Detecting key features in popular music: case study – voice detection

Rui Nóbrega
CITI, Faculdade de Ciências e Tecnologia,
Universidade Nova de Lisboa,
2829-516 Caparica, Portugal
rui.nobrega@di.fct.unl.pt
nº 29600

ABSTRACT
Detecting distinct features in modern pop music is an important problem that can have significant applications in areas such as multimedia entertainment. They can be used, for example, to give a visually coherent representation of the sound. The work developed for this project is meant to be used in the context of a multimedia, multi-touch game where the user has to perform simple tasks at the rhythm of the music. The ultimate goal of this project is to make the creation of visual content automatic using as input, features extracted from the music sound. In this work special focus will be given to the detection of voice segments inside music songs. The solution presented extracts the MFCC's of the sound and uses a Hidden Markov Model to infer if the sound has voice or not. Using this method some results are presented with the best parameters to extract this feature.

Author Keywords
Singing voice detection, features detection, music game, visual sound synchronization, MFCC, Hidden Markov Model

INTRODUCTION
The sound component plays a vital part in today's interactive games and applications. Entertainment applications with good graphics lose their appeal when the sound is not present or is poorly integrated. Having this in mind a prototype game was built where the user has to do multi-touch gestures as shown in examples. The application has background music and the examples appear when there's a beat, a voice starts singing or the guitar starts. This synchronization between the examples and the music is currently done manually. It would be interesting to make this task automatic for any song. To automate this it is necessary to detect several features from the sound such as voice, rhythm, instruments playing, loudness or pitch and see how they change through time.

The main goal of this work is to detect when someone is singing in a song. Specific voice detection in music is not a widely studied subject, but most of its ideas come from speech recognition topics which is a well established research area.

Voice segment detection has applications in singer identification, music information retrieval, music transcription or karaoke lyric alignment.

The human voice ranges from 80 Hz to 400 Hz for male and female speech[5]. In singing, the voice can achieve 1400 Hz. The initial research was done in female voices, in pop-rock songs using short music segments with only one voice at a time.

A prototype was built using speech recognition strategies to detect voice segments in small music samples. The output of the application is the set of ranges where voice was detected. The application uses a training algorithm to detect the segments. To train the application a set of sounds was gathered, half of them contain voices. Mel-Frequency Cepstrum Coefficients (MFCC's) are extracted from these sounds and used to train a Hidden Markov Model (HMM). This model is then used to classify consecutive small blocks of the sound. To evaluate the solution the detected blocks were compared with previously manually classified songs.

In the next section there is some literature review. Next is the description of the solution and in the end the results are presented and discussed.

RELATED WORK
This project was born from the experiences described in [8] where several prototypes where created to explore multi-touch interaction. One of the prototypes is the music game described in the introduction. This game is inspired in the online flash game UpBeat[14] where the player has to follow sequences of movements according to specific rhythms. Another inspiring work is the KeepOn[3] robot which can detected the rhythm of the song and dance to it.

There is extensive research in features detection and audio classification, as an example, in [2] it is presented a method to detect harmonic change in musical audio based on pitch differences between two consecutive sound frames. Several features sets for audio classification are summarized in [1] including what the authors call Low-level signal parameters such as zero-crossing rate, band energy or pitch; MFCC; Psycoacoustic features such as roughness, loudness or sharpness and finally Auditory filterbank temporal

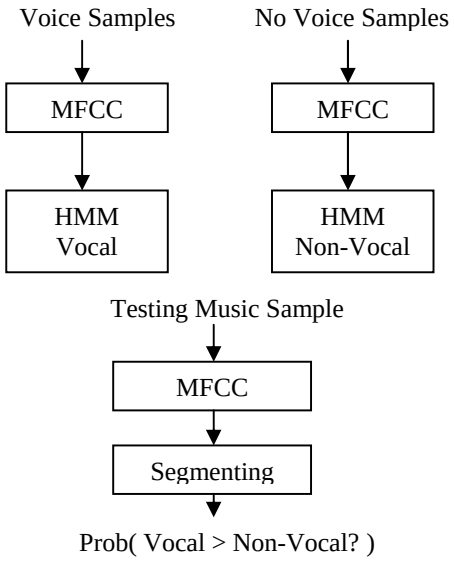

Figure 1. System solution diagram. It initially trains two HMMs and uses them to classify music samples

envelopes. These features are used to train classifiers to detect a wide range of sounds.

