Visually and Acoustically Exploring the High-Dimensional Space of Music

Lukas Bossard
Computer Engineering and Networks Laboratory
ETH Zurich, Switzerland
lbossard@ee.ethz.ch

Michael Kuhn
Computer Engineering and Networks Laboratory
ETH Zurich, Switzerland
kuhnmi@tik.ee.ethz.ch

Roger Wattenhofer
Computer Engineering and Networks Laboratory
ETH Zurich, Switzerland
wattenhofer@tik.ee.ethz.ch

***Abstract*—The permanent growth of personal music collections caused by the ongoing digital revolution asks for novel ways of organization. Traditional list based approaches are—particularly for use in mobile devices—no longer appropriate. In this paper we discuss two alternative exploration schemes, both taking advantage of a recently proposed high-dimensional music similarity space. First, we present an interface that facilitates visual navigation through a collection. Then, we propose the use of an acoustic approach, which adapts to the user's taste. The proposed solutions have been integrated into an Android based prototype application. The conducted user experiments indicate that both methods outperform alternative approaches that head into similar directions.**

***Keywords*-User Interfaces; Mobile Applications; Music; Interactive data exploration and discovery; Information Visualization**

I. Introduction

Over the past years we have observed a tremendous change in the way people interact with media.

Nowadays it is simple and cheap to acquire music. Thus, the collections of many music lovers have reached sized that make it hard to keep an overview by traditional organization techniques, such as browsing hierarchies of folders or searching for songs by means of manually assigned labels. This is particularly true for mobile devices, that feature limited input and output capabilities.

We are convinced that the concept of music similarity will play a crucial role in future organization strategies. Recently, Goussevskaia et al. [2] have derived a measure of music similarity by investigating the listening behavior of users of a social music platform, and have shown that this similarity measure is intrinsically high-dimensional.[1] In form of a web-service they provide the coordinates of more than 400K songs, which have been mapped into a 10-dimensional Euclidean space that well reflects the underlying similarities. In this paper we investigate how this high-dimensional space of music can be explored by a user on a mobile device. We address the problem from two sides, by means of a visual as well as an acoustic interface. These interfaces have been integrated into a proof-of-concept mobile application which is based on the Android[2] mobile platform.

Our interfaces are designed such that users can quickly find music that matches their taste, even within collections they are partly or entirely unfamiliar with. Moreover, the methods attempt to give an overview of the entire collection, such that tracks from different music areas can be discovered. For visual exploration we introduce a lens metaphor that allows to focus on one part of the collection but at the same time retains a global overview. The acoustic scheme, on the other hand, tries to stay as broad as possible to make sure no areas that match the user's taste are missed. It senses the user's behavior and is thus able to adapt its output accordingly.

The usability of our methods has been evaluated in an experiment comprising 9 participants. In this experiment our visual interface outperforms the recently introduced SensMe™feature of Sony Ericsson, which, to the best of our knowledge, is the only currently available commercial system that heads into a similar direction. Due to the lack of commercial mobile solutions in the area, we compare our acoustic approach to the widely used concept of random shuffling as well as to an algorithm proposed by Pampalk et al. [14] that aims at playlist generation based on an audio-feature space. In the conducted experiments, our algorithm scores better than the two alternative approaches.

II. Related Work

The exploration of music collections by means of sophisticated interfaces is an active area of research. The proposed approaches can be classified along different criteria.

The biggest class of solutions tries to visualize collections based on previously extracted audio features. A popular approach is the use of self-organizing maps (SOMs) to create pleasant drawings of the underlying space. Mörchen et al.[11] proposed to apply emergent SOMs (ESOMs) to a complex audio-feature space to retain as much information as possible in the 2-dimensional output space. A 3-dimensional visualization of relatively small collections (50 songs), also by means of SOMs, is described in [7].

[1]Under high-dimensional we understand everything that exceeds the intuitively visualizable number of 2 or 3 dimensions.

[2]http://code.google.com/android/

A spherical SOM has been used by Leitich and Topf [8] to create the globe of music. The PocketSOMPlayer [12] applies SOMs for visualizations on small devices. All these approaches map the audio space into some low (2 or 3) dimensional representation, which can then be explored by traditional navigation schemes.

Goussevskaia et al. [2] have investigated the loss in accuracy of mapping a user behavior based music similarity measure into lower dimensions, and found that 8 or 10 rather than 2 or 3 dimensions are required to adequately reflect the similarity values. The use of the space in [3] demonstrates that mobile devices are able to cope even with it. However, the application only offers textual interfaces as opposed to the more user friendly interfaces proposed in this paper.

