MUSESCAPE: AN INTERACTIVE CONTENT-AWARE MUSIC BROWSER

George Tzanetakis

Computer Science Department
Carnegie Mellon University, USA
gtzan@cs.cmu.edu

ABSTRACT

Advances in hardware performance, network bandwidth and audio compression have made possible the creation of large personal digital music collections. Although, there is a significant body of work in image and video browsing, there has been little work that directly addresses the problem of audio and especially music browsing. In this paper, *Musescape*, a prototype music browsing system is described and evaluated. The main characteristics of the system are automatic configuration based on Computer Audition techniques and the use of continuous audio-music feedback while browsing and interacting with the system. The described ideas and techniques take advantage of the unique characteristics of music signals. A pilot user study was conducted to explore and evaluate the proposed user interface. The results indicate that the use of automatically extracted tempo information reduces browsing time and that continuous interactive audio feedback is appropriate for this particular domain.

1. INTRODUCTION

Recent advances in hardware performance, network bandwidth and audio compression have made possible the creation of large personal digital music collections. It is not uncommon for users to have thousand of music files stored on their personal computer and a large percentage of network traffic is used for transferring such files. Although the business and copyright issues are still debated it is probable that in the near future all of recorded music will be available digitally. The use of digital technology has also significantly lowered the cost of recording and increased the music production of non-professionals who use the web as their medium of distribution.

Despite the increasing interest in digital music distribution, the searching, browsing and organizing of music remains unchanged. The user can only search by artist, genre, album name, and in some cases by additional metadata information. In addition the only way to browse large music collections is using a standard file manager. Recent software music players allow the creation of playlists to avoid having to explicitly select each file for playing. However in all these cases, fundamentally, the user can only search for known music and there is little support for browsing unknown music. Music signals have unique characteristics that differentiate them from other multimedia signals such as speech and video. Although there has been a considerable amount of work in image and video browsing there is little work that directly addresses the problem of audio and especially music browsing.

2. RELATED WORK

The concept of browsing is central to this work. In contrast to direct search where the target is known, in browsing the user wants to explore a collection of items without a specific goal in mind. Although browsing seems to be a common behavior of music listeners (observe consumers in a record store) most currently available systems only support traditional keyword and metadata text-based direct search. The main disadvantage of this approach is the need for manual annotation to initially acquire the metadata information.

An alternative to current keyword-based search is the idea of content-based analysis and retrieval. Query-by-example has been a common query specification method. In this approach the user provides a sample multimedia object for the query and similar multimedia objects are returned. The references provided are representative and were mainly chosen based on their influence in the design of our system.

Examples of systems for image organization and retrieval include [1, 2]. Systems for browsing video such as [3, 4] provide various capabilities such as pause removal and time compression, textual and visual indexes, and personalized navigation using shot boundaries. Similar capabilities are provided in speech and notebook interfaces such as [5, 6]. Another related area is audio-based media spaces such as [7] that use audio for multi-user communication.

The most related work to *Musescape* is the Sonic Browser described in [8], which is a graphical user interface based on direct manipulation-sonification for browsing collections of audio signals. Each sound file is represented by a visual shape on a two dimensional plane and a cursor with an aura around it that is used for exploration. The sound files that fall inside the aura are simultaneously played back spatialized based on their distance and location relative to the cursor. Although initially the placement of objects-sounds was performed manually, recently the Sonic Browser has been extended with automatic placement based on feature extraction and audio analysis [9]. The idea of using simultaneous playback and audio spatializing has been explored for audio document browsing in the Dynamic Soundscape described in [10].

It is important to note that there has been a large amount of work in music retrieval and browsing using symbolic music representations such as notated scores or MIDI (Musical Instrument Digital Interface) files. The symbolic and hierarchical nature of these representations allows higher levels of processing and understanding resulting in interesting interfaces such as query-by-humming systems [11] and content-based navigation in scores [12]. Automatic musical genre classification of audio signals is described in [13] and tempo detection in [14, 15]. A general source of information for music information retrieval is [16].

