A Genetic Algorithm Approach with Harmonic Structure Evolution for Polyphonic Music Transcription

Gustavo Reis†, Nuno Fonseca†, Francisco Fernandez, Anibal Ferreira
†School of Technology and Management, University of Extremadura, University of Porto
†Polytechnic Institute of Leiria - Portugal, Spain, Portugal
{gustavo.reis, nfonseca}@estg.ipleiria.pt, fcofdez@unex.es, ajf@fe.up.pt

***Abstract*— *This paper presents a Genetic Algorithm approach with Harmonic Structure Evolution for Polyphonic Music Transcription. Automatic Music Transcription is a very complex problem that continues waiting for solutions due to the harmonic complexity of musical sounds. More traditional approaches try to extract the information directly from the audio stream, but by taking into account that a polyphonic audio stream is no more than a combination of several musical notes, music transcription can be addressed as a search space problem where the goal is to find the sequence of notes that best models our audio signal. By taking advantage of the genetic algorithms to explore large search spaces we present a new approach to the music transcription problem. In order to reduce the harmonic overfitting several techniques were used including the encoding of the harmonic structure of the internal synthesizer inside the individual's genotype as a way to evolve towards the instrument played on the original audio signal. The results obtained in polyphonic piano transcriptions show the feasibility of the approach.**

Keywords*— *Automatic Music Transcription, Genetic Algorithms, Harmonic Overfitting, Harmonic Structure Evolution.

I. Introduction

Automatic music transcription is the process in which a computer program extracts the instrument's partitures or score from a given song or acoustic signal. Therefore the term *Polyphonic Music Transcription* refers to the automatic transcription of music where there is more than one sound occurring at the same time: multiple notes on a single instrument (e.g. piano music), single notes in multiple instruments, etc. Usually only pitched musical instruments are considered. Music transcription is a very difficult problem from the musical point of view (it can only be addressed by the most skilled musicians) and also from the computational point of view: although there has been much research devoted to pitch estimation, it is still an unsolved problem even for monophonic signals (as reviewed by Gomez et al.[1]). The fact is that polyphonic music creates a complex frequency lattice that is computationally infeasible to deconstruct.

A. Related Work

Traditional approaches to Automatic Music Transcription try to extract the information directly from audio source signal. Almost since the first works in polyphonic music transcription by Moorer[2] and Piszcalski & Galler [3], polyphonic music transcription systems rely on the analysis of information in the frequency domain. Klapuri [4] uses iterative calculation of predominant fundamental frequencies in separate frequency bands and Martin [5] uses blackboard systems. There are also techniques which use the principles of human auditory organization for pitch analysis, as implemented by Kashino et al. [6] by means of a Bayesian probability network, where bottom-up signal analysis could be integrated with temporal and musical predictions, and by Wamsley et al. [7], [8], that use the Bayesian probabilistic framework to estimate the harmonic model parameters jointly for a certain number of frames. The use of a hidden Markov Model and spectral feature vectors were proposed by Raphael [9] to describe chord sequences in piano music signals. Neural Networks were used by Carreras et al. [10] for spectral-based harmonic decompositions of signals. Marolt [11] uses networks of adaptive oscillators to track partials over time. A physical model of the piano was used by Ortiz et al. [12] to generate spectral patterns in order to compare them with the incoming spectral data.

In 2001 Garcia[13] addressed the Polyphonic Pitch Detection problem as a search space problem. This method took advantage of the power of genetic algorithms [14] to explore very large search spaces with numerous local optima. Usually search space approaches are not addressed in music transcription problems due to the huge size of the search space, nevertheless genetic algorithms have proven to be an excellent tool to find solutions in extremely large search spaces, since they only need to use a very small subset of the entire search space. The goal of a search space approach is to find the set of musical notes that best models the polyphonic signal. Reis and Fernandez[15] combined both Genetic Algorithms and Sound Synthesis for automatic transcription and later Reis et al.[16] applied Genetic Algorithms for solving polyphonic transcriptions in real audio using sample based techniques for the internal synthesizer with previously recorded piano samples. The major drawback of this approach is the harmonic overfitting [16]: during the evolutionary process the algorithm tends to create additional notes (typically with low amplitudes and in harmonic locations) in order to minimize the spectral differences between the piano played on the original audio song and the internal synthesizer.