Several approaches try to present the complex structures behind music similarity on a 2-dimensional display. Donaldson and Knopke, for example, apply a special node repulsion technique to overcome occlusion effects. Moreover, several more abstract visual interfaces have been proposed to overcome the inherent problems of the low dimensional output space. The artist map proposed by van Gulik et al. [20] uses a spring embedder to visualize a collection along a user definable set of properties, which are partly extracted from the audio content and partly consist of meta-data. The approach is directed at small devices and restricted to artist similarity. A circular layout has been chosen by MusicRainbow [13] as well as AudioRadar [5], which both operate on an audio-feature space, in case of MusicRainbow augmented by keywords retrieved from the web. The interface proposed by Torrens et al. [18], finally, does not rely on any music similarity measure. Rather, it directly visualizes meta-data, such as genre or year of release, in a circular, a rectangular, and a tree-based fashion.

The visualization of high-dimensional structures has also been studied outside of the context of music. A comprehensive overview is given by [17]. Not all the described techniques can be applied to our setting, though, such as the rank-by-feature [16] approach, which depends on known meanings of the axes. Two major ingredients, often seen in high-dimensional data visualization are hierarchical decompositions (e.g. used in [21]) as well as icon based techniques (e.g. applied in [6]). The visual interface we propose in this paper combines these two techniques with a generalization of the fish-eye [15] view for higher dimensions.

Besides visual, also acoustic interfaces have been described in literature. Moreover, techniques that combine the two approaches have been proposed, such as [10] and [19]. Both of these interfaces try to take advantage of the *cocktail party effect*. That is, they play different pieces of music simultaneously. At the same time, they guide the user through the music space by means of visual aids. A purely acoustic exploration method has been proposed by Pampalk et al. [14]. Even though their algorithm has been described for audio-feature spaces, it can also be used in our environment, and thus serves as a benchmark for our experiments. Other approaches that go into a similar direction, but are geared at the large listening community of a web radio, are presented in [4] and [1].

III. Overview

Our work is based on a 10-dimensional Euclidean space of music, which was developed by Goussevskaia et al. [2] and is available through a web-service at *www.musicexplorer.org*. After first startup, our Android based application retrieves the coordinates of all the songs in the user's music collection.[3] After this step, the application has (offline) access to the high-dimensional representation of the entire collection. The visual as well as the acoustic interface proposed by this paper are designed to present this high-dimensional information in a user understandable manner.

Before introducing our interfaces, we highlight some aspects of the underlying space of music that are important for this work. For more detailed information, the reader is referred to [2].

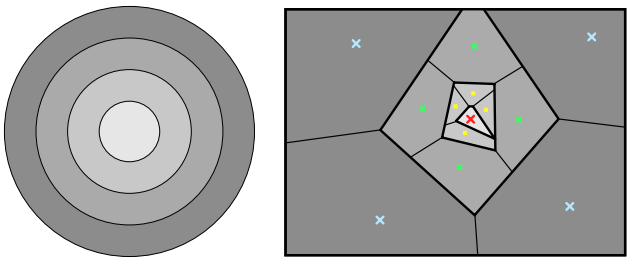
- *Similarity measure*: The underlying similarity measure is based on an analysis of *last.fm*[4] usage data. The actual similarity calculation closely resembles Amazon's item-to-item collaborative filtering idea [9].
- *Dimensionality*: Goussevskaia et al. have shown that an intuitively navigable 2 or 3-dimensional space is not able to adequately reflect music similarity. Thus, they propose a 10-dimensional space.
- *Orientation*: The space merely represents relative similarity of songs. Axes in this space are not assigned a special meaning, which makes it hard to guide the user through the space.
- *Coverage*: The space comprises approximately 430K tracks, which covers about 50%-60% of a typical music collection—enough to conduct meaningful experiments.

IV. Visual Browsing

The taste of a single user is typically restricted to few styles of music. Thus, a smart visual interface should group similar tracks and provide a facility to quickly guide the users to their favorite regions within the underlying space. More generally, we have identified the following requirements for a visual music exploration scheme:

- *Locality*: When browsing in a specific region of music, the local proximity is interesting and should thus be in the user's focus.
- *Globality*: Simultaneously to the *local* information, more distant tracks should be visualized, making it simple to cross over different regions in the collection.

[3]This process has to be repeated whenever the collection changes.
[4]www.last.fm



(a) Lens metaphor (b) Recursive partitioning of a 2D space

Figure 1. Lens view

- *Orientation*: Moving from one region to another should be transparent and predictable to the user.