3. DESIGN CONSTRAINTS AND GUIDELINES

From the beginning of this project an effort was made to leverage knowledge gained from previously published work in multimedia browsing rather than building from scratch. Because the main focus of this work is music it is important to comment on the characteristics that differentiate musical signals from other types of multimedia signals such as images, video and speech. These characteristics provide unique guidelines and constraints for developing a music browsing system.

Unlike video or speech signals, recorded music has a basic processing unit: the CD track, which corresponds to a self-contained musical piece for the majority of commercially recorded music (exceptions are for example long symphonic ”Classical” music pieces). Therefore for this work, browsing within a piece of music, although interesting and challenging, will not be addressed. In this respect music browsing is closer to image browsing where the basic unit is the image. On the other hand the temporal nature and sequential nature of listening does not allow multiple musical pieces to be presented at the same time in the same way as images. Although spatial audio has been used for simultaneous presentation of multiple audio streams such as speech signals in [], sound effects in [8], and music in [9], it can only be applied to a small number of items (4-5) something which is not true for images. Another important distinction from video and speech signals is that although music conveys emotions and structure it does convey specific information. Therefore, the use of text analysis and retrieval has limited applicability to musical signals in audio format. Although external text information about the music such as critic reviews and descriptions can be analyzed, the techniques of using this information for searching and browsing are well known and therefore will not be considered in this paper. Time compression and pause removal are techniques that have been used for quick navigation-browsing of temporal media such as video and speech signals. However, these techniques produces audible artifacts therefore are not applicable to music.

To summarize many of the techniques employed in current multimedia browsing interfaces are not directly applicable to musical signals. On the other hand audio signals have some advantages over other types of multimedia systems. A relative consistent set of musical genres can be used to characterize musical pieces. Although genre boundaries are sometimes fuzzy, their perception by different subjects is relatively consistent. Unlike, video or speech genres which are not as well defined, identification of the genre of a musical piece can be done surprisingly fast. In the study reported in [17] subjects could identify genre much better than random guessing (40% in a 10-genre forced choice paradigm) using only 250 milliseconds or 0.25 seconds. Their identification ability improves up to 70% at 3 seconds. Using more information than 3 seconds does not increase genre identification and the remaining confusion is caused by disagreement between subjects about genre definitions. In contrast, genre identification in video or speech signals requires more time as more semantic information needs to be gathered. Finally, musical signals require less cognitive attention than video and speech signals (more people work while listening at the same time to music than people who work watching video at the same time). Based on these observations, a set of design guidelines were established and used for developing our system: 1) the basic object or unit is a musical piece that typically corresponds to a CD track and no time compression and within-piece navigation is supported, 2) music should be playing all the time while the system is used unless the user explicitly stops it, 3) the system directly exposes basic attributes that have perceptual significance such as genre and tempo directly to the user interface level, 4) the system should be easy to learn and use.

In addition to the guidelines and constraints posed specifically for the task of music browsing we had a number of practical issues related to our resources and goals. Probably the most important constraint we had to work with was the lack of resources to have a large subject pool and perform long experiments over time. In order to address this issue, a two-stage approach was used. Two groups of users where used for the design, development and evaluation of the system. The first group, consisting of 4 users, was used in a user-centered iterative design and development process. Users were provided successive working prototypes of the system, used the system on their own computers, reported their suggestions, problems and experiences, and the process was repeated. No formal questionnaires and usage statistics were used at this stage as our purpose was to investigate the system in a natural everyday usage setting. This approach was facilitated by the fact that music browsing was an activity that all subjects enjoyed doing and were all very interested in participating in the design process. Once the prototype reached a stage that the authors and the first user group considered satisfactory, a second group of users who had no prior exposure to the system were used to conduct a more specific monitored usage study designed to answer some basic questions we had about our system. Another design consideration was the desire to have a system that can be used in a mode that requires limited visual screen estate. The two main reasons behind this desire were: the possibility of using the developed system on portable digital music players, and the observation that users in many cases would like to use the music browser while simultaneously using most of their computer screen real estate for other purposes. Finally, in the overall design of the system, the Shneiderman [18] mantra for the design of interaction systems ”overview, zoom and filter, then details on demand” was followed.

