Indian Institute of Technology, Guwahati

BTP REPORT: PHASE 2

Automatic Music Organization Systems

Abhinav Singh & Ashna Dhanda
Supervisor: Dr. S.R.M. Prasanna

Bachelor of Technology Program,
Electronics and Communication Department,
Indian Institute of Technology,
Guwahati, Assam, India

Abstract

The availability of large music collections calls for ways to efficiently access and explore them. We present an approach which combines descriptors derived from audio analysis with meta-information to create different views of a collection. Such views can have a focus on timbre, rhythm, artist, style or other aspects of music. For each view the pieces of music are organized on a map in such a way that similar pieces are located close to each other. The maps are visualized using an Islands of Music metaphor where islands represent groups of similar pieces.

We demonstrate our approach on a small collection using a meta-information-based view and two views generated from audio analysis, namely, beat periodicity as an aspect of rhythm and spectral information as an aspect of timbre.

Chapter 1
Introduction

Music is magic. It influences our emotions. It has the power to make us happy or sad, just as it can make us relaxed or aggressive. Often it is associated with some very special moments in our lives. Moreover, music is an important part of our cultures and identities. However, the most fascinating aspect of music might be the fact that the annual turnover for record sales only within the US has a magnitude of several billion USD.

This huge industry would not exist without its customers, who are always looking for something new to listen to. There are many ways in which customers find their desired products. For example, one way is to listen around. Customers might listen to what is being played on the radio or to what friends are listening to. However, this type of search is restricted to the subjective taste of others. Furthermore, it might take a while until a new release reaches ones ears.

Another approach is to follow the development of artists, who have been appreciated in the past, assuming that their work will also be appreciated in the future. However, this kind of search does not include unknown artists or newcomers. Customers might also follow the development of a genre like Jazz, Hip Hop, Classic or Funk. Relying on the classification skills of other people it is possible to search music stores for new releases in certain genres. However, classifying music into a limited number of genres is not an easy task. Lots of music is located somewhere in between many genres.

The tool presented in this report is meant to help customers find music without limiting the search to a specific genre or artist. This tool is based on a metaphor of geographic maps. Genres of music are represented by islands and continents. Similar genres are located close together and might even be connected through some land passage. On these islands there might be some further sub-genres that are represented as mountains and hills. Again these sub-genres might be more or less similar to each other and are arranged accordingly.

The mountains and hills on the map are labeled with words that describe certain attributes of the associated genre, for example the type of rhythm is described rather than using words like Pop, Jazz or Classical. The pieces of music in the collection are placed on the map according to their genre or sub-genre. Most of the music will be located around the mountains. However the few located in the valleys between typical genres might be the most interesting ones. The user can listen to the music by clicking on its representation on the map and can explore island after island according to his or her musical taste. Furthermore, music known to the user can be used as landmarks, to identify interesting regions on the map.

The maps with the islands of music could easily be placed on a web page of an internet store. Or they could be used in any conventional music store. Simple earphones and a touch screen monitor connected to a server would be sufficient. These maps could also be applied to digital libraries containing music, or simply at home to organize the private music collection. They could reveal some interesting properties of the inherent structure of the music collection that might not have been obvious before.
The technical requirements to develop music maps have only recently been met by the tremendous increase in computational power as well as the availability of affordable large storage. Now it is possible to handle the huge amount of data within music collections and to do the complex calculations leading to the music maps described above within reasonable time.

1.1. Scope and Overview

This report explores two main aspects related to music maps. One is how to compute the similarity of two pieces of music, so a whole music collection can be organized accordingly. The second aspect is how to present this information to the user in an intuitive way. The main goal of this work is to demonstrate the possibility of building a system, which enables efficient exploration of unknown music collections, given only the raw pieces of music without any Meta information.

This work uses a music collection of 77 pieces with a total length of about 6 hours to illustrate and evaluate the methods. A detailed list of all pieces of music, their authors and titles can be found in the Appendix. The music collection consists of a mixture of pieces of music from different genres. Most of these pieces are well known so that the reader can easily verify the presented results.

Related work can be found in *Chapter 2*. In particular the fields of content-based music analysis and approaches based on the Self-Organizing Map are discussed. Work related to details on psychoacoustics, clustering algorithms, and the visualization and automatic summarization of the results is presented in the corresponding chapters.

In *Chapter 3* the methods used to extract relevant features, which enable the computer to compare two pieces of music, are presented. These features are derived from the low-level raw audio signal without any additional Meta information. Based on psychoacoustic findings, features are constructed which reflect the dynamic and rhythmic properties of music. All feature extraction steps are illustrated using pieces of music from the music collection.

Chapter 4 deals with approach used to combine and analyze the extracted features. In *Chapter 5* a novel method to visualize clusters in a Self-Organizing Map is presented along with methods to give automatic summaries for groups of music. This approach is able to describe pieces of music with different lengths. The method is evaluated with the music collection

Finally in *Chapter 6* this work is concluded. A summary and further work together with interesting directions, with possibly very promising results are discussed. We end the report ends with References and Appendix.

Chapter 2
Related Work

Music has been analyzed since the ancient Greeks. Pythagoras is credited with recognizing that strings whose lengths are related as the ratio of small integers sounds good when plucked at the same time. Since then a lot of research has been conducted and very sophisticated models and systems have been developed.

In the scope of this work especially systems which are designed to search for music based on its content are interesting, since this is the main motivation for this thesis. Section 2.1 reviews the literature on content-based music retrieval and section 2.2 focuses on approaches using the SOM algorithm. Work related to details on psychoacoustics, clustering algorithms, and the visualization and automatic summarization of the results is presented in the respective chapters.

2.1. Content-Based Music Retrieval

There are several possibilities to search for music based on its content. One is to use metadata information consisting of descriptions which have manually been assigned to each piece of music. These descriptions can be as simple as the name of the piece of music, but also more complex like an assignment to a specific genre or a verbal description of the music. The necessary standards are provided, for example, by the MPEG 7 standard [1]. A system based on MPEG 7 to compare sounds has, for example, been presented by [2]. Often pieces of music have lyrics, thus another possibility would be to apply methods from text document retrieval to search for music. Such systems are currently in use, for example, *BigLyrics.com* and *LyricCrawler.com* are two of the currently biggest music lyric search engines on the web. Using the lyrics as descriptions of the music it is possible to create an interface to allow an exploration of music archives where pieces of music with similar lyrics are located close to each other on a 2-dimensional map display using methods which have been developed mainly for text document collections, such as the SOMLib [3] or the WebSOM [4].

However, not always metadata is available and the metadata available might be erroneous, incomplete or inaccurate due to the deficiencies of manual labor. Likewise song lyrics are not always available; speech recognition systems only have a limited capability of extracting the lyrics automatically. And finally most music is available in MP3 or other similar formats rather than in MIDI. Thus content-based systems which directly analyze the raw music data (acoustical signals) have been developed. An overview of systems analyzing audio databases was presented by Foote [5]. However, Foote focuses particularly on systems for retrieval of speech or partly-speech audio data. Several studies and overviews related to content-based audio signal classification are available (e.g. [6]), however, they do not treat content-based music classification in detail.

Other approaches (e.g. [7, 8]) are based on methods developed in the digital speech processing community using *Mel Frequency Cepstral Coefficients* (MFCCs). MFCCs are motivated by perceptual and computational considerations, for example, instead of calculating the exact loudness sensation only decibel values are used. Furthermore the techniques appropriate to process speech data are not necessarily the best for processing music. For example, the MFCCs ignore some of the dynamic aspects of music.Recently Scheirer [9] presented a model of human perceptual behavior and briefly discussed how his model can be applied to classifying music into genre categories and performing music similarity-matching. However, he has not

applied his model to large scale music collections. The collection he uses consisted of 75 songs from each of which he selected two 5-second sequences.

2.2. Approaches using Self-Organizing Maps

This work is built upon the work of Fruhwirth and Rauber [10], who have shown that it is possible to cluster and organize music using neural networks. In their work they extract features from MP3 files which enable a self-organizing map (SOM) [11] to learn the inherent structure within a music collection. The feature extraction process consists of several steps.

They first transform the audio signals into the frequency domain using a fast Fourier transformation (FFT) with about 20 millisecond windows. In the frequency domain they select 17 frequencies for further processing. They split each piece of music into 5-second sequences. They remove the first and the last sequences to avoid fade-in and fade-out effects. From the remaining they select a subset using only every second to third sequence. Each frequency band from the selected sequences is then transformed into the frequency domain yielding 256 coefficients. They combine these 256 values for the 17 bands in a 4352-dimensional vector representing a 5-second sequence. These vectors reflect the dynamic properties of the selected frequencies.

A SOM is used to organize the large number of 5 second sequences on a 2-dimensional map in such a way that similar sequences are located close to each other. The different sequences of one piece of music might be scattered across the map if it contains a lot of variations. To get the pieces together again another feature vector is created using the information on where the different sequences of one piece of music are located. With this information another SOM is trained which organizes the pieces of music on a 2-dimensional map.

This work follows some of the proposals for further work presented by Fruhwirth and tries to combine the well working approach with psychoacoustics methods to improve the performance. An overview of psychoacoustics can be found in [12]. Psychoacoustics deals with the relationship of physical sounds and the human brain's interpretation of them.

Chapter 3
Feature Extraction

Music with duration of 5 minutes is usually represented by 13 million values. These values describe the physical properties of the acoustical waves, which we hear. When analyzing this data it is necessary to remove the irrelevant parts and emphasize the important features. The extraction of these features from the raw data is the most critical part in the process of creating a content-based organization in a music collection. If it were possible to extract one single feature that directly indicates which genre a piece of music belongs to, everything else would be trivial? Good features should be intuitively meaningful, based on psychoacoustic findings, and robust towards variations which are insignificant to our hearing sensation. Furthermore, they should lead to an organization of the music collection that makes sense and not be too expensive to compute.

It is necessary to consider computational aspects because the raw data of even small music collections easily consumes several gigabytes of storage. A detailed analysis of all this information and all its possible meanings would be computationally prohibitive. It thus is necessary to reduce the amount of information to what is relevant in respect to the overall goal, which is to organize music according to its genre. These genres are not clearly defined and different people might assign the same piece of music to different genres. However, there are some attributes of the raw data, which definitely do not determine the genre. For example, removing the first second of a piece of music does not change its genre, but the raw data compared bit wise will be completely different. Generally the duration of a piece of music is not relevant. Neither does a particular melody define a genre. The same melody can be interpreted in different genre styles just as different melodies might be members of the same genre. Likewise, the number of instruments involved plays a minor role in defining the genre.

One of the attributes that is rather typical for a genre is its rhythm which is why this work primarily focuses on the dynamics of music, and in particular on the fluctuation strength [13] of the specific loudness per critical-band [14]. The following sections describe the feature extraction steps starting with the raw data, which is transformed from the time-domain to its frequency-domain representation. In the frequency-domain several transformations are applied to obtain the specific loudness per critical-band. Based on the specific loudness per critical-band the loudness fluctuation in a time interval of about 6 seconds is analyzed and an image in the dimensions of critical-band, modulation frequency, and fluctuation strength is created. To this image gradient filters (for edge detection) and Gaussian filters (for smoothening) are applied to emphasize important characteristics and remove insignificant ones. The modified fluctuation strength is used as final feature for the clustering algorithms.

