# GENERATION OF CONTROL SIGNALS USING PITCH AND ONSET DETECTION FOR AN UNPROCESSED GUITAR SIGNAL

*Kapil Krishnamurthy*

Center for Computer Research in Music and Acoustics
Stanford University
kapilkm@ccrma.stanford.edu

## ABSTRACT

This paper aims at extracting the pitch information of the notes being played on an electric guitar in real time. An efficient onset detection algorithm is employed to detect the transients in the signal which signifies that a note has been played. Once a transient has been detected, it sets off a pitch detection routine which identifies the frequency of the note. The onset and pitch information is used to generate control signals in the form of MIDI messages which are fed back to the host Digital Audio Workstation. This information can be used to trigger other software instruments such as sampled violins or soft synths for instance. The main idea here is to enable a performer to be accompanied in a live performance by other instruments without actually needing physical players to play on these instruments.

In this paper, a new onset detection scheme called the *Peak*$^2$ detection scheme has been proposed in place of existing onset detection methods, as it has been found to work more efficiently for guitar signals.

## 1. INTRODUCTION

Pitch tracking is a popular topic in the computer music world and there are a number of pre-existing methods that are currently used in various algorithms. Some of the well known pitch tracking methods include the Average Magnitude Difference Function (AMDF) [9], Harmonic Product Spectrum (HPS) [3], Cepstral Pitch Determination [1], Super Resolution Pitch Detection [6], Mcleod Pitch Method (MPM) [7] and YIN [2].

This paper aims at taking some key points from some of these papers and employing a scheme to extract pitch information from the notes being played on an electric guitar in real time. The first step after taking a windowed chunk of the guitar signal involves conditioning the signal to get rid of the unnecessary low and high frequencies. Since, it is known that the frequency range of the guitar is between around 70 Hz to 700 Hz, frequencies above and below those limits can be attenuated. The high pass filter cutoff is set at about 65 Hz to get rid of 60 Hz electrical hum and the low pass filter cutoff is set to about 1000 Hz. The next step involves having an effective yet computationally efficient onset detection algorithm. The *Peak*$^2$ onset detection algorithm was used in this case and it has been found to be excellent for tracking guitar signals. When an onset is detected, it signifies that a new note has been played and that the frequency of this note must be computed. The autocorrelation method is used to find the frequencies of these notes.

The real time implementation of the method described above was done in C++ and compiled into a VST plugin. The VST format is a popular format for digital audio plugins and was developed by Steinberg.

## 2. ONSET DETECTION

The two methods used for onset detection that have been discussed are the *Peak*$^2$ method and Spectral Flux method. Initially, the Spectral Flux method was chosen as the onset detection scheme however now the *Peak*$^2$ method is being used in place of it.

### 2.1. The *Peak*$^2$ Method

The *Peak*$^2$ Method of onset detection basically works on the concept of tracking the peaks in the input signal and looking for a change in the slope of the curve of the peak detector at the onset. In this method, there are two peak detectors that are functioning at all times. The first peak detector, tracks the peaks of the absolute value of the input signal. The peak detector has an attack time and release time which are tuned based on the type of input signal. The attack and release times determine the speed at which the peak detector state exponentially rises or falls with respect to the amplitude of the input signal. For this particular application, the attack time was kept very short and the release time was set fairly long. This permits a quick jump in state of the peak detector to match the input signal peak with a slow exponential decay past that.