3.1. Notes on terminology

Musical genres are categorical descriptions created by humans to organize and characterize different pieces of music. Pieces of a particular musical genre such as ”Rock” or ”Reggae” music share similar characteristics related to the texture (number, density and types of instrument playing), rhythm (the way the sounds are structured in time) and harmony (the way the sounds are structured in frequency). The beat or tempo of a musical signal can be loosely defined as the average frequency corresponding to the sequence of pulses resulting from tapping along with the music. It is measured in beats-per-minute (BPM) and faster pieces have higher BPM.

4. MUSESCAPE DESCRIPTION

The design and development of the system follows the Model-View-Controller paradigm [19]. The Model part comprises the actual underlying data and the operations that can be performed to manipulate it. The View part describes the specific ways the data model can be displayed and the Controller part describes how user input can be used to control the other two parts.

4.1. Model

The Model part is a collection of music files in audio format each of which is annotated with categorical and ordinal attributes that characterize the content. The attribute annotation can be manual, automatic or both. It can also be edited while the users interact with the system using the same interface used for browsing and searching. Any number of attributes can be supported. For the experiments described in this paper two attributes are used: musical genre and tempo.

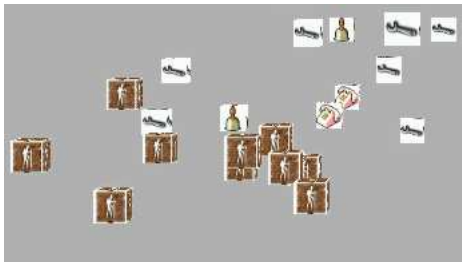
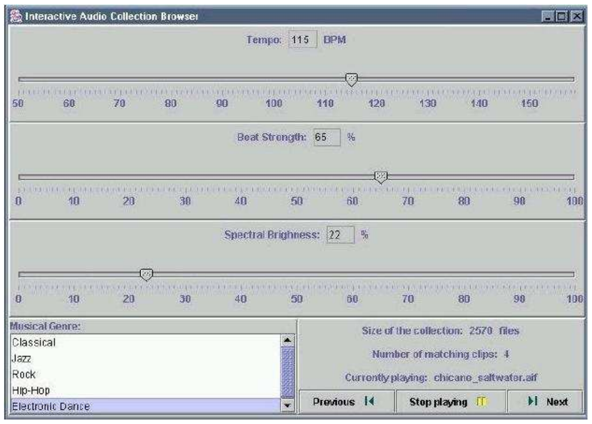


Figure 1: *Visual space for sound effects collection.*

4.2. View

For viewing we use a visual space where each music file is represented as an object whose appearance and position reflect its corresponding attributes. Because our approach is inspired by the system presented in [8] it will not be further described. Although our system is described mainly in the context of music browsing it can also be used for other types of audio signals. Figure 1 shows a visual space for sound effects where the brown cubes correspond to walking sounds and the white cubes to various types of tools.



Tempo: 115 BPM

Beat Strength: 65 %

Spectral Brighness: 22 %

Size of the collection: 2570 files

Number of matching clips: 4

Currently playing: chicano_saltwater.aif

Figure 2: *SoundSliders and SoundList.*

4.3. Controller

The Controller part of the system is the main contribution of the paper and therefore will be described in more detail. There are three operations that we want to unify under a common interface: searching, browsing and annotating. In current systems each of these operations requires a separate tool. Another problem is that in current systems audio feedback is only available during playback and not during query specification. The fundamental idea for solving these problems is to provide controller components that provide direct interactive sonfication of the user actions and are re-used for searching, browsing and annotation. This idea will be illustrated with the example of a volume control slider. If such a slider were to be designed as current music browsing systems, then its value would be adjusted without any actual change in volume and then a submit button would be pressed for the actual volume change to occur. The usual way of having the volume change directly based on the user actions is preferable. For each music content attribute such as tempo, a component which we call "filter" is created with the purpose of specifying the value for that attribute. "Sound-Sliders" are used for continuous and ordinal attributes and "Sound-Lists" for categorical attributes (see Figure 2). Visually, these component look like standard components. The main difference is that when there is a user action, the sound immediately reflects it.

