University Name

Doctoral Thesis

A content-aware interactive explorer of digital music collections: The Phonos music explorer

Author:
John Smith

Supervisor:
Dr. James Smith

A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

in the

Research Group Name
Department or School Name

February 2015

"Thanks to my solid academic training, today I can write hundreds of words on virtually any topic without possessing a shred of information, which is how I got a good job in journalism."

Dave Barry

UNIVERSITY NAME (IN BLOCK CAPITALS)

Abstract

Faculty Name
Department or School Name

Doctor of Philosophy

A content-aware interactive explorer of digital music collections: The Phonos music explorer

by John Smith

The Thesis Abstract is written here (and usually kept to just this page). The page is kept centered vertically so can expand into the blank space above the title too...

Acknowledgements

The acknowledgements and the people to thank go here, don't forget to include your project advisor...

Contents

Abstract ii

Acknowledgements iii

Contents iv

List of Figures vi

List of Tables vii

Abbreviations viii

1 Introduction 1
- 1.1 The importance of music analysis . . . 1
- 1.2 Phonos Project . . . 3
- 1.3 GiantSteps . . . 3
- 1.4 Purpose of this work . . . 5
- 1.5 Introduction to the problem of Playlist Generation . . . 5
- 1.6 Structure of the dissertation . . . 6

I Background 9

2 Music Analysis Techniques 11
- 2.1 Metadata . . . 11
  - 2.1.1 Vector Space Model . . . 13
  - 2.1.2 Co-Occurence Analysis . . . 13
  - 2.1.3 Frequent Pattern Mining . . . 13
- 2.2 Audio Content Analysis . . . 13
  - 2.2.1 Low-level Descriptors . . . 15
    - 2.2.1.1 MFCC . . . 15
  - 2.2.2 Mid-level Descriptors . . . 15
    - 2.2.2.1 Rhythm . . . 15
    - 2.2.2.2 Tonality . . . 18
  - 2.2.3 High-level Descriptors . . . 20
  - 2.2.4 Main Tools For Extracting Audio Content . . . 20

2.3 Computing Music Similarity with Audio Content Descriptors 21
2.4 Conceptual Differences Between Metadata and Audio Content Information 21

3 Assessing the performance of a music similarity computation system 23
3.1 Literature Review 23

II Methodology 24

4 Case Study: Requirements and Approach 26
4.1 Catalogue of music 26
4.2 Requirements 26
4.3 Design of the system 27

5 Computation of Audio Features 28
5.1 Tools used for feature extraction, features extracted 28
5.2 Similarity computation (fast map) 28

6 The Real-Time Application 29
6.1 Implementation (python server + html client), Gstreamer 29
6.2 Functioning 29

III Results and Discussion 30

7 Evaluation 32

8 Future Work 33

A List of Essentia Features 34

B List of Echonest Features 35

C Phonos: list of songs 36

Bibliography 37

List of Figures

1.1 Phonos 3
1.2 GiantSteps 5

List of Tables

2.1 Main tonal descriptors. 19

Abbreviations

MFCC	**L**ist **A**bbreviations **H**ere
GUI	**L**ist **A**bbreviations **H**ere
HPCP	**L**ist **A**bbreviations **H**ere
MIR	**L**ist **A**bbreviations **H**ere
RS	**R**ecommender **S**ystems
BPM	**R**ecommender **S**ystems
FFT	**R**ecommender **S**ystems

For/Dedicated to/To my…

Chapter 1

Introduction

1.1 The importance of music analysis

The incredible growth of the Web over the last pair of decades has drastically changed many of our habits. One of the areas that have been highly affected by this fast-paced growth is our consumption of multimedia contents: the use of physically-stored content is seeing itself heavily reduced, as we are more and more getting used to the access of huge databases of multimedia content through the Web.
(it would be nice to cite [3] here to present the subject in a more elegant way) Music is one of these fields that have been revolutionized by this trend: the last decade has seen the rise of several Web services (iTunes, Spotify, Pandora, Google Music just to name a few) that offer their users an easy way to access their enormous catalogue of songs. Statistics show an increasing rate of annual growth for each of these services, in both the amount of users and of revenues: now they are among the most used ways of enjoying and discovering music.
However, the transition to this type of services has brought to some new problems. One of them relies on the vastness of these databases: given that users want to easily discover new music suitable to their tastes through intelligently created playlists, a way to reasonably pick songs and artists among the entire catalogue is needed.
This, among others, has been one reason of the rapid growth of **Music Information Retrieval** (MIR), an interdisciplinary research field whose subject is to provide new ways of finding information in music. Main techniques of describing music can be grouped into two categories:

- Metadata (literally *data describing data*), descriptors of music not directly retrieved from the audio signal but instead from external sources [1]
- Audio content descriptors, automatically computed from audio.

When it comes to choosing one method over the other, it becomes clear that both these categories of tools have their own pros and cons. Regarding metadata, major concerns arise from the questionable consistency of the descriptors among the entire catalogue catalogue of music, given that they may have been extracted from several sources. Other concerns also arise from how well they actually describe the audio track. On the other hand, audio content descriptors (especially the low-level ones) may have no musical meaning and therefore they could be hard to understand. Many efforts have be taken in order to improve the methods of information extraction of both these categories. In general, however, audio content descriptors are thought to be more flexible, since they can be easily and equally computed for any track. One advantage of this technique relies on the fact that these kind of descriptors could easily be computed not just for each kind of song, but also for any segment inside of it. This has for example been exploited by *Shazam*, a widely-used smartphone app for music identification that analyzes peaks in the frequency-time spectrum throughout all song length to build a very robust song identification system [1]. Another popular product that performs audio content analysis just for short segments of a song is The Infinite Jukebox[2], a web-application built upon Echonest library and written by Paul Lamere, that allows users to indefinitely listen to the same song, with the playback automatically jumping to points that sound very similar to the current one. The Infinite Jukebox can be considered an application of the so-called *creative-MIR* [6], an emerging area of activity inner to MIR whose subject is to exploit MIR techniques for creative purposes. Other relevant software that exploit Echonest library for similar purposes is Autocanonizer[3] and Wub Machine[4]. However, there aren't many commercial or research-based software tools that exploit this kind of techniques for creative interaction or manipulation of audio tracks at the moment. Probably the most relevant commercial system is Harmonic Mixing Tool[5], that performs audio content analysis on the user's music collection in order to allow a pleasant and harmonic fade when mixing between songs. More recently, the research-based software

