Automated Transcription of Guitar Music

Carl-Fredrik Arndt
Institute for Computational & Mathematical Engineering
Stanford University
Stanford, California 94305
Email: cfarndt@stanford.edu

Lewis Li
Electrical Engineering
Stanford University
Stanford, California 94305
Email: lewisli@stanford.edu

***Abstract*—In this report, we present our methodology of analyzing music to extract and transcribe guitar notes. Our approach is divided into three steps, the first isolates the guitar track from the music using Independent Subspance Analysis (ISA). Meanwhile, the second step uses both frequency and time domain based approaches to transcribe the guitar track into the corresponding notes. A duration detection algorithm is then used to isolate the length of each note. We will apply this methodology to a series of test data ranging from diatonic scales to segments of classic rock songs. One important distinction is that we do not attempt transcription of guitar chords, focusing only on monophonic scores.**

I. Introduction

A. Motivation

The concept of Automated Music Transcription (AMT) has been extensively studied in recent years. The premise is the autonomous translation of digital audio waveforms into a symbolic music representation akin to what would be on a score. This can be almost viewed as a dis-assembler routine for acoustic music. Such a program would have widespread applications, allowing the modification, and re-arrangement of music, in addition to providing an easier platform for aspiring musicians to learn how to play specific songs. From a different perspective, music transcription is presently performed by trained humans. The development of an effective automated transcription algorithm could provide insight into complicated processes of human perception of music. Another application could be improved identification of musical genres through the analysis of the notes rather than just the frequency signature. Consequently, the possibility of automated music transcription has piqued the interest of individual in a wide variety of fields.

B. Prior and Related Work

With the increase in computing power, research into AMT has grown substantially over the past decade. A closely related field is that of automated speech recognition, which has enjoyed considerably greater research efforts, mainly due to the commercial implications of such products. Real time speech recognition and natural language processing has now entered the mainstream with products such as Siri. While AMT lags behind in research exertion, many of the ideas from speech recognition can be borrowed [14].
Music is separated into two main categories: *monophonic* and *polyphonic*. The former refers to music where no more than one note is simultaneously being played, while latter describes music where no such constraints are defined. Transcribing monophonic music has been studied previously in [2]. The basic steps require disseminating between silence and noise, determining a pitch's tone, and finally mapping to an appropriate note. A variety of techniques such as zero-crossing [18] or auto-correlation have been shown to provide accurate results [16]. Alternatively, frequency domain methods such as Harmonic Product Spectrum (HPS) [20], and Cepstrum [26] have shown comparably favorable results. Conversely, automated transcription of polymorphic music is a much more challenging problem as many spectral peaks arise in the frequency domain corresponding to the various harmonies and fundamentals being played [21]. A number of recent PhD and Masters thesis have been dedicated to addressing this problem. The most recent general approach has centered upon non-negative matrix factorization [25]. Approaches such as [8] have used prerecorded note samples to build a library of note templates, which incoming notes are then mapped to. Despite such efforts, no general-purpose system has been developed that can be on par with a trained human musician. Generally, constraints such as the drums and percussive instruments are not allowed, or the number of other instruments must be known at run-time.
A related problem is that of source separation, commonly referred to as the *Cocktail Part Problem*. This is far from being a solved problem, but approaches such as Principal Component Analysis (PCA) together with Independent Component Analysis (ICA) have demonstrated success on simplified problems [11]. There have been attempts to use pitch detection to aid in source separation [12], but very limited proposals to combine the two to transcribe polyphonic music.

C. Goals

This project intends on developing an implementation that combines the two aforementioned techniques to provide an attempt at AMT. Obviously, the development of a general purpose AMT is not possible, thus a secondary goal is identify the constraints under which such a system would work. It is well known that chord recognition is a very challenging problem [24], hence the scope of this project will focus on single guitar notes. Furthermore, it is known that delays and echos makes source separation very difficult [19]. Consequently, this will complicate separating tracks where electric guitar effects such as flanger, reverb, and chorus are used. In the following sections, we will present out methodology for track separation and music transcription, describe the method of which we collected test data and present and discuss the experimental results before delineating any future steps.

