DETECTION OF KEY CHANGE IN CLASSICAL PIANO MUSIC

Wei Chai **Barry Vercoe**
MIT Media Laboratory
Cambridge MA, USA
{chaiwei, bv}@media.mit.edu

ABSTRACT

Tonality is an important aspect of musical structure. Detecting key of music is one of the major tasks in tonal analysis and will benefit semantic segmentation of music for indexing and searching. This paper presents an HMM-based approach for segmenting musical signals based on key change and identifying the key of each segment. Classical piano music was used in the experiment. The performance, evaluated by three proposed measures (recall, precision and label accuracy), demonstrates the promise of the method.

Keywords: key detection, music segmentation, Hidden Markov Models.

1 INTRODUCTION

Tonality is an important aspect of musical structure. It describes the relationships between the elements of melody and harmony. Detecting key of music is one of the major tasks in tonal analysis. Developing computational models to mimic the perception and detection of key will help automate the analysis of development of musical themes and emotion.

From the practical perspective, semantic segmentation of music, including segmentation based on key change, will benefit intelligent music editing systems and automatic indexing of music repository.

Furthermore, detection of key is a critical step for finding repeated patterns in music for music indexing and searching. For example, Foote [1] proposed a representation called self-similarity matrix for analyzing the recurrent structure of music, where a repetition typically will result in a diagonal pattern in the self-similarity matrix. However, if a theme repeats at a different key, without considering the key change, the diagonal pattern will not appear and the repetition will not be detected. For example, Figure 1 shows the self-similarity matrix zoomed in at a repetition of the theme at a different key in Mozart's piano sonata. The diagonal pattern could not be seen from the original self-similarity matrix representation without considering key change. However, if know the key change in advance and adjust accordingly when comparing two frequency vectors to get the self-similarity matrix, the diagonal pattern will come out.

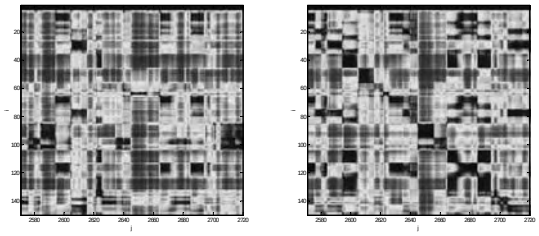

Figure 1. Zoom in of the last repetition in "Mozart: Piano Sonata No. 15 In C" (left: original self-similarity matrix; right: key-adjusted self-similarity matrix).

This paper presents an HMM-based generative model for automatic key detection of music. Specifically, given a musical piece (or part of it), the system will segment it into sections based on key change and identify the key of each section. Please note that here we want to segment the piece and identify the key of each segment at the same time. A simpler task could be, given a segment of a particular key, detecting the key of it. In fact, previous research on key detection of acoustic musical signals typically assumes that the musical segment remains the same key, so that the algorithm can analyze the pitch profile of the segment to infer the key [2,3,4].

Another related work was done by Sheh [5], who investigated a similar problem of segmenting musical signals based on chord change and identifying the chord of each segment, where EM-trained Hidden Markov Models were employed.

The remainder of this paper is organized as follows. Section 2 gives a brief introduction of musical key and other relevant terms. Section 3 presents the chromagram representation and the framework of using Hidden Markov Models (HMMs) for detecting key change. Section 4 demonstrates the promise of the method by experiments using classical piano music and some evaluation metrics. Section 5 concludes the paper and proposes future work.

2 MUSICAL KEY AND MODULATION

In Music theory, the **key** is the tonal center of a piece. It can be either in major or minor mode. A **scale** is an ascending or descending series of notes or pitches The **chromatic scale** is a musical scale that contains all twelve pitches of the Western tempered scale. The **diatonic scale** is most familiar as the major scale or the "natural" minor scale. The major mode has half-steps

between scale steps 3 and 4, and 7 and 8. The natural minor mode has half-steps between 2 and 3, and 5 and 6.

A piece may change key at some point. This is called **modulation**. Modulation to the dominant (a fifth above the original key) or the subdominant (a fourth above) is relatively easy, as are modulations to the relative major of a minor key or to the relative minor of a major key. A thing needs to mention is that there might be ambiguity of key. It can be hard to determine the key of a quite long passage. Some music is even **atonal**, meaning there is no tonal center. Thus, in this paper, we will focus on tonal music with least ambuiguity of tonal center.

3 APPROACH

This section presents an HMM-based approach for detecting key change in classical piano music.

