A. Georgaki and G. Kouroupetroglou (Eds.), Proceedings ICMC|SMC|2014, 14-20 September 2014, Athens, Greece

Optimizing Melodic Extraction Algorithm for Jazz Guitar Recordings Using Genetic Algorithms

Sergio Giraldo
Music Technology Group, Pompeu Fabra University
sergio.giraldo@upf.edu

Rafael Ramirez
Music Technology Group, Pompeu Fabra University
rafael.ramirez@upf.edu

ABSTRACT

Extraction of the main melody of a musical piece is a preliminary step in the process of transcribing the piece. Automatic melodic extraction is the task of computationally extracting what a human listener would perceive as the main melody of a polyphonic recording. Several melodic extraction systems have been proposed. However, such systems normally require a number of parameters to be manually tuned in order to accurately perform melody extraction in different contexts, i.e. instruments combinations. In this study, we propose a methodology for automatically optimizing some relevant parameters of the extraction algorithm *Melodia* [1] using genetic algorithms. We simultaneously obtained both MIDI and audio recordings of jazz standards, and we collected commercial audio recordings extracted from jazz guitar CDs. Based on the MIDI recordings as ground truth, two different instrument settings are compared (Jazz trio and quartet), as well as different audio mixing of the melody with respect to the accompaniment track. We show that, compared to using the default parameters, the overall accuracy of the melody extraction with the optimized parameters is improved.

1. INTRODUCTION

In jazz education, the extraction of the main melody of musical piece is a ”must to do” task. In contrast to classical music, in popular music melody extraction is the way to understand melodic variations or performance actions that lead to play in certain style. Similar to spoken language, popular music is often taught mainly by oral tradition, thus the student learn by listening and copying experts' performances. The task of automatically extracting the main melody of a polyphonic recording is a computational problem widely studied in recent years. Given a recording of a polyphonic musical piece (performed either by several instruments or by a polyphonic instrument), the task is to automatic melody extraction is to computationally identify the main melody of a piece. The main melody may be defined [2] as the monophonic sequence of pitches that a listener would hum (or wistle) when listening to a polyphonic piece of music regardless the music style. Several applications of automatic melody extraction can be mentioned, for instance query by humming [3], soloist identification [4], among others.

In this study we focus on the salience pitch approach method used by Salamon et al. [1] *”Melodia”*, in which melodic extraction is performed in several steps: sinusoidal extraction, salience function computation, pitch contour computation and characterization, and melody selection. For the computation of each of these steps different parameters such as minimum and maximum frequency, peak distribution and peak frame threshold, pitch and time continuity need to be defined. In the system, these parameters have been tuned using the grid search method over different data bases in which melody was performed mainly by singing voice. However, the default parameters' values fail to extract the melody for some specific instrument settings (jazz guitar recordings in our case). We propose a method to optimize the algorithm's main parameters using genetic algorithms, based on specific instrument settings. In this study we focus on two instrument settings: Trio setting consistinf of electric guitar, bass and drums, and Quartet setting consisting of electric guitar, bass, drums and piano. In both cases the melody is performed by the guitar.

The rest of the paper is organized as follows: section 2 presents the background to automatic melodic extraction, a brief description of the *Melodia* [1] algorithm, and a brief introduction to genetic algorithms. Section 3 describe our data set of songs and presents our methodology for optimizing the algorithm parameters. In section 4 we present our results, and finally, section 5 presents our conclusions and future work.

2. BACKGROUND

2.1 Melody extraction

The most widely used methods for melodic extraction are the ones based on salience pitch calculation. Usually these methods are performed in three steps. First, an spectral representation of the signal is computed using spectral analysis techniques (e.g. Fast Fourier Transform). Second, a salience function (time-frequency representation of pitch salience) is computed to obtain several f0 candidates, usually using weighted harmonic summation methods. Finally, the melody peaks are selected based on tracking methods over frequency and time. Other methodologies use source separation based on timber models and grouping principles [ref], and some use stereo separation to estimate the panning of each source.

