A GENETIC ALGORITHM FOR THE AUTOMATIC GENERATION OF PLAYABLE GUITAR TABLATURE

D.R. Tuohy and W.D. Potter
Artificial Intelligence Center
University of Georgia
Athens, Georgia USA

ABSTRACT

This paper describes a method for mapping a sequence of notes to a set of guitar fretboard positions (tablature). The method uses a Genetic Algorithm (GA) to find playable tablature through the use of a fitness function that assesses the playability of a given set of fretboard positions. Tests of the algorithm on a variety of compositions demonstrate an excellent ability of the GA to discover easily playable tablature that maintains a high degree of consistency with published tablatures transcribed by humans. The algorithm is also shown to generally outperform commercial software designed for the same purpose. We conclude that the GA can reliably produce good tablature for any piece of guitar music.

1. INTRODUCTION

Some instruments, such as the piano, have only one way to produce a given pitch. To play a piece of music on a piano, one need only read the notes sequentially from the page and depress the corresponding keys in order. Stringed instruments, however, require a great deal of experience and decision making on the part of the performer. A given note on the guitar may have as many as six different positions on the fretboard on which it can be produced. Figure 1 shows a guitar fretboard, and four different fretboard positions on which a third octave "E" can be played. A fretboard position is described by two variables, the string and the fret. To play a piece of music, the performer must decide upon a sequence of fretboard positions that minimize the mechanical difficulty of the piece to at least the point where it is physically possible to be executed. This process is time-consuming and especially difficult for novice and intermediate players and, as a result, the task of reading music from a page as a pianist would is limited only to very advanced guitar players. To address this problem, a musical notation known as tablature was devised. Tablature describes to the performer exactly how a piece of music is to be played by graphically representing the six guitar strings and labeling them with the corresponding frets for each note, in order.

The goal of our research is to automatically discover very good tablature for any piece of music. In the next section we will describe recent commercial and academic attempts to accomplish the same feat and explain their disadvantages. We will then describe our genetic algorithm approach and offer an explanation for why it is best suited for this problem. This will be supported with the analysis of tablatures produced both by our approach and by commercial software. Finally, we will discuss why the reliability of our approach is an advancement towards creating software that can automatically and intelligently arrange music for guitar.

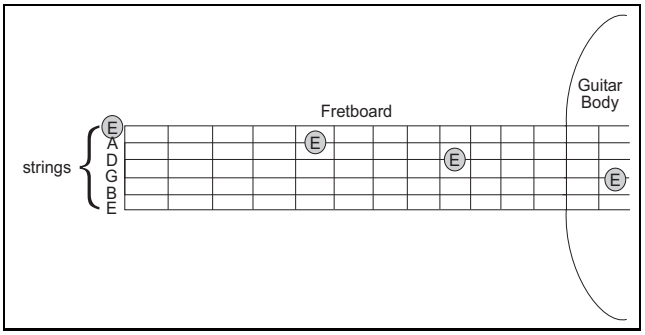

Figure 1. Four different fretboard positions for the 3rd octave "E".

2. BACKGROUND

2.1. Commercial Software

Several programs on the market today convert music into tablature. These programs are, however, notorious for producing unplayable or unnecessarily difficult tablatures[6]. This is because they often depend too heavily upon rules of music theory relating to harmony and guitar composition[14]. For example, "open" strings are considered to be easier to play on guitar because they do not require depressing the string on a fret. However, it is often easier for a note to be played on a depressed string if that string happens to already be depressed at that point in the music. Software that we tested also seems to take the inadvisable approach of creating tablature note by note, rather than devising a strategy for the piece as a whole. In other words, when the algorithm makes a decision about where to place a given note on the fretboard, it consults only the locations of earlier notes and gives no consideration to where future notes are likely to be played. This is understandable, as

doing otherwise is exceptionally difficult for conventional algorithms (how does one design a section of tablature to fit with a section that has not yet been created?). Our genetic algorithm approach, however, is well suited to this task, as will be explained later. For these reasons, current commercial software often depends upon the user to edit a tablature after it has been generated[14].