3.1 Chromagram Representation

Chromagram, also called the Pitch Class Profile features (PCP), is a frame-based representation very similar to Short-time Fourier Transform (STFT). It combines the frequency components in STFT belonging to the same pitch class and results in a 12-dimensional representation, corresponding to C, C#, D, D#, E, F, F#, G, G#, A, A#, B in music, or a generalized version of 24-dimensional representation simply for higher resolution. Specifically, for the 24-dimensional representation, let $X_{STFT}[K,n]$ denote the magnitude spectrogram of signal $x[n]$. The chromagram of $x[n]$ is

$$X_{PHP}[K',n] = \sum_{K:\,P(K)=K'} X_{STFT}[K,n] \qquad (1)$$

The mapping between frequency index K in STFT and frequency index K' in PCP is

$$P(K) = [24 \cdot \log_2(K / NFFT \cdot f_s / f_1)] \bmod 24 \qquad (2)$$

where NFFT is the FFT length, f_s is the sampling rate, f_1 is the reference frequency corresponding to a note in the standard tuning system, for example, MIDI note C3 (32.7031956626Hz). In the following, we will use the 24-dimensional PCP representation for better resolution.

In the following, we will focus on the chromagram representation for key analysis of classical piano music, simply because of its advantage of direct mapping to the musical meaning. It doesn't mean it is best for any types of applications or any musical genres. However, all the following approaches should be generalized fairly easily using other representations.

3.2 Parameters and Configuration of HMM

In the following, the task of key detection will be divided into two steps:

1. Detect the key without considering its mode. For example, both C major and A minor will be denoted as key 1, C# major and A# minor will be denoted as key 2, and so on. Thus, there could be 12 different keys in this step.

2. Detect the mode (major or minor).

The task is divided in this way, because diatonic scales are assumed and relative modes share the same diatonic scale. Thus, step 1 attempts to determine the height of the diatonic scale. And again, both steps involve segmentation based on key (mode) change as well as identification of keys (modes).

The model used for key change detection should be able to capture the dynamic of sequences, and to incorporate prior musical knowledge easily since large volume of training data is normally unavailable. Thus, we propose to use Hidden Markov Models for this task, because HMM is a generative model for labelling structured sequence and satisfies both of the above properties [6].

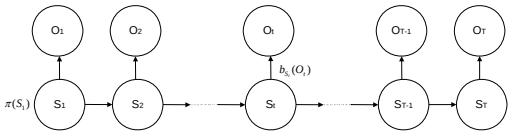

Figure 2. Demonstration of Hidden Markov Models.

Figure 2 shows a graph of HMM used for key change detection. The hidden states correspond to different keys (or modes). The observations correspond to each frame represented as 24-dimensional chromagram vectors. The task will be decoding the underlying sequence of hidden states (keys or modes) from the observation sequence using Viterbi approach.

The parameters of HMM need to be configured include:

- The number of states N corresponding to the number of different keys (=12) or the number of different modes (=2), respectively, in the two steps.
- The state transition probability distribution $\mathbf{A} = \{a_{ij}\}$ corresponding to the probability of changing from key (mode) *i* to key (mode) *j*. Thus, A is a 12×12 matrix (in step 1) and a 2×2 matrix (in step 2), respectively.
- The initial state distribution $\Pi = \{\pi_i\}$ corresponding to the probability at which a piece of music starts from key (mode) *i*.
- The observation probability distribution $\mathbf{B} = \{b_j(v)\}$ corresponding to the probability density at which a chromagram *v* is generated by key (mode) *j*.

Due to the small amount of labeled audio data and the clear musical meanings of the parameters, Π and A were empirically set as follows:

$$\Pi = \frac{1}{12} \cdot \mathbf{1}$$

where $\mathbf{1}$ is a 12-dimensional vector in step 1 and a 2-dimensional vector in step 2. This configuration denotes equal probabilities of starting from different keys (modes).

$$\mathbf{A} = \begin{bmatrix} stayprob & b & ... & b \\ b & stayprob & ... & b \\ b & b & ... & b \\ b & b & ... & stayprob \end{bmatrix}_{d \times d}$$

where d is 12 in step 1 and is 2 in step2. *stayprob* is the probability of staying in the same state and $stayprob + (d-1) \cdot b = 1$. For step 1, this configuration denotes equal probabilities of changing from a key to a different key. It can be easily shown that when *stayprob* gets smaller, the state sequence gets less stable (changes more often). In our experiment, *stayprob* will be varying within a range (e.g., [0.9900 0.9995]) in step 1 and be set to $1-10^{-20}$ in step 2 to see how it impacts the performance.