In this study we focus on Salamon et al. 2011 [1]. In their

approach melodic extraction is performed by several steps. First, *sinusoidal extraction* filters the signal to enhance the most audible frequencies of human hearing range. Then, the Short Fourier Transform is calculated using a Hann window of 46.4ms with a hop size of 2.9ms and a 4 zero padding factor. Finally the peaks are corrected using instantaneous frequency method [ref].In a second step, the *salience function computation* is performed based on the summation of the weighted energy of the harmonic peaks of a given frequency in order to obtain the f0 candidates. The number of harmonics considered and the weighting scheme is an important factor that affects the salience computation. The third step is *pitch contour computation*, in which the peaks detected in the previous step are grouped into pitch contours, based on several thresholds defined per frame basis, as well as per time continuity basis. Some of the algorithm's key parameters in this step are the following:

- Peak Distribution Threshold: Allowed deviation below the peak salience mean over all frames (fraction of the standard deviation)
- Peak Frame Threshold: Per-frame salience threshold factor (fraction of the highest peak salience in a frame)
- Pitch Continuity: Pitch continuity cue (maximum allowed pitch change during 1 ms time period)
- Time Continuity: Time continuity cue (the maximum allowed gap duration for a pitch contour)

The fourth step is *contour characterization* in which the melody contour is chosen among the contours created in the previous step. To do so, a set of features that guide the system to select the main melody is implemented. These features include pitch deviation, pitch trajectory, presence of vibrato, as well as contour pitch, length and salience. Finally the last step is *melody selection* which is performed in four main steps: voicing detection, octave error minimization, pitch outliers removal, and final melody selection. A complete explanation of each step can be found in [ref. *Melodia*]

2.2 Genetic Algorithms

Genetic Algorithms (GA) are stochastic optimization algorithms which imitate the biological mechanisms of natural selection. GA are widely used in several optimization problems involving discontinuous, noisy, high-dimensional, and multi-modal objective functions. In contrast to other optimization algorithms, the search that GA's perform is done in a more global context, whereas others (e.g. gradient descent) perform search in a more local context. Genetic algorithms work as follows: First, an initial random population of individuals is created. This initial population serves as seed to generate future generations by means of combining different parents (crossover) and randomly modifying a single individual (mutation). The algorithm iterates, and in each iteration it selects the best subjects based on a fitness function. The next generation of individuals is generated by applying mutation and crossover. The individuals with best fitness values are preserved in the next generation, while the others are discarded. The process iterates until a maximum number of iterations is reached or the fitness relative value of the best individual does not change more than a tolerance value during several generations. The most popular applications of GA in a musical context have been done for music composition [7] and [8]. Other approaches have used GA's for automatic music transcription [9], music segmentation [10], and interactive music applications [11], among many others.

3. METHODOLOGY

Two experiments were proposed to test our methodology for optimizing melody extraction. A first experiment was done using 22 simultaneous audio midi recordings. Ground truth was generated from the recorded MIDI data, and mixes were created using the synthesized version of the MIDI recording. The second experiment was done in a more realistic context using four recordings in which melody is recorded separately from the accompaniment track. In this case ground truth was build from automatic extraction of the pitch contour of the melody track, using YIN algorithm [12] and performing manual correction afterwards.

3.1 First Experiment Data

The train data set consisted of 22 jazz standard tunes melodies, recorded by a professional guitarist. The indications given to the musician were that melodies should be played monophonic, with no chord strums, or double notes. Only the main melody of the tune was recorded. Recordings were segmented in order to eliminate repetitions. For example, if a tune had the form AABA, only the second A and the B parts were used. The melodies were recorded over commercial backing tracks recordings, in which bass and piano parts are recorded separately on each stereo channel [13]. This made possible to mix different instrument settings: drums, bass and guitar (trio) and piano, drums, bass and guitar (quartet).