2.2. Academic Research

The process of creating high quality guitar tablature has traditionally required human decision-making skills. It requires weighing numerous factors that contribute to difficulty, and deciding upon the best approach from among several[5]. A method for discovering playable tablature was described by Samir I. Sayegh of Purdue University[12]. His "optimum path paradigm" describes a sequence of fretboard positions as a sequence of hand states, the goal being to find the optimum path through the states. More recently, there have been algorithms that build upon Sayegh's approach at the University of Torino[9] and the University of Victoria[10]. Both approaches have produced excellent tablatures for the pieces on which they were tested, but both have limitations. The prototype from the University of Torino cannot account for situations where more than one note needs to be played simultaneously (a chord). The program from the University of Victoria was tested by either manually tuning a set of cost function weights to more accurately suit a piece, or by "training" the algorithm using other pieces of similar styles with known (published) tablatures. So while the former can produce a tablature directly from a sequence of notes, it does not handle chords, and while the latter can handle any composition, it requires customized training for each piece. Another group from Doshisha University reports success in generating tablature superior to that of commercial software, but once more the program is limited to producing tablature for melodies without chords[6].

3. TABLATURE GENERATOR OVERVIEW

The duty of the generator is to accept as an input some representation of the note sequence that defines a piece of music and to generate a tablature as output. Our generator currently accepts a text string of all the notes in order, but could be modified to extract notes from a MIDI file or an ASCII tablature. Figure 2 is an example of an ASCII tablature of "Stairway to Heaven" by Jimmy Page and Robert Plant[1]. The strings are represented as dashed lines and the numbers indicate the fret on which the string needs to be depressed. A zero indicates that the string is to be played without depressing a fret. After the generator has evolved a playable tablature using the genetic algorithm, the tablature is printed to the screen or to a file in the ASCII format.

```
e|- - - 5-7- - -7-8- - -8-2- - -2-0- - - -0- - - - - -|
B|- - 5- - -5- - - -5- - - -3- - - -1- -1 - 1- -0-1-1-|
G|- 5- - - - -5 - - - 5- - - -2- - - -2- - - -2-0-2-2-|
D|7- - - -6- - -  5- - - -4-- - - 3- - - - - - - - - -|
A|- - - - - - - - - - - - - - - - - - - - - - - 2-0-0-|
E|- - - - - - - - - - - - -  - - - - - - - - - - - - -|
```

Figure 2. ASCII tablature of "Stairway to Heaven" by Jimmy Page and Robert Plant[1].

4. THE GENETIC ALGORITHM APPROACH FOR TABLATURE GENERATION

4.1. The Genetic Algorithm

A Genetic Algorithm (GA)[7][8] is a stochastic search algorithm that aims to find good solutions to difficult problems by applying the principles of evolutionary biology. Evolutionary concepts such as natural selection, mutation and crossover are applied to a "population" of solutions in order to evolve a very good solution. Problems suited for a GA are those whose number of possible solutions (search space) is so large that finding good solutions by exhaustive search techniques is computationally impractical. The genetic algorithm is also useful for tackling search spaces with many scattered maxima and minima. Imagine a search space as a mountain range, where the goal is to get to the highest point(say, Mt. Everest). Each mountain peak is the local optimal solution for the section of the search space it occupies, but Mt. Everest is the global optimal solution and is considerably higher than many of the other peaks. Local optima in the search space can lead many search algorithms to prematurely conclude that the best solution has been found. Genetic Algorithms, however, are very effective at exploring the search space. This property of GAs is essential to an intelligent search for playable guitar tablature.

4.2. Why the GA is useful for this problem

4.2.1. Large search space with many optima

Because a note can be played on many different positions on the fretboard, the number of possible tablatures for a piece of music is enormous. Estimating that a note can be played in an average of three fretboard positions, a song with n notes has 3^n possible tablatures. A phrase with just 40 notes, for example, would have approximately $1.2 * 10^{19}$ possible tablatures. However, only a small percentage are playable and only a handful could be considered to be desirable. This property of stringed instruments creates an excellent search space for a Genetic Algorithm. Additionally, the search space for a piece of music can have a large number of optima throughout. A given piece of music may have good tablatures nearly identical in terms of difficulty but with very few fretboard positions in common. These tablatures are reflected in the search space as widely scattered optima.