For typical 2-dimensional settings there exist several well known concepts to satisfy these criteria, such as fish-eye views, or a combination of zooming with satellite views. As indicated earlier, however, the space of music cannot directly be mapped into a low-dimensional representation. Thus, the mapping into the display space has to be dependent on the current area of focus. We will next introduce the *lens* metaphor that fulfills the outlined requirements and is applicable to the small screen of mobile devices.

A. The Lens View

The *lens* metaphor is directly derived from the (2-dimensional) fish-eye concept. The idea is to show the most detailed view in the center of the screen and to blur out more and more details with increasing distance from the area of focus. The distance from the center is schematically illustrated by means of rings, which indicate the lens (see Figure 1(a)). A user can define the area of focus by selecting a center song. In the innermost circle, few, highly related songs are then displayed. The outer circles contain clusters of songs, rather than single songs. With increasing distance from the center these clusters grow bigger and represent more distant regions of the space. This idea is illustrated in Figure 3(a). The uniformly colored points in the innermost circle denote songs similar to the center song (star), whereas the colorful symbols in the outer circles represent clusters of songs.

To realize this lens view, i.e. increasing cluster sizes with increasing distance from the center, we apply a technique which we refer to as *recursive k-means*. The algorithm, as the name indicates, clusters the space in a recursive manner, as schematically depicted in Figure 1(b) for a two dimensional space.

First the entire (10-dimensional) space is clustered into a fixed number (k, 5 in our case) of clusters. For clustering, a modified version of the k-means algorithm is used, which fixes the centroid of one cluster to the center song. This results in $k-1$ outer and one center cluster. In the consecutive steps, the same procedure is applied to the songs residing in the center cluster. This process is repeated until either the center cluster contains less than q points, or until a maximum number of recursion levels is reached. The last condition prevents the generation of too many levels, which could not be visualized on small screens.

The initial cluster centroids are chosen at random. Hence, the space partition is not deterministic, and the result presented to the user varies each time the algorithm is run. This is not a flaw, but rather a desired behavior, as it allows to view the same area of focus in slightly different variations.[5]

Recursive k-means defines the content of the clusters to be used in the lens view. For visualization, the obtained positions (i.e. cluster centroids) in the high dimensional space have to be embedded into the 2-dimensional visualization plane. This mapping is done by a special spring embedder tailored to the needs of the lens view.

Similar to the hierarchical clustering, the visualization is also done in multiple steps. First, the songs belonging to the center cluster are embedded. Thereby the algorithm is restricted to only choose positions within the innermost circle. The remaining clusters are then successively embedded within the outer rings. Thereby each ring corresponds to one recursion level. That is, the position of clusters stemming from a given recursion level are restricted to lie within one particular ring. The embedding process is directed from the center towards the border. Already embedded points of the inner rings are kept fixed but contribute to the force that acts on the points that are being embedded.

To guarantee a consistent view, the initial positions of the points are set corresponding to the first two dimensions of the high-dimensional space. Thus the outermost clusters are always placed at similar positions (e.g. the area containing mostly Electronica is always placed in, say, the upper right corner). Observe that this strategy also reduces the number of iterations (as compared to random positions), as the initial positions are likely to be close to the final positions.

B. The Cake Metaphor

In [2] it was shown that the space exhibits a fairly good clustering in terms of genres. We have thus augmented the obtained cluster centroids with genre information, to make browsing more intuitive. The center of mass of positions of songs with known genre information (acquired from *www.allmusic.com*) defines a centroid for each genre. Measuring the distances to these genre centroids allows to define a genre relationship to each point in the 10-dimensional space.

This information is visualized using the *cake* metaphor, which consists of an inner circle and an outer ring. The outer ring is subdivided into segments—or *cake* slices—that represent the inverse squared distances from the different

[5]Observe that a deterministic behavior could easily be achieved by the use of random seeds.

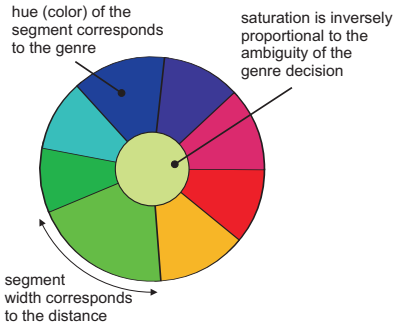


Figure 2. Cake metaphor

genre centroids. The inner circle represents the nearest cluster. The hue (i.e. color) of the inner circle corresponds to the hue of the nearest cluster, whereas its saturation indicates, how clear the given position could be assigned to the corresponding genre. A cluster centroid which has almost the same distance to all genre centroids, for example, is represented by a very pale color. An example of a cake diagram is depicted in Figure 2.