3.1.1 Building ground truth

All the melodies were simultaneously recorded in both audio and MIDI, using a commercial guitar to MIDI converter (Sonnus). Because the accuracy of the device is not perfect, manual correction of the audio to MIDI converted melody was performed after each recording. Each MIDI event (note) has information of pitch (midi number), onset (in seconds) and duration (in seconds). Using this information and a hop size of 46.4ms (as used by the melody extractor algorithm) it was possible to obtain the pitch contour of the melody frame by frame. Finally, to assured that the pitch contour corresponds exactly to the melody to be extracted from the created mix, the MIDI sequence of the melody was synthesized in wav format (using the bluilt-in guitar synthesizer of the DAW - Digital Audio Workstation) and mixed with the accompaniment backing track.

3.1.2 Experiment overview

Our aim was to find a set of parameters that work well for tunes of similar type. We defined two main groups of tunes: one was an audio mix of the recorded tunes with synthesized guitar, bass and drums (trio set), and the other was the audio mix of the same recorded tunes with synthesized guitar, bass, piano, and drums (quartet set). For each of the above sets we followed a similar method as [1], creating different audio mixes in which the melody is at three different sound levels (-6dB, 0dB and +6dB) with respect to the overall sound level of the backing track. Eighteen of the tracks were used as train data to optimize parameters. The the optimized parameters were tested in the four songs that were left aside for testing.

3.2 Second Experiment Data

In a second experiment, the data set consisted of four standard jazz tunes obtained from a guitar teaching series book [14] in which melody is recorded separately from the accompaniment track. This set was chosen to test our optimization strategy in a more real context in which common effects such as reverb can make melody extraction more difficult.

3.2.1 Building ground truth

In this case, the pitch contour of the melody was obtained from the melody track using Yin algorithm [Ref. Yin]. We set the frame resolution for extraction of the algorithm to 46.4ms to be consistent with the frame resolution of *Melodia*. However, manual correction needed to be performed. To do so, we implemented a simple GUI (guided user interface) to plot the obtained pitch contour over the spectrum of the melody. This facilitated the visualization and correction of errors such as octave shift and/or missing pitches.

3.2.2 Experiment overview

We set up different mixes with different sound levels of the melody, similar to experiment number one. For each mix we used the method of "leave one out", so we used tree of the song to optimize parameters, and the forth one to test the optimized parameters. This process was repeated four times leaving each time a different song, so all the songs were used for training, as well as for testing.

3.3 Parameter Optimization

As explained in 2 *Melodia* algorithm uses several steps in the extraction of the pitch contour. Each of these steps uses different parameters and thresholds for the calculation. Each of these parameters were reported to have been calculated using grid search, over different test sets [1], in which the main melody was mainly vocals. Our aim in this study is to optimize these parameters for the extraction of the melody played by a clean electric guitar in a jazz guitar context.

3.3.1 Parameters

After some initial tests the parameters found to be more sensitive for melodic extraction accuracy were:

- Maximum Frequency
- Minimum Frequency
- Peak Distribution Threshold
- Peak Frame Threshold
- Pitch Continuity
- Time Continuity

3.3.2 Cost function: Fitness function

In order to manipulate the *Melodia* parameters we used the algorithm implementation found in Essentia Library [15]. We manipulate the code to set the selected parameters as input variables. A fitness function was implemented in Matlab in which the algorithm is called with some initial parameters (the default ones) and for each parameter a threshold is defined according to the values reported in [1]. The cost of the fitness function is calculated based on the overall accuracy measure reported in [1], which is defined as the total proportion of frames correctly estimated by the algorithm. This proportion is measured based on the true negatives (TN) and true positives (TP) for which the pitch estimation is correct (between a range of +/- 1/4 of tone of the ground truth). TN and TP were calculated as follows:

$$TP = 1200 * log2(\frac{ExtractedMelody}{GroundTruth}) > 50cents$$

$$TP + TN = 1200 * log2(\frac{ExtractedMelody}{GroundTruth})$$

Thus, the cost (j) was calculated as:

$$j = 1 - (\frac{TP}{TP+TN})$$

For the cost calculation all the frames of all the set of melodies were summed.