[1] There is a lack of agreement on the use of the term metadata, therefore its meaning could be different in other resources. For instance, it may be used to indicate all the data describing an audio file, including the ones derived from some computation on the audio signal itself.

[2] `http://infinitejuke.com`

[3] `http://static.echonest.com/autocanonizer`

[4] `http://thewubmachine.com`

[5] `http://www.idmt.fraunhofer.de/en/Service_Offerings/products_and_technologies/e_h/harmonic_mixing_tool.html`

AutoMashUpper has been developed with the intent of automating generating multi-song mashup[6] while also allowing the user a control over the music generated [7]. WRITE MORE ABOUT AUTOMASH HERE

1.2 Phonos Project

Phonos project[7] is an initiative of the **Music Technology Group** (Universitat Pompeu Fabra, Barcelona) in collaboration with **Phonos Foundation**. Phonos was founded in 1974 by J.M. Mestres Quadreny, Andres Lewin-Richter and Luis Callejo, and for many years it has been the only studio of electroacoustic music in Spain. Many of the electroacoustic musicians in Spain attended the courses of the composer Gabriel Brncic at Phonos. It became Phonos Foundation 1982 and in 1984 it was registered at the Generalitat de Catalunya. In 1994, an agreement of co-operation with Music Technology group was established, with the purpose of promoting cultural activities related to research in the music technology. In 2014, an exhibition at Museum de la Musica has been planned, with the purpose of celebrating the 40th anniversary of Phonos and showing many of the instruments used in the studio, while allowing visitors to listen to the music works produced there during all these years. Given the songs' average length and their complexity, a way for the visitors to quickly and nicely explore a catalogue of songs produced in these 40 years was needed.

phonos
PHONOS FUNDACIÓ PRIVADA

Figure 1.1: Phonos Logo.

1.3 GiantSteps

GiantSteps[8] is a STREP project coordinated by JCP-Consult SAS in France in collaboration with the MTG funded by the European Commission. The aim of this project

[6]A mashup is a composition made of two or more different songs playing together.
[7]`http://phonos.upf.edu/`
[8]`http://www.giantsteps-project.eu/`

is to create the "seven-league boots" for music production in the next decade and beyond, that is, exploiting the latest fields in the field of MIR to make computer music production easier for anyone. Indeed, despite the increasing amount of software and plugins for computer music creation, it's still considered very hard to master these instruments and producing songs[9] because it requires not only musical knowledge but also familiarity with the tools (both software and hardware) that the artist decide to use, and whose way of usage may greatly vary between each other. The GiantSteps project targets three different directions:

- Developing **musical expert agents**, that could provide suggestions from sample to song level, while guiding users lacking inspiration, technical or musical knowledge
- Developing improved **interfaces**, implementing novel visualisation techniques that provide meaningful feedback to enable fast comprehensibility for novices and improved workflow for professionals.
- Developing **low complexity algorithms**, so that the technologies developed can be accessible through low cost portable devices.

Started on November 2013, GiantSteps will last 36 months and the institutions involved are:

- **Music Technology Group**, Universitat Pompeu Fabra, Barcelona, Spain
- **JCP-Consult SAS**, France
- **Johannes Kepler Universität Linz**, Austria
- **Red Bull Music Academy**, Germany
- **STEIM**, Amsterdam, Netherlands
- **Reactable Systems**, Barcelona, Spain
- **Native Instruments**, Germany

[9] "Computer music today is like piloting a jet with all the lights turned off." (S. Jordà). `http://vimeo.com/28963593`

Figure 1.2: GiantSteps Logo.

1.4 Purpose of this work

The purpose of this work is to develop a software to be used by visitors during the exhibition *Phonos, 40 anys de música electrònica a Barcelona* and that allows users to easily explore a medium-sized collection of music. This software is intended to exploit latest MIR findings to create a flow of music, composed of short segments of each song, concatenated in a way that the listener can barely realize of the hops between different songs. The application must also allow users to interact with it in order to have some control over the generation of the playlist; specifically, the user should be able to give a general direction to this flow (through some sliders or others GUI elements) in regards to some relevant music features, in a way that the change in the musical output can be perceived. The application developed is meant to be part of the GiantSteps project and therefore should follow the three guidelines explained in the previous page. In addition to this, given its future use on a public place, the application is required to be easy to use also for non-musicians, as many of the visitors of the exhibition could be.

1.5 Introduction to the problem of Playlist Generation

The problem of playlist generation has already been addressed by many popular music platforms, such as *Last.fm* [10], *Pandora* [11] and *Musicovery* [12]. The main objective of such services is to help users find tracks or artists that are unknown to them and

[10] `http://last.fm`
[11] `http://www.pandora.com`
[12] `http://musicovery.com`

that they may like, providing *personalized radio playlists*. However, a playlist may be defined, in a broad way, just as a sequence of tracks [16] and therefore its use could be more general. For instance, a common use of the term plylist refers to the broadcasting radio playlists, i.e. playlists made by DJs in a radio stations and often involving popular tracks. We can therefore define the problem of playlist generation as follows [16]:

> Given (1) a pool of tracks, (2) a background knowledge database, and (3) some target characteristics of the playlist, create a sequence of tracks fulfilling the target characteristics in the best possible way.