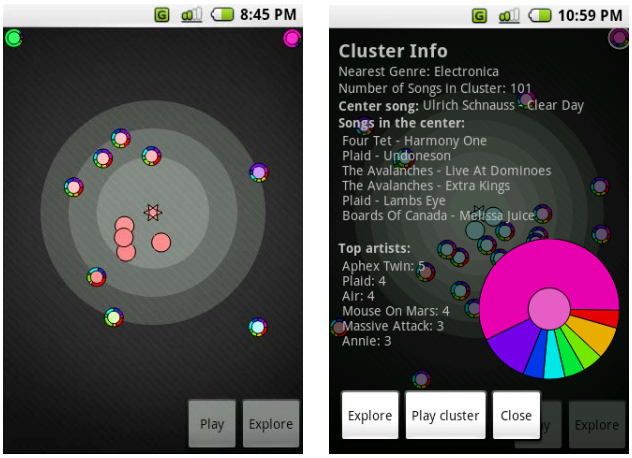
Observe that both, the lens as well as the cake metaphor are not restricted for use in music collections in conjunction with genres. Rather the lens metaphor can be used in conjunction with any high-dimensional space and, similarly, the cake metaphor applies to any meaningful anchor points, be it for music collections (where, e.g., moods could be used), or any other high-dimensional data spaces.

C. The Final Interface

The final interface combines the cake with the lens metaphor. The cake metaphor is used visualize the clusters in the outer rings. Each songs in the center area is represented by a simple circle, colored analogously to the inner circle of a cake diagram. The resulting interface is illustrated in Figure 3.

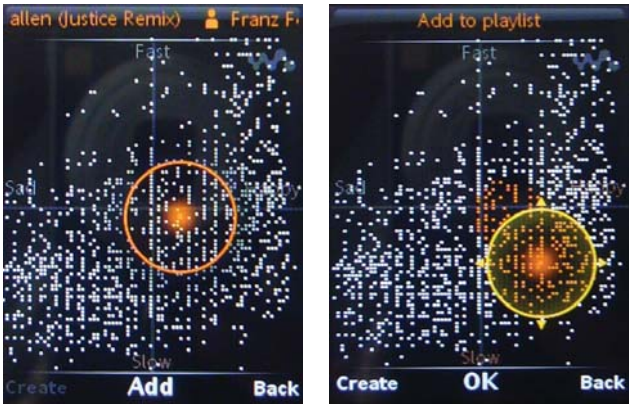
The different elements (songs and clusters) in this interface react to touch events. Selecting a single song in the inner circle allows to either play it, or to make it the new center song. Selecting a cluster opens a dialog (see Figure 3(b)) that presents the cluster's detailed content. Moreover, the dialog contains a button to play the contained songs, as well as a button that allows to move to the cluster's centroid (meaning the song closest to the cluster's centroid becomes the new center song).

The primary goal of exploring a music collection is to later listen to it. We have thus incorporated a playlist creation mechanism. To create a playlist a user can add each center song to a set of *seed* songs. Once enough seed songs are selected, a playlist consisting of the chosen seed songs as well as of songs randomly sampled from each seed song's neighborhood can be generated. A user settable parameter allows to control the size of these neighborhoods and thus the diversity of the resulting playlist.



(a) The lens metaphor with visualized clusters (b) Detail information for a selected cluster

Figure 3. Our graphical user interface for visual browsing.



(a) Browsing (b) Adding areas to a playlist

Figure 4. SensMe™: A commercial visual browsing interface shipped with Sony Ericsson devices.

D. Evaluation

For evaluation we have conducted a preliminary user study with 9 participants. The study mainly compares our interface with SensMe™. SensMe™was developed by Sony Ericsson and is shipped with their latest phone models. To the best of our knowledge it is currently the only commercial visual interface for music exploration on mobile devices.

SensMe™is based on audio-analysis which classifies the songs according to the two properties tempo (slow vs. fast) and mood (sad vs. happy). The songs are arranged on the screen along these two coordinate-axes, as illustrated in Figure 4. To create playlists, circular areas of adjustable size can be selected. The final playlist then contains the union of all the songs within the selected areas.