B. Genetic Algorithms

Genetic Algorithms (GAs) are a subclass of the Evolutionary Algorithms (EAs). Genetic Algorithms are optimization, search and learning algorithms inspired by the Darwin's theory of evolution that uses principles of inheritance, mutation, selection and crossover (also called recombination).

978-1-4244-3555-5/08/$25.00 ©2008 IEEE

The main idea behind Genetic Algorithms is to have a set of candidate solutions (individuals) which evolve towards the desired solution. In each iteration (generation) those candidate solutions are evaluated according to their quality (fitness). The worst ones are discarded and the best will generate new candidate solutions, which result by merging (recombination) their parents characteristics (genes) and applying minor variations (mutation). Candidate solutions with best quality will tend to live longer and to generate better solutions, improving the robustness of the algorithm.

Genetic Algorithms model the evolution process as a succession of gene changes, with solutions analogous to chromosomes. The entire search space is explored by applying transformations to these candidate solutions, just has it happens in the living beings: recombination, selection and mutation [17]. The evolution usually starts from an initial population of randomly generated individuals and happens in generations. In each generation, the fitness of every individual in the population is evaluated, multiple individuals are selected from the current population (based on their fitness), and modified (recombined and possibly mutated) to form a new population. The new population is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population. If the algorithm has terminated due to a maximum number of generations, a satisfactory solution may or may not have been reached (see algorithm below).

1: $t \leftarrow 0$
2: $GenerateInitialPopulation(P(t))$
3: $Evaluate(P(t))$
4: **while** $TerminationCriteriaIsNotMet$ **do**
5: $\quad P'(t) \leftarrow Select(P(t))$
6: $\quad P''(t) \leftarrow ApplyRecombinationOperator(P'(t))$
7: $\quad P'''(t) \leftarrow ApplyMutationOperator(P''(t))$
8: $\quad P(t+1) \leftarrow CreateNextPopulation(P(t), P'''(t))$
9: $\quad Evaluate(P(t+1))$
10: $\quad t \leftarrow t+1$
11: **end while**
12: **return** $BestIndividual$

Genetic Algorithms constitute one of the most complete paradigm on the Evolutionary Computation, since they resume in a natural way all the fundamental ideas of natural evolution. Genetic Algorithms are also very flexible, thus it is easy to adopt new ideas coming from any evolutionary computation sub-field. They are also easy to hybridize with other paradigms which are not related with Evolutionary Computation, like local search (for instance, Memetic Evolutionary Algorithms [18] are hybrid Genetic Algorithms with local search operators). Genetic algorithms find application in biogenetics, computer science, engineering, economics, chemistry, manufacturing, mathematics, physics and other fields.

This paper presents a new approach to the music transcription problem using Genetic Algorithms with Harmonic Structure evolution. Harmonic Structure evolution is used as a mean to avoid the harmonic overfitting: the individuals's harmonic structure evolve towards the instrument played on the original audio signal, matching not only the original notes but also the harmonic structure of the instrument playing in the original song.

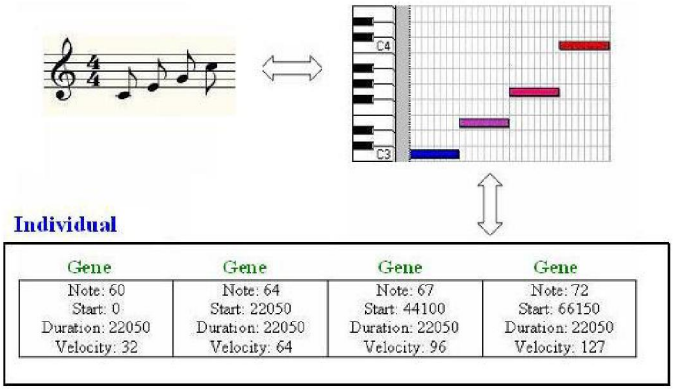

Fig. 1. Encoding of the individuals.

The rest of the paper is structured in the following way: Section 2 describes our proposal while Section 3 presents our experiments and results. Finally Section 4 summarizes our conclusions.