This task is made particularly challenging by the average size of the music database on which the generation of the playlist is needed: already, personal music libraries can be huge [17], hence the corresponding amount of information to be processed in order to the generate the playlist leads to very heavy computational tasks. Depending on the need of the application, these tasks may also be performed offline, although a real-time user interaction should be supported in many cases in order to allow the user to have some control over this generation process (such as in the case study of this work). As we will see in Chapter 2, extracting information from an audio signal is not a trivial task and many algorithms have considerable time-complexity, and this may lead to very long computational times already for the analysis of small-sized catalogues. Playlist generation is a well-known problem inside MIR [18] [19], since this task can be considered as a retrieval task if its definition is limited to the selection of tracks satisfying a user query [16]. Other major topics of MIR also include extraction of features and similarity analysis, that can be seen as a basis for building a playlist generation system [20].

1.6 Structure of the dissertation

This dissertation is organized as follows:

- The first part will at first give an overview regarding music analysis techniques, explaining *metadata*, audio content analysis and the differences between them. Then, common techniques of music similarity computation will be explained.
- The second part will be about the methodology, explaining the different stages of the development, the problems faced and the techniques used. A presentation of the case study will introduce to an explanation of the reasons that lead to prefer the use of some techniques over others.

- Finally, experimental results will be shown, together with some ideas regarding future development of the application.

Part I

Background

Chapter 2

Music Analysis Techniques

The main subject of MIR regards the *extraction and inference of musically meaningful features, indexing of music* (through these features) and the development of *search and retrieval schemes* [2]. In other terms, the main target of MIR is to make all the music over the world easily accessible to the user [2]. During the last two decades, several approaches have been developed, which mainly differ in the music perception category of the features they deal with. These categories generally are: *music content*, *music context*, *user properties* and *user context* [4]. *Music content* deals with aspects that are directly inferred by the audio signal (such as melody, rhythmic structure, timbre) while *music context* refers to aspects that are not directly extracted from the signal but are strictly related to it (for example label[23], artist and genre information [24] [25], year of release [26], lyrics [27] and semantic labels). Regarding the user, the difference between *user context* and *user properties* lies on the stability of aspects of the user himself. The former deals with aspects that are subject to frequent changes (such as mood or social context), while the latter refers to aspects that may be considered constant or slowly changing, for instance his music taste or education [4].
In this chapter, we will focus on the differences between the categories *music content* and *music context.*

2.1 Metadata

By metadata we mean all the descriptors about a track that are not based on the *music content.* Therefore, they are not directly extracted from the audio signal but rather from external sources. They began to be deeply studied since the early 2000s, when first doubts about an upper threshold of the performance of audio content analysis systems arised [5]. Researchers then started exploring the possibility of performing

retrieving tasks on written data that is related to the artist or to the piece.
At first, the techniques were adapted from the Text-IR ones, but it was immediately clear that retrieving music is fairly more complex than retrieving text, because the music retrieved should also satisfy the musical taste of the user who performed the query.
The techniques used in this category may differ both in the sources used for retrieving data and in the way of computing a similarity score, and clearly the performance of a system using metadata for similarity computation is highly affected by both of these factors. Sources may include [12]:

- Manual annotation: description provided by experts; they may be referred to genre, mood, instrumentation, artist relations.
- Collaborative filtering data: data indirectly provided by users of web communities, in the form of user ratings or listening behaviour information.
- Social tags: data directly provided by users of social network of music (such as *Last.fm*[1]) or social games.
- Information automatically mined from the Web. Sources in these cases may include web-pages related to music or microblogs (for instance the very popular Twitter).

The availability of some of them greatly depends on the size of the music collection under consideration; for instance, as manual expert annotations might be very accurate, they would be extremely costly and probably infeasible on large collections [8]. In contrast, collaborative filtering data may be the most studied technique, given that it may be applied to other different fields (such as movies or books recommendation) with just little changes. It is the predominant approach in the field of Recommender Systems (RS) [35] and is mainly focused on user ratings, generally leading to better results [11]. However, some concerns are related to this technique. First, collaborative filtering approaches have not been designed to be used for playlist generation, but mainly for recommending artists or music. Second, the availibility of datasets for user ratings in the field of music is very limited compared to other fields, and research is often based on very small samples [36]. Regarding listening behaviour information, they might be inaccurate since they don't keep track of song durations and of the user activities while listening to music [38]. In addition, there's no way of collecting negative feedback (*dislikes*) through them and, more in general, listening to a specific song doesn't necessarily imply liking that song [12].
Sources are picked also in relation to the subject of the research or of the system, that may be for example a recommendation or a similarity computation system. At this point,

[1] `http://last.fm`

it's important to highlight the difference between the two of them: a recommendation system not only has to find similar music, but has also to take into account the personal taste of the user, and therefore it's generally considered as a basic tool to produce recommendation [22]. In any case, the terms "similarity" and "recommendation" cannot be substituted, given that a good similarity computation system doesn't necessarily equate to a good recommendation system [37]. The computation of similarity may happen through a Vector Space Model (a technique adapted from the Text-IR), co-occurence analysis or frequent-pattern mining. In the next subsections we will briefly explain the characteristics and the performance of these techniques.

2.1.1 Vector Space Model

The main idea of this technique lies on building a bag-of-words representation [2] of a retrieved document, and then computing a term weight vector for each document. It's a frequently used technique in Text-IR (and in Computer Vision) which can safely be used when retrieving web pages related to music, in an attempt of computing similarity. One of the first work in this field [39] provided an analysis of this kind on music-related web pages retrieved with the queries (to the *Google* search engine) "artist" `music review` and "artist" `genre style`, where words such as music and review where added to improve the chances of automatically retrieving webpages related to music.