The main part of the user experiment consisted in the creation of a playlist (20 songs) with both interfaces. In addition, the participants had to fill in a questionnaire. For

both, SensMe™as well as our interface, the participants were given the same collection, which they were not familiar with. The collection contained approximately 1400 files (7.5GB) and covered a broad range of music. For each interface, the participants were given 5 minutes to create a playlist that matched their taste and mood as closely as possible. Afterward they had to listen through the playlists and rate each single song on a scale from 0 (worst) to 10 (best). The resulting average ratings were 6.3 (our interface) and 5.5 (SensMe™).

In addition, we prepared a small questionnaire the participants had to fill in. If not indicated otherwise, the answers were measured on a scale ranging from 1 (worst) to 5 (best). The major findings can be summarized as follows:

- *Playlist (overall)*: We were not only interested in the rating of individual songs, but also asked the participants to judge the overall impression of the obtained playlist. The result is in line with the song ratings (3.33 (this paper) vs. 2.44 (SensMe™)) and suggests that the presented interface is in fact better suited for playlist generation.
- *Diversity*: People were asked to judge the playlist diversity on a scale from 1 (too diverse) to 5 (not diverse enough), with 3 as a neutral value (just right). While our algorithm is not quite diverse enough (3.44), the playlists generated by SensMe™are too diverse (2.44). The result suggests that both algorithms perform similar in this respect with slight advantages for our algorithm.[6]
- *Usability*: We wanted to know whether the interface is intuitive to use. As expected, the simple 2-dimensional interface of SensMe™outperformed our exploration scheme for a high dimensional space. The ratings were 4.67 (SensMe™) versus 3.67 (our interface).
- *Underlying space*: We asked our participants whether, in their opinion, similar songs well group together in the respective system. In this question our approach (4.00) clearly outperforms SensMe™(2.44). A possible reason for this result is that the two dimensional representation of SensMe™is not capable to adequately reproduce the underlying similarity. It might thus indicate that it is worth to operate in higher dimensional spaces at the expense of a less intuitive interface. Another possible explanation, however, is that the user-behavior driven similarity measure underlying our interface provides better results than the audio-analysis based methods of SensMe™.
- *Overall*: We wanted to know whether our users would use the system again. Thereby, only the answers *yes* and *no* were allowed. Again, the result favors the interface presented in this paper (67% yes) over SensMe™(44% yes), which is in line with the playlist ratings discussed earlier.

Finally, we wanted to know how useful the cake metaphor is. The results show that the most valuable part is the center circle color (3.22). Cake slices (2.67) and the center circle saturation (2.55), however, have also been used for navigation.

V. Acoustic Exploration

As music is primarily perceived by the sense of hearing, acoustic navigation through the space of music also seems to be a natural approach. Similar as in the presented visual exploration scheme, the goal of our acoustic interface is to quickly guide users to music of their taste.

As opposed to the visual situation, it is impossible to acoustically provide an instant global impression of the space, as we are not able to listen to many tunes in parallel.[7] An acoustic exploration scheme thus has to rely mainly on local information. Moreover, people's notion of orientation or direction in acoustic space is much more fuzzy than in geometric space. As a consequence, the control of the exploration process should shift from the user towards the device.

The goal of our interface is to dynamically generate a sequence of songs (a playlist) that fits the user's taste. This is achieved by constantly appending new items to the sequence dependent on the user's (implicit or explicit) feedback about the preceding songs. Thereby, the algorithm learns the user's taste and, ideally, selects ever better items.

Our primary interface uses explicit user feedback defined on a continuous rating scale ranging from 0 to 1 (see Figure 5). For scenarios that do not allow for explicit feedback, we offer the possibility to switch to a binary scheme that assumes a song was disliked if it was skipped, and liked otherwise (analogously as proposed in [14]). A continuous rating scale, however, allows to more precisely estimate and react to the user's needs. Thus it should be applied whenever possible, such as when explicit feedback is applicable.

Before we come to our exploration algorithm in more detail, let us briefly review its major requirements:

- *Taste*: Clearly, the exploration algorithm should visit areas and songs the user likes.
- *Diversity*: People tend to get bored when listening to several very similar (e.g., same artist) songs in a row. The exploration scheme should thus also provide sufficient diversity.
- *Adaptability*: The algorithm should be able to adapt to the changing mood of a person. That is, if, after some time of listening to rock music, the listener starts feeling

[6]Observe that our algorithm provides a parameter to control diversity. We can thus expect that people that regularly use the system are able to produce better playlists, as they get a better feeling for the right parameter setting.

[7]As pointed out by Tzanetakis and Cook [19], 8 simultaneous tunes form an upper limit.

Figure 5. Acoustic exploration interface: Note the rating bar at the top.