II. Genetic Algorithm Approach

Although genetic algorithms have been employed for signal processing [19], the nature of music transcription problem is different from a standard signal processing, as we have explained before. In order to apply a Genetic Algorithm approach to automatic music transcription problem there are several important considerations that must be addressed: How do we encode each Individual (candidate solution)? How will they breed? How they will be selected for breeding? How shall we initialize the population? What kind of mutation operators should we have? How should we evaluate each individual? How to avoid the harmonic overfitting, which is the major drawback in previous GA approaches [16] to the problem?

A. Individual Encoding

Each individual or chromosome corresponds to a candidate solution and is made of a sequence note events (similar to MIDI [20] structure), where each note will have: pitch, onset, duration[1] and velocity[2]. It might also be useful to have the timbre or the corresponding instrument associated with each note. Since each individual is a set of notes, the number of notes (genes) will most likely be different from individual to individual: the number of genes is not fixed. Figure 1 illustrates the individual encoding.

[1] $duration = offset - onset$
[2] Velocity as in MIDI terminology, as a dynamics representation

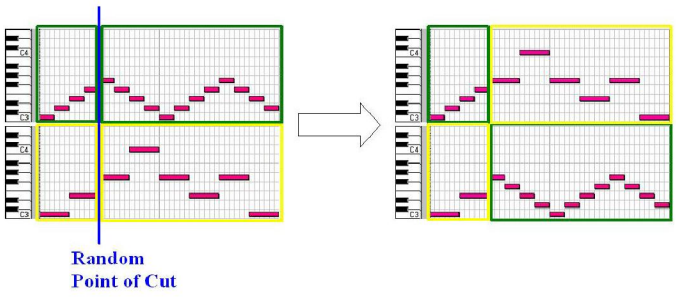

Fig. 2. Recombination operator.

Fig. 3. Starting Population Process.

B. Recombination Operator - Breeding

For the recombination operator we decided to implement an operator based on the classic "1 point of cut" crossover [17]. Since the number of genes (notes) is not the same in both parents a point of cut is randomly chosen in time, eventually splitting note events that start before and end after that point. Two new children are breed, with a different part of each parent (see Figure 2).

C. Selection Operator - Fathers Selection

The selection operator is the deterministic tournament [17], with tournament size = 5. In the deterministic tournament several individuals (in our case 5) are randomly selected from the population, and then the most fit (with best fitness value) of those individuals is chosen to be one of the fathers. The same process happens for choosing the other father.

D. Population Initialization

The initialization of the population process starts by creating an individual based on the FFT peaks. For each time frame of the original audio, the frequency with the highest peak creates (or maintains) a note with the same fundamental frequency. This might lead to several octave errors but since this is only the starting point it does not really matter. This individual passes trough 80 forced mutations to change his events (genes) in terms of velocity, duration and start to overcome decay and levels differences between the original instrument and the internal synthesizer. The additional individual population is created based on the initial individual, after 10 forced mutations. Figure 3 illustrates this process.

E. Mutation

The mutation operator forces some changes inside the individual (for keeping biodiversity) like: deletion of note events, creation of new note events (totally random, based on existing events or based in the existing frequency peaks) or changing the existing note events, modifying the note, start time, duration and dynamics. Several mutations were implemented: note change (± 1 octave, $\pm$ half tone), onset (up to $\pm$ 0.5 second change), duration (from 50% to 150%), velocity (up to $\pm$ 16 in a scale of 128), event split (split in two with a silent between), event remove, new event (random or event duplication with different note). Besides these event mutations, there are 2 mutation operators (with lower probability) that are applied equally to all events: velocity change (up to $\pm$ 4 in a scale of 128) and duration (from 50% to 150%).

F. Fitness Function - Individual Evaluation

To measure the quality of each individual and ensure that the most fit are the ones who will reproduce (survival of the fittest) it is necessary to compare how each individual sounds like with the original audio stream. Since each individual is encoded as a set of notes, each note has to pass through an internal synthesizer made of a simple sampler with piano samples of a Bosendorfer piano and a Steinway piano.

