STREAMCATCHER: INTEGRATED VISUALIZATION OF MUSIC CLIPS AND ONLINE AUDIO STREAMS

Martin Gasser, Arthur Flexer
Austrian Research Institute
for Artificial Intelligence (OFAI)
Freyung 6/6
A-1010 Vienna, Austria

Gerhard Widmer
Department of Computational Perception
Johannes Kepler University
Linz, Austria

ABSTRACT

We propose a content-based approach to explorative visualization of online audio streams (e.g., web radio streams). The visualization space is defined by prototypical instances of musical concepts taken from personal music collections. Our system shows the relation of prototypes to each other and generates an animated visualization that places representations of audio streams in the vicinity of their most similar prototypes. Both computation of music similarity and visualization are formulated for online real time performance. A software implementation of these ideas is presented and evaluated.

1 INTRODUCTION

A massive amount of music is already available on the web in the form of internet radio streams. At the time of writing this paper, the "radio-locator" website [1] lists more than 2500 entries in its directory of free internet radio stations, a number that is likely to increase rapidly in the future. Because it is infeasible for users to keep an overview of available radio streams, software that recommends possibly interesting radio streams would be desirable.

When talking about music, humans usually use examples of similar music to describe new and unknown songs or artists. For example, one might characterize *Nirvana* as a combination of the *Pixies'* and *Sonic Youth's* guitar sound, the *Beatles'* sense for catchy melody lines, and the heritage of American folk-rock music in the style of *Neil Young*. So, instead of resorting to predefined taxonomies for music categorization (e.g., by *Genre*), an alternative is to present a "query by example" based user interface directly to the user.

To support the user in finding new music he or she might like in audio streams, we propose a simple interactive visualization approach that incorporates the user's musical vocabulary into the definition of semantic spaces for music similarity judgement. The user defines her own semantic space by supplying, coloring and labeling an arbitrary number of reference songs from her own music collection. In this way, she can define her personal musical concepts via examples she is familiar with.

In our application, songs from personal music collections are imported in an incremental, user-feedback based manner: (1) A new sound clip (concept prototype) is loaded into the system and it is compared to already loaded clips with a content-based music similarity measure, (2) the software shows the user what it thinks the new music is similar to and puts a preliminary label on it, (3) the user refines the software suggestion by correcting the label, (4) go back to (1). When the user decides that the concepts he or she is interested in are sufficiently well represented, unknown music from internet radio streams can be compared to the sound prototypes (using the same similarity metric), and a visualization is generated by means of a multidimensional scaling [4] technique. The visualization also provides direct interaction facilities that allow the playback of streams and clips to interactively explore the audio streams audio-visually.

Additionally to mapping acoustic similarity to proximity information in the 2D visualization, color is used to encode the user's view of what the music sounds like. Colors can be assigned to songs upon loading them into the system by the user. This idea is motivated by the neurologically based phenomenon of *synesthesia* [1, 14], in which stimulation of one sense automatically triggers experiences in another sense.

2 RELATED WORK

Berenzweig et al. [3] construct an *Anchor Space* by training N classifiers to prototypical instances of concepts, classifying unknown music, and mapping posterior probabilities to the individual dimensions of vectors in an N-dimensional space. Distances in this space have been shown to better reflect the user's notion of music similarity.

Lidy and Rauber [7] calculated features derived from long-term observations of (terrestrial) radio stations and used a SOM to generate a 2D visualization of the types of music

[1] http://www.radio-locator.com/

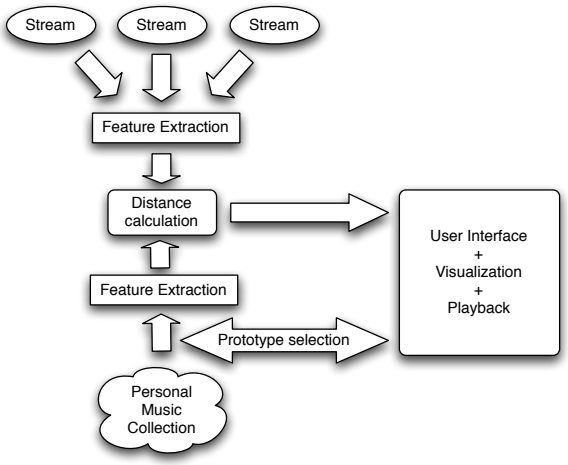

Figure 1. System components

played in different radio stations. However, they do not perform online analysis of radio streams.