Papers [4,5,9] present strategies to detect singing in songs. In [4] four acoustic features are defined and analyzed: vibrato, harmonic, timbre and cepstral coefficient computation such as MFCC. The first three features are used essentially as a cue for where is it more likely to be a voice segment. The MFCC of voice classified songs are used to train a HMM. The final model detects vocal segments in songs which are then inserted back in the HMM. This work presents a generic model and some ideas but lacks a complete solution description. One of its main contributions are its experiment results which will be used in the evaluation section.

Using a very similar strategy paper [9] presents several experiments in singing voice detection but with different cepstral coefficients. In [4] experiments where made with Octave Frequency Cepstral Coefficients (OFCC) and MFCC, with no major differences in the two methods (80% success), but slightly better results when combined (83% success). In [9], four types of coefficients were tested; the Harmonic Attenuated Log Frequency Power Coefficients (HA-LFPC) obtained the best results (86.7%). These coefficients are obtained by using a triangular bandpass filter on the spectrogram which reduces harmonic sounds. This way non-vocal sounds will have much less energy than vocal sounds. It must be stated that the authors do not specify what type of music instruments are being used in their tests, probably most of them have harmonic sounds, thus making this method have a larger success rate. In the same tests MFCC's obtained an average 81.3% of success.

A complete solution to separate singing voice from music is presented in [5]. This is actually a harder problem than just voice detection. Their system has three steps, first the voice segments are detected, then the predominant pitch is detected and finally the voice is separated from the rest of the music. For the purpose of this work the only important part is the voice detection.

The singing voice detection has several stages. First of all a spectral change detector selects several segments of audio where the energy has significant spectral changes between frames. After the input is portioned it is classified as vocal or not by a HMM likelihood function. This method appears to be very similar to the ones presented in [4,9] but has additional details about implementation issues. For this reason, this was the main paper followed in this project. The method proposed has a HMM with two classes c_v and c_{nv},one for vocal and other for non-vocal sounds. Each class is trained separately, c_v with voice sounds and c_{nv} with music sounds. For each sound the MFCC are used as the feature vector. The classes are trained using these MFCC as input for the HMM with gaussian mixture model (GMM). To classify a given sound segment the HMM likelihood function value is obtained for each class. It will be classified as a voice segment if the likelihood of c_v is higher than the one of c_{nv}. The success rate of this method is around 79%. Note that the evaluation process used the same samples that served as input for the training of the HMM, this might have produced biased results. Papers [4,9] do not state the origin of the samples.

Additional related work about MFCC can be found in [12,13] and more about HMM can be found in [7,10,11]. These topics will be further detailed in the next sections.

TRAINING ALGORITHM

Following the models of [4,9] and the implementation described in [5] a system was developed which uses two HMM for voice and non-voice music segments. The MFCCs of two groups of training sounds are taken as input for the HMM. Final segment testing is done comparing the log-likelihood probability of the two HMM. The overall solution can be seen in figure 1.

MEL-FREQUENCY CEPSTRUM COEFICIENTS

MFCCs are the coefficients that capture the perceptual information of sound. They have many application in sound retrieval in genre classification, audio similarity measure or speech recognition.

The MFCCs are based on the mel-sale which is a perceptual scale of pitches. This scale is based on the fact that the correlation between human perceive pitch distance between two sounds is not linear in frequency. It is easier for a person to distinguish two low frequency sounds such as 300Hz and 400Hz than it is to distinguish between 6000Hz and 6100Hz. The mel-scale is a logarithmic function which can be seen in figure 2.

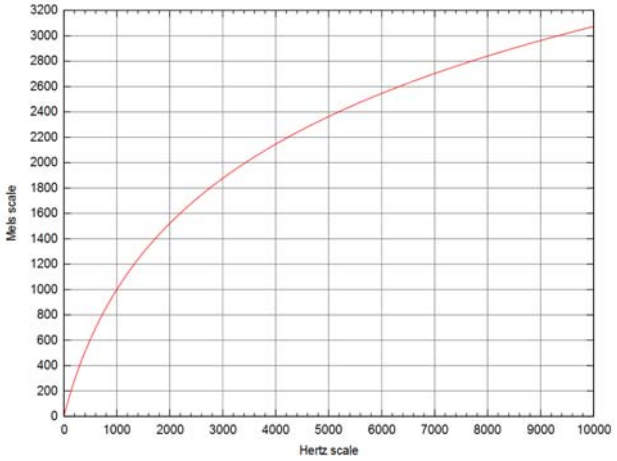

Figure 2. Mel scale.