4.2.2. *Implicit Parallelism*

The implicit parallelism of a genetic algorithm refers to its ability to search for the best "parts" of an individual simultaneously[7]. This is not to say that there is any sort of distributed processing being performed, but only that a GA tries to improve upon each gene in the individuals simultaneously. The beginning of the ideal tablature may be located in one individual, the middle in another, and the end in yet another. By themselves, perhaps these parts only make the individual in which they are contained only slightly more fit than other individuals, but that is all that is needed. These individuals are then more likely to be selected for mating, and crossover can combine the beneficial parts into an offspring even more fit than either of the parents. When the GA has finished, it is likely that parts of the tablature at the beginning of the piece have been directly influenced by the parts in the middle or at the end. This results in the robust search capability characteristic of GAs. This is vastly preferable to creating tablature linearly from beginning to end, because decisions made at the beginning of a tablature may turn out to be very inappropriate when the tablature has been completed. This weakness will be demonstrated to be a major downfall for other approaches.

4.3. Implementation

The population for our GA is a collection of tablatures that are valid, though not necessarily desirable, for a given piece of music. The initial population is generated randomly. A tablature "chromosome" is defined as a sequence of chords. A chord is a "gene" and consists of fretboard positions for all the notes in that chord. A chord, in this sense, is any combination of notes that are played simultaneously. Quite often notes are played by themselves, and these are considered to be chords composed of only one note and hence only one position on the fretboard.

The parameters for the GA are those which have led to convergence on the most fit individuals. A single run of the GA takes only a few seconds on a fast PC, so we can afford to fortify the GA with a healthy population size of 300 individuals. The GA is generational, which means that each generation is replaced by the subsequent generation all at once, rather than replacing the tablatures individually. Parents are selected using Binary Tournament Selection, whereby two potential parents are selected randomly and the most fit of the two becomes a mate. Once two mates have been selected by this process, there is a sixty percent chance that they will create children through two-point crossover. Two-point crossover divides the chromosomes of the parents into three sections of random size and gives a child different sections from each parent. The sections that are not used to create the first child are then combined to make a second child. Forty percent of the time the parents do not crossover and simply become "children" themselves. After the children have been generated, whether through crossover or not, they then face a seven percent chance of mutation. If a child is selected for mutation, then a random chord in the tablature is switched to an alternative fingering. These operators are demonstrated in figure 3. The process of "selection-crossover-mutation" is repeated until 300 new individuals have been created for the next generation. The GA terminates when it seems to have converged upon a solution.

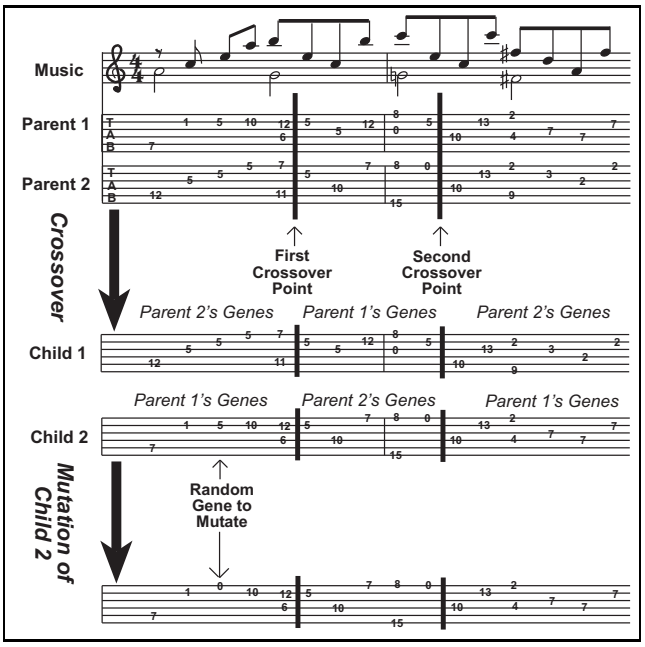

Figure 3. Obtaining children from two parents. Music from "Stairway to Heaven"[1].

