neuroscience. *Journal of Intelligent Information Systems*, 41(3):483–497.

Bader, R. (2017). *Springer Handbook of Systematic Musicology*. Springer.

Bannach, D., Amft, O., and Lukowicz, P. (2009). Automatic event-based synchronization of multimodal data streams from wearable and ambient sensors. In *EuroSSC 2009: Proceedings of the European Conference on Smart Sensing and Context*, volume 5741 of *Lecture Note in Computer Science*, pages 135–148. Springer.

Barry, D., Fitzgerald, D., Coyle, E., and Lawlor, B. (2005). Drum Source Separation using Percussive Feature Detection and Spectral Modulation. In *Proceedings of the Irish Signals and Systems Conference (ISSC 2005)*.

Bellettini, C. and Mazzini, G. (2008). Reliable automatic recognition for pitch-shifted audio. In *Proceedings of 17th International Conference on Computer Communications and Networks (ICCCN 2008)*, pages 838–843. IEEE.

Benavides, S. D. D. (2012). Raciocnio de Agentes Musicais Composion Algorotmica, Vida artificiale Interatividade em Sistemas Multiagentes Musicais.

Bentley, J. L. (1975). Multidimensional Binary Search Trees Used for Associative Searching. *Communications of the ACM*, 18(9):509–517.

Berry, D. M. (2012). Introduction: Understanding the digital humanities. In *Understanding digital humanities*, pages 1–20. Springer.

Bertin-Mahieux, T., Ellis, D. P., Whitman, B., and Lamere, P. (2011). The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*.

Bigand, E., Delbé, C., Poulin-Charronnat, B., Leman, M., and Tillmann, B. (2014). Empirical evidence for musical syntax processing? Computer simulations reveal the contribution of auditory short-term memory. *Frontiers in systems neuroscience*, 8.

Bogdanov, D., Wack, N., Gómez, E., Gulati, S., Herrera, P., Mayor, O., Roma, G., Salamon, J., Zapata, J., and Serra, X. (2013). ESSENTIA: an Audio Analysis Library for Music Information Retrieval. In *Proceedings of the 14th International Symposium on Music Information Retrieval (ISMIR 2013)*, pages 493–498.

Boston, G. (1998). *Safeguarding the Documentary Heritage. A guide to Standards, Recommended Practices and Reference Literature Related to the Preservation of Documents of all kinds.* UNESCO.

Bozkurt, B. (2008). An Automatic Pitch Analysis Method for Turkish Maqam Music. *Journal of New Music Research (JNMR)*, 37(1):1–13.

Bressan, F., Canazza, S., Vets, T., and Leman, M. (2017a). Hermeneutic implications of cultural encoding: A reflection on audio recordings and interactive installation art. In Agosti, M., Bertini, M., Ferilli, S., Marinai, S., and Orio, N., editors, *Digital Libraries and Multimedia Archives – Proceedings of the 12th Italian Research Conference on Digital Libraries (IRCDL 2016)*, Procedia - Computer Sciences, pages 47–58. Elsevier.

Bressan, F., Six, J., and Leman, M. (2017b). Applications of duplicate detection: linking meta-data and merging music archives. The experience of the IPEM historical archive of electronic music. In *Proceedings of 4th International Digital Libraries for Musicology workshop (DLfM 2017)*, page submitted, Shanghai (China). ACM Press.

Brossier, P. (2006). Automatic Annotation of Musical Audio for Interactive Applications.

Brown, A. R. and Sorensen, A. C. (2000). Introducing jMusic. In Brown, A. R. and Wilding, R., editors, *Australasian Computer Music Conference*, pages 68–76. ACMA.

Brown, J. and Puckette, M. S. (1992). An Efficient Algorithm for the Calculation of a Constant Q Transform. *Journal of the Acoustical Society of America*, 92(5):2698–2701.

Bujas, G., Bonetti, L., Car, Z., and Vukovic, M. (2017). *Voice Controlled Quiz for People with Hearing Impairment*, pages 166–175. Springer International Publishing, Cham.

Burdick, A., Drucker, J., Lunenfeld, P., Presner, T., and Schnapp, J. (2012). *Digital_Humanities*. Mit Press.

Burgoyne, J. A., Fujinaga, I., and Downie, J. S. (2016). *Music Information Retrieval*, pages 213–228. In Schreibman et al. (2015).

Burk, P. (1998). JSyn - A Real-time Synthesis API for Java. In *Proceedings of the 1998 International Computer Music Conference (ICMC 1998)*. Computer Music Associaciation.

Camurri, A., Coletta, P., Massari, A., Mazzarino, B., Peri, M., Ricchetti, M., Ricci, A., and Volpe, G. (2004). Toward real-time multimodal processing: EyesWeb 4.0. In *AISB 2004 Convention: Motion, Emotion and Cognition*.

Cannam, C. (2008). The Vamp Audio Analysis Plugin API: A Programmer's Guide. `http://vamp-plugins.org/guide.pdf`.

Cannam, C., Landone, C., and Sandler, M. (2010). Sonic visualiser: An open source application for viewing, analysing, and annotating music audio files. In *Open Source Software Competition, ACM*.