Tzanetakis [13] extracted features from audio streams in real time and displayed genre membership of the streams in a real time visualization. By reducing dimensionality of the feature space with a Principal Component Analysis, he also mapped audio signals from high-dimensional timbre-similarity space into a three-dimensional visualization space. To the authors' knowledge, this was the first published implementation of online/real time audio signal classification and similarity analysis.

Lamere [6] uses a technique based on acoustic similarity and multidimensional scaling to visualize and explore music collections in 3D space. This work is related to our work in that it also shows the disparity between acoustic and high-level similarity.

Lübbers [9] derives prototypical songs by hierarchically clustering music collections, and proposes a multi-modal user interface for interactively exploring the cluster centers' neighborhood.

Recently, some work in the field of Human-Computer Interaction has been published (see also [1, 14]) that gives rise to the presumption that organizing music by color is quite intuitive. We also used this idea in our application to identify music that belongs to a common abstract concept, which could be a genre, a particular kind of instrumentation, a certain guitar sound, and so on.

3 SYSTEM OVERVIEW

To evaluate our approach, we implemented an application prototype that performs online (a) feature extraction, (b) similarity calculation, and (c) visualization. Figure 1 sketches the main components of the application.

The feature extraction subsystems are responsible for extracting feature frames from offline clips and online streams of audio data. The distribution of these feature frames is then modeled as a single Gaussian with full covariance [10] per clip/stream.

Central to our framework is the similarity calculation component that calculates distance matrices holding clip-to-clip and stream-to-clip similarities by calculating the symmetric Kullback-Leibler (KL) divergence [5] between each pair of Gaussians.

Distances between clips are then projected to a 2D visualization space with a Multidimensional Scaling [4, 2] algorithm, whereas streams are linearly embedded into a local subspace spanned by the 3 nearest neighbors of a stream model in feature space.

3.1 Implementation notes

The application was implemented in C++ using the Open-Source QT [2] toolkit , and we use OpenGL [3] for the visualizations. All signal processing and statistical computing was implemented within the *FLOWer* framework, a portable and efficient C++ library for data flow-oriented media processing, which is being developed by the first author of this paper.

4 FEATURE EXTRACTION & SIMILARITY CALCULATION

In order to be able to extract features from offline files as well as from live streams, the system contains two feature extraction pipelines. In both pipelines, Mel Frequency Cepstral Coefficients (MFCCs) [8] are computed for each frame of audio data. The main difference between online and offline analysis is the way statistics (means and covariances) are calculated. While in the offline case, the distribution of MFCC's in an entire audio file is estimated, the online scenario requires more diligence; since the distribution of features in an internet radio stream is not likely to stay constant, we only take the most recent feature frames into account. For performance reasons, we decided to use a recursive estimator for means and covariances instead of a sliding window approach.

The input to the feature extraction stage is either an MP3 file or an MP3 stream (which can be transmitted over an HTTP connection). This data is decoded to PCM at a sample rate of 44.1kHz, converted to mono, and sliced into frames of size 2048 samples with 50% overlap. Then, the frames are multiplied with a Hamming window, the magnitude spectrum and MFCCs are calculated, and a statistical model is derived from the data.

[2] http://www.trolltech.com/products/qt
[3] http://www.opengl.org

4.1 Offline processing

In the offline case, the calculation the mean vector μ and the covariance matrix Σ in the stream processing framework is straighforward. Let $\mathbf{X}$ be a vector of random variables, and $\mathbf{x}_i$ a concrete instantiation (sample) of $\mathbf{X}$ (in our case an MFCC vector from frame i).
Since

$$\Sigma = E(\mathbf{X}\mathbf{X}^\top) - \mu\mu^\top \tag{1}$$

and

$$\mu = \frac{1}{n}\Sigma\mathbf{x_i} \tag{2}$$

$$E(\mathbf{X}\mathbf{X}^\top) = \frac{1}{n}\Sigma\mathbf{x_i}\mathbf{x_i}^\top \tag{3}$$

all that is needed is one accumulation vector and one accumulation matrix for the sums and the sums of products of observations, respectively.