From a close-up view of the peak detector function, it is seen that the edges are still jagged and need to be smoothed further in order to be useful for onset detection. In order to achieve this, a second peak detector is used with the state of the first peak detector as the input signal. Once again the

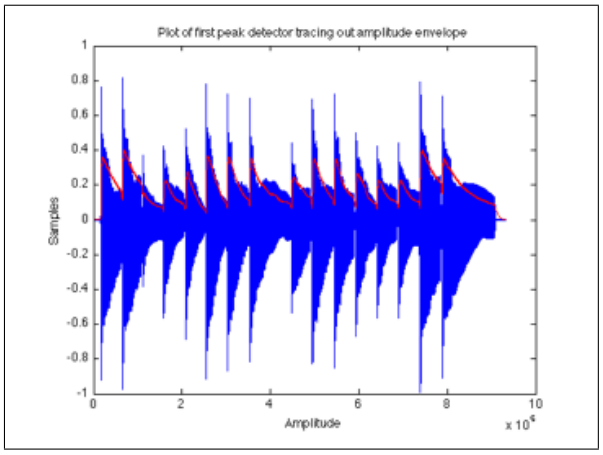


**Figure 1**. Plot of the first peak detector tracing out the amplitude envelope

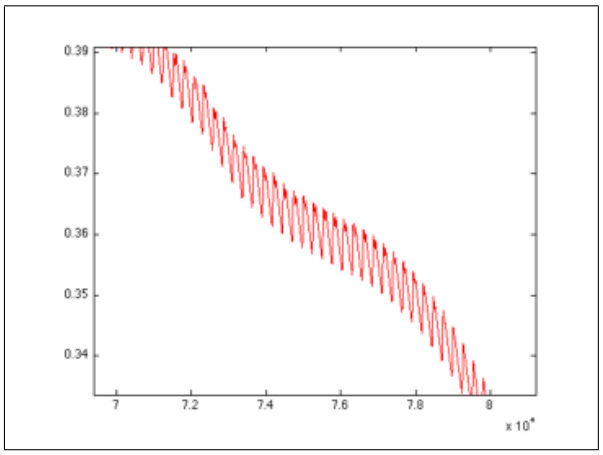


**Figure 2**. Closeup plot of the jagged curve structure of the first peak detector

settings are kept identical with a fast attack and slow release characteristic. The state of the second peak detector is a smoothed version of the first peak detector.

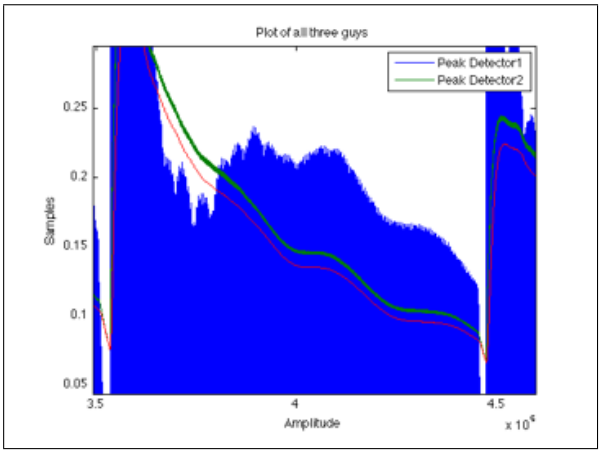


**Figure 3**. Plot of the states of both peak detectors

Now, for each input buffer of audio, if there are $N$ samples, then for each corresponding buffer, there are $N$ samples of the second peak detector. These $N$ samples are summed up and averaged in order to give just one value for each buffer. By monitoring the decibel level of these averaged values, it is seen that at the onsets, there is a change in slope which indicates that an onset has occurred.

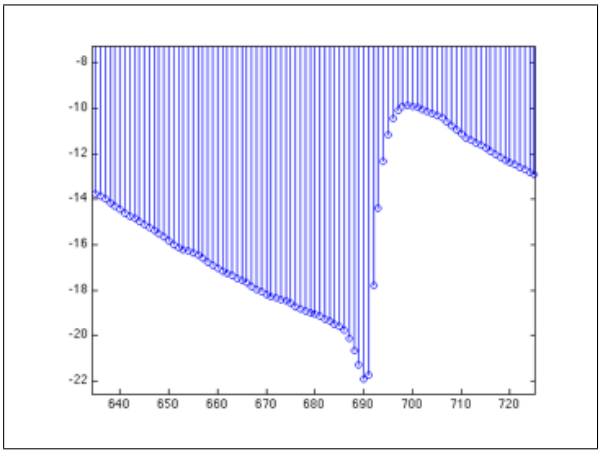

**Figure 4**. The averaged peak detector values over each buffer plotted in dB shows a sudden change in slope at the onset

This method was applied to a number of guitar input signals and found to be quite robust. The other advantages is that apart from the initial tuning of the peak detectors, no threshold needs to be set depending on the level of the input signal.