For observation probability distribution, instead of Gaussian probabilistic models, commonly used for modeling observations of continuous random vectors in HMM, the cosine distances between the observation (the 24-dimensional chromagram vector) and the pre-defined template vectors were used to represent how likely the observation was emitted by the corresponding keys or modes, i.e.,

$$b_j(v) = \frac{v.\theta_j}{\| v \| . \| \theta_j \|} \qquad (3)$$

where θ_j is the template of state j (corresponding to the j[th] key or mode). Note that, strictly speaking, the model using cosine distances is not a probability density, because it does not integrate to 1; however, since we only care about the relative likelihood of being at different keys, it is still a reasonable model.

The advantage of using cosine distance instead of Gaussian distribution is that the key (or mode) is more correlated with the relative amplitudes of different frequency components rather than the absolute values of the amplitudes. Figure 3 shows an example for demonstrating this. Suppose points A, B and C to be three chromagram vectors. Based on musical knowledge, B and C are more likely to be generated by the same key (or, mode) than A and C, because B and C have more similar energy profile. However, if we look at the Euclidean space, A and C are closer to each other than B and C; thus, if we use Gaussian distribution to model the observation probability distribution, A and C will be more likely to be generated by the same key, which is not true.

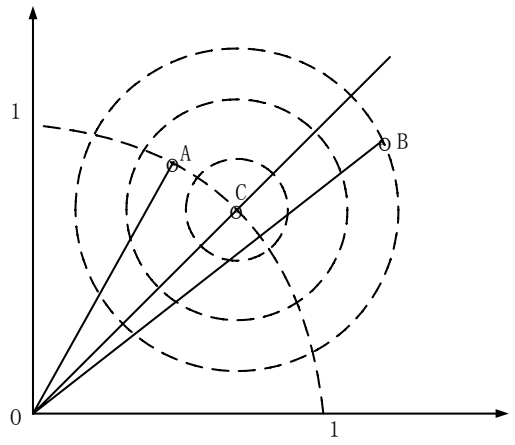

Figure 3. Comparison of observation distributions of Gaussian and cosine distance.

For step 1, the template of a key was empirically set corresponding to the diatonic scale of that key. For example, the template for key 1 (C major or A minor) is $\theta_1^{odd} = [1\,0\,1\,0\,1\,1\,0\,1\,0\,1\,0\,1]^T$ (Figure 4), $\theta_1^{even} = \mathbf{0}$, where θ_1^{odd} denotes sub-vector of θ_1 with odd indexes (i.e., $\theta_1(1:2:23)$) and θ_1^{even} denotes sub-vector of θ_1 with even indexes (i.e., $\theta_1(2:2:24)$). This means we ignore the elements with even indexes when calculating the cosine distance. The templates of other keys were set simply by rotating θ_1 accordingly:

$$\theta_j = r(\theta_1, 2 \cdot (j-1)) \qquad (4)$$

$$\beta = r(\alpha, k), \; s.t. \; \beta[i] = \alpha[(k+i) \bmod 24]$$

where j=1, 2, …, 12 and i, k=1, 2, …, 24. Let us also define $24 \bmod 24 = 24$.

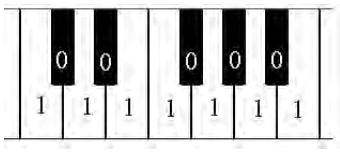

Figure 4. Configuration of the template for C major (or A minor).

For step 2, the templates of modes were empirically set as follows:

$$\theta_{major}^{\;odd} = [0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0]^T,$$

$$\theta_{minor}^{\;odd} = [0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0]^T,$$

$$\theta_{major}^{\ even} = \theta_{minor}^{\ even} = 0,$$

This setting comes from musical knowledge that typically in a major piece, the dominant (G in C major) appears more often than the submediant (A in C major), while in a minor piece, the tonic (A in A minor) appears more often than the subtonic (G in A minor). Please note the templates need to be rotated accordingly (Equation 4) based on its key detected from step 1.

Apparently, the above is a simplified model and there can be several refinements of it. For example, if we consider the prior knowledge of modulation, we can encode in **A** the information that each key tends to change to its "close" keys rather than the other keys. The initial key or mode of a piece may not be uniformly distributed as well. But to quantize the numbers, we will need a very large corpus of pre-labeled musical data, which is not available here.