4.2 Online processing

Since in an online scenario the complete sequence of feature frames is not available beforehand, we use a recursive estimate of mean and covariance to parameterize the distribution of timbral features (we preferred a recursive estimator to windowed statistics for performance reasons).
The recursive estimate of the mean vector μ is calculated as:

$$\mu_n = (1-\alpha)\mu_{n-1} + \alpha\mathbf{x_n} \tag{4}$$

The covariance matrix Σ of a multidimensional vector of random variables $\mathbf{X}$ can be written as

$$\Sigma = E(\mathbf{X}\mathbf{X}^\top) - \mu\mu^\top \tag{5}$$

$E(\mathbf{X}\mathbf{X}^\top)$ can be estimated recursively as

$$E(\mathbf{X}\mathbf{X}^\top) = R_n = (1-\alpha)R_{n-1} + \alpha\mathbf{x}\mathbf{x}^\top \tag{6}$$

and

$$\Sigma_n = R_n - \mu\mu^\top \tag{7}$$

which can be refactored to

$$\begin{aligned}\Sigma_n &= (1-\alpha)R_{n-1} + \alpha\mathbf{x}\mathbf{x}^\top - \mu\mu^\top \\ &= (1-\alpha)[\underbrace{R_{n-1} - \mu_{n-1}\mu_{n-1}^\top}_{\Sigma_{n-1}} + \\ &\ldots\quad \alpha\underbrace{(\mathbf{x}_n\mathbf{x}_n^\top - \mu_{n-1}\mu_{n-1}^\top - \mathbf{x}_n\mu_{n-1}^\top + \mathbf{x}_n^\top\mu_{n-1})}_{(\mathbf{x}_n-\mu_{n-1})(\mathbf{x}_n-\mu_{n-1})^\top}]\end{aligned}$$

Thus,

$$\Sigma_n = (1-\alpha)[\Sigma_{n-1} + \alpha(\mathbf{x}_n - \mu_{n-1})(\mathbf{x}_n - \mu_{n-1})^\top] \tag{8}$$

Small values of the parameter α assign small weights to present values, thus the sequence of estimates is smoother with smaller α. In experiments, we found $\alpha = 0.001$ to be a reasonable setting.

For the implementation of the recursive estimation algorithms only one vector and one matrix accumulator are necessary. This is a clear advantage over windowed estimation approaches, where all measurements that fall inside the window must be memorized.

4.3 Distance calculation

Clip-to-clip distance is calculated upon loading of a new clip by storing the Kullback-Leibler (KL) divergences between each clip pair to a distance matrix. Clip-to-stream distances must be recalculated continuously because the distribution of stream features can change at any time.

4.4 Evaluation of the incremental similarity measure

In figure 2, we have plotted the similarities of a set of music clips to a music stream. We used the 4 reference clips listed in table 1.

The 4 plots correspond to the 4 clips, each plot shows the similarity of the current stream content to the reference clip (x axis is in audio frame numbers, 1 frame corresponds to the FFT hop size of 1024 samples). We calculate the similarity of the stream at time t to the clip C_i as the reciprocal value of the KL divergence between the stream model and the clip model, and we normalize the sum of clip-to-stream similarities over all clips at time t to one.

Artist/Composer	Title
Absolute Beginner	Rock On
Metallica	The Four Horsemen
Al Green	Listen
Heitor Villa-Lobos	Etude n. 1 en mi mineur

Table 1. Reference clips used for the evaluation

Artist/Composer	Title	Frames
A Tribe Called Quest	Buggin Out	0–9388
Megadeth	This was my Life	9389–18776
Al Green	Listen	18777–25150
Heitor Villa-Lobos	Etude n.2 en la majeur	25151–29715
Male	Speech	29716–34539

Table 2. Stream used for the evaluation

The similarity measure successfully identifies music from the same genre, style, or artist ("A Tribe Called Quest" have the highest similarity with the "Absolute Beginners" clip, "Megadeth" are most similar to "Metallica", and the clip-to-stream self-similarity for Al Green's "Listen" stays at ~ 0.8 almost over the entire duration of the song). However, the system tends to confuse Rap vocals with pure speech although it should emit low similarity scores with respect to all reference clips (last part of the stream versus the "Absolute Beginner" song) - this observation seems to be related to the fact that simple timbral similarity based on MFCC statistics wrongly classifies "Rap" and "Speech" as similar. Another shortcoming can be observed using the Absolute Beginner's and Al Green's songs as examples - both yield high similarity scores in the beginning of the Al Green tune, which seems to be related to the use of strong percussion in both pieces.