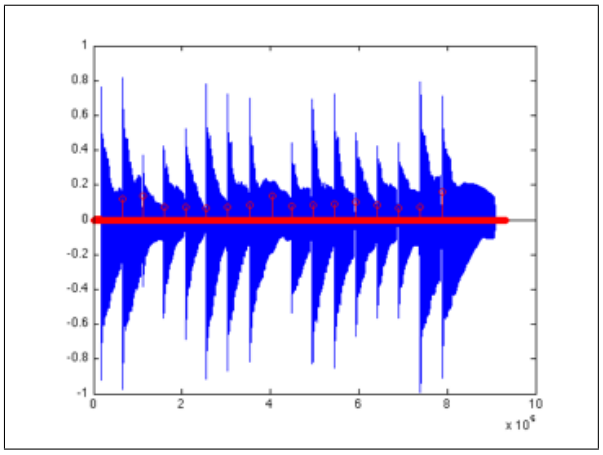

**Figure 5**. Plot of the detected onsets using the $Peak^2$ method on a clean electric guitar signal

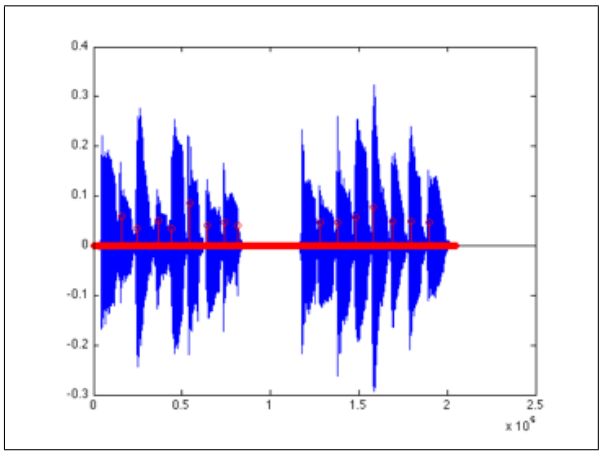

**Figure 6**. Plot of the detected onsets using the $Peak^2$ method on a noisy acoustic guitar signal

### 2.2. Spectral Flux

The Spectral Flux [4][5] method of onset detection measures the change in magnitude of the power of the entire spectrum across consequent spectrums. If there is a transient or a sudden attack, the change in energy will be denoted by a jump in the difference of energy between consequent spectrums.

$$SF(n) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} H(|X(n,k)| - |X(n-1,k)|) \qquad (1)$$

It is important to note that after taking the difference in the spectrums, a positive difference value indicates a rise in energy while a negative difference value indicates a dip in energy. If this method is employed to detect transients, a threshold value should be set only for a positive difference value. The disadvantage of using this method is having to set threshold values which may not be very accurate if the input signal level changes. The other constraint is that this method is computationally quite expensive as it requires having to take FFT's for each buffer of audio.