One obvious question is how to combine these filters in such a way that whenever there is a user action the music that is returned satisfies all the filter settings. The naive solution of searching for all the files that satisfy the settings has the problem of being to slow of real time interactive playback even when using indexing methods for the data. Because of the desired continuous audio feedback, changes must happen as fast as possible else users are confused about the effect of their actions. To address this problem, an indexing-preprocessing step is utilized. The main idea is to simply pre-sort the data according to all the possible user settings. Another way to view this process is that rather than indexing the data to support arbitrary search, it is specifically indexed to support the possibilities offered to the user by the graphical user interface. This approach will be referred to as "Usage-based indexing".

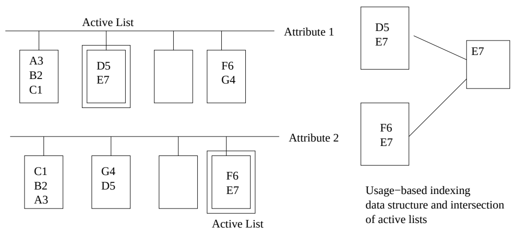


Figure 3: *Usage-based indexing*

Each filter is represented as a list of lists of files. The lists are doubly linked. The first level corresponds to the possible settings for each filter and the lists of files correspond to the files that satisfy a particular filter setting. For categorical items the first level list has a list of files for each category. For continuous attributes quantization bins are used for the first level. For example, a tempo slider with quantization size 5 BPM (beats per minute) will have a second level list of files with tempos ranging from 100-105 BPM. An example of this structure is illustrated in Figure 3. The list corresponding to the current setting is called the active list. Therefore for each attribute there is an active list, and the files that satisfy all the attributes, can be found by finding their intersection (all the files that are common to all active lists). Because only active lists are considered rather that the whole dataset, this operation is fast. When there is a user action, the active list of the appropriate attributed is changed to the active list of the new setting.

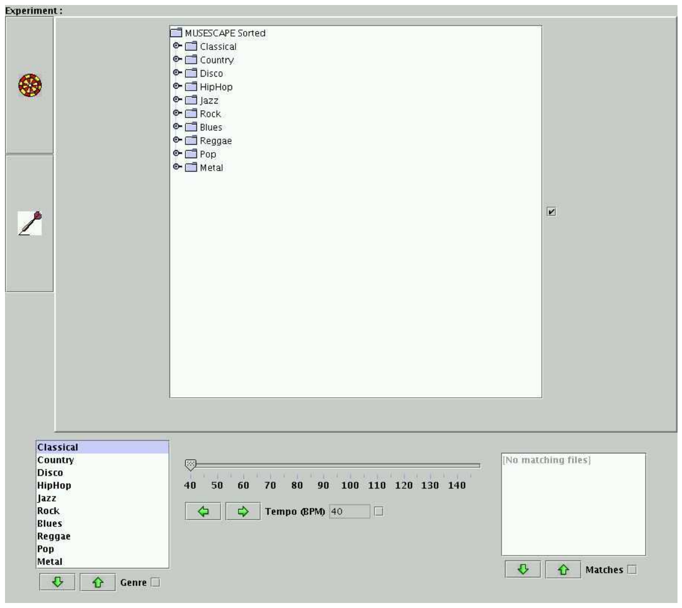


Figure 4: *User study system.*

5. USER STUDY

5.1. Design

A small scale user study was conducted to answer basic questions about the usage of the proposed interface. Our initial plan was to evaluate the full system and compare it with browsing using an hierarchical file manager together with a media player which is the most common way of browsing music today. This initial plan was revised in two ways. The first was that after talking with the first group users and conducting informal tests with users it was clear that continuous direct sonification was much faster and preferable. This has been showed also in other systems such as the Sonic Browser [8]. Once exposed to our new interface users were frustrated with the non-direct interface because it took much longer to complete tasks and couldn't understand why we bothered. In order to address this problem, we decided to still use the hierarchical tree approach of file browsers but enhance it with direct sonification. The tree is the standard collapsible nodes tree and whenever a user selects a node the corresponding musical piece is immediately played back. The first level of the tree corresponds to genre and the second to tempo. The second way we revised our design was to remove the visual display from the evaluation. We had observed that although users liked the visual display after a short time they browsed without paying much attention to it. An additional reason was that we were interested in a small screen real estate version suitable for portable digital music players. Figure 4 shows a screenshot of the system containing both the slider-list configuration and the tree configuration. The target and dart buttons are used for target playback and matching confirmation and will be described in the following section. In the actual study, for each trial either the tree or the slider-list was visible.