2.1.2 Co-Occurence Analysis

2.1.3 Frequent Pattern Mining

2.2 Audio Content Analysis

The main idea behind this kind of analysis is to directly extract useful information, through some algorithms (or library of algorithms), from the audio signal itself. The type of content information extracted may greatly vary in relation to the need of the research, but we can mainly distinguish four categories [12]:

- *Timbral* information: related to the overall quality and color of the sound.

[2] A bag-of-words can be basically seen as an extension of a programming language "dictionary": it collects words (that sometimes may just be an abstraction of much more complex features, such as computer vision descriptors) from a document, and then computes the frequency with which each of them appears in the document. Two different documents are considered similar if they contain the same or similar words with a comparable frequency.

- *Temporal* information: related to rhythmic aspects of the composition, such as tempo or length of measures.
- *Tonal* information: directly linked to the frequency analysis of the signal and to the pitch. It can describe what notes are being played or the tonality of a given track.
- *Inferred semantic* information: information inferred (usually through machine learning techniques) from the previous categories, in the attempt of giving a more defined and understable shape to the data collected. This kind of information may include descriptors such as genre, valence or arousal.

Information extracted through this family of techniques may also be categorized in the following way:

- Low-level data: information that has no musical meaning and that, more in general, is not interpretable by humans. Examples of this kind of descriptors are Mel Frequency Cepstral Coefficients (MFCCs) and Zero Crossing Rate (ZCR).
- Mid-level data: information that has musical meaning but that is related to low-level music features. This kind of category mainly includes temporal and tonal descriptors.
- High-level data: corresponding to inferred semantic information.

Many of the studies conducted on the computation of music similarity through audio content descriptors have solely focused on low-level and timbral information, because this has been proved to bring alone to acceptable results with proper similarity measures [13]. However, more recent studies have shown some evidence of advantages in using high-level descriptors [14] [15] and, more in general, the most performant systems use data from all of these categories. When computing low and mid-level descriptors, the procedure requires the following operations:

- Conversion of the signal from stereo to mono, in order to compute all the descriptors for just one signal
- Down-sampling of the signal to improve the performance while computing the descriptors
- Segmentation of the signal into frames, short segments (usually from 512 to 2048 audio samples). Consecutive frames are usually not disjoint: the so-called *hop-size* determines the hop of samples between the beginning of a frame and the next one, and is normally half or a quarter as big as the *frame size*.

- Computation of Fast Fourier Transform, with an appropriate prior windowing technique [3].

The computation of descriptors is then performed on each frame, and finally a single value for each descriptor is computed by the means of some statistical analysis. Mean, median, variance and covariance are the most used statistical tools for calculating representative global values out of the enormous *pool* of values computed in each frame. Some more operations may sometimes be needed, such as de-noising [4] of time-scaling of the signal.

In the next sections, a more detailed look among most important descriptors will be given.

2.2.1 Low-level Descriptors

2.2.1.1 MFCC

2.2.2 Mid-level Descriptors

2.2.2.1 Rhythm

In traditional music notation, there are several notations for tempo. It may be expressed in BPM (beats per minute), MPM (measures per minute; commonly used in ballroom dance music) or by semantic notations indicating a range of BPM; an example of this last category of notations may be the popular system of Italian markings, such as *presto* (168-200 BPM), *andante* (84-90 BPM) or *allegro* (120-128 BPM).
In the field of MIR, accurate notations are needed, therefore semantic annotations are disregarded in favour of more precise notation such as BPM and Onset Rate (OR).

Onset Rate

IDEAS: image of ADSR, some flowchart for a standard onset detection algorithm (look at figures folder) Onsets are generally defined as the beginning of a new musical note, and onset rate is therefore defined as the number of onsets in a time interval. This definition however hides several difficulties: in polyphonic music, nominally simultaneous notes may be spread over tens of seconds, making this definition more blurred [40]. Moreover, several instruments have a long attack time and this makes the task of defining an onset

[3] Although this last step may not be strictly seen as a necessary operation, many descriptors rely on frequency analysis of the signal and therefore they require the computation of the Fourier Transform.

[4] A set of operations which purpose is to reduce the amount of background noise in a signal, therefore incrementing the signal-to-noise ratio (*SNR* or sometimes *S/N*).

time even harder.
Several ways of computing an onset detection function have been proposed, according to what aspects are taken into account for defining an onset. Actually, onset detection may be performed in time domain (when looking for significant changes in the overall energy), frequency domain (if looking for events regarding just a specific range of frequencies), phase domain or complex domain. Important algorithms for this task are:

- *HFC*, the High Frequency Content detection function that looks for important changes on highest frequencies. It is very useful for detecting percussive events.
- Spectral Flux, that decomposes the entire audible range of frequencies (approxitamely the interval 20-20000 Hz) into bins, measures changes in magnitude in each bin, and then sums all the positive changes across all the bins.
- the Complex-Domain spectral difference function [41] taking into account changes in magnitude and phase. It emphasizes note onsets either as a result of significant change in energy in the magnitude spectrum, and/or a deviation from the expected phase values in the phase spectrum, caused by a change in pitch.