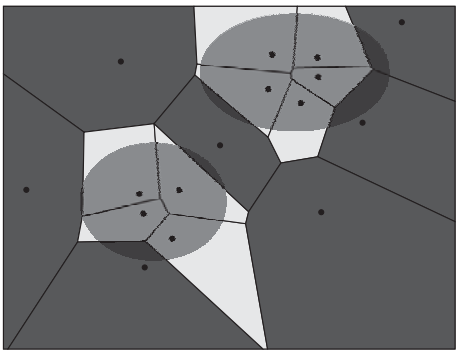


Figure 6. Acoustic browsing (simplified): The entire collection is decomposed by means of Voronoi tessellation. The generating points for the Voronoi cells are given by the rated songs. Dependent on the rating each cell is either considered *good* (light gray) or *bad* (dark gray). The shaded areas indicate the user's region of interest. Note that in the application a 10 dimensional, rather than a 2 dimensional space, as illustrated here, is used.

more like jazz, the algorithm should be able to quickly adapt to this changed condition.

- *Exploration*: As the name suggests, the exploration scheme should be able to *explore* a collection. That is, it should not just stick to songs the user listens to frequently, but should be able to find songs the user might even not have been aware of.

A. Acoustic Exploration Algorithm

The basic idea of our exploration scheme is that on completion of a song, the algorithm looks at the ratings acquired so far and then selects the next song in an intelligent manner. A naive algorithm might simply select the song most similar to the currently best rated song, thereby only considering songs that have not yet been played. This method, however, would most likely violate the *diversity* requirement. We thus follow another strategy, which is based on the Voronoi tesselation.

We start by outlining a simplified version of the algorithm, which works as follows: The previously rated songs are used as the input of a Voronoi tesselation, which is applied to the entire collection. As a result, each previously rated song becomes the *generating point* of one Voronoi cell. We assume that ratings above 0.5 (on a scale from 0 to 1) indicate that the user likes a song, whereas lower ratings are seen as a sign of dissatisfaction. As a consequence, each Voronoi cell (and thus also all the contained songs) can be seen as either *good* or *bad*, dependent on the rating of its *generating point*. The Voronoi tesselation is re-executed on each song completion. After tessellation the exploration algorithm selects a not yet played song from within a *good* Voronoi cell. This strategy is schematically illustrated in Figure 6. The lighter areas, which indicate *good* cells, roughly approximate the user's region of interest (shaded).

The entire process is started by selecting a song uniformly at random from the entire collection. If this song is rated *good*, everything continues as outlined before. A rating below 0.5, however, would lead to a single Voronoi cell that is considered *bad* and covers the entire collection (resulting in no playable songs). Whenever no positively rated song is available, we therefore add a *floating centroid* to the system, the position of which is determined by the modified version of the k-means algorithm described in the visual exploration part (keeping all previously rated positions fixed). This procedure is repeated until the first positively rated song is found. Afterwards, the floating centroid is removed.

Observe that this simple algorithm exhibits various weaknesses. First, it does not take advantage of a continuous rating scale. Moreover, it is not able to satisfy the *adaptability* requirement, since regions once marked *bad* will never be visited again. Finally, the *good* Voronoi cells only provide a rough approximation of the region of interest. In particular the border areas of these cells are likely to disagree with the user's taste. To overcome these shortcomings we improve the outlined scheme in several points:

- *Weighting*: Dependent on the rating of the corresponding generating point, a weight is assigned to each Voronoi cell. The more this rating deviates from 0.5 (in either positive or negative direction), the higher the weight. When selecting the next song to play, the algorithm considers these weights by selecting a song with a probability proportional to the weight of its enclosing Voronoi cell.
- *Aging*: Weights are not static. Rather, on each song completion, all weights are reduced by some constant value. If the weight of a generating point falls below a certain threshold t, the corresponding point is removed. Due to the aging technique, old ratings eventually become obsolete. Thus, the algorithm can well react to changes in the user's mood.
- *Centering*: Songs close to a positively rated generating point lie within the user's region of interest with higher probability than songs at the border of the correspond-

ing Voronoi cell. Thus, such songs are selected with higher probability by the algorithm. For simplicity, we currently use a fixed probability distribution. However, making this distribution adjustable by means of a parameter might allow the user to control the sequence's diversity.

- *Escaping*: At some point, the algorithm might start playing songs from within a certain narrow area only, as preliminary experiments have shown. The missing diversity is then likely to become manifest in form of repeated low ratings. We thus allow the algorithm to break out of such gridlocked situations by selecting a completely random song with a certain probability. This probability is adaptive within a given upper and lower bound. It decreases with each positive, and increases with each negative rating.