To increase the overall performance of the program, we found it useful to run the GA several times for 100 generations and save all of the most fit individuals found. We then run the GA once more with a population composed predominately of these individuals. Very often this final run will produce an individual more fit than any other found so far. This is often, but not always, the optimal individual as defined by our fitness function. The chance that this individual is optimal varies with the length of the piece and the complexity of the corresponding search space.

4.4. Fitness Function

The fitness function determines the fitness of an individual. A more fit individual is more likely to be selected for breeding. Our function analyzes a tablature in many different ways to assess its playability. This assessment is partially informed using the analysis of finger-positioning complexity by Heijink and Meulenbroek[5]. The function can be thought of as calculating two separate classes of tablature complexity, difficulty of hand/finger movement and difficulty of hand/finger manipulation.

4.4.1. *Hand Movement*

These calculations essentially estimate the total amount of hand and finger movement across the fretboard that is nec-

essary to perform a tablature by executing simple calculations on the fret numbers. The more movement that is required, the more the function penalizes the tablature's fitness. Some factors whose values penalize the function include the total number of times strings must be depressed by a finger, the fret-wise distance between adjacent notes or chords, and the fret-wise distance of a position from the average of the surrounding notes. A factor that rewards the fitness of the tablature is the number of open notes, as they do not require any lateral movement by the left hand to be executed.

4.4.2. Hand Manipulation

This portion of the fitness function attempts to analyze what the left hand itself is doing and assess the difficulty of the requisite hand positions. There are two predominant methods by which a guitarist can finger multiple notes. The first is the open chord method, where each note is depressed individually with one of the four fingers (thumbs are generally not used). The second method is to place the index finger across a fret over several strings, and to use the remaining three fingers for any other notes on higher frets. This is known as a barre chord. The function looks for both of these chord types and rewards the fitness of a tablature proportionally to the chord's difficulty and how long the position can be held without having to move the hand.

5. EXPERIMENT SETUP AND RESULTS

5.1. Experiment Setup

A common metric for establishing the success or failure of a tablature generator is the percentage of fretboard positions in the generated tablature that are consistent with a published tablature. Over a very large body of work this should provide a rough indication of the generator's reliability, but the method is inherently flawed. While playability is the main concern for a human when creating tablature, it is not the only concern. In cases where notes can be placed on the fretboard in multiple positions without significant differences in playability, the position chosen by a professional could seem essentially arbitrary. Because each guitar string has a slight but noticeable difference in timbre, tone quality becomes a factor for notes with more than one viable position. If differences in playability are very slight, the chosen position could even be little more than the personal preference of the tablature creator. The effect of this problem is that a tablature that differs from the published tablature by only one note could conceivably be unplayable while a tablature differing at every position could be just as playable.

Therefore, we used two criteria to judge the reliability of our approach. The whole process should *usually* create tablature consistent with published tablature and should *never* create a significantly more difficult tablature than is necessary. We will also compare the tablatures generated by our approach with those created by Guitar Pro 4, a very popular piece of commercial tablature software. This piece of software was chosen for comparison because it generated tablatures for the pieces we looked at that were generally superior to those created by any other software that we tested.