HFC was proposed by Masri in [42]. Let us consider the short-time Fourier transform (STFT) of the signal $x(n)$:

$$X_k(n) = \sum_{m=-\frac{N}{2}}^{\frac{N}{2}-1} x(nh+m)w(m)e^{-\frac{2j\pi mk}{N}} \tag{2.1}$$

where $w(m)$ is again an N-point window, and h is the hop size, or time shift, between adjacent windows. The idea behind HFC is to give more weight to higher frequencies, by defining a onset function whose values are computed in the follwing way:

$$HFC(n) = \frac{1}{N} \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} k|X_k(n)|^2 \tag{2.2}$$

The HFC function produces sharp peaks during attack transients and is notably successful when faced with percussive onsets, where transients are well modeled as bursts of white noise [43].
On the other hand, the Spectral Flux SF function is defined as follows:

$$SF(n) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} H(|X(n,k)| - H(|X(n-1,k)|) \tag{2.3}$$

where $H = \frac{x+|x|}{2}$ is the half-wave rectifier function. This algorithm greatly characterizes changes in magnitude spectrum but it quite weak to frequency-modulation phenomena (such as vibrato). To this end, the recently proposed variant SuperFlux [44] seems to achieve much better results.
Another interesting onset function is the *Complex Domain*, that calculates expected the expected amplitude and phase of the current bin $X(n,k)$ based on the previous two bins $X(n-1,k)$ and $X(n-2,k)$. By assuming constant amplitude the expected value $X_T(n,k)$ is then computed:

$$X_T(n,k) = |X(n-1,k)|e^{\psi(n-1,k)+\psi'(n-1,k)} \tag{2.4}$$

and therefore a complex domain onset detection function CD can be defined as the sum of absolute deviations from the target values [40]:

$$CD(n) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} |X(n,k) - X_T(n,k)| \tag{2.5}$$

Given an onset function (for instance one of the already cited $HFC(n)$, $SF(n)$ or $CD(n)$), onsets are then extracted by a peak-picking algorithm which finds local maxima in the detection function, subject to various constraints. Threshold and constraints used in the peak-picking algorithm has a large impact on the results, specifically on the ratio of false positives[5] to false negatives[6]. For instance, a higher threshold may lead to a lower number of false negatives but to a higher number of false positive, while a lower threshold may have the opposite effect. A compromise, mostly specific to the application, has to been found.

BPM

The algorithms for detecting the beats-per-minute (generally called *beat tracking algorithms*) greatly rely on onset detection funcions. The basic idea is to look for some time-pattern that may explain the distribution of onsets over time, and hence derive BPM. They usually require more than one onset detection function to achieve good results. One of the most performant beat tracking algorithm is TempoTapDegara, presented by N. Degara et al. in [45].
EXPLAIN ALGORITHM HERE

[5] True onsets that are not detected by the algorithm.
[6] Points that are classified as onsets by the algorithm, while they are actually not.

2.2.2.2 Tonality

Many efforts have been taken in order to improve the techniques for detecting tonality or harmonic content of a song, as this is one of the most main aspects of western music (a direct consequence of tonality is the detection of predominant melody; to understand why this is so important just ask yourself how many times you whistled or sang a song to let other people recognize it). Many studies have focused on this aspect of music were not oriented toward the computation of similarity between tracks, but instead toward different tasks, such as *automatic trascription of a polyphonic audio signal* (mainly into a MIDI representation) and *source separation*, that is the task of isolating a single and specific instrument among many playing together.
From a music point of view, in western music, an octave is made of 12 different pitches, and seven differents notes take place in this discrete range. According to the pitch assigned to each note, we may have different *keys*, that are a combination of a *tonic* (the central pitch) and the mode. *Major* and *minor* are the most popular modes. (ADD IMAGE OF THESE TWO MODES)
Harmony is a term that denotes the simultaneous combination of notes, called *chords*, and over time, *chord progressions*.
One of the most important descriptor for extracting information related to tonality is called Harmonic Pitch Content Profile (***HPCP***, also called chromagram). This is directly related to tonality and chord detection: chords can be recognized from from the *HPCP* without even precisely detecting what notes are being played, and tonality can also be inferred by *HPCP* (and in this case a previous estimation of chords is not necessary).
An *HPCP* is a $12k$-sized vector indicating the level energy for each profile class. If $k = 1$, the *HPCP* represents the intensities of the twelve semitone pitch classes, otherwise of subdivision of these[7]. In [47], Gómez proposes to distinguish tonality features on temporal scales:

- Instantaneous: features attached to an analysis frame.
- Global: features related to a wider audio segment, for instance a phrase, a chorus or the whole song.

Furthermore, Gómez proposes to split tonal descriptors in both low-level and high-level descriptors. We hence obtain the representation of tonal descriptors shown in Table 2.1.

[7]It may be extremely useful to study subdivision of semitone pitch classes when dealing with non-chromatic scales, that are very popular in eastern music.

Name	Temporal Scale	Level of abstraction
HPCP	Instantaneous	Low
Chord	Instantaneous	High
Average HPCP	Global	Low
Key	Global	High

Table 2.1: Main tonal descriptors.

The general approach for computing *HPCP* is indicated in figure PUT BLOCK DIAGRAM HERE and can be summarized as follows:

- At first, some pre-processing of the audio signal may be performed. For instance, a transient detection algorithm may be used to detect and eliminate regions where the harmonic structure is noisy. This step is usually performed to decrease the computational cost of the *HPCP* without affecting its output [48].
- At this point, spectral analysis is required: once the signal is segmented into frames of a proper size and a windowing function is applied, the Fast Fourier Transform (*FFT*) is computed to get the frequency spectrum. Frames should not be too short, in order to have a better frequency resolution.
- A peak-picking algorithm is run to find those frequencies corresponding to local maxima in the spectrum. Usually, these algorithms are not run only on the interval [100, 5000] Hz: this has shown much better results, because outside this range the signal is predominantly noisy, due to some percussion and instrumental noise [47].
- The *HPCP* is finally computed; many approaches have been developed for this task, all based on the pitch content profile algorithm presented by Fujishima in [46]. At first, a mapping of each frequency bin of the *FFT* to a pitch class is needed (for instance, *FFT* bins corresponding to frequencies like 430Hz, 432Hz or 444Hz are mapped to the A at 440Hz). Then, the amplitudes inside each region are summed up and divided by the number of bins inside that region. Finally, the bins obtained are collapsed, so that bins referring to the same note but in a different octave (for example A4 and A5) are collapsed in a single bin for that note, indicating the overall energy of it in the frame. The *HPCP* is different from the PCP in the sense that a higher resolution may be used for *HPCP* bins (decreasing the quantization level to less than a semitone) and a weight function is introduced into the feature computation. The *HPCP* value of the n-th *HPCP* bin is calculated as:
$$HPCP(n) = \sum_{i=1}^{nPeaks} w(n, f_i) a_i^2 \tag{2.6}$$