Cannam, C., Landone, C., Sandler, M., and Bello, J. (2006). The Sonic Visualiser: A visualisation platform for semantic descriptors from musical signals. In *Proceedings of the 7th International Symposium on Music Information Retrieval (ISMIR 2006)*.

Cano, P., Batlle, E., Kalker, T., and Haitsma, J. (2005). A review of audio fingerprinting. *The Journal of VLSI Signal Processing*, 41:271–284.

Cha, S.-h. (2007). Comprehensive Survey on Distance / Similarity Measures between Probability Density Functions. *International Journal of Mathematical Models and Methods in Applied Sciences*, 1(4):300–307.

Chordia, P. and Rae, A. (2007). Raag Recognition Using Pitch-Class and Pitch-Class Dyad Distributions. In *Proceedings of the 8th International Symposium on Music Information Retrieval (ISMIR 2007)*.

Clarisse, L. P., Martens, J. P., Lesaffre, M., Baets, B. D., Meyer, H. D., and Leman, M. (2002). An Auditory Model Based Transcriber of Singing Sequences. In *Proceedings of the 3th International Symposium on Music Information Retrieval (ISMIR 2002)*, pages 116–123.

Cohen, D. J., Frisch, M., Gallagher, P., Mintz, S., Sword, K., Taylor, A. M., Thomas, W. G., and Turkel, W. J. (2008). Interchange: The promise of digital history. *The Journal of American History*, 95(2):452–491.

Collaboration, O. S. et al. (2015). Estimating the reproducibility of psychological science. *Science*, 349(6251):aac4716.

Cook, N. (2013). *Beyond the score: Music as performance*. Oxford University Press.

Coorevits, E. and Moelants, D. (2016). Tempo in baroque music and dance. *Music Perception: An Interdisciplinary Journal*, 33(5):523–545.

Coover, B. and Han, J. (2014). A power mask based audio fingerprint. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1394–1398.

Cornelis, O. (2013). *Exploring the symbiosis of Western and non-Western music: a study based on computational ethnomusicology and contemporary music composition*. PhD thesis, Ghent University.

Cornelis, O., De Caluwe, R., Detré, G., Hallez, A., Leman, M., Matthé, T., Moelants, D., and Gansemans, J. (2005). Digitisation of the ethnomusicological sound archive of the RMCA. *IASA Journal*, 26:35–44.

Cornelis, O., Lesaffre, M., Moelants, D., and Leman, M. (2010a). Access to ethnic music: Advances and perspectives in content-based music information retrieval. *Signal Processing*, 90(4):1008 – 1031. Special Section: Ethnic Music Audio Documents: From the Preservation to the Fruition.

Cornelis, O., Lesaffre, M., Moelants, D., and Leman, M. (2010b). Access to ethnic music: Advances and perspectives in content-based music information retrieval. *Signal Processing*, 90(4):1008–1031.

Cornelis, O. and Six, J. (2012). Towards the Tangible: Mircotonal Scale Exploration in Central-African Music. In *Proceedings of the Analytical Approaches to World Music Conference, AAWM 2012*.

Cornelis, O., Six, J., Holzapfel, A., and Leman, M. (2013). Evaluation and recommendation of pulse and tempo annotation in ethnic music. *Journal of New Music Research (JNMR)*, 42(2).

Cotton, C. V. and Ellis, D. P. W. (2010). Audio fingerprinting to identify multiple videos of an event. In *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pages 2386–2389. IEEE.

Crouch, S., Hong, N. C., Hettrick, S., Jackson, M., Pawlik, A., Sufi, S., Carr, L., Roure, D. D., Goble, C., and Parsons, M. (2013). The software sustainability institute: Changing research software attitudes and practices. *Computing in Science & Engineering*, 15(6):74–80.

de Cheveigné, A. and Hideki, K. (2002). YIN, a Fundamental Frequency Estimator for Speech and Music. *The Journal of the Acoustical Society of America*, 111(4):1917–1930.

de Valk, R., Volk, A., Holzapfel, A., Pikrakis, A., Kroher, N., and Six, J. (2017). Mirchiving: Challenges and opportunities of connecting mir research and digital music archives. In *Proceedings of the 4th International Digital Libraries for Musicology workshop (DLfM 2017)*.

Desmet, F., Lesaffre, M., Six, J., Ehrlé, N., and Samson, S. (2017). Multimodal analysis of synchronization data from patients with dementia. In *Proceedings of the 25th Anniversary Conference of the European Society for the Cognitive Sciences of Music (ESCOM 2017)*.

Dixon, S. (2001). Automatic Extraction of Tempo and Beat From Expressive Performances. *Journal of New Music Research (JNMR)*, 30(1):39–58.

Dixon, S. and Widmer, G. (2005). Match: A music alignment tool chest. In *Proceedings of the 6th International Symposium on Music Information Retrieval (ISMIR 2005)*, pages 492–497.

Downie, J. S. (2004). The scientific evaluation of music information retrieval systems: Foundations and future. *Computer Music Journal*, 28(2):12–23.

Duckles, V. and Pasler, J. (2007). Musicology, § 1: The nature of musicology. *Grove Music Online*, 46710.

Elliott, M. T., Wing, A., and Welchman, A. (2010). Multisensory cues improve sensorimotor synchronisation. *European Journal of Neuroscience*, 31(10):1828–1835.