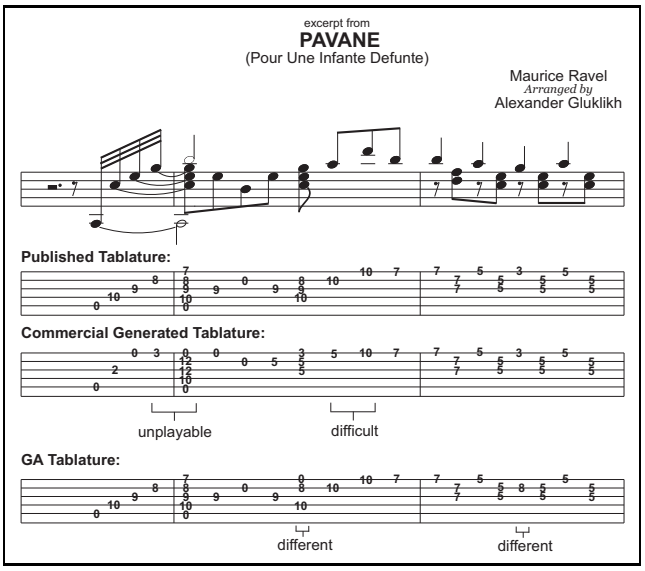

Figure 4. Three tablatures for an excerpt from Pavane by Maurice Ravel[2].

5.2. Results and Conclusion

We ran our GA on a wide variety of musical literature. The GA produced tablatures for contemporary rock groups like Aerosmith and Led Zeppelin as well as for transcriptions of classical pieces by composers such as Bach and Ravel. In all, selections from about 20 pieces of music were tested. The majority of tablature coincided precisely with published tablatures and the algorithm never produced a tablature that was significantly more difficult than the published tablature. We consider the GA a success because it reliably produces playable tablature even when departing from the published versions. Figure 4 shows an example of this. It is an excerpt from Ravel's "Pavane pour une Infante Défunte"[2] and shows three tablatures. The first is the published tablature, the second is generated by the commercial software, and the third is generated by our GA.

The second tablature exemplifies the danger in attempting to produce tablature note by note without looking ahead in the piece of music. The tablature begins the piece low on the fretboard, around the second and third frets. For these first four notes the tablature is actually easier to play than the published tablature. However, the next chord cannot be played low on the fretboard and the software places this chord on frets 10 through 12. Once more the chord is easier to play than that of the published tablature, but there now exists a jump from the bottom of the fretboard to the top. This jump measures about a foot on the fretboard and is to be executed in 1/16th of a second, which is essentially impossible. Later in the tablature is a jump from the

fifth fret to the tenth fret and is to be executed in 1/4th of a second. This is possible but unnecessarily difficult.

The third tablature, the one created by our GA, has two fretboard positions which differ from the published tablature. The first moves a note on the ninth fret to an open string. This has little to no effect on the overall playability of the piece, as it merely requires lifting a finger that needs to be lifted within the next second anyway. The other change moves a note from the third fret to the eighth fret. This change is arguably easier since it allows the performer to use the same barre chord for the last nine notes of the piece without any movement at all. The reason our GA does not make the same mistakes as the commercial generator is that, due to the implicit parallelism of the GA, the different sections are processed in an overlapping fashion. While the GA evaluates the first four positions of the commercially generated tab as the most fit when they are by themselves, when those positions are viewed in context the tablature's fitness suffers and hence tablatures containing those positions are selected less often and eventually become extinct.

Figure 5 illustrates a similar scenario in an excerpt from the jazz standard "As Time Goes By" by Herman Hupfeld. In this case the tablature generated by our GA is the same as the published tablature, but the Guitar Pro software again generates an unplayable tablature. The first eight notes of the piece are placed low on the fretboard and, as in the excerpt from "Pavane", this results in an unplayable jump from the second fret to the tenth fret. The human professional and the GA, however, account for the fact that the hand will have to be high on the fretboard and place early notes in the tablature in that area as well.

The "Pavane" and "As Time Goes By" tablatures illustrate well the faults of conventional approaches to tablature generation and how the GA improves upon them. Mistakes made by the commercial software usually only result in tablatures that are unnecessarily difficult but, as is shown in the examples, sometimes create tablatures that are unplayable. The advantages of our Genetic Algorithm over other approaches becomes apparent when creating tablature for sections of music that require knowledge of later sections to be transcribed correctly. As a result, the GA was the only approach that we tested that was able to reliably produce playable tablature. It did not always maintain consistency with the published tablatures, but deviations proved to be mostly inconsequential to overall playability and never disastrous.