where a_i and f_i are respectively the magnitude and the frequency of the ith peak, $nPeaks$ is the number of spectral peaks considered, and $w(n, f_i)$ is the weight of the frequency bin f_i when considering the *HPCP* bin n.
The performance of the *HPCP* builder strongly relies on the weight function [49]. Note that, for how the common procedure of building *HPCP* is defined, *HPCP* are usually considered robust to noise and different tuning references.
HPCP values are usually normalized in order to store the relative importance of the nth *HPCP* bin:

$$HPCP_{normalized}(n) = \frac{HPCP(n)}{Max_n(HPCP(n))} \tag{2.7}$$

Once the *HPCP* are computed, additional tonal features may be computed, such as tonality or chords. Regarding tonality estimation, this is generally computed through a correlation analysis between the *HPCP* obtained and a matrix of *HPCP* profiles corresponding to different keys.

2.2.3 High-level Descriptors

2.2.4 Main Tools For Extracting Audio Content

Many tools are available for the extraction of audio content descriptors from an audio signal. Many of them have been developed by researchers following the research necessities of MIR. This great variety of tools offers support to several operative systems (mainly Linux, Mac OS X and Windows) and programming languages (Java, C++, C, Python, Matlab). Some of this tools may be offered as standalone software or as a Vamp plugin. Not all the tools for extracting audio content are open-source, therefore not being particularly useful for the research community. In the following paragraphs, we'll briefly show the features of the tools taken into consideration on the development of this work.

Essentia

Essentia[8] is an open-source C++ library of algorithms for audio analysis and audio-based music information retrieval. It has been developed at Music Technology Group[9], Universitat Pompeu Fabra, and has released under the Affero GPL license[10]. In its current version 2.0.1, it contains a large collection of spectral, temporal, tonal, and

[8] `http://essentia.upf.edu/`
[9] `http://mtg.upf.edu/`
[10] `http://www.gnu.org/licenses/agpl.html`

high-level music descriptors, algorithms for audio input/output functionality, standard digital signal processing blocks and statistical tools. The library can be complemented with Gaia [11], a C++ library to apply similarity measures and classifications on the results of audio analysis. Both these libraries include Python 2.* bindings and support Linux, Mac OS X and Windows. Essentia has been in developed for over 7 years, incorporating the work of more than 20 researchers and developers through its history. It offers two different modes: standard and streaming, the first being imperative while the latter being declarative. Each processing block is offered as an algorithm, and has three different types of attributes: inputs, outputs and parameters. Different blocks may be linked in order to perform the required processing task. In figure INSERT FIGURE a block diagram of a processing task is shown, composed of several different algorithms linked together. See Appendix A for consulting the full list of descriptors provided by Essentia 2.0.1.

Echonest

jMIR

MIRtoolbox

2.3 Computing Music Similarity with Audio Content Descriptors

2.4 Conceptual Differences Between Metadata and Audio Content Information

The performance of content-based approaches is considerably lower [9]. It is challenging to try to make the so-called *semantic gap* smaller [10]

The advantage of relying on the audio signal over, say, expert annotations, is that the process is objective and can be automated to a large extent. However, extracting the features can be computationally costly [21]. Another limitation is that there might be features like the release date, the "freshness," or popularity of a track, which can be relevant in the playlist generation process but that cannot be extracted from the audio signal [22].

[11] `https://github.com/MTG/gaia`

When used in an automated process, data completeness and consistency are crucial. Another potential problem is that not all types of metadata are objective, and annotations regarding, for example, the mood or the genre of a track can be imprecise or inconsistent [28].

(speaking of tags) Although such annotations can be rich and diverse, the perception of music is again subjective and can even be influenced by the perception of other people [29]; tags only for popular songs [28]

When dealing with track ratings: grabbed from a wall posting on Facebook [30] or a tweet on twitter [31], 1-to-5 rating scales like on iTunes. Challenges: problem of data sparsity (especially for the tracks from the long tail), a positivity bias (the phenomenon that most of the ratings are highly positive and negative feedback is rare [28]).

Chapter 3

Assessing the performance of a music similarity computation system

3.1 Literature Review

The coherence of the tracks is a typical quality criterion for playlists [32]. Therefore, selecting and ordering tracks based on their similarities is an obvious strategy to generate playlists. The core of any similarity-based approach is its distance function, which characterizes the closeness of two tracks. How the distance function is actually designed depends on the available data, which could include the raw audio signal along with the features that can be derived from it, but also metadata, such as the artists, the genres, playcounts, or ratings [Slaney and White 2007]. In many cases, a signature or model of each track is determined first, in which the distance function is then applied. Typical examples for such functions applied on more abstract models of a track's features are the earth-mover's distance [32], the Kullback-Leibler (KL) divergence [33], or the Euclidean distance [34].

