MCIPA: A MUSIC CONTENT INFORMATION PLAYER AND ANNOTATOR FOR DISCOVERING MUSIC

Geoffroy Peeters
Ircam - CNRS STMS
peeters@ircam.fr

David Fenech
Ircam
fenech@ircam.fr

Xavier Rodet
Ircam - CNRS STMS
rod@ircam.fr

ABSTRACT

In this paper, we present a new tool for intra-document browsing of musical pieces. This tool is a multimedia player which represents the content of a musical piece visually. Each type of musical content (structure, chords, downbeats/ beats, notes, events) is associated which a distinct visual representation. The user sees what he/ she is listening too. He can also browse inside the music according to the visual content. For this, each type of visual object has a dedicated feedback, either as an audio-feedback or as a playhead feedback. Content information can be extracted automatically from audio (using signal processing algorithms) or annotated by hand by the user. This multimedia player can also be used as an annotator tool guided by the content.

1 INTRODUCTION

Content description of music based on audio signal analysis has been the subject of many researches over the last few years. Applications such as query over large music collections for specific music characteristics (melody, genre, mood, singer type ...) or search-by-similarity (based on melody, chord progression or timbre ...) have now become possible. However, few works address the problem of using content-information to guide the user during its listening, to allow him/ her to have a better understanding of the musical content, to help him/ her to browse inside a track by the content (intra-document browsing) or to help him comparing several pieces of music visually.

On the other side, the development of music content-extraction algorithms relies for a large part on the availability of annotated music-audio. These annotations are used to learn concepts such as audio structure, audio chords or to check the performances of developed algorithms (multi-pitch or beat-positions estimation algorithms). Tools that allow annotating music audio files in terms of specific music characteristics (chords, beats, structure) are still missing.

In this paper, we present a tool which has been developed in order to allow both user interaction with the music content and annotation of the music content. The paper is organized as follows. In part 2, we give an overview of currently existing music player and audio annotation tools. From this overview we give a set of requirements for our tool. In part 3, we detail the various parts of our tool: its general architecture, the various types of content described and explain how the tool is used for annotation. In part 4, we give technical details about the development of our tool.

2 RELATED WORKS AND REQUIREMENTS

Computer based media players are numerous (iTunes, Windows Media Player, Real Player, Win Amp ...). However, currently none of them propose a visualization of the content of the music track in order to allow browsing of the document by its content. On the opposite, there exist several tools dedicated to the annotation of audio. **"AS (AudioScuplt) Annotation"** [12] is a free software developed by Ircam for Mac-OS-X . The main paradigm of this software is annotation over the visualization of the spectrogram. Many algorithms are integrated into the software such as transient detection, pitch detection ... Annotation is done over the spectrogram using temporal markers or by drawing midi notes over a note-scaled spectrogram. **"Sonic Visualizer"** [5] is an open source software (Linux/ Mac-OS-X/ Windows) developed by Queen Mary University of London. It is also based on annotation over the visualization of either the waveform or the spectrogram. The various annotations can be super-imposed using a set of visual masks. Content analysis is performed using external plug-ins (Vamp format) so that the user can plug-in its favourite content-analyzer. **"CLAM annotator"** [1] is an open source software (Linux/ Mac-OS-X/ Windows) developed by IUA-UPF. It is a framework for developing graphical interfaces and signal analysis algorithms. **"MUCOSA"** [11] also developed by IUA-UPF is an online system for global (a single description assigned to the whole file duration) annotations of music tracks. **"Wavesurfer"** [19] is an open source software (Linux/ Mac-OS-X/ Windows) developed by KTH. It is based on waveform and spectrogram visualization. Some algorithms for energy, pitch or formant estimation are included. Annotation is done by placing markers over the file duration. Some plug-ins for music content analysis (such as beat analysis [9]) have been developed. Wavesurfer has functionalities for browsing by content (markers) over the file duration. **"Praat"** [3] is an open source software (Linux/ Mac-OS-X/ Windows) developed by IPS. It includes many signal analysis algorithms (pitch, formant) but is mostly dedicated to speech analysis. **"Acousmographe"** [10] is a Windows XP software developed by the GRM. It is mainly based on annotation over spectrogram and includes a wide range of graphical tools in order to facilitate annotation reading (specific shape and colour for each annotation). **"Transcriber"** [6] is an open source software (Linux/ Mac-OS-X/ Windows) developed by the French DGA. It is mostly dedicated to the transcription of speech to text. It has functionalities for browsing by content (text) over the file duration. **"Audac-**