3.1. Loudness Sensation

Loudness belongs to the category of intensity sensations. The loudness of a sound is measured by comparing it to a reference sound. The 1 kHz tone is a very popular reference tone in psychoacoustics, and the loudness of the 1 kHz tone at 40dB is defined to be 1 Sone. A sound perceived to be twice as loud is defined to be 2 sone and so on. To calculate the loudness sensation from raw audio data several transformations are necessary. The raw audio data is first decomposed into its frequencies using a discrete Fourier transformation. These frequencies are bundled according to the nonlinear critical-band rate scale (bark). Then spectral masking effects are applied before the decibel values are calculated. The decibel values are transformed to equal loudness levels (phon) and finally from these the specific loudness sensation is

calculated (sone). At the end of this section each transformation is illustrated using examples from the music collection used for the experiments conducted for this work, as well as using some artificially generated sinusoidal signals

3.1.1. Discrete Fourier Transformation

Complex acoustical signals consist of several waves with different frequencies and amplitudes. The inner ear *(cochlea)* of humans decomposes the incoming acoustical waves into separate frequencies. The energy of different frequencies is transferred to and concentrated at different locations along the basilar membrane. Thus, it is appropriate to transform the PCM data into the frequency domain before analyzing it further. This can be achieved using, for example, *Fourier Transformations*. Alternatives include *Wavelets,* but are not considered in this work.

In this subsection only the most important characteristics are summarized. A more detailed description can be found, for example, in [15]. One of the aspects of music is that the frequencies change continuously; however, within very short time frames the frequencies are approximately constant. These very short sequences can be seen as fundamental building blocks of music. Thus, a piece of music can be described with subsequent frequency patterns, each representing a time quantum. A common choice for this interval is 20ms. The music data used for this work is sampled at 22050 Hz. To optimize the FFT the number of samples N should be a power of 2. However, this does not mean that it is necessary to have N samples. Shorter signals can be padded with zeros. Using 23ms time frames corresponds to 512 samples and results in a frequency resolution of about 43Hz in a range from 0 to 11 kHz. A 50% overlap between the windows is used. Notice that the sum of the two overlapping Hanning windows at any point always equals 1. A 50% overlap increases the time resolution by a factor 2 to about 12ms.

The calculations are implemented as follows. The raw audio data is given in vectors y(t) of length N corresponding to 23ms at the time frame t. The power spectrum matrix P (n; t), where n is the index for the frequency and t for the time frame can be calculated using,

$$\begin{aligned} y'_t &= W_N\, y_t, \\ Y_t &= fft(y_t), \text{ and} \\ P(n,t) &= |Y_t(n)|^2 \frac{1}{N} \end{aligned} \qquad \text{..... (1)}$$

The index n ranges from 1 to N/2+1. The matrix W_N contains the Hanning function weights for N points on the diagonal with zeros elsewhere where N = 512 at 11 kHz. The *fft* function is taken from the FFTW library. The data after this feature extraction step basically still has the same size. While the discrete Fourier transformation yields 256 values for 512 sample values, the 50% overlap increases the amount of data by 2.

3.1.2. Critical-Bands

So far a piece of music is represented by a frequency snapshot every 12ms. These have one value every 43Hz starting at 0Hz up to 11 kHz, where each value represents the power of the respective frequency. As stated previously, the inner ear separates the frequencies, transfers, and concentrates them at certain locations along the basilar membrane. The inner ear can be regarded as a complex system of a series of band-pass filters with an asymmetrical shape of frequency response. The center frequencies of these band-pass filters are closely related to the critical-band rates. Where these bands should be centered or how wide

they should be, has been analyzed throughout several psychoacoustic experiments [12]. While we can distinguish low frequencies of up to about 500Hz well, our ability decreases above 500Hz with approximately a factor of 0:2*f*, where *f* is the frequency. This is shown in experiments using a loud tone to mask a quieter one. At high frequencies these two tones need to be rather far apart regarding their frequencies, while at lower frequencies the quiet tone will still be noticeable at smaller distances. In addition to these masking effects the critical-bandwidth is also very closely related to just noticeable frequency variations. Within a critical-band it is difficult to notice any variations. This can be tested by presenting two tones to a listener and asking which of the two has a higher or lower frequency.

Since the critical-band scale has been used very frequently, it has been assigned a unit, the *bark*. The name has been chosen in memory of Barkhausen, a scientist who introduced the phon to describe loudness levels for which critical-bands play an important role. Figure 2 shows the main characteristics of this scale. At low frequencies below 500Hz the critical-bands are about 100Hz wide. The width of the critical bands increases rapidly with the frequency. The 24th critical-band has a width of 3500Hz. The 9th critical-band has the center frequency of 1 kHz. The critical-band rate is important for understanding many characteristics of the human ear.

A critical-band value is calculated by summing up the values of the power spectrum within the respective lower $f_a(i)$ and upper $f_b(i)$ frequency limits of the i^{th} critical band. This can be formulated as:

$$B(i,t) = \sum_{n \in I(i)} P(n,t),\ I(i) = \left\{ n \middle| f_a(i) < f_{\text{res}}(n-1, 256, 1/11025) \le f_b(i) \right\} \quad \text{..... (2)}$$

Where *i*, *t*, *n* are indexes and B is a matrix containing the power within the *i*-th critical band at a specific time interval *t*. P is the matrix representing the power per frequency and time interval *t* obtained from Equation (1). Notice that P (1*; t*), which represents the power at 0Hz, is not used. While the critical-band rate is defined having 24 bands, only the first 20 are used in this work, since the highest frequencies in the data are limited to 11 kHz. The 256 power spectrum values are now represented by 20 critical-bands values. This corresponds to a data reduction by a factor of about 6.5.

3.1.3. Masking

As mentioned before, the critical-bands are closely related to masking effects. Masking is the occlusion of one sound by another sound. A loud sound might mask a simultaneous sound (simultaneous masking), or a sound closely following (post-masking) or preceding (pre-masking) it. Pre-masking is usually neglected since it can only be measured during about 20ms. Post-masking, on the other hand can last longer than 100ms and ends after about a 200ms delay. Simultaneous masking occurs when the test sound and the masker are present simultaneously. For this thesis the spreading function is used to estimate the effects of simultaneous masking across the critical-bands. The spreading function defines the influence of the *j*-th critical-band on the *i*-th and is calculated as:

$$S(i,j) = 15.81 + 7.5(i - j + 0.474) - 17.5\sqrt{1 + (i - j + 0.474)^2} \quad \text{..... (3)}$$

The spread critical-band rate spectrum matrix B*S* is obtained by multiplying B with S as follows:

$$B_s(i,t) = \sum_{j=1}^{20} S(i,j)B(j,t), \text{ which is equivalent to} \quad \text{..... (4)}$$

$$B_s = SB.$$

The simultaneous masking asymmetrically spreads the power spectrum over the critical bands. The masking influence of a critical-band is higher on bands above it than on those below it.

3.1.4. Decibel

Before calculating sone values it is necessary to transform the data into decibel. The intensity unit of physical audio signals is sound pressure and is measured in *Pascal* (Pa). The values of the PCM data correspond to the sound pressure. It is very common to transform the sound pressure into *decibel* (dB). Decibel is the logarithm, to the base of 10, of the ratio between two amounts of power. The decibel value of a sound is calculated as the ratio between its pressure and the pressure of the hearing threshold given by 20uPa. The sound pressure level in dB is calculated as

$$S_{dB} = 10\log_{10}\frac{p}{p_0} \quad \text{..... (5)}$$

Where p is the power of the sound pressure, and p_0 is the power of the sound pressure of the hearing threshold. The power is calculated as the squared sound pressure. Parseval's theorem states that the power of the signal is the same whether calculated in the time domain or the frequency domain, so the dB values can be calculated for the spread critical-band matrix B*S*. A parameter to adjust is the reference value $p0$. Note that the influence of the hearing threshold p_0 on the decibel calculations is non-linear.

If the hearing threshold is too low insignificant sounds will become significant, on the other hand if it is too high significant sounds will become insignificant. Knowing that the sound pressure of the signals is digitized using 16 bit, it could be assumed that the most quiet, just noticeable power of the sound pressure level corresponds to 1 (or -1). When using $p_0 = 1$ the notation db (SPL) is used, where SPL stands for sound pressure level. The maximal decibel value for the energy at a certain frequency using db (SPL) is 96dB. However, the use of this assumption has led to sone values beyond the limit of damage risk in the experiments conducted for this thesis. This occurs because the energy of the frequencies are added together in the critical-bands and the masking function used is additive. The problem with decibel values beyond the limit of damage risk is that only equal loudness contours for levels below the limit are available, thus it is not possible to calculate accurate phon values (see next section), which have a non-linear correlation to the decibel values. To be able to calculate the phon values the PCM amplitudes of the music collection were scaled so that all sounds are below the limit of damage risk. This corresponds to turning down the volume to a level at which the loudness of all music listened to, is within healthy ranges. The hearing threshold parameter p_0 was set to 1/0.35. The loudness matrix in decibel, L_{dB}, is calculated as follows,

$$\mathrm{B}'_s(i,t) = \min(B_s(i,t), p_0), \text{ and}$$

$$\mathrm{L}_{\mathrm{dB}}(i,t) = 10\log_{10}\frac{1}{p_0}\mathrm{B}'_s(i,t) \quad \text{..... (6)}$$

Note that the first step is made in order to avoid the logarithm of zero.

3.1.5. Phon

The relationship between the sound pressure level in decibel and our hearing sensation measured in sone is not linear. The perceived loudness depends on the frequency of the tone. The phon is defined using the 1 kHz tone and the decibel scale. For example, a pure tone at any frequency with 40 phon is as loud as a pure

tone with 40dB at 1 kHz. We are most sensitive to frequencies around 2 kHz to 5 kHz. The hearing threshold rapidly raises around the lower and upper frequency limits, which are respectively about 20 Hz and 16 kHz.

Although the equal loudness contours are obtained from experiments with pure tones, they are frequently applied to calculate the specific loudness of the critical band rate spectrum. The loudness matrix in phon, L_{phon}, can be calculated using the equal loudness contour matrix C_{elc} and the corresponding phone values to each contour c_{phon} = [3; 20; 40; 60; 80; 100]. $C_{elc}(i; j)$ contains the decibel values of the j-th loudness contour at the i-th critical-band. Values in between two equal loudness contours are interpolated linearly, as follows:

$$
\begin{aligned}
&L'_{dB}(i,t) = \max(L_{dB}(i,t), C_{elc}(i,1)),\\
&level_{i,t} = \arg\min_{j}(L'_{dB}(i,t) < C_{elc}(i,j)),\\
&r_{i,t} = \frac{L'_{dB}(i,t) - C_{elc}(i, level_{i,t}-1)}{C_{elc}(i, level_{i,t}) - C_{elc}(i, level_{i,t}-1)}, \text{ and}\\
&\mathrm{L}_{\mathrm{phon}}(i,t) = c_{phon}(level_{i,t}-1) + r_{i,t}c_{phon}(level_{i,t})
\end{aligned}
\qquad \text{..... (7)}
$$

Note that $L_{phon}(i; t)$ can only be calculated if there exists an equal loudness contour j in $C_{elc}(i; j)$ so that $C_{phon}(j) > L_{phon}(i; t)$. The highest decibel level used here is 100 dB and p_0 (see above) is adjusted manually so all sounds are below this limit.

3.1.6. Sone

Finally, from the loudness level L_{phon} the specific loudness sensation L_{sone} per critical band, following [16], is calculated as,

$$
L_{sone}(i,t) = \begin{cases} 2^{\frac{1}{10}(L_{phon}(i,t)-40)} & \text{if } L_{phon}(i,t) > 40 \\ \left(\frac{1}{40}L_{phon}(i,t)\right)^{2.642} & \text{otherwise.} \end{cases}
\qquad \text{..... (8)}
$$

For low values up to 40 phon the sensation raises slowly until it reaches 1 sone at 40 phon. Beyond 40 phon the sensation increases at a faster rate. The highest values that occurred in the experiments conducted for this thesis are below 60 sone, due to the adjustment of the threshold in quiet $p0$.