Both the original stream and the synthesized one are cut in time frames with 4096 samples (fs = 44.100kHz) and an overlapping of 75%. A Hanning window [21] is applied before FFT to decrease spectrum leakage. The fitness values are based on the difference between the FFT bins over time (Equation 1). Additional work has been done in exploring other fitness domains (like: FFT with logarithmic scale, Cepstrum, SACF, ACF, etc.), that are not shown here due to the paper size limitations, but to the moment, linear FFT differences presented the best results so far.

$$Fitness = \sum_{t=0}^{tmax} \sum_{f=27.5Hz}^{\frac{fs}{2}} \frac{||O(t,f)| - |X(t,f)||}{f} \quad (1)$$

Fitness value is computed from frame slot 0 to $tmax$, traversing all time from the beginning to the end. The lower part of the frequency spectrum is limited to the fundamental frequency of the first note of the piano's keyboard, i.e., the fitness function is created using the difference of the FFT bins in the frequency range between the lowest piano's keyboard note (from MIDI-note 21 - 27,5 Hz - to 22.100 Hz - the Nyquist frequency of 44.100 kHz).

Although GAs usually consider higher fitness values for the best individuals, our approach considers the opposite since we are trying to minimize the error between the original audio stream and the individual's audio stream.

1) Frequency Normalization: Since FFT bins are not equally distributed by all octaves (higher octaves are spread over much more FFT bins than the lower octaves), it is important to create a frequency normalization process, to avoid the increase of impact of higher octaves. The division by f in Equation 1 acts as a frequency normalization. The $|O(t,f)|$ is the magnitude of frequency f at time frame t in the source audio signal, and $|X(t,f)|$ is the same for each individual's audio signal.

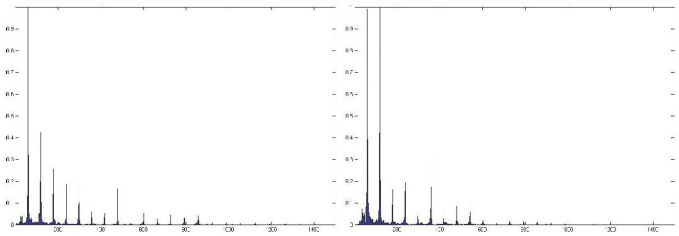

Fig. 4. Spectrum difference between two piano sounds: the internal synthesizer is the piano from left and the original piano is the piano from right.

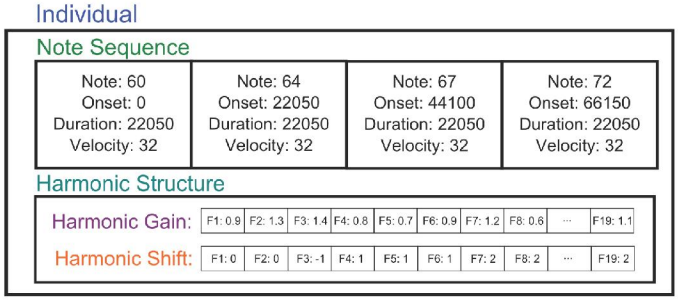

Fig. 5. Encoding of the Individual with the Harmonic Structure.

G. Avoid Harmonic Overfitting

In previous proposed Genetic Algorithm approaches to Polyphonic Music Transcription, Reis et al.[16] noticed that the genetic algorithm tends to create additional notes (with lower amplitudes) in harmonic locations of the original notes to overcome the timbre differences between the internal samples and original piano sounds. Despite the fitness values continues to decrease through generations, the quality of their results started to decrease after some point, mainly because of a harmonic overfitting. Detected notes continued there (shown by recall values) but many additional notes begin to emerge, dropping the precision value.

The fact of these additional notes have low amplitude and are in harmonic locations, many times even with similar onsets, strongly decreases their impact from the perception point of view. Nevertheless, for the metrics or in situations where the dynamic information is discarded (for instance: creating music sheets), these errors are very undesirable.

1) Harmonic Series Evolution: Although we use piano sounds on the original audio stream and in the internal synthesizer, those pianos are not the same. They will present differences in their harmonic structure and envelope/release behavior. For instance: lets consider that the original piano has a spectrum like the right part of the Figure 4 and the internal synthesizer has a spectrum like the left part of the Figure 4. The F1 of original piano has much higher amplitude than the F1 of the internal synthesizer which may lead the algorithm to create an additional note on the first harmonic of the original note to overcome this difference. Therefore, to minimize these errors between the transcriptions, the genetic algorithm will create additional notes in those frequency regions, until the lowest possible error. Those additional notes tend to have short durations and low velocity values, just the necessary to compensate the timbre differences between both pianos. In the end, what the algorithm achieves is a sound that, from the perception, is very near the original audio, but since the focus of the problem is to discover only the notes that are played in the original song, it is necessary the removal of all those additional notes.