To extract the MFCC of a sound segment several steps have to be done. In [12] several implementations of MFCC are explained, tested and compared. In this work the implementation used was the Auditory Toolbox by Slaney [13]. The most common extraction methods follow these steps:

1. Separation of the signal in several frames.
2. Find the Fast Fourier Tranform(FFT) of the frame.
3. Multiply the frequencies by a Mel filter bank which is composed of log-spaced triangular overlapping windows as seen in figure 3.
4. Take the log of the powers at each mel frequency.(fig 4)
5. Find the Discreat Cosine Transform (DCT) to the result to reduce dimensionality.
6. The MFCCs are the amplitudes of the resulting spectrum.

The Mel Filter bank used in the Auditory Toolbox is constructed using 13 linearly-spaced filters (133.33Hz between center frequencies,) followed by 27 log-spaced filters (separated by a factor of 1.0711703 in frequency). The center frequencies of the filter bank triangles are computed by approximating the Mel scale with:

$$\phi = 2595 \log_{10}\left(\frac{f}{700} + 1\right)$$

The amplitude of the FFT bin is then combined with the triangular filters seen in figure 3.

The final result is for each frame, 13 coefficients, which can be drawn consecutively as seen in figure 4, in a cepstrum. The top row of the cepstrum, known as C0 has a different aspect because is a function of the power in the signal and has a negative value. The other coefficients, C1-C12, have values close to 0. These are the values that will be used in the HMM.

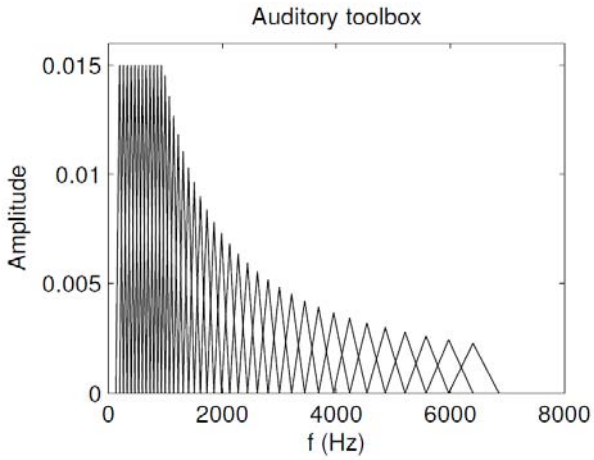

Figure 3. Auditory Toolbox implementation of the Mel filter bank [12].

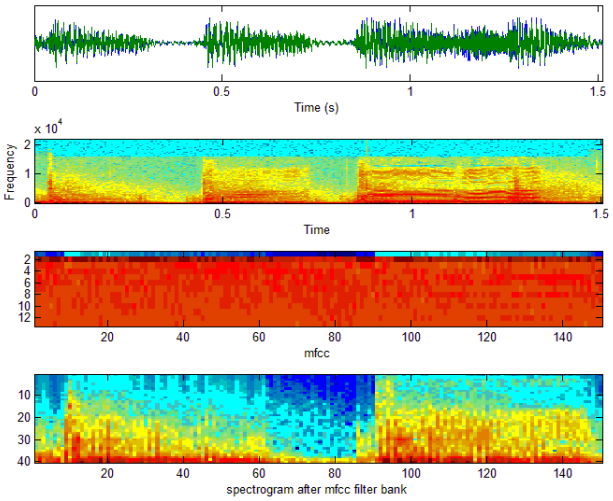

Figure 4. Mel-Frequency Cepstral Coefficients can be seen in the third graph.

HIDDEN MARKOV MODEL

Hidden Markov Model is a type of stochastic signal model which has many applications such as recognition of speech, handwriting or musical score. In this work it will be used for singing voice recognition. Russel and Norvig in their book [11] dedicate an entire chapter to statistical methods and probabilistic reasoning over time. The chapter explains thoroughly how HMM work and relates them with Bayesian networks. It also has some comprehensible algorithms that implement the model.