As an example we present in Figure 1 the extracted contour by *Melodia* algorithm and the ground truth pitch contour extracted from the MIDI data. On the bottom graph differences between both contours are plotted. The main idea behind the optimization process is to minimize the overall errors shown in the graph using the overall accuracy measure explained above.

3.3.3 Implementation: Genetic Algorithm

In order to optimize the parameters of the melody extraction algorithm we implemented a genetic algorithm using the Matlab implementation of GA [16]. After defining a list of input parameters, a threshold for each parameter and a cost function, the algorithm randomly generates a initial population, in our case it consisted of a set of 20 vectors with different parameter values combinations, randomly generated between defined upper and lower bounds. The stropping criteria used in this study was set to a maximum of 500 iterations, and a relative threshold change in fitness function of $1x10^{-6}$. Crossover factor was set to 0.8 and mutation factor was set to 0.02.

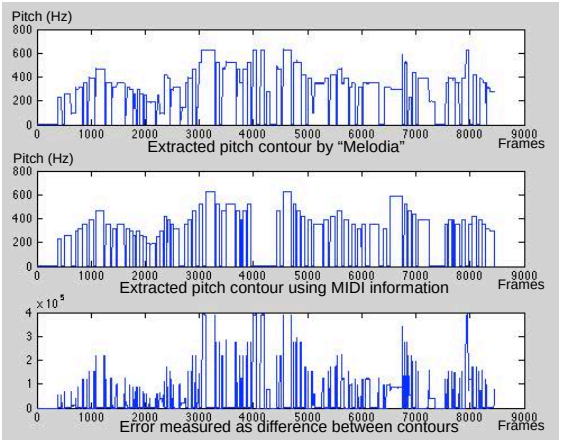

Figure 1. In this figure we present the pitch contour extracted by *Melodia* in a music excerpt (top), and the ground truth pitch contour extracted form the MIDI file (middle). On the bottom we present the detection errors calculated as the difference between the two contours

4. RESULTS

For the first experiment (18 songs) the results for the training set is presented in table Table 1. For each instrument setting (Trio and Quartet) we apply the optimization methodology at the three different mixing levels. The Overall Accuracy was calculated summing all the frames of all the analyzed tunes. The obtained algorithm's accuracy is presented in the first column of each instrument setting (OADP). After the optimization process the obtained accuracy is presented in the second column (OAAO) for each instrument setting.

	Trio		Quartet	
	OADP %	OAAO %	OADP %	OAAO %
Mix at 0dB	54.25	73.16	46.43	73.49
Mix at -6dB	41.93	55.44	46.34	54.62
Mix at +6dB	79.75	82.80	81.15	82.54

Table 1. Overal Accuracy for Train set in experiment one. First column presents the Overall Accuracy with Default Parameters (OADP) and second one the Overall Accuracy After Optimization (OAAO), at different mixing levels for each of instrument settings (Trio and Quartet).

In table 2 we present the Overall Accuracy obtained in the test set. The melodies of the test songs were extracted using the default parameters, and the optimized ones. The results are shown for both of the instrument settings at the three different mixing levels. First column presents the Overall Accuracy with Default Parameters (OADP) and second one the Overall Accuracy with Optimized Parameters (OAOP)

For the second experiment, we used a set of four songs extracted from a jazz guitar education series. We optimized the parameters using three of the songs and test them on the extraction of the fourth song (leave one out approach). In table 3 we present, for each fold, the accuracy using the default parameters and the accuracy using the optimized parameters.