II. Methodology

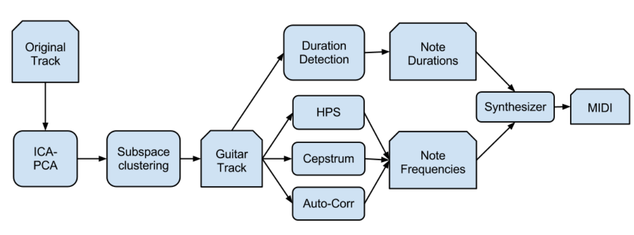

Fig. 1. Overview of methodology

Music is consisted of notes, each with characteristics such as *pitch*, *duration*, *loudness* and *timbre*. Pitch refers to the perception of how "high" or "low" a note is. Notes are classified into sets of

12 semi-tones, termed an octave. Pitch is determined primarily by the *fundamental frequency* of the note. Table II shows the fundamental frequencies of the notes a guitar with standard tuning is capable of producing. Duration refers to the temporal length of each note which is measured in beats. Loudness is measured by the amplitude of the waveform. Finally, timber describes the "color" of a note, which allows humans to desiminate between different instruments playing the same note. In reality, notes are comprised of the fundamental frequency f_0 as well as harmonics, frequencies manifesting at integer multiples of f_0. The initial step of our methodology uses PCA and ICA to isolate guitar tracks. The preceding phase demarcate the portions of the track corresponding to rests; allowing us to evaluate the duration of each note. This involves extracting features of the waveform and applying an appropiate threshold. The penultimate step estimates the pitch of each individual note using both time and frequency domain techniques. Finally, we match the estimated pitch to Table II, and produce a Musical Instrument Digital Interface (MIDI) version of our initial track to verify correctness.

Octave	2	3	4	5	6
E	82.41	164.8	329.6	659.3	-
F	87.31	174.6	349.2	698.5	-
F♯	92.50	185.0	370.0	740.0	-
G	98.00	196.0	392.0	784.0	-
G♯	103.8	207.7	415.3	830.6	-
A	110.0	220.0	440.0	880.0	-
B♭	116.5	233.1	466.2	932.3	-
B	123.5	246.9	493.9	987.8	-
C	-	130.8	261.6	523.3	1047
C♯	-	138.6	277.2	554.4	1109
D	-	146.8	293.7	587.3	1175
E♭	-	155.6	311.1	622.3	1245

TABLE I
FUNDAMENTAL FREQUENCIES OF NOTES THAT A GUITAR CAN PRODUCE WITH STANDARD TUNING [1]

A. Track Separation

The track separation will be done by a method called Independent Subspace Analysis (ISA) which was proposed in [3]. ISA uses ICA create number of different signals, where each signal is matched to a source (instrument).
We here look at a setting where we assume that the number of sources in the audio signal is stationary, but the extension to the non-stationary case is trivial. We now describe all the steps of ISA separately.

1) Divide Signal: If we are given an audio signal x we split this signal into m separate pieces of length w. Here the first piece corresponds to the first part of the song and etc. Denote each of these pieces x^k, $1 \leq k \leq m$ and put them into a $w \times m$ matrix X.

2) Spectral Transformation: Since we are interested in the spectral properties of the signal we apply a linear spectral transformation (in out implementation we used the Short Term Fourier Transform) to the signal. For each x^k, this means that we are using a linear operator M (an $n \times w$ matrix where n is the number of spectral components) and find a representation s^k such that $x^k = Ms^k$. A $n \times k$ matrix S in then created in the same way as X.

3) Feature extraction: To extract features from the one dimensional signals we apply the SVD to S, so that $S = U\Sigma V^T$. Here the V-vectors corresponds to our principal components/features. To determine how many principal components to use, we put a lower threshold on the information ratio. Assuming ρ is the number of principals components used, this means that $\frac{\sum_{i=1}^{\rho}\sigma_i}{\sum_{i=1}^{n}\sigma_i}$, (here σ_i tea the singular values) needs to be greater then some given number (chosen to be 0.7 in [3]).