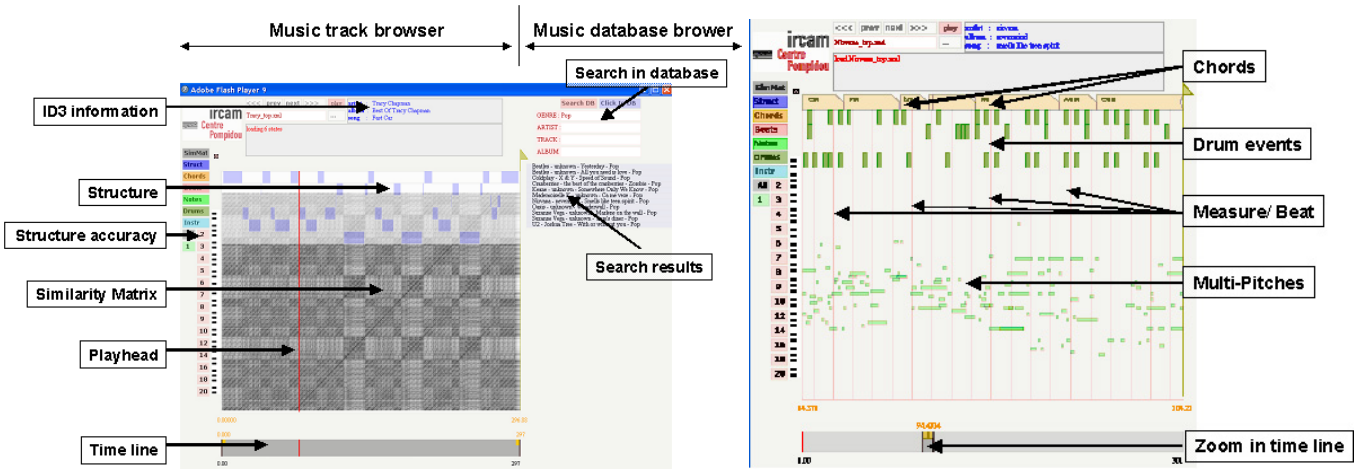


Figure 1. MCIpa interface: [left] large view of the interface with music track browser (similarity matrix and structure representation) and music database browser, [right] detailed view of the music track browser (chord progression, drum events, downbeat/ beat positions and multi-pitch).

ity" [2] [13] is an open source audio multi-tracks recorder with MIDI integration. Because of the audio/ MIDI synchronization it is used for annotation although no specific tools are dedicated to annotation.

2.1 Discussion

According to the previous list, there are currently no tools dedicated specifically to the annotation of music in terms of music content (structure, chords ...). Most of the tools indeed aim to annotate generic audio or speech and propose a representation of the signal (waveform or spectrogram) to help this annotation. A signal representation is useful to annotate a file in low-level audio terms (transients, onsets, fundamental frequency ...) but not to annotate it in high-level music terms (structure, chords ...). Our tool only concentrates on high-level music description. We mean by high-level music description a description that an every-day user can understand (a spectrogram still requires some knowledge about signal processing). Our point of view is that the visualization of one type of music content can help the annotation of another type of music content. Therefore the ground visualization of our interface is the music content description itself (by default it is a similarity matrix representing the global structure of the track) and not a signal representation. However some functionalities present in the list of softwares described above are also interesting in our case: - visual masking and transparency system (AS-Annotation, Sonic Visualizer, Acousmographe), - use of specific colour and shape for each type of annotations (Acousmographe), - separation between the graphical interface and the content-extraction tools (Sonic Visualizer, Wavesurfer), - possibility to quickly browse the files by annotations (Wavesurfer, Transcriber). In our tool, each type of music content is displayed on the same panel and has a specific colour and shape. A large use is made of visual masking and transparency. The content-extraction algorithms are not part of the tool and are provided to it by a set of XML files. These files can be generated by any content-extraction algorithms or annotated by hand. The tool acts either as a media player with music-content visualization or as an annotator of music-content. In the following of this paper, we call **Music Content Information (MCI) object** a description of the music-content that has a specific visual representation, a specific interface-feedback, a dedicated content-extraction algorithm and a dedicated file format.

