MuVis: An Application for Interactive Exploration of Large Music Collections

Ricardo Dias, Manuel J. Fonseca
Department of Information Systems and Computer Science
INESC-ID/IST/Technical University of Lisbon
ricardo.dias@ist.utl.pt, mjf@inesc-id.pt

ABSTRACT
In this paper we present MuVis, an interactive visualization and exploration tool for large music collections, based on music content and metadata. We combined a user-centered design with three main components: information visualization techniques (based on semantic ordered treemaps), music information retrieval mechanisms (for semantic and content-based information extraction) and dynamic queries, to offer users a more efficient, flexible and yet, easy to use solution for browsing music collections and to create playlists. Preliminary results reveal that our solution is faster and easier to use than the Windows Media Player, allowing users to perform a more effective and fast navigation, while getting a deeper knowledge of their library. Satisfaction survey revealed that users liked our approach for browsing, filtering and creating playlists, while at the same time they were able to “re-discover” forgotten music, due to the similarity mechanisms incorporated in our solution.

Categories and Subject Descriptors
H.5.2 [Information Interfaces and Presentation Tools]: User Interfaces - GUI

General Terms
Design, Experimentation

Keywords
Music Exploration, Dynamic Filtering, Playlist Generation

1. INTRODUCTION
Due to the increase in popularity of digital music during the last decade, the size of personal music libraries has enlarged, rising the need for efficient and easy to use mechanisms for organizing and exploring large collections.

Although there are some solutions to organize and browse music, they present some limitations when dealing with large collections, making tasks like search and browse cumbersome. Furthermore, despite the fact that semantic information have been widely used, its presentation has shown little progress or innovation during recent years. Additionally, the flexibility offered by playlist creation mechanisms need to be improved, through the usage of semantic and content-based information of music.

Some recent works explored visualization techniques like the Self-Organizing Map (SOM) [5], the use of graphs (Musicover1, LivePlasma2), treemaps [10] and other original approaches [6] to present the content of music collections. Our solution differs from these, since it provides a wider approach for music collection visualization, combining an overview of the library, an interactive technique for browsing (and filtering) and a mechanism for playlist creation using both semantic and content information from the music.

In this paper, we describe MuVis3, an application with a novel approach for large music libraries exploration and visualization that allows users to easily and efficiently navigate, explore and view their music collection, as well as to create playlists. MuVis uses both semantic and content-based information from the music, to provide an overview of the overall collection and to offer an interactive browsing and filtering mechanism (see Figure 1).

In the remainder of this paper, we discuss in section 2 a user study conducted to collect information about users’ habits and behaviours when listening to music. Section 3 de-

3The tool is available for download from http://sourceforge.net/projects/fmuvis/
scribes our approach, detailing the visual design of the user interface and the interactive filtering mechanism. In Section 4 we present the results from the experimental evaluation with users. Finally in Section 5 we conclude this paper and discuss future work.

2. USERS STUDY

Although some studies have been conducted to understand users’ needs while listening to music [2, 4], they did not study how users execute typical tasks. So, before developing our solution, we started by performing a study about users’ habits and behaviours in the exploration and browsing of their music collections.

Our study was composed of an online survey and a contextual inquiry. For these, we created a questionnaire to discover how users find music, explore their libraries and generate playlists. We also tried to identify which features and criteria for querying they would like to have in a new application for music visualization and exploration.

In the online survey, 127 participants participated, being 65% male, and the majority (81%) aged between 20 and 29 years old. Almost all of them (94%) listen to music on their personal computers and have a digital music library (84%). The most typical library size is between 1,000 and 10,000 tracks (40%). However, their collections were not very diverse. Indeed, more than two thirds of users’ collections have rock and pop songs. From these values, we believe that genre is not a good criterion for structuring personal music libraries.

Results revealed that there is no difference between using a specific audio/media-player (47%) and the filesystem (45%) for searching songs, leading us to infer that current applications for music organization do not have any relevant mechanism that distinguishes it from the traditional filesystem structure. However, most users reported they would like to search music using tags (54%), 48% using mood and 46% by song similarity. Regarding the music library browsing, we identified a clear order in the way users navigate through their collections: Artist (46%), Album (41%) and Genre (41%). Concerning playlists, most of them are created by selecting individual tracks (60%), 10% using filters and automatic smart playlists and 11% randomly. Participants would like to use genre (51%) as the main criterion to create automatic playlists, but also by using general tags (37%) and mood (37%).

For the contextual inquiry, we selected ten users from both genders aged between 20 and 40 years old; with collections around one thousand tracks and that listen to music on the computer often. The contextual inquiry consisted in performing some typical tasks in their audio player application and answering the same questionnaire, complemented with an informal interview. Results from the contextual inquiry supported those obtained with the online survey.