4) Independent Feature Creation: If we group all the ρ vectors from previous step into a $\rho \times n$ observation matrix V^ρ we can now use the standard case ICA on this. This will create an independent set of feature vectors z_i, $1 \leq i \leq \rho$.

5) Subspace Identification: To create clusters containing separate instruments, we will use the Kullback-Leibler divergence. First calculate the so called IXEGRAM matrix, defined in [3], as

$$D = \begin{pmatrix} \delta_{KL}(z_1,z_1) & \delta_{KL}(z_1,z_2) & \cdots & \delta_{KL}(z_1,z_\rho) \\ \delta_{KL}(z_2,z_1) & \delta_{KL}(z_2,z_2) & \cdots & \delta_{KL}(z_2,z_\rho) \\ \vdots & \vdots & \ddots & \vdots \\ \delta_{KL}(z_\rho,z_1) & \delta_{KL}(z_\rho,z_2) & \cdots & \delta_{KL}(z_\rho,z_\rho) \end{pmatrix}.$$

Here $\delta_{KL}(z_i,z_j)$ denotes the Kullback-Leibler divergence between z_i and z_j. Since the Kullback-Leibler divergence is calculated using a distribution function, and we only have access to a samples, these has to be approximated numerically. Next introduce the cost-function

$$H(M;D) = \sum_{c=1}^{\kappa} \frac{1}{\sum_{j=1}^{\rho} M_{jc}} \sum_{i=1}^{\rho}\sum_{k=1}^{\rho} M_{ic}M_{kc}D_{ik}. \tag{1}$$

Here M is a $\rho \times \kappa$ binary valued matrix where κ is the number of clusters and $M_{i,j}$ denotes whether signal i is in cluster κ or not. Note that the number of clusters is prespecified, so we have to run this part of the algorithm multiple times to approximate right number of clusters. To find M for a fixed k we minimize the cost function (1). This is be done by using a deterministic annealing algorithm (basically a fixed point algorithm) which solves

$$\epsilon_{i,\lambda} = \frac{\sum_{k=1}^{\rho}\left(D_{i,k} - \frac{\sum_{j=1}^{\kappa} M_{j,\lambda}D_{j,k}}{2\sum_{j=1,j\neq i}^{n} M_{j,\lambda}}\right)}{1+\sum_{j=1,j\neq i}^{\rho} M_{j,\lambda}},$$
$$M_{i,\lambda} = \frac{-\exp(\epsilon_{i,\lambda}/T)}{\sum_{\mu=1}^{\kappa}\exp(-\epsilon_{i,\mu}/T)},$$

where T a variable parameter. This is described in [10].

6) Retrieve signals: In the last step we simply have to retrieve the κ audio signals of the different instruments. This is done by first inverting the ICA step by using the mixing matrix, then inverting the SVD and at last using the inverse transform of the spectral transformation.

B. Duration Detection

Silence detection is a commonly studied problem in Voice Activity Detection (VAD) [13], and we will borrow some of the commonly used audio features to detect rest beats in music.

1) Short Term Energy: The short term energy is defined as:

$$X_i = \sum_{m=1}^{M} x^2[m]w^2[\hat{n}-m]$$

where w refers to a Hamming window. The short term energy gives a representation of the temporal amplitude variation. This is perhaps the best feature for detecting periods of silence in a an audio sequence.

2) Zero-Crossing Rate: The zero-crossing rate is defined as:

$$Z_i = \sum_{m=1}^{M} 0.5|\operatorname{sgn}(x[m]) - \operatorname{sgn}(x[m-1])|w[\hat{n} - m]$$

Zero-crossing measures the noisiness of a signal, and has been used to discern between music and other noises (background, vocals etc) [7]

3) Spectral Flux: The spectral flux is defined as:

$$FL_i = \sum_{m=1}^{M} (EX_i[m] - EX_{i-1}[m])^2$$

where EX_i is the normalized DFT coefficients of the i^{th} window. This measures the local spectral change which is notably higher due to noise during silence.