Ellis, D., Whitman, B., and Porter, A. (2011). Echoprint - an open music identification service. In *Proceedings of the 12th International Symposium on Music Information Retrieval (ISMIR 2011)*.

Fenet, S., Richard, G., and Grenier, Y. (2011). A Scalable Audio Fingerprint Method with Robustness to Pitch-Shifting. In *Proceedings of the 12th International Symposium on Music Information Retrieval (ISMIR 2011)*, pages 121–126.

Fischinger, T. (2013). Preface by the guest editor of the special issue.

Flexer, A., Schnitzer, D., and Schlueter, J. (2012). A MIREX meta-analysis of hubness in audio music similarity. In Gouyon, F., Herrera, P., Martins, L. G., and Müller, M., editors, *Proceedings of the 13th International Society for Music Information Retrieval Conference, ISMIR 2012, Mosteiro S.Bento Da Vitória, Porto, Portugal, October 8-12, 2012*, pages 175–180. FEUP Edições.

Fuller, C., Mallinckrodt, L., Maat, B., Baskent, D., and Free, R. (2013). Music and quality of life in early-deafened late-implanted adult cochlear implant users. *Otology & Neurotology*, 34(6):1041–1047.

Gedik, A. C. and Bozkurt, B. (2010). Pitch-frequency Histogram-based Music Information Retrieval for Turkish Music. *Signal Processing*, 90(4):1049–1063. Special Section: Ethnic Music Audio Documents: From the Preservation to the Fruition.

Godøy, R. I. and Leman, M. (2010). *Musical gestures: Sound, movement, and meaning*. Routledge.

Gold, M. K. (2012). *Debates in the digital humanities*. University of Minnesota Press.

Goto, M. (2001). An audio-based real-time beat tracking system for music with or without drum-sounds. *Journal of New Music Research*, 30(2):159–171.

Gowing, M., Kelly, P., O'Connor, N. E., Concolato, C., Essid, S., Feuvre, J. L., Tournemenne, R., Izquierdo, E., Kitanovski, V., Lin, X., and Zhang, Q. (2011). Enhanced visualisation of dance performance from automatically synchronised multimodal recordings. In Candan, K. S., Panchanathan, S., Prabhakaran, B., Sundaram, H., Chi Feng, W., and Sebe, N., editors, *ACM Multimedia*, pages 667–670. ACM.

Gómez, E. (2012). Teaching MIR: Educational Resources Related To Music Information Retrieval. In *Proceedings of the 13th International Symposium on Music Information Retrieval (ISMIR 2012)*. International Society for Music Information Retrieval.

Gómez, E. and Bonada, J. (2008). Automatic Melodic Transcription of Flamenco Singing. In *Proceedings of 4th Conference on Interdisciplinary Musicology (CIM 2008)*.

Gómez, E., Herrera, P., and Gómez-Martin, F. (2013). Computational ethnomusicology: perspectives and challenges. *Journal of New Music Research*, 42(2):111–112.

Haitsma, J. and Kalker, T. (2002). A highly robust audio fingerprinting system. In *Proceedings of the 3th International Symposium on Music Information Retrieval (ISMIR 2002)*.

Haitsma, J. and Kalker, T. (2003). A highly robust audio fingerprinting system with an efficient search strategy. *Journal of New Music Research*, 32(2):211–221.

Halmos, I. (1978). *Computational Ethnomusicology in Hungary in 1978*. Ann Arbor, MI: Michigan Publishing, University of Michigan Library.

Hannon, E. E. and Trehub, S. E. (2005). Tuning in to musical rhythms: Infants learn more readily than adults. *Proceedings of the National Academy of Sciences of the United States of America*, 102(35):12639–12643.

Hayles, N. K. (2012). How we think: Transforming power and digital technologies. In *Understanding digital humanities*, pages 42–66. Springer.

Henbing, L. and Leman, M. (2007). A Gesture-based Typology of Sliding-tones in Guqin Music. *Journal of New Music Research (JNMR)*, 36(2):61–82.

Herre, J., Hellmuth, O., and Cremer, M. (2002). Scalable robust audio fingerprinting using mpeg-7 content description. In *Multimedia Signal Processing, 2002 IEEE Workshop on*, pages 165–168. IEEE.

Higginson, A. D. and Munafò, M. R. (2016). Current incentives for scientists lead to underpowered studies with erroneous conclusions. *PLoS biology*, 14(11):e2000995.

Hochenbaum, J. and Kapur, A. (2012). Nuance: A software tool for capturing synchronous data streams from multimodal musical systems. In *International Computer Music Conference*, pages 1 – 6. ICMC.

Honing, H. (2004). The comeback of systematic musicology: new empiricism and the cognitive revolution.

Honingh, A. and Bod, R. (2011). In Search of Universal Properties of Musical Scales. *Journal of New Music Research (JNMR)*, 40(1):81–89.

IASA-TC 04 (2004). *Guidelines on the Production and Preservation of Digital Objects*. IASA Technical Committee.

IFLA - Audiovisual and Multimedia Section (2002). Guidelines for digitization projects: for collections and holdings in the public domain, particularly those held by libraries and archives. Technical report, International Federation of Library Associations and Institutions (IFLA), Paris (France).