From the users’ study we identified three main design implications: 1) Browsing: The exploration of the music library should start by the selection of Artist, then Album, and finally Genre; 2) Playlists: Users enjoy having control over the playlist generation process, to be sure that some particular songs are on the list. To that end, we would like to provide some filtering mechanisms based on tags and/or music content to help users select the desired tracks; 3) Searching: To satisfy users’ requirements, we should provide at least three types of searching: i) using tags (those included in MP3 files, or added manually); ii) by selecting the mood (e.g. calm, energetic, etc.); and, finally, iii) by music similarity.

3. OUR SOLUTION

3.1 Design Objectives

The main design objectives of MuVis are to facilitate the exploration of large music collections and the creation of playlists. To achieve these objectives, MuVis provides an overview of the music collection, offers a retrieval mechanism based on music similarity and semantics, and supports dynamic filtering to search and explore music.

We expect to help users answer questions like: What type of music do I have in my collection?; From what artist do I have more albums/music?; What Rock music do I have from the 80’s?; What calm music do I have?; What are the music/albums/artists most similar to this music/album/artist?

3.2 Architecture

The architecture of our solution (see Figure 2) is composed by a feature extraction module, a retrieval component and the user interface.

![Figure 2: MuVis architecture.](image)

The feature extraction module is responsible for the extraction of both semantic information (common tags, like track title, duration, genre, etc.) and content-based audio features (Fluctuation Patterns [8] computed using the CoMIRVA toolkit [9]). Additionally, this component extracts audio snippets, a representative excerpt of the audio track with a short duration (10-15 seconds).

The retrieval component searches for similar tracks using the two types of information: tags, stored in a relational database, and content information, stored in a multidimensional indexing structure (the NB-Tree [3]). The indexing structure is used because the fluctuation pattern is a vector of 1200 positions and we need to perform efficient k-nearest neighbours (KNN) searches in a multidimensional space.

The user interface includes two visualization techniques, the list view and the spatially ordered treemap, and is responsible for managing the interface and the interaction with the user. A detailed description of the user interface and their main components is provided in the next subsections.

3.3 Visual Design

The user interface of our application is divided in five areas, as illustrated in Figure 1, and has three key elements: the two views over the collection (appear in area 1) and the dynamic filtering component (area 2).

Area 1 is the main area where the audio tracks are presented to users using two different views: list view and treemap view. For the latter we used a modified version of the spatially ordered treemap [12]. In this view, items

(that represent artists) are mapped by their musical similarity to the pivot artist that is located in the top left corner. The properties of the treemap nodes, such as size, color and layout can be controlled through the user interface. For the layout, users can select the pivot artist at any time, and re-organize the treemap. Rectangles’ size can be the same for all artists, proportional to the number of tracks, or proportional to the number of albums. Additionally, the rectangles can be decorated using colors (which represent the “artist genre”, according to the color code used in the Musicover system), the albums’ artwork or both. To avoid a large number of small rectangles that could make the visualization and browsing difficult, we group artist with a small number of music (three tracks) in the “Others” node. Users can zoom-in by selecting an artist to see its albums and the respective tracks.

The audio player (area 4) and the playlist manager (area 3) are the other modules of the interface. Area 2 contains the interactive filtering mechanism, which explores the treemap concept for visualization and filtering, and is described in the next subsection.

### 3.4 Interactive Filtering

Dynamic query interfaces [1] allow users to formulate queries dynamically and get immediate feedback. We used this concept in our solution to provide users with an interactive filtering mechanism (Figure 1, area 2), which allows them to interactively filter their collection and explore it, through the combination of several individual filters that restrain the current view of the collection, just to see the songs that meet the requirements (filters). Users can in real time add or remove any filter.

Our application offers seven filters: plain text, musical similarity (song, album and artist), duration, year, genre, beat, and mood, which are a direct consequence of the findings from the users’ study. The usage of the treemap visualization technique in this component, provides a visual clue about the distribution of music per category and per values (based on their size), and also the possibility to interactively combine multiple filters/categories, driving users to perform queries like those described in section 3.1.

Each filter is internally represented as a set of rules that define the constrains to be applied to the raw data. The rules combine information from the relational database (tags) and from the indexing structure (content), allowing a fast filtering, and not requiring extra elements to support the browsing and filtering operations.

### 3.5 Dynamic Playlist Generation

MuVis allows users to create playlists automatically using dynamic or static generation.

Dynamic playlist generation denotes a continuous addition of new tracks to the current list, based on the similarity of new tracks to those already in the list. Users start by selecting a group of music to serve as the initial seeds and then the algorithm adds new tracks every time the playlist is reaching the end. New music are found by performing KNN queries to the retrieval component that searches for tracks similar to the specified seeds. This behaviour maximizes the number of different tracks that are added to the playlist, allowing users to “re-discover” tracks and to keep a continuous play. Nevertheless, users can delete unwanted tracks from the playlist, preventing them from being used as seeds.

The static playlist differs from the dynamic, because no new tracks are added automatically to the list. Music added to this list result from the application of a set of filters.