4 EXPERIMENTAL EVALUATION

4.1 Data Set

Ten classical piano pieces (Table 1) were used in the experiment of key detection, since the chromagram representation of piano music has very clear mapping between its structure and its musical meaning (Section 3.1). These pieces were chosen randomly as long as they have fairly clear tonal structure (relatively tonal instead of atonal). The truth was manually labeled by the author based on the score notation to be compared with the computed results.

The data were mixed into 8-bit mono and down-sampled to 11kHz. Each piece was segmented into frames of 1024 samples with 512 samples overlap.

Table 1. Ten classical piano pieces in the experiment.

1. Mozart: Piano Sonata No. 15 In C (I. Allegro)
2. Schubert: Moment Musical No. 2
3. Dvorak: Humoresque No. 7
4. Rubenstein: Melody In F
5. Paderewski: Menuett
6. Chopin: 'Military' Polonaise
7. Beethoven: Minuet In G
8. Mozart: Sonata No. 11 In A 'Rondo All Turca'
9. Schumann: From Kinderszenen (1. Von Fremden Landern Und Menschen)
10. Chopin: Waltz In D-flat, Op. 64 No. 1 'Minute Waltz'

4.2 Evaluation Measures

To evaluate the results, two aspects need to be considered: label accuracy (how the computed label of each frame is consistent with the actual label) and segmentation accuracy (how the detected locations of transitions are consistent with the actual locations).

Label accuracy is defined as the proportion of frames that are labeled correctly, i.e.,

$$Label\ accuracy = \frac{\#\ frames\ labeled\ correctly}{\#\ total\ \ frames} \quad (5)$$

Two metrics were proposed and used for evaluating segmentation accuracy. *Precision* is defined as the proportion of detected transitions that are relevant. *Recall* is defined as the proportion of relevant transitions detected.

Thus, if B={relevant transitions}, C={detected transitions} and $A = B \cap C$, from the above definition,

$$Precision = \frac{A}{C} \quad (6)$$

$$Recall = \frac{A}{B} \quad (7)$$

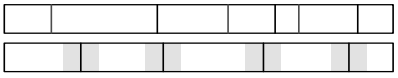

Figure 5. An example for measuring segmentation performance (above: detected transitions; below: relevant transitions).

To compute precision and recall, we need a parameter *w*: whenever a detected transition t_1 is close enough to a relevant transition t_2 such that $|t_1-t_2|<w$, the transitions are deemed identical (a *hit*). Obviously, greater *w* will result in higher precision and recall. In the example shown in Figure 5, the width of each shaded area corresponds to $2w-1$. If a detected transition falls into a shaded area, there is a *hit*. Thus, the precision in this example is 3/6=0.5; the recall is 3/4=0.75. Given *w*, higher precision and recall indicates better performance. In my experiment (512 window step at 11kHz sampling rate), *w* will vary within a range to see how precision and recall vary accordingly: for key detection, *w* varies from 10 frames (~0.46s) to 80 frames (~3.72s). The range of *w* for key detection is fairly large because modulation of music (change from one key to another key) is very often a smooth process that may take several bars.

Assume we randomly segment a piece into $(k+1)$ parts, i.e., *k* random detected transitions. Let *n* be the length of the whole piece (number of frames) and let *m* be the number of frames "close enough" to each relevant transition, i.e., $m=2w-1$. Also assume there are *l* actual segmenting points. To compute average precision and recall of random segmentation, the problem can be categorized as a hyper-geometric distribution: if we choose *k* balls from a box of *ml* black balls (i.e., m black balls corresponding to each segmenting point) and $(n-ml)$ white balls, assuming no overlap occurs, what is the distribution of the number of black balls we get. Thus,

$$\text{Precision} = \frac{\text{E}[\#\, black\ balls\ chosen]}{k} = \frac{1}{k} \cdot \frac{mlk}{n} = \frac{ml}{n} \quad (8)$$

$$\text{Recall} = \frac{E[\#\, detected\ \ segmenting\ \ points]}{l} = \frac{l \cdot P(B>0)}{l}$$
$$= 1 - P(B=0) = 1 - \frac{C_m^0 C_{n-m}^{k-0}}{C_n^k}$$
$$= 1 - (1 - \frac{k}{n})(1 - \frac{k}{n-1})...(1 - \frac{k}{n-m+1}) \quad (9)$$

where B denotes the number of black balls chosen corresponding to a particular segmenting point. If we know the value of l in advance and make $k=l$ (thus, not completely random), and $n>>m$,

$$\text{Recall} \approx 1 - (1 - \frac{l}{n})^m \quad (10)$$

The equations shown that, given n and l, precision increases by increasing w (i.e., increasing m); and recall increases by increasing k *or* w. Equation 8 and 10 will be used later as the baseline (upper bound of the performance of random segmentation) to be compared to the performance of the segmentation algorithm.