Jackson, M., Crouch, S., and Baxter, R. (2011). Software evaluation: criteria-based assessment. *Software Sustainability Institute*.

Jaimovich, J. and Knapp, B. (2010). Synchronization of multimodal recordings for musical performance research. In Beilharz, K., Bongers, B., Johnston, A., and Ferguson, S., editors, *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, pages 372–374.

Juillerat, N., Muller Arisona, S., and Schubiger-Banz, S. (2007). Real-time, low latency audio processing in java. In *Proceedings of the International Computer Music Conference (ICMC 2007)*, pages 99–102.

Kirschenbaum, M. (2012). What is digital humanities and what's it doing in english departments? *Debates in the digital humanities*, 3.

Kirschenbaum, M. (2016). *Track Changes*. Harvard University Press.

Klapuri, A. (2003). Multiple Fundamental Frequency Estimation Based on Harmonicity and Spectral Smoothness. *IEEE Transactions on Speech and Audio Processing*, 11(6):804–816.

Klapuri, A. and Davy, M. (2006). *Signal Processing Methods for Music Transcription*. Springer.

Knorr-Cetina, K. (1999). *Epistemic Cultures: How the Sciences Make Knowledge*. Harvard University Press.

Knuth, D. E. (1979). *TEX and METAFONT: New directions in typesetting*. American Mathematical Society.

Koelsch, S., Jentschke, S., Sammler, D., and Mietchen, D. (2007). Untangling syntactic and sensory processing: An erp study of music perception. *Psychophysiology*, 44(3):476–490.

Krishnaswamy, A. (2004a). Melodic Atoms for Transcribing Carnatic Music. In *Proceedings of the 5th International Symposium on Music Information Retrieval (ISMIR 2004)*.

Krishnaswamy, A. (2004b). Multi-dimensional Musical Atoms in South-Indian Classical Music. In *Proceedings of the 8th International Conference on Music Perception & Cognition (ICMPC 2004)*.

Krumhansl, C. L. (1990). Tonal Hierarchies and Rare Intervals in Music Cognition. *Music Perception*, 7(3):309–324.

Krumhansl, C. L. and Shepard, R. N. (1979). Quantification of the Hierarchy of Tonal Functions Within a Diatonic Context. *Journal of Experimental Psychology: Human Perception and Performance*, (5):579–594.

Larson, E. and Maddox, R. (2005). Real-Time Time-Domain Pitch Tracking Using Wavelets. Self-published.

Lazzarini, V. (2001). Sound Processing with the SndObj Library: An Overview. In *Proceedings of the 4th International Conference on Digital Audio Effects (DAFX 2001)*, pages 6–8.

Lebrun, M.-A., Moreau, P., McNally-Gagnon, A., Goulet, G. M., and Peretz, I. (2012). Congenital amusia in childhood: a case study. *Cortex*, 48(6):683–688.

Lehman, P. L. and Yao, s. B. (1981). Efficient locking for concurrent operations on b-trees. *ACM Transactions Database Systems*, 6(4):650–670.

Leman, M. (2007). *Embodied Music Cognition and Mediation Technology*. The MIT Press.

Leman, M. (2008). Systematic musicology at the crossroads of modern music research. In Schneider, A., editor, *Systematic and Comparative Musicology: Concepts, Methods, Findings. Hamburger Jahrbuch*, pages 89–115.

Leman, M. (2016). *The expressive moment: How interaction (with music) shapes human empowerment*. MIT press.

Leman, M., Lesaffre, M., and Tanghe, K. (2001). Introduction to the ipem toolbox for perception-based music analysis. *Mikropolyphonie-The Online Contemporary Music Journal*, 7.

Leman, M., Moelants, D., Varewyck, M., Styns, F., van Noorden, L., and Martens, J.-P. (2013). Activating and relaxing music entrains the speed of beat synchronized walking. *PLoS ONE*, 8(7):e67932.

Leman, M. and Schneider, A. (1997). Origin and nature of cognitive and systematic musicology: An introduction. *Music, Gestalt, and Computing*, pages 11–29.

Leonelli, S. (2016). Locating ethics in data science: responsibility and accountability in global and distributed knowledge production systems. *Philosophical Trasnactions of the Royal Society A*, 374.

Leonelli, S. and Ankeny, R. A. (2015). Repertoires: How to transform a project into a research community. *BioScience*, 65(7):701–708.

Leroi, A., Mauch, M., Savage, P., Benetos, E., Bello, J., Pantelli, M., Six, J., and Weyde, T. (2015). The Deep History of Music Project. In *Proceedings of the 5th International Folk Music Analysis Workshop (FMA 2015)*.

Lesaffre, M. (2006). *Music information retrieval: conceptuel framework, annotation and user behaviour*. PhD thesis, Ghent University.

Lesaffre, M., Maes, P.-J., and Leman, M. (2017). *The Routledge Companion to Embodied Music Interaction*. Taylor & Francis.

Lesley Mearns, Emmanouil Benetos, S. D. (2011). Automatically Detecting Key Modulations in J.S. Bach Chorale Recordings. In *Proceedings of the Sound Mucic and Computing Conference (SMC 2011)*.

Levin, N., Leonelli, S., Weckowska, D., Castle, D., and Dupré, J. (2016). How do scientists define openness? Exploring the relationship between open science policies and research practice. *Bulletin of Science, Technology and Society*, 36(2):128–141.