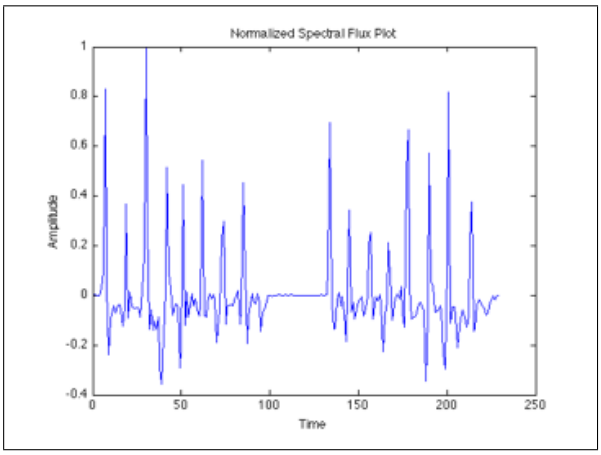


**Figure 7**. Plot of the spectral flux output when applied on a guitar signal

## 3. PITCH DETECTION

For the pitch detection, the autocorrelation method was found to be sufficiently accurate. Although it is rather expensive computationally, it is only called upon when an onset is detected and thus is not always running for each buffer of audio.

### 3.1. Autocorrelation

Autocorrelation [8] is a time domain technique that is used to identify the period of a signal. The autocorrelation of a

signal is the cross-correlation of the signal with itself. From a signal processing aspect, this is useful tool to identify the period of a complex signal that is buried under noise or is a sum of various sinusoids of different frequencies. From a mathematical context, the autocorrelation of a signal can be expessed as

$$(x \star y)_n = \sum_{n=0}^{N-1} \overline{x(m)} x(m+n) \quad (2)$$

From a signal processing context, the convolution of a signal by itself in the time domain is equivalent to squaring its magnitude in the frequency domain. This indicates that the autocorrelation operation keeps only the magnitude of the signal but throws away the phase. This is handy in applications where only the energy or power of the spectrum is of interest. The period of the occurance of the peaks of the autocorrelation gives the period (or frequency) of the signal. In terms of computation, convolution in the time domain is quite an expensive process. However, it is a lot cheaper to do the transform to the frequency domain and then do a pointwise multiplication of the spectrums of the respective signals. One important consideration while doing the autocorrelation is that the window length should take several periods of the input signal into consideration.

In order to increase the accuracy of the estimated frequency from looking at the autocorrelation peaks, instead of having to take longer FFT's, the highest three points of a peak can be interpolated to give the actual peak value. This is extremely useful in cases where the length of the FFT's are short, say of the order of 512 samples or less.

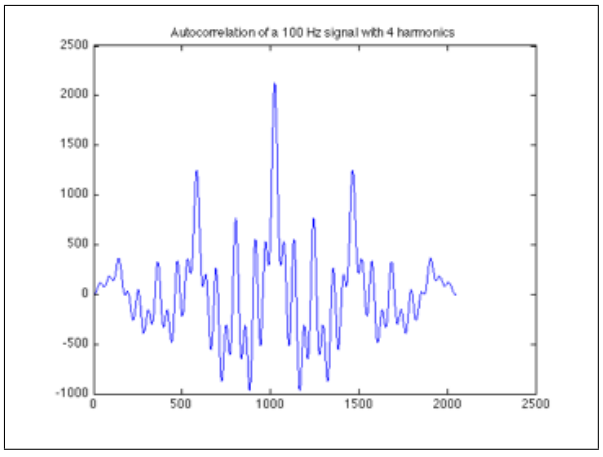


**Figure 8**. Plot of the autocorrelation of a 100Hz signal with several harmonics

## 4. REAL TIME IMPLEMENTATION

The real time implementation of this project was done using C++ as the programming language. The code was compiled into a VST plugin which can be opened by any Digital Audio Workstation(DAW) that supports VST.