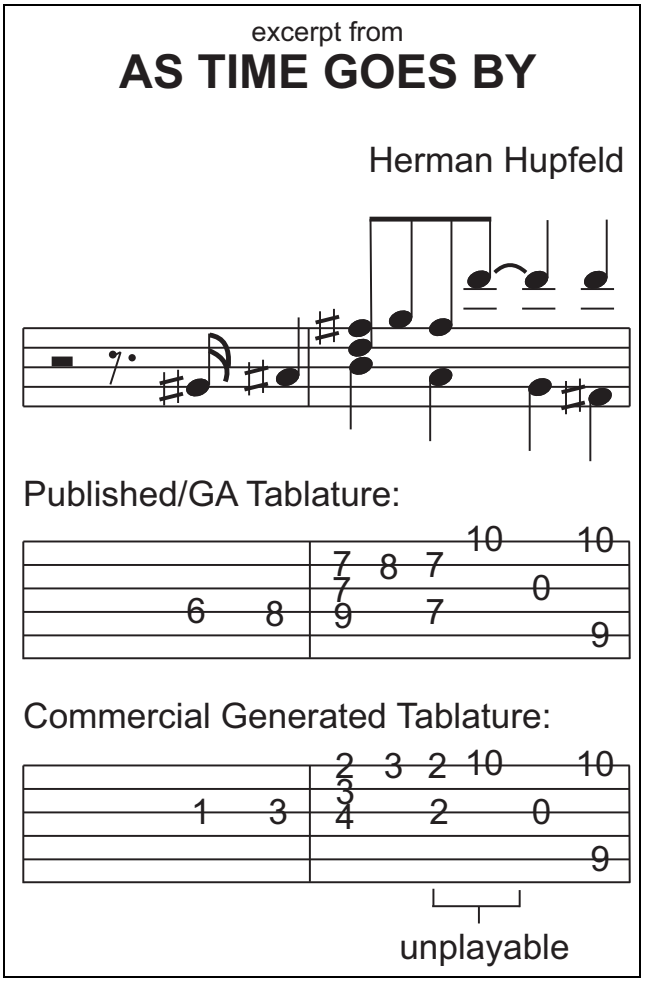

Figure 5. Tablatures for an excerpt from "As Time Goes By" by Herman Hupfeld[2].

6. FURTHER DIRECTIONS

6.1. Arranging Music For Guitar

Our GA is concerned only with choosing the fretboard positions for a given sequence of notes, it does not choose the notes themselves. The process of analyzing a piece of music and determining which notes to play on an instrument is known as arranging. Arranging for the guitar, as well as the piano, is a balance between remaining loyal to a piece of music while also making sure that the piece can

be played on the instrument for which it is being arranged. For example, it would be impossible to completely play Beethoven's fifth symphony on a guitar because there are too many instruments playing far too many notes, and it is the responsibility of the arranger to choose which notes are most important and which can be sacrificed for the sake of playability.

Because tablature generation and evaluation is so problematic, music arranging is very difficult to automate. How can an algorithm evaluate whether or not an arrangement is playable if it does not know what the tablature for the arrangement looks like? Using our GA, it is conceivable that another Genetic Algorithm built on top of ours could evolve arrangements for a guitar using a fitness function composed of our fitness function and a function that evaluates the degree to which an arrangement mimics the original composition. The "optimal" arrangement would be the one which has minimal difficulty and maintains the features of a piece's melody and harmony to the greatest extent. If this approach were to work, it would allow those without the time or expertise to transcribe an arrangement by hand to obtain a playable guitar version of any piece of music they wish by providing the notes the piece contains and specifying important melodic lines where appropriate.

7. REFERENCES

[1] Colgan, B., Stang, A. *Led Zeppelin: Acoustic Classics Vol. 1*. Warner Brothers, Miami, 1995.

[2] Colgan, B., Stang, A. *The Solo Guitar Big Book*. Warner Brothers, Miami, 2001.

[3] Davis, L. *The Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York, 1991.

[4] Guitar Pro. *Guitar Pro 4 Demo*. 2004

[5] Heijink, H., Meulenbroek, R.G.J. "On the complexity of classical guitar playing: functional adaptations to task constraints", *Journal of Motor Behavior*. 34(4), 339-351, 2002.