3.1.7. System Architecture for Sone Calculation

Prior to Sonogram Extraction, each mp3 (stereo) file is converted to wav (mono) file. Also each audio file is segmented into 6-second sequences. For feature extraction, the first and the last 6-sec sequence of the audio song are left out to eliminate the fade-in and fade-out effect, also only every third 6-sec sequence is chosen for feature extraction. Now every 6-sec sequence is given as input to the block diagram below for Sonogram extraction:

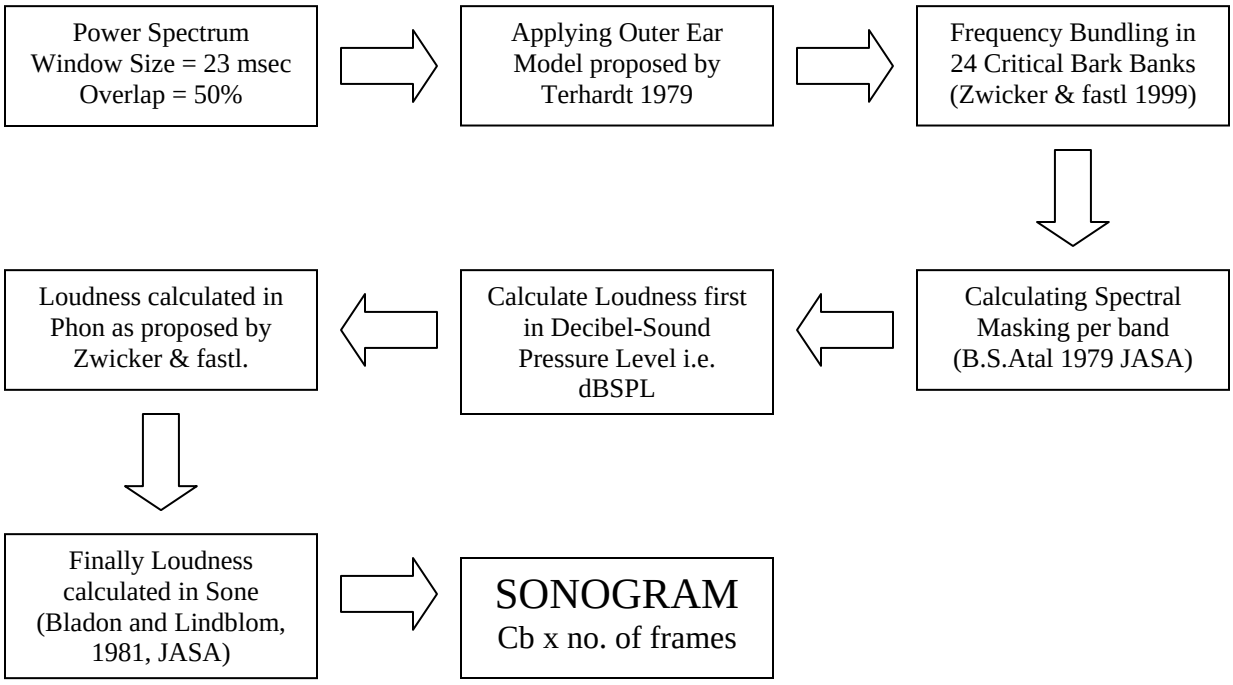

Figure 1. Flow Diagram for Sonogram Calculation

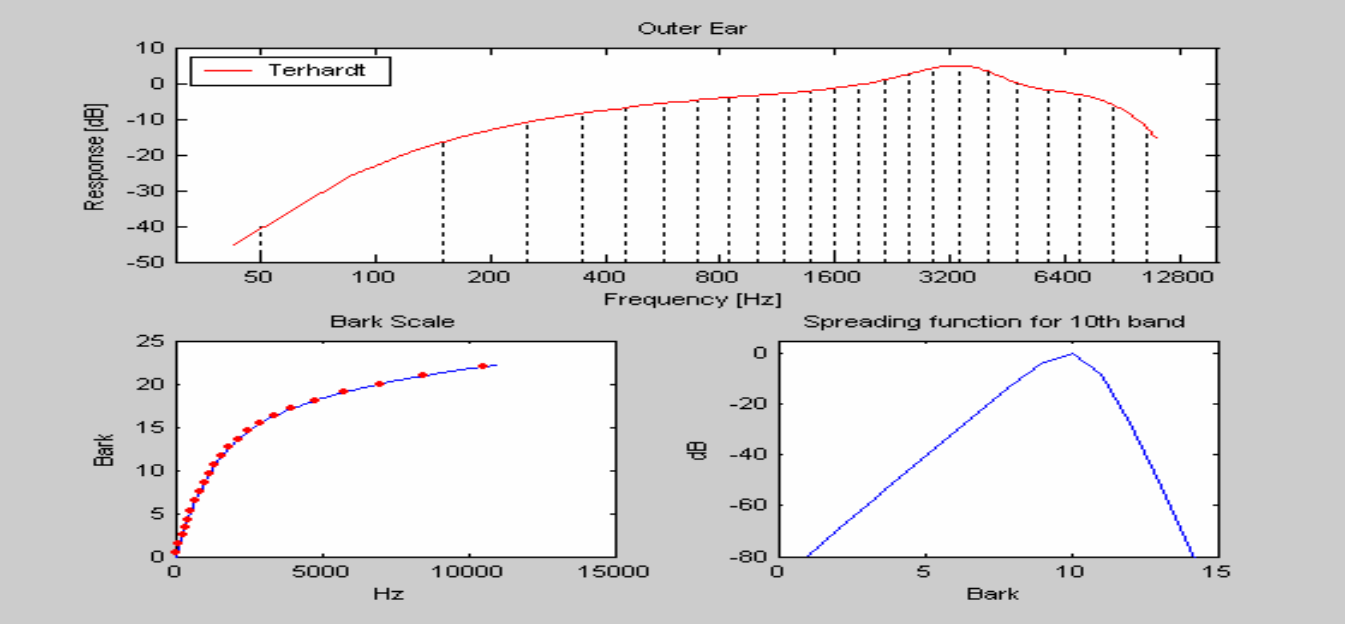

Figure 2. The main characteristics of the sone model. The upper subplot shows the width of the critical-bands and the outer-ear model. The lower left shows the relationship between the Bark-scale and Hz. The lower right shows the spectral masking function. Some things to notice: The Bark-scale is linear up to about 500Hz. The spectral masking is not symmetric. Higher tones are masked stronger by lower ones than vice versa. The outer-ear is most responsive to frequencies around 3-4 kHz.

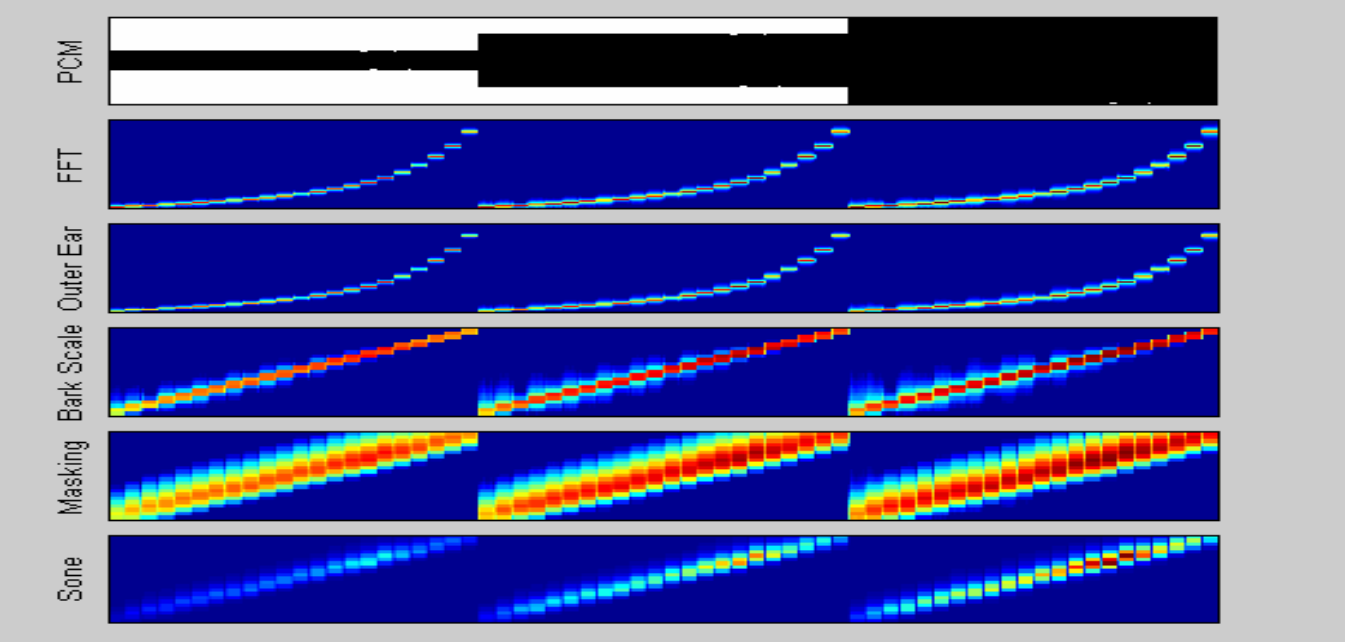

Figure 3. The characteristics of the sone model can be demonstrated with a simple sine-tone. The tone is generated at different frequencies (center frequencies of the critical-bands) with fixed amplitude (3 times, each time the fixed amplitude is increased). Note the difference between the linear frequency scale (used in the outer-ear and FFT representations) and the Bark-scale (the lower 3 subplots). The sinusoids in each sweep from low to high have the same amplitudes, they are not perceived equally loud. This pitch dependent loudness sensation is reflected in the model (in contrast to the MFCC model).