Another important effort is the work done by Rabiner in [10] where a complete tutorial for HMMs is presented and some applications of them to speech recognition are discussed. He starts by reviewing the theory of Markov chains and then builds several examples to extend the ideas to hidden Markov models. The main focus of the work is to explain three problems of the HMM design: the evaluation

of the probability (likelihood) of a sequence of observations given a specific HMM; the determination of a best sequence of model states; and the adjustment of model parameters to best account for the observed signal.

To better understand HMM it is necessary to first review the Markov chains for discrete Markov processes. A Markov model is composed of a oriented graph of *N* states where all states are connected with a transition matrix *A* and there is an initial state *I* at *t=1*. Has an example consider a 3-state Markov model for weather. Imagining the states are:

- State 1: rain
- State 2: cloudy
- State 3: sunny

Considering the transition probability from one state to the other to be the 3 by 3 matrix *A*:

$$A = \{a_{ij}\} = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$$

This means that the probability of having a cloudy (state 2) event after a rain (state 1) event is:

$$a_{12} = 0.3$$

Given that the weather on day 1(t=1) is sunny (state 3), with the Markov model the we can ask the following question: *What is the probability that the weather for the next 7 days will be "sun-sun-rain-rain-sun-cloudy-sun"?* This corresponds to an observation sequence *O*:

$$O = \{S_3, S_3, S_1, S_1, S_3, S_2, S_3\}, t = 1,2,...,7$$

This translates into the following probability:

$$\begin{aligned} P(O|\text{Model}) &= P(S_3, S_3, S_1, S_1, S_3, S_2, S_3 \mid \text{Model}) \\ &= P(S_3) \cdot P(S_3|S_3) \cdot P(S_3|S_3) \cdot P(S_1|S_3) \\ &\quad \cdot P(S_1|S_3) \cdot P(S_3|S_3) \cdot P(S_2|S_3) \cdot P(S_3|S_3) \\ &= 1 \cdot 0.8 \cdot 0.8 \cdot 0.1 \cdot 0.4 \cdot 0.3 \cdot 0.1 \cdot 0.2 \\ &= 1.536 \times 10^{-4} \end{aligned}$$

The above example was a simple Markov model in which each state corresponded to a physical observable event. This may not be always possible in all problems. In a regular Markov model, the state is directly visible to the observer, and therefore the state transition probabilities are the only parameters. In a hidden Markov model, the state is not directly visible; the only thing that the observer sees is the sequence of observations produced by a *hidden* process.

As an example of a HMM consider a coin tossing example. In this scenario we have a room with a curtain. The observer sits on one side of the curtain and another person is behind the curtain tossing one or multiple coins. He will tell the result of each coin flip. The observer never knows how many coins are being tossed. He only knows the sequence *O*:

$$O = \{O_1, O_2, ..., O_n\}$$

"heads-heads-tails-tails-tails-heads-tails-tails-heads…"

Given the above scenario, the problem is how to build an HMM to explain the observed sequence of heads and tails. Since the observer doesn't know how many coins are being tossed, he doesn't know how many states there is. For example, for 2 coins, there would be 4 states corresponding to the all the combinations of heads and tails (2^2). In a HMM the main problems are deciding what the states in the model correspond to and how many states should be in the model.

In all these examples we considered only the case where the observations where discrete symbols from a finite alphabet. In the coin tossing problem the symbols where just heads and tails. This is not always true in all problems. In some applications, such as signal processing, the observations are continuous. Although it is possible to quantify the signal it is better to describe it using continuous observation densities. One of the most common are the Gaussian Mixture model, where the probability of each state can be described as a set of *M* Gaussian probability density functions.

One important factor about HMM is that they can be "trained". Instead of giving a transition matrix *A* it is possible to infer *A* from a set of observed states. This is done iteratively with an algorithm that tries to optimize the parameters of the model until there is a combination that best describes the observed pattern. This last property is very important because it is possible to train a HMM with a set of voice sounds, and then use the model obtained to detected if another sound has voice.

This was just a glimpse of what HMMs can do. For implementation details and further reading please refer to [10,11]. The HMM Toolbox [7] is a complete implementation of HMM with many related algorithms and solutions.

SOLUTION

To adapt HMM for singing voice recognition a prototype was implemented using MatLab following the ideas described in [5] (see related work). Two classes of HMM were created, one for singing voice and other for instrument only sound. Following the diagram in figure 1 there were several steps for the implementation.