One solution to this problem is to create harmonic gains, that boost or cut the value of the first 20 harmonic peaks. Almost like an equalizer, but instead of operating on fixed frequency bands, it operates on each note harmonic. From the implementation point of view, this is not done with real filters, but by changing the values of the FFT bins that correspond to the note harmonic locations.

This means, that each individual, besides having a sequence of note events as their candidate solution to the problem, also include additional parameters to help the synthesizer to get a timbre more similar with the original instrument (see Figure 5). The gain of the Fundamental Frequency of the note - F0 - is always 1.

2) Inharmonicy Evolution: Sometimes the harmonics are not located in integer multiples of the Fundamental Frequency. Those harmonics are often shifted some bins to the left or to right of the real multiple corresponding frequency bin. To solve this problem, the amount of shifting for each harmonic in the harmonic structure was also encoded within the Individual's genotype, among with the harmonic structure (see Figure 5). This way each Individual had his own synthesizer with a complete evolving harmonic structure towards the original synthesizer and also with evolving notes towards the original song's notes. The shift of the Fundamental Frequency of the note - F0 - is always 0.

By introducing the harmonic gains and the harmonic shifts in the individuals genotype, recombination and mutation must support those additional features:

Recombination
: The recombination operator still splits the note events by applying a random point of cut in time as it already did. Two more random points of cut are used: one for splitting the harmonic series and another for splitting the harmonic shifting.

Mutation
: Regarding mutation operator, two new mutations were created: one that changes and harmonic gain up to ± 0.50 and another which changes the harmonic overfitting up to $\pm 3 bins$.

3) Note discard: Other implemented feature for avoiding harmonic overfitting is note discard. The idea is that within the note local range, most notes have similar dynamics. Considering that each note has a dynamic scale between 1 and 128, this feature will discard all notes present at transcription that

have a dynamic difference of 20 between its note dynamics and the maximum value of dynamics of other notes existing during the note duration.

4) Dynamic Range: Since noise, weak harmonics or harmonics frequency neighborhood can also lead to harmonic overfitting a dynamic range feature is also implemented. In each time frame the FFT bin with the highest value is used as reference, and all bins with values 40dBs below this reference have their values set to 0.

III. Experiments and Results

All the tests were based on an audio fragment with the first 30 seconds audio file of Mozart's Piano Sonata n. 17 in B flat K570, played with a Bechstein piano. To test the proposed approach two banks of tests were made using Bosendorfer piano samples or Steinway piano samples inside the internal synthesizer. Since the base audio stream uses different piano sounds from our internal piano synthesizers, they will present differences in their harmonic structure and envelope/release behavior.

Our current approach uses the following parameters: population size of 200, in each generation 100 more individuals are breed, only the best 200 pass to the next generation (elitism), the maximum number of generations is 1500, the probability of crossover is 75%, for mutation is 4% and the note minimal duration is 10 ms.

For better performance the original audio fragment was divided into 5 second fragments and a different run was processed for each fragment. After those 1500 generations runs the results were merged to create the transcribed 30 seconds sequence. The CPU time needed for transcribing a 5 seconds fragment with 1500 generations is around 7 hours, using one core of a Dual Core 2.0 GHz processor.

The used metrics were based on MIREX 2007 [22], and consists of "onset only" and "onset/offset" analysis. In "onset only" a note is considered correctly transcribed if pitch is $\pm\frac{1}{2}$ semitone apart and with onset inside $\pm 50ms$ tolerance. In "onset/offset" a note is considered correct if, besides "onset only" requirements, offset is within $\pm 50ms$ or 20% of the note duration (which is the bigger value).

Results are presented using recall (percentage of original notes that were transcribed), precision (percentage of transcribed notes that were present on the original stream), F-measure and mean overlap ratio. This last parameter is the average of notes overlap ratio, that in each corrected transcribed note, measures the overlap between original and transcribed note (Equation 2).

$$OverlapRatio = \frac{\min(offsets) - \max(onsets)}{\max(offsets) - \min(onsets)} \quad (2)$$

It is also considered that the same note cannot overlap (e.g. two C4 notes paying at the same time) and that notes with duration smaller than 10 ms are discarded.