2.2 Semantic HIFI intra-document player

In the Semantic HIFI project, a first intra-document-player based on the visualization of/ and the interaction with the musical structure has been integrated into the remote-controller of an HIFI system (see Fig. 2). User testing of this interface has been performed and are presented in [4]. The results of this study show that users found the interface "interesting, useful and innovating". However, some weak points were specified by the users: • no possibility to assign a label to each block of the structure, • no possibility to exchange structure annotations among user, • no possibility to assign a colour to each block, • a timeline with tempo annotation would be welcome. These user-comments were taken as the basis for the development of the MCIpa interface.

2.3 Requirements

Generic requirements: The tool must be easy to use, to understand and to install. The same tool is used for both annotation and visualization. The automatic extraction is not part of the tool (the content-description is provided by a set of XML files). This allows the user to use its favourite extraction algorithm and allows the application to remain light. The tool must be cross-platform. The interface should read and play directly the most currently used music formats, mp3 included.

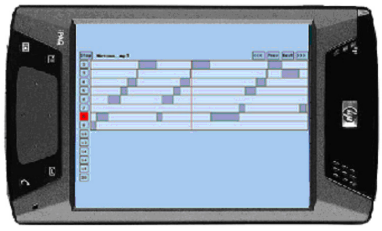


Figure 2. PDA used as a remote-controller for the Semantic-HIFI system: music structure representation.

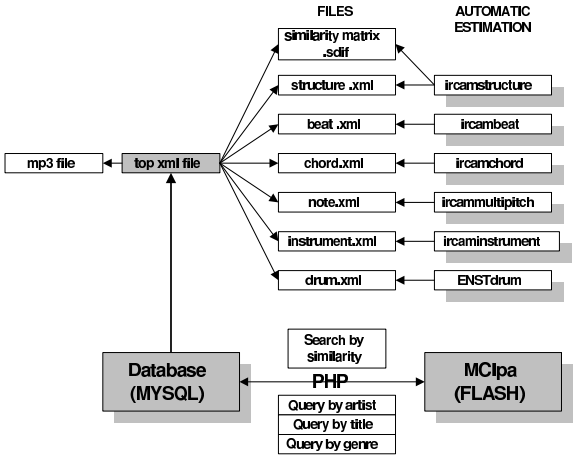


Figure 3. Schema of communication between the MCIpa interface, the database, the audio and content-description files and the corresponding automatic estimation tools.

Interface requirements: The interface must be intuitive. The action must be quick and quickly reachable (keyboard shortcuts are used extensively in order to do that). The main paradigm is "click and listen to what/where you have clicked on". It should be easy for the user to navigate in the representation and to change the type of representations being displayed.

3 DEVELOPED INTERFACE AND FUNCTIONALITIES

The interface is composed of two main parts: the "music database browser" and the "music track browser" (see Fig. 1).

3.1 Music database browser

The music database browser (MDB) allows searching inside a music database by music genre, artist-name, track-name or album-name. The result of the query is displayed as a list of music tracks. Clicking on any item of this list will automatically load the corresponding music track into the "music track browser".

3.2 Music track browser

The music track browser (MTB) represents graphically all Music Content Information of a given track and allows navigation inside it. The top part of the MTB groups the actions for loading directly the file (when not using the "music database browser"), for displaying the editorial metadata (ID3 tags) and for performing the basic playing actions (play, pause, stop, forward/backward).