See section 5.2 of [16] for a background on how to assess the quality of a playlist: user studies, log analysis, objective measures, comparison with handcrafted playlists

Part II

Methodology

Chapter 4

Case Study: Requirements and Approach

4.1 Catalogue of music

The catalogue of music provided features NUM songs, for a total length of 91 hours, 43 minutes and 35 seconds. This catalogue has been provided with metadata indicating only artist, year of release and title of each song. Furthermore, all of these work can be labelled as belonging to the electro-acoustic genre, which usually indicates very abstract and arrhythmic, for which is difficult to provide semantic descriptors or tags. Given this latter feature of the music and the length of the entire catalogue, the possibility of manually annotating it with proper metadata has been soon disregarded. This collection of music has therefore represented a great chance for developing a system based on the latest findings on audio content analysis.

4.2 Requirements

Despite its intended use as part of the exhibition *"Phonos, 40 anys de música electrònica a Barcelona"*, the software developed should feature good flexibility to different catalogues of music, in order to be exploited as a part of the research for the GiantSteps project. This has represented a strong requirement during the development, and has induced the adoption of several descriptors that may not be particularly meaningful for the Phonos catalogue of songs, but that has extended the range of possible music catalogues in which the system performance could be satisfactory. Furthermore, as a part of a research project, the system developed should be easily extendable in other research activities, hence a robust, consistent and well-document code is preferred.

4.3 Design of the system

Justify the choice of audio content analysis over metadata, **of splitting app in two parts (analysis + rt app)**

Chapter 5

Computation of Audio Features

5.1 Tools used for feature extraction, features extracted

5.2 Similarity computation (fast map)

Chapter 6

The Real-Time Application

6.1 Implementation (python server + html client), Gstreamer

6.2 Functioning

Descriptors of first bar, similarity computation (both as an Euclidean Distance and as SKL)

Part III

Results and Discussion

Chapter 7

Evaluation

Also some words on the Kiosk

Chapter 8

Future Work

Appendix A

List of Essentia Features

Write your Appendix content here.

Appendix B

List of Echonest Features

Write your Appendix content here.

Appendix C

Phonos: list of songs

Write your Appendix content here.

Bibliography

[1] A. Li-chun Wang, and Th Floor Block F. *An industrial-strength audio search algorithm.* Proceedings of the 4 th International Conference on Music Information Retrieval, 2003.

[2] J.S. Downie. *The Scientific Evaluation of Music Information Retrieval Systems: Foundations and Future.* Computer Music Journal, 28:12-23, 2004.

[3] N. Orio. *Music Retrieval: A Tutorial and Review.* Foundations and Trends®in Information Retrieval, 1(1):1-90, 2006.

[4] M. Schedl, E. Gómez, and J. Urbano. *Music Information Retrieval: Recent Developments and Applications.* Foundations and Trends®in Information Retrieval, 8(2-3):127-261, 2014.

[5] J.J. Aucouturier, and F. Pachet. *Improving Timbre Similarity: How High is the Sky?.* Journal of Negative Results in Speech and Audio Sciences, 1(1):1-13, 2004.

[6] X. Serra, M. Magas, E. Benetos, M. Chudy, S. Dixon, A. Flexer, E. Gómez, F. Gouyon, P. Herrera, S. Jorda, O. Paytuvi, G. Peeters, J. Schlüter, H. Vinet, and G. Widmer. *Roadmap for Music Information ReSearch.* Geoffroy Peeters (editor), Creative Commons BY NC ND 3.0 license, 2013.

[7] M.E.P. Davies, P. Hamel, K. Yoshii and M. Goto. *AutoMashUpper: Automatic Creation of Multi-Song Music Mashups.* IEEE/ACM Transactions on Audio, Speech, and Language Processing, 22(12):1726-1737, 2014.

[8] G. Szymanski. *Pandora, or, a never-ending box of musical delights.* Music Reference Services Quarterly, 12(1):21-22, 2009.

[9] M. Slaney. *Web-scale multimedia analysis: Does content matter?.* IEEE Multimedia, 18(2):12-15, 2011.

[10] J.J. Aucouturier. *Sounds like teen spirit: Computational insights into the grounding of everyday musical terms.* Language, Evolution and the Brain. Frontiers in Linguistics, 35-64, 2009.

[11] S.J. Green, P. Lamere, J. Alexander, F. Maillet, S. Kirk, J. Holt, J. Bourque, and X.W. Mak. *Generating transparent, steerable recommendations from textual descriptions of items.* ACM Conference on Recommender Systems (RecSys'09), 281-284, 2009.

[12] D. Bogdanov, and X. Serra. *From music similarity to music recommendation: Computational approaches based on audio features and metadata.* PhD Thesis, Universitat Pompeu Fabra, 2013.

[13] D. Schnitzer. *Mirage – High-Performance Music Similarity Computation and Automatic Playlist Generation.* Master's Thesis, Vienna University of Technology, 2007.

[14] L. Barrington, D. Turnbull, D. Torres, and G. Lanckriet. *Semantic similarity for music retrieval.* Music Information Retrieval Evaluation Exchange (MIREX), 2007.

[15] K. West, and P. Lamere. *A model-based approach to constructing music similarity functions.* EURASIP Journal on Advances in Signal Processing, 149-149, 2007.

[16] G. Bonnin, and D. Jannach. *Automated Generation of Music Playlists: Survey and Experiments.* ACM Computing Surveys, 47(2), Article 26, 2014.

[17] R. Dias, and M. J. Fonseca. *MuVis: An application for interactive exploration of large music collections.* Proceedings of MM, 1043-1046, 2010.

[18] J.S. Downie. *Music information retrieval.* Annual Review of Information Science and Technology, 37(1):295-340, 2003.

[19] M. Grachten, M. Schedl, T. Pohle, and G. Widmer. *The ISMIR cloud: A decade of ISMIR conferences at your fingertips.* Proceedings of ISMIR, 63-68, 2009.

[20] M. Dopler, M. Schedl, T. Pohle, and P. Knees. *Accessing music collections via representative cluster prototypes in a hierarchical organization scheme.* Proceedings of ISMIR, 179-184, 2009.