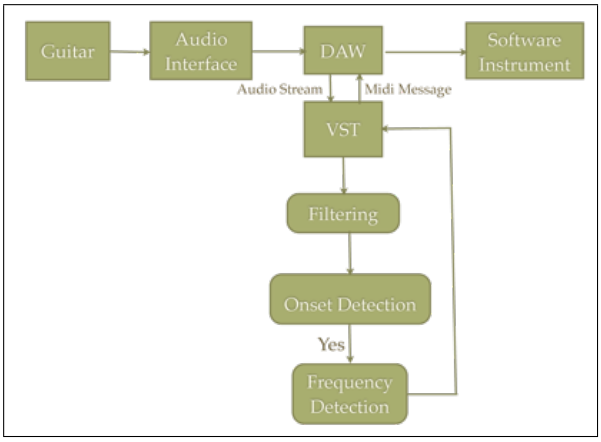


**Figure 9**. Block Diagram Overview of the real-time implementation scheme

The DAW contains the options like setting the sample rate, timing information and other such parameters for the project. This is common to the VST plugin too which receives that information from the DAW during its initialization. Based on the sampling rate and buffer size, the DAW sends the VST plugin a buffer of audio at regular intervals. The VST plugin first conditions this buffer of audio and then runs the *Peak*[2] onset detection algorithm on it. When an onset is detected, the autocorrelation routine is called and the fundamental frequency of the note is estimated.

Now, this information is converted to an equivalent midi message. Midi messages have a few parameters for expression such as note on, note off, note length, velocity and aftertouch. If the algorithms used for detecting the onsets and pitch are accurate and elaborate, it is possible to convert this information to all the equivalent midi parameters. The more parameters there are to work with, the more expressive the sounds produced can get.

## 5. CONCLUSIONS AND FUTURE WORK

This project was aimed at extracting useful information from an incoming guitar signal and using that information to trigger other software instruments. The main portion of the implementation was done in C++ and compiled into a VST plugin. For now, the section of the implementation that involves returning midi messages back to the host is not working. However this will be implemented in the near future along with other modifications. Currently, the pitch and onset detection methods seem to be working well and giving accurate information for their respective purposes.

The future work in store for this project is to first try and be able to estimate polyphonic passages such as chords on the guitar. To accomplish this, more advanced spectral

domain methods will have to be used in order to isolate the pitches. The other major issue to be covered is getting the $Peak^2$ method to work with non-staccato forms of guitar playing, such as legato. Some of the concepts applied here will probably be applicable to other instruments as well such as violins and horns.

## 6. REFERENCES


[1] P. J. Andrew M.S., Degroat R.D., "Robus cepstral based pitch determination," in *Twenty-Third Asilomar Conference on Signals, Systems and Computers*, 1989.

[2] A. de Cheveigné and H. Kawahara, "Yin, a fundamental frequency estimator for speech and music," *The Journal of the Acoustical Society of America*, vol. 111, no. 4, pp. 1917–1930, 2002.

[3] H. Ding, B. Qian, Y. Li, and Z. Tang, "A method combining lpc-based cepstrum and harmonic product spectrum for pitch detection," in *IIH-MSP '06: Proceedings of the 2006 International Conference on Intelligent Information Hiding and Multimedia*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 537–540.

[4] S. Dixon, "Onset detection revisited," in *Proc. of the Int. Conf. on Digital Audio Effects (DAFx-06)*, Montreal, Quebec, Canada, Sept. 18–20, 2006, pp. 133–137, http://www.dafx.ca/proceedings/papers/p_133.pdf.

[5] S. Hainsworth, , S. Macleod, and Malcolm, "Onset detection in musical audio signals," in *In Proc. Int. Computer Music Conference*, 2003.

[6] C.-l. Q. Jian-Qi Yin, Xixian Chen, "Super resolution pitch determination based on cross-correlation and interpolation of speech signals," in *Singapore ICCS/ISITA '92. 'Communications on the Move'*, 1992.

[7] P. McLeod and G. Wyvill, "Visualization of musical pitch," *Computer Graphics International Conference*, vol. 0, p. 300, 2003.

[8] L. Rabiner, "On the use of autocorrelation analysis for pitch detection," *Acoustics, Speech, and Signal Processing [see also IEEE Transactions on Signal Processing], IEEE Transactions on*, vol. 25, no. 1, pp. 24–33, 1977. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1162905

[9] M. J. Ross, H. L. Shaffer, A. Cohen, R. Freudberg, and H. J. Manley, "Average magnitude difference function pitch extractor," *IEEE Trans. Acoust., Speech, Signal Processing*, no. 22, pp. 353–362, 1974.