5.2. Setup

Participants were two female and four male volunteers with ages ranging from 20 to 30. All participants were regular computer users and familiar with software for music playback. They had no prior experience with *Musescape*. A target matching experiment was used. For each trial, a target sound file had to be located in a collection of 200 sound files. Users could hear again the target file at any time and had to explicitly indicate the match by pressing a button. The reason for the explicit match button was to make sure that users heard the match rather than arrived at it randomly. Two automatically extracted attributes were used: genre (10 genres) and tempo (in Beats-per-minute BPM). The genres were: classical, country, disco, hiphop, jazz, rock, reggae, blues, pop, and heavy metal. The choice of attributes was based on the idea of providing a common categorical and numerical music content annotation. Genre is typically found in current systems based on manually annotated metadata but annotating tempo is time consuming and therefore more difficult to acquire. Both attributes were calculated automatically in our system using the techniques described in [13]. The experiment parameters (200 files, 20 trials, 2 attributes) were chosen as a tradeoff between the desire to have enough data points for statistical analysis while, at the same time, making the experiment enjoyable and not frustrating to the participants. Browsing larger collections than 200 files is not only feasible but it is more satisfying because of the density of choices. On the other hand locating a specific file takes much longer and it would be difficult to conduct user studies.

The experiments were designed to answer the following questions: 1) how long on average do users have to hear a song in order to establish a match 2) does tempo information assist browsing 3) how does the a slider and list-based interface compare to the more traditional tree browsing interface.

Based on these questions the independent variables were: configuration (slider, tree) and tempo information (tempo, no tempo) resulting in four conditions. For the tree interface the first level of the tree was genre and in the tempo case the leaves were sorted from slow to fast. Each subject did twenty matching experiments with five trials for each condition in one session. The session varied from 30 to 45 minutes depending on the subject.

The order was randomized but common across subjects to avoid or at least have common learning effects. In addition, before the experiment a learning session, consisting of eight trials, was conducted. Dependent variables were: matching time, number of wrong matches, number of target replays and average playing time. Matching time was measured as the difference between the time the users first heard the target and the time they correctly identified the match. Number of wrong matches refers to the number of times users pressed the match button without hitting the target file. Number of target replays is the number of times the user pressed the target button to hear again the target and the average playing time is the average time each file was heard before a new user action.

In order to measure these quantities, a detailed time-stamped log of all the user actions was kept and analyzed. The sequence of files browsed was also logged in order to investigate browsing patterns. The data was analyzed using linear ANOVA and an informal questionnaire was conducted after the end of each experiment.