These tweaks have shown to improve the quality of the resulting sequences. However, our first experiments have also revealed some performance issues. After all, a user is not willing to wait for several seconds until the next song is being played. As we only want to select a single song in the end, it is obviously an unnecessary overhead to classify each song as either *good* or *bad* in each round (i.e. after each song completion). Instead, we move part of the random selection process to the beginning of a round by first selecting a random sample of the not yet played songs.

B. Evaluation

For evaluation we have compared the algorithm to two alternative approaches:

- *Shuffling*: Most music players offer the possibility to listen to a collection in a purely random fashion. Moreover, 7 of our 9 participants said that they at least sometimes do listen to music completely randomly.
- *Pampalk*: Pampalk et al. [14] have proposed an algorithm that dynamically creates playlists based on the user's skipping behavior. Although the method was originally proposed for use in conjunction with an audio-feature space, it can be applied to our space, too. The algorithm works as follows: Each song for which the nearest *good* song is closer than the nearest *bad* song is added to a set S. If S is non-empty, its element with smallest distance to the nearest *good* song is selected. Otherwise, the song with the lowest d_g/d_b ratio is played, where d_g denotes the distance to the nearest *good*, and d_b the distance to the nearest *bad* song. The algorithm does not make use of rated feedback, but uses a binary scheme solely based on skipping behavior.

For comparison, all participants had to create a sequence of 20 songs with each of the three methods. In all cases, the start song was chosen randomly. Moreover, we let the users uninformed about the internals of the different algorithms to avoid biased results. They were, however, told that the algorithms were supposed to find their taste (which is not exactly true for random shuffling). We used the interface with explicit song ratings (recall Figure 5) for all experiments. The acquired ratings were not only used to feed the algorithms, but also to later assess the qualities of the different playlists.

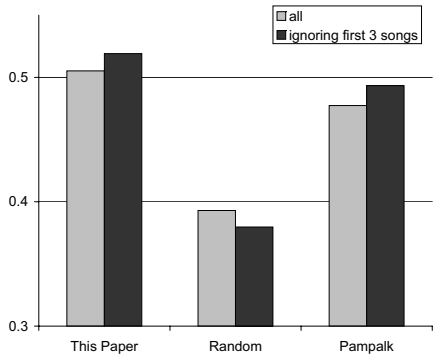


Figure 7. Comparison of the three different acoustic exploration methods. The light bars indicate the average rating over all songs. To demonstrate the learning effect, the dark bars ignore the ratings of the first 3 songs in each sequence.

The results are summarized in Figure 7. The figure depicts the overall average song rating for each algorithm (light bars), as well as the average rating after ignoring the first three songs of each sequence (dark bars). For both algorithms that are supposed to adapt to the users taste, the dark bars are higher, indicating that the algorithm in fact improved over time. In the random case, by contrast, the rating got worse, which might be explained by the unsatisfied user expectation. The figure further reveals that the adaptive algorithms by far outperform random shuffling. Moreover, we can see that only the algorithm presented throughout this paper reaches a positive average rating (i.e., above 0.5).

As indicated earlier, we were not only interested in the overall rating of the resulting sequences, but also in the algorithm's capability to explore a collection. We have assessed this property by means of participant questioning as well as path visualization.

Using a questionnaire, we had the participants judging their satisfaction concerning the sequence's diversity on a scale from 1 to 5 (1: too diverse, 3: just right, 5: not diverse enough). The results are summarized in Figure 8. As expected, the diversity of the random shuffling algorithm was considered too high (average: 2.56, min: 1, max: 4). The diversity of the algorithm of Pampalk et al., on the other hand, was considered to low by all participants (average: 4.44, min: 4, max: 5). The best score (i.e. closest to 3) was reached by our algorithm (average: 2.67, min: 1, max: 5), which scores slightly better than random shuffling, and clearly outperforms the approach of Pampalk et al.. The fact

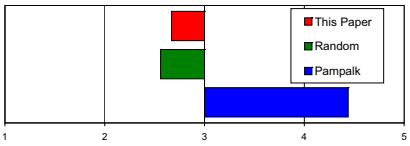


Figure 8. Diversity: Average values for the diversity ratings of the participants (1: too diverse, 3: just right, 5: not diverse enough).

that for one single algorithm the entire rating scale has been used suggests that diversity is a very subjective measure.