The middle panel of the MTB represents visually the musical content of the track. Time is represented horizontally. The current playing time is represented as a play-head (red vertical bar) sliding from left to right over the file duration. In this panel are superimposed the various visual objects representing the musical content of the track (the MCI objects). The main paradigm of this panel is "click and listen to what/where you have clicked on". For this, each type of MCI object has a specific behaviour, either as an audio feedback or as a play-head positioning feedback. We have decided to represent all information on the same panel in order to allow comparison between the various MCIs and highlight their musical dependencies.The MCI objects represent the music-content with decreasing temporal scales: similarity matrix, temporal structure, chord progression, music temporal grid (downbeat and beat positions), various notes occurring over time and various sound events occurring over time (musical instruments or percussive sounds). Each type of MCI object has a specific visual representation (shape and colour). They are super-imposed graphically thanks to a masking and transparency system (see Fig. 1 [left] with the transparent structure over the similarity matrix). The user can quickly change the kind of representation during playing using either interface buttons or keyboard shortcuts (showing/masking similarity matrix, structures, chords, beats, notes, sound-events representation).

The bottom part of the MTB represents the time-line of the track. It allows the user to zoom in/ out or to scroll over the track. In this case, the time-line indicates the currently displayed content with a darker rectangle. Auto-scroll functionality is included and can be turned on and off.

3.3 Overall architecture and file formats

The interface directly reads the mp3 format. The descriptions of the various music-contents are stored as a set of XML files. These files are organized in a hierarchical way: a "top" XML file contains a list of pointers to the described mp3 file and to the various content-description files (one file per type of description). The description can • come from manual annotations (either done externally or using the interface itself), • be generated using a set of automatic music-content extraction tools (which are not part of the application). The interface only displays existing data in the XML files (not all of them are mandatory). The XML formats read by MCIpa are described at the following URL [1] . It should be noted that we choose not to use the MPEG-7 Audio XML schema [15] here. This is because, for the kind of music description used here, implementing MPEG-7 is too heavy, morevover MPEG-7 lacks several important music descriptions used here. The proposed XML format has been chosen to be very simple to use and to read. The general schema of the tool is represented in Fig. 3.

[1] http://recherche.ircam.fr/equipes/analyse-synthese/peeters/mcipa/

3.4 Various types of Music Content Information objects

We now describe the various types of Music Content Information (MCI) displayed on the interface, their meaning, their visual representation, their interface-feedback, and the way they can be obtained (by automatic extraction or manual annotation). Table 1 summaries the various MCIs, their representations and corresponding actions.

3.4.1 Similarity matrix

Meaning: The default representation of the music track (representation loaded by default on the background of the MTB) is the similarity matrix [7]. This matrix describes the similarity between the content at the various times of a music track. A point (x,y) in the matrix represents the similarity between the content of the track at time x and time y. *Visual representation:* This matrix is represented as a 2D-image, low similarity values are represented with bright colours, high similarity values with dark colours. Dark lines parallel to the main-diagonal in the matrix represents repetition of segments of times (for example repetition of the same melody). *Extraction and annotations:* This matrix is automatically generated by analyzing the audio signal. It represents the similarities of timbre (instrumentation) and harmony (melodies and chords) as well as their evolution over short periods of time [17]. The similarity matrix cannot be manually generated. *Storage:* The similarity matrix is stored either as an SDIF file or directly as a bitmap file (for faster loading). *Interface feedback:* The user can click anywhere inside the matrix, the playing will starts immediately at the given position. The user can therefore quickly compare several occurrences of a melody by clicking on the various diagonals.

3.4.2 Music structure

Meaning: The music structure represents a music track as a succession of parts (verse, chorus, bridge ...) that can be repeated over time. *Visual representation:* The structure is represented visually as a set of rectangles placed inside various corridors. Each corridor represents a specific type of part; the width of the rectangle inside it represents the time extend of a specific part. The accuracy of the structure (number of corridors used for the decomposition) can be chosen by the user ("structure accuracy" buttons in Fig. 1). The structure can be super-imposed to the similarity matrix in order to allow comparison between the estimated structure and the similarity between the individual times. *Extraction and annotations:* The structure can be estimated automatically from the audio signal using for example the algorithms described in [17] or any algorithm conforming to the proposed XML format definition. The structure can also be manually annotated (or corrected) using the marker system explained in part 3.5. *Storage:* The structure is stored in an XML files describing the decomposition specific to each "structure accuracy" level. *Interface feedback:* Clicking anywhere inside a rectangle starts immediately the playing at the beginning of the corresponding part. The user can therefore quickly compare several parts or several occurrences of the same part. Functionality of forward/ backward by part is also included.