	Trio		Quartet	
	OADP %	OAOP %	OADP %	OAOP %
Mix at 0dB	52.29	72.58	59.92	73.16
Mix at -6dB	43.1	53.06	46.95	55.45
Mix at +6dB	81.52	82.80	74.08	81.37

Table 2. Overal Accuracy for Test set in experiment one. First column presents the Overall Accuracy with Default Parameters (OADP) and second one the Overall Accuracy with Optimized Parameters (OAOP), at different mixing levels for each of instrument settings (Trio and Quartet).

	Fold 1		Fold 2	
	OADP %	OAOP %	OADP %	OAOP %
Mix at 0dB	48.41	58.00	49.90	55.86
	Fold 3		Fold 4	
	OADP %	OAOP %	OADP %	OAOP %
Mix at 0dB	56.52	59.57	62.91	68.85

Table 3. Overal Accuracy for Test set in experiment two. First column presents the Overall Accuracy with Default Parameters (OADP) and second one the Overall Accuracy with Optimized Parameters (OAOP), at different mixing levels for each leave one out fold.

5. DISCUSSION

In figure 2 the error against the number of iterations is plotted for the quartet setting and an mix of +6dBs. In this figure it is possible to notice how after approximately 250 iterations the algorithm finds a minimum. Also, during the first iterations of the algorithm, it is possible to notice how sensitive is the algorithm to the chosen parameters. In table 1 (training data) on each scenario the performance of the algorithm increases after optimization. Also the performance of the algorithm increases as the sound level of the mix increases. The improvement on -6dB scenario is less than in +6dB scenario. This is expected since, as it is the case for human listeners, the melody is harder to be differentiated from the backing track if the sound level is too low. In table 2, the results are similar as the ones described above, but the accuracy is lower in general. This is due to the fact that we are now testing on unseen data. However in all the cases there is a considerable improvement of the performance using optimized parameters compared to the default ones.

On the second experiment in table 3, it is possible to see how the performance of the algorithm increases in all cases after optimization. Even though the improvement in performance is less than for the MIDI generated melodies, in

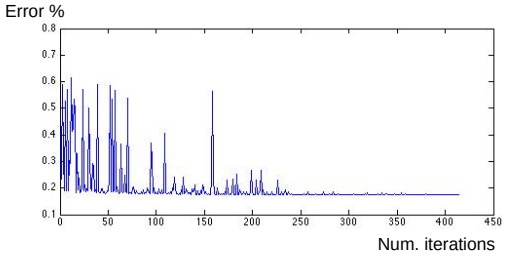

Figure 2. Error against number of iterations on scenario quartet and +6db. After 250 iterations a minimum is found

this scenario we are dealing with more realistic recordings, in which effects, such as reverb, and other artifacts are present. Also hand annotated ground truth may contain human biases, as for example the duration of the notes, or level of legato, may differ between different listeners.

6. CONCLUSIONS

In this paper we have proposed a method to optimize the extraction parameters of *Melodia*, an algorithm for melody extraction, using Genetic Algorithms. Our approach is oriented to improve the melodic extraction for specific instrument settings, for problems in which the extraction has to be performed on the same instrument set (e.g. electric guitar, bass, piano and drums). We have performed two experiments, a first one using 22 simultaneous midi and audio recordings, in which the MIDI information has been used to build the ground truth. Optimization was performed in 18 songs and 4 songs were left for testing. Two instrument settings were created: trio (guitar, bass, drums) and quartet (guitar, bass, piano, drums). Also different audio mixes were created in which the melody is at different sound levels with respect to the accompaniment track. The second experiment was performed on four commercial recordings in which the melody was recorded in a separated channel from the accompaniment track. In this experiment we used a ”leave one out approach” for optimization and testing. Results show that our optimization methodology improves in all cases the overall accuracy of the detection for the first experiment. Improvement in overall accuracy in second experiment is less as we are applying the methodology on a more real context in which aspects like reverberation in the melody track may induce more errors in the melodic detection.