It is important to note that Pampalk's algorithm possibly provides more diversity if applied to audio-feature spaces, for which the algorithm was originally designed. Due to the strong dependence of diversity on the user, and, possibly, the underlying space, we believe that an algorithm should be adaptive in this respect (either by means of a user settable parameter, or by means of self-regulation). As mentioned before, we could realize this idea by incorporating a parameter that controls the *centering* effect.

VI. Conclusion

We have presented a visual as well as an acoustic exploration scheme, to overcome the problems of music organization on mobile devices in the context of ever growing collections. In our experiments, both interfaces outperformed state-of-the-art alternatives in the corresponding field. To deal with the high-dimensional properties of the underlying space, we have introduced the lens metaphor for visualization, which allows to focus at the certain area without losing the global overview. We have further proposed to augment the lens view with cake diagrams that support the orientation by means of semantic information. A radically different approach is followed by the acoustic interface. By decomposing the high-dimensional music similarity space dependent on the given feedback, it is able to identify regions of interest of a given user, can thus constantly improve its estimate of the user's taste, and finally play ever better songs. The successful integration into a prototype application in conjunction with the conducted user experiments demonstrates the practical applicability of the proposed solutions. Our interfaces show how the attention data gathered by social web platforms can be used to enhance the music experience of end users.

References

[1] C. Baccigalupo and E. Plaza. A case-based song scheduler for group customised radio. In *ICCBR*, 2007.

[2] O. Goussevskaia, M. Kuhn, M. Lorenzi, and R. Wattenhofer. From Web to Map: Exploring the World of Music. In *Web Intelligence*, 2008.

[3] O. Goussevskaia, M. Kuhn, and R. Wattenhofer. Exploring music collections on mobile devices. In *Mobile HCI*, 2008.

[4] D. B. Hauver and J. C. French. Flycasting: On the fly broadcasting. In *DELOS Workshop: Personalisation and Recommender Systems in Digital Libraries*, 2001.

[5] O. Hilliges, P. Holzer, R. Klüber, and A. Butz. Audioradar: A metaphorical visualization for the navigation of large music collections. In *Smart Graphics*, 2006.

[6] A. Hinneburg, D. A. Keim, and M. Wawryniuk. Hd-eye - visual clustering of high dimensional data. In *ICDE*, 2003.

[7] P. Knees, M. Schedl, T. Pohle, and G. Widmer. An innovative three-dimensional user interface for exploring music collections enriched with meta-information from the web. In *ACM Multimedia*, 2006.

[8] S. Leitich and M. Topf. Globe of music - music library visualization using geosom. In *ISMIR*, 2007.

[9] G. Linden, B. Smith, and J. York. Industry report: Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Distributed Systems Online*, 4(1), 2003.

[10] D. Lübbers. Sonixplorer: Combining visualization and auralization for content-based exploration of music collections. In *ISMIR*, 2005.

[11] F. Mörchen, A. Ultsch, M. Nöcker, and C. Stamm. Databionic visualization of music collections according to perceptual distance. In *ISMIR*, 2005.

[12] R. Neumayer, M. Dittenbach, and A. Rauber. Playsom and pocketsomplayer, alternative interfaces to large music collections. In *ISMIR*, 2005.

[13] E. Pampalk and M. Goto. Musicrainbow: A new user interface to discover artists using audio-based similarity and web-based labeling. In *ISMIR*, 2006.

[14] E. Pampalk, T. Pohle, and G. Widmer. Dynamic playlist generation based on skipping behavior. In *ISMIR*, 2005.

[15] M. Sarkar and M. H. Brown. Graphical fisheye views. *Commun. ACM*, 37(12):73–83, 1994.

[16] J. Seo and B. Shneiderman. A rank-by-feature framework for interactive exploration of multidimensional data. *Information Visualization*, 4(2), 2005.

[17] B. Shneiderman. Extreme visualization: squeezing a billion records into a million pixels. In *SIGMOD Conference*, 2008.

[18] M. Torrens, P. Hertzog, and J. L. Arcos. Visualizing and exploring personal music libraries. In *ISMIR*, 2004.

[19] G. Tzanetakis. Marsyas3d: a prototype audio browser-editor using a large scale immersive visual and audio display. In *ICAD*, 2001.

[20] F. Vignoli, R. van Gulik, and H. van de Wetering. Mapping music in the palm of your hand, explore and discover your collection. In *ISMIR*, 2004.

[21] J. Yang, M. O. Ward, E. A. Rundensteiner, and S. Huang. Visual hierarchical dimension reduction for exploration of high dimensional datasets. In *VisSym*, 2003.