4) Spectral Centroid: The spectral centroid is defined as:

$$C_{\hat{n}} = \frac{\sum_{m=-\infty}^{\infty} mX[m]}{\sum_{m=-\infty}^{\infty} X[m]}$$

Here X refers to the DFT of x. The spectral centroid gives a metric for the 'brightness' of the sound [23].

5) Weiner Entropy: The Weiner Entropy is defined as:

$$W_i = \frac{\exp\left(\frac{1}{M}\sum_{m=0}^{M-1} \ln x[n]\right)}{\frac{1}{M}\sum_{m=0}^{M-1} x[n]}$$

Weiner Entropy is used as a metric for determining if a sound is more tone like or noise like [5].

6) Thresholding: With these features, it is possible to train a Support Vector Machine (SVM) or another other classifier to label portions as noise or music. However, it has been shown that much simpler techniques can be employed to provide excellent separation [17]. We adapted the following steps from [6].

1) Compute histogram of feature values
2) Detect histogram local maximas
3) Compute threshold $T = \frac{WM_1+M_2}{W+1}$, where W is a user defined weight, and M_1 and M_2 are the first and second maximas.

We will use a combination of the features defined above and a smaller window size (10ms to 25ms), since rests in music are generally shorter than that of human speech.

C. Music Transcription

Once the tone portions of the track have been isolated, a pitch detection algorithm can be deployed to compute the notes themselves. Pitch detection is also a common problem in voice recognition and a variety of algorithms have been proposed [4]. As there is no overall consensus of an ideal pitch recognition algorithm, we have elected to implement the a few of the most popular. This can offer us insight to any causable relationships between dataset and an optimal algorithm.

1) Harmonic Product Spectrum: Harmonic Product Spectrum (HPS) is a frequency domain algorithm [15]. The basic idea is to take the Fourier Transform of a windowed portion of the waveform. The windowing function that was proposed was the Hamming Window. Ideally, this will yield a set of peaks at the fundamental frequency and its associated harmonics. f_0 can be abstracted simply by finding the greatest common denominator of peak frequencies. A computationally efficient method of evaluating this is to down-sample the spectrum by an integer factor, and then computing the element wise product with the original spectrum. This will eliminate high frequency harmonics, while magnifying the fundamental frequency. Figure 2 gives a visual representation of this algorithm. A few iterations of the downsample-multiply procedure will yield a spectrum that has a maxima at the fundamental frequency.

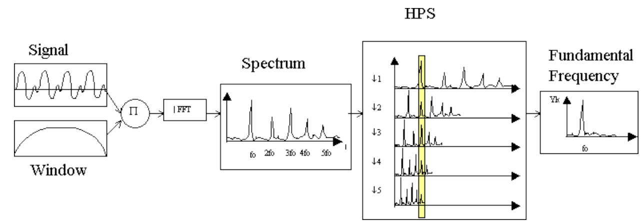

Fig. 2. Overview of Harmonic Product Spectrum

2) Cepstrum: Cepstrum is another Fourier analysis technique which involves evaluating the logarithmic amplitude spectrum of the signal.That is:

$$\mathcal{C} = |\mathcal{F}\{\log X\}|^2$$

Assuming this log amplitude spectrum contains many regularly space harmonics, the Fourier analysis will yield peaks corresponding to the fundamental frequency. In essense, the agorithm searches for periodicity within the spectrum.
The cepstrum has units of *quefrency*, and peaks within the cepstrum are known as *rahmonics*. These correspond to periodicies within the spectrum. Therefore, to obtain an estimate of f_0, we search for peaks within the quefrency regions corresponding to fundamental frequencies of notes a guitar can produce. Quefrency and frequency are inverses of one another.