Lorenzoni, V., Van Dyck, E., and Leman, M. (2017). Effect of music synchronization on runners foot strike impact. In *proceedings of the 25th Anniversary Conference of the European Society for the Cognitive Sciences of Music (ESCOM 2017)*, pages 118–122. Ghent University.

Loughran, R., Walker, J., O'Neill, M., and O'Farrell, M. (2008). The use of mel-frequency cepstral coefficients in musical instrument identification. In *ICMC*. Citeseer.

Lv, Z., Esteve, C., Chirivella, J., and Gagliardo, P. (2015). Clinical feedback and technology selection of game based dysphonic rehabilitation tool. In *2015 9th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth)*, pages 253–256.

Maes, P.-J., Lorenzoni, V., Moens, B., Six, J., Bressan, F., Schepers, I., and Leman, M. (Submitted - 2018). Embodied, participatory sensemaking in digitally-augmented music practices: Theoretical principles and the artistic case "soundbikes". *Critical Arts*.

Malekesmaeili, M. and Ward, R. K. (2013). A local fingerprinting approach for audio copy detection. *Computing Research Repository (CoRR)*, abs/1304.0793.

Marmel, F., Tillmann, B., and Delbé, C. (2010). Priming in melody perception: tracking down the strength of cognitive expectations. *Journal of Experimental Psychology: Human Perception and Performance*, 36(4):1016.

Mathieu, B., Essid, S., Fillon, T., Prado, J., and Richard, G. (2010). YAAFE, an Easy to Use and Efficient Audio Feature Extraction Software. In *Proceedings of the 11th International Symposium on Music Information Retrieval (ISMIR 2010)*, pages 441–446. International Society for Music Information Retrieval.

Mauch, M. and Ewert, S. (2013). The audio degradation toolbox and its application to robustness evaluation. In de Souza Britto Jr., A., Gouyon, F., and Dixon, S., editors, *Proceedings of the 14th International Society for Music Information Retrieval Conference, ISMIR 2013, Curitiba, Brazil, November 4-8, 2013*, pages 83–88.

Mayor, O., Llimona, Q., Marchini, M., Papiotis, P., and Maestre, E. (2013). RepoVIZZ: A framework for remote storage, browsing, annotation, and exchange of multi-modal data. In *Proceedings of the 21st ACM international conference on Multimedia*, pages 415–416. ACM.

McCarty, W. (2005). *Humanities Computing*. Palgrave Macmillan.

McDermott, H. J. (2004). Music perception with cochlear implants: a review. *Trends in amplification*, 8(2):49–82.

McEnnis, D., McKay, C., and Fujinaga, I. (2005). jAudio: A Feature Extraction Library. In *Proceedings of the 6th International Symposium on Music Information Retrieval (ISMIR 2005)*.

McLeod, P. (2009). Fast, accurate pitch detection tools for music analysis.

McLeod, P. and Wyvill, G. (2005). A Smarter Way to Find Pitch. In *Proceedings of the International Computer Music Conference (ICMC 2005)*.

Merz, E. (2011). *Sonifying Processing: The Beads Tutorial*. CreateSpace.

Mesirov, J. P. (2010). Accessible reproducible research. *Science*, 327(5964):415–416.

Mills III, J. A., Fede, D. D., and Brix, N. (2010). Music programming in minim. In *Proceedings of the New Interfaces for Musical Expression++ Conference (NIME++)*.

Moelants, D., Cornelis, O., and Leman, M. (2009). Exploring african tone scales. In *Proceedings of the 10th International Symposium on Music Information Retrieval (ISMIR 2009)*.

Moelants, D., Cornelis, O., Leman, M., Matthé, T., Hallez, A., Caluwe, R. D., and Gansemans, J. (2007). Problems and Opportunities of Applying Data- and Audio-Mining Techniques to Ethnic Music. *Journal of Intangible Heritage*, 2:57–69.

Moens, B., Muller, C., van Noorden, L., Franěk, M., Celie, B., Boone, J., Bourgois, J., and Leman, M. (2014). Encouraging spontaneous synchronisation with d-jogger, an adaptive music player that aligns movement and music. *PLoS ONE*, 9(12):1–40.

Moretti, F. (2005). *Graphs, maps, trees: abstract models for a literary history*. Verso.

Müller, M. (2015). *Fundamentals of Music Processing - Audio, Analysis, Algorithms, Applications*. Springer.

Nobutaka, O., Miyamoto, K., Kameoka, H., Roux, J. L., Uchiyama, Y., Tsunoo, E., Nishimoto, T., and Sagayama, S. (2010). Harmonic and percussive sound separation and its application to MIR-related tasks. In Ras and Wieczorkowska (2010).

Nosek, B. A., Alter, G., Banks, G. C., Borsboom, D., Bowman, S. D., Breckler, S. J., Buck, S., Chambers, C. D., Chin, G., Christensen, G., Contestabile, M., Dafoe, A., Eich, E., Freese, J., Glennerster, R., Goroff, D., Green, D. P., Hesse, B., Humphreys, M., Ishiyama, J., Karlan, D., Kraut, A., Lupia, A., Mabry, P., Madon, T., Malhotra, N., Mayo-Wilson, E., McNutt, M., Miguel, E., Paluck, E. L., Simonsohn, U., Soderberg, C., Spellman, B. A., Turitto, J., VandenBos, G., Vazire, S., Wagenmakers, E. J., Wilson, R., and Yarkoni, T. (2015). Promoting an open research culture. *Science*, 348(6242):1422–1425.