3.1.8. System Architecture for Sonogram Calculation

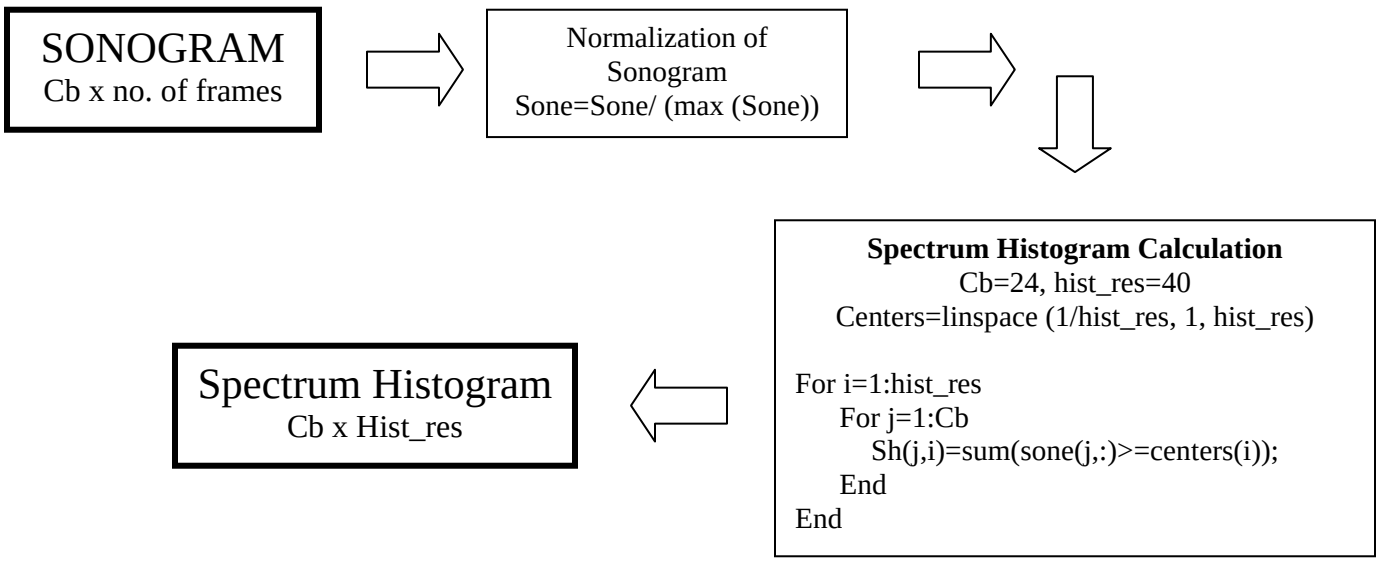

Figure 4. Flow Diagram for Spectrum Histogram calculation

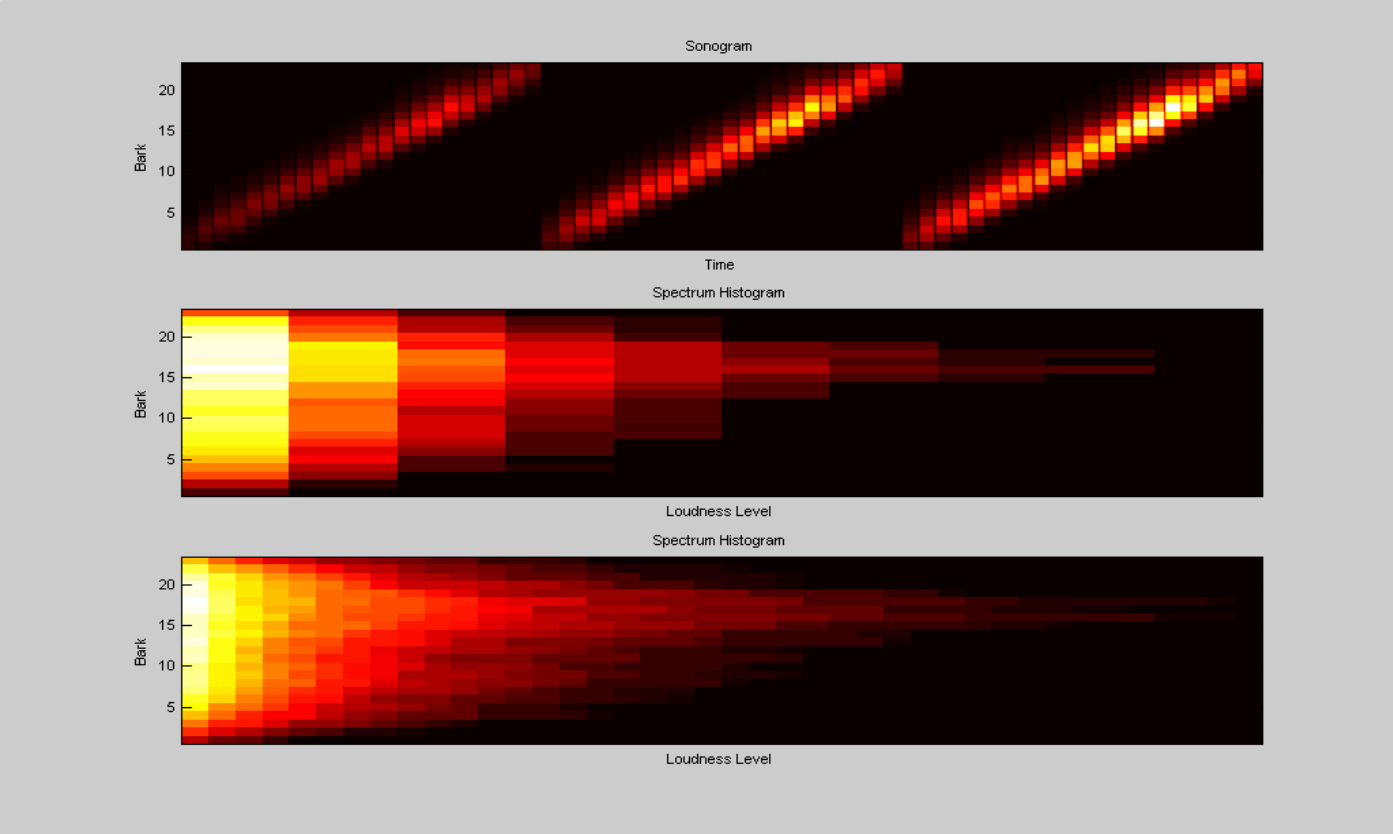

Figure 5. Same test sound as in the examples above. The spectrum histogram counts how often a loudness level was exceeded in each critical-band. Two spectrum Histogram are shown with hist_res=10 and hist_res=25 respectively.

3.2. Dynamics

So far each song is represented by several 6-second sequences of the specific loudness sensation per critical-band, $L_{sone}(i; t)$. It would be possible to use $L_{sone}(i; t)$ to calculate similarities between the data. One option would be to compare two sequences La_{sone} and Lb_{sone} point-wise, i.e. comparing $La_{sone}(i; t)$ and $Lb_{sone}(i; t)$ for all i and t. The result might be quite surprising. For example, shifting *Rock DJ* by only 40ms would result in a huge difference to the un-shifted sequences - although they sound the same. The same problem would occur for any of the other sequences presented in the previous section. Thus, the final representation of the data must be invariant to time shifts.

As mentioned before, the aim is to gather information on the dynamics of a sequence. Weil [17] and Fruhwirth have used the Fourier transforms of the activities in the frequency bands, which are also used here. Ellis [18] has shown that using a similar concept it is possible to predict the pitch of a sound. Ellis analyzed periodically recurring peeks in the loudness of a frequency band by calculating the autocorrelation. The main difference between Weil, Fruhwirth and Ellis is the frequency ranges they analyze. While Ellis analyzes patterns with periods of about one millisecond (which corresponds to frequencies up to 1 kHz), Fruhwirth limits his investigation to frequencies up to 25Hz. Weil uses a similar spectrum as Fruhwirth, though limits his analysis to 15Hz.

3.2.1. Amplitude Modulated Loudness

The loudness of a critical-band usually rises and falls several times. Often there is a periodical pattern, also known as the rhythm. At every beat the loudness sensation rises, and the beats are usually very accurately timed. The loudness values of a critical-band over a certain time period can be regarded as a signal that has been sampled at discrete points in time. The periodical patterns of this signal can then be assumed to originate from a mixture of sinusoids. These sinusoids modulate the amplitude of the loudness, and can be calculated by a Fourier transform. An example might illustrate this. To add a strong and deep bass with 120 *beats per minute* (bpm) to a piece of music, a good start would be to set the first critical-band (bark 1) to a constant noise sensation of 10 sone. Then one could modulate the loudness using a sine wave with a period of 2Hz and amplitude of 10 sone.

The modulation frequencies, which can be analyzed using the 6-second sequences and time quanta of 12ms, are in the range from 0 to 43Hz with an accuracy of 0.17Hz. Notice that a modulation frequency of 43Hz corresponds to almost 2600bpm. The modulation amplitude delta L*i*(*n*) with the frequency of the *i*-th critical-band is calculated as follows:

$$\Delta L_i = fft(L_{some}(i,1....511)), \qquad \text{..... (8)}$$

Where $L_{sone}(i;\ 1 \ldots. 511)$ is a 6-second sequence of the *i*-th critical-band of any piece of music. The *fft* function is the same as in Equation (1). Since there are only 511 values for the FFT, the signal is padded with one zero.

3.2.2. Fluctuation Strength

The amplitude modulation of the loudness has different effects on our sensation depending on the frequency. The sensation of *fluctuation strength* [13] is most intense around 4Hz and gradually decreases up to a modulation frequency of 15Hz (cf. Figure 3.9). At 15Hz the sensation of *roughness* starts to increase, reaches its maximum at about 70Hz, and starts to decreases at about 150 Hz. Above 150 Hz the sensation of hearing *three separately audible tones* increases [12]. The fluctuation strength of a tone with the loudness delta *L*, which is 100% amplitude modulated with the frequency f_{mod} can be expressed by,

$$f_{flux}(\Delta L, f_{\mathrm{mod}}) \propto \frac{\Delta L}{(f_{\mathrm{mod}} / 4H_z) + (4H_z / f_{\mathrm{mod}})} \qquad \text{..... (9)}$$

The modulation amplitudes F(*i; n*) of the *i*-th critical-band are weighted according to the fluctuation strength sensation as follows,

$$F(i,n) = f_{flux}(\left|\Delta L_i(n+1)\right|, f_{res}(n)), \qquad \text{..... (10)}$$