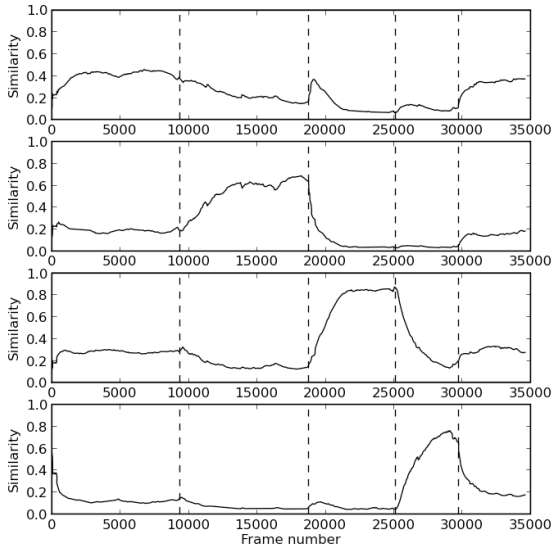

Figure 2. Online stream similarity to offline reference clips

5 GRAPHICAL USER INTERFACE

Figure 4 shows a screenshot of the running *StreamCatcher* application. The GUI of the application currently consists of a visualization that allows direct manipulation (zooming, panning, rotating, playback of clips and streams) of the visualized objects via a pointing device like a mouse or a touch screen.

By pressing the "Load Clip" button, a new sound clip is loaded into the application. After analyzing and k-NN-classifying the clip, a label (consisting of a textual description and a color) is suggested by the software in a dialog window. Now the user has the possibility to accept the suggestion, to change the label to another – already existing

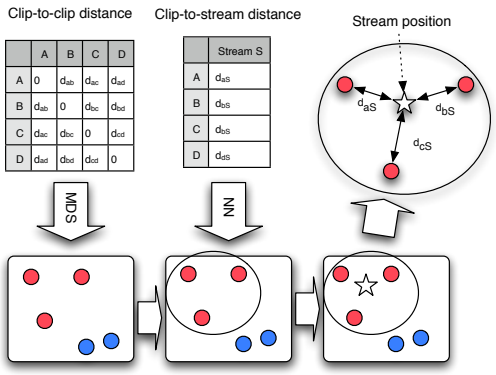

Figure 3. Placement of online streams in visualization space

– label or create a new label. Furthermore, it is always possible to alter the labeling of the clips later on. If the "Open Stream" button is pressed, the user can load an MP3 stream, which is subsequently visualized with the algorithm described below.

5.1 Visualization algorithm

It is well known that the symmetric KL divergence between Gaussians does not satisfy the requirements to a proper distance measure [12]. Therefore, we use a visualization technique that seeks to preserve the KL divergence values in a low-dimensional visualization space and relaxes this demand if that is not possible. A classic distance-preserving data mining/visualization technique is Multidimensional Scaling [4], which aims at placing data points in low dimensional visualization space while approximating the distances in feature space as closely as possible. Multidimensional scaling can be implemented as a spring model [2, 11], a physically inspired model of spring-connected nodes aiming at finding a node placement that minimizes the cumulative deflection from the springs' resting states. By mapping distances in feature space to spring lengths, the spring model solves the MDS problem approximately.

We chose to use a spring model variant of MDS because of its ease of implementation and because of the attractive visualizations that can be generated by constantly drawing the gradual relaxation of the system. By using a gradient-descent based solution algorithm (see algorithm 1), a placement of the nodes that minimizes the overall stress (the deflection from the springs' resting state) is constructed.

While the overall stress in the model (S_{cum}) is larger than a threshold (S_{thresh}), the algorithm loops over all nodes and determines for each node a force acting upon the node by calculating a weighted sum of unit vectors pointing from the current node to all other nodes. The weights can be positive or negative, depending on the difference between high- and low-dimensional distances. Then, a velocity vector is