Ogle, J. and Ellis, D. P. W. (2007). Fingerprinting to identify repeated sound events in long-duration personal audio recordings. In *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*.

Omar, R., Hailstone, J. C., Warren, J. E., Crutch, S. J., and Warren, J. D. (2010). The cognitive organization of music knowledge: a clinical analysis. *Brain*, 133(4):1200–1213.

Orio, N. (2016). Searching and classifying affinities in a web music collection. In *Italian Research Conference on Digital Libraries (IRCDL 2016)*, pages 59–70. Springer.

Ouali, C., Dumouchel, P., and Gupta, V. (2014). A robust audio fingerprinting method for content-based copy detection. In *Content-Based Multimedia Indexing (CBMI), 2014 12th International Workshop on*, pages 1–6. IEEE.

Parncutt, R. (2007). Systematic musicology and the history and future of western musical scholarship. *Journal of interdisciplinary music studies*, 1(1):1–32.

Pashler, H. and Wagenmakers, E.-J. (2012). Editors' introduction to the special section on replicability in psychological science a crisis of confidence? *Perspectives on Psychological Science*, 7(6):528–530.

Peeters, G. and Fort, K. (2012). Towards a (better) definition of the description of annotated mir corpora. In *ISMIR*, pages 25–30. Citeseer.

Peng, R. D. (2011). Reproducible research in computational science. *Science*, 334(6060):1226–1227.

Phillips-Silver, J., Toiviainen, P., Gosselin, N., Piché, O., Nozaradan, S., Palmer, C., and Peretz, I. (2011). Born to dance but beat deaf: a new form of congenital amusia. *Neuropsychologia*, 49(5):961–969.

Plapous, C., Berrani, S.-A., Besset, B., and Rault, J.-B. (2017). A low-complexity audio fingerprinting technique for embedded applications. *Multimedia Tools and Applications*, pages 1–20.

Pope, S. T. and Ramakrishnan, C. (2003). The create signal library (Sizzle): design, issues and applications. In *Proceedings of the 2003 International Computer Music Conference (ICMC 2003)*.

Porter, A., Bogdanov, D., Kaye, R., Tsukanov, R., and Serra, X. (2015). Acousticbrainz: a community platform for gathering music information obtained from audio. In *International Society for Music Information Retrieval Conference (ISMIR'15)*.

Ramona, M., Fenet, S., Blouet, R., Bredin, H., Fillon, T., and Peeters, G. (2012). A public audio identification evaluation framework for broadcast monitoring. *Applied Artificial Intelligence*, 26(1-2):119–136.

Ramona, M. and Peeters, G. (2013). AudioPrint: An efficient audio fingerprint system based on a novel cost-less synchronization scheme. In *Proceedings of the 2013 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP 2013)*, pages 818–822.

Ras, Z. W. and Wieczorkowska, A., editors (2010). *Advances in Music Information Retrieval*, volume 274 of *Studies in Computational Intelligence*. Springer.

Reagan, A. J., Mitchell, L., Kiley, D., Danforth, C. M., and Dodds, P. S. (2016). The emotional arcs of stories are dominated by six basic shapes. *EPJ Data Science*, 5(1):31.

Reed, D. B. (2016). *Abidjan USA: Music, Dance, and Mobility in the Lives of Four Ivorian Immigrants*. Indiana University Press.

Reyes, G., Zhang, D., Ghosh, S., Shah, P., Wu, J., Parnami, A., Bercik, B., Starner, T., Abowd, G. D., and Edwards, W. K. (2016). Whoosh: Non-voice acoustics for low-cost, hands-free, and rapid input on smartwatches. In *Proceedings of the 2016 ACM International Symposium on Wearable Computers*, ISWC 2016, pages 120–127, New York, NY, USA. ACM.

Ross, M. J., Shaffer, H. L., Cohen, A., Freudberg, R., and Manley, H. J. (1974). Average Magnitude Difference Function Pitch Extractor. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 22(5):353–362.

Sandulescu, V., Andrews, S., Ellis, D., Dobrescu, R., and Martinez-Mozos, O. (2015). Mobile app for stress monitoring using voice features. In *E-Health and Bioengineering Conference (EHB), 2015*, pages 1–4. IEEE.

Scavone, G. P. and Cook, P. R. (2005). RTMidi, RTAudio, and a Synthesis Toolkit (STK) update. In *Proceedings of the International Computer Music Conference (ICMC 2005)*.

Schneider, A. (2001). Sound, Pitch, and Scale: From "Tone Measurements" to Sonological Analysis in Ethnomusicology. *Ethnomusicology*, 45(3):489–519.

Schreibman, S., Siemens, R., and Unsworth, J. (2008). *A companion to digital humanities*. John Wiley & Sons.

Schreibman, S., Siemens, R., and Unsworth, J. (2015). *A New Companion to Digital Humanities*. John Wiley & Sons.