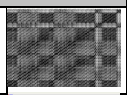
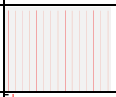
MCI	Graphical representation		User Interaction / Interface feedback
Similarity Matrix	As a 2D image on background		click anywhere inside the image starts playing at the given position
Music Structure	As a part-roll (each type of part is represented on a specific line)		- choose the number of parts used for the subdivision - click inside a part starts playing at the part beginning - forward-backward by parts
Chord progression	As a set of TABs with chord labels		click inside a chord 1) starts playing at the chord beginning 2) plays the corresponding chord prototype
Downbeat/ beat positions	As a set of vertical lines (thick lines for downbeats, thin lines for beats)		Audio click when the play-head crosses a beat marker
Multi-pitch	As a piano-roll (each note-stream is represented by a specific color)		- choose the displayed note channels - click inside a note plays the corresponding note prototype
Sound-events	As a sound-event-roll (each type of sound-event is represented on a specific line)		not yet

Table 1. Music Content Information (MCI) objects.

3.4.3 Chord progression

Meaning: Chord progression denotes the location and duration of each chord of the music track (C Major ... B Major, c minor ... b minor). *Visual representation:* Each chord is represented as a TAB on the top of the middle panel with a specific location and extent. On each TAB is indicated the name of the chord (GM, DM ...). *Extraction and annotations:* The chord progression (chord positions and labels) can be estimated automatically from the audio signal using for example the algorithm described in [16] or any algorithm conforming to the proposed XML format definition. The chords can also be manually annotated (or corrected) using the marker system explained in part 3.5. *Storage:* The chord progression is stored in an XML files. *Interface feedback:* When the user clicks on a specific TAB, the playing starts immediately at the beginning of the corresponding chord. The user can thus quickly skip by chords inside the track. Another feedback, currently under development, is the audio feedback. Clicking on a TAB automatically plays the corresponding prototype chord.

3.4.4 Downbeat and beat positions

Meaning: The downbeats and beats represent the musical temporal grid of a music track. This information is necessary to understand the role of the various musical events existing in the music track. *Visual representation:* Therefore, the downbeats and beats are represented as vertical lines crossing all other types of description. Thick vertical lines are used for downbeats, thin ones for the other beats. *Extraction and annotations:* The beat positions can be estimated automatically from the audio signal using for exam-

ple the algorithm described in [18], the downbeat positions using the one described in [16] or any algorithm conforming to the proposed XML format definition. Beat positions can be corrected manually using the marker system explained in part 3.5. They can also be manually "annotated while listening" by putting "on the fly" markers. This is done using keyboard shortcuts. *Storage:* The downbeat and beat positions are stored on a XML file. *Interface feedback:* The interface provides an audio feedback each time the play-head crosses a beat marker. This allows to "check by listening" the accuracy of the beat marker positions.

3.4.5 Multi-pitch information

Meaning: The multi-pitch information represents the various pitches occurring over time. *Visual representation:* They are represented in a piano-roll way as in most current midi-sequencers. Each line represents a specific pitch; the presence of a note at a given time is indicated by a rectangle at this time and this pitch. Up to 16 different note-channels (based on the 16 midi-channels) can be displayed simultaneously. In order to distinguish the various channels, each note-channel has its own specific colour. The left part of the interface displays the number of existing channels in the description loaded. It allows the user to select which channels to display. *Extraction and annotations:* The multi-pitch information can come from an aligned midi file (converted to the XML format) or can be estimated automatically from the audio signal using for example the algorithm described in [20] or any algorithm conforming to the proposed XML format definition. No multi-pitch annotation or correction is possible in the current version of the interface. *Storage:* The various multi-pitch of the note-channels are stored in an XML files. *Interface feedback:* When the user clicks on a specific note-rectangle, a piano sound at the corresponding pitch is immediately played. This allows, for example, comparing a transcription to the original audio file.