	Recall	Precision	F-Measure	Overlap Ratio
Bosendorfer "onset only"	0.673	0.726	0.699	0.631
Steinway "onset only"	0.659	0.745	0.699	0.631
Mean "onset only"	0.666	0.736	0.699	0.631
Bosendorfer "onset/offset"	0.369	0.398	0.382	0.798
Steinway "onset/offset"	0.368	0.398	0.383	0.798
Mean "onset/offset"	0.369	0.398	0.383	0.798

Table 1. Results of the proposed approach.

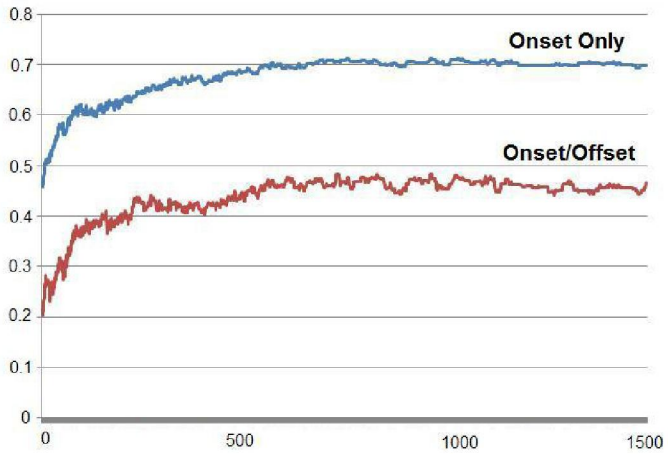

Fig. 6. F-Measure evolution over 1500 generations.

Table 1 shows our results. Figure 6 shows the F-Measure evolution over 1500generations.

Both Bosendorfer piano and Bechstein piano presents similar results. Despite of algorithm started with different timbres (different pianos) it ended with the same results, this happens due to the harmonic structure evolution: both instruments evolve until they match the original piano.

Table III presents the results of the Piano transcription Task ("onset only" metrics) of the MIREX 2007 contest [22] (a Musical Information Retrieval contest organized every year). Figure 7 shows the min, max and average values from transcriptions of 11 different methods (A-K) transcribing 6 different piano pieces. Although the metrics are the same, the used music database is different from ours, which means that is not possible to do a real comparison to decide what is the best approach. Nevertheless it shows that this is a perfectly valid approach to the polyphonic piano transcription.

Despite not being presented, due to paper size limitations, it is obvious that in every test the fitness value of the best individual in each generation is always equal or lower that the one in the prior generation.

IV. Conclusions and Future Work

Although search space approaches are not commonly used on automatic music transcription due to the huge size (al-

	Proposed *	A	B	C	D	E	F	G	H	I	J	K
Precision	0.736	0.169	01.59	0.591	0.541	0.4	0.441	0.672	0.32	0.24	0.72	0.653
Recall	0.666	0.147	0.174	0.651	0.723	0.421	0.361	0.63	0.376	0.194	0.669	0.156
F-Measure	0.699	0.168	0.183	0.602	0.601	0.405	0.386	0.647	0.343	0.186	0.692	0.23
Overlap	0.631	0.438	0.428	0.513	0.543	0.44	0.368	0.615	0.495	0.573	0.606	0.516

Table 2. MIREX 2007

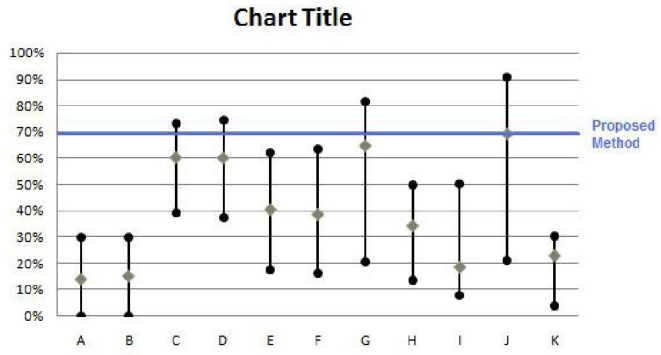

Fig. 7. A piano transcription (onset only) task results from MIREX 2007 and proposed method (tested with different music database).

most infinite) of the search space, the obtained results have shown that applying Genetic Algorithms to Polyphonic Music Transcription is a feasible approach. These kind of approaches can also be used as a means to improving existing algorithms (placing their results as original population, and start improving from there).