The first step was to create a database of sounds. Several two second samples were taken from different songs and from different artists. Half of the samples had voice. The samples had a quality of 44100Hz, 16 bits, Mono. The entire algorithm works only on a single channel but would be trivial to replicate it for stereo sound.

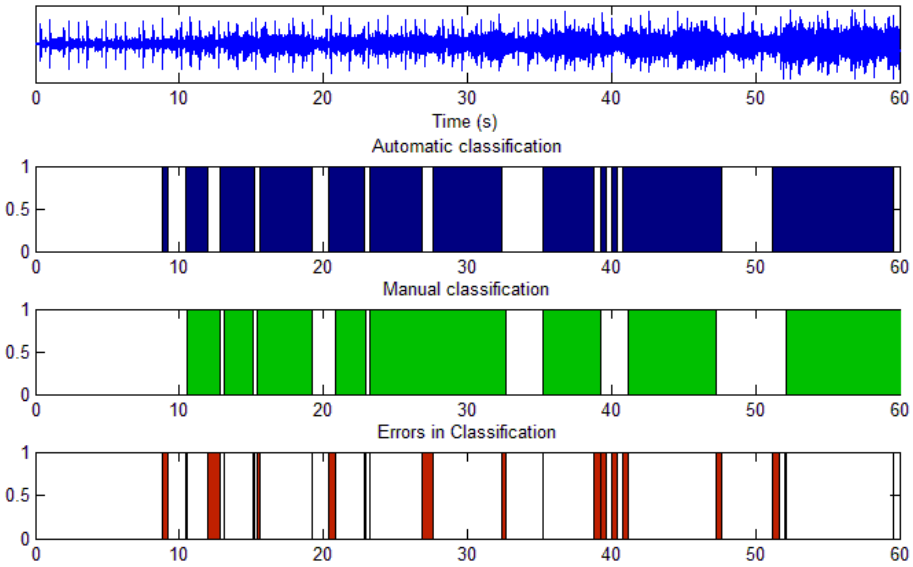

Figure 5. Singing voice classification. The dark areas in the second graph represent voice detect by the algorithm. The third graph shows a manual voice classification done by a person. The last graph shows the differences. In this 60 second example the success rate was 90%.

After reading the samples to the system, the second step was to divide the samples in small frames and convert them to MFCC vectors using the Auditory Toolbox[13]. The samples were divided in 100 frames per second. For each frame a vector of 13 Mel-frequency cepstral coefficients was obtained, thus resulting in a matrix *D* of 13 by 200, for each two second sample.

The third step was the training of the two HMMs using the MFCCs. To implement the HMMs the prototype uses the already referred HMM Toolbox [7]. The HMMs were defined using as input *D*. Each column of *D* is considered an observation and is composed by thirteen coefficients. Since sound is a continuous signal the possible values of the coefficients are not discreet symbols of a finite alphabet, for that reason a Gaussian Mixture model with *M* mixtures to represent *Q* possible states. *M* and *Q* are left open as parameters. Finally the HMM is generated using an approximation process described in the last section with 1000 iterations.

Having the two HMMs constructed the fourth step is to evaluate a music and detect voice segments. The main idea is to read the music and convert it into MFCC vectors. Then take the resulting 13 by *n* matrix and divide it into segments of size *S* (the S used was 40 in all tests). Then use this data to compute the likelihood probability of it belonging to the vocal HMM class or non vocal. The segment will be classified as belonging to the class with higher probability.

EVATUATION AND RESULTS

To evaluate the method several samples were manually classified with voice and no voice segments. To do this in real time, a small application was built where a person would listen to a song and would push a button whenever the singer started and ended singing. This is a very fast method to obtain classified data although it has some errors and possible delays associated with the person's reaction time. Using the classified songs it was possible to make comparisons between the proposed algorithm results and the manual result resulting in the graph seen in figure 5.

Several tests where made changing several parameters of the HMMs, first of all the HMMs where trained using only samples from the same band of the music that it would be tested. Then they were trained using all the samples. Finally the parameters: *M* the number of Gaussian mixtures and *Q* the number of states, were tested with different values. The success rate of each approach can be seen in table 1.