Schwartz, D. A. and Purves, D. (2004). Pitch is Determined by Naturally Occurring Periodic Sounds. *Hearing Research*, 194(1):31–46.

Serra, X. (2011). A multicultural approach in music information research. In *Proceedings of the 12th International Symposium on Music Information Retrieval (ISMIR 2011)*, pages 151–156.

Sethares, W. A. (2005). *Tuning Timbre Spectrum Scale*. Springer, 2 edition.

Shrestha, P., Barbieri, M., and Weda, H. (2007). Synchronization of multi-camera video recordings based on audio. In *Proceedings of the 15th International Conference on Multimedia*, MULTIMEDIA '07, pages 545–548, New York, NY, USA. ACM.

Six, J., Arens, L., Demoor, H., Kint, T., and Leman, M. (2017). Regularity and asynchrony when tapping to tactile, auditory and combined pulses. In *Proceedings of the 25th Anniversary Conference of the European Society for the Cognitive Sciences of Music (ESCOM 2017)*.

Six, J., Bressan, F., and Leman, M. (In press – 2018a). A case for reproduciblity in MIR. Replication of 'a highly robust audio fingerprinting system'. *Transactions of the International Society for Music Information Retrieval (TISMIR)*.

Six, J., Bressan, F., and Leman, M. (In press – 2018b). Applications of duplicate detection in music archives: From metadata comparison to storage optimisation - The case of the Belgian Royal Museum for

Central Africa. In *Proceedings of the 13th Italian Research Conference on Digital Libraries (IRCDL 2018)*.

Six, J. and Cornelis, O. (2011). Tarsos - a Platform to Explore Pitch Scales in Non-Western and Western Music. In *Proceedings of the 12th International Symposium on Music Information Retrieval (ISMIR 2011)*.

Six, J. and Cornelis, O. (2012a). A Robust Audio Fingerprinter Based on Pitch Class Histograms - Applications for Ethnic Music Archives. In *Proceedings of the Folk Music Analysis conference (FMA 2012)*.

Six, J. and Cornelis, O. (2012b). A robust audio fingerprinter based on pitch class histograms: applications for ethnic music archives. In *International Workshop of Folk Music Analysis, Proceedings*, page 7. Ghent University, Department of Art, music and theatre sciences.

Six, J. and Cornelis, O. (2013). Computer Assisted Transcription of Ethnic Music. In *Proceedings of the 2013 Folk Music Analysis conference (FMA 2013)*.

Six, J., Cornelis, O., and Leman, M. (2013). Tarsos, a modular platform for precise pitch analysis of Western and non-Western music. *Journal of New Music Research*, 42(2):113–129.

Six, J., Cornelis, O., and Leman, M. (2014). TarsosDSP, a real-time audio processing framework in Java. In *Proceedings of the 53rd AES Conference (AES 53rd)*. The Audio Engineering Society.

Six, J. and Leman, M. (2014). Panako - A scalable acoustic fingerprinting system handling time-scale and pitch modification. In *Proceedings of the 15th ISMIR Conference (ISMIR 2014)*, pages 1–6.

Six, J. and Leman, M. (2015). Synchronizing Multimodal Recordings Using Audio-To-Audio Alignment. *Journal of Multimodal User Interfaces*, 9(3):223–229.

Six, J. and Leman, M. (2017). A framework to provide fine-grained time-dependent context for active listening experiences. In *Proceedings of the AES Conference on Semantic Audio 2017*. AES.

Six, J. and Viro, V. (2011). Peachnote Piano: Making MIDI Instruments Social and Smart Using Arduino, Android and Node.js. In *Late breaking Demos at the 12th International Symposium on Music Information Retrieval (ISMIR 2011)*.

Smith, J. and Gosset, P. (1984). A Flexible Sampling-Rate Conversion Method. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP 1984)*, volume 2.

Sonnleitner, R., Arzt, A., and Widmer, G. (2016). Landmark-based audio fingerprinting for dj mix monitoring. In *ISMIR*, pages 185–191.

Sonnleitner, R. and Widmer, G. (2014). Quad-based Audio Fingerprinting Robust To Time And Frequency Scaling. In *Proceedings of the 17th International Conference on Digital Audio Effects (DAFx-14)*.

Sonnleitner, R. and Widmer, G. (2016). Robust quad-based audio fingerprinting. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, PP(99):1–1.

Sowiński, J. and Dalla Bella, S. (2013). Poor synchronization to the beat may result from deficient auditory-motor mapping. *Neuropsychologia*, 51(10):1952–1963.

Stubbe, T. (2013). Geautomatiseerde vorm- en structuuranalyse van muzikale audio.

Sturm, B. L. (2012). Two systems for automatic music genre recognition: What are they really recognizing? In *Proceedings of the second international ACM workshop on Music information retrieval with user-centered and multimodal strategies*, pages 69–74. ACM.

Sturm, B. L. (2016). Revisiting priorities: Improving MIR evaluation practices. In Mandel, M. I., Devaney, J., Turnbull, D., and Tzanetakis, G., editors, *Proceedings of the 17th International Society for Music Information Retrieval Conference, ISMIR 2016, New York City, United States, August 7-11, 2016*, pages 488–494.