4.3 Results

Figure 6 shows key detection result of Mozart's piano sonata No. 11 with *stayprob*=0.996 for step 1 and *stayprob2*=1-.1^{-20} in step 2. The figure above presents the result of key detection without considering mode (step 1) and the figure below presents the result of mode detection (step 2).

To show label accuracy, recall and precision of key detection averaged over all the pieces, we can either fix w and change *stayprob* (Figure 7), or fix *stayprob* and change w (Figure 8).

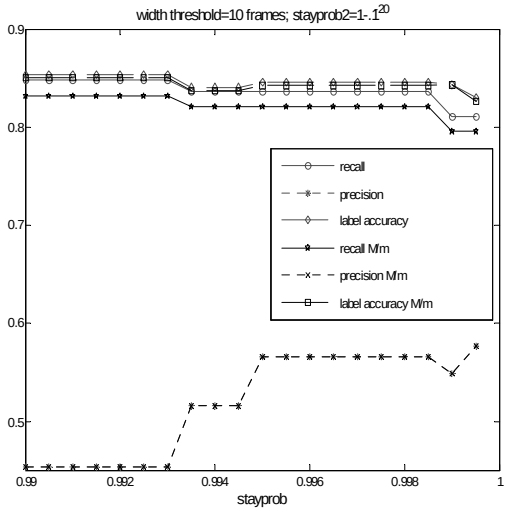

Figure 7. Performance of key detection with varying *stayprob* (w=10; *stayprob2*=1-.1^{-20}).

In Figure 7, two groups of results are shown in the plot: one corresponds to the performance of step 1 without considering modes; the other corresponds to the overall performance of key detection with mode into consideration. It clearly shows that when *stayprob* is increasing, precision is also increasing while recall and label accuracy are decreasing.

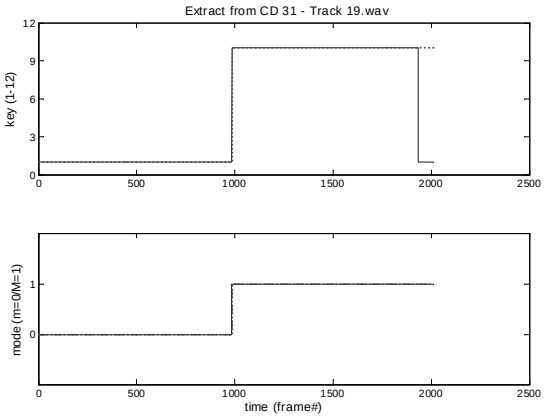

Figure 6. Key detection of "Mozart: Sonata No. 11 In A 'Rondo All Turca'" (solid line: computed key; dotted line: truth)

In Figure 8, three groups of results are shown in the plot: one corresponds to the performance of step 1 without considering modes; one corresponds to the overall performance of key detection with mode into consideration; and one corresponds to recall and precision based on random segmentation (Equation 8 and 10). Additionally, label accuracy based on random should be around 8%, without considering modes.

It clearly shows that when w is increasing, recall and precision are also increasing. Please note that label accuracy is irrelevant to w.

The above two figures show that the segmentation

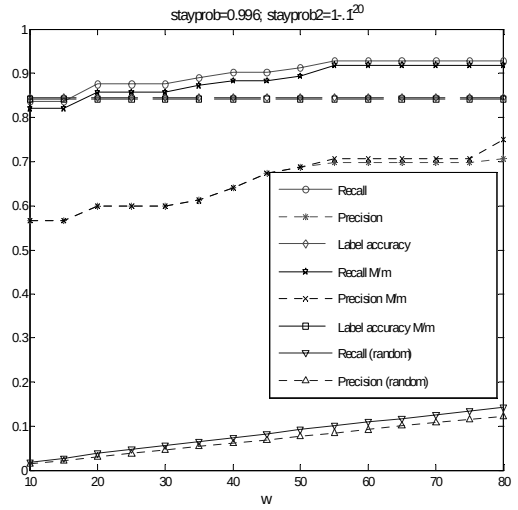

Figure 8. Performance of key detection with varying w (*stayprob*=0.996; *stayprob2*=1-.1^{-20}).

performance (recall and precision) base on the algorithm is significantly better than random segmentation.