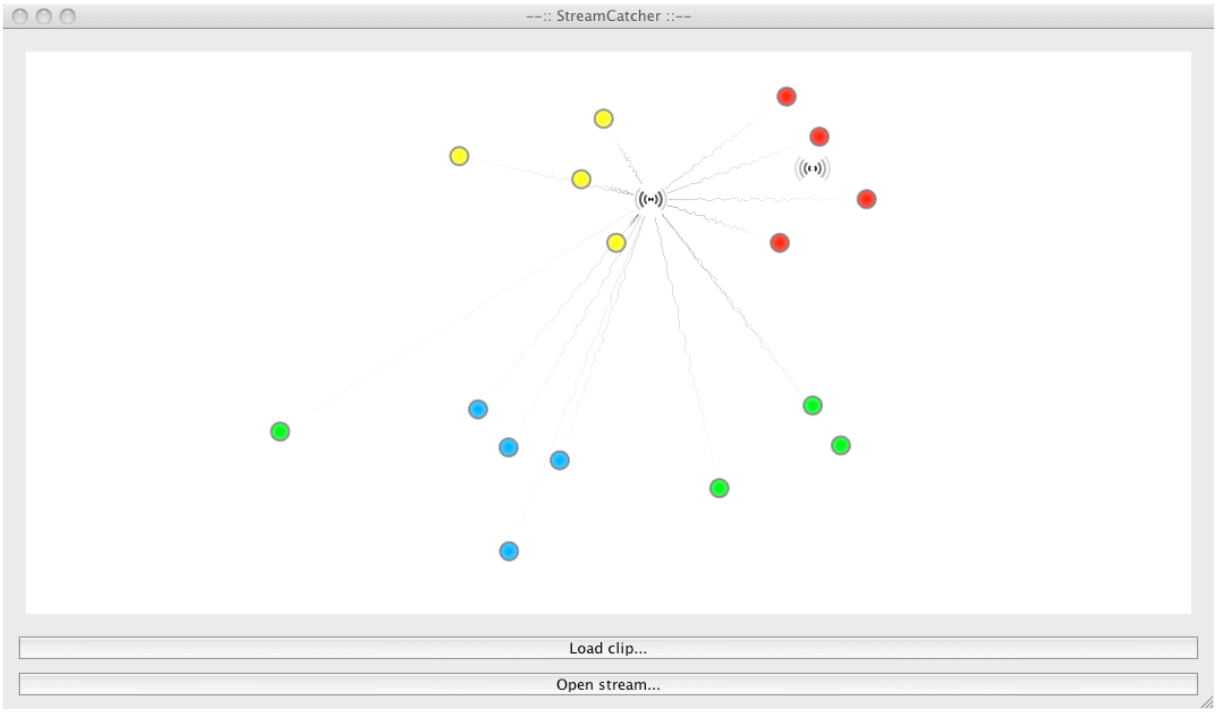

Figure 4. Screenshot showing the running application

Algorithm 1 Spring model layout algorithm

repeat
  $S_{cum} = 0$
  for $C_a \in \mathcal{C}$ **do**
    $\overrightarrow{f} = \overrightarrow{0}$
    for $C_b \in \mathbf{C}$ **do**
      $\overrightarrow{u} \leftarrow unit(p(C_a) - p(C_b))$
      $S \leftarrow d_{high}(C_a, C_b) - d_{low}(C_a, C_b)$
      $\overrightarrow{f} \leftarrow \overrightarrow{f} + \overrightarrow{u} * S$
      $S_{cum} \leftarrow S_{cum} + |S|$
    end for
    $v(C_A) \leftarrow v(C_A) + \overrightarrow{f}$
    $p(C_a) \leftarrow p(C_a) + v(C_A)$
    $v(C_A) \leftarrow v(C_A) * dampingFactor$
  end for
until $S_{cum} < S_{thresh}$

S_{cum} : Cumulative stress in the model

$\mathcal{C}$: Set of clip nodes

$unit$: Function returning a unit vector

p : Vector-valued function returning the 2D position of a clip node

d_{high} : Distance of clips in feature space

d_{low} : Distance of clips nodes in visualization space

v : Vector-valued function returning the current velocity of a clip node

$\overrightarrow{f}$: Force acting upon a node

updated for the current node by adding the force vector to its current value. Finally, the node's position is updated by moving it into the direction of its velocity, and the node's velocity is multiplied with a damping factor.

We use spring model MDS to place the static clips in a 2D visualization. Upon loading of a new clip, an MDS procedure is executed until the cumulative stress goes below a threshold value. The placement of the dynamic streams is determined by calculating the distances to the 3 nearest neighbors in the feature space and using the reciprocal value of those distances to calculate a convex combination of the positions of the nearest neighbors that have been placed by the MDS algorithm (see figure 3). Additionally, the true distances to the anchor clips are visualized by drawing the audio waveform of the stream's audio signal between the stream's position and the clips' positions, and by modulating the opacity of this waveform with the feature space distance. The smaller the feature space distance, the more opaque is the drawing of the waveform.