3.2.3. System Architecture for Fluctuation Pattern Calculation

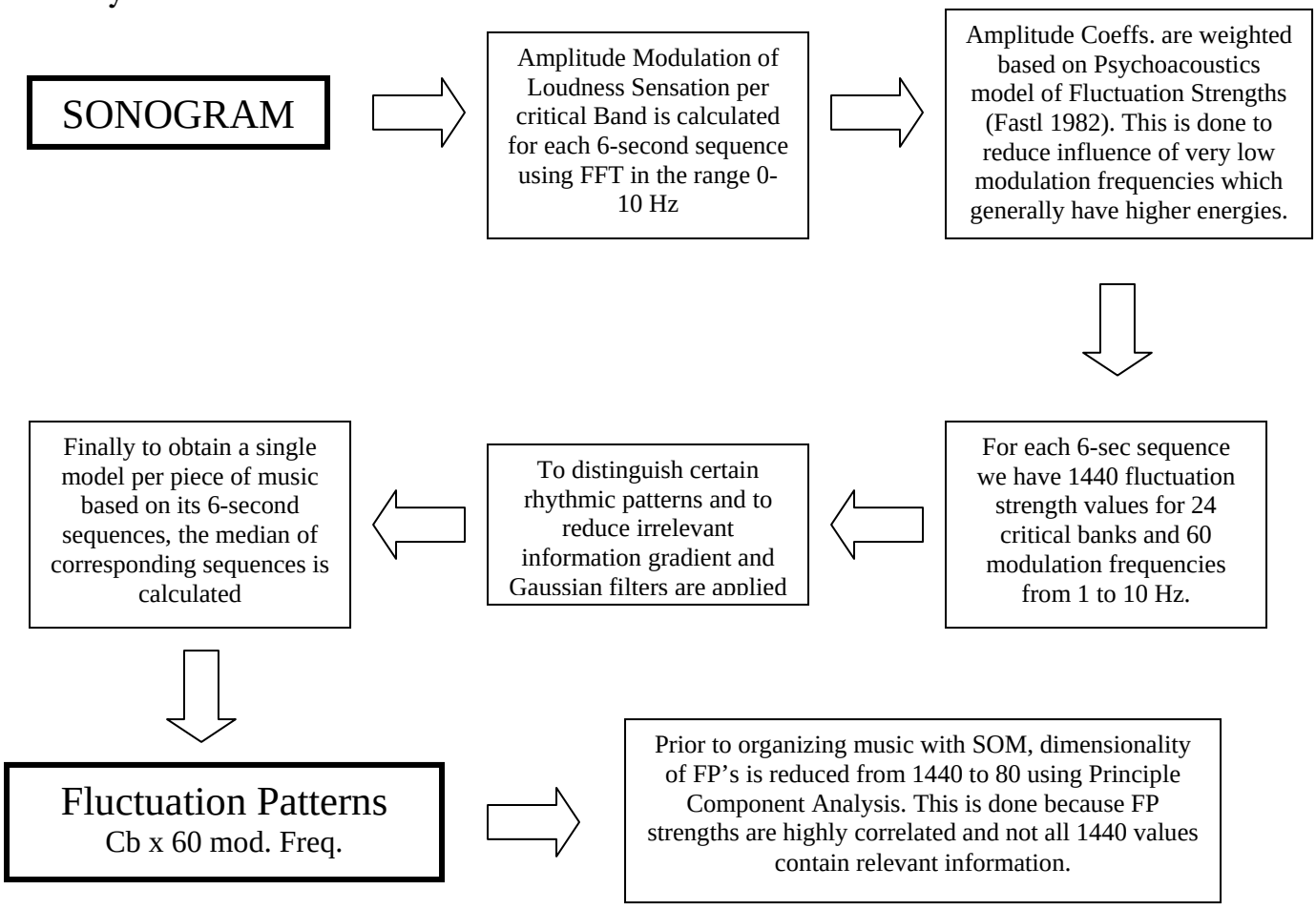

Figure 6. Flow Diagram for Fluctuation Pattern Calculation

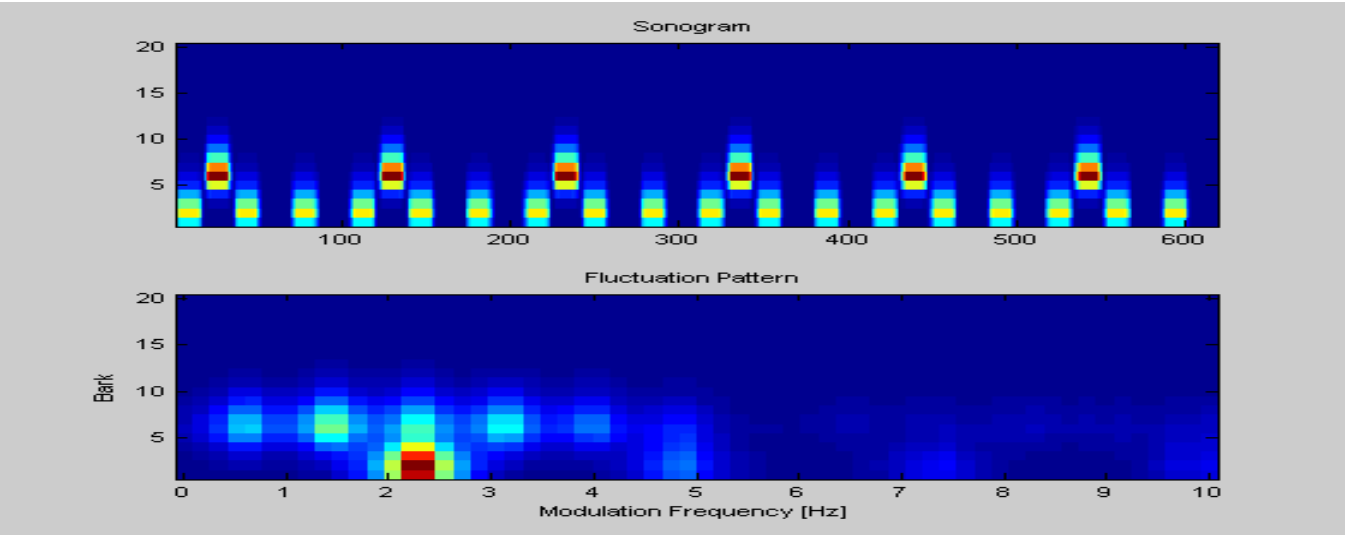

Figure 7. Final Result after computing the fluctuation pattern for a 6sec sequence.

3.3. Periodic Histograms

To obtain periodicity histograms we use an approach presented by Scheirer (1998) in the context of beat tracking. A similar approach was developed by Tzanetakis and Cook (2002) to classify genres. There are two main differences to this previous work. First, we extend the typical histograms to incorporate information on the variations over time which is valuable information when considering similarity. Second, we use a resonance model proposed by Moelants (2002) for preferred tempo to weight the periodicities and in particular to emphasize differences in tempos around 120 beats per minute (bpm). We start with the preprocessed data and further process it using a half wave rectified difference filter on each critical-band to emphasize percussive sounds. We then process 12 second windows (1024 samples) with 6 second overlap (512 samples). Each window is weighted using a Hann window before a comb filter bank is applied to each critical-band with a 5bpm resolution in the range from 40 to 240bpm. Then we apply the resonance model of Moelants (2002) with Beta = 4 to the amplitudes obtained from the comb filter. To emphasize peaks we use a full wave rectified difference filter before summing up the amplitudes for each periodicity over all bands.

That gives us, for every 6 seconds of music, 40 values representing the strength of recurring beats with tempos ranging from 40 to 240bpm. To summarize this information for a whole piece of music we use a 2-dimensional histogram with 40 equally spaced columns representing different tempos and 50 rows representing strength levels. The histogram counts for each periodicity how many times a level equal to or greater than a specific value was reached. This partially preserves information on the distribution of the strength levels over time. The sum of the histogram is normalized to one, and the distance between two histograms is computed by interpreting them as 2000-dimensional vectors in a Euclidean space.

Examples for periodicity histograms are given in Figure 4. The histogram has clear edges if a particular strength level is reached constantly and the edges will be very blurry if there are strong variations in the strength level. It is important to notice that the beats of music with strong variations in tempo cannot be described using this approach. Furthermore, not all 2000 dimensions contain information. Many are highly correlated, thus it makes sense to compress the representation using principal component analysis. For the experiments presented in this paper we used the first 60 principal components. A first quantitative evaluation of the periodicity histograms indicated that they are not well suited to measure the similarity of genres or artists in contrast to measures which use spectrum information. One reason might be that the pieces of an artist might be better distinguishable in terms of rhythm than timbre. However, it is also important to realize that using periodicity histograms in this simple way (i.e., interpreting them as images and comparing them pixel-wise) to describe rhythm has severe limitations. For example, the distance between two pieces with strong peaks at 60bpm and 200bpm is the same as between pieces with peaks at 100bpm and 120bpm.

3.3.1. System Architecture for Periodic Histogram Calculation

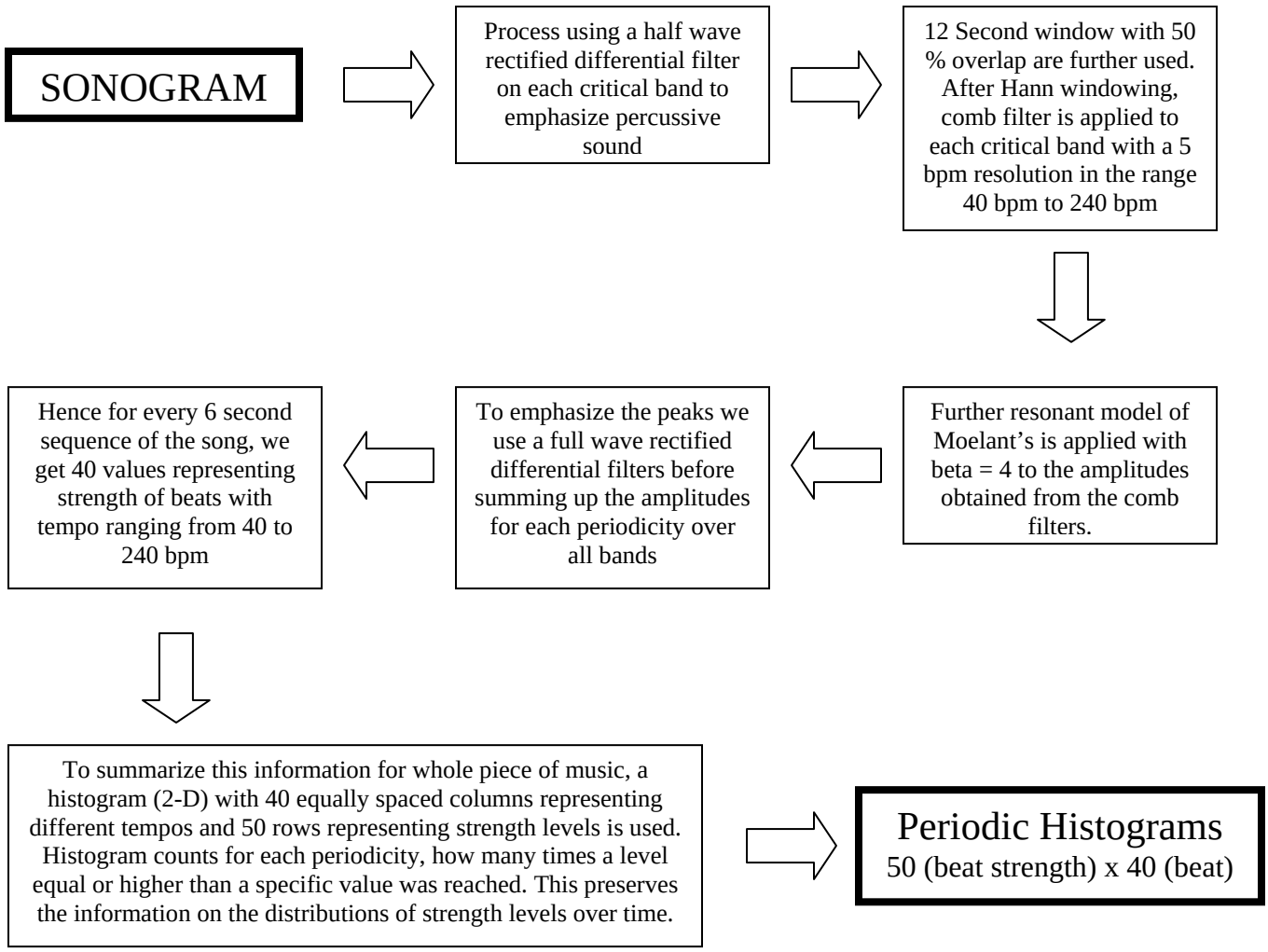

Figure 8. Flow Diagram for Periodic Histogram Calculation

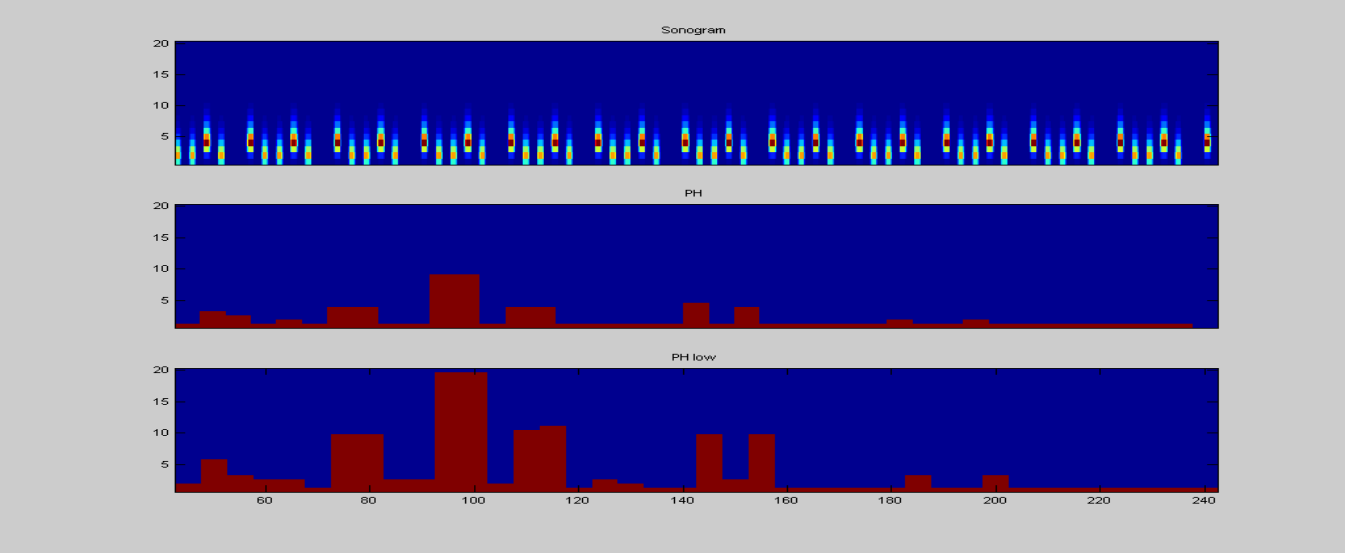

Figure 9. If the input is long enough, then the results from each 12sec frame are summarized in a histogram, counting for each periodicity (bpm) how often specific energy levels were exceeded.