In the future the multitimbral feature will be implemented to support additional instruments besides piano, by using several intruments samples (like a General MIDI sampler) or by using modular synthesizer whose parameters evolve to simulate several different musical instruments.

REFERENCES

[1] E. Gómez, A. Klaupuri, and B. Meudic, "Melody description and extraction in the context of music content processing," *Journal of New Music Research*, vol. 32, no. 1, 2003.

[2] J. A. Moorer, "On the transcription of musical sound by computer," *Computer Music journal*, vol. 1, no. 4, pp. 32–38, 1977.

[3] M. Piszczalski and B. A. Galler, "Automatic music transcription," *Computer Music Journal*, vol. 1, no. 4, pp. 24–31, 1977.

[4] A. Klapuri, "Multiplitch analysis of polyphonic music and speech signals using an auditory model," *IEEE Transactions on Audio and Language Processing*, vol. 16, no. 2, pp. 255–264, February 2008.

[5] K. D. Martin, "A blackboard system for automatic transcription of simple polyphonic music," Tech. Rep. 385, MIT Media Lab, Perceptual Computing Section, Tech. Rep., July 1996.

[6] K. Kashino, K. Nakadai, T. Kinoshita, and H. Tanaka, "Organization of hierarchical perceptual sounds: Music scene analysis with autonomous processing modules and a quantitative information integration mechanism," in *IJCAI*, 1995, pp. 158–164.

[7] P. Walmsley, S. Godsill, and P. Rayner, "Bayesian graphical models for polyphonic pitch tracking," 1999.

[8] ——, "Polyphonic pitch tracking using joint bayesian estimation of multiple frame parameters," 1999.

[9] C. Raphael, "Automatic transcription of piano music," in *Proceedings of ISMIR 2002*, 2002,.

[10] C. F., "Automatic harmonic description of musical signals using schema-based chord decomposition," *Journal of New Music Research*, vol. 28, pp. 310–333(24), December 1999.

[11] M. Marolt, "Networks of adaptive oscillators for partial tracking and transcription of music recordings," *Journal of New Music Research*, vol. 33, pp. 49–59(11), March 01, 2004.

[12] L. Ortiz-Berenguer, F. Casajus-Quiros, and S. Torres-Guijarro, "Multiple piano note identification using a spectral matching method with derived patterns," *Journal of the Audio Engineering Society*, vol. 53, no. 1/2, pp. 32–43, January/Frebuary 2005.

[13] G. Garcia, "A genetic search technique for polyphonic pitch detection," in *Proceedings of the International Computer Music Conference (ICMC)*, Havana, Cuba, Sept. 2001.

[14] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. The MIT Press, April 1992.

[15] G. Reis and F. Fernandez, "Electronic synthesis using genetic algorithms for automatic music transcription," in *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*. New York, NY, USA: ACM Press, 2007, pp. 1959–1966.

[16] G. Reis, N. Fonseca, and F. Fernandez, "Genetic algorithm approach to polyphonic music transcription," *Proceedings of WISP 2007 IEEE International Symposium on Intelligent Signal Processing*, pp. 321–326, 2007.

[17] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Professional, January 1989.

[18] W. Hart, N. Krasnogor, and J. Smith, *Recent Advances in Memetic Algorithms*. Springer, 2004, ch. Memetic Evolutionary Algorithms.

[19] J. T. Alender, "An indexed bibliography of genetic algorithms in signal and image processing," University of Vaasa, Department of Information Technology and Production Economincs, report 94-1-SIGNAL, 1995.

[20] *The Complete MIDI 1.0 Detailed Specification*, MIDI Manufacturers Association, September 1995.

[21] F. J. Harris, "On the use of windows for harmonic analysis with the discrete fourier transform," *Proceedings of the IEEE*, vol. 66, no. 1, pp. 51–83, 1978.

[22] "Music information retrieval evaluation exchange (mirex 2007)," http://www.music-ir.org/mirex/2007/index.php, 2007.