3) Auto-Correlation: Auto-correlation is the most commonly used time-series pitch detection algorithm in speech recognition [9]. The idea relies on the fact that the correlation signal will have a peak at the lag corresponding to the pitch period. The autocorrelation for a discrete signal is given as:

$$R[k] = \sum_{m=0}^{N} -k - 1s[m]s[m+k]$$

Here *N* refers to the number of samples in the window, and *k* is the lag index. The choice of *N* is important in the sense that it must cover 2 or more periods such that periodicity can be seen. Conversely, a larger window will dimish the algorithm's capability to detect temporal variations in pitch. Now to find the fundamental frequency we use the following:

$$f_0 = \frac{1}{k_{max} f_s}, R[k_{max}] = \arg\max_k R[k]$$

Here f_s is the sampling frequency. The key assumption is that if $s[m]$ is periodic, $R[k]$ should exhibit peaks at integer multiples of the period. The main peak in the autocorrelation function will be at $k = 0$ (since a signal should always correlate to itself). The location of the next peak gives an estimate of the period. The method usually require a number of periods of data to form a reliable estimate, and thus some averaging of the frequency signal is unavoidable.

D. Post-Processing & MIDI Synthesis

Pitch detection and duration detection will both yield a vector corresponding to the number of windows. The pitch detection vector will contain the estimated fundamental frequencies. By finding the closest match for each frequency in Table II, we can obtain a vector of semi-tone notes. This vector is then multiplied element wise by the boolean vector generated by duraction detection to yield the estimated transcription, where rests correspond to 0 frequency. To generate an audio waveform, we transformed the appropiate notes into MIDI notes, allowing for playback.

III. Data Collection

For the intial testing of pitch detection and duration detection, we recorded monophonic guitar tracks using iJam and Apple's Garageband. The test cases were comprised of a E-Major scale, the introduction to Led Zeppelin's Stairway To Heaven, and Guns N'Roses' Sweet Child O' Mine. To test the track seperation we used Red Hot Chilli Peppers' Otherside, which features drums, guitar, bass and vocals.

IV. Experimental Results

A. Track Separation

We ran the previously described algorithm on approximately 10 seconds of the initial part of Otherside. This part of the song only contains a bass guitar and a regular guitar and thus we could set the number of clusters, κ, to be fixed to two. After doing this we transformed the received clusters back to Wav files and played them. One could clearly here which part contained the bass note versus the regular guitar. In figure 3 we show the initial Wav file and then both the first (regular guitar) and second (bass) clusters. The main problem with the algorithm is that it is build to separate pitched instruments and thus fail to separate the song from the instruments as noted in [3].

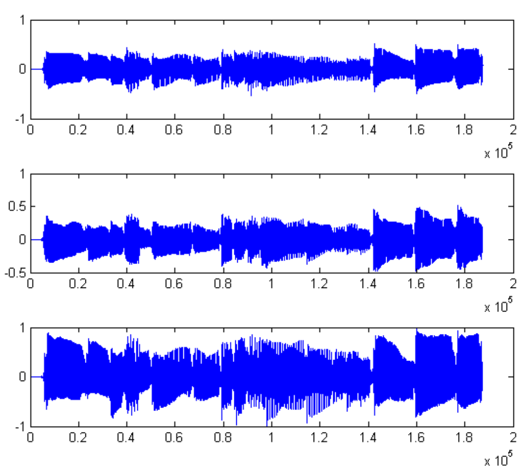

Fig. 3. Time-series plot of the original Wav file(top), separated guitar part(middle) and bass guitar part(bottom).

B. Duration Detection

We attempted duration detection using permutations of the features described in Section II-B. It was found that short time energy and spectral centroid provided acceptable results, while additional features did not necessarily yield any improvements. Figure 4 shows the features after a median filter has been applied. The bottom two plot indicates the periods during which a note is being played compared to the actual waveform. As we can see, the duration detection is reasonably successful at identifying the rests and notes.