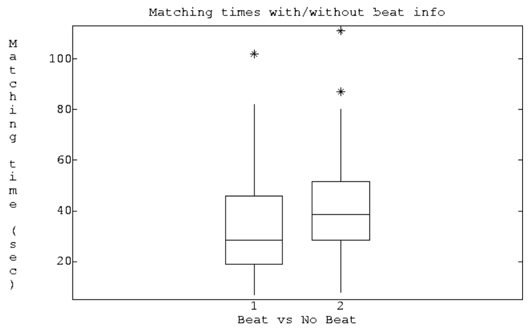


Figure 5: *Matching times with/without tempo information.*

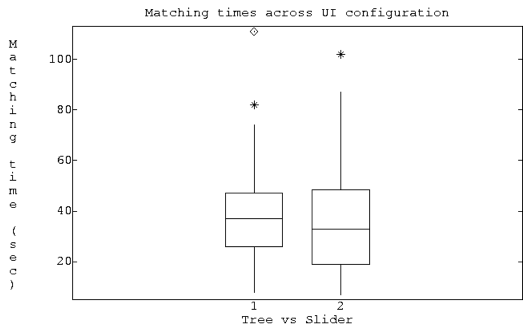


Figure 6: *Matching times for tree/slider configuration.*

5.3. Matching Time Results

The matching time is the main measure of interest in this study. Shorter matching times indicate that the user had to listen to fewer files in order to reach the intended target. Matching time was significantly lower when tempo information was utilized. Figure 5 shows a box plot of matching time with and without tempo information. The means are 33.56 seconds with tempo and 41.43 without tempo. The difference is statistically significant with F = 4.49 and p = 0.03 (for alpha = 0.05). There were few cases where the user would guess the wrong genre and have to visit more than one genre for matching. Without beat information the users had to search linearly through a particular genre to find a file whereas when using beat information a type of binary search based on tempo was performed. The user would initially guess the approximate tempo and then would approach the target by moving the slider left or right depending on if the current file being played was slower or faster than the target.

Figure 6 shows a box plot of matching time across different UI configurations (sliders and tree). Matching time was slightly lower for the sliders. The means are 35.56 seconds for sliders and 39.43 for tree. This small difference is not statistically significant with F = 1.05 and p = 0.3. Therefore there doesn't seem to be any advantage to using trees.

5.4. Other measures

There was no statistically significant difference in the average playing time across all 4 conditions. This is expected as the average playing time is based on how long a user needs to hear a song in order to recognize it. This independence to conditions was also observed in the other two dependent variables: number of wrong matches and number of target replays. The mean playing time was 2.97 seconds with standard deviation of 2.69 seconds. It is interesting how close this average playing time is to the 3 seconds reported in [17] as the minimum playback time for maximum human musical genre classification. These two pieces of evidence suggest that 3 seconds are a good choice for the creation of audio thumbnails. The average number of wrong matches across the subjects was 9 with a standard deviation of 10.9. Three users had zero or 1 wrong matches, one had 8 and two had 24. This indicates two types of strategies with approximately similar results in terms of average matching time. These are: quick listening and paying a time penalty because of wrong matches or slower listening with no wrong matches. The mean number of target replays across subjects for each trial was 2 with a standard deviation of 0.97. That means that approximately 2 replays were done per trial. The average number of files played across subjects for each trial was 20 files with a standard deviation of 4.5 This indicates that after approximately hearing ten files the users had to refresh their target memory.

5.5. Post-study questionnaire

After completing the study, an informal questionnaire was conducted. Given the small number of subjects, simple questions were asked and no statistical analysis of the results was performed. All subjects said that they liked and enjoyed *Musescape*, more than traditional interfaces for browsing music and thought it was more efficient. All subjects also agreed that beat information helped them with browsing. One of the participants said that when he couldn't remember easily a target song he would just memorize the beat and search only based on that information. There was no clear preference between the slider version and the tree version. These answers are in agreement with the results of the experiments. The most common complaint was that sometimes the files was classified in a different genre that what the subject thought it was.

6. CONCLUSIONS - FUTURE WORK

Musescape is a content-aware automatically generated browsing system specifically designed for music signals. Overall both user groups enjoyed their interaction with the system and their input was valuable for the design and evaluation of the system. The usage study results indicate that beat information can decrease music browsing time. Continuous audio feeedback (direct sonification) of all the user actions, seems to work well in the context of music browsing. In addition, it was shown that the less visually demanding slider-list interface performed as well as the standard tree based approach for music browsing. Another important results of this user study, is the average playing time of 3 seconds which seems to contain enough information to adequately characterize a piece of music both for browsing and genre identification. For the future we plan to release *Musescape* as free software. In order to do that we plan to add support for mp3 audio compressed files. In addition, the current support for textual metadata is limited to simple attributes stored in text files so we are planning to migrate the whole system on top of a database. Finally, we plan to add spatial audio for the presentation of the intersection active list.

7. ACKNOWLEDGEMENTS

This work was supported by NSF grant 0085945 of the National Science Foundation.