3.4.6 Sound-events information

Meaning: Sound events represent the various sound occurrences over time (instruments, percussive sounds). *Visual representation:* The various occurrences over time of sound events are represented in a "sound-event-roll": each line represents a specific type of sound; the presence of a specific sound is represented by a rectangle at the corresponding line and time. *Extraction and annotations:* This information can be estimated automatically from the audio signal using for example the algorithm described in [8] for drum-event sounds and [14] for instruments or any algorithm conforming to the proposed XML format definition. No sound-events annotation or correction is possible in the current interface. *Storage:* The instrument and drum sound-events are stored in XML files. *Interface feedback:* No interface feedback has been implemented so far.

3.5 Specific actions for annotation

An extension of the music track browser has been realized in order to allow annotation by the user: enter or correct the displayed content description. Indeed, it has been showed

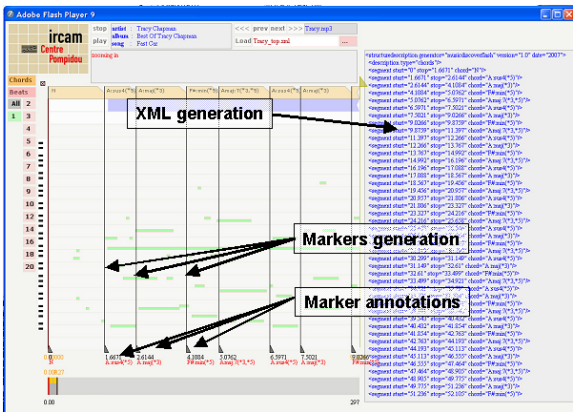


Figure 4. Annotation using the MCIpa interface: markers edition and XML generation

that annotating the content of a music track is greatly facilitated by the knowledge (visualization) of other types of content information. For example, annotating the temporal structure of a track is greatly facilitated by the knowledge (visualization) of the similarity matrix, annotating the chord progression by the knowledge of structure and beat positions ... Annotation is performed using a marker system. The markers can be generated using various methods: • create a marker at the current mouse-cursor position • put "on the fly" markers while listening at the play-head positions (beat annotation) • create a marker at the beginning of the currently selected MCI • generate automatically all the markers corresponding to a specific type of MCI. The markers can then be moved or deleted. Each marker has an associated text label which is editable. If the markers are generated from an existing MCI, the label will be the one of the MCI (for example the one of the chord TAB). An annotation can start from scratch (by generating manually the markers) or by correcting an existing annotation (by generating automatically all the MCI markers and then correcting the markers position and labels). Once finished, the XML code corresponding to the given annotation can be automatically generated and reloaded (see Fig. 4).

4 DEVELOPMENT DETAILS

Graphical interface: The graphical interface has been developed using the FlashDevelop ActionScript 3 environment. The interface is written in the Flash/ ActionScript 3 language, for faster display of numerous items and easy user interaction with the displayed objects. It can easily be run on any platform supporting the Adobe Flash 9 plugin (Linux PC, Windows PC, Mac-OS-X apple computers, portable devices ...). A standalone version can also be run on Mac-OS-X or Windows PC.

Database management: The database management part (music database browser) is calling a PHP script that automatically polls a MySQL database of music titles. This PHP script returns the polled information under the form of a top XML file, which is then read back into the Flash interface.

Any music title has 4 identifiers: title, album, author, music-genre and the path to the top XML file. The top XML file provides the path to the mp3 file and to the various content description XML files (see Fig. 3). When the user asks for a specific title, he just has to click in the list of tracks. The PHP / MySQL / Apache management has been done using the XAMPP environment.

5 CONCLUSION

In this paper, we presented "MCIpa" a Music Content Information player and annotator. The "MCIpa" is a Flash-based (cross-platform) graphical interface that allows intra-document browsing of music track based on content-description. It is also a music-content annotator tool guided by the visualization of other music-contents. The tool represents each type of music-content (similarity matrix, music structure, chord progression, downbeat and beat positions, multi-pitches, sound-events) by a specific Music Content Information object with a specific meaning, visual representation, interface-feedback, extraction-algorithm and XML file format. The XML formats used by MCIpa as well as the software itself are available [2] . We hope this will help people use their favourite content-extraction algorithm with this tool. Because new types of content-description can easily be added to this tool, it could be used as a standard music-content-description player and annotator.