5 DISCUSSION

Ideally, all the HMM parameters should be learned from a labeled musical corpus. The training can be made (efficiently) using a maximum likelihood (ML) estimate since all the nodes are observed. Especially, if the training set has the similar timbre property as the test set, the observation distribution can be more accurately estimated employing the timbre information besides prior musical knowledge, and the overall performance should be further improved.

However, this training data set should be very huge. Manually labelling it will involve tremendous amount of work. For example, if the training data set is not big enough, the state transition matrix will be very sparse (0's at many cells) and this may result in many test errors, because any transition that does not appear in the training set will not be recognized. One possibility for future improvement is using Bayesian approach to combine the prior knowledge (via empirical configurations) and the information obtained from a small amount of training data.

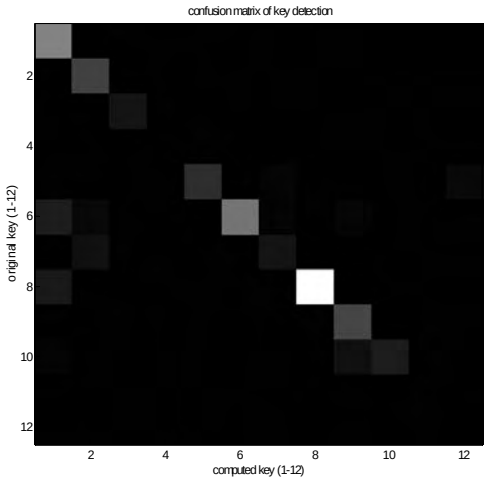

Figure 9. Confusion matrix of key detection.

Another interesting thing to investigate is how the algorithm was confused with keys and whether the errors make a musical sense. Figure 9 shows the confusion matrix of key detection (without considering modes; stayprob=0.996; stayprob2=1-.1^{-20}). It shows that most errors came from confusion between the original key and the dominant or sub-dominant key (e.g., F → C, G → C, F# → C#). This is consistent with music theory presented in Section 2 that these keys are closer to each other and share more common notes.

6 CONCLUSION AND FUTURE WORK

This paper presented an HMM-based approach for detecting key change. Experimental result, evaluated by three proposed measures, demonstrates the promise of the method. Although constraints on music have been made to build simplified models, e.g., diatonic scales, the framework should be easily generalized to handle other types of music.

Each step in the presented framework has been carefully designed with consideration of its musical meaning: from using chromagram representation, to employing cosine-distance observation probability distribution, to empirical configurations of HMM parameters. The experimental result is fairly robust and significantly better than random segmentation.

Future improvement could be adding a training stage (if training data is available) to make this general model customized to specific types of music. More representations need to be explored for other music genres. Furthermore, the HMM parameters should be chosen most appropriate for different applications: for segmentation-based applications, we should maximize precision and recall; for key relevant applications (such as detecting repeated patterns that was presented is Section 1), we should maximize label accuracy.

Similar framework has also been applied to chord detection task for classical piano music, which will not be covered in this paper.

REFERENCES

[1] Foote, J. and Cooper, M. "Visualizing Musical Structure and Rhythm via Self-Similarity," Proceedings of International Conference on Computer Music, Habana, Cuba, September 2001.

[2] Chuan, Ching-Hua and Chew, Elaine. "Polyphonic Audio Key-Finding Using the Spiral Array CEG Algorithm," Proceedings of International Conference on Multimedia and Expo, Amsterdam, Netherlands, July 6-8, 2005.

[3] Gomez, E. and Herrera, P. "Estimating The Tonality Of Polyphonic Audio Files: Cognitive Versus Machine Learning Modelling Strategies," Proceedings of International Conference on Music Information Retrieval, 2004.

[4] Pauws, S. "Musical key extraction from audio," Proceedings of International Conference on Music Information Retrieval, 2004.

[5] Sheh, A. and Ellis, D. "Chord Segmentation and Recognition using EM-Trained Hidden Markov Models," Proceedings of International Symposium on Music Information Retrieval ISMIR-03, Baltimore, October 2003.

[6] Rabiner, L. R. "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," Proceedings of the IEEE, Volume: 77 Issue: 2, Feb. 1989, Page(s): 257 –286.