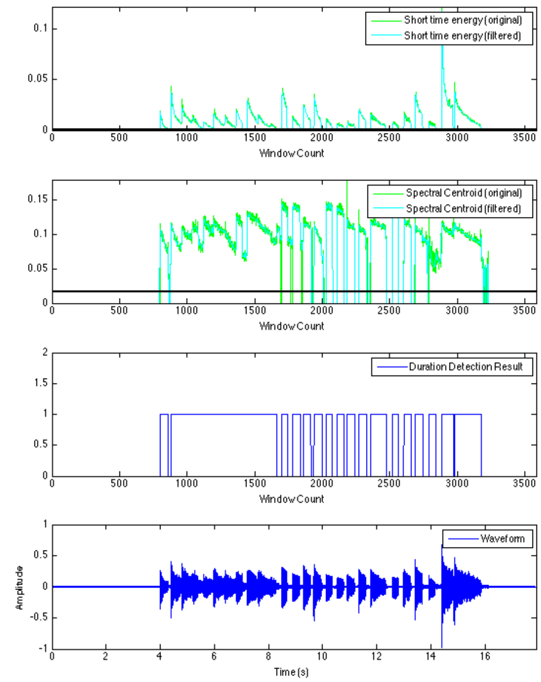

Fig. 4. Duration detection algorithm performed on Stairway To Heaven. A) Short time energy with median smoothing. B) Spectral centroid with median smoothing. C) Duration detection result. D) Waveform of the original track.

C. Music Transcription

We attempted music transcription using HPS, Cepstrum and Auto-Correlation on the various sets of test data. For our selected test cases, HPS appeared to yield the best results.

1) E-Major Scale: This was a simple test case where we played 8 notes that are part of a E-Major scale. All three of the algorithms were able to recognize all notes as shown in Figure 5.

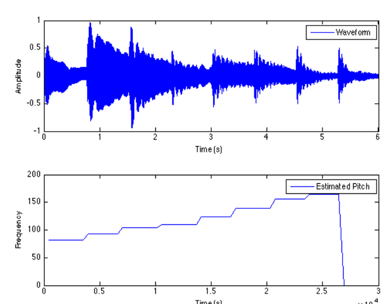

Fig. 5. Pitch detection result combined with duration detection for E-Major scale using HPS

2) Stairway to Heaven: Stairway to Heaven is an slow tempo song (approximately 80bpm in 4-4 time). We recorded a version that was strictly monophonic. All three algorithms generated small incorrect spikes in frequency at the begining of each note. This can be rectified by using a median filter of 3 elements.

3) Sweet Child O' Mine: Sweet Child O' Mine is a fast tempo song (200 bpm), that was originally played by Guns N' Roses using an overdriven electric guitar. For the purpose of this project, we recorded a version played using a clean guitar. The output is shown in Figure 6.

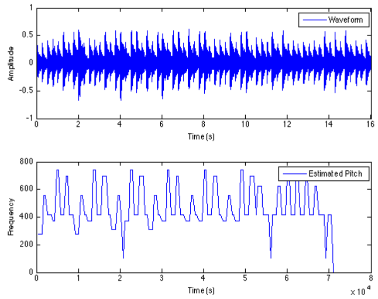

Fig. 6. Pitch detection result combined with duration detection for Sweet Child O Mine

D. Summary

		HPS		Cepstrum		Auto-Corr	
Song	Notes	✓	%	✓	%	✓	%
E-Major	8	8	100	8	100	8	100
Stairway	26	24	92	22	85	23	89
SChild	32	29	90	18	56	24	66
OSide	50	48	96	42	84	44	88
Sandman	24	24	100	22	92	23	96
Fad2Blk	32	30	93	29	91	27	84

In our test cases, we were able to demonstrate over 90% success rates using HPS. It should however be noted that we selected our test cases in order to match the capabilities of our algorithms. That is, monotonic tracks with no chords. In order to cater to all songs, this restriction will have to be lifted.

V. Future Work

In this project, we have demonstrated that ISA and HPS can be used to achieve music transcription at least on select test data. The greatest challenge is the transcription of chords. Bayesian approaches such as [24], and generated success rates of over 70% and could be adapted to serve an appropiate purpose here. Furthermore, to generate an actual score the beat of the music will have to be determined. Certain attempts such as [22] have also achieved around 70% success rates. Once the tempo is established, it is simple to transform the durations into note lengths. Finally, certain aspects of a score such as dynamics should also be detected. The combination of all these components would result in a complete automated music transcription program.

Acknowledgements

The authors would like to thank Prof. Andrew Ng and the CS229 Teaching Staff for their Machine Learning course.

References

[1]

[2] Juan Pablo Bello, Giuliano Monti, Mark Sandler, and Mark S. Techniques for automatic music transcription. In *in International Symposium on Music Information Retrieval*, pages 23–25, 2000.