Chapter 4
Clustering

The previous chapter dealt with the extraction of features from the raw music data. In this chapter the music collection (Appendix B) is organized based on these features. The main tool used for this is the self-organizing map (SOM) algorithm, which is a clustering method with intuitive visualization capabilities. In the process of developing this thesis the SOM has been applied several times to support the evaluation of the feature extraction process and the SOM is the basis of the final user interface to the music collection presented in this thesis.

In Section 4.1 the SOM is described briefly. Applying the SOM in Section 4.2 a brief evaluation of the features extracted in Chapter 3 is presented and discussed. Section 4.3 deals with different approaches to represent one piece of music based on the representation of its sequences. And finally in Section 4.4 the chapter is summarized.

4.1. Self-Organizing Maps

The goal of clustering data is to find groups (clusters) of data items that are similar to each other and different from the rest of the dataset. Clustering is a method to summarize the main characteristics of a dataset. For example, a music collection could be summarized by describing the groups (genres) it consists of. Each group could be described, for example, by the number and variation of its members. It might also be interesting to know the relationship between these groups. For example, a genre might have several sub-genres. For the purpose of clustering data several algorithms have been developed. A recent review can be found in [19]. One very frequently employed clustering algorithm is the SOM.

4.1.1. Background

The SOM was developed 1981 [20] as an artificial neural network, which models biological brain functions. Since then it has undergone thorough analysis [Koh01]. The algorithm and its variations have been employed several times in domains such as machine vision, image analysis, optical character recognition, speech analysis, and engineering applications in general. The SOM is a powerful tool that can be used in most data-mining processes especially in data exploration. Moreover, the SOM is very efficient compared to other non-linear alternatives such as the Generative Topographic Mapping [21], Sammon's Mapping [22], or generally Multi Dimensional Scaling. An ex- ample for the efficiency of the SOM is the WebSOM1 project.

4.1.2. The Batch SOM Algorithm

One of the variations of the original SOM is the batch SOM algorithm, which is significantly faster and has one parameter less to adjust [23]. Although the algorithm is different the architecture of the map is the same. The map consists of map units, which are ordered on a grid. Usually this grid is rectangular and 2-dimensional and is used to visualize the data. An example can be seen in Figure 5. Each of the map units is assigned to a reference vector, also known as model or prototype vector. This vector lies in the data space and represents the data items that are closest to it. Units, which are close to each other on the grid, also have similar model vectors and thus represent similar data.

The batch SOM algorithm consists of two steps that are iteratively repeated until no more significant changes occur. First the distance between all data items x_i and the model vectors m_j is computed and each

data item *i* is assigned to the unit that represents it best *Ci*. V*j* denotes the Vernoi set of data items which are best represented by unit m_j .

In the second step each model vector is adapted to better fit the data it represents. To ensure that each unit *j* represents similar data items as its neighbors, the model vector m*j* is adapted not only according to V*j* but also in regard to the Vernoi sets of the units in the neighborhood. Which units are considered to be neighbors and how much influence they have on the unit *j* is defined by a neighborhood function. A common choice for the neighborhood function *ht(j; k)* is a Gaussian function which is centered on m*j* and has a standard deviation $sigma_t$ which decreases with each iteration *t*. Assuming a Euclidean vector space, the two steps of the batch SOM algorithm can be formulated as

$$c_i = \arg\min_j \left\| x_i - m_j \right\|^2, and$$
$$m_j^* = \frac{\sum_i h_t(j, c_i) x_i}{\sum_{i'} h_t(j, c_{i'})}, \qquad \text{..... (11)}$$

Where m*$_j$ is the updated model vector. Although it is usually very unlikely it might occur that a data vector x*i* is equally closest to two or more model vectors. In this case randomly one of these model vectors should be chosen to be *Ci*. An efficient way to implement this is to first calculate the sum S*j* of all data vectors in a Vernoi set V*j* and then use them to calculate the weighted average,

$$V_j = \{ x_i \mid c_i = j \},$$
$$s_j = \sum_{x_i \in V_k} x_i, and \qquad \text{..... (12)}$$
$$m_j^* = \frac{\sum_k h_t(j,k) s_j}{\sum_{k'} h_t(j,k') |V_{k'}|}$$

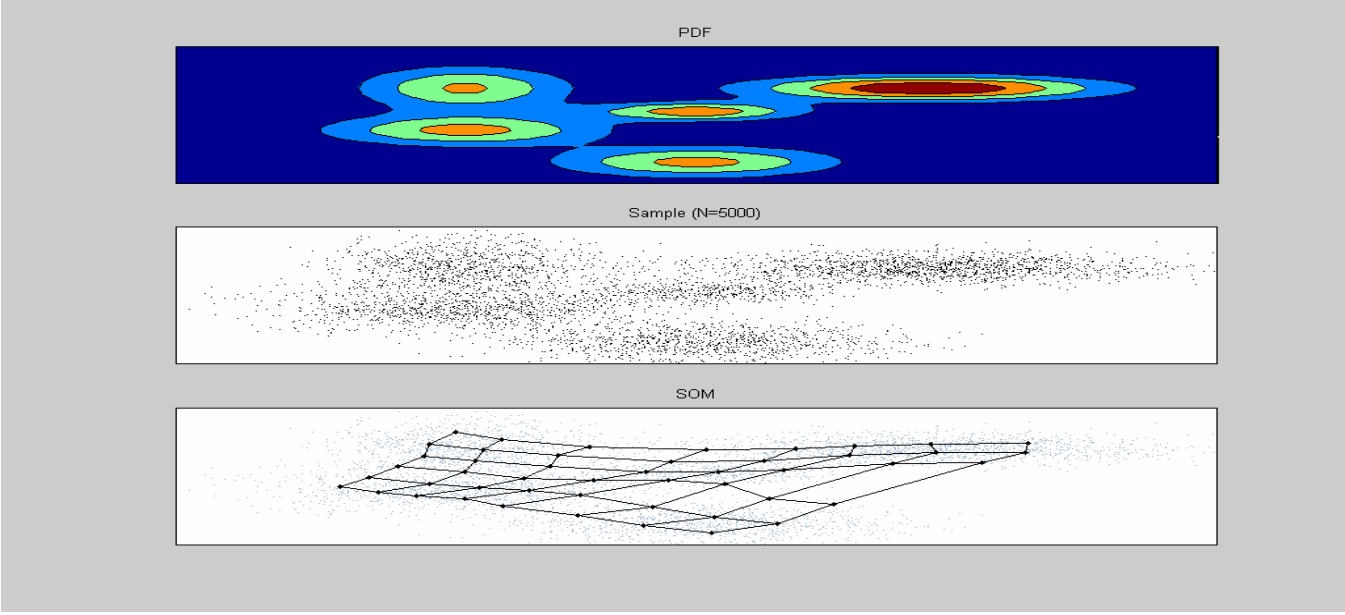

Figure 10. A test result after running SOM Algorithm on the above shown 5 clusters.

Chapter 5
Visualization and User Interface

In the previous Chapter SOMs labeled with the song identifiers were used to evaluate the music collection. Basically these maps could be used as user interfaces. Similar songs are located close together on the map and are identified by a string short enough to fit the width of a map unit. While these maps surely are a bigger help then a simple alphabetical list there are two major deficiencies.

The first is the lacking support trying to understand the cluster structure. Looking, for example, at Figure 4.13 it is rather difficult to recognize clusters on the first sight. Only after carefully studying the whole map it appears, for example, that there is a cluster of classical music in the lower left corner. Thus some visual support to identify clusters would be desirable. The second deficiency derives from the assumption that the music collection and the contained pieces of music are unknown to the user, thus any information on artists or titles are not very useful for the user. To a user a map labeled with unknown music titles, interprets or authors might not be much more useful than randomly generated text. By far more interesting for the user would be some text, which explains what type of music, is mapped to a map unit.

In the following sections methods are described which aim at creating a more intuitive user interfaces. In Section 5.1 the problem of visualizing cluster structure is addressed followed by Section 5.2 where important map areas are summarized and labeled. Both sections use the map presented in Figure 4.13 to illustrate the different possibilities. To enable the reader to explore Islands of Music a small demonstration has been made available on the internet and is briefly presented in Section 5.3. The chapter is summarized in Section 5.4.

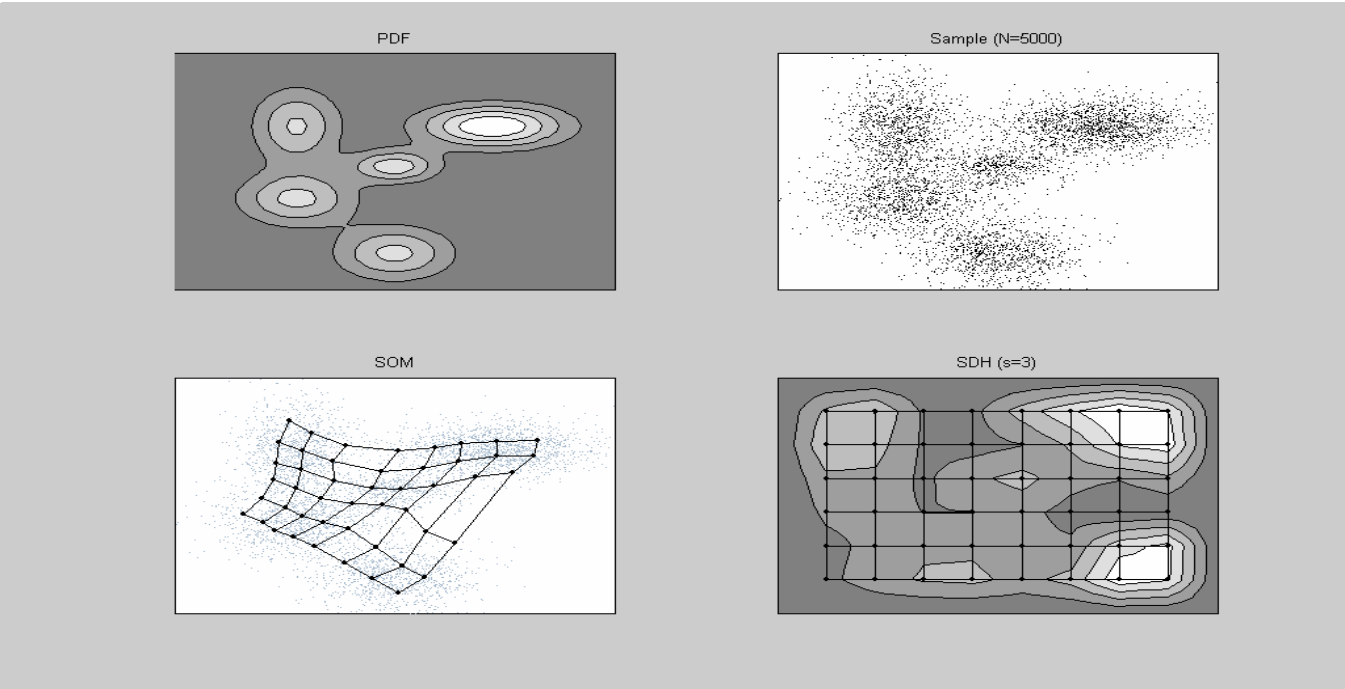

Figure 11. A Smooth Data Histogram representation of results obtained by SOM algorithm in Figure 10.