%		***Q***			
Train Samples	***M***	1	4	8	16
Same band from test	2	82.4	82.3	81.2	80.8
	10	83.1	82.8	77	76.6
All bands	2	80.3	80.4	81.4	76.4
	10	81	80.1	78.1	79.4

Table 1. Success rate with different parameters.

The results had a variation of around ±5%. The tests had a slightly better result when the training set is composed of samples from the songs being tested. The other group of samples also had samples from the song being played but

had also more different music bands. For this reason the transition probabilities probably become more diluted in the HMM resulting in worst results. The best results are obtained using fewer states and fewer Gaussian Mixtures. There are not large differences between the results but it is safe to say that using between 1 and 4 states with 2 to 10 Gaussian Mixtures is a good parameter solution.

CONCLUSION

With this work it is possible to conclude that a solution based on HMM using MFCCs as classifiers is a valid answer to detect singing voice in music. The results show a high degree of success comparing them with similar results from other authors (around 80% for [4] and [5]). This implementation may need additional tuning to make detections in other types of music but taking into account the scope defined in the introduction the results were satisfactory.

For the future it would be necessary to extend the evaluation with a larger set of samples to train the models and to test the system. One idea would be to feed the results back into the model when the degree of confidence in the system starts getting higher. To improve the system it would be necessary to cross the detected segments using HMMs with other types of detectors based on energy, pitch or harmonic frequency.

Finally using the detected singing sections it is possible to build interactive multimedia applications that react to a given music, as said in the example described in the introduction.

ACKNOLEDGMENTS

The author would like to thank to everyone at IMG for all the support and input. Additional acknowledgments go to CITI at Faculdade de Ciências e Tecnologias da Universidade Nova de Lisboa.

REFERENCES

1. Breebaart, J., McKinney, M.: Features for audio classification. In *Proc. SOIA2002, Philips Symposium on Intelligent Algorithms*, 2002.
2. Harte, C., Sandler, M., and Gasser, M.: Detecting harmonic change in musical audio. In *Proceedings of the 1st ACM Workshop on Audio and Music Computing Multimedia* ,AMCMM '06, 2006
3. Keep On robot, 2009 http://beatbots.org/, http://www.youtube.com/watch?v=3g-yrjh58ms
4. Khine, S.Z.K., Tin Lay New, Haizhou Li,: Singing voice detection in pop songs using co-training algorithm, . In *IEEE International Conference on Acoustics, Speech and Signal Processing, 2008.* ICASSP 2008, 2008
5. Li, Y., Wang, D.: Separation of Singing Voice From Music Accompaniment for Monaural Recordings, In *Audio, Speech, and Language Processing, IEEE Transactions on* , 2007
6. Min Xu, Maddage, N.C.: Changsheng Xu; Kankanhalli, M.; Qi Tian, Creating audio keywords for event detection in soccer video,In *Multimedia and Expo.* ICME '03, 2003
7. Murphy, K., HMM Toolbox for Matlab, 2009 http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html
8. Nóbrega, R., Sabino, A., Rodrigues, A., and Correia, N.: Flood Emergency Interaction and Visualization System. In *Proc. of Visual information Systems: Web-Based Visual information Search and Management* ,VISUAL'08,2008
9. Nwe, T. L., Shenoy, A., Wang, Y.: Singing voice detection in popular music. In *Proceedings of the 12th Annual ACM international Conference on Multimedia*. MULTIMEDIA '04, 2004
10. Rabiner, L. R.: A tutorial on hidden Markov models and selected applications in speech recognition. In *Readings in Speech Recognition*, A. Waibel and K. Lee, Eds. Morgan Kaufmann Publishers, San Francisco, 1990
11. Russel, S., Norvig, P., Artificial Inteligence: A Modern Approach, 2nd Edition, Prentice Hall, Cap15, International Edition, 2003
12. Sigurdsson, S., Petersen,K.B., Lehn-Schiøler, T.: Mel Frequency Cepstral Coefficients: An Evaluation of Robustness of MP3 Encoded Music, In *Proceedings of the Seventh International Conference on Music Information Retrieval*. ISMIR'06, 2006
13. Slaney, M., Auditory Toolbox, 2009 http://cobweb.ecn.purdue.edu/~malcolm/interval/1998-010/
14. Upbeat, music game,2009 http://www.primarygames.com/arcade/upbeat/