Future works: The paradigms used by MCIpa have been partially tested during the user-testings of the Semantic HIFI intra-document "player" [4]. However they still need to be tested for annotation tasks. Future works will therefore concentrate on that. For this, an experimental protocol for annotation (choice of a set of annotation tasks to be performed and a set of music items) must first be established. Future works will also concentrate on extending the current architecture of MCIpa to a plug-in architecture.

MCIpa usages: We believe that MCIpa can be used for many different purposes since the visual representations used can be easely understand by a large number of people without dedicated signal processing or musical knowledge. MCIpa could be used as a standard media player, for musical education, for comparative musicology (some of the graphical representations of MCIpa are currently used by musicologist at Ircam to compare various interpretations of the same piece; integration into Ircam online mediatheque is also under study), as a musician practicing tools (when playing over music -such as with Aebersold methods- being able to visually locate "theme", "chorus", bars and repetitions is of great help), for research purposes (for quick visualization of results obtained with content-extraction tools) and of course to create annotated training or test-sets.

[2] http://recherche.ircam.fr/equipes/analyse-synthese/peeters/mcipa/

6 ACKNOWLEDGEMENTS

This work was supported by the French projects ANR "Music Discover" [3] and Oseo "Quaero" [4] . Thanks to Koen Tanghe and to Frederic Cornut for comments and helps.

7 REFERENCES

[1] X. Amatriain, J. Massaguer, D. Garcia, and I. Mosquera. The clam annotator: A cross-platform audio descriptors editing tool. In *ISMIR*, London, UK, 2005.

[2] Audacity-Development-Team. Audacity: Free audio editor and recorder, 2006.

[3] P. Boersma and D. Weenink. Praat: Doing phonetics by computer, 2006.

[4] G. Boutard, S. Goldszmidt, and G. Peeters. Browsing inside a music track, the experimentation case study. In *LSAS*, Athens, Greece, 2006.

[5] C. Cannam, C. Landone, el al. The sonic visualiser: A visualisation platform for semantic descriptors from musical signals. In *ISMIR*, Victoria, Canada, 2006.

[6] DGA. Transcriber, 2007.

[7] J. Foote. Visualizing music and audio using self-similarity. In *Proc. of ACM Int. Conf. on Multimedia*, pages 77–84, Orlando, Florida, USA, 1999.

[8] O. Gillet and G. Richard. Supervised and unsupervised sequence modelling for drum transcription. In *Proc. of ISMIR*, Vienna, Austria, 2007.

[9] F. Gouyon, N. Wack, and S. Dixon. An open source tool for semi-automatic rhytmic annotation. In *Proc. of DAFX*, Naples, Italy, 2004.

[10] GRM. Acousmographe, 2007.

[11] P. Herrera, O. Celma et al. Mucosa: A music content semantic annotator. In *ISMIR*, London, UK, 2005.

[12] Ircam. As (audioscuplt) annotation, 2007.

[13] B. Li, J. A. Burgoyne, and I. Fujinaga. Extending audacity for audio annotation. In *ISMIR*, Victoria, Canada, 2006.

[14] A. Livshin and X. Rodet. Musical instrument identification in continuous recordings. In *Proc. of DAFX*, Naples, Italy, 2004.

[15] MPEG-7. Information technology - multimedia content description interface - part 4: Audio, 2002.

[16] H. Papadopoulos and G. Peeters. Simultaneous estimation of chord progression and downbeats from an audio file. In *IEEE ICASSP*, Las Vegas, USA, 2008.

[17] G. Peeters. Sequence representation of music structure using higher-order similarity matrix and maximum-likelihood approach. In *Proc. of ISMIR*, Austria, 2007.

[18] G. Peeters. Template-based estimation of time-varying tempo. *EURASIP Journal on Advances in Signal Processing*, 2007 Article ID 67215

[19] K. Sjolander and J. Beskow. Wavesurfer - an open source speech tool. In *ICSLP*, 2000.

[20] C. Yeh, A. Roebel, and X. Rodet. Multiple fundamental frequency estimation of polyphonic music signals. In *Proc. of IEEE ICASSP*, Philadelphia, PA, USA, 2005.

[3] http://recherche.ircam.fr/equipes/analyse-synthese/musicdiscover/

[4] http://www.quaero.org