In the screenshot (figure 4), example clips for four different styles/genres of music are loaded (Heavy Metal: Red, HipHop: Yellow, Jazz Guitar: Green, Classical Guitar: Blue). The green outlier, which is closer to the Classic Guitar cluster than to Jazz Guitar, is a solo guitar piece by Pat Metheny, which makes the placement quite reasonable. The two streams (the loudspeaker-like symbols emanating the waveforms) play Heavy Rock music and Soul/Funk. The Rock stream is placed near the Heavy Metal-cluster, whereas the Soul stream

is placed close to the HipHop clusters, which characterizes the music quite well.

6 CONCLUSIONS & FUTURE WORK

We have presented an application prototype for similarity analysis and visualization of audio files and online audio streams in a common framework. Music clips from personal music collections are used as prototypical instances of musical concepts that define a visualization space in which online streams are embedded. For the online calculation of music similarity from streams we derived a similarity measure which is based on incrementally updated statistical models. The system comprises a user interface that supports the user in identifying prototypical examples of high level musical concepts. Acoustic similarity is mapped to proximity data in a 2D visualization, which in turn is derived from a high dimensional timbre similarity space by means of multidimensional scaling. Sound prototypes can be labeled with user-definable colors as well as textual labels.

The system successfully identifies streams that sound similar to prototypical clips. Thus, it can be used to get a rough overview of audio stream content, e.g., of internet radio stations. By exploring the suggestions of the software, the user may find music he or she likes faster than by just blindly trying radio stations.

We are currently working on improving the underlying distance measure and on special cases like speech-music discrimination, which could be a very useful feature for this application.

7 ACKNOWLEDGEMENTS

This research is supported by the Austrian Research Promotion Agency (FFG) under project number 815474 B1, and by the Austrian Science Fund (FWF) under project number L511-N15.

8 REFERENCES

[1] S. Baron-Cohen and J. Harrison. *Synaesthesia: Classic and Contemporary Readings*. Oxford: Blackwell Publishers, 1997.

[2] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, 1999.

[3] A. Berenzweig, D. P. W. Ellis, and S. Lawrence. Anchor space for classification and similarity measurement of music. In *ICME '03: Proceedings of the 2003 International Conference on Multimedia and Expo*, pages 29–32, Washington, DC, USA, 2003. IEEE Computer Society.

[4] M.F. Cox and M.A.A Cox. *Multidimensional Scaling*. Chapman and Hall, 2001.

[5] S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

[6] Paul Lamere and Douglas Eck. Using 3d visualizations to explore and discover music. In *Proceedings of the 8th International Conference on Music Information Retrieval*, Vienna, Austria, September 2007.

[7] Thomas Lidy and Andreas Rauber. Visually profiling radio stations. In *Proceedings of the 7th International Conference on Music Information Retrieval*, 2006.

[8] Beth Logan. Mel frequency cepstral coefficients for music modeling. In *Proceedings of the 1st International Conference on Music Information Retrieval*, Plymouth, Massachusetts, 2000.

[9] Dominik Lübbers. Sonixplorer: Combining visualization and auralization for content-based exploration of music collections. In *Proc. of the 6th International Conference on Music Information Retrieval, London, United Kingdom*, 2005.

[10] Michael Mandel and Dan Ellis. Song-level features and support vector machines for music classification. In *Proceedings of the 6th International Conference on Music Information Retrieval*, London, United Kingdom, 2005.

[11] Alistair Morrison, Greg Ross, and Matthew Chalmers. Fast multidimensional scaling through sampling, springs and interpolation. *Information Visualization*, 2(1):68–77, 2003.

[12] Elias Pampalk. *Computational Models of Music Similarity and their Application in Music Information Retrieval*. PhD thesis, Vienna University of Technology, Vienna, Austria, March 2006.

[13] George Tzanetakis and Perry Cook. Marsyas3d: A prototype audio browser-editor. In *Proceedings of the 7th International Conference on Auditory Display (ICAD)*, Helsinki, Finland, 2001.

[14] M. Voong and R. Beale. Music organisation using colour synesthesia. In *CHI '07 Extended Abstracts on Human Factors in Computing Systems*, San Jose, CA, USA, April 28 - May 03 2007.