5.1. Islands

A useful visualization system should offer the user a good summary of the relevant information. To be effective the distinguishing features (e.g. position, form, and color) in the visual dimensions should be detectable effortlessly and quickly by the human visual system in the preattentive processing phase. In general the efficiency of visualization will depend on the domain, culture, and personal preferences of the users.

The results of the previous chapters can be visualized in several different manners, only a few will be discussed in this section. This thesis has been titled *Islands of Music* because the metaphor of islands is used to visualize music collections. The metaphor is based on islands, which represent groups of similar data items (pieces of music). These islands are surrounded by the sea, which corresponds to areas on the map where mainly outliers or data items, which do not belong to specific clusters, can be found. The islands can have arbitrary shapes, and there might be land passages between islands, which indicate that there are connections between the clusters. Within an island there might be sub-clusters. Mountains and hills represent these. A mountain peek corresponds to the center of a cluster and an island might have several mountains or hills.

More accurately a simple approximation of the probability density function is visualized using a color scale which ranges from dark blue (deep sea) to light blue (shallow water) to yellow (beach) to dark green (forest) to light green (hills) to gray (rocks) and finally white (snow). The exact color scale represented by HSV (see e.g. [24]) values. The sea level has been set to 1/3 of the total color range, thus map units with a probability of less then 1/3 are under water. The sea level could be adjusted by the user in real time, the involved calculations are neglectable. The visible effects might aid understanding the islands and the corresponding clusters better.

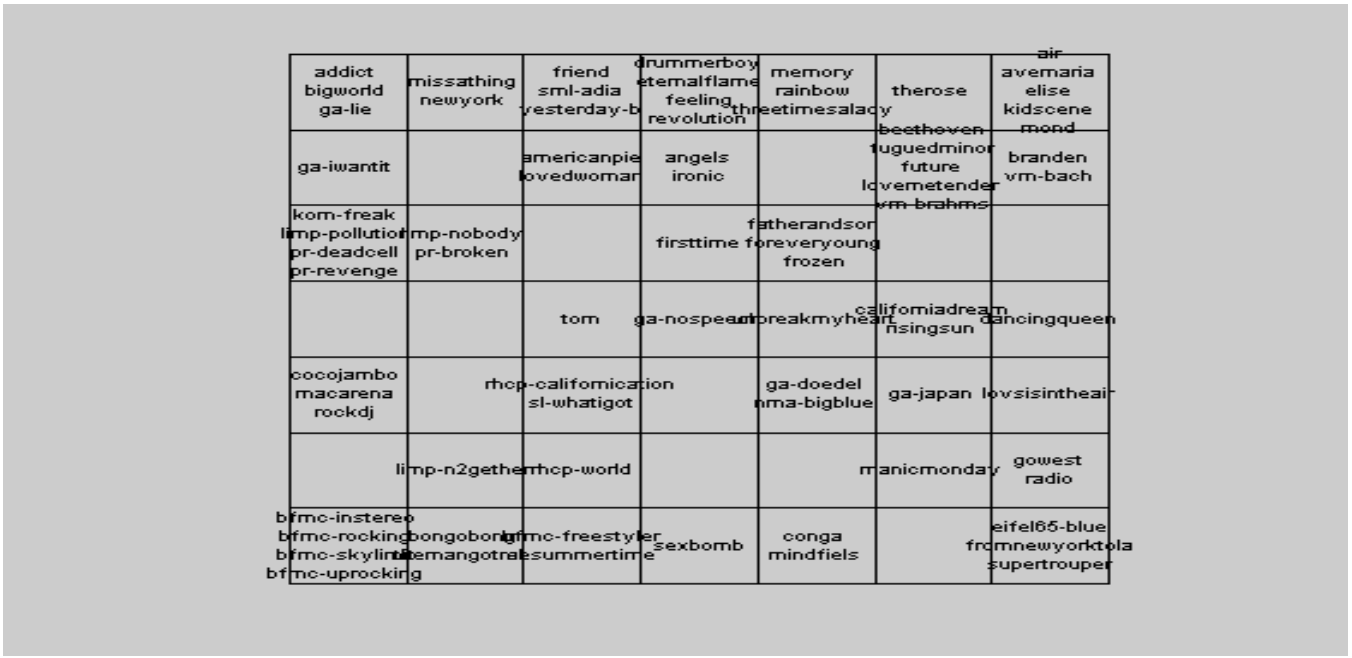

Figure 12. SDH Results after applying SOM algorithm on a dataset of 77 songs from various genres

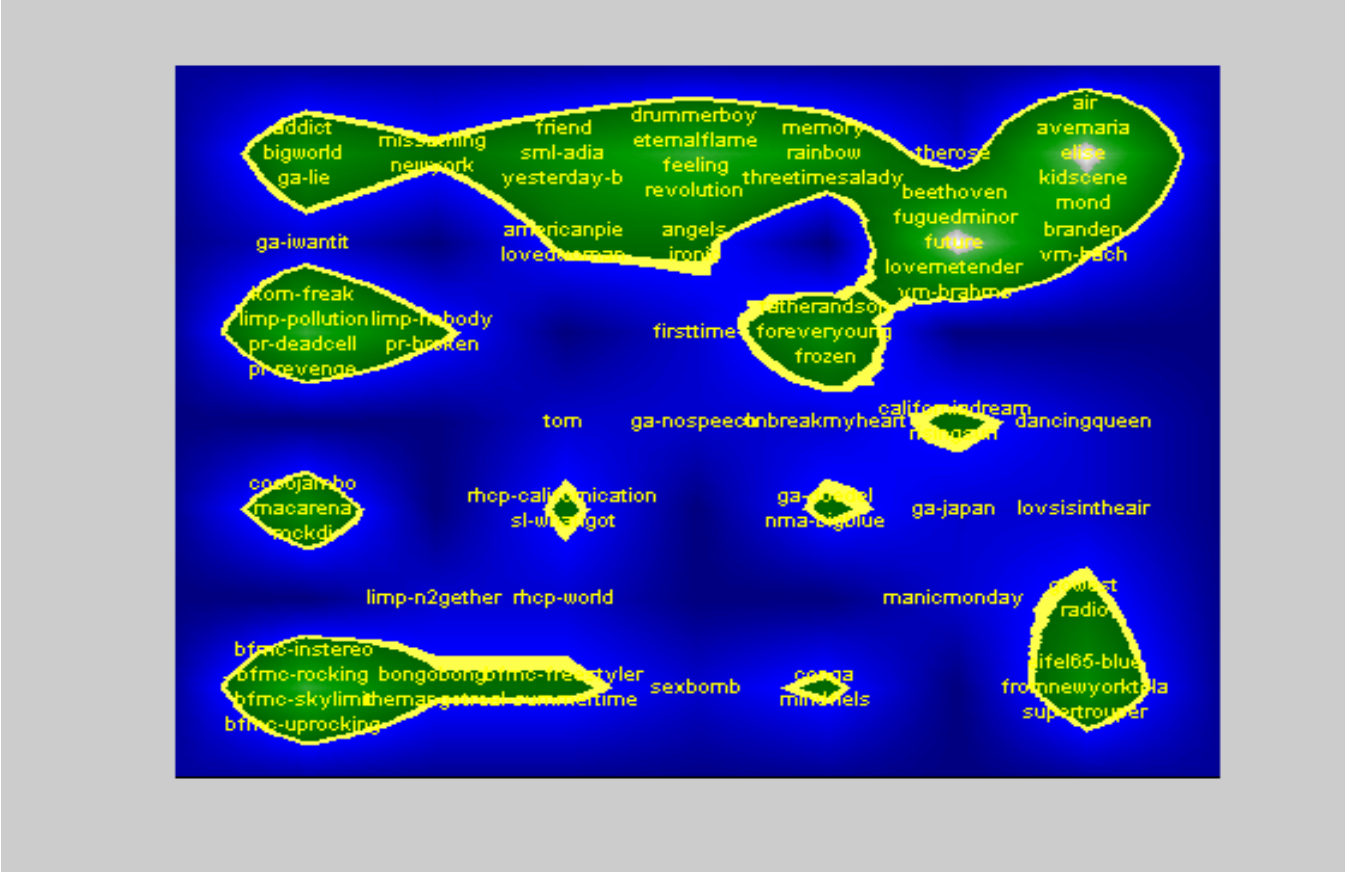

Figure 13. Final Island of Music for the test database of 77 songs

The density function is estimated using the technique presented in the context of the k-means algorithm. Each piece of music votes for the clusters (map units), which represents it best. The first closest model vector of a corresponding unit gets n points, the second n - 1, the third n - 2, and so forth. Thus units, which are close to several pieces of music, will get many points. While clusters, which are not close to any pieces, will hardly have any points. A map unit which is close to many pieces of music is very likely to be close to the center of a cluster, whereas a unit, which does not represent any pieces of music well, is likely to be an intermediate unit between clusters.

Usually one could expect that the second best matching unit is located right next to the first best matching. Thus dividing the hit response of a data item 2/3 for the best matching and 1/3 for the second best matching would lead to the same results with some fuzziness. The often non-spherical shape of the clusters (islands) becomes more apparent and seemingly separated clusters might get connected.

Note that it is not always the case that the first and second best matching units lay next to each other. In fact some quality measures to compare trained SOMs have been developed upon these criteria. However, it is rather unlikely that the two units are separated completely on the map and mostly they will both be located in the same map area. Using $n = 3$ connects more islands and lets them grow bigger. For $n = 4$ the differences to $n = 3$ are not very obvious, however, the islands are slightly more connected and the higher n gets, the more islands will become connected until finally only one big island remains with its peak around the center of the map.

This visualization has been implemented using interpolating units. Between each row and column units have been inserted and around the whole map as well. These interpolating units do not have a corresponding model vector and are only assigned interpolated values of the approximated density function.