### 4. EVALUATION

To check whether our objectives have been met, we conducted an experimental evaluation with ten volunteer users from both genders, aged between 19 and 40 years. All of the participants have a large music collection and use the computer on a daily basis, both for work and to listen to music. We performed individual tests using the same computer, with MuVis and Windows Media Player (WMP), loaded with a collection of around 5,000 songs from different genres such as, rock, pop, r&b, metal, jazz, etc. This way all users were in the same situation, all of them have the same set of music and we have a ground truth to compare the results.

The experimental evaluation consisted of four phases and took around 40 minutes. First, we described the objectives of the evaluation and explained the functionalities of the two applications. Next, we provide an adaptation period, where we let users freely use the applications. After this, we gave users four tasks (see Table 1) which they performed in the two applications. To avoid biasing the results, half of the users started by the MuVis application while the other half started by using WMP. Finally, users answered a satisfaction questionnaire and we performed an informal interview.

### 4.1 Results

In general, all participants performed faster in our application than in WMP (except three in Task 1). Moreover, they understood well the filtering mechanism based on treemaps and liked the similarity searching functionality.

The average execution time for the four tasks is smaller in our application (two minutes) than in WMP (eight minutes). A t-test with pairwise samples revealed that the average time taken using MuVis (mean = 109 sec, SD = 37.0) was significantly smaller than using WMP (mean = 532 sec, SD = 153.2, p < 0.001). We also analyzed the time

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>Find the song “Wonderful Tonight” from Eric Clapton and listen to it.</td>
</tr>
<tr>
<td>T2</td>
<td>Create a playlist with the top 12 Pop songs from the 90’s, lasting between 3-5 minutes.</td>
</tr>
<tr>
<td>T3</td>
<td>Write the name of the artist with most songs and of the artist with most albums in the collection.</td>
</tr>
<tr>
<td>T4</td>
<td>Find similar albums to the first one of the artist with most albums.</td>
</tr>
</tbody>
</table>

Table 1: Tasks used for the usability evaluation.
that users took to perform each task on the two applications, to understand how each system behaves for the different kind of tasks (tag search, content similarity or exploration).

On average, users performed Tasks 2, 3 and 4 faster in our system than in WMP (see Figure 4). The pairwise t-test shows that the average time taken on each task is significantly smaller in MuVis than in WMP, with a value of $p < 0.002$ for the three tasks. For Task 1, although on average users took less time in MuVis (13.4 sec) than in WMP (14.5 sec), we cannot say that it is faster since $p < 0.32$.

These results show that our application performs as well as the WMP application for tasks that involve (typical) text-based searching (e.g. artist, album, song, genre, etc.), but performs better on tasks that involve browsing and exploration of the collection (Tasks 2 and 3) or similarity searching (Task 4). According to users, this was due to the fact that both music collection and filters have a spatial and visual common representation (treemaps), which helped users explore the collection. Indeed, in Task 2, which is a browsing task, we can see that the difference in time is enormous (more than three minutes). Users took 85% less time performing the task in our tool than in WMP.

The majority of the participants expressed high satisfaction with our solution, mainly because it provides a simple and interactive browsing of the music collection, using a visual and easy to use filtering mechanism (almost 80% agreed), that allows them to efficiently locate music. The creation of playlists was also described as easy (60%) because of its flexibility, since we can create a playlist almost in all screens of the application. Another aspect mentioned by users is that the sequence “filtering + playlist creation” is appropriate for a simple and practical use.

Finally, we noticed that users were able to get a correct overview of the collection content, by identifying the main genres (Pop, Rock, R&B), the artists with most albums and with most music. The users mentioned that the size of the different rectangles helped them to have a good visual perception of the overall collection. Moreover, they also said that the visual representation of the filters (treemaps) assisted them in understanding the set of music and the way they were distributed by duration, genre, mood, etc. On the other hand, all the participants said that it was impossible to answer this question using the WMP application.

5. CONCLUSIONS

In this paper we presented a new solution for interactive visualization and exploration of large music collections. The resulting application, MuVis, provides a quick and simple way to interact with the music library, where users can get a visual overview of their collection and/or create playlist using interactive filters. Experimental results showed that our goal of providing a tool for efficient visualization and exploration of music collections was achieved and fully satisfied the users. The visual representation of the filters using the paradigm of treemaps also allowed the users to get a better insight about their music libraries.

Our future work will consist of the development of new visualization techniques, applying approaches like the ones explored in the CollectTable [7], and those like zoom, fisheye, multi-focus and context aware solutions (like the one introduced in [11]). Additionally, we want to go further in the evaluation of the proposed solution, using more participants and defining new tasks, to fully determine the consequences of the usage of similarity in the treemap visualization.

6. ACKNOWLEDGMENTS

This work was supported by FCT (INESC-ID multiannual funding) through the PIDDAC Program funds, by ADI through the ColaDI project and through the Crush project, PTDC/EIA-EIA/108077/2008.

7. REFERENCES