7. REFERENCES

[1] J. Salamon and E. Gómez, “Melody extraction from polyphonic music signals using pitch contour characteristics,” *IEEE Transactions on Audio, Speech and Language Processing*, vol. 20, pp. 1759–1770, 08/2012 2012. [Online]. Available: http://mtg.upf.edu/system/files/publications/SalamonGomezMelodyTASLP2012.pdf

[2] G. E. Poliner, D. P. W. Ellis, F. Ehmann, E. Gomez, S. Steich, and B. Ong, “Melody transcription from music audio: Approaches and evaluation,” *IEEE Trans. on Audio, Speech and Language Process*, vol. 15, no. 4, pp. 1247–1256, 2007.

[3] R. B. Dannenberg, W. P. Birmingham, B. Pardo, N. Hu, C. Meek, and G. Tzanetakis, “A comparative evaluation of search techniques for query- by-humming usig the musart testbed,” *J. of the American Soc. for Inform. Science and Technology*, vol. 58, no. 5, pp. 687–701, 2007.

[4] A. Mesaros, T. Virtanen, and A. Klapuri, “Singer identification in poly- phonic music using vocal separation and pattern recognition methods,” in *Proc. 8th Int. Conf. on Music Inform. Retrieval*, 2007, pp. 375–378.

[5] M. Ryynanen and A. Klapuri, “Automatic transcription of melody, bass line, and chords in polyphonic music,” *Computer Music J.*, vol. 32, no. 3, pp. 72–86, 2008.

[6] M. Goto, “A real-time music-scene-description system: predominant- f0 estimation for detecting melody and bass lines in real-world audio signals,” *Speech Communication*, vol. 43, no. 3, pp. 311–329, 2004.

[7] S. Koga, T. Inoue, and M. Fukumoto, “A Proposal for Intervention by User in Interactive Genetic Algorithm for Creation of Music Melody,” *2013 International Conference on Biometrics and Kansei Engineering*, pp. 129–132, Jul. 2013. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6603488

[8] Y. Journal, “A GENETIC ALGORITHM FOR COMPOSING MUSIC Dragan MATI Ć,” vol. 20, no. 1, pp. 157–177, 2010.

[9] G. Reis, N. Fonseca, F. de Vega, and A. A. Ferreira, “Hybrid genetic algorithm based on gene fragment competition for polyphonic music transcription,” in *Proc. Conf. applications of evolutionary computing*, 2008, pp. 305–314.

[10] B. Rafael, M. Affenzeller, and S. Wagner, “Application of an Island Model Genetic Algorithm for a Multi-track Music Segmentation Problem,” pp. 13–24.

[11] J.-C. Hung and A.-C. Chang, “Combining genetic algorithm and iterative MUSIC searching DOA estimation for the CDMA system,” *Expert Systems with Applications*, vol. 38, no. 3, pp. 1895–1902, Mar. 2011. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0957417410007499

[12] A. D. Cheveigne and H. Kawahara, “Yin, a fundamental frequency estimator for speech and music,” *Journal of the Acoustical Society of America*, vol. 111, no. 4, pp. 1917–1930, 2002.

[13] G. Kennedy and B. Kernfeld, *”Aebersold, Jamey” in B. Kernfeld. The new Grove dictionary of jazz.* New York: Grove’s Dictionaries Inc, 2002, vol. 1.

[14] W. Marshall, *Best of Jazz.* Milwaukee, W, USA: Hall Leonard, 2000.

[15] D. Bogdanov, N. Wack, E. Gómez, S. Gulati, P. Herrera, O. Mayor, G. Roma, J. Salamon, J. Zapata, and X. Serra, "Essentia: an audio analysis library for music information retrieval," in *International Society for Music Information Retrieval Conference (ISMIR'13)*, Curitiba, Brazil, 04/11/2013 2013, pp. 493–498.

[16] MATLAB, *version 7.12.0 (R2010a)*. Natick, Massachusetts: The MathWorks Inc., 2011.