[6] Miura, M., Hirota, I., Hama, N., Yanigida, M. "Constructing a System for Finger-Position Determination and Tablature Generation for Playing Melodies on Guitars", *Systems and Computers in Japan*. 35(6), 755-763, 2004.

[7] Holland, J. "Genetic Algorithms", *Scientific American*. 267, 44-50, July, 1992.

[8] Liepens, G.E., Potter, W.D. "A Genetic Algorithm Approach to Multiple-Fault Diagnosis", *Handbook of Genetic Algorithms*. 237-250, Van Nostrand Reinhold, New York, 1991.

[9] Radicioni, D., Anselma, L., Lombardo, V. "A segmentation-based prototype to compute string instruments fingering", *Proceedings of the Conference on Interdisciplinary Musicology*. Graz, Austria, 2004.

[10] Radisavljevic, A., Driessen, P. "Path Difference Learning for Guitar Fingering Problem", *Proceedings of the International Computer Music Conference*. Miami, USA, 2004.

[11] Robinson, M. *Fingerpicking J.S. Bach*. Amsco, New York, 1981.

[12] Sayegh, S. "Fingering for String Instruments with the Optimum Path Paradigm", *Computer Music Journal*. 13(6), 76-84, 1989.

[13] Stix, J. "Guitar & Bass Sheet Music", *Guitar Classics VII*. 57-63, Cherry Lane Magazines, Port Chester, N.Y., 1993.

[14] Wang, J., Tsai-Yen, L. "Generating Guitar Scores from a MIDI Source", *International Symposium on Multimedia Information Processing*. Taipei, Taiwan, 1997.

The proposed automatic transcription algorithm extracts es-sential information about the recorded music piece that allows com-parison with a ground truth notation. Hence possible application scenarios are music education software such as Songs2See1 and BandFuse2 as well as music games such as RockSmith3.Â For the case of electric guitar recordings on-sets corresponds to single plucks. The signal part between two plucks is interpreted as a note event. First, seven state-of-the-art onset detection functions (see appendix 8.1) were tested against a separate development set of guitar note recordings (see Sec-tion 5.1) using the same default blocksize and hopsize values. 6 Guitar Tablature Generation. 7 Optical Music Recognition Result Evaluation. 8 Future Work and Conclusion.Â OMR software interprets the sheet music into editable and playable digital form. Normally, the result is saved as a midi le for play back or MusixTeX for music engraving [wik11]. Unlike Optical Character Recognition (OCR) which recognizes the text and parses the words sequentially, OMR should handle more complicated situations such as multiple voices on the same sta which should be played simultaneously.Â Base on the genetic algorithm, this author also explores Evolved Neural Network for tablature generation. â€œTraining data was parsed from an online repository of human-created tablaturesâ€. Genetic algorithm for mapping the note sequences to a set of guitar fretboard positions. Training data will be based on existing tablatures. A musical score can be represented as guitar tablature. The notes can also be generated in different ways by playing different strings with different positioning of the fingers. Metal.Â This system is limited to note transcription of modern songs into guitar tablature and to a particular set of playing styles; namely, plucking, strumming, picking and fingering. To implement a method of conversion of audio-signal to a frequency spectrum. To develop an automatic segmentation algorithm. To convert the frequency spectrum into a Pitch Class Profile. alternative rock. Genetic algorithms have (sring and fret number) to play most notes. Tablature notation also been explored as a means of eficiently exploring the large (or simply tab, plural tabs), which explicitly notates the position search space created in the ingering decision problem [17]. in which to play each note in a piece, was developed to Hori et al.Â On the basis of these observations, genetic algorithm for the generation of jazz solos. GenJam takes a chord sequence as input and outputs an improvised our model has three hidden states at each time point t: solo, with melody phenotype itness optimized over several Xt E {no phrase, phrase, phrase end} (note that phrase starts can be uniquely deined as following immediately rom a 'no iterations via user feedback.