[3] M. A. Casey and A. Westner. Separation of mixed audio sources by independent subspace analysis. In *Proc. Int. Comput. Music Conf.*, July -00 2000.

[4] Patricio De La Cuadra and Aaron Master. Efficient pitch detection techniques for interactive music. In *In Proceedings of the 2001 International Computer Music Conference, La Habana*, 2001.

[5] S. Dubnov. Generalization of spectral flatness measure for non-gaussian linear processes. *Signal Processing Letters, IEEE*, 11(8):698 – 701, aug. 2004.

[6] Theodoros Giannakopoulos. Silence removal in speech signals, 2010.

[7] Fabien Gouyon, Francois Pachet, and Olivier Delerue. On the use of zero-crossing rate for an application of classification of percussive sounds. In *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFX-00*, 2000.

[8] John E. Hartquist. Real-time musical analysis of polyphonic guitar audio. Master's thesis, California Polytechnic State University, 2012.

[9] W.J Hess. pitch and voicing determination". *Advancesin Speech Signal Processing*, 1992.

[10] T. Hofmann and J. M. Buhmann. Pairwise data clustering by deterministic annealing. *IEEE T. Pattern Anal*, pages 1–14, 1997.

[11] Aapo Hyvrinen and Erkki Oja. Independent component analysis: algorithms and applications. *Neural Networks*, 13:411–430, 2000.

[12] M. Karjalainen and T. Tolonen. Multi-pitch and periodicity analysis model for sound separation and auditory scene analysis. In *Proceedings of the Acoustics, Speech, and Signal Processing, 1999. on 1999 IEEE International Conference - Volume 02*, ICASSP '99, pages 929–932, Washington, DC, USA, 1999. IEEE Computer Society.

[13] Tomi Kinnunen, Evgenia Chernenko, Marko Tuononen, Pasi Frnti, and Haizhou Li. Voice activity detection using mfcc features and support vector machine, 2007.

[14] Anssi P. Klapuri. Automatic music transcription as we know it today. *Journal of New Music Research*, pages 269–282, 2004.

[15] Gopala Krishna Koduri, Joan Serr, and Xavier Serra. Characterization of intonation in carnatic music by parametrizing pitch histograms. In *Proceedings of the 13th International Society for Music Information Retrieval Conference*, Porto, Portugal, October 8-12 2012. http://ismir2012.ismir.net/event/papers/199-ismir-2012.pdf.

[16] Philip Mcleod, Damon Simpson, Robert Visser, Mike Phillips, Ignas Kukenys, Yaoyao Wang, Arthur Melissen, Natalie Zhao, and All The. Fast, accurate pitch detection tools for music analysis.

[17] M. H. Moattar and M. M. Homayounpour. A simple but efficient real-time voice activity detection algorithm, 2009.

[18] C. Panagiotakis and G. Tziritas. A speech/music discriminator based on rms and zero-crossings. *IEEE Transactions on Multimedia*, 7:155–166, 2004.

[19] Barak Pearlmutter and Lucas Parra. A context-sensitive generalization of ica, 1996.

[20] L. Rabiner and R. Schafer. *Digital Processing of Speech Signals*. Englewood Cliffs: Prentice Hall, 1978.

[21] Matti Ryynnen and Anssi Klapuri. Transcription of the singing melody in polyphonic music. In *in Proc. 7th International Conference on Music Information Retrieval*, 2006.

[22] Eric D. Scheirer. Tempo and beat analysis of acoustic musical signals. *Journal of the Acoustical Society of America*, 103(1):588–601, 1998.

[23] Emery Schubert and Joe Wolfe. Does timbral brightness scale with frequency and spectral centroid?

[24] Alexander Sheh and Daniel Ellis. Chord segmentation and recognition using em-trained hidden markov models, 2003.

[25] Paris Smaragdis and Judith C. Brown. Non-negative matrix factorization for polyphonic music transcription. In *In IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 177–180, 2003.

[26] Christopher Wendt and Athina P. Petropulu. Pitch determination and speech segmentation using the discrete wavelet transform. In *in Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 45–48, 1996.