[21] I. Schmädecke, and H. Blume. *High performance hardware architectures for automated music classification.* Algorithms from and for Nature and Life, 539-547, 2013.

[22] O. Celma, and X. Serra. *FOAFing the music: Bridging the semantic gap in music recommendation.* Web Semantics: Science, Services and Agents on the World Wide Web, 6(4):250-256, 2008.

[23] F. Pachet, P. Roy, and D. Cazaly. *A combinatorial approach to content-based music selection.* Multimedia, 7(1):44-51, 2000.

[24] D. Bogdanov, and P. Herrera. *How much metadata do we need in music recommendation? A subjective evaluation using preference sets.* WProceedings of ISMIR, 97-102, 2011.

[25] N. Aizenberg, Y. Koren, and O. Somekh. *Build your own music recommender by modeling internet radio streams.* Proceedings of WWW, 1-10, 2012.

[26] R. Van Gulik, and F. Vignoli. *Visual playlist generation on the artist map.* Proceedings of ISMIR, 520-523, 2005.

[27] F. Coelho, J. Devezas, and C. Ribeiro. *Large-scale crossmedia retrieval for playlist generation and song discovery.* Proceedings of OAIR, 61-64, 2013.

[28] O. Celma. *Music Recommendation and Discovery: The Long Tail, Long Fail, and Long Play in the Digital Music Space.* Springer, 2010.

[29] J. McDermott. *Auditory preferences and aesthetics: Music, voices, and everyday sounds.* Neuroscience of Performance and Choice. Elsevier, 227-256, 2012.

[30] A. Germain, and J. Chakareski. *Spotify me: Facebook-assisted automatic playlist generation.* Proceedings of MMSP, 25-28, 2013.

[31] D. Hauger, J. Kepler, M. Schedl, A. Kosir, and M. Tkalcic. *The million musical tweets dataset: What can we learn from microblogs.* Proceedings of ISMIR, 189-194, 2013.

[32] B. Logan. *Music recommendation from song sets.* Proceedings of ISMIR, 425–428, 2004.

[33] F. Vignoli, and S. Pauws. *A music retrieval system based on user driven similarity and its evaluation.* Proceedings of ISMIR, 272-279, 2005.

[34] P. Knees, T. Pohle, M. Schedl, and G. Widmer. *Combining audio-based similarity with Web-based data to accelerate automatic music playlist generation.* Proceedings of MIR, 147-154, 2006.

[35] D. Jannach, M. Zanker, M. Ge, and M. Gröning. *Recommender systems in computer science and information systems — A landscape of research.* Proceedings of EC-Web, 76-87, 2012.

[36] N.H. Liu, S.W. Lai, C.Y. Chen, and S.J. Hsieh. *Adaptive music recommendation based on user behavior in time slot.* Computer Science and Network Security, 9(2):219-227, 2009.

[37] S. McNee, J. Riedl, and J. Konstan. *Being accurate is not enough: how accuracy metrics have hurt recommender systems*. CHI'06 extended abstracts on Human Factors in Computing Systems, p.1001, 2006.

[38] G. Jawaheer, M. Szomszor, and P. Kostkova. *Comparison of implicit and explicit feedback from an online music recommendation system*. Int. Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec'10), p.47-51, 2010.

[39] B. Whitman, and S. Lawrence. *Inferring Descriptions and Similarity for Music from Community Metadata*. Proceedings of the 2002 International Computer Music Conference (ICMC 2002), p.591-598, 2012.

[40] S. Dixon *Onset Detection Revised*. Proc. of the 9th Int. Conference on Digital Audio Effects (DAFx'06), p.133-137, 2006.

[41] J.P. Bello, C. Duxbury, M. Davies, and M. Sandler. *On the use of phase and energy for musical onset detection in the complex domain*. Signal Processing Letters, IEEE, 11(6):553-556, 2004.

[42] P. Masri. *Computer Modeling of Sound for Transformation and Synthesis of Musical Signal*. Ph.D. dissertation, Univ. of Bristol, 1996

[43] J.P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. B. Sandler. *A tutorial on onset detection in music signals*. IEEE Transactions on Speech and Audio Processing, 13(5):1035-1047, 2005.

[44] S. Böck, and G. Widmer. *Maximum filter vibrato suppression for onset detection*. Proceedings of the 16th International Conference on Digital Audio Effects (DAFx-13), Maynooth, Ireland, 2013.

[45] N. Degara, E.A. Rua, A. Pena, S. Torres-Guijarro, M.E. Davies, and M.D. Plumbley. *Reliability-informed beat tracking of musical signals*. IEEE Transactions on Audio, Speech, and Language Processing, 20(1):290-301, 2012.

[46] T. Fujishima. *Realtime chord recognition of musical sound: A system using Common Lisp Music*. Proceedings of the International Computer Music Conference, Beijing. 1999.

[47] E. Gómez. *Tonal Description of Polyphonic Audio for Music Content Processing*. INFORMS Journal on Computing, 18(3):294-304, 2006.

[48] J. Bonada. *Automatic technique in frequency domain for near-lossless time-scale modification of audio*. Proceedings of International Computer Music Conference, 396-399, 2000.

[49] G. Cabral, J.P. Briot, and F. Pachet. *Impact of distance in pitch class profile computation*. Proceedings of the Brazilian Symposium on Computer Music, 319-324, 2005.

[50] D. Bogdanov, N. Wack, E. Gómez, S. Gulati, P. Herrera, O. Mayor, G. Roma, J. Salamon, J. Zapata, and X. Serra. *ESSENTIA: an open-source library for sound and music analysis*. Proceedings of the 21st ACM international conference on Multimedia, 855-858, 2013.