Sturm, B. L. and Noorzad, P. (2012). On automatic music genre recognition by sparse representation classification using auditory temporal modulations. *Computer music modeling and retrieval*, pages 379–394.

Sundberg, J. and Tjernlund, P. (1969). Computer Measurements of the Tone Scale in Performed Music by Means of Frequency Histograms. *STL-QPS*, 10(2-3):33–35.

Terras, M., Nyhan, J., and Vanhoutte, E. (2013). *Defining Digital Humanities: A Reader*. Ashgate Publishing Company, Brookfield, VT, USA.

Timm, L., Vuust, P., Brattico, E., Agrawal, D., Debener, S., Büchner, A., Dengler, R., and Wittfoth, M. (2014). Residual neural processing of musical sound features in adult cochlear implant users. *Frontiers in human neuroscience*, 8.

Tsai, T. J. (2016). *Audio Hashprints: Theory & Application*. University of California, Berkeley.

Tzanetakis, G. and Cook, P. (1999). MARSYAS: a Framework for Audio Analysis. *Organized Sound*, 4(3):169–175.

Tzanetakis, G., Ermolinskyi, A., and Cook, P. (2002). Pitch Histograms in Audio and Symbolic Music Information Retrieval. In *Proceedings of the 3th International Symposium on Music Information Retrieval (ISMIR 2002)*, pages 31–38.

Tzanetakis, G., Kapur, A., Schloss, W. A., and Wright, M. (2007). Computational ethnomusicology. *Journal of Interdisciplinary Music Studies*, 1(2).

Van Assche, W. (2016). Realtime signaal synchronisatie met acoustic fingerprinting.

Van Balen, J. (2016). *Audio description and corpus analysis of popular music*. PhD thesis, Utrecht University.

Van Balen, J., Serrà, J., and Haro, M. (2012). Sample identification in hip hop music. In *International Symposium on Computer Music Modeling and Retrieval*, pages 301–312. Springer.

Van Den Berge, P., Six, J., Gerlo, J., De Clerq, D., and Leman, M. (Submitted - 2018). Real-time measures of tibial acceleration as biofeedback on impact intensity during overground running. *Journal of Biomechanics*, 0(0).

Van Den Doel, K. and Pai, D. K. (2001). JASS: A Java Audio Synthesis System for Programmers. In Hiipakka, J., Zacharov, N., and Takala, T., editors, *Proceedings of the 7th International Conference on Auditory Display (ICAD 2001)*, pages 150–154.

Van Dyck, E. and Six, J. (2016). The relaxing effect of tempo on music-aroused heart rate. In *Proceedings of the 14th International Conference for Music Perception and Cognition (ICMPC 2016)*.

Van Dyck, E., Six, J., and Leman, M. (2016). The relaxing effect of tempo on music-aroused heart rate. In *Proceedings of the 14th International Conference for Music Perception and Cognition (ICMPC 2014)*.

Van Dyck, E., Six, J., Soyer, E., Denys, M., Bardijn, I., and Leman, M. (2017). Adopting a music-to-heart rate alignment strategy to measure the impact of music and its tempo on human heart rate. *Musicae Scientiae*, 0(0):10.

Vanhecke, F., Moerman, M., Desmet, F., Six, J., Daemers, K., Raes, G.-W., and Leman, M. (2017). Acoustical properties in Inhaling Singing: a case-study. *Physics in Medicine*.

Vanhoutte, E. (2013). *The Gates of Hell: History and Definition of Digital/ Humanities/ Computing*, pages 119–56. In Terras et al. (2013).

Verhelst, W. and Roelands, M. (1993). An Overlap-Add Technique Based on Waveform Similarity (WSOLA) for High Quality Time-Scale Modification of Speech. In *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP 1993)*, pages 554–557.

von Helmholtz, H. and Ellis, A. J. (1912). *On the Sensations of Tone as a Physiological Basis for the Theory of Music*. Longmans, Green, London, translated and expanded by Alexander J. Ellis, 2nd English edition.

Wager, M. (2011). Entwicklung eines Systems zur automatischen Notentranskription von monophonischem Audiomaterial.

Wang, A. and Culbert, D. (2009). Robust and invariant audio pattern matching. US Patent 7,627,477.

Wang, A. L.-C. (2003). An industrial-strength audio search algorithm. In *Proceedings of the 4th International Symposium on Music Information Retrieval (ISMIR 2003)*, pages 7–13.

Wang, A. L.-C. and Culbert, D. (2002). Robust and invariant audio pattern matching. US Patent US7627477.

Warwick, C., Terras, M., and Nyhan, J. (2012). *Digital humanities in practice*. Facet Publishing.

Wittenburg, P., Brugman, H., Russel, A., Klassmann, A., and Sloetjes, H. (2006). ELAN: A professional framework for multimodality research. In *In Proceedings of Language Resources and Evaluation Conference (LREC)*.

Wright, M., Schloss, W. A., and Tzanetakis, G. (2008). Analyzing afro-cuban rhythms using rotation-aware clave template matching with dynamic programming. In *ISMIR*, pages 647–652.

Zhu, B., Li, W., Wang, Z., and Xue, X. (2010). A novel audio fingerprinting method robust to time scale modification and pitch shifting. In *Proceedings of the international conference on Multimedia (MM 2010)*, pages 987–990. ACM.