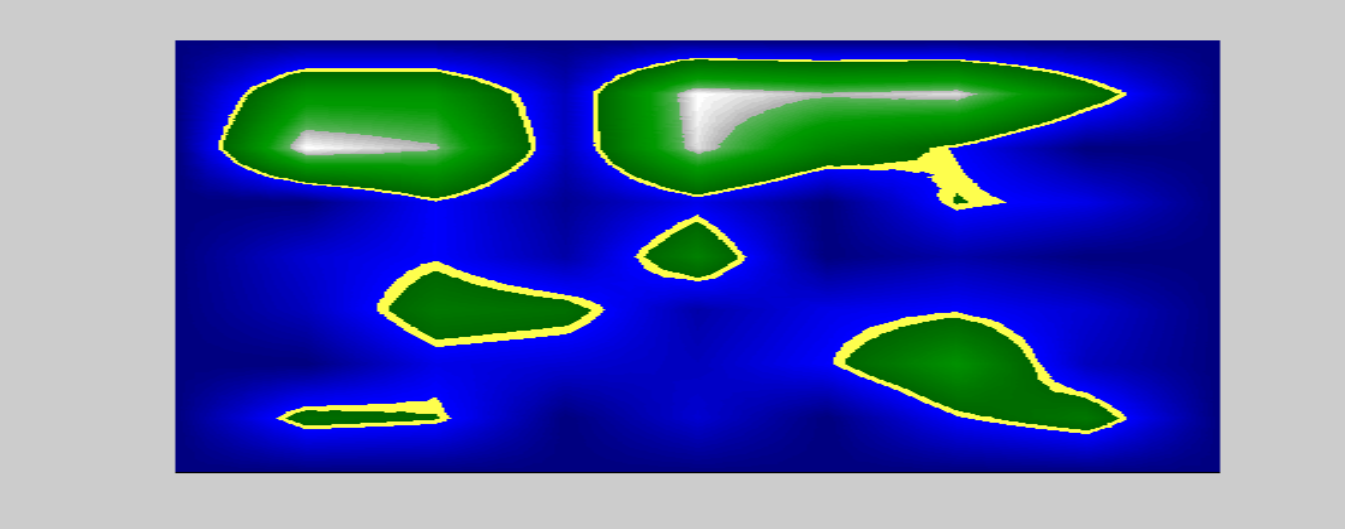

Figure 14. Island of Map with Voting N=3

The units inserted around the map are inserted so that the islands are bounded by the map area. The values of these boarder units are set to 1/10 of their immediate neighbor within the map, and thus are always under water. All density and interpolated density values are then interpolated by Matlab using the pcolor function in combination with shading interp to create the islands of Figure above. If desired it would be possible to use fractal algorithms or textures to create more naturally looking islands.

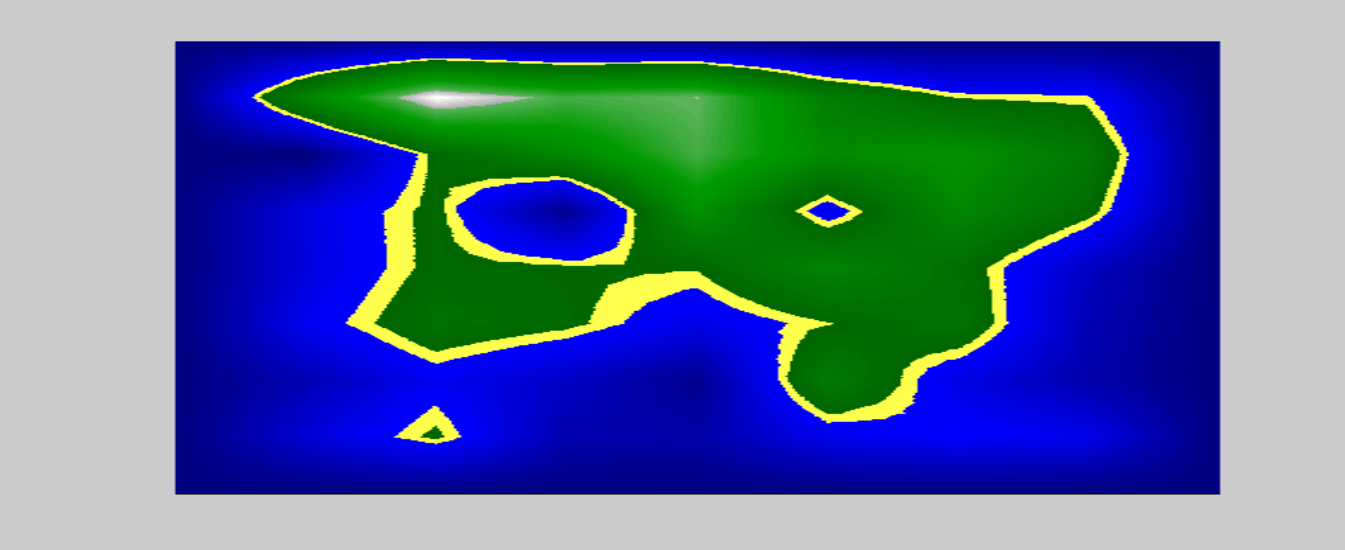

Figure 15. Island of Map with Voting N=10

Chapter 6
Conclusion

This final chapter summarizes the work presented in this thesis. In addition, opportunities for future research are pointed out and briefly discussed.

6.1. Summary

In this thesis models and techniques from the fields of signal processing, psychoacoustics, image processing, and data mining were combined to develop a system which automatically builds a graphical user interface to music archives, given only the raw music collection with no further information, such as the genres to which the pieces of music belong.

The most challenging part is to compute the perceived similarity of two pieces of music. Even though currently no final solution to this can be offered, a novel and straightforward approach based on psychoacoustic models is presented and evaluated using a collection of 77 pieces of music. Despite being far from perfect, this approach yields encouraging results.

A neural network algorithm, namely the self-organizing map, is applied to organize the pieces of music so that similar pieces are located close together on a 2-dimensional map display. A novel visualization technique is applied to obtain the map of islands, where the islands represent clusters in the data. To support navigation in unknown music collections, methods to label landmarks, such as mountains or hills, with descriptions of the rhythmic and other properties of the music located in the respective area, are presented. The Islands of Music have not yet reached a level, which would suggest their commercial usage; however, they demonstrate the possibility of such a system and serve well as a tool for further research. Any part of the system can be modified or replaced and the resulting effects can easily be evaluated using the graphical user interface.

6.2. Future Work

Much research is being conducted in the area of content-based music analysis with new results being published frequently. Incorporating these results into the presented system would increase the quality of the Islands of Music.

Based on the approach presented in this thesis there are some immediate possibilities that might yield improvements. One major problem is the loudness of the pieces of music. Most pieces in the presented collection are from different sources with significant differences regarding the recorded loudness. The loudness has direct impact on the perceived beats and thus strong influence on the whole system. Methods to normalize the loudness would increase the quality.

Another interesting aspect is the sequencing. Each piece of music is divided into 6-second sequences and only a small subset of these sequences is further analyzed to reduce the computational load. Using more sequences or even overlapping them would result in more accurate representations of the pieces of music. Furthermore, the optimal length of the sequences is not yet decided. When calculating the amplitude modulation less then half of the obtained FFT coefficients are used, in particular those which correspond to the modulation frequencies below 10Hz. Thus it would be possible to reduce the length of the sequences to

3 seconds without modifying any other parts of the system. The advantage of a shorter sequence is that it is less likely that it contains more than one specific style.

The applied image processing filters, which emphasize important aspects in the fluctuation strength images, need to be evaluated more thoroughly as well. Alternative parameter settings as well as alternative filters should be considered. In this report several methods to represent a piece of music based on the representation of its sequences were discussed. The finally chosen method, the median, does not coincide with intuitive assumptions; however the alternatives presented were not able to produce significantly better results. A thorough analysis is necessary and perhaps a method, which combines the advantages of the median with the advantages of the other methods presented, could be developed.

Depending on the dataset alternative ways to label the mountains and hills on the islands could be developed which better help in understanding the genres they represent. It is unlikely that such improved labels can be derived directly from the modified fluctuation strength (MFS) data thus the incorporation of different content-based approaches to analyze music is desirable.

Chapter 7
References

[1] F. Nack and A. Lindsay. Everything you wanted to know about MPEG7 - Part 1. *IEEE MultiMedia*, pages 65–77, July/September 1999.

[2] G. Peeters, S. McAdams, and P. Herrera. Instrument Sound Description in the Context of MPEG-7. In *Proceedings of the ICMC2000*, 2000.

[3] A. Rauber and D. Merkl. The SOMLib Digital Library System. In *Proceedings of the 3rd Europ. Conf. On Research and Advanced Technology for Digital Libraries (ECDL'99)*, Paris, France, 1999. Lecture Notes in Computer Science (LNCS 1696), Springer.

[4] S. Kaski, T. Honkela, K. Lagus, and T. Kohonen. WEBSOM-self organizing maps of document collections. *Neurocomputing*, 21:101–117, 1998.

[5] J. Foote. An overview of audio information retrieval. *Multimedia Systems*, 7(1):2–10, 1999.

[6] M. Liu and C. Wan. A Study of Content-Based Classification and Retrieval of Audio Database. In *Proceedings of the 5th International Database Engineering and Applications Symposium (IDEAS'2001)*, Grenoble, France, 2001. IEEE.

[7] J. T. Foote. Content-based retrieval of music and audio. In C. Kuo, editor, *Proceedings of SPIE Multimedia Storage and Archiving Systems II*, volume 3229, pages 138–147, 1997.

[8] B. Logan. Mel Frequency Cepstral Coefficients for Music Modelling. In *International Symposium on Music Information Retrieval (MUSIC IR 2000)*, Plymouth, Massachusetts, 2000.

[9] E. D. Scheirer. *Music-Listening Systems*. PhD thesis, MIT Media Laboratory, 2000.

[10] M. Fruhwirth and A. Rauber. Self-Organizing Maps for Content-Based Music Clustering. In *Proceedings of the 12th Italien Workshop on Neural Nets (WIRN01)*, Vietri sul Mare, Italy, 2001. Springer.

[11] T. Kohonen. *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Springer, Berlin, 3rd edition, 2001.

[12] E. Zwicker and H. Fastl. *Psychoacoustics, Facts and Models*, volume 22 of *Springer Series of Information Sciences*. Springer, Berlin, 2nd updated edition, 1999.

[13] H. Fastl. Fluctuation strength and temporal masking patterns of amplitude-modulated broad-band noise. *Hearing Research*, 8:59–69, 1982.

[14] E. Zwicker, G. Flottorp, and S. S. Stevens. Critical band width in loudness summation. *Journal of the Acoustical Society of America*, 29:548–557, 1957.

[15] E. O. Brigham. *The Fast Fourier Transform*. Prentice Hall, Englewood Cliffs, NJ, 1974.

[16] R. Bladon. Modeling the judgment of vowel quality differences. *Journal of the Acoustical Society of America*, 69:1414–1422, 1981.

[17] S. Weil. Music Analysis and Characterization. Student Project (CS 152 - Neural Networks), Harvey Mudd College, Claremont, CA. http://www.newdream.net/~sage/nn, 1999.

[18] D. P. W. Ellis. *Prediction-Driven Computational Auditory Scene Analysis*. PhD thesis, MIT Department of Electrical Engineering and Computer Science, Cambridge, MA, 1996.

[19] A. K. Jain, M. N. Murty, and P. J. Flynn. Data Clustering: A Review.*ACM Computing Surveys*, 31(3):264–323, 1999.

[20] T. Kohonen. Self-organizing formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69, 1982.

[21] C. M. Bishop, M. Svens´en, and C. K. I. Williams. GTM: A principled alternative to the Self-Organizing Map. *Proceedings of ICANN'96, International Conference on Artificial Neural Networks*, volume 1112 of *Lecture Notes in Computer Science*, pages 165–170, Berlin, 1996. Springer.

[22] J. W. Sammon. A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, 18:401–409, 1969.

[23] T. Kohonen. New developments of learning vector quantization and the self-organizing map. In *SYNAPSE'92, Symposium on Neural Networks*, Osaka, Japan, 1992. Alliances and Perspectives in Senri.

[24] D. Hearn and M. P. Baker. *Computer Graphics, C Version*. Prentice Hall, NJ, 2nd edition, 1997.