8. REFERENCES

[1] Hyunmo Kang and Ben Shneiderman, "Visualization Methods for Personal Photo Collections: Browsing and Searching in the PhotoFinder," in *Proc. Int. Con.f on Multimedia and Expo*, New York, 2000, IEEE.

[2] Alex Pentland, Rosalind Picard, and Stanley Sclaroff, "Photobook: Tools for Content-Based Manipulation of Image Databases," *IEEE Multimedia*, pp. 73–75, July 1994.

[3] Michael G. Christel, Michael A. Smith, Roy C. Taylor, and David B. Winkler, "Evolving video skims into useful multimedia abstractions," in *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, Los Angeles, USA, 1998, pp. 171–178.

[4] S. M. Drucker and et al., "SmartSkip: Consumer level browsing and skipping of digital video content," in *Proc. CHI 2002*, Minneapolis, Minnesota, July 2002, ACM Press, pp. 219–226.

[5] Barry Arons, "SpeechSkimmer: a system for interactively skimming recorded speech," *ACM Transactions Computer Human Interaction*, vol. 4, pp. 3–38, 1997, http://www.media.mit.edu/people/barons/papers/ToCHI97.ps.

[6] L. Stifelman, B. Arons, and C. Shmantdt, "The Audio Notebook: paper and pen interaction with structured speech," in *Proc. Computer Human Interaction Conf. (CHI)*, Seattle, WA, July 1, pp. 182–192, ACM Press.

[7] A. Singer and et al., "Tangible Progress: Less is more in Somewire audio spaces," in *Proc. of Computer Human Interaction (CHI)*, Pittsburgh, PA, May 1999, pp. 104–111, ACM Press.

[8] Mikael Fernstrom and Eoin Brazil, "Sonic Browsing: an auditory tool for multimedia asset management," in *Proc. Int. Conf. on Auditory Display (ICAD)*, Espoo, Finland, July 2001.

[9] Eoin Brazil, Mikael Fernstrom, George Tzanetakis, and Perry Cook, "Enhancing Sonic Browsing using Audio Information Retrieval," in *Proc. of International Conference on Auditory Display (ICAD)*, Kyoto, Japan, 2002.

[10] M. Kobayashi and C. Schmandt, "Dynamic Soundscape: mapping time to space for audio browsing," in *Proc. Computer Human Interaction Conf. (CHI)*, Atlanta, GA, Apr. 1997, pp. 194–201, ACM Press.

[11] Asif Ghias, Jonathan Logan, David Chamberlin, and Brian Smith, "Query by Humming: Musical Information Retrieval in an Audio Database," *ACM Multimedia*, pp. 213–236, 1995.

[12] C. DeRoure, D. and S Blackbrun, "Content-based navigation of music using melodic pitch contours," *ACM Multimedia Systems*, vol. 8, no. 3, pp. 190–200, 2000.

[13] George Tzanetakis and Perry Cook, "Musical Genre Classification of Audio Signals," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, July 2002.

[14] Eric Scheirer, "Tempo and beat analysis of acoustic musical signals," *Journal of the .Acoustical Society of America*, vol. 103, no. 1, pp. 588,601, Jan. 1998.

[15] Jean Laroche, "Estimating Tempo, Swing and Beat Locations in Audio Recordings," in *Proc. Int. Workshop on applications of Signal Processing to Audio and Acoustics WASPAA*, Mohonk, NY, 2001, IEEE, pp. 135–139.

[16] *Proc. Int. Conference on Music Information Retreival (ISMIR)*, Paris, France, 2002.

[17] D. Perrot and Robert Gjerdigen, "Scanning the dial: An exploration of factors in identification of musical style," in *Proc. Society for Music Perception and Cognition*, 1999, p. 88, (abstract).

[18] Ben Shneiderman, *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, Addison-Wesley, 3rd ed. edition, 1998.

[19] Glenn E. Krasner and Stephen T. Pope, "A cookbook for using the model-view-controller user interface paradigm in Smalltalk-80," *Journal of Object-Oriented Programming*, vol. 1, no. 3, pp. 26–49, Aug. 1988.